



16 al 18 de Noviembre
Montevideo, Uruguay



*V Congreso Iberoamericano
de Seguridad Informática*

Actas del Congreso

*Actas del V Congreso Iberoamericano de Seguridad Informática
CIBSI'09*

Montevideo, Uruguay, 16 al 18 de Noviembre de 2009

Editores

Gustavo Betarte

Jorge Ramió Aguirre

Arturo Ribagorda Garnacho

ISBN: 978-9974-0-0593-8

©

Universidad de la República, Uruguay. Facultad de Ingeniería. Instituto de
Computación, 2009

Universidad Politécnica de Madrid, España

Prefacio

Estimados colegas:

Este volumen contiene los trabajos presentados en el V Congreso Iberoamericano de Seguridad Informática (CIBSI'09) realizado en Noviembre en Montevideo, Uruguay.

Esta edición del Congreso Iberoamericano de Seguridad Informática, iniciativa de la Red Temática Iberoamericana de Criptografía y Seguridad de la información CriptoRed, ha convocado al igual que en sus anteriores ediciones a un gran número de investigadores y expertos de Latinoamérica, España y Portugal.

De 65 trabajos recibidos, el Comité de Programa Científico ha seleccionado 41 trabajos, 38 de los cuales se presentan en el evento. Los mismos proceden de investigadores de Argentina, Brasil, Chile, Colombia, España, EE.UU., Francia, México, Portugal, Uruguay y Venezuela.

El congreso cuenta asimismo con tres conferencistas que han sido invitados a presentar su trabajo de investigación en sesiones plenarias, el Dr. Gilles Barthe, de IMDEA Software de España, el Dr. Eduardo Giménez, de la Universidad de la República de Uruguay y el Dr. José Luis Piñar Mañas de la Universidad CEU San Pablo de España. Tendrá también lugar en el congreso un taller, titulado *Los retos de la protección de datos: la Ley 18331 de protección de datos personales*, a cargo del Dr. José Luis Piñar Mañas.

Desde estas páginas queremos hacer llegar nuestro profundo agradecimiento a los organizadores, autores, revisores, patrocinadores y asistentes, que son los que han hecho posible que una vez más tenga lugar este encuentro académico de expertos e investigadores en Seguridad Informática, esta vez en la ciudad de Montevideo.

Noviembre 2009

Gustavo Betarte
Jorge Ramío Aguirre
Arturo Ribagorda Garnacho
CIBSI'09

Organización de la Conferencia

CIBSI'09 es organizado por la Facultad de Ingeniería de la Universidad de la República en conjunto con la Universidad Politécnica de Madrid, a cuyos directivos así como a todos y cada uno de los miembros del comité organizador queremos agradecer desde estas páginas.

Organización General

Gustavo Betarte, Universidad de la República, Uruguay
Jorge Ramíó Aguirre, Universidad Politécnica de Madrid, España

Comité de Programa

Santiago Martín Acurio Del Pino, Pontificia Universidad Católica del Ecuador, Ecuador
Nicolás C.A. Antezana Abarca, Sociedad Peruana de Computación, Perú
Javier Areitio Bertolín, Universidad de Deusto, España
Walter Baluja García, Instituto Superior Politécnico José Antonio Echeverría, Cuba
Tomás Barros, NIC Labs, Chile
Gustavo Betarte (co-chair, Universidad de la República, Uruguay)
Joan Borrel Viader, Universidad Autónoma de Barcelona, España
Pino Caballero Gil, Universidad de La Laguna, España
Jeimy Cano Martínez, Universidad de los Andes, Colombia
Adriano Mauro Cansian, Universidade Estadual Paulista, Brasil
Hugo César Coyote Estrada, Instituto Politécnico Nacional, México
Enrique Daltabuit Godas, Universidad Nacional Autónoma de México, México
Jorge Dávila Muro, Universidad Politécnica de Madrid, España
Ángel Martín del Rey, Universidad de Salamanca, España
Josep Domingo-Ferrer, Universidad Rovira i Virgili, España
Josep Lluís Ferrer-Gomilla, Universidad de Las Islas Baleares, España
Amparo Fúster Sabater, Consejo Superior de Investigaciones Científicas CSIC, España
Luis Javier García Villalba, Universidad Complutense de Madrid, España
Roberto Gómez Cárdenas, ITESM Monterrey, México
Juan Pedro Hecht, Universidad de Buenos Aires, Argentina
Marco Aurelio Henriques, Universidade Estadual de Campinas, Brasil
Emilio Hernández, Universidad Simón Bolívar, Venezuela
Leobardo Hernández Audelo, Universidad Nacional Autónoma de México, México
Luis Hernández Encinas, Consejo Superior de Investigaciones Científicas CSIC, España
Alejandro Hevia, Universidad de Chile, Chile
Juan Guillermo Lalinde, Universidad EAFIT, Colombia
Javier López Muñoz, Universidad de Málaga, España
Julio César López, Universidade Estadual de Campinas, Brasil
Vincenzo Mendillo, Universidad Central de Venezuela, Venezuela
Carlos Mex Perera, ITESM Monterrey, México
Josep María Miret Biosca, Universidad de Lleida, España
Gaspar Modelo Howard, Universidad Tecnológica de Panamá, Panamá

Raúl Monge, Universidad Técnica Federico Santa María, Chile
Edmundo Monteiro, Universidad de Coimbra, Portugal
Guillermo Morales Luna, Centro de Investigación y Estudios Avanzados del IPN, México
Alberto Peinado Domínguez, Universidad de Málaga, España
Yoan Pinzón, Universidad Nacional de Colombia, Colombia
Tamara Rezk, INRIA Sophia Antipolis, Francia
Arturo Ribagorda Garnacho, (co-chair) Universidad Carlos III de Madrid, España
Josep Rifà Coma, Universidad Autónoma de Barcelona, España
Miguel Soriano Ibáñez, Universidad Politécnica de Cataluña, España
Horacio Tapia Recillas, Universidad Autónoma Metropolitana, México
Routo Terada, Universidade de São Paulo, Brasil
Alfredo Viola, Universidad de la República, Uruguay
Horst von Brand, Universidad Técnica Federico Santa María, Chile

Organización Local

Alejandro Blanco, Universidad de la República, Uruguay
Eduardo Carozo, ANTEL, Uruguay
Carlos Luna, Universidad de la República, Uruguay
Marcelo Rodríguez, Universidad de la República, Uruguay
Leonardo Vidal, Universidad de la República, Uruguay
Felipe Zipitría, Universidad de la República, Uruguay

Revisores Externos

Joao Afonso Abrunhosa, Universidade de Lisboa, Portugal
Almudena Alcaide, Universidad Carlos III de Madrid, España
Jorge Blasco Alis, Universidad Carlos III de Madrid, España
Philippe Camacho, Universidad de Chile, Chile
Eduardo Carozo, ANTEL, Uruguay
Eduardo Cota, Universidad de la República, Uruguay
Tiago Cruz, Universidade de Coimbra, Portugal
Eduardo Giménez, Universidad de la República, Uruguay
Jorge Granjal, Universidade de Coimbra, Portugal
Daniel Hedin, Chalmers University of Technology, Suecia
Julio César Hernandez-Castro, University of Southampton, Inglaterra
Jesús Manjón, Universitat Rovira i Virgili, España
Mireya Morales, Universidad Simón Bolívar, Venezuela
Gloria Pujol, Universitat Rovira i Virgili, España
Alejandro Russo, Chalmers University of Technology, Suecia
Miguel Torrealba, Universidad Simón Bolívar, Venezuela
Rolando Trujillo, Universitat Rovira i Virgili, España
Alexandre Viejo, Universitat Rovira i Virgili, España
Arnau Vives, Universitat Rovira i Virgili, España

Tabla de Contenidos

iPhone 3G: Un nuevo reto para la informática forense	1
<i>Andrea Ariza, Juan Ruíz, Jeimy Cano</i>	
Uso del DNle para reforzar el anonimato en el voto telemático mediante tarjetas inteligentes	16
<i>Emilia Pérez Belleboni, Justo Carracedo Gallardo</i>	
Command dimension reduction in masquerader detection	31
<i>Carlos Benitez, Pablo Fierens</i>	
A Survey on Masquerader Detection Approaches	46
<i>Maximiliano Bertacchini, Pablo Fierens</i>	
Propostas para apoiar a preservação documental de longo prazo na ICP-Brasil . .	61
<i>Viviane Bertol, Rafael Timóteo de Sousa Júnior, Ricardo Custodio</i>	
Generación de ambientes para entrenamiento en seguridad informática	73
<i>Alejandro Blanco, Juan Diego Campo, Lucía Escanellas, Carlos Pintado, Marcelo Rodríguez</i>	
Robust Declassification for Bytecode	88
<i>Eduardo Bonelli, Francisco Bavera</i>	
Técnicas anti-forenses en informática: ingeniería reversa aplicada a TimeStomp .	103
<i>Armando Botero, Ivan Camero, Jeimy Cano</i>	
Aplicar el modelo de amenazas para incluir la seguridad en el modelado de sistemas	118
<i>Marta Castellaro, Susana Romaniz, Juan Carlos Ramos, Carlos Feck, Ivana Gaspoz</i>	
Sobre el número de funciones bent obtenidas a partir de funciones de máximo peso	133
<i>Joan-Josep Climent, Francisco J. García, Verónica Requena</i>	
Um IDS Cooperativo para Redes de Acesso de Banda Larga	148
<i>Tiago Cruz, Thiago Leite, Patricio Baptista, Rui Vilão, Paulo Simões, Fernando Bastos, Edmundo Monteiro</i>	
An Extended Reference Monitor for Security and Safety	163
<i>Eduardo B. Fernandez, Michael VanHilst, David laRed Martinez, Sergio Mujica</i>	
Detección y limitaciones de ataques clásicos con Honeynets virtuales	173
<i>Hugo Fernández, Jorge Sznek, Eduardo Grosclaude</i>	
Un modelo compacto de criptografía asimétrica empleando anillos no conmutativos	188
<i>Juan Pedro Hecht</i>	
Watermarking in the encrypted Domain	202
<i>Jordi Herrera-Joancomartí, David Megías</i>	

VIII

La protección de datos y el diseño de tratamientos de datos personales. Especificaciones funcionales necesarias	213
<i>Angel Igualada Menor</i>	
Una marca de agua inteligente aplicada al dinero electrónico	225
<i>Patricia Jaimes, Gabriel Hermosillo, Gomez Roberto</i>	
Aumento de la fiabilidad de la evidencia en un protocolo de intercambio justo mediante la división del entorno de firma	240
<i>Jorge L. Hernandez-Ardieta, Ana I. Gonzalez-Tablas, Benjamin Ramos Álvarez, Arturo Ribagorda Garnacho</i>	
Análisis formal del estándar NIST para modelos RBAC	255
<i>Carlos Luna, Cristian Rosa</i>	
FACTOIDS: Modelos y Herramientas para el Análisis e Intercambio Seguro de Datos Colectados por Sensores	270
<i>Carlos Martinez-Cagnazzo</i>	
Autorización de Acceso en MIDP 3.0	283
<i>Gustavo Mazeikis, Carlos Luna</i>	
Diseño básico de la seguridad para un servicio sacional de salud pública en Venezuela	298
<i>Mireya Morales, Emilio Hernández</i>	
Estegoanálisis aplicado a la generación automática de estegotextos en lengua española	310
<i>Alfonso Muñoz Muñoz, Justo Carracedo Gallardo</i>	
Metodología de implantación de un SGSI en grupos empresariales de relación jerárquica	325
<i>Gustavo Pallas, María Eugenia Corti</i>	
Verificación formal de la equidad de un protocolo de firma de contratos mediante Colored Petri Nets	340
<i>M. Magdalena Payeras-Capellà, Macia Mut-Puigserver, Andreu Pere Isern- Deyà, Josep L. Ferrer-Gomila, Llorenç Huguet-Rotger</i>	
Criptoolanálisis del generador Auto-Shrinking: Una propuesta práctica	355
<i>María Eugenia Pazo Robles, Amparo Fuster Sabater</i>	
Diseño e implementación de una función hash basada en caos	368
<i>César José Ramírez López, Leobardo Hernández Audelo</i>	
Gestión Automatizada de Requisitos de seguridad para proyectos de desarrollo de líneas de producto Software	383
<i>Jesus Rodriguez, Daniel Mellado, Eduardo Fernandez-Medina, Mario Piattini</i>	
SLSB: improving the steganographic algorithm LSB	398
<i>Juan José Roque, Jesús María Minguet</i>	

Hacia una arquitectura de servicios de seguridad para entornos Grid móviles	409
<i>David G. Rosado, Eduardo Fernandez-Medina, Javier Lopez</i>	
Gestión de identidad en las administraciones públicas: Interoperabilidad pan-Europea	423
<i>Sergio Sánchez, Ana Gómez</i>	
MGSM-PYME: Metodología para la gestión de la seguridad y su madurez en las PYMES	437
<i>Luis Enrique Sánchez Crespo, Daniel Villafranca Alberca, Eduardo Fernández-Medina Patón, Mario Gerardo Piattini Velthuis</i>	
Metodología para la selección de métricas en la construcción de un cuadro de mando integral	452
<i>Daniel Villafranca Alberca, Luis Enrique Sánchez Crespo, Eduardo Fernández-Medina Patón, Mario Piattini</i>	
Towards Secure Distributed Computations	467
<i>Felipe Zipitría</i>	
Protocolo de creación de evidencias en entornos vehiculares	482
<i>José María de Fuentes, Ana Isabel González-Tablas, Arturo Ribagorda, Benjamín Ramos</i>	
Control de calidad en imágenes de iris mediante razonamiento ontológico	497
<i>Alberto de Santos Sierra, Javier Guerra Casanova, Carmen Sánchez Ávila, Vicente Jara Vera</i>	
Una propuesta de arquitectura biométrica de control de acceso basada en ontologías	509
<i>Alberto de Santos Sierra, Javier Guerra Casanova, Carmen Sánchez Ávila, Vicente Jara Vera</i>	
Índice de Autores	520

iPhone 3G: Un Nuevo Reto para la Informática Forense

Andrea Ariza, Juan Ruíz y Jeimy Cano

Departamento de Ingeniería de Sistemas, Pontificia Universidad Javeriana,
Carrera 7 No. 40 – 62, Bogotá, Colombia
{a-ariza, jc.ruiz, j.cano}@javeriana.edu.co

Resumen. El presente artículo reúne y expone el estado actual de la informática forense en el dispositivo móvil iPhone 3G. Se presenta un panorama general de la seguridad en dispositivos móviles. Describe el iPhone 3G, su funcionamiento, características principales y las vulnerabilidades identificadas hasta la fecha, a las cuales puede estar expuesto. Presenta los modelos generales de un análisis forense, su aplicación sobre los dispositivos móviles, especificando herramientas y procedimientos utilizados en la actualidad para llevar a cabo dicho proceso sobre el iPhone 3G. Finalmente detalla por qué es importante contar con un procedimiento estándar para realizar un análisis forense sobre éste dispositivo.

Palabras Clave: Informática forense, seguridad informática, procedimientos, estándares, teléfonos celulares, iPhone 3G, metodología, herramientas.

1 Introducción

Según [1], en la actualidad y desde hace aproximadamente diez años, el empleo de dispositivos móviles se ha incrementado notablemente principalmente por su facilidad de uso y la propiedad de mantener en contacto permanente a sus usuarios. A partir de esto, se ha generado un cambio significativo en la forma en que las personas se comunican, pero también por su proliferación, se ha incrementado su uso en actividades de orden delictivo.

Existe gran variedad de gamas de dispositivos móviles, dentro de los cuales el mayor crecimiento en popularidad y uso se presenta en los dispositivos móviles inteligentes, debido a su capacidad tanto para realizar llamadas como para navegar por Internet, y además porque permiten desarrollar y ejecutar aplicaciones que no necesariamente son incluidas por el fabricante [2]. Canalys, empresa que realiza análisis expertos de la industria de alta tecnología, realiza anualmente una investigación acerca del estado del mercado de los dispositivos móviles inteligentes. En los dos últimos años (2007-2008) se ha presentado el mayor crecimiento en el uso de estos dispositivos comparándolo con años anteriores. De los resultados obtenidos cabe rescatar que Apple¹ se introdujo en el mercado en tercer lugar con el dispositivo

¹ Formalmente Apple Computer Inc., empresa norteamericana que se caracteriza por producir computadores, portátiles, impresoras, cámaras y sistemas operativos con tecnología innovadora por más de 30 años [9].

móvil iPhone, esto gracias a la innovación en cuanto a diseño e interfaz de usuario que soporta [3]. Además, Canalys estimó que Apple en el 2007 tomó el 28% del mercado de dispositivos móviles de EE.UU [3] y posteriormente en el 2008, con la llegada del iPhone 3G al mercado, Apple se posicionó en el segundo lugar de ventas de dispositivos móviles inteligentes [4], convirtiéndose rápidamente en líder en el mercado de dispositivos móviles.

Al alcanzar tanta popularidad en el mercado mundial de dispositivos móviles y por la variedad de funcionalidades que ofrecen los teléfonos inteligentes, también se han incrementado notablemente los ataques a las vulnerabilidades que presentan ellos. Los ataques fuertes son más fáciles de realizar y son más lucrativos para quienes los desarrollan y llevan a cabo no solo a nivel de dispositivos móviles sino a nivel de cualquier otro sistema informático dentro de una organización [7].

Anualmente el Instituto de Seguridad Informática CSI (Computer Security Institute) [6] publica un reporte llamado “CSI Computer Crime and Security Survey” [5] en el cual se realiza un análisis de la situación actual de la seguridad informática y del crimen informático de las organizaciones participantes, y cuyo objetivo principal es conocer si las mejores prácticas de seguridad informática implantadas producen resultados. Dicha encuesta arrojó que por la necesidad de movilidad y disponibilidad de la información en las organizaciones, el uso de dispositivos móviles inteligentes dentro de estas se ha incrementado notablemente (una de las razones por las cuales su popularidad ha aumentado), al igual que los incidentes de inseguridad (robo de información de propietarios y pérdida de datos de clientes) que ocurren sobre ellos [8]. Cuando un incidente se presenta, es más fácil y debe ser más importante para lograr la identificación de los perpetradores, monitorear los terminales de comunicación que la comunicación en sí misma. Por ejemplo si un criminal utiliza un iPhone para llevar a cabo algún tipo de ataque su operación estará comprometida en cierto nivel [5], pues el hecho que haya utilizado este dispositivo para cometer el ataque implicará que se siga un procedimiento estándar establecido para la realizar la investigación.

En la actualidad existen gran cantidad de proveedores y fabricantes de dispositivos móviles inteligentes, lo que produce heterogeneidad en todo sentido en el campo de la informática forense, especialmente en las herramientas que se utilizan para obtener evidencia de estos dispositivos en una investigación, de tal manera que los fabricantes crean sus propios protocolos para este propósito [2].

A diferencia de la informática forense clásica, el análisis forense sobre dispositivos móviles inteligentes y específicamente sobre iPhone, que es en esencia un computador [5], es un campo relativamente nuevo y los procedimientos y normas para su análisis aún se encuentran en desarrollo [9]. Un análisis forense que se lleve a cabo sobre un dispositivo como éste en una corte, puede ser admitido o no dependiendo de lo que considere el juez que lleve dicho caso y la formalidad con que se desarrolle el procedimiento de recolección, control, análisis y presentación de las evidencias.

2 ¿Qué es el iPhone?

El iPhone es un dispositivo móvil creado por Apple, caracterizado principalmente por combinar tres productos en uno: un teléfono revolucionario, un iPod² de pantalla ancha y un innovador dispositivo de internet que permite navegar en la web, recibir correos electrónicos con HTML enriquecido y navegación web completa [18] [27].

2.1 Conociendo el iPhone

Desde hace mucho tiempo, diferentes dispositivos móviles, incluyendo teléfonos celulares, han podido hacer lo que el iPhone es capaz de hacer. Según [9] lo que diferencia al iPhone de los demás, y ha logrado hacerlo tan popular en el mercado de la tecnología móvil, es su diseño de pantalla táctil con un teclado virtual, que permite que el tamaño de la pantalla sea superior a cualquier otro dispositivo de forma tal que se puedan visualizar películas, fotos y navegar por la web de manera sencilla y cómoda [24] [17].

Actualmente, existen dos generaciones de iPhone. De la primera generación se lanzaron modelos con capacidad de almacenamiento de 4 GB y 8 GB, el modelo de 4 GB fue descontinuado del mercado dos meses después de su lanzamiento, por lo cual se lanzó el modelo de 16 GB de capacidad de almacenamiento. De la segunda generación se lanzaron modelos con las mismas capacidades (8GB y 16 GB), la diferencia radica en que el iPhone de segunda generación utiliza tecnología 3G (tercera generación de telefonía móvil), y añade más funcionalidades [9] [16] [29].

2.2 Estructura lógica y física del iPhone

Como ya se ha mencionado anteriormente el iPhone es un teléfono inteligente que integra funcionalidades de iPod y teléfono celular [26]. Es en esencia un computador ejecutando una versión del sistema operativo Leopard UNIX OS de Apple, diseñado principalmente para minimizar las escrituras sobre la memoria flash, de forma tal que se puede conservar y preservar datos por periodos largos de tiempo, incluso por mucho más tiempo que lo que un computador de escritorio podría hacerlo [5] [19].

Aunque los primeros modelos del iPhone tuvieron variaciones, como cualquier dispositivo electrónico complejo, el iPhone es una colección de módulos, chips y otros componentes electrónicos de diferentes fabricantes [21] [23], que ejecuta una versión móvil de MAC OS X 10.5 (Leopard) que es similar a la versión del sistema operativo MAC para computadores de escritorio [5]. Los componentes que se muestran en la siguiente tabla pertenecen al modelo del iPhone 3G.

Tabla 1. Hardware iPhone 3G. Tomado de [23], traducción libre de los autores

Función	Fabricante	Modelo
Unidad central de	Samsung	S5L8900B01 – 412 MHz ARM1176Z(F)-

² Reproductor de música creado por Apple Inc., caracterizado por ser una biblioteca multimedia digital [28].

procesamiento (CPU)		S
		RISC, 128 Mbytes of stacked, package-on-package, DDR SDRAM
Aceleración Gráfica 3D	Tecnologías Imagination	Power VR MBX Lite
Amplificador de potencia UMTS	TriQuint	TQM676031 – Band 1 – HSUPA TQM666032 – Band 2 – HSUPA TQM616035 – Band 5/6 WCDMA/HSUPA PA-duplexer
Transceptor UMTS	Infineon	PMB 6272 GSM/EDGE and WCDMA PMB 5701
Procesador Base	Infineon	X-Gold 608 (PMB 8878)
Memoria base de apoyo	Numonyx	PF38F3050M0Y0CE - 16 Mbytes of NOR flash y 8 Mbytes of pseudo-SRAM
Amplificador de cuatro bandas GSM / EDGE	Skyworks	SKY77340 (824- to 915-MHz)
Antena GPS, Wi-fi y BT	NXP	OM3805, a variant of PCF50635/33
Gestor de comunicaciones	Infineon	SMARTi Power 3i (SMP3i)
Gestor a nivel de sistema	NXP	PCF50633
Cargador de batería / controlador USB	Tecnología Linear	LTC4088-2
GPS	Infineon	PMB2525 Hammerhead II
Flash NAND	Toshiba	TH58G6D1DTG80 (8 GB NAND Flash)
Chip flash serial	SST	SST25VF080B (1 MB)
Acelerómetro	ST	LIS331 DL
	Microelectronics	
Wi-Fi	Marvell	88W8686
Bluetooth	CSR	BlueCore6-ROM
Codec de audio	Wolfson	WM6180C
Controlador de la pantalla táctil	Broadcom	BCM5974
Interfaz de enlace con la pantalla	National Semiconductor	LM2512AA Mobile Pixel Link
Controlador de línea de pantalla táctil	Texas Instruments	CD3239

Dos de los componentes más importantes del iPhone son:

- *Unidad central de procesamiento (CPU)*: es una unidad de procesamiento RISC³ (Conjunto de Instrucciones Reducidas) que ejecuta el núcleo de los procesos del iPhone y trabaja conjuntamente con el coprocesador PowerVR para la aceleración de gráficos. La CPU ejecuta a una tasa de reloj más baja que la especificada por el fabricante, se ejecuta a 412 MHz de 667 MHz posibles, esto para extender la vida útil de la batería [23].
- *Procesador Base*: es el componente del iPhone que gestiona todas las funciones que requieren la antena (servicios celulares). El procesador base

³ Es un tipo de microprocesador con las siguientes características fundamentales: Instrucciones de tamaño fijo y presentadas en un reducido número de formatos. Sólo las instrucciones de carga y almacenamiento acceden a la memoria por datos.

tiene su propia memoria RAM⁴ y firmware en flash NOR⁵, como recurso de la CPU principal. El Wi-Fi y el Bluetooth son gestionados por la CPU principal a pesar que el procesador base almacena su dirección MAC en su NVRAM⁶ [23].

El esquema de partición del disco de un iPhone se asemeja a un Apple TV⁷. Por defecto el iPhone está configurado con dos particiones de disco que no residen en una unidad de disco física, debido a que utiliza una NAND flash⁸ de estado sólido que es tratada como un disco de almacenamiento. Allí se almacenan una tabla de particiones y un formato de sistema de archivos [5] [9]. La primera partición se conoce como Master Boot Record⁹ (MRB), responsable de cargar el sistema operativo. Esta partición es de solo lectura para conservar el estado de fábrica durante todo el ciclo de vida del iPhone, y es donde se encuentran todas las aplicaciones que vienen por defecto en el iPhone. A continuación existe un área libre conocida como Apple_Free area, seguida de la segunda partición que se divide en dos: una parte HFSX¹⁰ que en un principio almacena el sistema operativo, otra área libre Apple_Free area y la segunda parte HFSX que contiene todos los datos de usuario, donde se almacena música, videos, fotos, información de contactos, entre otros [5] [9].

Los orígenes del iPhone provienen del iPod y del Apple TV, que cuentan con dos particiones: una únicamente para operación y otra únicamente para almacenamiento. Aunque el esquema de partición es diferente, los sistemas de archivos son similares en su estructura [9]. El tamaño de la primera partición esta generalmente entre los 300MB y los 500MB, y usa el sistema de archivos HFSX. Esta partición es la encargada de cargar el sistema operativo del iPhone, el iPhone OS, el cual es una variante del núcleo del sistema operativo OS X de Apple. Basado en el mismo núcleo MACH y compartiendo algunos elementos básicos con el sistema operativo OS X 10.5 (Leopard), el iPhone se compone de 4 capas, incluyendo el sistema operativo básico, el núcleo API de servicios, los medios de comunicación y una capa de cacao resistente [23].

El tamaño de la segunda partición, en un iPhone de 16GB es de 14.1GB, en uno de 8GB es de 7.07GB. Dicha partición, contiene el sistema de archivos HFSX y un volumen llamado "Data", donde la mayoría de la información y de datos de valor se aloja y puede ser localizada [9].

⁴ Memoria de acceso aleatorio desde donde el procesador recibe las instrucciones y guarda los resultados.

⁵ Memoria que permite que múltiples posiciones de memoria sean escritas o borradas en una misma operación de programación mediante impulsos eléctricos.

⁶ La memoria de acceso aleatorio no volátiles un tipo de memoria que no pierde la información almacenada al cortar la alimentación eléctrica.

⁷ El Apple TV permite escuchar los temas de iTunes, visualizar fotos en HD y visualizar podcasts gratuitos. <http://www.apple.com/es/appletv/whatis.html> [22]

⁸ Memorias que usan un túnel de inyección para la escritura y para el borrado un túnel de 'soltado'. Permiten acceso secuencial (más orientado a dispositivos de almacenamiento masivo).

⁹ Sector de almacenamiento de datos que contiene código de arranque de un sistema operativo almacenado en otros sectores del disco.

¹⁰ Es un sistema de archivos desarrollado por Apple Inc [25].

3 Problemas de seguridad en el iPhone 3G

Como se mencionó en la sección anterior, el iPhone 3G es un dispositivo móvil celular que presenta múltiples funcionalidades, por tal razón, está sujeto a sufrir ataques que cualquier otro tipo de teléfono celular puede recibir. Pero, el limitado conocimiento sobre el aseguramiento del dispositivo es el que abre la posibilidad a vulnerabilidades conocidas o desconocidas, las cuales pueden ser aprovechadas por los intrusos. En la actualidad no existe una taxonomía detallada que especifique que tipo de ataques puede sufrir un iPhone 3G, sin embargo, se han identificado algunos ataques, los cuales se describirán a continuación.

Según [14], *“este tipo de dispositivos móviles no están expuestos solamente a ataques de “código malicioso”; existen otros tipos de amenazas como son los ataques directos, la interceptación de la comunicación y los ataques de autenticación”*. Los ataques de código malicioso o malware son algún tipo de software que se ejecuta en el dispositivo sin que el usuario se de cuenta, algunos ejemplos son los troyanos y gusanos. Los ataques directos, los cuales no son efectivos para todos los dispositivos móviles, consisten en que el perpetrador es capaz de localizar el dispositivo para vulnerarlo [20]. Generalmente este tipo de ataques se realiza utilizando Bluetooth y una serie de herramientas, como GhettoTooth, BTScanner y BlueScan, que identifican el dispositivo, ejecutan el ataque mediante una serie de técnicas como BlueJacking, BlueSpam, BlueSnarfing, BlueBug, BlueSmac y BackDoor, y por último ejecuta algún comando para obtener datos, cargar datos o cambiar la configuración del dispositivo. La interceptación de la comunicación consiste en que el perpetrador se conecta a los dispositivos o redes para obtener información. Por último, están los ataques de autenticación como el spoofing (suplantación de identidad) y sniffing (captura de tramas de la red) [11]. *“Blue MAC Spoofing es el nombre de uno de los ataques que realizan suplantación de identidad en Bluetooth. Este escenario combina varias de las técnicas nombradas anteriormente en los ataques directos debido a que involucra varias fases tales como emparejamiento, descubrimiento de dispositivos, suplantación de identidad (BlueSmac) y transferencia de archivos sin confirmación”*. [14]

Además de los ataques comunes para dispositivos móviles, el iPhone 3G esta sujeto a ser víctima de otro tipo de ataques por contar con un sistema operativo similar al de un computador [41]. Actualmente se han identificado otros tipos de ataque sobre este dispositivo. Uno es conocido como ‘Jailbreak’, es decir, el firmware original del dispositivo es reemplazado con un firmware alternativo o “hackeado” que libera el dispositivo evitando las restricciones puestas por el fabricante. Este firmware contiene paquetes de instalación que permiten instalar herramientas para manejar el dispositivo de acuerdo a las necesidades [9]. Para ello existen herramientas que realizan el ‘Jailbreak’ como Pwnage Tool [30] y Quickpwn [10].

“Un experto en seguridad ha descubierto que la técnica denominada “URL Spoofing” puede ser utilizada en las aplicaciones de correo y navegación web del iPhone, lo que puede provocar ataques de phishing en este dispositivo.” [31]. Esto es que un atacante puede simular que una página web maliciosa tenga el aspecto y funcionalidades de una web válida, y así aprovecharse instalando algún tipo de malware.

Considerando lo anterior y entendiendo la estructura lógica del iPhone los rastros de un ataque se pueden obtener en la segunda partición. Según [5] otro tipo de datos que se almacenan en esta partición además de la música, videos, fotos y contactos, y que pueden llegar a ser evidencia en una investigación dada son:

- Caches de teclado, nombres de usuario, contraseñas, términos de búsqueda y fragmentos de documentación tecleada.
- Pantallazos del último estado de una aplicación son preservados cuando el botón Home es presionado o cuando se cierra alguna aplicación.
- Fotos eliminadas de la librería y de la memoria.
- Direcciones, contactos, eventos de calendario y otros datos personales.
- Historial de llamadas, además de las que se despliegan, las ultimas 100 llamadas y entradas eliminadas.
- Imágenes de mapas de Google Maps y búsquedas por longitud/latitud de coordenadas.
- Cache del buscador y objetos eliminados del buscador que identifican que sitios web fueron visitados.
- Correos electrónicos eliminados, mensajes de texto, marcas de tiempo y banderas de comunicación (con quien y en que dirección la comunicación tuvo lugar).
- Grabaciones de correo de voz eliminadas.
- Información entre el dispositivo y el computador relacionado.

4 Informática forense en Teléfonos Inteligentes

La informática forense aplicada a dispositivos móviles y especialmente a teléfonos inteligentes, es un área de investigación relativamente nueva. Es por esto que existe poca literatura sobre el tema, y más aún en lo relacionado con la atención de incidentes [14]. A continuación explicaremos que es la informática forense y la importancia de su aplicación en dispositivos móviles como el iPhone 3G.

4.1 Informática forense clásica

El constante aprovechamiento de fallas sobre los sistemas de computación por parte de agentes mal intencionados, ofrece un escenario perfecto para que se cultiven tendencias relacionadas con intrusos informáticos [32], estos agentes malintencionados varían según sus estrategias y motivaciones, que abarcan desde lucro monetario hasta satisfacción personal, y en algunos casos es un estilo de vida [33].

Con respecto a lo citado anteriormente, según [32], la criminalística ofrece un espacio de análisis y estudio sobre los hechos y las evidencias que se identifican en el lugar donde se llevaron a cabo las acciones criminales, por lo tanto, es necesario establecer un conjunto de herramientas, estrategias y acciones que ayuden a identificar estos hechos y evidencias dentro del contexto informático [42].

La informática forense es una rama de las ciencias forenses, que involucra la aplicación de la metodología y la ciencia para identificar, preservar, recuperar, extraer, documentar e interpretar [5] pruebas o evidencias procedentes de fuentes digitales con el fin de facilitar la reconstrucción de los hechos encontrados en la escena del crimen [36], para luego usar dichas evidencias como elemento material probatorio en un proceso judicial [34] [14].

Con respecto a lo anterior según [32] podemos considerar la evidencia como “cualquier información, que sujeta a la intervención humana u otra semejante, ha sido extraída de un medio informático”. La evidencia digital tiene como características principales el hecho de ser volátil, anónima, duplicable, alterable y eliminable, en consecuencia, a diferencia de la evidencia física en un crimen clásico, la evidencia digital es un desafío para aquellas personas que la identifican y analizan debido a que se encuentran en un ambiente cambiante y dinámico [43] [46].

Por consiguiente, se hace necesaria la aplicación de procedimientos estrictos y cuidadosos, desde el momento en que se realiza la recolección de la evidencia, hasta que se obtienen los resultados posteriores a la investigación [14] [38], por tal razón a continuación se muestra un modelo generalizado para realizar este tipo de investigaciones.

1. *Identificación de la escena del crimen:* en esta fase se hace el reconocimiento del incidente con el objetivo de identificar el tipo de crimen [36]. Es primordial tener precauciones para evitar la contaminación de la escena de esta fase en adelante [39].
2. *Preparación:* en esta fase se realiza la preparación de herramientas, técnicas, órdenes de registro y autorizaciones para acceder a la escena del crimen [36].
3. *Preservación de la escena del crimen:* en esta fase se realiza el aislamiento de la evidencia física y digital del mundo externo, es decir, de personas no autorizadas y dispositivos electromagnéticos [36].
4. *Recolección de la evidencia:* en esta fase se realiza la recolección de toda la evidencia encontrada en la escena del crimen, empezando por la más volátil hasta la menos volátil, es importante que al momento de realizar la inspección de (los) equipo (s) evitar alterar cualquier variable ya que un cambio a la vista insignificante podría invalidar todo el proceso de investigación en un proceso judicial. Por otro la recolección debe ser realizada mediante herramientas especializadas y certificadas con el objetivo de evitar modificaciones en las fechas de acceso y en la información del registro del sistema [35] [38] [40].
5. *Preservación de la evidencia:* es importante que durante todo el proceso de investigación forense la evidencia conserve sus propiedades con las que fue recolectada para evitar que pierda su valor de carácter legal, por lo tanto, es primordial realizar toda la investigación sobre copias exactas de la evidencia obtenida en la escena del crimen, comprobando periódicamente la integridad de dicha copia [35]. Por otro lado es necesario actualizar la cadena de custodia de la evidencia cada vez que esta cambie de responsable, este documento debe contar con la información de donde, cuando y quien manejo la evidencia, quien la custodia y en donde esta almacenada [37] [38] [40].
6. *Análisis de la evidencia:* el objetivo de esta fase es lograr identificar como fue efectuado el ataque, cual fue la vulnerabilidad explotada y en lo posible identificar al atacante [40], para lograr lo anterior es necesario reconstruir la

secuencia temporal del ataque, para lo cual se debe recolectar la información de los archivos asociados, marcas de tiempo, permisos de acceso y estado de los archivos.

7. *Presentación del informe forense*: Finalmente culmina la investigación y en este paso se presentan los resultados por parte del investigador sobre su búsqueda y análisis de los medios, lo que se encontró en la fase de análisis de la evidencia, así como información puntual de los hechos y posibles responsables [44]. Debido al rigor que requiere una investigación de este tipo, cada movimiento por parte del investigador o su equipo de trabajo se debe documentar hasta que se resuelva o se dé por concluido el caso. Esta documentación se debe llevar a cabo por medio de formularios que hacen parte del proceso estándar de investigación, entre los cuales se encuentran el documento de custodia de la evidencia nombrado en el numeral 5 de esta sección, el formulario de identificación de equipos y componentes, el formulario de incidencias tipificadas, el formulario de recogida de evidencias y el formulario de medios de almacenamiento [14].

Luego de realizar las fases del modelo anterior según [5] y [35], para que la evidencia digital pueda ser usada en procesos judiciales debe cumplir con las siguientes características:

- *Admisibilidad*: toda evidencia recolectada debe ajustarse a ciertas normas jurídicas para presentarlas ante un tribunal.
- *Autenticidad*: la evidencia debe ser relevante al caso, y el investigador forense debe estar en capacidad de representar el origen de la misma.
- *Compleitud*: la evidencia debe contar todo en la escena del crimen y no una perspectiva en particular.
- *Fiabilidad*: las técnicas usadas para obtener la evidencia deben gozar de credibilidad y aceptadas en el campo en cuestión, evitando dudas sobre la autenticidad y veracidad de las evidencias.
- *Entendimiento y Credibilidad*: se debe explicar con claridad y pleno consentimiento, que proceso se siguió en la investigación y como la integridad de la evidencia fue preservada, para que esta sea comprensible y creíble en el tribunal.

4.2 Informática forense en iPhone 3G

El objetivo principal de la informática forense aplicada a dispositivos móviles, al igual que el objetivo de la informática forense clásica como se mencionó anteriormente, es la búsqueda y recolección de información relacionada con un incidente en el cual se pueda encontrar posible evidencia digital incriminatoria, y esta sea utilizada como elemento material probatorio en un proceso judicial [14]. Actualmente, el 80% de los casos en procesos judiciales contienen algún tipo de evidencia digital [2], razón por la cual el proceso de recolección y el valor de la evidencia resulta ser de vital importancia.

Sin embargo, y debido a que el mercado de la telefonía móvil inteligente se ha incrementado primordialmente por la variedad de fabricantes y modelos de teléfonos que existen, la heterogeneidad que se presenta tanto en su configuración de hardware,

su sistema operativo y tipo de aplicaciones que manejan, como en las herramientas adoptadas para recuperar contenidos que se puedan utilizar en una investigación forense, es bastante significativa [12] [47]. En la mayoría de los casos, los fabricantes de dispositivos móviles optan por crear y aplicar sus propios protocolos para uso de sus sistemas operativos y cables, ocasionando que para los analistas forenses sea más difícil realizar una investigación por la variedad de herramientas que existen para cada tipo de dispositivo [2] [14].

Es por esto que para que se logre un análisis forense digital confiable, y la evidencia que se recoja logre ser admisible, auténtica, completa, fiable, entendible y creíble, es importante conservar los lineamientos presentados en la sección 4.1, añadiendo parámetros concernientes a la manipulación de teléfonos móviles celulares, y los puntos que pueden variar dependiendo del fabricante y modelo [13].

Hoy en día, como ya se ha mencionado, existe poca literatura sobre el análisis forense en dispositivos móviles y específicamente sobre el iPhone. Sin embargo, se han desarrollado herramientas y algunas técnicas forenses no formales para llevar a cabo dicho proceso.

Como en cualquier investigación forense, existen variedad de enfoques que se pueden utilizar para la recolección y análisis de información [13]. Un aspecto clave para ello, por no decir el más importante, es que el procedimiento que se siga, no modifique la fuente de información de ninguna manera, o que de ser esto absolutamente necesario, el analista esté en la capacidad de justificar por qué realizó esta acción [23]. Según [23], existen tres técnicas diferentes para la recolección de evidencia sobre un iPhone:

1. *Obtener datos directamente del iPhone*: este enfoque es preferible a la recuperación de archivos desde el computador con el que el iPhone se sincronizó. Sin embargo, el analista forense debe entender cómo ocurrió la adquisición de información, si el iPhone fue modificado en algún aspecto y qué tipo de información no se logró adquirir.
2. *Obtener una copia de respaldo o una copia lógica del sistema de archivos del iPhone utilizando el protocolo de Apple*: esta aproximación consiste en la lectura de archivos del iPhone, mediante el protocolo de sincronización de Apple, el cual solo es capaz de obtener archivos explícitamente sincronizados a través de él. Piezas claves de información son almacenadas en bases de datos de tipo SQLite, las cuales son soportadas por el protocolo. Hacer consultas sobre estas bases de datos generará la recuperación de información como mensajes de texto y correos electrónicos eliminados del dispositivo.
3. *Una copia física bit a bit*: este proceso crea una copia física bit a bit del sistema de archivos del iPhone, de manera similar al procedimiento que se sigue sobre computadores personales. Este proceso, de todos los anteriores, es el que más potencial tiene para recuperar información, incluyendo los datos eliminados, pero resulta ser complicado pues requiere que se modifique la partición del sistema del iPhone.

Según [5], el proceso técnico que se debe seguir para realizar un análisis forense sobre un iPhone, debe seguir fuertemente los lineamientos nombrados en la sección 4.1, y consta de cuatro pasos que se describen a continuación:

1. *Manipulación física*: Es el paso más importante antes de examinar el dispositivo, ya que se debe contar con el equipo adecuado para mantener el

dispositivo cargado y conectado. Se debe retirar la tarjeta SIM o bloquear cualquier tipo de comunicación o alteración que pueda sufrir el dispositivo mediante una jaula de Faraday que bloquee los campos eléctricos alrededor de él, incluyendo las transmisiones celulares [12].

2. *Establecer comunicación*: este paso consiste en organizar las conexiones físicas y de red apropiadas para instalar una herramienta forense y realizar la recuperación de información.
3. *Recuperación forense*: extraer la información original para crear una copia y trabajar sobre ella. Esto requiere revisiones de integridad para evitar alteraciones de datos.
4. *Descubrimiento electrónico*: es el proceso en el cual toda la información extraída es procesada y analizada. Durante esta etapa, los archivos eliminados son recuperados y el sistema de archivos es analizado, para la recolección de evidencia.

Para llevar a cabo el proceso anteriormente descrito, se han desarrollado de igual manera herramientas que permiten la extracción de información del dispositivo, y se ha hecho un enfoque del equipo necesario que se requiere para ello [15]:

- Un computador ejecutando el sistema operativo Mac OS X Leopard ó Windows XP. Algunas de las herramientas que existen hasta el momento funcionan sobre Mac OS Tiger y Windows Vista pero no han sido probadas en su totalidad sobre estos sistemas, por lo cual no se garantiza que la extracción de datos sea completa.
- Un cable de tipo USB para instalar las herramientas de recuperación, y mantener el dispositivo cargado.
- Una implementación de SSH en el computador incluyendo ssh y scp, para establecer comunicación con el dispositivo.
- Una instalación en el computador de iTunes 7.6 o 7.7 dependiendo del firmware del iPhone en cuestión. De igual manera versiones superiores deberían funcionar correctamente.
- Espacio adecuado en el computador, para almacenar las copias del sistema de archivos del iPhone. Se recomienda reservar un espacio de mínimo tres veces la capacidad del iPhone en cuestión.

Una vez se cree un ambiente más confiable que limite la contaminación cruzada sobre las copias del sistema de archivos obtenidas, se debe contar con una herramienta para su análisis. Existen variedad de herramientas para este propósito cada una con características y propiedades diferentes. A continuación describiremos algunas de las que existen en la actualidad según [23] [45].

1. *WOLF de Sixth Legion*: Es una herramienta forense diseñada específicamente para el iPhone. Soporta todos los modelos, 2G y 3G, que ejecutan cualquier versión de firmware desde la 1.0 hasta la 2.2. Este software solo se puede ejecutar sobre el sistema operativo Mac OS X, y es capaz de eludir el código de seguridad tanto del dispositivo como el de la tarjeta SIM, si se tiene acceso al computador en el que fue usado el iPhone, sin necesidad de realizar el “Jailbreak” (descrito en la sección 3). WOLF no modifica el iPhone para realizar la adquisición de información ya que usa una copia de la lógica de datos, sin embargo una desventaja es que no recuperar contenido suprimido del dispositivo.

2. *Cellebrite UFED*: El sistema forense Cellebrite UFED es un dispositivo autónomo (stand-alone), capaz de adquirir datos desde dispositivos móviles y almacenar la información en una unidad USB, tarjeta SD¹¹ o PC. UFED incorpora un lector y generador de copias de tarjetas SIM. La posibilidad de clonar una tarjeta SIM es una potente característica que puede crear e insertar un clon de la tarjeta SIM original y el teléfono funcionará con normalidad. Sin embargo, no se registrará con el operador de telefonía móvil de la red, eliminando la necesidad de bolsas de Faraday y la posibilidad de que los datos del teléfono sean actualizados o eliminados. El paquete viene con alrededor de 70 cables para conectar a la mayoría de los dispositivos móviles disponibles en la actualidad. Incluyen protocolos de conexión serial, USB, infrarrojos y Bluetooth. Permite extraer información de contactos, mensajes de texto, historial de llamadas, mensajes de texto eliminados, grabaciones, video, fotos y detalles del teléfono entre otros.
3. *MacLockPick II (MLP)*: Esta herramienta tiene un enfoque único para la adquisición forense. El objetivo de MLP es proporcionar una solución de plataforma cruzada forense que realiza una adquisición en vivo de una máquina sospechosa. La información se almacena en un dispositivo USB y el software se proporciona para analizar los resultados. MLP no funciona directamente en el iPhone, en su lugar se centra en el directorio de copia de seguridad MDBACKUP donde se almacenan la gran mayoría de los archivos. Entre los datos que son recuperables se encuentran el historial de llamadas, mensajes de texto, contactos, notas, fotos, cuentas de correo sincronizadas, entradas rápidas de marcación, estado, historial y bookmarks del navegador y detalles del teléfono, entre otros.

5 Conclusiones

Se ha evidenciado que si bien el investigador forense tiende a seguir una metodología para realizar un análisis, frecuentemente la aparición de nuevas tecnologías y la heterogeneidad que estas presentan, no permiten que el seguimiento sea estricto y que varíen los procedimientos que se utilizan. Por otra parte, la variedad de herramientas de análisis de datos que existen y sus diferentes niveles de confiabilidad, presentan características particulares que facilitan o dificultan algunas actividades necesarias en el proceso de análisis de datos.

Este documento establece un conjunto de elementos conceptuales y aplicados sobre el dispositivo móvil iPhone 3G, perteneciente a una de las áreas que requiere hoy en día más investigación y profundización [14], debido a que los procedimientos y las normas para su análisis aún se encuentran en desarrollo, y al crecimiento tecnológico y popularidad alcanzada. En la segunda fase de la investigación se propondrá una guía metodológica que defina el proceso especializado en la obtención de detalles de incidentes ocurridos en un iPhone 3G, además de probar escenarios reales en incidentes de seguridad informática sobre el dispositivo.

¹¹ Es un formato de tarjeta de memoria flash utilizado en dispositivos portátiles.

6 Referencias

1. Mellar, B.; Forensic Examination of Mobile Phones. Digital Investigation – Elsevier. United Kingdom. (2004), <http://faculty.colostate-pueblo.edu/dawn.spencer/Cis462/Homework/Ch4/Forensic%20examination%20of%20mobile%20phones.pdf>
2. Rossi, M.: Internal Forensic Acquisition for Mobile Equipments. Dipartimento di Informatica, Sistemi e Produzione, Università di Roma “Tor Vergata”. (2008).
3. Canalys, Expert Analysis for High-tech Industry.: Smart Mobile Device Shipments hit 118 Million in 2007, up 53% on 2006. Singapore and Reading (UK) (2008). <http://www.canalys.com/pr/2008/r2008021.htm>.
4. Canalys, Expert Analysis for High-tech Industry.: Global Smart Phone Shipments Rise 28%”. Reading (UK) (2008). <http://www.canalys.com/pr/2008/r2008112.htm>.
5. Zdziarski, J.: iPhone Forensics, Recovering Evidence, Personal Data & Corporate Assets. O’Reilly Media, Inc. (2008)
6. CSI, Computer Security Institute, <http://www.gocsi.com/>
7. CSI.: Computer Crime and Security Survey: The latest results from the longest-running project of its kind. (2008). <http://i.cmpnet.com/v2.gocsi.com/pdf/CSISurvey2008.pdf>
8. Castillo, C., Romero, A.: Guía Metodológica Para el Análisis Forense Orientado a Incidentes en Dispositivos Móviles GSM. Pontificia Universidad Javeriana. (2008).
9. Varsalone, J., Kubasiak, R.: Mac Os X, iPod and iPhone Forensic Analysis DVD Toolkit. Syngress Publishing, Inc, pp. 355-475. (2009)
10. Jailbreak iPhone and iPod Touch Games and More.: www.quickpwn.com
11. Bluetooth SIG.: Funcionamiento de la Tecnología Bluetooth. <http://spanish.bluetooth.com/Bluetooth/Technology/Works/Default.htm>
12. Willassen, S.: Forensic analysis of mobile phone internal memory. Norwegian University of Science and Technology. http://digitalcorpora.org/corpora/bibliography_files/Mobile%20Memory%20Forensics.pdf
13. Agualimpia, C., Hernández, R.: Análisis Forense en Dispositivos Móviles con Symbian OS. Documento de maestría, Dept. Ingeniería electrónica, Pontifica Universidad Javeriana. http://www.criptored.upm.es/guienteoria/gt_ml42e1.htm.
14. Castillo, C., Romero, A., Cano, J.: Análisis Forense Orientado a Incidentes en Teléfonos Celulares GSM: Una Guía Metodológica. Conf. XXXIV Conferencia Latinoamericana de Informática, Centro Latinoamericano de Estudios en Informática (CLEI). (2008). <http://www.clei2008.org.ar/>
15. Baggilo, I., Milan, R., Rogers, M.: Mobile Phone Forensics Tool Testing: A Database Driven Approach”. International Journal of Digital Evidence. Purdue University, Volume 6 Issue 2. (2007). <http://www.utica.edu/academic/institutes/ecii/publications/articles/1C33DF76-D8D3-EFF5-47AE3681FD948D68.pdf>
16. Vogelstein, F.: The Untold Story: How the iPhone Blew Up the Wireless Industry. WIRED, Wired Magazine: Issue 16.02. (2008). http://www.wired.com/gadgets/wireless/magazine/16-02/ff_iphone?currentPage=all
17. Penalva, J.: Historia del iPhone. Xataka, (2008). <http://www.xataka.com/moviles/historia-del-iphone>
18. Product Reviews.: The Apple iPhone Story: 1999 to 2008. (2008). <http://www.product-reviews.net/2008/08/01/the-apple-iphone-story-1999-to-2008/>
19. Dumas, D.: iPhone Timeline Highlights the Handset Through The Ages. WIRED. (2008). <http://blog.wired.com/gadgets/2008/07/iphone-timeline.html>,
20. Hackers pueden entrar a tu iphone mediante OpenSSH. Crackea tu iPhone. (2008). <http://crackeatuiphone.com/2008/09/19/hackers-pueden-entrar-a-tu-iphone-mediante-openssh/#more-1523>

21. Apple Inc.: Especificaciones iPhone 3G. (2009). <http://www.apple.com/es/iphone/specs.html>
22. Apple Inc.: Qué es iTunes. (2009). <http://www.apple.com/es/itunes/whatis/>
23. Hoog, A.: iPhone Forensics: Annual Report on iPhone Forensic Industry. Chicago Electronic Discovery. (2009).
24. Apple Inc.: iPhone 3G: Gallery. (2009). <http://www.apple.com/iphone/gallery/>
25. Macedonia, M.: iPhones Target the Tech Elite. Entertainment Computing, pp. 94-95. (2007).
26. Grossman, L.: Invention Of the Year: The iPhone. TIME in partnership with CNN. http://www.time.com/time/specials/2007/article/0,28804,1677329_1678542,00.html
27. Apple Inc.: iPhone 3G: Características. (2009). <http://www.apple.com/la/iphone/features/>
28. Apple Inc.: iPod Classic. (2009). <http://www.apple.com/la/ipodclassic/features.html>
29. Apple Inc.: Apple introduce el Nuevo iPhone 3G: dos veces más rápido y a mitad de precio. Sala de prensa. (2009). <http://latam.apple.com/pr/articulo/?id=1486>
30. Spiner, T.: Crackers Claim iPhone 3G Hack, A group of developers has claimed to have cracked the iPhone 3G. ZDNet. (2008). <http://news.zdnet.co.uk/security/0,1000000189,39446756,00.htm>
31. Pastor, J.: El iPhone Vulnerable al Phishing. The Inquirer. http://www.theinquirer.es/2008/07/25/el_iphone_vulnerable_al_phishing.html
32. Cano, J.: Introducción a la informática forense: Una disciplina técnico-legal. Revista Sistemas, Asociación Colombiana de Ingenieros de Sistemas (ACIS). Vol.96, pp. 64-73. (2006). http://www.acis.org.co/fileadmin/Revista_96/dos.pdf
33. Cano, J.: Computación Forense: Conceptos Básicos.
34. Del Pino, S.: Introducción a la informática forense. Pontificia Universidad Católica del Ecuador. (2007). http://www.criptored.upm.es/guiateoria/gt_m592b.htm
35. Brezinski, D., Killalea, T.: Guidelines for Evidence Collection and Archiving. IETF RFC 3227. (2002). <http://www.ietf.org/rfc/rfc3227.txt>
36. Reith, M., Clint, C., Gunsch G.: An Examination of Digital Forensic Models. International Journal of Digital Evidence, Air Force Institute of Technology, Volume 1 Issue 3. (2002). www.utica.edu/academic/institutes/ecii/publications/articles/A04A40DC-A6F6-F2C1-98F94F16AF57232D.pdf
37. Baryamureeba, V., Tushabe, F.: The Enhanced Digital Investigation Process Model. Institute of Computer Science, Makerere University. (2004).
38. Meyers, M., Rogers, M.: Computer Forensics: The Need for Standardization and Certification. International Journal of Digital Evidence, CERIAS, Purdue University, Volume 3 Issue 2. (2004). www.utica.edu/academic/institutes/ecii/publications/articles/A0B7F51C-D8F9-A0D0-7F387126198F12F6.pdf
39. International Organization on Computer Evidence.: Guidelines for best practice in the forensic examination of digital technology. IOCE Best Practice Guide V1.0. (2002). http://www.ioce.org/fileadmin/user_upload/2002/ioce_bp_exam_digit_tech.html
40. Delgado, M.: Análisis Forense Digital. Hackers y Seguridad, 2nd ed. (2007) http://www.criptored.upm.es/guiateoria/gt_m335a.htm
41. Sher, M., Magedanz, T.: 3G-WLAN Convergence: Vulnerability, Attacks Possibilities and Security Model. (2007).
42. Leigland, R.: A Formalization of Digital Forensics. International Journal of Digital Evidence. University of Idaho. Volume 3, Issue 2. (2004). <http://www.utica.edu/academic/institutes/ecii/publications/articles/A0B8472C-D1D2-8F98-8F7597844CF74DF8.pdf>
43. DFRWS.: A Road Map for Digital Forensic Research. Report From the First Digital Forensic Research Workshop (DFRWS). (2001). <http://www.dfrws.org/2001/dfrws-rm-final.pdf>

44. Bem, D., Huebner, E.: Computer Forensic Analysis in a Virtual Environment. International Journal of Digital Evidence. Volume 6, Issue 2. (2007).
<http://www.utica.edu/academic/institutes/ecii/publications/articles/1C349F35-C73B-DB8A-926F9F46623A1842.pdf>
45. Geiger, M.: Evaluating Commercial Counter-Forensic Tools. Carnegie Mellon University. Digital Forensic Research Workshop (DFRWS). (2005).
http://dfrrws.org/2005/proceedings/geiger_couterforensics.pdf
46. IOCE: Guidelines For Best Practice in the Forensic Examination of Digital Technology.
http://www.ioce.org/fileadmin/user_upload/2002/ioce_bp_exam_digit_tech.html
47. Adelstein, F.: MFP: The Mobile Forensic Platform. International Journal of Digital Evidence. Volume 2, Issue 1. (2003).
<http://www.utica.edu/academic/institutes/ecii/publications/articles/A066554D-DCDD-75EE-BE7275064C961B5D.pdf>

Uso del DNLe para reforzar el anonimato en el voto telemático mediante tarjetas inteligentes

Emilia P. Belleboni¹ y Justo Carracedo Gallardo²

Departamento de Ingeniería y Arquitecturas Telemáticas. Escuela Universitaria de Ingeniería Técnica de Telecomunicación. Universidad Politécnica de Madrid
Ctra. Valencia km. 7. 28031 Madrid.

Teléfono: (+34) 91 336 78 02. Fax: (+34) 91 336 78 17,
{belleboni¹, carracedo²}@diatel.upm.es

Resumen. Debe llegar el momento en cual el documento de identidad electrónico, DNLe, que en España expide la Dirección General de Policía, sea una herramienta que goce de la confianza ciudadana a la hora de operar como tal documento electrónico. Este artículo avanza en las ventajas que conlleva introducir la utilización de este DNLe en un sistema de voto telemático (Votescrypt+) basado tanto en criptografía avanzada como en la fortaleza que proporcionan las tarjetas inteligentes. Se incorpora la utilización del DNLe como elemento determinante para la identificación del votante, tanto dentro del proceso de obtención de la autorización para votar como dentro del proceso de verificación individual de resultados, y se describe resumidamente el protocolo de funcionamiento del sistema en sus distintas fases.

Palabras clave: DNI electrónico, voto electrónico, voto telemático.

1. Introducción y antecedentes

El voto es solo un mecanismo para posibilitar la democracia. Por ello, la automatización y modernización de los sistemas de votación solo se justifica si se adapta a las necesidades de los electores y si contribuye a mejorar la forma en la que la ciudadanía manifiesta su opinión y traslada sus decisiones.

1.1 Voto electrónico, voto telemático y voto por Internet

Actualmente, en distintos países (por ejemplo, Brasil (1) y Venezuela(2) se han implantado, con bastante éxito, sistemas de voto electrónico consistentes en la de automatización de todo el proceso de emisión y recuento del voto. Para ello se dispone de urnas electrónicas ubicadas a la vista del votante en las que éste deposita el voto mediante un procedimiento electrónico (en sustitución de las antiguas urnas donde se introducen los votos en papel). En algunos casos se dispone, además, de redes de comunicación para transmitir los resultados del recuento realizado en las urnas electrónicas.

En el caso del *voto telemático* la votación se lleva a cabo mediante el uso de redes telemáticas y agentes telemáticos específicos de tal forma que los votos se depositan

en una *urna remota* fuera de la vista del votante. Tanto la *autorización* para votar como el *voto* “viajan” por la red.

En líneas generales, se pueden distinguir dos escenarios para voto telemático: a) el votante puede votar desde cualquier sitio, también desde casa; y b) para votar es necesario acudir a *puntos específicos de votación*.

Con frecuencia se denomina también *voto electrónico* (*eVoting*) a lo que aquí denominamos *voto telemático*. Nosotros mantenemos una nomenclatura diferenciada debido a que el voto en urna “in situ” y el voto en urna remota son dos alternativas con una dimensión tecnológica y unos requerimientos sociopolíticos radicalmente distintos, y no distinguirlo claramente crea confusión en la ciudadanía. Podría optarse por decir a cada paso “voto electrónico con urna in situ” o “voto electrónico remoto”, pero eso supondría una complicación innecesaria.

Por último, algunas propuestas de voto telemático que permiten votar desde cualquier sitio (también **desde casa**) usando los servicios convencionales de Internet podrían denominarse *voto por Internet*. En los casos en los que, como el que se presenta en este artículo, la votación se realiza utilizando agentes telemáticos propios y una red propia, consideramos más adecuado utilizar la denominación *voto telemático*. (Aunque en la práctica esa “red propia” no sea sino una red virtual “apoyada” en la infraestructura de transporte de datos de Internet).

1.2 Requerimientos para el voto telemático

Al formular una propuesta de *voto telemático* se debe ser consciente de la magnitud de los ataques que busquen alterar el resultado final de la elección, mucho más significativos que cuando se trata de ataques a los sistemas de votación tradicional o en los sistemas de voto electrónico con urna *in situ*. Cualquier ataque puede venir tanto de agentes externos como de agentes internos o de colusión entre ellos, que por motivaciones políticas, económicas o técnicas quieran producir una alteración en los resultados finales. Existe la necesidad de proteger a los votantes de las amenazas y coacciones de los que puede ser víctima si el sistema no toma precauciones específicas ya que la automatización de los procedimientos puede derivar también en la automatización y masificación de la adición, eliminación o alteración de los votos y de la coacción hacia los votantes (3).

Con el fin de contrarrestar estas amenazas, los sistemas de votación telemáticos usan técnicas robustas de criptografía que bien combinadas garantizan la confidencialidad e integridad de la información, así como garantías de que son emitidos por fuentes válidas.

Como paso previo al diseño de los sistemas de votación que se describen en el presente artículo, se llevó a cabo una investigación sociológica mediante la cual se identificó y desglosó una larga lista de requisitos (se discuten en (4) páginas 497 a 501) que podrían resumirse en los siguientes principios básicos:

1. Es imprescindible que se realice una identificación fiable de los votantes, impidiendo la suplantación de identidad, la emisión de múltiples votos y la denegación arbitraria del derecho a voto.
2. Una vez satisfecho el paso anterior, es necesario permitir al votante ejercer su derecho con garantías de que nadie (en solitario o coludiendo con otros) es capaz

de relacionar el voto con el votante que lo ha emitido. Estas garantías deben tener en consideración que no se debe exponer al votante a tener en su mano información que pueda ponerle en riesgo de sufrir una potencial coacción. Aquí adquiere especial importancia el esmero que se pone en la información que el votante usará para verificar el tratamiento seguido por su voto.

3. En ningún momento deber ser posible que se añadan votos ilegítimos, ni que se modifiquen o eliminen votos legítimos.

Una de las dificultades que presenta el diseño adecuado de un sistema de votación telemática es que dentro de los requisitos se presentan exigencias que son, en principio, aparentemente **contradictorias**. Destaquemos las siguientes:

- a) El sistema de votación debe garantizar que solamente las personas con derecho a voto pueden emitir un voto y a la vez debe garantizar que es imposible relacionar a esta persona, fehacientemente identificada, con el voto que ha emitido. Es decir, hay que combinar (aunque en fases distintas) dos servicios de seguridad habitualmente contrapuestos: **autenticación** y **anonimato**.
- b) El sistema de votación telemático debe proporcionar mecanismos para que los votantes **verifiquen** el tratamiento que se le ha dado a su voto, pero ese mecanismo no debe permitir al votante probar ante terceros cuál ha sido el contenido de ese voto (para evitar la coacción y la venta de votos).

1.3 Las propuestas Votescrypt

Desde principios del año 2000 los autores del presente artículo han estado participando el diseño, prueba e implementación de sistemas de votación telemática en el marco del Proyecto Votescrypt (5)¹. En 2003 se llevó a cabo en el municipio de Hoyo de Pinares (provincia de Ávila, España) una experiencia piloto de una implementación simplificada de Votescrypt desarrollada en colaboración con la Fábrica Nacional de Moneda y Timbre (6). Se han desarrollado diversas mejoras y prototipos sobre los que se han ido realizando diversas publicaciones (cabe destacar CIBSI 2003(7); (4) pág. 458 a 480 y 495 a 531]; (8) y (9).

En todas estas versiones se mantienen una serie de características comunes que giran en torno la utilización de una tarjeta inteligente, del tipo *JavaCard*, que previamente a la votación se habrá hecho llegar a todos los votantes, en la que se ejecutarán los algoritmos propios de Votescrypt y se almacenarán de forma segura todos los datos sensibles, como pueden ser las claves usadas para cifrar la información. En el presente artículo, a partir del apartado 3, se describe una nueva versión del sistema, que hemos denominado **Votescrypt+**², que incorpora la

¹ La primera financiación pública se obtuvo a través del proyecto “Votación Electrónica Segura basada en criptografía avanzada” patrocinado por Plan Nacional de I+D+I (TIC 2000-1630).

² Este artículo se enmarca dentro de los trabajos que este grupo de investigación lleva a cabo en el proyecto TSI2006-4864 *Plataforma telemática de Administración Electrónica basada en coreografía de servicios* subvencionado por el Ministerio de Educación y Ciencia, en el Plan nacional de I+D+I

utilización una tarjeta inteligente adicional, el DNIE, para separar de forma más robusta la fase de identificación-autorización de la fase de emisión del voto.

2. Resumen de las características de Votescrypt

Como se acaba de comentar, la propuesta Votescrypt+, que incorpora la utilización de dos tarjetas inteligentes para la votación telemática, una de ellas el DNIE del votante, se realiza mediante modificaciones a Votescrypt. En los párrafos que siguen se resumen las características más relevantes de este sistema.

Es fundamento básico de Votescrypt que toda la información que transita por la red lo haga con garantías de integridad, prueba de origen y confidencial. Toda vez que sea posible, las labores que se realizan en los componentes que podemos denominar oficiales estarán supervisadas por otros equipos informáticos a cargo de interventores o testigos que actúan en representación de los distintos colectivos de ciudadanos que participan en la votación. Además, se almacena la información suficiente para permitir que la auditoría contraste el valor de los registros con los resultados finales publicados. Con todo es posible realizar adecuadamente las verificaciones que soliciten interventores a título colectivo y votantes a título individual.

2.1 Escenario de comunicación

Para dar satisfacción a los requisitos demandados por los ciudadanos, se ha establecido (9) la utilización de tres redes independientes: una para identificar al votante, otra para emitir el voto y la tercera para permitir que el votante pueda realizar la verificación sobre el tratamiento de su voto (figura 1). Todos los recursos utilizados en cada una de estas tres redes deben ser auditables y haber sido auditados antes del inicio el proceso electoral.

El escenario de comunicación de Votescrypt incluye un conjunto de sistemas automáticos que se describen a continuación:

- . Puntos de autenticación (Authentication Points, APs): son equipos informáticos, sin capacidades criptográficas, que permiten al votante realizar el procedimiento de autenticación para obtener la autorización que le permitirá emitir el voto.
- . Un sistema Administrador (Administration System, AS) que puede considerarse como sistema *oficial* el cual emite las autorizaciones solicitadas por los votantes.
- . Varios Sistemas de Intervención (Intervention Systems, Iss) encargados de supervisar y contrastar el funcionamiento del sistema Administrador *oficial*.
- . Cabinas de Votación (Voting Cabin, VoCs), que facilita el aislamiento del votante protegiéndolo de injerencias externas y le permite, una vez que ha sido ya autorizado, emitir el voto de forma segura guardando en su tarjeta de votación el comprobante de voto que, en su caso, le permitirá aportar la prueba para la verificación.
- . Una Urna (Ballot Box, BB) que almacena los votos hasta el momento del recuento y devuelve los comprobantes de votación.

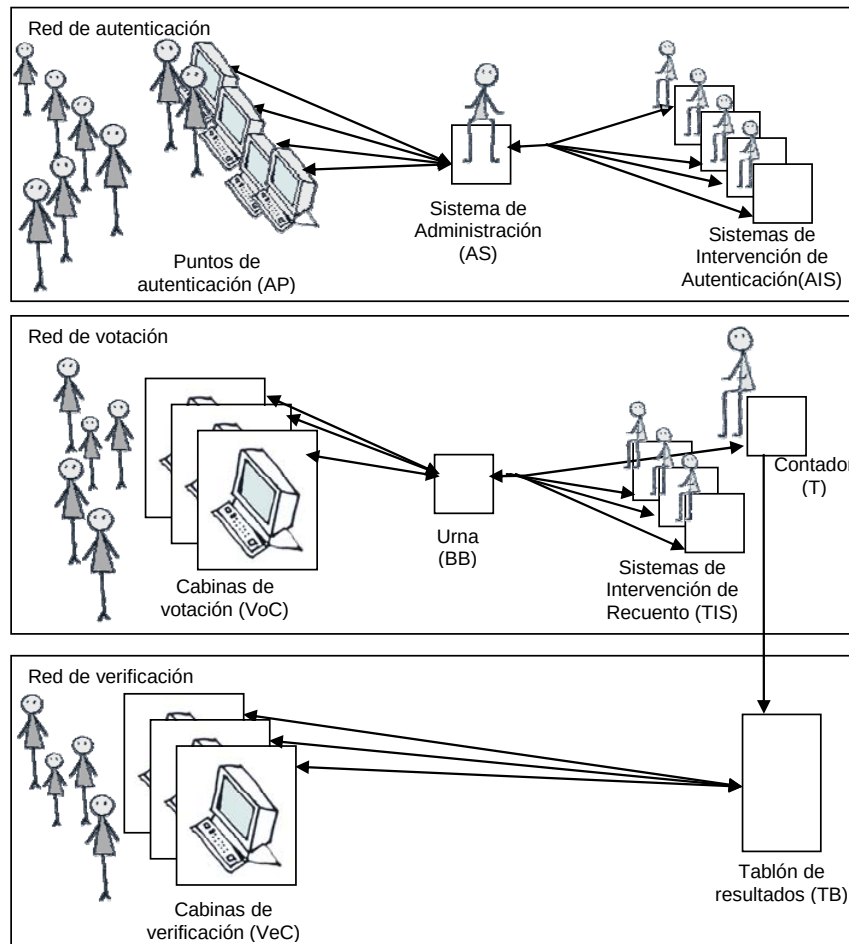


Figure 1: Visión general: 3 redes independientes componen el sistema Votescrypt

- . Un Contador (Tallier, T). que *abre los votos* y realiza el recuento *oficial* teniendo en cuenta toda la información existente en el interior de los denominados Sobres Seguros.
- .Varios Sistemas de Intervención de Recuento (Tally Intervention Systems, TIS) que realizando en paralelo la tarea del Contador impedirán la introducción, modificación o eliminación de votos válidos.
- . Un Tablón de Resultados (Tally Board, TB), donde el Contador hace públicos los resultados, y a partir del cual los Sistemas de Intervención de Recuento comparan los resultados.
- . Cabinas de verificación (Verification Cabin, VeC), que tal como las Cabinas de Votación, aíslan al votante para permitirle acceder a los recursos de la red que se

encargan de verificar que los datos almacenados en la tarjeta de votación y en los registros publicados por el sistema son congruentes.

2.2 Comportamiento global de Votescript

Cada una de las personas implicadas en el proceso de votación posee la tarjeta inteligente que le permite realizar su tarea. Junto con los procedimientos diseñados para permitir la auditoría del software y de los resultados, el otro pilar en el que se sustenta la fortaleza de la propuesta Votescript es la posesión por parte de todos y cada uno de los votantes de su Tarjeta Inteligente de Votación (Smart Voting Card, SVC). Esta tarjeta incluye mecanismos de protección contra intentos de lectura o escritura por parte de dispositivos no autorizados y permite al votante ofrecer prueba de origen, descifrar información confidencial y además ejecutar en su interior otros procesos criptográficos. A partir de ahí, la actividad se realizará de la siguiente forma:

Comienza con las fases 1 (autenticación) y 2 (votación), las cuales son consecutivas desde el punto de vista del votante, quien como primer paso debe presentarse ante la red de autenticación del sistema, y luego dirigirse a la red de votación donde entregará su voto. Desde el punto de vista del sistema estas dos fases son simultáneas durante todo el periodo dedicado a que los votantes acudan a los sitios donde se encuentran los equipos adecuados y emitan cada uno su voto.

En la fase 1, el votante, haciendo uso de su tarjeta inteligente de votante se identifica frente a uno cualquiera de los Puntos de Autenticación del sistema y solicita la autorización para emitir el voto la cual será denegada si el votante ya ha realizado previamente la misma gestión o no pertenece al cuerpo de votantes. Los Sistemas de Intervención de Autenticación y el Sistema de Administración realizan las mismas tareas para decidir emitir o no la autorización. La solicitud de autorización que realiza la tarjeta consiste en una clave (K_{dv} , ocultada por la tarjeta con un factor de opacidad para que posteriormente no se pueda relacionar el voto con el votante) que más tarde servirá para descifrar el voto. Esta *autorización*, clave (K_{dv}), firmada a ciegas por el Sistema de Administración y por los Sistemas de Intervención de Autenticación servirá de garantía de que el voto ha sido emitido por un legítimo votante.

La fase 2 (votación) empieza para el votante después de haber obtenido la autorización. El votante debe acudir con su tarjeta a una cualquiera de las Cabinas de Votación. En el interior de la cabina y aislado de injerencias externas, el votante encuentra los recursos que le permite emitir su voto que irá cifrado con una clave solo conocida por la tarjeta que la genera, (K_{cv}), simétrica de (K_{dv}). El voto está cifrado de forma que la Urna no puede descifrar la pieza de información que recibe. Después se almacena en la tarjeta, de forma protegida, el comprobante de voto que ésta recibe.

La fase 3 (escrutinio) dará comienzo una vez cerradas las dos fases anteriores. En esta fase se procede a la *apertura de la Urna*, la cual trasvasa al Contador (T) y a los Sistemas de Intervención de Recuento los votos custodiados por ella mientras las fases 1 y 2 estaban activas. Cada receptor del contenido de la Urna, actuando en paralelo, realizará el recuento con los mismos datos. El Contador envía los resultados al Tablón de Resultados, para que todos los Interventores puedan corroborarlos.

La fase 4 (verificación) se divide en dos partes. En la primera (fase 4.1) se procede a aclarar todas las posibles divergencias que puedan aparecer entre los resultados

obtenidos en el recuento *oficial* y el que realicen los interventores, pues todos han recibido los mismos datos de la Urna. En la segunda parte (fase 4.2) el votante vuelve a tomar protagonismo ya que es el momento en el que puede acceder a la *red de verificación*, donde podrá revisar el tratamiento que ha tenido su voto, presentando algunos de los datos almacenados en su tarjeta inteligente que se contrastarán con la información hecha pública por el Tablón de Resultados, pudiendo presentar reclamación ante la Autoridad de Elección en caso de desacuerdo, aportando pruebas robustas como el comprobante de voto recibido de la Urna. La Autoridad de Elección accederá a los registros de los distintos agentes, a la información almacenada en la tarjeta inteligente de votante y a la información publicada para dirimir todas las reclamaciones presentadas tanto por votantes como por interventores.

La fase 5 (publicación) es la última, una vez tomadas las decisiones oportunas por parte de la Autoridad de Elección, se procede a la publicación definitiva de resultados y a la destrucción de toda la información generada en el proceso, en atención a la duda razonable por parte de ciudadanos que argumentan que los sistemas que son seguros hoy en día pueden no serlo en un futuro. (3)

3. Dos tarjetas inteligentes para la votación telemática

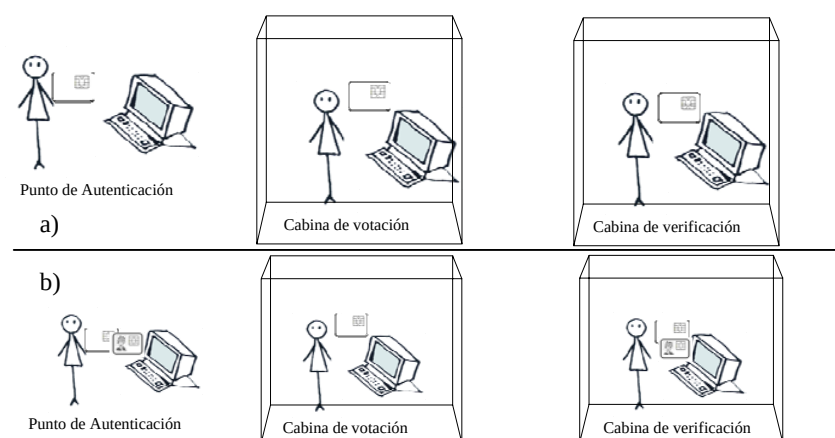


Figure 2: Uso de tarjetas inteligentes en Votescrypt y Votescrypt+

En la figura 2 a) se representa de forma esquematizada la forma en que el votante utiliza la Tarjeta Inteligente de Votación en el esquema Votescrypt; a esta tarjeta, en la explicación que sigue, la denominaremos, para simplificar, tarjeta Votescrypt. En la parte b) de la misma figura se representa el proceso seguido en Votescrypt+, en donde se refleja además la utilización que se hace del DNIE. En este caso a la Tarjeta Inteligente de Votación la denominaremos simplificadaamente SVC+.

Por cuestiones de espacio y de oportunidad no se describe aquí de forma pormenorizada las características y procedimientos previstos para el uso del DNIE

cuya información oficial es pública (10). Para Votescrypt+ el dato de mayor importancia es la presencia en el chip de los certificados electrónicos reconocidos de autenticación y de firma.

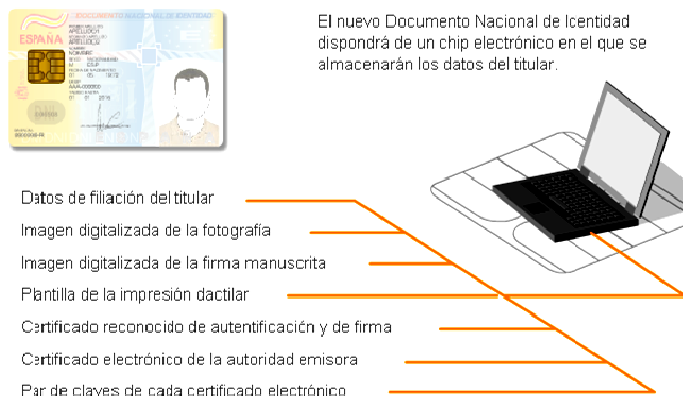


Figure 3: Datos del Chip en el DNIe español

En la figura 3 se recoge el aspecto físico del DNIe y los datos que contiene el chip.

Fase de Autenticación. El proceso diseñado en Votescrypt (a) y el correspondiente a Votescrypt+ (b), se describe brevemente continuación, para cada fase, con el fin de facilitar la comparación entre ambas propuestas.

- a) La tarjeta Votescrypt contiene los datos y las claves que sirven para identificar al votante, en base a las cuales puede demostrar su pertenencia al cuerpo electoral cuando solicita la autorización para votar. Esta tarjeta es también utilizada en la fase de emisión del voto, según se describió en el apartado 2.2.

La tarjeta Votescrypt tiene que ser individualizada para todos y cada uno de los votantes ya que soporta la identidad del votante mediante un par de claves, privada y pública (ésta última es, como su nombre indica, conocida por todos los actores presentes en el protocolo y está asegurada por el correspondiente certificado digital).

- b) Para reforzar aún más la emisión anónima del voto, el esquema de votación telemática **Votescrypt+** que se describe en este artículo incorpora la utilización una tarjeta inteligente adicional, el DNIe, que no se utiliza en la fase de votación. Con ello se consigue separar de forma más robusta la fase de identificación-autorización de la fase de emisión del voto.

La SVC+ **carece de elementos de identificación** del votante, es una tarjeta genérica e igual para todos los votantes, en tanto que la tarjeta Votescrypt tiene que ser personalizada para cada uno de ellos.

El votante, figura 2 (b), haciendo uso de una SVC+ y de su DNIe se identifica frente a uno cualquiera de los Puntos de Autenticación del sistema y solicita la *autorización* para emitir el voto. Para ello, el Punto de Autenticación, que no tiene capacidades criptográficas, dispone de dos lectores de tarjeta para poder insertar ambas simultáneamente.

La validación de la identificación del ciudadano que pretende votar se realiza mediante el DNle y con la colaboración del *Centro de Validación del DNle*³ conforme a los procedimientos y protocolos establecidos por los poderes públicos. Es decir, Votescrypt+ delega en el esquema DNle el aseguramiento de la identidad de los votantes. A medida de que, con el tiempo, este aseguramiento vaya siendo cada vez más robusto y vaya ganando la confianza ciudadana por la incorporación de su uso en la vida cotidiana, esa mejora la heredará Votescrypt merced a la delegación que estamos comentando.

La posesión de un documento de identidad válido no significa necesariamente tener derecho a voto (por motivos de edad, de sentencia judicial, de residencia en región ajena a la votación en curso, etc.). Por tanto, en la *fase de autenticación* la tarjeta Votescrypt+, “alter ego” del votante ante el sistema de votación telemática, deberá seguir ejecutando los protocolos correspondientes hasta obtener la *autorización* para votar que será almacenada en la tarjeta Votescrypt+.

Fase de votación

- a) En Votescrypt tanto la fase de *autenticación* (que incluye la identificación) como la fase *emisión del voto* se realizan con la misma tarjeta y en base a la información en ella contenida.

La exigencia de que el voto se emita de forma anónima, de manera que ningún dato identificativo del votante pueda ser utilizado en la red de votación está garantizada por los pasos seguidos en el protocolo de votación que se ejecuta en la CPU de la tarjeta (que han sido verificados y auditados previamente). Gracias al carácter *tamper-proof* de la tarjeta se garantiza que los programas en ella contenidos no pueden ser manipulados ni las claves almacenadas pueden ser leídas o alteradas fraudulentamente.

- b) En el esquema Votescrypt+, el DNle, que es la tarjeta que posee los datos identificativos del votante, no se usa en la *fase de de votación*. Solo se utiliza la tarjeta Votescrypt+ (que también es *tamper-proof*) en la que están almacenados los protocolos seguros para emitir el voto. Como la tarjeta Votescrypt+ solo contiene una autorización para que su poseedor pueda votar, existe una garantía procedimental añadida que asegura que el voto es anónimo. No obstante, debido a ese desacoplo, es necesario garantizar que el votante, desde el instante que retira su DNle y su tarjeta Votescrypt+ al finalizar la fase de autenticación (para ese votante) no pueda tener contacto con persona alguna con la que pueda intercambiar o ceder su tarjeta, situación que, caso de permitirse, facilitaría la venta o cesión de voto. Esta prevención procedimental no tiene porqué establecerse en Votescrypt porque en la fase de votación el votante tiene que autenticarse de nuevo ante su tarjeta.

Fase de Verificación

- a) En Votescrypt el votante acude a las cabinas de verificación solo con su tarjeta Votescrypt. Para garantizar la corrección de la operación y evitar la coacción o venta de votos se exige al votante se autentique ante su propia tarjeta Votescrypt.

³ El Centro de Validación es un recurso externo a Votescrypt, ofrecido por la infraestructura del DNle.

- b) En el caso de Votescrypt+ será necesario algo más que portar la tarjeta Votescrypt+ ya que ésta no contiene la identificación del votante. Por ello en la Cabina de Votación se exige al votante que aporte también su DNIe. Para no tener que recurrir a verificar de nuevo la identificación del votante (usando el *Centro de Validación del DNIe*) con los consiguientes riesgos de establecer conexiones entre voto y votante, se recurre a un artificio consistente en que en la fase de autenticación la tarjeta Votescrypt+ genera un testigo aleatorio para verificación que el DNIe firma con su clave privada. Este testigo en claro y firmado es almacenado en la tarjeta Votescrypt+, y en la fase de verificación la tarjeta solo tiene que pedir al DNIe que se introduce en el lector correspondiente que firme de nuevo el testigo para verificación. Comprobando esta firma con la que tenía almacenada (que no conlleva identificación del votante) podrá averiguar si se trata o no del propietario verdadero de la tarjeta Votescrypt+. También en este caso, Votescrypt+ delega en el esquema DNIe la garantía de que el portador del DNIe es su verdadero propietario.

Como se ha comentado, todos los actores que intervienen en el proceso (Administrador, Interventores, Autoridad de Elección, etc.) lo hacen a través de tarjetas inteligentes diseñadas “ad hoc”, de tal modo que esas tarjetas contienen de forma segura todos los datos y todas las claves que se corresponden con el rol que desempeñan en el sistema.

4. Breve descripción de los protocolos en Votescrypt+

La fase de autenticación del votante comienza con el proceso de identificación del mismo de acuerdo a los protocolos y recursos de Votescrypt teniendo en consideración la existencia del DNIe con capacidad para autenticar a un ciudadano y firmar en su nombre, así como los procedimientos para su uso. Con esta disposición (figura 4) se pone en contacto el votante con el Sistema Administrador, realizando los pasos consecutivos que, con la colaboración del Centro de Validación del DNIe, conducen a la identificación del ciudadano por canal seguro.

En esta fase, se generan piezas de información que permiten aportar pruebas robustas a una auditoría, que detectaría la denegación arbitraria del derecho a voto en caso de producirse. En el paso “a”, la Tarjeta Votescrypt+ (SVC+) solicita al DNIe que comience el procedimiento de identificación, el DNIe contesta con la información “b” que es enviada por medio de la SVC+ y del Sistema Administrador hacia el Centro de Validación (paso “c”), cuya respuesta (paso “d”) es la aprobación (o no) de la información presentada por el DNIe. El Sistema Administrador enviará hacia la Tarjeta SVC+ una pieza de información firmada (paso “e”) con el resultado de la operación. Esta pieza de información, en caso de denegación debe indicar el motivo por el que produjo este evento. La SVC+ comprueba la firma sobre dicha información del Sistema Administrador y la almacena en un sitio seguro, que impida su borrado o modificación, accidental o voluntario. Estas piezas de información guardadas en los registros del Sistema Administrador y en la SVC+ permitirán a un auditor, desvelar la causa del impedimento a votar, disuadiendo con ello de la tentación de aplicar en este punto una denegación arbitraria al derecho a voto de determinados ciudadanos.

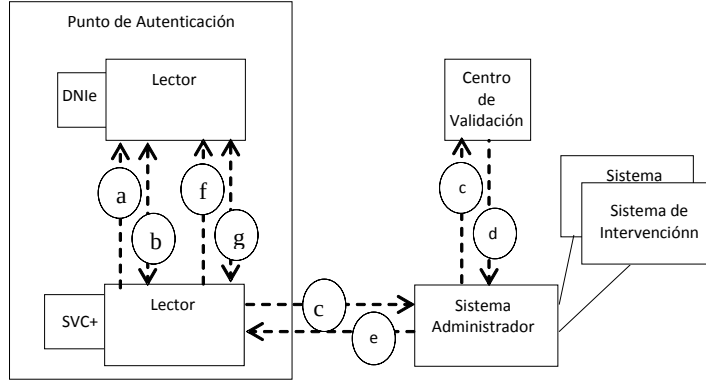


Figura: 4 Proceso de Identificación del ciudadano.

Puesto que el DNIe no está pensado para descifrar información que se le envíe de forma confidencial al votante y que con el acto anterior solamente se ha satisfecho un paso previo de la autenticación de ciudadano (todavía no se puede decir que es un votante), la SVC+ genera una clave para la recepción confidencial de información (K_{aut}), un *testigo de verificación*, un par de claves (k_{dv} , k_{cv}) y los factores de opacidad necesarios para opacar k_{dv} . Los pasos “f” y “g” representan la firma con el DNIe de la clave k_{dv} opacada para todos los destinatarios, el identificador del votante y del *testigo de verificación*.

La figura 5 representa el intercambio de información que se explica a continuación (el número de la viñeta coincide con el número dentro de cada círculo en la figura):

En la red de autenticación

1. El paso 1 representa que el proceso de Identificación descrito anteriormente se ha realizado satisfactoriamente. La Tarjeta SVC+ almacena de forma segura y protegida el *testigo de verificación*, el identificador de votante (*Voter Id*), y la k_{dv} opacada para el sistema administrador y para cada uno de los sistemas de intervención, todo ello firmado por el DNIe.
2. La identificación del votante y las claves opacadas y firmadas por el votante, concatenada con la clave de sesión (k_{aut}) que se usará en la comunicación de vuelta, es enviada confidencialmente desde la SVC+ hacia el Sistema Administrador a través del Punto de Autenticación.⁴

$ASP [(k_{aut}), V_s(Voter Id), (O_{AS}(k_{dv}), V_s(O_{AS}(k_{dv}))), AS),$
 $(O_{IS1}(k_{dv}), V_s(O_{IS1}(k_{dv}))), IS1), ...]$

⁴ $AS_P[xx]$ significa que el mensaje xx es cifrado con la clave pública de AS (Administrator System)

$V_s[xx]$ significa que el mensaje xx es cifrado con la clave secreta de V (Voter)

$O_{AS}[xx]$ significa que el mensaje xx ha sido opacado para que AS lo firme a ciegas.

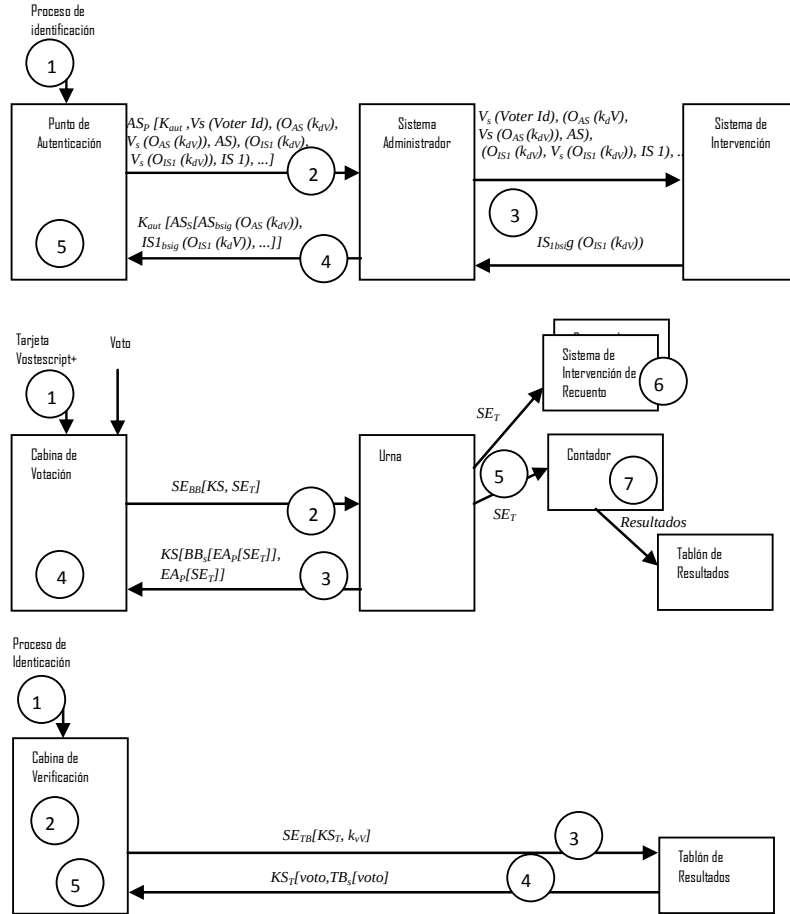


Figura 5: Protocolo Votescrypt+ en las tres redes virtuales

3. El Sistema Administrador descifra los datos recibidos y los reenvía hacia todos los Sistemas de Intervención de autenticación:

$[Vs (Voter Id), (O_{AS} (k_{dV}), Vs (O_{AS} (k_{dV})), AS), (O_{ISI} (k_{dV}), Vs (O_{ISI} (k_{dV})), IS1), ...]$

A continuación tanto el Sistema Administrador como los Sistemas de Intervención verifican la identidad del votante y su firma, y si no le han firmado la autorización con anterioridad, proceden a firmar a ciegas la clave k_{dV} opacada correspondiente. El Sistema Administrador recolecta toda esta información.⁵

$IS_{1bsig} (O_{ISI} (k_{dV}))$

4. Este paquete de claves opacadas y firmadas es firmado a su vez por el Sistema Administrador y cifrado con la clave de sesión de autenticación del votante para

⁵ $IS_{1bsig} [xx]$ significa que el mensaje xx ha sido firmado a ciegas por IS_1 (Intervention system 1)

enviarlo al Punto de Autenticación de forma confidencial con prueba de origen.⁶

$k_{aut} [AS_S[AS_{bsig}(O_{AS}(k_{dV})), IS_{1bsig}(O_{IS1}(k_{dV})), \dots]]$

5. El Punto de Autenticación pasa la información a la Tarjeta SVC+, que es el único dispositivo capaz de conocer el contenido y verificar las firmas. Una vez desopacadas las k_{dV} firmadas, conformarán la *autorización* para votar que en la siguiente fase será ligada de forma inseparable al voto.

En la red de votación

Finalizados los pasos que se dan en la red de autenticación, el votante se dirige a la Cabina de Votación donde procede a entregar el voto usando únicamente de la SVC+.

1. En el dispositivo que se encuentra dentro de Cabina de Votación el votante inserta su SVC+.
2. La Cabina de Votación le solicita el voto el cual es cifrado dentro de la SVC+ con k_{cV} (pareja de k_{dV}), concatenado con k_{dV} y con la autorización (k_{dV} firmada por los Sistemas de intervención de autenticación y por el Sistema de Administrador). Esta información, que en última instancia será descifrada por el Contador y por los Sistemas de Intervención de Recuento podría posteriormente poner en riesgo el secreto del voto, por lo que se protegerá dentro de un *Sobre Seguro entre la tarjeta y el Contador* (Secure Envelope, SE_T).

$SE_T = [k_{cV}[Voto], k_{dV}, AS_{bsig}(O_{AS}(k_{dV})), IS_{1bsig}(O_{IS1}(k_{dV})), \dots]$

Se genera una clave simétrica KS, que junto al *Sobre Seguro* anterior, es introducida en un *Sobre Seguro* externo, esta vez *entre la tarjeta y la Urna*. Esta pieza de información es enviada a la Urna (Ballot Box): $SE_{BB}[KS, SE_T]$

3. La Urna elimina el *Sobre Seguro* externo y separa las dos piezas de información: SE_T y KS. El SE_T es almacenado en la Urna hasta el que culmine el período destinado a la emisión de votos y a la vez es convertido en *comprobante* al ser cifrado con la clave pública de la Autoridad de Elección, firmado por la Urna y cifrado con KS para ser enviado confidencialmente hacia la Cabina de Votación. $KS[BBs[EA_p[SE_T]], EA_p[SE_T]]$
4. La Cabina de Votación entrega la información a la SVC+ la cual obtiene y verifica la firma de la Urna y almacena de forma segura el *comprobante* que solamente podrá ser desvelado si el votante decide presentarlo a la Autoridad de Elección.
5. Finalizado el período de recepción de votos se requiere la presencia de las personas que ejercen de Administrador y de Interventores para, entre todos, construir la clave necesaria para *abrir la Urna* e iniciar el proceso de volcado de su contenido hacia el Contador y los Sistemas de Intervención de Recuento, ofreciendo así garantías de que nadie ha poseído previamente los resultados intermedios de la votación a modo de información privilegiada. En realidad se requerirá la presencia del Administrador y de una cantidad “suficiente” de

⁶ $k_{aut} [xx]$ significa que el mensaje xx ha sido cifrado con la clave simétrica k_{aut} .

Interventores (para impedir que uno de ellos, o un número reducido de los mismos, puedan boicotear todo el proceso electoral) los cuales insertarán sus DNIE y las tarjetas inteligentes que identifican sus respectivos roles frente al sistema y tras un proceso de identificación biométrica recompondrán la clave que permite abrir la Urna. Se establecen los mecanismos de comparación que permite verificar que todos los sistemas reciben la misma información.

6. Nuevamente se requiere la presencia de las personas que ejercen de Administrador y de Interventores para, entre todos, construir esta vez la clave privada del contador. Esta clave es entregada al Contador y a todos los Sistemas de Intervención de Recuento para que puedan conocer el contenido interno de cada uno los votos cifrados que han recibido de la Urna. Una vez abiertos todos los sobres SE_T se lleva a cabo el recuento
7. El Contador envía al Tablón de Resultados (Tally Board, TB) la información compuesta por la clave k_{dv} y las firmas del Administrador y los Sistemas de Intervención sobre clave k_{dv} , junto con el voto descifrado. Una vez publicados los resultados, cada Interventor verificará que esta información coincida con la que su sistema tiene y en caso contrario, si hace falta, recurrirá a la autoridad de Elección para que se realicen las comprobaciones de registros necesarios.

En la red de verificación

1. Después la publicación de resultados, durante un período limitado de tiempo, cualquier votante puede dirigirse, con su DNIE y su SVC+, a una Cabina de Verificación (VeC) donde encuentra los dos dispositivos adecuados que le permiten inicialmente autenticar tal como lo hizo para obtener la autorización que le dio derecho a votar, aunque en esta ocasión no se usa el Centro de Validación.
2. La SVC+ le presenta al DNIE el *testigo de verificación* y comprobado que el votante está presentado su propia SVC+, continúa la operación de verificación.
3. La tarjeta introduce la k_{dv} y una clave de sesión KS_T en un *sobre seguro* entre la cabina de verificación y el Tablón de Resultados SE_{TB} y lo envía al Tablón. ($SE_{TB}=[k_{dv}, KS_T]$) El Tablón de Resultados recupera k_{dv} y localiza el voto que ha sido descifrado con dicha clave, firma el voto y lo cifra con KS_T para enviarlo de vuelta a la Cabina de Verificación: $KS_T[voto, TB[voto]]$
4. La VeC pasa la pieza de información a la SVC+ la cual abre el sobre, verifica la firma y envía el voto a la Cabina para que lo muestre al votante.

Si el votante no está de acuerdo con el voto que visualiza, puede presentar su SVC+ (acompañado o no por un perito de su confianza) ante la Autoridad de Elección, dónde, tras recuperar el comprobante de voto firmado por la Urna, se determinará si el comportamiento del sistema ha sido correcto o por el contrario ha existido error o fraude. Esta posibilidad de verificación actúa como disuasión frente a cualquier intento de destruir votos por parte de la Urna. La inserción de votos o modificación de voto quedaría en evidencia debido al conjunto de mecanismos criptográficos propuestos.

4. Conclusiones, trabajos futuros y agradecimientos

Con el desarrollo del esquema Votescript+ vuelve a ponerse de manifiesto que la seguridad del proceso de votación telemática no debe descansar solamente en la honestidad y capacitación profesional de los gestores del sistema sino fundamentalmente en la existencia de mecanismos criptográficos y protocolos telemáticos seguros que impiden la comisión de errores involuntarios, actuaciones ilícitas y la colusión maliciosa entre los distintos agentes que participan en la votación.

En los sistemas convencionales de votación (al menos en el caso de España) las garantías sobre la identidad de los votantes (y otros actores) descansan en la confianza en la fiabilidad del DNI convencional. De igual forma, en Votescript+, que propone una traslación del voto al ciberespacio, el aseguramiento de la identidad de los votantes se delega en el esquema DNIE. A medida de que, con el tiempo, este aseguramiento vaya siendo cada vez más robusto y vaya ganando la confianza ciudadana, esa mejora la heredará, consecuentemente, Votescript+.

Como trabajos futuros cabe destacar la necesidad de adaptar el sistema de votación a personas con diferentes discapacidades, así como la necesidad de contemplar votaciones multi-urna para los casos en los que los recuentos deban realizarse separadamente para distintos grupos de votantes.

Agradecimientos. Los autores de este artículo quieren manifestar su agradecimiento a los restantes miembros del grupo de investigación que han participado en las anteriores propuestas y desarrollos enmarcados en el Proyecto Votescript. De forma especial quieren trasladar este agradecimiento a Ana Gómez Oliva, Sergio Sánchez García, José David Carracedo Verde y Jesús Moreno Blázquez.

Referencias

1. <http://www.brunazo.eng.br/voto-e/textos/index.htm#portugues>
2. <http://www.cne.gov.ve/>.
3. Mercuri R. Testimony presented to the U.S. House of Representatives Committee on Science. <http://www.nist.gov/hearings/2001/votetech.htm>.
4. Carracedo Gallardo, J. Seguridad en redes telemáticas. Madrid, McGraw-Hill, 2004. EAN 9788448141578.
5. Proyecto Votescript. <http://vototelematico.diatel.upm.es/>.
6. Hoyo de Pinares. <http://vototelematico.diatel.upm.es/pag/pinares.htm>.
7. Carracedo Gallardo, J. Gómez Oliva, A. Carracedo Verde, J.D. Sistema VOTESCRIPT: Una propuesta innovadora desarrollada para resolver los problemas clásicos de votación electrónica, Actas CIBSI '03. México DF. Servicio de Publicaciones del Instituto Politécnico Nacional, 2003. ISBN: 970-36-010409.
8. Gómez Oliva, Sánchez García, S. Pérez Belleboni, E. Electronic Voting: pg39-49. Contributions to Traditional Electronic Voting Systems in order to Reinforce Citizen Confidence. Bregenz, Austria : Robert Krimmer, 2006. ISBN 978-3-88579-180-5.
9. Pérez. Belleboni, E., Carracedo, J. trabajo tutelado. http://tsic.diatel.upm.es/docs/Trabajo_tutelado_Emilía.pdf
10. Portal oficial sobre el DNIE. <http://www.dnielectronico.es>

Experiments on command dimension reduction in masquerader detection

Carlos Benitez¹ and Pablo Fierens²

¹ CITEFA - Instituto de Investigaciones Científicas y Técnicas para la Defensa
San J. B. de La Salle 4397 (B1603ALO), Villa Martelli, Buenos Aires, Argentina
Tel.: 5411-4709-8289, Fax: 5411-4709-5363
cbenitez@citefa.gov.ar

² ITBA - Instituto Tecnológico de Buenos Aires
Av. Eduardo Madero 399 (C1106ACD), Buenos Aires, Argentina
Tel.: (5411) 6393-4822
pfierens@itba.edu.ar

Abstract. We deal with the problem of dimension reduction in masquerader detection through command-line behavior. Although it has been previously suggested that unpopular commands are more relevant for this task, it is shown that there is no conclusive evidence in favor of this hypothesis. Moreover, it is also shown that selection of a fraction of the most popular or frequently used commands leads to a smooth degradation of a masquerader detection algorithm, while the selection of the most unpopular or infrequent commands produces a degradation which is worse than that of simple random selection.

Some evidence is provided that the best performance of a masquerader detection algorithm may not necessarily correspond to accounting for all commands in the training data, but for a smaller, adequately chosen fraction of them. We verify this conclusion using two different datasets and two different masquerader detection algorithms.

Finally, the empirical evidence provided in this paper suggests that, for many masquerader detection techniques, it may be convenient to work with a small fraction of the most popular or frequently used commands.

Keywords. masquerader detection, command-line behavior, dimension reduction, computer security.

1 Introduction

In computer intrusion detection, masqueraders are illegitimate users that try to impersonate legitimate ones stealing passwords and using real user accounts. There are different aspects of masquerader behavior which may be taken into account to detect her, e.g., the list of directories she accessed, the programs she executed, the websites she visited, etc. Among all these aspects, the sequence of commands typed by the masquerader has been one of the most popular choices in the literature (see, e.g., [1][2][3][4][5][6][7][8][9] and [10]). The usual approach

has been to monitor the command-line behavior of legitimate users for a long period in order to build a sufficiently large data corpus, and to apply a technique that looks for anomalies when new commands are typed.

As the “normal”-behavior dataset grows, the detection of masqueraders may become more accurate, but it may also become more complex, i.e., there is more data that needs to be considered before taking a decision. Moreover, intruder detection should be done in real-time, requiring efficient algorithms. In order to deal with both, the data set growth and the efficiency requirement, the dimension reduction of the detection algorithms input might be very useful. This paper deals exclusively with the problem of the dimension reduction in masquerader detection through command-line behavior.

The approach to dimension reduction proposed in this work is very simple: only a subset of all possible commands is considered when applying a masquerader detection technique. Then, the problem of choosing that subset arises. Related work, summarized in Section 2, suggests several criteria which are analyzed in Section 4. In particular, selection criteria are compared on the basis of their behavior when used with the same masquerader detection algorithm. The detection algorithm is based on a variation of a simple statistic put forward by Schonlau et al. [2] which is studied in Section 3.

Finally, in Section 6 the findings are summarized.

2 Related work

It has been argued by Schonlau et al. [1,2] that unpopular commands are more important than popular ones for the detection of masqueraders. Intuitively, there are some commands which are used by a small group of users and which are rarely used by other users: these commands allow to almost uniquely identify users. This idea suggests the possibility of reducing the dimension of the masquerader detection problem by keeping only a fraction of the most unpopular commands. However, Section 3 shows that the quantitative analysis carried out in [2] does not necessarily lead to the conclusion that keeping the most unpopular commands is the best choice.

Wan et al. [5] use high-frequency commands as signatures of normal user behavior and, by means of a particular decision mechanism, find better results than those reported by Schonlau in [1]. The authors of [5] obtain good results using the 40 highest-frequency commands, but they did not showed how the performance changes as the number of commands is varied. One of the goals of this paper is to explicitly evaluate this relation.

Kim and Cha [7] introduce the concept of “common commands”, that is, commands used by more than a given number of users. Moreover, they show that the use of common commands improves the performance of their masquerader detection algorithm. This observation suggests that it may be convenient to retain only the most popular commands. This hypothesis is tested in this paper.

2.1 Schonlau’s dataset

In this paper, the same data as Schonlau’s et al. [1,2] is used. Command data can be downloaded from Matthias Schonlau’s Internet homepage [11]. This dataset consists of commands collected from UNIX `acct` audit data. From all fields of audit data provided by `acct`, only the username and the command were taken. The command history of 70 users was used. Out of these 70 users, 50 were chosen randomly to represent the user base for testing, and the remaining 20 were used as masqueraders. Data was split into 50 files corresponding to 50 different users. Each file contains 15000 commands, one per line. The first 5000 commands of each file are considered genuine. The remaining 10000 commands of each user are divided into 100 blocks of 100 commands each. These blocks are seeded with masquerading users, i.e. with data from another user not among the 50 users. A block is a masquerader with a probability of 1%. If a block is a masquerader, then the following one is a masquerader too with a probability of 80%. As a result, approximately 5% of the test data contain masquerades. Not all users test data have masquerades interspersed.

It should be noted that Schonlau collected truncated commands, i.e., commands with no arguments nor any parameters, as this additional information was not provided by `acct`. It should also be mentioned that due to the way `acct` collects audit data from the system, it is impossible to tell commands typed by human users on the command line prompt from those run automatically from shell scripts. This can lead to some users having a very regular pattern of few commands, due to the repetitive automated execution of shell scripts, such as `.profile` and `make` files.

3 A masquerader detection algorithm

Schonlau et al. [1,2] argued that unpopular commands are more important than popular ones for masquerader detection. Their argument is based on a quantitative analysis of the results obtained by a particular masquerader detection algorithm. This algorithm has two steps:

- Whenever a new session of commands is analyzed, Schonlau et al. evaluate a simple statistic based on that session data and information from previous sessions (i.e., the “normal”-behavior dataset).
- This statistic is compared to a threshold in order to decide whether the commands were typed by an intruder or a legitimate user.

Schonlau and colleagues [2] show that good results are obtained when the particular statistic used takes into account the popularity of the commands, assigning a heavier weight to the unpopular ones. However, they did not compare those results with similar ones obtained using different weighting functions. In this section, such a comparison is made and it is found that the weighting function proposed by Schonlau et al. may not be the optimal and, hence, it may not be completely true that unpopular commands are more relevant to the detection task than popular ones.

3.1 Schonlau’s statistic

In order to analyze Schonlau’s statistic, some notation is needed:

- N_u (n_u) is the length of user u training (test) data;
- N_{uk} (n_{uk}) is the number of times user u used command k in training (test) data;
- K is the total number of distinct commands;
- U is the total number of users;
- U_k is the number of users that, in the training dataset, use command k .

Schonlau et al. [2] proposed a masquerader detection algorithm that they called *Uniqueness*. This approach is based on the following simple statistic:

$$x_u = \frac{1}{n_u} \sum_{k=1}^K F_{uk} \left(1 - \frac{U_k}{U}\right) n_{uk} \quad (1)$$

where F_{uk} is essentially the frequency of use of command k by user u in the “normal”-behavior dataset. More precisely, F_{uk} is defined as

$$F_{uk} = \begin{cases} \frac{v_{uk}}{v_k} & \text{if user } u \text{ training data contains command } k, \\ -1 & \text{otherwise,} \end{cases} \quad (2)$$

where

$$v_{uk} = \frac{N_{uk}}{N_u}, \quad v_k = \sum_u v_{uk}. \quad (3)$$

Whenever a new command session is available, this statistic is computed and compared to a threshold value: if the statistic is lower than the threshold, then it is decided that the session must correspond to a masquerader.

According to Schonlau et al. [2], U_k denotes command k popularity, so the term $(1 - U_k/U)$ acts as a *uniqueness index*, i.e., the index is 0 if all users have used this command before and is 1 if none of the users has used it before.

3.2 The importance of being unpopular

Note that the uniqueness index works as a weighting function for F_{uk} in Eqn. 1. As such, it has nothing special, except for the intuition offered by Schonlau and colleagues:

The uniqueness approach is based on the idea that commands not previously seen in the training data may indicate an attempted masquerade. Moreover, the fewer users that are known to use that command, the more indicative that command is of a masquerade. (from Schonlau et al. [1])

In order to support this intuitive lemma, the authors of [2] provided two quantitative elements:

1. They computed the values of the statistic using test data from user i and training data for user j , for all pair (i, j) of users. The results were displayed in an $U \times U$ figure in which darker squares corresponded to higher values of the statistics (“scores”) and lighter squares to lower scores. Ideally diagonal scores should be perfectly discriminated from non-diagonal ones and the only exceptions should be the masqueraders. The results are reproduced in Fig. 1(a), where it is clear that a good discrimination was achieved.
2. Schonlau et al. [1] computed the False Alarm Rate (FAR) and Missing Alarm Rate (MAR) as a function of the decision threshold. Moreover, they plotted FAR vs. MAR in a graph which is known as a ROC (Receiver Operating Characteristic) curve. The authors of [1] compared the resulting ROC curve to those obtained by other algorithms and they showed that the detection of masqueraders through the proposed statistics behaved reasonably well.

Although Fig. 1(a) shows good results when using the statistic proposed by Schonlau and colleagues, they do not provide enough information to support the claim that unpopular commands are more relevant to masquerader detection than popular ones. In order to effectively measure whether unpopularity is an important feature, the statistic in Eqn. 1 has been modified, replacing the uniqueness index $(1 - U_k/U)$ by a constant, i.e., the same weight is assigned to all commands. Fig. 1(b) and 2 show the results. As can be seen, there are no substantial differences neither between Fig. 1(a) and Fig. 1(b) nor between the corresponding ROC curves shown in Fig. 2. It can be concluded that Fig. 2 shows that there are almost no differences between ROC curves using a constant weight from others applying a higher weight to unpopular commands. Therefore, *no conclusive evidence is found in favor of the use of unpopular commands for masquerader detection.*

3.3 Generalization of the statistic

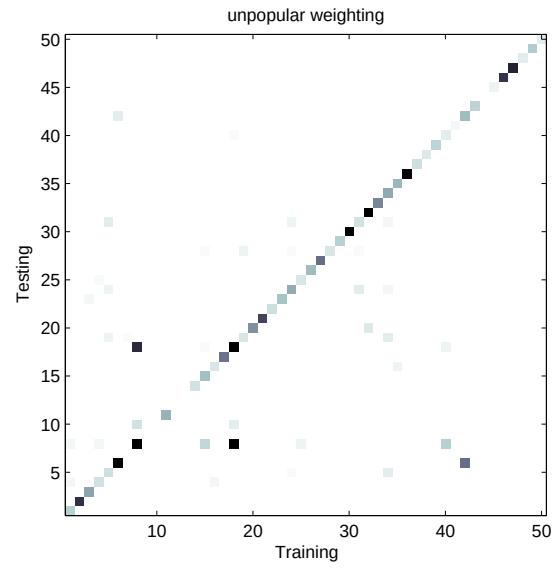
The uniqueness index and the constant weight are only two possible choices for the weighting function of a statistic like in Eqn. 1. In general, it can be said

$$x_u = \frac{1}{n_u} \sum_{k=1}^K F_{uk} W_{uk} n_{uk}, \quad (4)$$

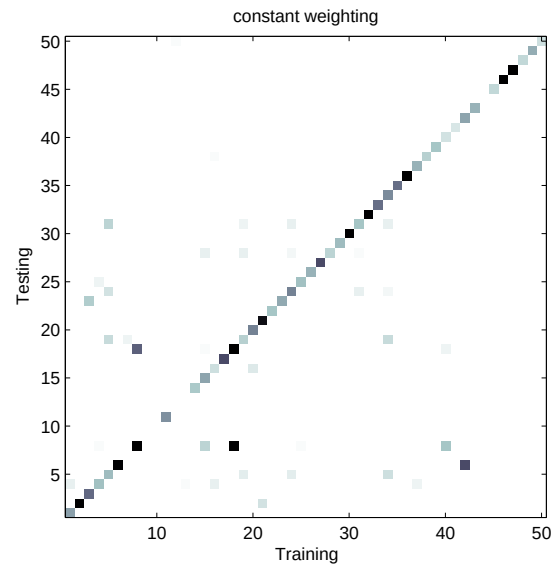
where W_{uk} is a weighting function which depends on the command and the user. Table 1 shows five possible different choices for the weighting function. Among those weighting functions, the command use index and the related inverse command use index require some comment.

The command use index is defined as v_k/V , where $V = \max(v_k)$. This index is defined using the same criteria and format than for popular commands. The variables v_{uk} and v_k are defined in Eqn. 3 so, the command use index is defined as:

$$\frac{v_k}{V} = \frac{1}{V} \times \left[\frac{1}{N_u} \sum_u N_{uk} \right], \quad (5)$$



(a)



(b)

Fig. 1. Visual display of Schonlau et al.'s statistic.

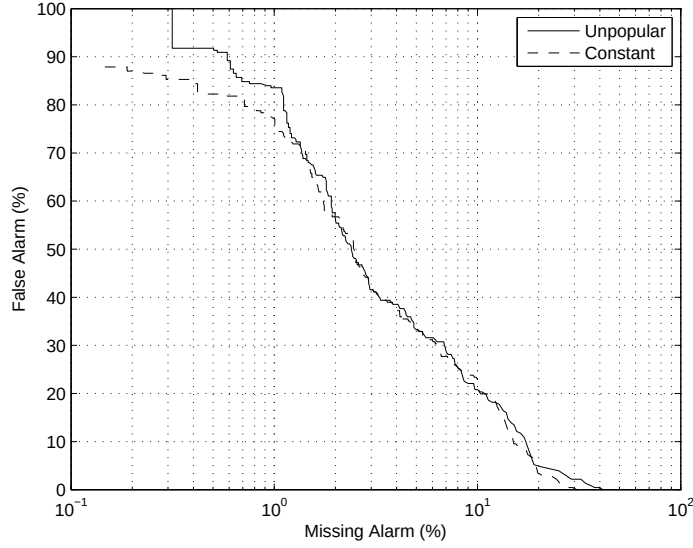


Fig. 2. ROC curves using Schonlau et al.'s statistic

then, it is clear that the command use index is the average frequency of use of command k , where the proportionality constant $\frac{1}{V N_u}$ acts as a normalization factor.

Criterion	Comments	Weighting vector
uniqueness index	$\uparrow$ unpopularity $\Rightarrow \uparrow$ weight	$1 - U_k/U$
inverse uniqueness index	$\uparrow$ unpopularity $\Rightarrow \downarrow$ weight	U_k/U
command use index	$\uparrow$ use frequency $\Rightarrow \uparrow$ weight	$1 - v_k/V$
inverse command use index	$\uparrow$ use frequency $\Rightarrow \downarrow$ weight	v_k/V
constant	same weight for all commands	1

Table 1. Weighing functions

ROC curves were calculated using the statistic in Eqn. 4 and the five different weighting functions in Table 1. Instead of reproducing the curves as it has been done in Fig. 2, the behavior of each ROC curve has been summarized by means of the Area Under the Curve (AUC): the lower the AUC, the better is the discrimination power of the masquerader detection algorithm. The area under ROC curves are shown in Table 2, where, in the third column, a relative order score is shown (1 is the best, 5 is the worse). There, all weighting functions give similar results, except for the command use index. Therefore, the results

in Table 2 support the hypothesis that there is nothing special about using unpopular commands for masquerader detection.

Weight	AUC score	
uniqueness index	0.059	3
inverse uniqueness index	0.063	4
command use index	0.083	5
inverse command use index	0.056	2
constant	0.053	1

Table 2. Area under ROC curves.

4 Command Extraction

The approach of this paper to the dimension reduction problem is to simply select the proper command subset. Therefore, the difficulty is shifted to find the criterion for such selection. In Section 3.2 it has been shown that choosing only unpopular commands might not be the best choice. In this section other possibilities are explored.

The weighting functions in Table 1 may serve as motivation for several command subset selection criteria. Some of them are:

1. unpopular commands, i.e., those with high uniqueness index;
2. popular commands, with high inverse uniqueness index;
3. frequently used commands, with high command use index;
4. infrequently used commands, with high inverse command use index.

Note that there is no immediate translation of the constant weighting function to a criterion for command selection. Two reasonable alternatives are:

1. to select the subset of commands at random;
2. to select the subset of commands according to its alphanumerical order.

The selection of a subset of commands is more difficult for some of these criteria than others. Fig. 3 shows the (sorted) weights of each command in the training data according to each of the five weighting functions in Table 1. It can be seen that many commands are unpopular and have a similar (high) uniqueness index. Therefore, it is difficult to choose a subset of commands of a given size according to their uniqueness indices: at some point, a rather arbitrary choice between commands of (almost) the same index has to be done. A similar argument holds for infrequently used commands, i.e., with low command use index. On the other hand, clear distinctions can be made among popular (high inverse uniqueness index) and frequently used (high command use index) commands.

In order to compare the different selection criteria, we proceed as follows. For each criterion, the size of the selected subset is varied. For each subset,

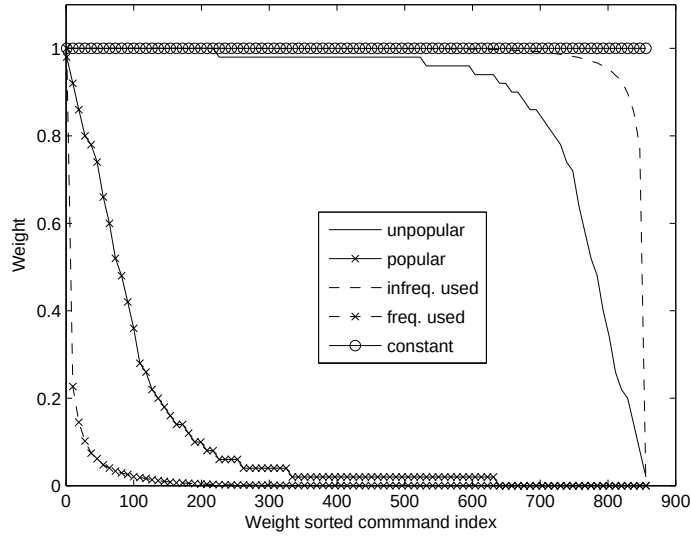


Fig. 3. Weights of training data commands.

the corresponding ROC curve is calculated, using the statistic in Eqn. 4 with *constant weight*³ and changing the decision threshold. We finally summarize the information of each ROC curve by the area under it. Fig. 4 shows the results (AUC) as a function of the fraction of commands selected for each criterion. Note that all curves agree on two points. On one hand, all curves start at 0.5, i.e., when no commands are taken into account to make a decision, then the decision becomes completely random: either a session corresponds to a legitimate user or a masquerader with equal probability. On the other hand, as the fraction of selected users is increased, each curve tends to the value 0.053 reported in the last row of Table 2. In the particular case of random selection, the curve presented is the average of ten runs.

Two important conclusions can be extracted from the results shown in Fig. 4. First of all, the degradation of the masquerader detection algorithm as the fraction of commands decreases, varies a lot depending on the criterion used. Indeed, *selection of popular and frequently used commands leads to a smooth degradation, while the selection of unpopular and infrequent commands produces a degradation which is worse than that of simple random selection*. Second, *the best performance of the algorithm may not necessarily correspond to accounting*

³ In Section 3.2 it was shown that the particular weight chosen is not very important. Moreover, there is evidence in favor of a constant weight.

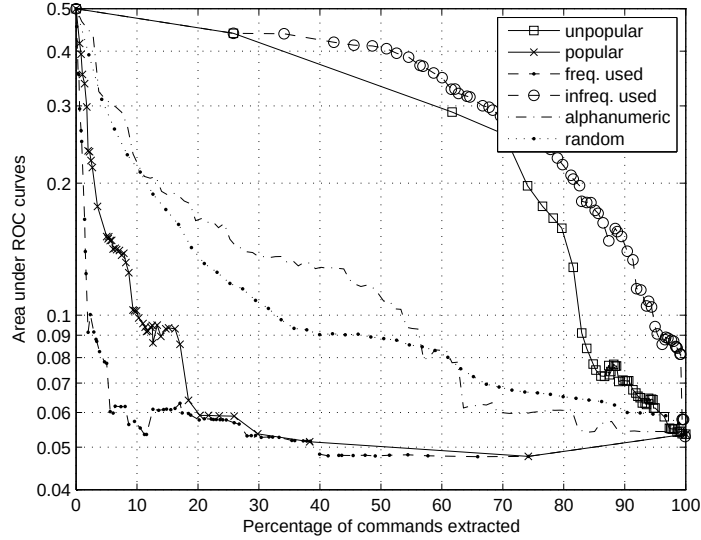


Fig. 4. Area under ROC curves vs. percentage of extracted commands.

for all commands in the training data, as the curves corresponding to the popular and frequently-used command selection criteria show.

5 Validation

Last section showed the results of calculating ROC curves by extracting commands from Schonlau’s dataset and detecting masqueraders using Schonlau’s statistic. In order to test the validity of the conclusions presented at the end of Section 4, further tests were performed with another publicly available command dataset, Greenberg’s Dataset [12]. Moreover, another detection algorithm, based on Naive-Bayes, was used.

5.1 Greenberg’s Data

This dataset[12] was collected by Saul Greenberg as part of a project of his PhD thesis in 1988. It contains data from 168 Unix users using *csh* (C shell) as command line shell. All of these users were unpaid volunteers and were either students or employees of the University of Calgary. Greenberg classified users in four groups as follows: Novice Programmers, Experienced Programmers, Computer Scientists and Non-programmers.

Greenberg’s data was preprocessed in order to compare results with Schonlau’s dataset in exactly the same way as Maxion did in his work [3]. Only truncated command-line data, that is, without command parameters, was preserved. Out of 168 Greenberg’s users, 50 were selected as testing subjects having a total number of commands between 2000 and 5000. Out of the remaining 118 users, 25 were selected randomly to serve as a source of masquerader commands. The data of the subject users was truncated to 2000 command lines: the first 1000 commands to be used as training data and the other 1000 commands to be used as testing data. The last 100 commands from each of the 25 masquerader users were concatenated, from which 30 blocks of 10 commands each were selected randomly to be injected into the testing data of every user without replacement, resulting in a testing stream of 130 blocks of 10 command lines for each subject user. Although the masquerader blocks and injection positions were selected randomly, they were held constant across the 50 subject users for consistency purposes.

5.2 Naive-Bayes algorithm

Maxion et al. [3,4] have shown that Naive-Bayes classifiers may be successfully used for detection of masqueraders. The detection algorithm implemented in this work is the no-update Naive-Bayes detector described in [4]. For each user u and for each command k in the training data, the conditional probability of appearance of command k in a block of user u is estimated as

$$P(k|u) = \frac{N_{uk} + \delta}{N_u + K \times \delta},$$

where N_{uk} , N_u and K are defined as in Section 3.1 and δ is a small value which is introduced to avoid zero counts. Given a test sequence of commands, $k_1 k_2 \dots k_m$, where m is the length of the sequence, the probability that the command k was generated by user u is given by⁴:

$$P(u|k_1 k_2 \dots k_m) = \frac{P(u) \times \prod_{i=1}^m P(k_i|u)}{P(k_1 k_2 \dots k_m)}.$$

For each user u , a model of “Not User u ” ($\bar{u}$) may also be built based on the training data of the remaining users. Indeed, we may estimate the probability

$$P(k|\bar{u}) = \frac{\sum_{i \neq u} N_{ik} + \delta}{\sum_{i \neq u} N_i + K \times \delta}.$$

Given a test sequence of commands, $k_1 k_2 \dots k_m$, we may compute the probability

$$P(\bar{u}|k_1 k_2 \dots k_m) = \frac{P(\bar{u}) \times \prod_{i=1}^m P(k_i|\bar{u})}{P(k_1 k_2 \dots k_m)}.$$

⁴ Bayes Theorem and conditional independence are used, that is, the hypotheses behind Naive-Bayes classifiers.

Therefore, the relative likelihood

$$L(u|k_1k_2\cdots k_m) = \frac{P(u|k_1k_2\cdots k_m)}{P(\bar{u}|k_1k_2\cdots k_m)}$$

is a measure of how likely it is that the given sequence of commands was generated by user u . Since Maxion et al. [4] assume, although not explicitly, that $P(u) = P(\bar{u})$, the relative likelihood simplifies to

$$L(u|k_1k_2\cdots k_m) = \frac{\prod_{i=1}^m P(k_i|u)}{\prod_{i=1}^m P(k_i|\bar{u})} = \prod_{i=1}^m \frac{P(k_i|u)}{P(k_i|\bar{u})}.$$

The detection algorithm decides whether the sequence corresponds to a masquerader by comparing the likelihood ratio to a threshold: if the likelihood ratio is too small, the algorithm assumes that the sequence was generated by a masquerader. If threshold is varied, then a ROC curve can be obtained.

5.3 Experiments

Following the same criteria than in Sec. 4, experiments were performed processing Schonlau's and Greenberg's datasets.

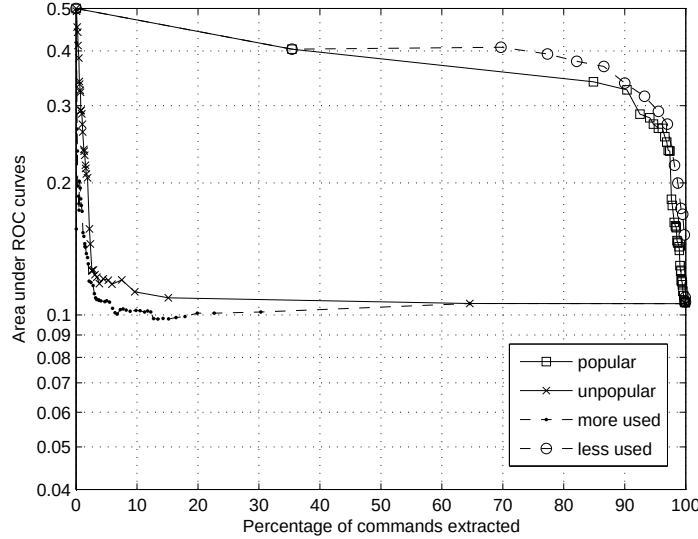
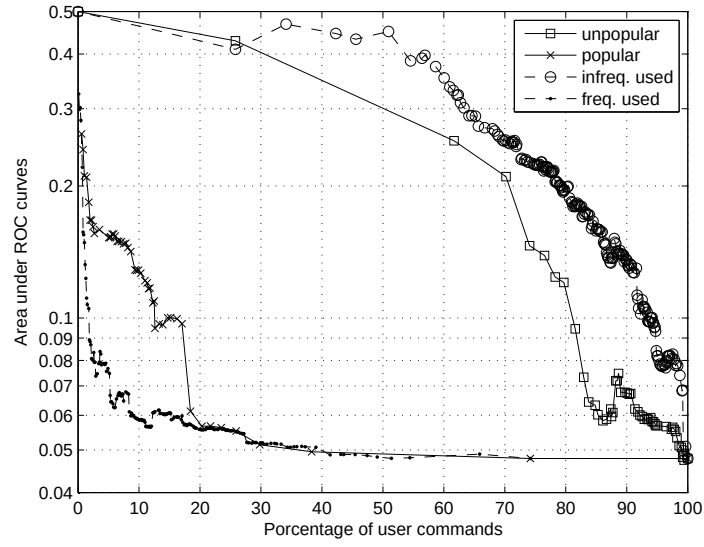
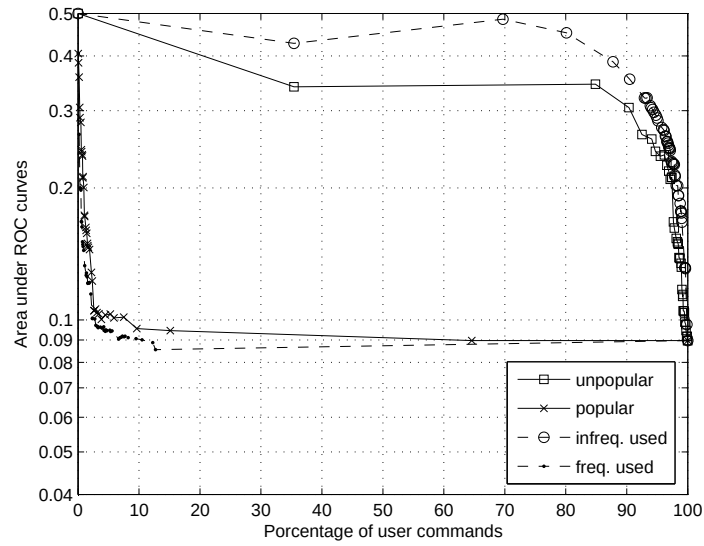


Fig. 5. Area under ROC curves for Greenberg dataset using Schonlau detector



(a) Schonlau dataset



(b) Greenberg dataset

Fig. 6. Area under ROC curves using Naive-Bayes detector

Figures 4, 5, 6(a), and 6(b) show a similar behavior of the Area under ROC curve as function of the fraction of commands extracted for the two datasets and the two detection algorithms. Moreover, it can be verified that the conclusions at the end of Section 4 are valid for both datasets and both detection algorithms. It should be noticed that random or alphabetical order for comparison are included only in 4.

6 Conclusions and future work

Command space dimension reduction in masquerader detection was tested. Figures 4, 5, 6(a), and 6(b) show that a small number of commands can be used without considerable degradation of the masquerader detection algorithm. According to the experiments, 20% of the most popular commands or 10% of the most frequently used commands are enough to get results comparable to those obtained using all commands. This reduction of the number of commands taken into account, improves masquerader detection by reducing computational complexity.

Posadas et al. [10] studied the case of a masquerader that knows the profile of the legitimate user. Such a knowledge is proved to be very detrimental for most masquerader detection algorithms, with the exception of the algorithm presented by the authors. It must be noted that one of the drawbacks of extracting just the most popular or frequently used commands is precisely that impersonation of a legitimate user is made easier for a masquerader with some knowledge of user's profile: the masquerader needs to know only the user's profile corresponding to a limited number of commands.

In the present work only Schonlau's statistic test and Naive-Bayes are used in order to compute masquerader detection efficiency using different subsets of extracted commands as input. In order to validate the dimension reduction techniques introduced here, also other detection algorithms like Normalized Compression Distance ([8],[9]), Grammar Extraction ([13]), Bayes One-Step Markov and Hybrid Multistep Markov ([1]) should be used. This is a matter of future work.

Dimension reduction analysis has been performed using truncated command line data, but Maxion [3] and Bertacchini and Benitez [9] shown that command line data enriched with command parameters allows a better performance in masquerader detection. Therefore, work needs to be done in order to extend the ideas of this paper to the case of enriched commands.

7 Acknowledgments

This work is part of a more extendend research project which is supported in part by ANPCyT (National Agency of Scientific Promotion and Technology) under the grant PICTO CITEFA 2004 18623. We gratefully acknowledge financial support from ANPCyT. We would also like to thank all members of CITEFA's Si6 Lab for their support specially to Luciano Bello and also to Cecilia Oriolo

for their suggestions and corrections. We would also like to thank the reviewers of this paper for their useful comments.

References

1. Schonlau, M., DuMouchel, W., Ju, W., M. Theus, A., Vardi, Y.: Computer intrusion: Detecting masquerades. *Statistical Science* **16** (2001) 58–74
2. Schonlau, M., Theus, M.: Detecting masquerades in intrusion detection based on unpopular commands. *Inf. Process. Lett.* **76**(1-2) (2000) 33–38
3. Maxion, R.: Masquerade detection using enriched command lines. In: *International Conference on Dependable Systems and Networks*, San Francisco, CA, IEEE (June 2003)
4. Maxion, R.A., Townsend, T.N.: Masquerade detection using truncated command lines. In: *International Conference on Dependable Systems and Networks (DSN-02)*, Washington, IEEE Computer Society Press (June 2002) 219–228
5. Wan, M.D., Wu, H.C., Kuo, Y.W., Marshall, J., Huang, S.H.S.: Detecting masqueraders using high frequency commands as signatures. In: *Advanced Information Networking and Applications - Workshops*. (March 2008) 596–601
6. Wang, W., Guan, X., Zhang, X.: Profiling program and user behaviors for anomaly intrusion detection based on non-negative matrix factorization. In: *43rd IEEE Conference on Decision and Control*, Paradise Island, Bahamas (Dec 2004) 99–104
7. Kim, H.S., Cha, S.D.: Empirical evaluation of svm-based masquerade detection using unix commands. *Computers & Security* **24**(2) (2005) 160–168
8. Bertacchini, M., Fierens, P.: Preliminary results on masquerader detection using compression based similarity metrics. *Electronic Journal of SADIO* **7**(1) (February 2007)
9. Bertacchini, M., Benitez, C.: NCD based masquerader detection using enriched command lines. In: *Proc. of the IV Congreso Iberoamericano de Seguridad Informatica (CIBSI'07)*, Mar del Plata, Argentina (November 2007) 329–338
10. Posadas, R., Max-Perera, C., Monroy, R., Nolasco, J.A.: Hybrid method for detecting masqueraders using session folding and hidden markov models. In: *MICAI 2006: Advances in Artificial Intelligence*, Mexico (November 2006) 622–631
11. Schonlau, M.: Masquerading user data. <http://www.schonlau.net/intrusion.html> (1998)
12. Greenberg, S.: Using unix: Collected traces of 168 users. Technical Report 1988-333-45, Department of Computer Science, University of Calgary, Calgary, Alberta, Canada (December 1988)
13. Latendresse, M.: Masquerade detection via customized grammars. In Julisch, K., Krgel, C., eds.: *DIMVA*. Volume 3548 of *Lecture Notes in Computer Science*., Berlin Heidelberg, Springer-Verlag (2005) 141–159

A Survey on Masquerader Detection Approaches

Maximiliano Bertacchini¹ and Pablo I. Fierens²

¹ CITEFA - Instituto de Investigaciones Científicas y Técnicas para la Defensa
San J. B. de La Salle 4397 (B1603ALO), Villa Martelli, Buenos Aires, Argentina
Tel.: 5411-4709-8285, Fax: 5411-4709-5363

`mbertacchini@citefa.gov.ar`

² ITBA - Instituto Tecnológico Buenos Aires
Av. Eduardo Madero 399 (C1106ACD), Buenos Aires, Argentina
Tel.: (5411) 6393-4822
`pfierens@itba.edu.ar`

Abstract. This paper presents a survey on the area of masquerader detection. The three most popular publicly available UNIX command-line datasets are showed and their features are compared. Several different masquerader detection approaches are reviewed and their results are compared applying the most popular measures of detection effectiveness in this area, introducing the most extensive quantitative comparison of results in literature. Possible ways for future work in this area are proposed as well.

Keywords. masquerader detection, command-line behavior, computer security, comparison, review, survey.

1 Introduction

In computer security, a masquerader is an intruder trying to impersonate a legitimate user. Early and effective intrusion detection is a critical factor in securing a computer system.

We are focusing on the detection of masqueraders in the UNIX command line history of computer users. Despite the prevalence of graphical user interfaces (GUIs), user interfaces based on a command line are still popular. Moreover, the same techniques can be applied for anomaly detection on other sequential data, such as sequences of system calls at the operating system level, packet flow in network activity, and even in other areas such as genetics and bioinformatics.

In the last decade, a vast amount of work has been published on this subject, with approaches covering a wide range of techniques such as statistics, artificial intelligence, data mining and bioinformatics. However, most of them share many details in common, such as the used dataset, data configuration and general training and testing methodology. Masquerader detection approaches have shown a rather steady progress with time, and probably better techniques will be conceived in the future.

Our goal is to present the most representative works to our knowledge in this area, from the seminal work by Schonlau et al. [1] in 2001, on to the present. This survey could serve as an introduction to the literature on the subject of masquerader detection for new researchers. We are also interested in exposing topics in this area which could be subject to further research.

This work is structured as follows: Section 2 presents the most popular publicly available UNIX command-line datasets and some of their shortcomings; Section 3 shows a review of masquerader detection approaches in literature; Section 4 briefly describes the most popular measures for evaluation of detection results; Section 5 makes a quantitative comparison of results achieved by the reviewed techniques; Section 6 presents our conclusions and possible areas for future work.

2 Datasets

2.1 Schonlau et al.

Matthias Schonlau et al. introduced their own dataset in [1] and [2] in 2000. It is also known as SEA dataset (Schonlau Et Al.), and it is freely available for download at [Matthias Schonlau's Internet homepage](http://www.schonlau.net)³. This dataset has since become the de-facto standard dataset and is widely used in the majority of papers about masquerader detection. Its associated configuration is also the most widely used. This allows, in principle, the comparison of results among different detection approaches.

This data set consists of commands collected from UNIX `acct` audit data. Of all fields of audit data provided by `acct` only the username and the command were taken. Data contains 50 different users with 15000 commands each. The first 5000 commands are considered genuine. The remaining 10000 commands of each user are divided into 100 blocks of 100 commands each. These blocks are seeded with masquerading users, i.e. with data of another user not among the 50 users. A block is a masquerader with a probability of 1%. If a block is a masquerader, then the following one is a masquerader too with a probability of 80%. As a result, approximately 5% of the test data contain masquerades. Not all users' testing data have masquerades interspersed.

Commands have no arguments nor any parameters, as this additional information was not provided by `acct`. Due to the way `acct` collects audit data from the system, it is impossible to tell commands typed by human users on the command line prompt from those run automatically from shell scripts. This can lead to some users having a very regular pattern of few commands, due to the repetitive automated execution of shell scripts.

1v49 Configuration An alternative configuration to the SEA dataset was proposed by Maxion et al. in [3] to address some methodological shortcomings found

³ Matthias Schonlau's homepage: <http://www.schonlau.net>.

in the original configuration, such as the fact that different masqueraders were injected into different users, and some users didn't even get any masquerader. This makes evaluation and error analysis very difficult. In 1v49 configuration, for each user the first 5000 commands of the other 49 users are used as masquerader data for testing purposes. Despite its methodological advantages, this configuration has not been widely used. Note that this configuration does not simulate the presence of masqueraders which is expected in real world data.

SEA-I A variation on the original SEA dataset is proposed by Posadas et al. in [4], called SEA-I, where the masquerader blocks are replaced by synthetic blocks created taking into account the command frequency of each user, trying to emulate the behavior of an intruder who tries to act like the legitimate user. As a result, masquerader detection on this dataset is more difficult, and more complex techniques are needed⁴.

2.2 Greenberg

This data set was collected by Saul Greenberg as part of his PhD thesis in 1988 [5]. It contains data from 168 Unix users using *cs*h (C shell) as command line shell. Greenberg classified users in four groups as follows: Novice Programmers, Experienced Programmers, Computer Scientists and Non-programmers. This dataset is available at [Saul Greenberg's homepage](#)⁵.

Data was collected and stored in plain text files which contain the following information: session start time, session end time, the command line as entered by the user, the current working directory, the alias expansion of the previous command (if any), an indication whether the line entered has a history expansion or not, and the error detected in the command line (if any).

This dataset was first used for masquerader detection purposes by Maxion et al. in [6]. The main advantage of this dataset in contrast with previous ones such as SEA is the availability of additional information for each command which may help to improve the effectiveness of masquerader detection. However, in practice, of all these additional data, only the arguments, parameters, and alias are used.

This dataset is often used in two different configurations simultaneously: *enriched* and *plain*. The enriched configuration includes the command, alias, arguments and parameters. The plain configuration only uses the command and alias, thus effectively mimicking the configuration of Schonlau et al's dataset. This allows the comparison between enriched and plain configurations of the same dataset in order to verify the hypothesis that these additional data improve the detection effectiveness.

2.3 Purdue University

Terran Lane and Carla E. Brodley introduced a dataset in 1997, which was later known as Purdue dataset or just "PU" dataset [7]. It consists of the UNIX shell

⁴ The authors of [4] present an alternative which is discussed in Section 3.3

⁵ Saul Greenberg's homepage: <http://pages.cpsc.ucalgary.ca/~saul/>

command histories of 4 users of the Purdue Millenium Lab, collected during a four month period. It is available at [the UCI KDD archive](#)⁶. The number of collected commands per user goes from 7769 to 22530, with an average of approximately 16500 commands per user. Commands are enriched, that is, command names, arguments and options are preserved, but filenames are omitted, due to the intuition that the *behavior* of users is more significant for their profiling than *content*. Very few works on masquerader detection use this dataset, probably because of the low number of users it contains.

2.4 Synthetic Datasets

Synthetic datasets provide a non-intrusive alternative to traditional ones, with the benefit of rapid availability and high customizability.

In 2004 Chinchani et al. introduced a framework for template-based command line data generation called RACOON [8]. It produces sequences of commands based on templates, which can be specified manually or created from any other data set. A template is composed of the following components and their statistical properties: commands, meta-commands, jobs, meta-command sequences, job hops and sessions. A session is basically a sequence of jobs, which are sequences of meta-commands. A meta-command is a set of similar commands in terms of their function.

One limitation of RACOON is that it is able to generate only truncated commands, not enriched ones. On the other hand, it can provide large amounts of command-line data, which is otherwise very difficult to obtain.

2.5 Comparison of Datasets

Table 1 shows a comparison between the previous three UNIX command-line datasets. As shown above, all of them consist of data from legitimate users. No public dataset of real intruders is known to the authors of this paper. Furthermore, although a thorough analysis and modelling of the behavior of intruders is lacking in this area, it is generally assumed that masqueraders can be detected by applying any of the techniques reviewed below. The number of users varies from just a few to more than one hundred; the size of each user’s dataset ranges from hundreds to tens of thousands of commands; and commands may be truncated. These datasets are also more than ten years old. Hence, a new UNIX command-line dataset improving on these features would be extremely useful. It could include real intruder data collected from honeypots and honeynets, should be sufficiently large, and should include additional pieces of information such as command parameters and options, and login sessions.

3 Review of Masquerader Detection Approaches

Masquerader detection approaches can be classified by means of several different criteria. Independently of the applied detection technique, most approaches are

⁶ The UCI KDD archive: <http://kdd.ics.uci.edu>

Table 1. Comparison of publicly available UNIX command line datasets

Dataset	Users	Cmds (total)	Cmds/user (average)	Enriched	Contam- inated	Sessions	Year	Time lapse
Greenberg	168	303628	1807	Yes	No	Yes	1988	4 months
PU	4	66177	16500	Yes	No	Yes	1997	4 months
SEA	50	750000	15000	No	Yes	No	2000	several months

based in a scoring system, where each block gets a score and is classified as normal or masquerader according to some detection *threshold*. We have chosen to subdivide these techniques according to the classifier algorithm each of them is based on. Other distinctive criteria for the classification of masquerader detection methods could be: data preprocessing and user profile construction, user profile updating, size of testing blocks (number of commands), the used dataset and its configuration.

3.1 Information-Theoretic-based Approaches

Schonlau et al. first proposed a rather simple compression-based approach in [1], called the Compression method. It is based on the premise that data from the same user compresses more readily than mixed data from different users. The score of a testing block is defined as the number of additional bytes needed to compress it when appended to the training data. The UNIX command `compress` was applied, which is based on the popular Lempel-Ziv algorithm [9]. In [1], results from the Compression method seem discouraging in comparison with the other five methods presented there.

In 2006 Bertacchini and Fierens applied a more sophisticated compression-based approach, the Normalized Compression Distance (NCD) on the SEA dataset [10]. NCD is a method for clustering and classification developed by Vitányi et al. [11], which is based on the notion of Kolmogorov Complexity [12]. This function is not computable, but may be approximated using real-world compressors such as `gzip` or `bzip`. Therefore, NCD relies on external real-world compressors to calculate the difference or “distance” between any two objects. In this case, the two objects being a suspicious block of commands and a user’s profile or their command-line history. Results were better than the Compression approach, and similar to Schonlau’s best results. A later work by the same authors [13] showed improved detection applying the NCD on Greenberg’s enriched command-line data.

In 2007 Evans et al. proposed a grammar-inference algorithm called MDL-compress [14] which uses Minimum Description Length (MDL) principles from the theory of Kolmogorov Complexity and Algorithmic Information Theory [12] to model legitimate user activity and use the resulting grammar to detect masqueraders. This algorithm is reported to produce results very similar to those of Schonlau et al.’s Compression approach [1], though no quantitative results are available to our knowledge.

3.2 Text Mining-based Approaches

Latendresse proposed in 2005 a text-mining approach [15] based on the Sequitur algorithm. The Sequitur algorithm extracts hierarchical structures from a string by constructing a context-free grammar [16]. For each user, a Sequitur grammar is generated to extract the repetitive sequences of commands and their associated frequencies, and each testing block is compared recursively with the legitimate user's profile. This method achieved high detection with low false alarm rates.

In 2004, Oka et al. proposed a masquerader detection method based on co-occurrence matrices in [17], called Eigen Co-occurrence Matrix (ECM). The ECM method models a user's behavior by correlating an event in a sequence with the following events appearing within a certain distance. It extracts the principal features of the so-called co-occurrence matrix of a user's sequence of UNIX commands through Principal Component Analysis (PCA) [18], thus reducing its inherent high dimensionality, and obtains a feature vector which can be classified with any classification technique based on vector representation. In this case, a simple Euclidean distance measure was used, achieving relatively poor detection results.

Oka et al. tried to improve their previous results in [19]. This approach builds a layered network to define the user's profile by multiplying the feature vector by the Eigen co-occurrence matrix. The same procedure is applied on each testing sequence and the resulting layered network is compared with the profile of the corresponding user; it is classified as anomalous or normal based on a per-user threshold. The results were promising, achieving a detection rate and false detection rate similar to those of the Naïve Bayes method. This approach has high computational cost, particularly during its training phase.

In 2006 Chen and Dong introduced a method called "One-Class Classification using Length Statistics of Emerging Patterns" (OCLEP) [20] and compared its detection effectiveness with OCSVM [21] on the SEA and 1v49 datasets. Emerging Patterns (EPs) are patterns whose support increases from one user command history to another with a big ratio. For classification, the length statistics of EPs extracted from data are used. Obtained results are similar to those of OCSVM [21], though with slightly higher false alarm rates.

3.3 HMM-based Approaches

A Hidden Markov Model (HMM) is a statistical model where the modeled system is assumed to be a Markov process with unobserved state. An HMM is defined in [22] as a doubly stochastic process with an underlying stochastic process that is not observable (it is hidden), but can only be observed through another set of stochastic processes that produce the sequence of observed symbols.

R. Posadas et al. [4] proposed a hybrid approach using session folding and Hidden Markov Models (HMM). It introduced a variation on the SEA dataset, called SEA-I (see Section 2). It applies the **Sequitur** algorithm [16] to extract the grammar from every session to compress them into production rules, which in turn are used as training data for the HMM model. The grammar extraction

process is called *session folding*. The detection results of this method are similar to those of the Uniqueness method [1] on the SEA dataset, but are better on the SEA-I dataset, where it outperforms the other three tested methods, showing its robustness against masquerader attempts based on command frequency due to its grammar extraction features.

An immunity-based approach for masquerader detection was presented in [23] and [24], and later extended in [25]. It consists on multiple agents, each trained to recognize the profile of one user, in a similar way to how organic immunity systems work by distinguishing “self” cells and tissues from “nonself” ones. A Hidden Markov Model is used to construct user profiles, but the approach is independent of the profile construction method. Each testing block gets a score from every agent and, by means of a voting mechanism and a threshold value, the legitimacy of that block is decided. Results were slightly better than those of Uniqueness [1].

3.4 Naïve Bayes-based Approaches

The Naïve Bayes classifier is one of the most popular classifiers in text classification. They are simple, probabilistic classifiers known for their inherent robustness to noise and their fast learning curve. It is generally used with the “bag of words” model, which profiles documents based on word frequencies, ignoring the sequence information [26].

The Naïve Bayes classifier was first applied on Schonlau’s dataset by Roy A. Maxion and Tahlia N. Townsend in 2002 [3] where, additionally, the alternative 1v49 dataset configuration was proposed. The Naïves Bayes approach with online profile updating, using the bag of words model, performed remarkably well and achieved one of the best detection results known to the authors of this paper. This paper was later extended by the same authors in [27] including a thorough error analysis, and in [28] by investigating a possible flaw in the original Naïve Bayes approach which prevented the detection of so-called “super-masqueraders”. Jia and Maxion later found that users who are more successful in Naïve Bayes self-recognition can achieve better results in masquerader detection [29]. Recently, Killourhy and Maxion presented a “fortified” version of the Naïve Bayes method which is able to detect super-masqueraders by adding a detector of never-before-seen commands (NBSCs) [30]. Maxion et al. extended their previous work by applying the Naïves Bayes classifier on Greenberg’s enriched command line data [6]. Furthermore, the block size was reduced from 100 commands to 10 commands, showing that real-time masquerader detection could certainly be possible to achieve.

Yung further extended the original work by Maxion et al. by trying a user profile updating scheme with feedback [31], in which the user is asked for confirmation about the classification of command blocks. The user feedback scheme showed improved results in comparison with those with no feedback, showing that user feedback is a valuable input to any masquerader detection algorithm.

Yung [32] presented an extension of the Naïve Bayes classifier used in [3] called self-consistent Naïve Bayes, which iteratively updates the user profile on

each new block, based on the expectation-maximization algorithm [33]. This technique reduced the missing alarm rate by 40%.

3.5 Sequence-based and Bioinformatics Approaches

The sequential nature of UNIX command-line data lends itself to many different approaches from diverse disciplines. Techniques used in bioinformatics have been applied with promising results. Sequence alignment is a tool commonly used to quantify the similarity between two genetic sequences.

The Sequence Match approach tried by Schonlau et al. in [1] is based on previous work on masquerader detection by T. Lane and C. E. Brodley [7]. It computes a similarity measure between the ten most recent commands and a user’s profile by counting the number of matches in a command-by-command fashion. This method showed good detection for very low false alarm rates (below 1%), but otherwise it did not achieve good results.

Gebski and Wong [34] modeled user sessions as trees where each node represents a command and the edges represent the probabilities between them. This approach was tested on the PU dataset, with good detection rates at the cost of high alarm rates.

Coull et al. [35] used a variation of the Smith-Waterman algorithm [36] for sequence local alignment to compute a semi-global alignment similarity between a user profile and a suspicious block of commands. Their scoring algorithm penalizes the insertion of gaps both into the user signature and the testing block and, of course, rewards matching commands. This method achieved a good hit rate, similar to that of Naïve Bayes, but with a slightly higher false alarm rate. Coull and Szymanski have recently improved this algorithm using a signature updating scheme with binary scoring, achieving good detection rates with a low false alarm rate [37].

3.6 SVM-based Approaches

Support Vector Machines (SVMs) are a set of machine learning algorithms used for classification and regression. SVM classifies data by constructing a separating hyperplane in the n -dimensional feature space of training inputs, which maximizes the margin between them [38].

Wang and Stolfo’s work [21] introduced the application of one-class training for masquerader detection. Their work compared the use of the Naïve Bayes classifier with SVM, in both one-class and multi-class variants, on the SEA dataset. They also compared two common models of data representation: multivariate Bernoulli, or binary; and multinomial model, also known as “bag of words”. The best results were obtained by one-class SVM using binary features and two-class multinomial Naïve Bayes, achieving similar hit and false positive rates as those previously obtained by Schonlau et al. [1].

Chen and Aritsugi [39] tried to reduce the computational cost of the ECM method [19] by rearranging the co-occurrence matrices (as they are generally

sparse), which they called “enhanced Co-occurrence Matrix”, and then applied one- and two-class SVM with updating for classification. Results were similar to previous studies, though false alarms were slightly higher. Two-class SVM produced better results than one-class SVM.

In 2006, Li, Li and Liu [40] also tried to reduce the computational burden of the ECM method [19], as well as reduce overfitting, by extracting the principal features of user behavior from the correlation eigen matrix using Principal Component Analysis (PCA) [18], and feeding these attributes to an SVM classifier, on the SEA dataset. Results are among the best reported so far, but very few details are available about this approach.

Kim and Cha applied SVM with a voting engine on SEA, 1v49 and Greenberg datasets [41]. In the voting scheme, each block of commands is divided into overlapping subblocks which are tested for masquerader activity and a majority voting algorithm decides if the whole block is legitimate or not, according to some threshold. Support Vector Machine (SVM) is used for classification, using a radial based function (RBF) kernel. The most common commands are grouped together (i.e., those used by more than a certain number of users, exceeding a certain percentage) and treated as a single command, which produces a lower false alarm rate than that obtained with no grouping of commands. Results show a slight improvement over Naïve Bayes. Unfortunately, it is not clear how the SVM classifier was trained.

Yang, Zhang and Cai applied one-class SVM with a string kernel on both SEA and PU datasets [42]. The string kernel has a practical advantage in that it can directly process the UNIX command sequences. Few details are given about this work. Results show good detection rates, but very high false positive rates.

3.7 Other Approaches

The Uniqueness approach was first introduced by Schonlau and Martin in [2] and was later compared with other five techniques in [1]. A test statistic is defined on terms of unpopular (ie, used only by a few users) and unique (i.e. used only by one user) commands. Only command frequencies are used; the order in which commands appear is not taken into account. This technique achieved the lower false alarm rate among all of Schonlau et al’s approaches, but at the cost of a low hit rate.

Ming Dong Wan et al. [43], on the other hand, showed that high frequency commands work as well as unique commands

The Bayes One-Step Markov [1] is based on one-step transitions from one command to the next, using a Bayes factor statistic to test the null hypothesis that the observed one-step command transition probabilities are consistent with the historical transition matrix. This approach, using profile updating, achieved relatively poor results.

Schonlau et al.’s Hybrid Multistep Markov method [1] is based on a high-order, or multilevel, Markov chain. In order to limit the high dimensionality of the Markov model, the number of commands is reduced by grouping together the least used commands and treating them as one command, and a mixture

transition distribution approach is used to model the transition probabilities. This approach showed poor results.

Schonlau et al. applied Incremental Probabilistic Action Modeling (IPAM) [1] on the SEA dataset. This algorithm is based on the one-step command transition probabilities, which are updated continually using an exponential updating scheme. A command is labeled as suspicious if it is not one of the four most probable commands to follow the previous one, according to its transition matrix. Few details are given about this technique. Its results were not particularly promising.

In 2004 Wang et al. introduced an algorithm called Non-negative Matrix Factorization (NMF) [44], which produced good detection results. Mex-Perera et al. later improved [45] the original NMF approach by using a data normalization phase which takes into account the least used commands, assigning them a higher weight.

In 2007 J. Beauquier and Y. Hu [46] tried an intrusion detection model based on the combination of known methods, called Pearson Correlation Coefficients-Rank (PCC-R) Combination on the SEA dataset. They use a notion of distance on different methods, and combine them by applying the Pearson correlation coefficients. The combined methods are: Uniqueness [1], Bayes one-step Markov [1], Naïve Bayes and Probabilistic Finite State Automata (PFSA). For each user, two types of detection thresholds are defined: one for each method and one for their combination. This approach showed improved hit rates (ie, lower missing alarms) compared with previous works, though the false alarm rates were not improved.

4 Evaluation

We review the most popular measures for the evaluation of detection effectiveness. This permits the comparison of different masquerader detection algorithms. Many other measures have been applied in literature, but not as frequently as these, which in most cases makes the comparison of results very hard.

4.1 Misses and False Alarms

A *miss*, *missing alarm* or false negative happens when a masquerader block is classified as legitimate. A *false alarm* or false positive occurs when the system regards a legitimate user as a masquerader. The percentage of missing alarms, or *Missing Alarm Rate*, and the percentage of false alarms, or *False Alarm Rate* (FAR) are the most popular measures of effectiveness in masquerader detection. It is generally accepted that a low false alarm rate is desirable. Schonlau et al. [1] proposed a target false alarm rate of 1%. Sometimes the Hit Rate is used instead of the Missing Alarm Rate. The Hit Rate and the Missing Alarm Rate are complementary percentages, so that $Hits = 100 - Misses$.

4.2 ROC Curves

Receiver operating characteristic curves, or ROC curves, are often utilized to show the trade-off between the missing alarm rate and the false alarm rate, by varying the masquerader detection threshold. They can be plotted in different ways such as Missing Alarm Rates vs. False Alarm rates or Hits vs. False Alarm Rates. ROC curves permit the visual comparison of different techniques applied on the same dataset. The area under the ROC curve (AUC) is a measure to quantitatively compare different ROC curves which, surprisingly, is seldom applied on masquerader detection literature.

4.3 Cost

Maxion et al. proposed a cost function [3] to assess the effectiveness of an intrusion detection scheme in terms of a linear combination of missing alarms and false alarms. False alarms are prioritized as there is a general agreement that a false alarm should be more expensive than a missing alarm. Of course, the lower the Cost function, the better the effectiveness of an algorithm.

$$Cost = Misses + 6 \times FalseAlarms \quad (1)$$

5 Detection Effectiveness Comparison

Table 5 shows a comparison of results achieved by the reviewed masquerader detection approaches. Results are expressed in terms of Hits, False Alarm Rates (FAR), and Maxion et al.’s Cost function (1). Methods are subdivided according to the dataset they were applied on, as a direct comparison is not possible between them. Results are showed in increasing order according to their Cost function, as it is capable to combine Hits and Misses in one result.

Overall, the most effective approaches seem to be those based on SVM or Naïve Bayes. The combination of different methods (as in [46]) also achieved very good results. On the other hand, the use of enriched data shows significant improvements in all cases. This can be clearly seen in the Greenberg dataset subsection in Table 5. Also, methods which were tested with and without user profile updating show an improvement when applying updating.

6 Conclusions

We have reviewed and compared different aspects of masquerader detection, including: datasets, classification algorithms, effectiveness measures and detection results. Their diversity makes comparisons hard or even impossible to achieve. A unified and coherent framework for testing and comparing different algorithms in a consistent, reliable and reproducible way could benefit the entire community

⁷ Results manually extracted from published graphs

Table 2. Comparison of detection results achieved by different approaches

SEA dataset [1]	Hits	FAR	Cost(1)	Config.	Upd.
NCD (bzip) [10]	30.4	0.6	73.1	SEA	No
Compression [1]	34.2	5.0	95.8	SEA	Yes
Customized Grammars [15] ⁷	65.0	1.0	41.0	SEA	Yes
ECM with Layered Networks [19]	72.3	2.5	42.7	SEA	No
OCLEP [20]	72.59	4.9	56.81	SEA	No
OCLEP [20]	46.63	5.14	84.21	1v49	No
ECM [17]	37	4.5	90.0	SEA	Yes
Immunity-based HMM [23] ⁷	60.0	1.0	46.0	SEA	No
Hybrid Grammar HMM (SEA) [4] ⁷	45.0	2.0	67.0	SEA	No
Hybrid Grammar HMM (SEA-I) [4] ⁷	25.0	3.0	93.0	SEA-I	No
Self-consistent Naïve Bayes [32] ⁷	79.0	2.0	33.0	SEA	Yes
Naïve Bayes updating w/ feedback [31] ⁷	76.0	2.0	36.0	SEA	Yes
Naïve Bayes [21] ⁷	70.0	2.0	42.0	SEA	No
Naïve Bayes [3]	61.5	1.3	46.3	SEA	Yes
Naïve Bayes [3]	66.2	4.6	61.4	SEA	No
Naïve Bayes [3]	62.8	4.6	64.8	1v49	Yes
Seq. Alignment (upd. binary scoring) [37]	68.6	1.9	42.8	SEA	Yes
Seq. Alignment [35]	75.8	7.7	70.4	SEA	Yes
Sequence-Matching [1]	36.8	3.7	85.4	SEA	Yes
SVM [41]	94.8	0	5.2	1v49	No
CEM-SVM with PCA [40]	82.6	3.0	35.4	SEA	No
SVM+ECM two-class [39]	72.24	3.00	45.7	SEA	Yes
OCSVM with String Kernel [42]	62.0	1.5	47	SEA	No
OCSVM [21] ⁷	70.0	4.0	54	SEA	No
SVM+ECM one-class [39]	62.77	6.00	73.2	SEA	Yes
SVM [41]	80.1	9.7	78.1	SEA	No
PCC-R combination [46]	78.9	3.2	40.3	SEA	Yes
Improved NMF [45]	42.0	1.0	48.0	SEA	No
PCC-R combination [46]	81.3	4.9	48.1	SEA	No
NMF [44]	61.1	3.9	62.3	SEA	No
Uniqueness [1]	39.4	1.4	69.0	SEA	Yes
Hybrid multistep Markov [1]	49.3	3.2	69.9	SEA	Yes
Bayes 1-Step Markov [1]	69.3	6.7	70.9	SEA	Yes
IPAM [1]	41.1	2.7	75.1	SEA	Yes
High Freq. Commands (1 st order) [43]	70.9	9.20	84.2	SEA	No
High Freq. Commands (1 st order) [43]	71.0	13.4	109.4	SEA	Yes
GREENBERG dataset [5]	Hits	FAR	Cost(1)	Config.	Upd.
NCD(bzip) [13]	60.0	1.0	46.0	enriched	No
SVM [41]	87.3	6.4	51.1	enriched	No
Naïve Bayes [6]	82.1	5.7	52.1	enriched	Yes
Naïve Bayes [6]	70.9	4.7	57.3	truncated	Yes
SVM [41]	71.1	6.0	64.9	truncated	No
NCD(bzip) [13]	38.0	1.0	68.0	truncated	No
PU dataset [7]	Hits	FAR	Cost(1)	Config.	Upd.
OCSVM with String Kernel [42]	60.0	2.0	52.0	-	No
Tree-based Model [34]	85.0	10.0	75.0	-	No

of intruder detection research. This paper tried to show the most complete comparison of masquerader detection methods to date, which we believe may be a first step in that direction.

Though often mentioned as a possible way for future work in literature, little research has been performed on the combination of different masquerader detection techniques and even different datasets, such as key-press timings for keystroke dynamics analysis, network activity logs for event correlation or sequences of system calls, etc.

On the other hand, the lack of real masquerader datasets prevents a thorough analysis on the behavior of intruders and forces researchers to make assumptions about their ways of attacking. A publicly available dataset including these features could be valuable for the improvement of intrusion detection techniques.

7 Acknowledgements

This work is part of a more extendend research project which is supported in part by ANPCyT (National Agency of Scientific Promotion and Technology) under the grant PICTO CITEFA 2004 18623. We gratefully acknowledge financial support from ANPCyT. We would also like to thank all members of CITEFA's Si6 Lab and the peer reviewers for their advice and support.

References

1. Schonlau, M., Dumouchel, W., hua Ju, W., Karr, A.F., Theus, M., Vardi, Y.: Computer intrusion: detecting masquerades. *Statistical Science* **16** (2001) 58–74
2. Schonlau, M., Theus, M.: Detecting masquerades in intrusion detection based on unpopular commands. *Inf. Process. Lett.* **76**(1-2) (2000) 33–38
3. Maxion, R.A., Townsend, T.N.: Masquerade detection using truncated command lines. In: *DSN '02: Proceedings of the 2002 International Conference on Dependable Systems and Networks*, Washington, DC, USA, IEEE Computer Society (2002) 219–228
4. Posadas, R., Mex-Perera, J.C., Monroy, R., Nolasco-Flores, J.A.: Hybrid method for detecting masqueraders using session folding and hidden markov models. In Gelbukh, A.F., García, C.A.R., eds.: *MICAI*. Volume 4293 of *Lecture Notes in Computer Science.*, Springer (2006) 622–631
5. Greenberg, S.: Using unix: Collected traces of 168 users. Research Report 88/333/45, Department of Computer Science, University of Calgary, Alberta, Canada (1988)
6. Maxion, R.A.: Masquerade detection using enriched command lines. *dsn* **00** (2003) 5
7. Lane, T., Brodley, C.E.: An application of machine learning to anomaly detection. In: *Proceedings of the 20th National Information Systems Security Conference*. (1997) 366–380
8. Chinchani, R., Muthukrishnan, A., Chandrasekaran, M., Upadhyaya, S.: RACOON: Rapidly Generating User Command Data for Anomaly Detection from Customizable Templates. In: *Proceedings of the 20th Annual Computer Security Applications Conference (ACSAC '04)*, Tucson, Arizona (2004)

9. Welch, T.A.: A technique for high-performance data compression. *IEEE Computer* **17**(6) (1984) 8–19
10. Bertacchini, M., Fierens, P.I.: Preliminary results on masquerader detection using compression based similarity metrics. *Electronic Journal of SADIO* **7**(1) (2007)
11. Cilibrasi, R., Vitányi, P.: Clustering By Compression. *IEEE Transactions on Information Theory* **51**(4) (2005) 1523–1545
12. Li, M., Vitányi, P.: An Introduction to Kolmogorov Complexity and Its Applications. Second edn. GTCS. pub-SV (1997)
13. Bertacchini, M., Benitez, C.: NCD based masquerader detection using enriched command lines. In: Proc. of the IV Congreso Iberoamericano de Seguridad Informatica (CIBSI'07), Mar del Plata, Argentina (2007) 329–338
14. Evans, S., Eiland, E., Markham, S., Impson, J., Laczo, A.: Mdlcompress for intrusion detection: Signature inference and masquerade attack. Military Communications Conference, 2007. MILCOM 2007. IEEE (2007) 1–7
15. Latendresse, M.: Masquerade detection via customized grammars. In Julisch, K., Krügel, C., eds.: DIMVA. Volume 3548 of Lecture Notes in Computer Science., Springer (2005) 141–159
16. Nevill-Manning, C.G., Witten, I.H.: Identifying hierarchical structure in sequences: a linear-time algorithm. *Journal of Artificial Intelligence Research* **7** (1997) 67–82
17. Oka, M., Oyama, Y., Kato, K.: Eigen co-occurrence matrix method for masquerade detection (2004)
18. Turk, M., Pentland, A.: Eigenfaces for recognition. *J. Cognitive Neuroscience* **3**(1) (1991) 71–86
19. Oka, M., Oyama, Y., Abe, H., Kato, K.: Anomaly detection using layered networks based on eigen co-occurrence matrix. In Jonsson, E., Valdes, A., Almgren, M., eds.: RAID. Volume 3224 of Lecture Notes in Computer Science., Springer (2004) 223–237
20. Chen, L., Dong, G.: Masquerader detection using oclep: One-class classification using length statistics of emerging patterns. In: Web-Age Information Management Workshops, 2006. WAIM '06. Seventh International Conference on. (2006) 5
21. Wang, K., Stolfo, S.J.: One-class training for masquerade detection (2003)
22. Böckler, S.B., Bateman, A.: An introduction to hidden markov models. Current protocols in bioinformatics / editorial board, Andreas D. Baxevis ... [et al.] **Appendix 3** (2007)
23. Okamoto, T., Watanabe, T., Ishida, Y.: Towards an immunity-based system for detecting masqueraders. In: KES. (2003) 488–495
24. Okamoto, T., Watanabe, T., Ishida, Y.: Mechanism for generating immunity-based agents that detect masqueraders. In: KES. (2004) 534–540
25. Okamoto, T., Ishida, Y.: Framework of an immunity-based anomaly detection system for user behavior. In Apolloni, B., Howlett, R.J., Jain, L.C., eds.: KES (3). Volume 4694 of Lecture Notes in Computer Science., Springer (2007) 821–829
26. McCallum, A., Nigam, K.: A comparison of event models for naive bayes text classification (1998)
27. Maxion, R., Townsend, T.: Masquerade detection augmented with error analysis. Reliability, *IEEE Transactions on* **53**(1) (2004) 124–147
28. Killhourhy, K.S., Maxion, R.A.: Investigating a possible flaw in a masquerade detection system. Technical Report 869, Newcastle University, School of Computing Science (2004)
29. Jia, P., Maxion, R.A.: Detect masqueraders using unix command sequences. Lab Report, Center for Automated Learning and Discovery (2005)

30. Killourhy, K.S., Maxion, R.A.: Learning from a flaw in a naive-bayes masquerade detector (2007)
31. Yung, K.H.: Using feedback to improve masquerade detection. In: ACNS. (2003) 48–62
32. Yung, K.H.: Using self-consistent naive-bayes to detect masquerades. In: PAKDD. (2004) 329–340
33. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)* **39**(1) (1977) 1–38
34. Gebski, M., Wong, R.K.: Intrusion detection via analysis and modelling of user commands. In Tjoa, A.M., Trujillo, J., eds.: DaWaK. Volume 3589 of *Lecture Notes in Computer Science.*, Springer (2005) 388–397
35. Coull, S., Branch, J., Szymanski, B., Breimer, E.: Intrusion detection: a bioinformatics approach. *Computer Security Applications Conference, 2003. Proceedings. 19th Annual (2003)* 24–33
36. Wagner, R.A., Fischer, M.J.: The string-to-string correction problem. *J. ACM* **21**(1) (1974) 168–173
37. Coull, S.E., Szymanski, B.K.: Sequence alignment for masquerade detection. *Comput. Stat. Data Anal.* **52**(8) (2008) 4116–4131
38. Burges, C.J.C.: A tutorial on support vector machines for pattern recognition. *Data Min. Knowl. Discov.* **2**(2) (1998) 121–167
39. Chen, L., Aritsugi, M.: An svm-based masquerade detection method with online update using co-occurrence matrix. In Büschkes, R., Laskov, P., eds.: DIMVA. Volume 4064 of *Lecture Notes in Computer Science.*, Springer (2006) 37–53
40. Li, Z., Li, Z., Liu, B.: Masquerade detection system based on correlation eigen matrix and support vector machine. *Computational Intelligence and Security, 2006 International Conference on* **1** (2006) 625–628
41. Kim, H.S., Cha, S.D.: Empirical evaluation of svm-based masquerade detection using unix commands. *Computers and Security* **24**(2) (2005) 160–168
42. Yang, M., Zhang, H., Cai, H.: Masquerade detection using string kernels. *Wireless Communications, Networking and Mobile Computing, 2007. WiCom 2007. International Conference on (2007)* 3681–3684
43. Wan, M.D., Wu, H.C., Kuo, Y.W., Marshall, J., Huang, S.H.: Detecting masqueraders using high frequency commands as signatures. *Advanced Information Networking and Applications - Workshops, 2008. AINAW 2008. 22nd International Conference on (2008)* 596–601
44. Wang, W., Guan, X., Zhang, X.: Profiling program and user behaviors for anomaly intrusion detection based on non-negative matrix factorization. *43rd IEEE Conference on Decision and Control (2004)*
45. Mex-perera, C., Posadas, R., Nolasco, J.A., Monroy, R., Soberanes, A., Trejo, L.: An improved non-negative matrix factorization method for masquerade detection. In: *Proceedings of the 1st Mexican International Conference on Informatics Security, MCIS 2006, Mexico (2006)*
46. Beauquier, J., Hu, Y.J.: Intrusion detection based on distance combination. In: *CESSE07, Venice, Italy, World Academy of Sciences, WAS (2007)*

Propostas para apoiar a preservação documental de longo prazo na ICP-Brasil

Rafael Timóteo de Sousa Jr¹, Ricardo Felipe Custódio², Viviane Bertol¹

¹ Universidade de Brasília, Faculdade de Tecnologia, Departamento de Engenharia Elétrica (ENE) - *Campus* Universitário Darcy Ribeiro, Brasília – BRASIL - CEP 70910-900 - Fax: +55 61 3274 6651 Telefones: +55 61 3307 2308 e 3273 5977

² Universidade Federal de Santa Catarina, Centro Tecnológico, Departamento de Informática e Estatística - *Campus* Universitário Cx.P. 476 - Florianópolis – SC - BRASIL - CEP 88040-900 Fax: +55 048 3721-7553 Telefone: +55 048 3721-9498
desousa@unb.br, custodio@inf.ufsc.br, bertol_viviane@yahoo.com

Resumo. Documentos eletrônicos estão sendo cada vez mais utilizados como substitutos para documentos em papel. A essa tecnologia somam-se as assinaturas eletrônicas – dentre as quais encontram-se as assinaturas digitais - que permitem conferir autenticidade e/ou autoria a um documento eletrônico. Uma importante questão que surge em decorrência da utilização de documentos eletrônicos assinados digitalmente é como preservá-los adequadamente no longo prazo. Neste artigo tratamos da preservação desse tipo de documentos, dando ênfase à situação na ICP-Brasil, a infraestrutura de chaves públicas criada pelo Governo brasileiro em 2001. Propomos diversas medidas com o objetivo de prover segurança aos processos de geração, guarda e recuperação de documentos eletrônicos assinados digitalmente no âmbito da ICP-Brasil.

Palavras-chave: ICP-Brasil, Assinatura Digital, Documento Eletrônico, Preservação a longo prazo

1 INTRODUÇÃO

Os documentos eletrônicos estão sendo cada vez mais utilizados pela sociedade como substitutos para os documentos em papel. Outra tecnologia que se soma ao documento eletrônico é a assinatura eletrônica, que permite assinar um documento eletrônico conferindo-lhe autenticidade ou autoria. Uma assinatura eletrônica pode ser obtida por meio de diversos dispositivos ou sistemas, como login/senha, biometria, impostação de PIN etc. Um dos tipos de assinatura eletrônica é a assinatura digital, que utiliza pares de chaves criptográficas associados a certificados digitais.

Essa situação levanta questões diversas, sendo uma delas: como preservar adequadamente um documento eletrônico assinado digitalmente, para que esteja disponível aos usuários legítimos no futuro? Estudos sobre esse tema estão sendo realizados por diversas áreas de pesquisa, sendo que muitos deles já sendo aplicados em projetos experimentais, porém não existe, ainda, um consenso sobre a melhor forma de se preservar esses documentos.

Neste artigo trata-se da preservação de documentos eletrônicos assinados digitalmente e dá-se ênfase às questões relacionadas com a preservação de documentos eletrônicos assinados com a utilização de chaves criptográficas privadas associadas a certificados digitais emitidos no âmbito da ICP-Brasil.

A ICP-Brasil é uma infraestrutura de chaves públicas criada pelo Governo Brasileiro em 2001, por meio da Medida Provisória 2.200-2, para emissão de certificados digitais que permitem aos seus titulares criar assinaturas digitais que possuam validade jurídica semelhante a de assinaturas de punho, perante a legislação brasileira.

Documentos assinados digitalmente no âmbito da ICP-Brasil estão sendo amplamente utilizados em inúmeras áreas de importância para o Brasil. Por esse motivo, a segurança desses documentos e a possibilidade de sua validação e recuperação no longo prazo são questões relevantes para o País.

O objetivo deste trabalho é analisar os processos utilizados atualmente para geração, arquivamento e recuperação de documentos assinados digitalmente no âmbito da ICP-Brasil e sugerir melhorias que facilitem sua preservação e validação segura no longo prazo.

Na sessão 2 faz-se um breve apanhado dos trabalhos científicos realizados sobre preservação documental e sobre preservação de documentos assinados digitalmente. Na sessão 3 propõem-se medidas, que agregariam segurança e aumentariam a disponibilidade dos documentos assinados no âmbito da ICP-Brasil. Essas medidas são analisadas na sessão 4 e as conclusões relatadas na sessão 5.

2 PRESERVAÇÃO DE DOCUMENTOS ASSINADOS DIGITALMENTE

2.1 Preservação de Documentos Eletrônicos

Segundo Thomaz [1], preservar informação em formato digital é mais difícil que preservar informação em formatos tradicionais, pois:

- a informação digital está cada vez mais distribuída;
- dados e metadados devem ser migrados para novas mídias, sistemas operacionais e sistemas aplicativos;
- representações e formatos devem obrigatoriamente acompanhar novas tecnologias e padrões;
- a informação digital precisa ser prontamente transportada de arquivo a arquivo.
- Uma das primeiras instituições a se preocupar com esse assunto foi o *National Space Science Data Center*, da NASA, que criou um comitê consultivo para preservação de dados espaciais, composto por um grupo internacional de agências espaciais.

Esse comitê desenvolveu diversos padrões independentes de área científica e se tornou, por volta de 1990, um grupo de trabalho do ISO TC 20/ SC 13.

A tarefa inicial foi o desenvolvimento de um modelo de referência para estabelecer termos e conceitos comuns. A seguir, o grupo definiu técnicas formais para especificação, como Diagramas de Fluxo de Dados para modelos funcionais e interfaces, e *Unified Modeling Language* (UML) para classes de dados. Por fim, foram desenvolvidos novos padrões para tratamento de arquivos.

O trabalho final, chamado de modelo de referência OAIS (*Open Archive Information System*) foi aprovado para publicação como padrão ISO 14721:2003 [2] em Fevereiro de 2003.

A ISO 14721:2003 especifica um modelo de referência para um sistema aberto de arquivamento de informação. O objetivo da ISO 14721:2003 é estabelecer um sistema para arquivar a informação, tanto digital quanto físico, com uma estrutura de organização composta de pessoas que aceitam a responsabilidade de preservar a informação e de torná-la disponível a um dado agrupamento de usuários.

Suas diretrizes aplicam-se a quaisquer tipos de organizações, mas principalmente para organizações com responsabilidade de tornar informação disponível por longo prazo como Arquivos, Bibliotecas e Centros de Pesquisa. Interessa, também, àquelas organizações e indivíduos que criam informação que pode vir a necessitar de preservação por longo prazo e àquelas que necessitam obter informação desses arquivos.

Outra iniciativa importante vem sendo conduzida pelo Public Record Office Victoria (PROV) da Austrália. Trata-se do projeto The Victorian Electronic Records Strategy (VERS), que objetiva fornecer uma solução de baixo custo para o problema de capturar, gerenciar e preservar registros eletrônicos no longo prazo. É um conjunto de padrões, guias, treinamentos, consultoria e projetos de implementação, centrado no objetivo de conferir confiabilidade e autenticidade aos registros eletrônicos arquivados. [3].

2.2 Preservação de Documentos Eletrônicos Assinados Digitalmente

Se já é um desafio armazenar documentos eletrônicos, um desafio ainda maior surge quando esses dados são assinados digitalmente. Isso ocorre porque a verificação da validade de uma assinatura digital depende da existência de diversas informações, além do próprio documento assinado. Entre essas informações estão:

- os certificados das autoridades certificadoras que compõem a cadeia de certificação
- as listas de certificados revogados ou as informações sobre revogação que foram obtidas quando da geração ou verificação da assinatura.

Além disso, muitas vezes é necessário que uma assinatura digital seja verificada muito tempo depois do prazo de validade dos certificados e dos carimbos de tempo que porventura tenham sido utilizados no documento eletrônico. Paralelamente, novas tecnologias podem ter surgido nesse intervalo de tempo, fragilizando os mecanismos criptográficos utilizados na geração da assinatura. Tudo isso implica a adoção de medidas adicionais, visando preservar a integridade da assinatura, tais como reassinatura, imposição de carimbos do tempo etc.

No meio acadêmico internacional, diversos autores já estudam o problema, como Ansper, Buldas e Willemson, que tratam da especificamente da validação de assinaturas a longo prazo em seu artigo [4]. Propõem a utilização de protocolo baseado em cadeias de *hashes*, que prescinde da utilização de notários ou terceiras partes confiáveis para garantir a validade das assinaturas.

No artigo [5], os autores Gritzalis e Lekkas questionam a utilização de cadeias de *hashes*, dada a dificuldade em fazer-se a conferência, num futuro distante, de todas as assinaturas apostas pelas autoridades de carimbo tempo, dado que essas podem nem mais existir, ou não ser mais confiáveis, na época da validação da assinatura. Além disso, citam as dificuldades que podem ocorrer em função das diferentes tecnologias e equipamentos usados ao longo do tempo, para geração e validação dos carimbos e assinaturas. Sugerem um esquema de Notarização Cumulativa, que, apesar de facilitar a validação das assinaturas, traz implícita a necessidade de que haja confiança nos notários por parte dos usuários da ICP.

No artigo [6], os autores Maniatis e Baker propõem, ao invés do uso de notarização, a utilização de mecanismos de carimbo do tempo combinados com um Serviço de Arquivamento de Chaves. Esse serviço guardaria em repositório seguro as chaves das Autoridades Certificadoras da ICP. Periodicamente, seus arquivos seriam carimbados por uma Autoridade de Carimbo de Tempo, como forma de provar aos usuários a existência dos certificados da AC, em dado instante de tempo. Com isso, entendem que ficaria assegurada a validação dos certificados emitidos, com a preservação da cadeia de certificação. Poderiam também ser arquivados nesse repositório seguro documentos, certificados digitais, trilhas de auditoria e demais arquivos com dados de grande valor.

O PKIX também tem buscado soluções para o problema. A RFC4810 *Long-Term Archive Service Requirements* [7] descreve uma classe de serviços de arquivamento para suportar a verificação de documentos assinados digitalmente a longo prazo e os requisitos técnicos para interagir com esse serviço. A RFC4998 *Evidence Record Syntax (ERS)* [8] especifica a sintaxe e o processamento de um Registro de Prova, que é uma estrutura destinada a apoiar por longo prazo o não-repúdio de dados existentes permitindo aos usuários provar a existência e a integridade dos dados, inclusive os assinados digitalmente.

No mercado, já existem soluções que utilizam processos de preservação que cuidam tanto da integridade como da autenticidade dos documentos digitais, fornecendo provas da autenticidade do documento. Porekar [9] examina essas soluções para determinar as melhores práticas de arquivamento no longo prazo e as captura de uma forma mais abstrata, num nível genérico, fornecendo um conjunto de padrões de segurança para arquivamento a longo prazo. Classifica esses padrões em 4 categorias:

- Carimbos do tempo emitidos por autoridades de carimbo do tempo confiáveis;
- Concatenação de valores de *hashes* criptográficos;
- Apresentação de evidências para grupos de documentos digitais;
- Cadeias de carimbos do tempo de arquivamento.

A questão dos requisitos a serem observados pelos serviços de arquivamento de longo prazo também é analisada em [10], que propõe a criação de uma classe de

serviços que satisfaça os requisitos de preservação das características dos objetos de dados, interação entre os objetos arquivados, gerenciamento dos objetos arquivados, manutenção de sua integridade e evidências, política de arquivamento, agrupamento de objetos, transferência de dados e confidencialidade dos dados.

Por outro lado, a comunidade arquivística olha com certa desconfiança para a tecnologia de assinatura digital. Nos dizeres de Boudret [11]:

“...fica evidente que, a partir de um ponto de vista puramente arquivístico, as assinaturas digitais não são um método adequado para provar a autenticidade e integridade dos registros eletrônicos. Na área de autenticidade, as assinaturas digitais não dizem nada sobre a identidade dos registros eletrônicos. A declaração de autenticação nada mais é do que uma prova de que o documento foi assinado por algo / alguém na posse da chave privada do signatário. No que diz respeito à integridade, uma assinatura digital só prova que os bits transmitidos do documento estão intactos. As assinaturas digitais não impedem corrupções ou manipulações não permitidas.”

Blanchette [12] define a dicotomia entre as abordagens técnica e arquivística como *“um choque entre dois conceitos diferentes de autenticidade eletrônica”*. O primeiro conceito, abraçado pela comunidade técnica e por segmentos da comunidade legal, baseia-se na medição de uma propriedade física do documento: a integridade dos bits. Um documento é considerado íntegro se os bits que o compõem estão íntegros. O segundo conceito, abraçado pela comunidade arquivística, é de que a integridade de um documento pode ser melhor atestada analisando-se a integridade do contexto que envolve o documento, ou seja *“a totalidade dos controles e procedimentos, sejam humanos ou baseados em computadores, que asseguram a identidade e integridade de um registro eletrônico através de todo seu ciclo de vida.”*

A solução para essa dicotomia está sendo buscada a partir de trabalhos conjuntos das comunidades envolvidas. Diversas iniciativas abrangentes vêm sendo realizadas. Uma das mais conhecidas é o InterPARES, acrônimo para International Research on Permanent Authentic Records in Electronic Systems, que visa a desenvolver o conhecimento essencial para a preservação a longo prazo de registros autênticos criados e/ou mantidos em forma digital e a prover as bases para padrões, políticas, estratégias e planos de ação capazes de assegurar a longevidade desse material e habilitar seus usuários a confiar em sua autenticidade.

Segundo Dumortier [13], que analisou a questão da preservação documental no ambiente da Comunidade Europeia, é necessário desenvolver padrões baseados nas melhores práticas e criar regulamentações que sirvam como uma janela legal mínima, de forma a trazer confiabilidade e estabilidade aos usuários de assinaturas digitais. Deve-se também reduzir a necessidade de migração usando padrões abertos para formatos de documentos (ex.: XML) e estimular a criação de serviços de arquivamento confiáveis, possivelmente separando serviços de arquivamento “normal” dos serviços de arquivamento “com guarda de assinatura”.

A aplicação desses princípios já pode ser observada na Comunidade Europeia que, premida pela necessidade de arquivar os documentos relativos a faturas eletrônicas assinadas digitalmente, emitidas em diferentes países do bloco, criou em 2007 o padrão *ETSI TS 102573 Policy requirements for trust service providers signing and/or storing data for digital accounting* [14], que especifica os requisitos aplicáveis

aos Prestadores de Serviços de Confiança que assinam documentos fiscais eletrônicos relevantes e/ou os arquivam, em nome da pessoa taxada. Esse documento regulamenta a produção e a guarda de forma confiável dos documentos fiscais, mesmo por mais de dez anos. Admite para isso a utilização de medidas técnicas ou organizacionais, desde que permitam preservar o documento de forma adequada.

2.3 Situação na ICP-Brasil

A preservação de documentos eletrônicos no Brasil vem sendo objeto de atenção de diversas entidades, entre as quais o Arquivo Nacional, que está conduzindo estudos sobre criação, manutenção e recuperação de arquivos a longo prazo, com base no modelo OAIS.

Em 2003 foi criado, pelo Decreto 4915/2003, o Sistema de Gestão de Documentos de Arquivo - SIGA, pelo qual se organizam, sob a forma de sistema, as atividades de gestão de documentos de arquivo no âmbito dos órgãos e entidades da administração pública federal.

Em abril de 2007 foi publicada a Resolução 25, do CONARQ [15], que dispõe sobre a adoção do Modelo de Requisitos para Sistemas Informatizados de Gestão Arquivística de Documentos (e-ARQ Brasil) pelos órgãos e entidades integrantes do Sistema Nacional de Arquivos - SINAR. Esse modelo orienta os responsáveis pela especificação, desenvolvimento e utilização de sistemas que tratam de preservação de informações eletrônicas, com relação aos procedimentos e operações técnicas para gestão dessas informações.

No meio acadêmico brasileiro há algumas entidades trabalhando nesse tema, como o Labsec, da Universidade Federal de Santa Catarina, que há vários anos realiza pesquisas sobre esse assunto, como se pode ver pelas publicações no sítio <http://www.labsec.ufsc.br>.

Quanto à preservação de documentos eletrônicos assinados digitalmente no âmbito da ICP-Brasil, o assunto somente foi regulamentado em 2009, com a publicação da Resolução 62 do Comitê Gestor [16]. Esse regulamento, todavia, tem por foco principal o formato das assinaturas, deixando em aberto algumas questões que podem comprometer o objetivo de propiciar segurança e disponibilidade no longo prazo aos documentos assinados digitalmente.

Tendo em vista que os certificados ICP-Brasil podem ser utilizados para assinar digitalmente documentos eletrônicos com validade jurídica, todos os processos que envolvem o ciclo de vida do certificado digital e da assinatura digital devem ser realizados com confiabilidade e segurança, contemplando-se, inclusive, requisitos de rastreabilidade e auditoria dos eventos crítico, para possibilitar a realização de perícias técnicas, em caso de eventuais disputas judiciais envolvendo a assinatura.

Na ICP-Brasil, os riscos inerentes aos processos relacionados ao ciclo de vida do certificado digital (Solicitação, Validação, Emissão e Revogação) estão suficientemente mapeados e controlados por meio de mecanismos de segurança física e lógica implantados nas entidades que executam os processos. Esses controles são checados por meio de auditorias anuais e fiscalizações realizadas pelo Governo Brasileiro.

Já o ciclo de vida de uma assinatura digital compreende os processos:

1. Criação - processo de criação de um resumo criptográfico logicamente associado a um conteúdo digital e a chave criptográfica privada do signatário;
2. Verificação Inicial - processo de verificação quanto à validade de uma ou mais assinaturas digitais logicamente associadas a um conteúdo digital;
3. Armazenamento – processo que trata da guarda da assinatura digital. Compreende, pelo menos, cuidados para conversão dos dados para mídias mais atuais, sempre que necessário;
4. Revalidação – processo que estende a validade do documento assinado, por meio da reassinatura dos documentos ou da aposição de carimbos do tempo, quando da expiração ou revogação dos certificados utilizados para gerar ou revalidar as assinaturas, ou ainda quando do enfraquecimento dos algoritmos ou tamanhos de chave utilizados;
5. Verificação Final – processo no qual o documento é apresentado como evidência para um juiz e a assinatura é novamente verificada, para se obter informações sobre a identidade do signatário e a integridade do documento.

Esses processos acontecem, em geral, em ambientes que não são auditados ou fiscalizados pela ICP-Brasil; entretanto os riscos relacionados às etapas de Criação, Verificação Inicial e Verificação Final podem ser minimizados com a utilização de sistemas e equipamentos homologados no âmbito da ICP-Brasil, em consonância com o regulamento constante da Resolução 36 do Comitê Gestor [17].

Já os processos de Armazenamento dos documentos assinados e de Revalidação das assinaturas digitais não estão regulamentados. Além disso, permanecem em aberto, nos regulamentos atuais, questões como:

- formato dos documentos eletrônicos a serem assinados;
- tratamento a ser dado aos documentos assinados digitalmente antes da publicação da Resolução 62;
- disponibilização de informações críticas, como versões anteriores de LCRs e de Políticas de Certificados, necessárias para validação da cadeia de certificação no longo prazo.

Para sanar essas lacunas, criaram-se as propostas apresentadas no capítulo seguinte.

3 PROPOSTA

Analisando os regulamentos da ICP-Brasil disponíveis no sítio www.iti.gov.br e tomando por base os trabalhos e experiências relatados anteriormente, foram encontradas lacunas na regulamentação de alguns processos executados na ICP-Brasil que podem dificultar a validação das assinaturas digitais de forma segura, a longo prazo.

Essas lacunas ensejaram a criação de propostas de melhoria na regulamentação brasileira, apresentadas a seguir.

3.1 Regularizar o serviço de arquivamento

Nos regulamentos da ICP-Brasil não está prevista a criação de entidades voltadas ao armazenamento de documentos assinados digitalmente. Com isso, corre-se o risco de que eles sejam armazenados de maneira inapropriada ou insegura, o que pode colocar em xeque sua confiabilidade.

Uma medida importantíssima, então, para permitir a validação de documentos a longo prazo na ICP-Brasil seria a criação de entidades para atuar como Prestadores de Serviços de Arquivamento de Longo Prazo (PSA). O Comitê Gestor da ICP-Brasil regulamentaria os procedimentos operacionais, a segurança das suas instalações e sistemas e os requisitos de contratação de pessoal.

Nesse processo, seria importante o envolvimento do Arquivo Nacional, que poderia contribuir na elaboração das políticas e práticas arquivísticas a serem observadas por aquelas entidades, incluindo-se nessa discussão a definição de formatos aceitáveis para os documentos eletrônicos e as medidas a serem adotadas quando da caducidade das assinaturas (revalidação).

Não se propõe aqui a implantação de uma entidade única, centralizada, que armazene todos os documentos assinados no âmbito da ICP-Brasil, que já passam de milhões. A utilização dos serviços do PSA seria facultativa: grandes usuários, que tenham milhares de documentos para armazenar, poderiam criar em seu próprio ambiente as condições para isso, desde que atendessem os requisitos para arquivamento definidos. Usuários menores, ou todos aqueles que não desejassem despendar recursos com a criação de ambiente próprio, utilizariam os serviços de um PSA. O importante é que sejam observados procedimentos regulamentados para armazenamento e revalidação de assinaturas. Essa medida vem ao encontro da proposta de preservação documental explanada em [18].

3.2 Criar repositório seguro de informações críticas

Para permitir a verificação de assinaturas digitais a longo prazo é preciso reconstruir de forma correta o caminho de certificação e obter informações adicionais; é necessário, portanto, dispor de Listas de Certificados Digitais, Certificados Digitais de Autoridades Certificadoras e diferentes versões de documentos obrigatórios, como políticas de certificados e declarações de práticas de certificações.

Hoje a maior parte desses arquivos encontra-se na guarda das entidades que os geraram, o que pode dificultar sua recuperação a médio e longo prazo, caso aquelas entidades não realizem os procedimentos de arquivamento adequados. Assim, é importante que a ICP-Brasil crie um repositório seguro com essas informações ou designe um PSA para fazê-lo. Os arquivos publicados nesse repositório teriam sua confiabilidade garantida por medidas técnicas ou organizacionais, de forma a serem pontos de referência confiáveis para a validação das assinaturas digitais realizadas por todos os usuários da ICP-Brasil, em qualquer momento que necessário.

3.3 Criar manuais orientando os usuários sobre o tratamento do legado

Até a publicação da Resolução 62, as aplicações que usam assinatura digital eram criadas de acordo com a necessidade e conveniência de cada usuário ou desenvolvedor, sem respeitar a qualquer padrão quanto ao formato da assinatura. Isso gerou um enorme legado de documentos assinados que podem não possuir as características necessárias à preservação por longo prazo.

De acordo com a Resolução 62, a definição dos critérios para validação desse legado ficou a cargo das partes interessadas. Como a definição desses critérios pode ser bastante complexa, especialmente para usuários leigos que não disponham de estrutura organizacional para apoiá-los, uma medida recomendada é a criação de um guia, orientando-os sobre as formas de realizar a validação e sobre os procedimentos a que os documentos assinados devem ser submetidos, em cada caso.

3.4 Alterar os termos de titularidade incluindo referências sobre arquivamento a longo prazo e formato dos documentos assinados

Ao adquirir um certificado digital, a maioria dos usuários conhece pouco ou quase nada sobre certificação digital, obsolescência de algoritmos criptográficos e a necessidade de realizar procedimentos complementares para manter documentos assinados com sua chave privada protegidos ante às ameaças já descritas.

Em função disso, recomenda-se que os documentos que esses usuários recebem ao obter um certificado ICP-Brasil (também conhecidos como “termos de titularidade”) passem a incluir orientações sobre aspectos importantes da preservação dos documentos assinados no longo prazo, tais como:

- em que situações um documento assinado digitalmente precisará ser submetido a procedimentos adicionais;
- que procedimentos são esses, caso o próprio titular deseje armazenar o documento;
- relação dos prestadores de serviço de arquivamento, caso ele decida por usar os serviços dessas entidades;
- conveniência de escolher formatos para o documento eletrônico assinado que facilitem sua recuperação no longo prazo.

3.5 Revisar os tipos e a aplicabilidade dos certificados ICP-Brasil

Na ICP-Brasil estão aprovados 4 tipos de certificados digitais para assinatura, conforme DOC-ICP-04 [19]: tipos A1, A2, A3 e A4.

Os certificados do tipo A1 e A2, cujas chaves privadas correspondentes são geradas e/ou armazenadas em software, oferecem menor nível de segurança aos seus proprietários, mas podem ser utilizadas para assinar qualquer tipo de documento.

Para evitar riscos, sugere-se que a utilização dos certificados tipo A1 e A2 fique restrita apenas para assinaturas com fins de autenticação (como acesso a sistemas), enquanto os certificados A3 e A4, cujas chaves privadas correspondentes estão armazenadas em dispositivos criptográficos mais seguros, como *smartcards* e *tokens*,

seriam usados para assinatura de documentos que representem comprometimento legal (como contratos, declarações etc.).

4 ANÁLISE DA PROPOSTA

As medidas propostas são bastante factíveis, pois apoiam-se em tecnologias já existentes e em utilização em outras ICPs. Sua implantação dependeria da criação e aprovação de regulamentos pelo Comitê Gestor da ICP-Brasil e, em alguns casos, do aporte de recursos para o órgão responsável pelas atividades operacionais da ICP-Brasil, o Instituto Nacional de Tecnologia da Informação (ITI).

A regulamentação do serviço de arquivamento, ao passo que criaria entidades na ICP-Brasil que precisariam ser credenciadas e auditadas periodicamente, aumentando a carga de trabalho do ITI, por outro lado traria evidentes vantagens, como:

- agregaria mais segurança aos documentos assinados e aos contratos e obrigações legais que eles representam;
- aproximaria as comunidades técnicas, legais e arquivísticas, em torno do objetivo de criar os regulamentos para as entidades de arquivamento por longo prazo;
- estimularia a geração e disseminação de conhecimentos nas áreas envolvidas.

A criação de repositório seguro de informações críticas, por sua vez, dependeria basicamente de decisão dos gestores da ICP-Brasil e de propiciar ao ITI as condições técnicas e humanas necessárias para gerir as bases de dados resultantes.

A criação de manuais orientando os usuários sobre o tratamento do legado e a alteração dos termos de titularidade são medidas que dependeriam apenas da disponibilidade de recursos humanos.

A revisão dos tipos e a aplicabilidade dos certificados ICP-Brasil necessitaria de ampla discussão por parte da sociedade brasileira, dadas as implicações legais decorrentes e face à existência de processos de negócio modelados para a utilização de certificados do tipo A1 e A2, que já geraram milhões de documentos assinados com aquele tipo de certificado.

5 CONCLUSÕES

Vários segmentos da sociedade e o crescimento da economia digital estão impulsionando a utilização de documentos eletrônicos e de assinaturas digitais. No Brasil, esse quadro é particularmente verdadeiro, dada a expansão observada no número de aplicações que utilizam assinaturas digitais com base em certificados emitidos pela ICP-Brasil. Assim, é de extrema importância para o País a regulamentação dos métodos, processos e soluções que serão empregados para criação, armazenamento e validação a longo prazo de documentos eletrônicos assinados digitalmente.

Analisando a regulamentação atual da ICP-Brasil, observa-se que algumas ações pontuais poderiam agregar um nível mais elevado de segurança e facilitar o atingimento de um dos seus principais objetivos, que é conferir validade legal a

documentos assinados digitalmente. Entre essas ações encontram-se medidas mais complexas, como a criação de novos tipos de entidades na ICP-Brasil até aquelas mais simples, como a criação de manuais e textos orientando os usuários.

Paralelamente, observa-se que ainda existe distanciamento entre as comunidades técnica e arquivística no que tange à preservação documental, não apenas no Brasil como no exterior. Essa situação deveria ser revertida o quanto antes, pois o arquivamento dos documentos eletrônicos assinados digitalmente possivelmente será realizado por especialistas arquivistas, cujas técnicas de preservação documental não visam a preservar a integridade dos bits, mas sim a dos documentos que estão representados pelos bits. Isso conflita com os processos usados na assinatura digital, onde a alteração de um único bit compromete a verificação de todo o documento. Uma solução para essa dicotomia deve ser buscada a partir de trabalho conjunto das comunidades envolvidas.

6 REFERÊNCIAS

1. Thomaz, K.: I Congresso de Tecnologias para Gestão de Dados e Metadados do Cone Sul (2003) Disponível em: <<http://conged.deinfo.uepg.br> >
2. International Organization for Standardization / International Electrotechnical Commission: ISO 14721:2003 - Space data and information transfer systems - Open archival information system - Reference model. Genève, Switzerland. ISO/IEC (2003)
3. Australia. Public Record Office Victoria. Management of Electronic Records PROS 99/007 (Version 2) <http://www.prov.vic.gov.au/vers/standard/version2.htm>
4. Ansper, A., et al.: Efficient long-term validation of digital signatures, in Proc. of 4th International Workshop on Practice and Theory in Public Key Cryptography (PKC 2001), pp. 402-415, Korea (2001)
5. Gritzalis, D., Lekkas, D.: Cumulative Notarization for Long-term Preservation of Digital Signatures, Computers & Security, Volume 23, Issue 5, July 2004, Pages 413-424, ISSN 0167-4048, DOI: 10.1016/j.cose.2004.03.002 (2004)
6. Maniatis, P., Baker, M.: Enabling the Archival Storage of Signed Documents, in Proc. of the FAST 2002 Conference on File and Storage Technologies, pp. 31-45, January 2002, USA (2002)
7. Wallace, C.: Long-Term Archive Service Requirements. Internet Engineering Task Force. RFC 4810 (2007). Disponível em: <<http://www.ietf.org/rfc/rfc4810.txt>>.
8. Gondrom, T.; Brandner, R.; Pordes, U.: Evidence Record Syntax (ERS). Internet Engineering Task Force. RFC 4998 (2007) Disponível em: <<http://www.ietf.org/rfc/rfc4998.txt>>.
9. Porekar, J., Saljic, S., Klobucar, T., Jerman-Blazic, A.: Technical Patterns for Long Term Trusted Archiving. Digital Society, 2009. ICDS '09. Third International Conference on, PP 241 – 246, Mexico (2009)
10. Aleksej, B., Toma K., Borka J.: Long-term trusted preservation service using service interaction protocol and evidence records. Computer Standards & Interfaces archive, Volume 29, Issue 3 (March 2007). pp 398-412, Slovenia (2007)
11. Boudrez, F.: Digital signatures and electronic records. Expertisecentrum DAVID vzw. Antwerpen (2005) Disponível em: <<http://www.expertisecentrumdavid.be/docs/digitalsignatures.pdf>>
12. Blanchette, J.: The digital signature dilemma - Le dilemme de la signature numérique. Pour publication dans Annales des Télécommunications, Mai/Juin (2006)

13. Dumortier, J.: Legal Reflexions concerning Digital Archiving. ECPRD Twin Seminar Brussels - The Hague (2002)
14. European Telecommunications Standards Institute: Policy requirements for trust service providers signing and/or storing data for digital accounting. ETSI TS 102 573. (2007) Disponível em: <<http://www.etsi.org/WebSite/Standards/Standard.aspx>>.
15. Brasil: Resolução Conarq 25/2007. Modelo de Requisitos para Sistemas Informatizados de Gestão Arquivística de Documentos (2007). Disponível em: <http://www.conarq.arquivonacional.gov.br/Media/publicacoes/earqbrasilv1.pdf>.
16. Brasil. Comitê Gestor da ICP-Brasil: Resolução N° 62 de 09 de janeiro de 2009. Aprova a versão 1.0 do documento Visão Geral sobre Assinaturas Digitais na ICP-Brasil. Infra-estrutura de Chaves Públicas Brasileira (2009). Disponível em: <<http://www.iti.gov.br/twiki/bin/view/Certificacao/Legislacao>>.
17. Brasil. Comitê Gestor da ICP-Brasil. Resolução N° 36, de 21 de Outubro de 2004 – Aprova o Regulamento para Homologação de Sistemas e Equipamentos de Certificação Digital no Âmbito da ICP-Brasil. Infra-estrutura de Chaves Públicas Brasileira (2004). Disponível em: <<http://www.iti.gov.br/twiki/bin/view/Certificacao/Legislacao>>.
18. Spanoudakis, G. et al. (eds.): Security and Dependability for Ambient Intelligence, Advances in Information Security 45, DOI: 10.1007/978-0-387-88775-3_19, © Springer Science + Business Media, LLC 2009 - In particular Chapter 19 "Applying the SERENITY methodology to the domain of trusted electronic archiving"
19. Brasil. Comitê Gestor da ICP-Brasil. Resolução N° 53, de 28 de Novembro de 2008 – Altera os Requisitos Mínimos para as Políticas de Certificado na ICP-Brasil (DOC-ICP-04). Infra-estrutura de Chaves Públicas Brasileira (2008). Disponível em: <<http://www.iti.gov.br/twiki/bin/view/Certificacao/Legislacao>>.

Generación de Ambientes para Entrenamiento en Seguridad Informática

Alejandro Blanco, Juan Diego Campo, Lucía Escanellas, Carlos Pintado, and
Marcelo Rodríguez

Grupo de Seguridad Informática, Instituto de Computación, Facultad de Ingeniería,
Universidad de la República
J. Herrera y Reissig 565, Montevideo, Uruguay
[ablanco,jdcampo,luciae,cpintado,marcelor]@fing.edu.uy
<http://www.fing.edu.uy/inco/gsi>

Resumen Los ataques informáticos son un tema latente en la actualidad. Para poder contrarrestar los riesgos, es necesario contar con profesionales informáticos especializados en seguridad. Dichos profesionales deben de ser capaces de entender cuáles son los riesgos, las amenazas y las vulnerabilidades a las que se están enfrentando. Para ello necesitan aprender tanto como sea posible de los problemas, y poder articular respuestas rápidas ante incidentes de seguridad.

En este contexto, poder contar con ambientes de entrenamiento en los cuales se puedan experimentar con técnicas, herramientas y problemas reales es fundamental.

Sin embargo, si se hace en forma manual, la creación y el mantenimiento de este tipo de ambientes se torna una tarea compleja y que consume mucho tiempo. Esto motiva el desarrollo de una herramienta que asista en la generación de ambientes para entrenamiento en seguridad informática. En este artículo se describe la arquitectura de una herramienta orientada a solucionar la problemática planteada. Como parte del diseño, se definió un lenguaje de especificación de prácticas de seguridad, que permite la declaración e instanciación de escenarios. Los escenarios, generados a partir de la especificación, emulan de manera fidedigna los ambientes de producción reales y permiten emplear herramientas comúnmente utilizadas en el mercado.

Palabras clave: Laboratorio de Seguridad Informática, Seguridad de la Información, Virtualización.

1. Introducción

El Grupo de Seguridad Informática (GSI) de la Facultad de Ingeniería de la Universidad de la República (Uruguay), imparte cursos de seguridad informática, que utilizan laboratorios basados en la metodología que se describe en “Concepción, Diseño e Implantación de un Laboratorio de Seguridad Informática” [2]. La metodología allí descripta está basada en el uso de virtualización para generar los ambientes de experimentación. Dicha solución fue aplicada con éxito en

varios cursos de grado y postgrado impartidos por el GSI desde el año 2006 a la fecha.

Sin embargo, la creación y el mantenimiento de laboratorios con estas características es una tarea compleja desarrollada en forma manual, y por ende, consume mucho tiempo. Esto se debe a los siguientes inconvenientes:

- Copiado y configuración manual de máquinas virtuales
El copiado de las máquinas virtuales y el copiado de archivos para cada estudiante se realiza individualmente. La configuración de red del hardware virtual también es manual.
- Recursos limitados
El espacio en disco necesario es proporcional a la cantidad de máquinas virtuales utilizadas. En el caso de una práctica para 40 estudiantes cuya imagen tenga un tamaño de 8GB se necesitaría almacenar 320GB.
- Corrección manual
Se debe verificar máquina por máquina si el estudiante logró llegar a la meta propuesta. Además, debido al problema anterior, la corrección de un laboratorio se debe realizar antes del armado del siguiente, dado que generalmente no es posible almacenar en el servidor las máquinas virtuales de ambos.
- Aislamiento de los entornos virtuales
No existe aislamiento entre las máquinas virtuales de cada estudiante. Esto permite que un estudiante interfiera con el entorno de otro estudiante.

Para resolver estos inconvenientes, surge la necesidad de desarrollar una herramienta que asista en la creación de laboratorios de seguridad informática. En este trabajo se discuten algunos detalles del análisis y diseño de la solución planteada.

El resto de este artículo está estructurado como se describe a continuación. En la Sección 2 se discute sobre trabajos relacionados. La Sección 3 describe la arquitectura de la herramienta y se especifica el lenguaje para definición de escenarios. La Sección 4 expone los componentes de la herramienta implementada. En la Sección 5 se reseña el enfoque utilizado para el desarrollo de experimentos con la herramienta desarrollada. La Sección 6 concluye y describe trabajo futuro.

2. Trabajo relacionado

Como parte del análisis, se hizo un relevamiento en busca de soluciones que atacaran la problemática de la enseñanza práctica de seguridad informática.

Muchas de las soluciones encontradas implementan los laboratorios de igual manera que el GSI (por ejemplo, los laboratorios de la Universidad de Indiana [5], Drexel [13], Toledo [16] o el Instituto Rochester [12]) y por lo tanto experimentan la misma clase de problemas en lo referente al mantenimiento y administración.

Otras soluciones atacan la problemática mediante la utilización de “técnicas de juego”, en donde se plantean desafíos de seguridad como ser: enseñanza de *ethical hacking*, prácticas de ingeniería inversa o cracking (por ejemplo: CyberCIEGE [4], WebGoat [17] o Mod-X [9]). En general este tipo de soluciones

utilizan simuladores que no son modificables por los usuarios, en lugar de aplicaciones reales y vulnerables. Este enfoque tiene la desventaja, si no se ha previsto en la simulación, de no contar con la realimentación proveniente de la interacción con procesos reales. Una ventaja es que se pueden diseñar las prácticas de manera tal que la evaluación se realice de manera automática.

Una iniciativa similar a la planteada en el presente trabajo es el proyecto Tele-Lab [7], desarrollado por el Departamento de Ciencias de la Computación de la Universidad de Trier en Alemania. La solución está orientada al e-Learning, basada en una aplicación Web mediante la cual se accede a un laboratorio implementado con máquinas virtuales. El inconveniente que se le encontró a la solución es que no se proveen mecanismos para que se puedan generar nuevos escenarios en forma sencilla.

3. Un Ambiente para Entrenamiento en Seguridad Informática

El trabajo que aquí se presenta es una continuación de lo expuesto en [2] y por lo tanto contribuye con el objetivo principal de contar con un “ambiente para el desarrollo de actividades de experimentación, investigación y enseñanza de seguridad informática”, que además cumpla con una serie de requerimientos ahí descritos. En ese trabajo, se discuten diferentes opciones de arquitectura para la solución, optando por la utilización de virtualización. Como aporte, se decidió estudiar otros paradigmas de virtualización, diferentes a los analizados, para incorporarlos a la solución. En particular, paravirtualización [11], y productos que la implementan (User Mode Linux [14]).

El objetivo que se comienza a explorar aquí es la concepción de una herramienta que facilite y automatice la creación de nuevos escenarios de experimentación. De manera adicional, se contempla la incorporación de mecanismos para la evaluación de prácticas realizadas sobre dichos escenarios.

3.1. Arquitectura de la Herramienta

La arquitectura que ha sido definida para la herramienta se ilustra en la Figura 1, la cual está inspirada en [7], contemplando las características que facilitan la generación de escenarios, así como también los inconvenientes que tiene la implementación actual de los laboratorios del GSI.

Se optó por una arquitectura en capas, que permite una implementación con modelo cliente servidor que brinda movilidad al laboratorio. La arquitectura consta de cuatro capas: la capa de presentación, la capa lógica, la capa de administración de prácticas y la capa de abstracción de máquinas virtuales.

La capa de presentación tiene dos *frontends*, uno para el estudiante y otro para el docente. Los mismos proveen una interfaz de usuario para el sistema, que puede ser remota, y proveen las operaciones necesarias para cada rol. Por ejemplo, el *frontend* del estudiante le brinda al mismo acceso a la práctica, y el *frontend* docente permite especificar prácticas y evaluarlas.

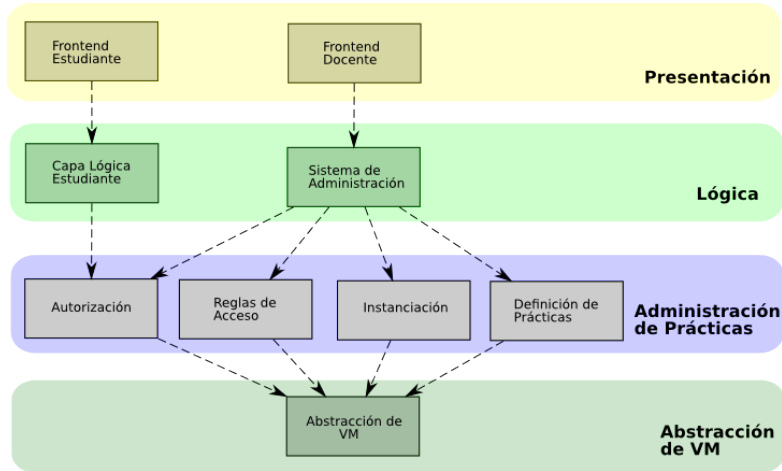


Figura 1. Arquitectura del Framework

La capa lógica comprende la *Capa Lógica del Estudiante* y el *Sistema de Administración*.

La *Capa Lógica del Estudiante* maneja los casos de uso que involucren al estudiante. Este componente es el encargado de proveer la funcionalidad necesaria para que el estudiante pueda acceder al sistema y permite mantener el estado de su sesión.

El *Sistema de Administración* brinda una abstracción para que se pueda manejar el sistema con facilidad. En primer lugar tiene el objetivo de servir de interfaz con el módulo de *Instanciación*, ofreciendo comandos de alto nivel, tales como instanciar o detener una práctica, mapeándolos a comandos de bajo nivel en la tecnología de virtualización utilizada. También tiene como objetivo servir de interfaz con el módulo de *Definición de Prácticas*, ofreciendo una interfaz amigable, que genere la especificación requerida por dicho módulo.

La capa de administración de prácticas contiene módulos encargados de aspectos inherentes al desarrollo de las mismas.

El módulo de *Autenticación* implementa autenticación de estudiantes y docentes para acceder a los distintos recursos del laboratorio.

El módulo de *Reglas de Acceso* es el encargado de controlar las reglas de firewall y de acceso a las máquinas. Es fundamental para implementar los requisitos de aislamiento que deben tener las prácticas.

El módulo de *Instanciación* se encarga de implementar la instanciación de prácticas. Su función es la de crear, ejecutar y detener las máquinas virtuales.

El módulo de *Definición de Prácticas* es el módulo encargado de implementar la creación de prácticas. A partir de una especificación de práctica independiente de la tecnología de virtualización, debe generar la información necesaria para instanciar la práctica con la tecnología utilizada. Dicha información puede incluir

máquinas virtuales, archivos de configuración, entradas de bases de datos, entre otros.

La capa de *Abstracción de VM* se encarga del manejo de la tecnología de virtualización utilizada, previendo la posibilidad de emplear más de una tecnología en forma concurrente. Contar con la posibilidad de utilizar más de una tecnología de virtualización en un mismo escenario permite sacar el máximo provecho de cada una de ellas.

La arquitectura aquí planteada es una primera aproximación al diseño general de un *framework* que implemente las funcionalidades necesarias para los laboratorios. En base a esto, se desarrolló una primera versión de la herramienta que permite avanzar en la investigación.

En la siguiente sección se detallan características del módulo de *Definición de Prácticas*, en particular, se destaca la definición de un lenguaje de especificación de prácticas de seguridad.

3.2. Lenguaje de definición de escenarios

Se considera un requerimiento importante que el módulo de *Definición de Prácticas* cuente con la facilidad de poder especificar los escenarios con un nivel de abstracción tal, que oculte detalles de implementación y configuración de bajo nivel.

Se investigaron alternativas para cumplir con este requerimiento, y en el contexto sobre el que se está trabajando, la búsqueda fue infructuosa. No así en el campo de simulación de redes, donde existen productos como Packet Tracer [10] (herramienta propietaria de Cisco [3]), ITguru [8] (herramienta para asistir en cursos de redes) y VNUML [15].

VNUML, Virtual Network User-Mode-Linux, es una herramienta de virtualización diseñada para definir y probar escenarios complejos de simulación de redes. Se basa en la tecnología de virtualización User Mode Linux y ejecuta máquinas virtuales a partir de una especificación del escenario en XML. Fue pensado para instanciar escenarios de cualquier tipo. En sus comienzos, el mismo fue desarrollado para simular redes IPv6, pero puede ser utilizado para simular escenarios de red genéricos basados en linux. Como contrapartida, no está pensado para generar ambientes contemplando problemas de seguridad y se encuentra ligado, como su nombre lo indica, al producto de virtualización UML.

Utilizar solamente esta herramienta en los laboratorios, si bien facilita aspectos de configuración de redes, tiene como costo asociado tener que migrar a la tecnología de virtualización UML. Esto implica convertir las máquinas utilizadas a la nueva tecnología, teniendo la desventaja de que no es posible utilizar sistemas operativos distintos a los basados en linux.

Por otro lado, disponer de UML tiene como ventaja que la tecnología permite un uso eficiente de los recursos, mediante su técnica de *Copy on write* ¹.

¹ *Copy on write* es una técnica que permite utilizar una imagen de disco base para una máquina virtual, en modo sólo lectura, y guardar en un archivo de diferencias los cambios realizados a esa imagen durante la ejecución de la máquina. Cuando varias

En busca de cumplir el requerimiento y ocultar los detalles de implementación y configuración a bajo nivel, se especificó un lenguaje formal. Este lenguaje, permite además independizar la especificación de escenarios del producto y paradigma de virtualización utilizado.

Dadas las características, el lenguaje adquiere una importancia sustancial dentro de la definición del *framework*.

A grandes rasgos, un escenario en este lenguaje se compone de una serie de definiciones de estructuras, que corresponden a elementos o actores del laboratorio. Cada una de estas estructuras tiene un identificador y un conjunto de atributos, algunos de los cuales son opcionales. Para el lenguaje se eligió una sintaxis similar a la definición de estructuras en C. Por ejemplo, la definición de una herramienta de ataque se muestra en la Figura 2.

```
Tool ettercap {  
    name = "ettercap";  
    description = "Multipurpose interceptor for switched LANs";  
    version = "0.7.3";  
    running = False;  
    operating_system = centos;  
    exploitable_vulnerabilities = {ARP_vulnerability};  
};
```

Figura 2. Ejemplo de definición de estructura

Como se puede observar, la definición de una instancia de estructura se realiza declarando el tipo de la estructura (en el ejemplo, *Tool*), un identificador para esta instancia (*ettercap*), una lista de atributos (*name*, *description*, *version*, *running*, etc), delimitada por llaves, y finaliza con un punto y coma. Los atributos de la lista se separan con punto y coma, y cada uno de ellos tiene un tipo determinado, que puede ser identificador (como el atributo *operating_system*), cadena de caracteres (como *name*), número, booleano (*running*) o lista (*exploitable_vulnerabilities*). En la Figura 3, se pueden ver los componentes más importantes del lenguaje, sus atributos y relaciones.

El lenguaje, de aquí en más LISLab, permite definir las siguientes estructuras: *address*, *application*, *attacker*, *disk*, *evidence* *expected_result*, *fw_rule*, *group*, *interface*, *ip_range*, *machine*, *network*, *operating_system*, *parameter*, *process*, *scenario*, *service*, *tool*, *user*, *victim*, *vulnerability*.

Los criterios que guiaron el diseño de estas estructuras fueron la necesidad de representar una topología de red con máquinas virtuales con aplicaciones y

máquinas comparten la misma imagen base, guardando sus respectivos archivos de diferencias, el consumo de espacio en disco es mucho menor, debido a que sólo se almacena una vez la mayor parte del contenido del disco.

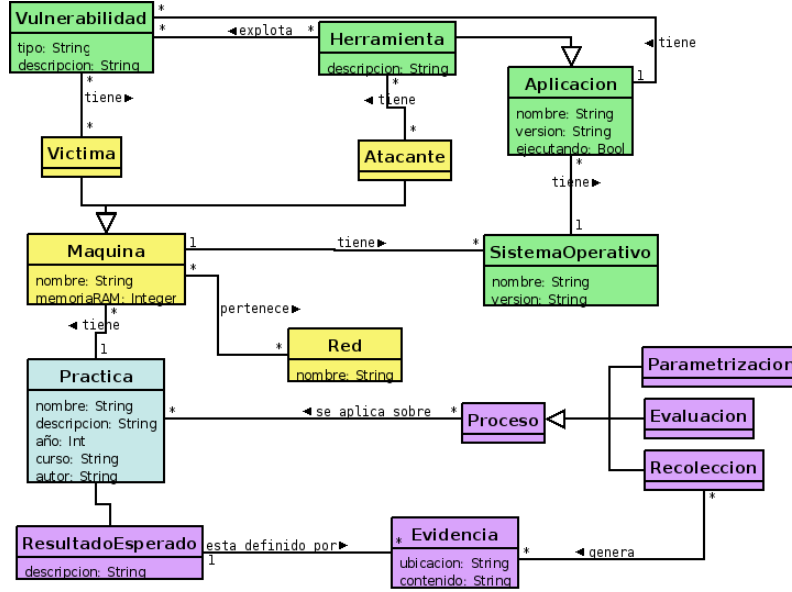


Figura 3. Algunos componentes del lenguaje

vulnerabilidades asociadas, y la necesidad de definir un mecanismo que asista en la evaluación de las prácticas.

Básicamente una especificación de escenario de experimentación en LISLab consiste en una definición de la estructura **Scenario** y definiciones de otras estructuras.

En la Figura 4 se muestra un fragmento de la gramática libre de contexto de LISLab, que reconoce la estructura *Tool* vista en el ejemplo.

4. Definición e Instanciación de escenarios

En la Sección “Metodología de trabajo” de [2] se describen un conjunto de componentes y procesos que son utilizados para definir los escenarios de experimentación.

Los componentes son esencialmente repositorios de máquinas virtuales y aplicaciones. Los procesos son planificación, preparación e instanciación de los escenarios. Si se piensa en evaluar las prácticas realizadas sobre los escenarios, existirían también procesos de evaluación.

Con estos antecedentes en mente, pensando en automatizar las tareas y utilizando el lenguaje LISLab, podemos inferir que:

- El resultado de la planificación es la especificación en LISLab del escenario planteado.

```

tool_def: TOOL_TOK tool_id OPENBRACE
        NAME_TOK EQUAL string SEMICOLON
        VERSION_TOK EQUAL string SEMICOLON
        OPERATING_SYSTEM_ATTR_TOK EQUAL operating_system_id
        SEMICOLON
        [RUNNING_TOK EQUAL BOOL SEMICOLON]
        [SERVICE_ATTR_TOK EQUAL service_id SEMICOLON]
        [VULNERABILITIES_TOK EQUAL OPENBRACE
        list_vulnerability_id CLOSEBRACE SEMICOLON]
        [EXPLOITABLE_VULNERABILITIES_TOK EQUAL OPENBRACE
        list_vulnerability_id CLOSEBRACE SEMICOLON]
        CLOSEBRACE SEMICOLON

```

Figura 4. Fragmento de la Especificación de la Gramática de LISLab

- Determinando en forma precisa cómo se manejan los repositorios y creando herramientas, se pueden automatizar los procesos de preparación e instanciaación.
- Para la evaluación, se pueden definir procesos que recolecten información que permita verificar el cumplimiento de los objetivos de la práctica, por ejemplo: recolectar logs de sistema.

A continuación se describen los componentes de la herramienta de especificación de escenarios diseñada. Es pertinente aclarar que ya existe una primera versión implementada de dicha herramienta, la cual está siendo evaluada.

4.1. Componentes de la Herramienta

En la Figura 5 se muestra un diagrama de componentes de la herramienta y la interacción entre los mismos.

El *Compilador* toma una especificación de práctica en el lenguaje LISLab y genera una práctica que pasará a formar parte del repositorio de prácticas.

El *Repositorio de Prácticas* almacena las prácticas generadas por el compilador, de forma tal que puedan ser utilizadas posteriormente por el módulo de instanciaación.

El *Repositorio de Imágenes de Disco* contiene las imágenes disponibles para la definición de prácticas. Las imágenes referenciadas en la especificación de la práctica son utilizadas por el módulo de instanciaación.

El *Repositorio de Aplicaciones* contiene las aplicaciones disponibles para ser instaladas en las máquinas. Las aplicaciones referenciadas en la especificación de la práctica son utilizadas por el módulo de instanciaación.

El *Módulo de Instanciación* es el encargado de instanciar una práctica a partir de su especificación. Utilizando las aplicaciones e imágenes de disco de los repositorios, genera un conjunto de escenarios virtuales.

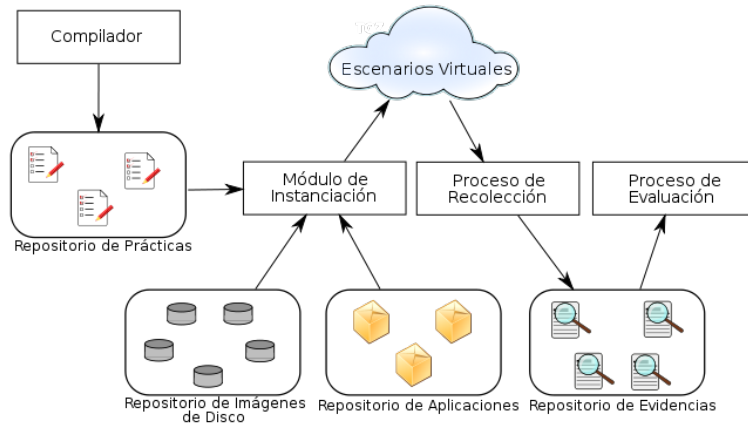


Figura 5. Componentes de la herramienta

Los *Escenarios Virtuales* son redes de máquinas virtuales que reflejan los escenarios definidos en la especificación de la práctica. Se genera uno de estos escenarios por cada participante que realice la práctica.

El *Proceso de Recolección* se encarga de obtener información de los escenarios virtuales, necesaria para evaluar la práctica, y almacenarla en el *Repositorio de Evidencias*. La información que debe obtenerse de los escenarios está definida en la especificación de la práctica.

El *Proceso de Evaluación* genera un reporte con los resultados para cada instancia de escenario virtual a partir de la información almacenada en el repositorio de evidencias.

4.2. Diseño del compilador

En la Figura 6 podemos observar los procesos que necesitamos implementar en el compilador y las transformaciones por las que pasa la especificación de una práctica. El objetivo es que a partir de una especificación en LISLab se generen los componentes necesarios para poder instanciar una práctica. Es pertinente aclarar, que es en este punto que se produce la asociación con el producto de virtualización utilizado, ya que los componentes generados serán dependientes del producto de virtualización.

En la primera versión de la herramienta, se decidió utilizar UML [14] como producto de virtualización y VNUML [15] como herramienta de instanciación. Para ello se realizaron extensiones a dicha herramienta para soportar conceptos no implementados, como ser: creación de usuarios, instalación de aplicaciones y herramientas de seguridad, parametrización de las instancias y definición de reglas de firewall.

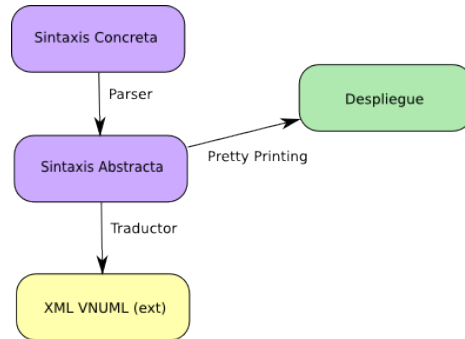


Figura 6. Módulos Involucrados en el Diseño del Compilador

El compilador consta de dos partes:

- *Frontend*

El *frontend* está compuesto por un *parser*, cuya salida es una representación, en sintaxis abstracta, de una definición válida del lenguaje, a partir de una entrada en el lenguaje de sintaxis concreta. La sintaxis abstracta es, a su vez, entrada para el traductor. También cuenta con un módulo de *pretty printing* que permite desplegar el contenido de las estructuras internas.

- *Backend*

El *backend* es un traductor que transforma la representación interna de la práctica, una instancia de la sintaxis abstracta, en un lenguaje objetivo. Recordamos que para la primera versión de la herramienta, el lenguaje objetivo es una extensión de VNUML.

Con este diseño se buscó que el lenguaje objetivo se pueda cambiar en un futuro, en cuyo caso habría que implementar otro *backend* para la sintaxis abstracta.

5. Discusión de una práctica: *Man in the Middle*

El objetivo de la práctica es mostrar cómo es posible interceptar, sin ser detectado, una comunicación entre dos equipos pertenecientes a una red, con el fin de obtener información confidencial o introducir modificaciones a dicha información.

El escenario de la práctica consiste de un equipo controlado por el estudiante, que cumple el rol de atacante, y dos equipos que se transmiten mensajes entre sí, los cuales cumplen el rol de víctimas.

Los tres equipos están en una misma red, en un mismo dominio de *broadcast*, pero en distintos dominios de colisión, lo que se conoce como una red *switchheada*. El propósito de utilizar ésta topología es obligar al estudiante a utilizar un

ataque de *ARP spoofing* previo a poder interceptar el mensaje. Para realizar dicho ataque el estudiante contará con la herramienta *ettercap* [6].

En la Sección “Discusión de una Práctica: *Man in the Middle*” de [2] se describía el proceso para armar el escenario correspondiente. Dicho proceso se llevaba adelante manualmente. Para crear el atacante se tomaba, de un repositorio de máquinas base, una máquina virtual con sistema GNU/Linux, a la cual se le instalaban las herramientas necesarias. Las máquinas víctimas se generaban de manera similar. Luego se replicaba el escenario tantas veces como participantes realizaban la práctica. Los procesos de ejecutar y detener el escenario también eran manuales, teniendo que levantar o detener tres máquinas por cada instancia de escenario.

En contraposición, con la herramienta desarrollada, luego de que se tiene definido el escenario a un nivel conceptual, se procede a especificarlo en el lenguaje LISLab. Es importante destacar que dicha especificación se hace una única vez y luego se almacena en el *repositorio de prácticas*

```

Scenario mitm {
  name = "Man In The Middle";
  description =
    "A scenario to show
    a MitM attack";
  year = 2008;
  course = "FSI";
  attackers = {mitm_attacker};
  victims = {client_victim,
    server_victim};
};
Attacker mitm_attacker {
  name = "vm-att";
  ram = 256;
  tools = {ettercap};
};

Victim client_victim {
  name = "vm-cv";
  ram = 256;
  applications = {send_message};
};
Victim server_victim {
  name = "vm-sv";
  ram = 256;
  applications = {reply_message};
};
Network NMitm {
  range = range_mitm;
  switched = true;
};
Tool ettercap {
  name = "ettercap";
  version = "0.7.3-1.2";
  running = false;
};

```

Figura 7. Especificación de MitM en LISLab

En la Figura 7 se presenta un fragmento de la especificación en LISLab de la práctica. Esta especificación incluye la definición del escenario (*mitm*), la máquina atacante (*mitm_attacker*), la víctima servidor (*server_victim*), y la víctima cliente (*client_victim*). También se muestra la definición de la red switchheada utilizada en la práctica (*NMitm*). Por último se puede ver la definición de la herramienta *ettercap*, que será instalada en *mitm_attacker*.

Una vez especificado, se procede a compilar y el *módulo de instanciación* es quien se encarga de replicar los escenarios tantas veces como se le indique.

Para instanciar la práctica, se ejecuta el comando `start_scenario`, indicando el nombre de la práctica y la cantidad de instancias como parámetro.

En este momento, por cada instancia se levantan tres máquinas virtuales de UML, una por cada máquina definida en la especificación. Estas máquinas se conectan en una red virtual, privada a la instancia, que refleja la topología definida en la especificación de la práctica.

Además de la red privada de cada instancia, se crean redes punto a punto entre el *host* y todas las máquinas virtuales. Es utilizando estas redes que, tanto estudiantes como docentes, acceden a las máquinas de la práctica. Además, a través de estas redes de administración, las máquinas pueden tener acceso a redes externas.

Para resumir, podemos destacar que con la herramienta se obtienen mejoras en el proceso de creación de escenarios, entre las cuales se destacan las siguientes:

- No se requiere conocimiento del producto de virtualización, o del sistema operativo utilizado al momento de definir las prácticas. Las diversas configuraciones de máquinas, redes, instalación de aplicaciones, definición de reglas de firewall, etc. se realizan de manera transparente al instanciar las prácticas.
- La replicación del escenario para cada participante se realiza de manera automática, disminuyendo el tiempo necesario para instanciar las prácticas.
- Las máquinas de las instancias pueden compartir la misma imagen base de disco, utilizando la tecnología de *Copy on write*.

6. Puesta en Producción

La herramienta se puso en producción en un curso en el cual se utilizaron 10 escenarios para una práctica. Cada escenario estaba compuesto de una máquina virtual que disponía de un servidor web con una aplicación vulnerable [1]. Las máquinas contaban con 750MB de espacio en disco y 256Mb de memoria RAM. El objetivo de la práctica consistía en explotar la vulnerabilidad y obtener una *shell reversa*².

Instancias anteriores de dicho laboratorio se ejecutaban sobre dos servidores Sun Fire X2100 x64 equipados con 2GB de memoria RAM, en la infraestructura que se describe en [2]. En este contexto, son necesarios 7.5GB de espacio en disco, y los servidores estaban destinados exclusivamente a la ejecución de las máquinas virtuales.

En contrapartida, utilizando la herramienta descrita en este trabajo, el laboratorio fue ejecutado en un *notebook* con 2GB de memoria RAM y con un procesador Core 2 Duo de 1.5Ghz. En esta experiencia se utilizó 900MB de disco físico en total.

Una particularidad a destacar de esta experiencia es que la utilización de la herramienta permitió que el curso, en particular el laboratorio, se pudiera dictar en aulas de la institución receptora. De esta manera, se pudo instrumentar una

² La técnica de shell reversa consiste en que la máquina víctima genere el pedido de conexión hacia el atacante, y le permita ejecutar comandos arbitrarios en la víctima.

modalidad de dictado en la cual los participantes no tenían que trasladarse de su lugar de trabajo.

Finalmente podemos concluir que este ensayo permitió comprobar que la herramienta cumple con los requerimientos funcionales y de performance necesarios.

7. Conclusiones y Trabajo Futuro

Se implementó una primer versión de la herramienta para la definición e instanciación de escenarios.

En particular se destaca la creación del lenguaje LISLab, siendo éste uno de los resultados más relevantes del trabajo desarrollado. La definición de este lenguaje permite expresar los componentes y actores de los escenarios de seguridad, a un nivel de abstracción adecuado, y en gran medida de forma independiente a la tecnología utilizada.

Este lenguaje también actuará como interfaz al resto del *framework*, permitiendo a los demás módulos manejar conceptos abstractos, sin necesidad de conocer los detalles de implementación.

Esta primera versión de la herramienta permite confirmar la factibilidad de concebir un *framework* que facilite y automatice la creación de nuevos escenarios de experimentación para seguridad informática. También permite verificar la hipótesis de que utilizando otros paradigmas y productos de virtualización, se pueden optimizar los recursos.

Durante la fase de verificación de la herramienta, se vio que es posible migrar la mayoría de los escenarios utilizados en el laboratorio de los actuales cursos impartidos por el GSI, lo cual constituye un aporte importante.

A partir de este trabajo, surgen varias líneas de trabajo a futuro. Con respecto a la definición del lenguaje de especificación de escenarios, es factible extenderlo para:

- Formalizar la definición de vulnerabilidades.

La especificación de las vulnerabilidades que se ilustran en las diferentes prácticas actualmente son simplemente un texto donde se las describe. Puede resultar de interés clasificar las vulnerabilidades y formalizar su especificación.

- Extender la definición de evidencias

Las evidencias son un conjunto de elementos de las máquinas (*logs*, archivos, procesos) a analizar, cuyo cometido es verificar si el estudiante cumplió el propósito de la práctica. Visto de esta manera, el análisis podría servir como parte del mecanismo evaluador de la práctica.

Se podrían utilizar lenguajes formales para la especificación de las evidencias que es necesario recolectar del escenario de la práctica. Durante el desarrollo de la primera versión de la herramienta, este aspecto fue simplificado, suponiendo que las evidencias las constituyen simplemente archivos con cierto contenido en las máquinas virtuales.

Tal como está definido, el proceso de evaluación personalizado está asociado a una práctica, por lo que no pueden existir dos procesos de evaluación personalizados para dos evidencias distintas. Si bien actualmente existe un solo tipo de evidencia, en el caso de generalizar este concepto, debería existir la posibilidad de evaluar estas evidencias según su tipo.

Además, la evaluación asociada a una evidencia podría especificar el momento en que se evalúa la evidencia, si debe hacerse durante la realización de la práctica, o puede hacerse luego de finalizada la misma.

Esta es una línea de trabajo en la cual se podrían incorporar técnicas y herramientas de análisis forense informático.

- Implementar intérpretes para otras tecnologías de virtualización.

Una limitación de la herramienta es que solamente se pueden especificar escenarios en los cuales se utiliza linux como sistema operativo. Intérpretes para otras tecnologías permitirían el uso de distintos sistemas operativos, así como hacer uso de las ventajas de los paradigmas de virtualización más adecuados para cada práctica. Una forma de hacerlo sería implementar completamente el intérprete y no hacer uso de VNUML.

- Implementar soporte para múltiples *kernels*

En la versión actual todas las máquinas de una práctica deben utilizar la misma versión del kernel Linux. Un posible trabajo a futuro es extender el lenguaje LISLab para que permita especificar qué kernel ejecuta cada máquina.

En cuanto al resto de la implementación, se podría:

- Desarrollar un módulo completo de instanciación de prácticas.

En la primera versión de la herramienta, para simplificar, se utilizó una versión especialmente modificada de VNUML.

Este módulo debería ser flexible y configurable. Esto se podría implementar definiendo un lenguaje de instanciación de escenarios, en el se pueda especificar, por ejemplo, grupos de estudiantes que participan del laboratorio, o los parámetros específicos de cada instancia.

- Formalizar el mecanismo de instalación de aplicaciones

La instalación de aplicaciones podría mejorarse incluyendo *metadata* que indique para cada aplicación aspectos como el sistema operativo, arquitectura de hardware soportada y dependencias. Por ejemplo, incluir una aplicación en la definición de la práctica requiere incluir también todas las dependencias de esa aplicación. Si ésta se incluye en varias prácticas, se deberán definir estas dependencias en cada una de sus especificaciones. En cambio, con la *metadata* en el repositorio de aplicaciones, sólo es necesario definir una vez la información asociada a la aplicación.

- Implementación de otros módulos que hasta el momento no fueron implementados.

Los módulos no implementados son: una interfaz de usuario para el estudiante y otra para el docente, y los módulos de Autorización y Reglas de Acceso. Las interfaces de usuario permitirán un acceso amigable a las funcionalidades

del sistema, y permitirán el acceso remoto a la infraestructura. Los módulos de Autorización y Reglas de Acceso permiten acceder al laboratorio a los estudiantes desde redes externas, de forma segura y controlada.

Como se puede ver, esta primera versión abre posibilidades de trabajos a futuro, tanto en el ámbito académico con la utilización de lenguajes formales en seguridad informática, como en la implementación del completa de la herramienta que permita su puesta en producción en laboratorios de seguridad informática.

Referencias

1. Awstats vulnerability. <http://www.securityfocus.com/bid/12543/exploit>, Último acceso: 25/04/09.
2. Gustavo Betarte, María Eugenia Corti, and Marcelo Rodríguez. Concepción, diseño e implementación de un laboratorio de seguridad informática. En Anales del Congreso Iberoamericano de Seguridad Informática (CIBSI'07), Mar del Plata, noviembre 2007.
3. Cisco systems, inc. <http://www.cisco.com>, Último acceso: 21/04/09.
4. CyberCIEGE, An innovative video game and tool to teach network security concepts. <http://cistr.nps.edu/cyberciege/>, Último acceso: 21/04/09.
5. David Dellacca and Connie Justice. Building tomorrow's information assurance workforce through experiential learning. Technical report, Indiana University-Purdue University Indianapolis, 2007.
6. Ettercap. <http://ettercap.sourceforge.net>, Último acceso: 26/04/09.
7. Ji Hu, Dirk Cordel, and Christoph Meinel. A virtual machine architecture for creating IT-security labs. Technical report, Hasso-Plattner Institut, October 2006.
8. Itguru academic edition. http://www.opnet.com/university_program/itguru_academic_edition/index.html, Último acceso: 26/04/09.
9. Mod-X Security Challenge. <http://www.mod-x.co.uk>, Último acceso: 21/04/09.
10. Cisco packet tracer. http://www.cisco.com/web/learning/netacad/course_catalog/PacketTracer.html.
11. Paravirtualization is state of the art virtualization, xen. <http://www.xen.org/about/paravirtualization.html>, Último acceso: 26/04/09.
12. Sylvia Perez-Hardy. A unique experiential model for teaching network administration. Technical report, Rochester Institute of Technology, 2003.
13. Vassilis Prevelakis. Supporting a security laboratory. Technical report, Drexel University, 2007.
14. User Mode Linux. <http://uml.jfdi.org/uml/>, Último acceso: 25/04/09.
15. VNUML. Virtual Network User Mode Linux. <http://www.dit.upm.es/vnumlwiki/>, Último acceso: 25/04/09.
16. James Walden. A real-time information warfare exercise on a virtual network. Technical report, University of Toledo, 2005.
17. OWASP Webgoat Project. http://www.owasp.org/index.php/Category:OWASP_WebGoat_Project, Último acceso: 21/04/09.

Robust Declassification for Bytecode

Francisco Bavera¹ and Eduardo Bonelli^{2*}

¹ UNRC - CONICET, Ruta 36 km 601, Río Cuarto, Argentina (fbavera@gmail.com)

² UNQ - CONICET, Quilmes, Argentina (ebonelli@unq.edu.ar)

Abstract. The noninterference property states that a system is void of insecure information-flows from private to public data. This constitutes too severe a requirement for many systems which allow some form of controlled information release. Therefore, an important effort is being invested in studying forms of intended information declassification. Robust declassification is a property of systems with declassification primitives that ensures that a third party cannot affect the behavior of a system in such a way as to declassify information other than was intended. This paper studies robust declassification for a core fragment of bytecode. We observe that, contrary to what happens in the setting of high-level languages, variable reuse, jumps and operand stack manipulation can be exploited by an attacker to declassify information that was not intended to be. A type system for ensuring robust declassification is introduced and proved to be sound.

Key words: Confidentiality, Information-Flow, Type System, Declassification, Security

1 Introduction

This work is concerned with practical, language-based information flow [SM03] policies for mobile code. Information flow policies in terms of noninterference ensure the absence of information channels from private to public data. Although an intuitively appealing information security policy, it has been recognized to be too strict for practical applications since many applications do allow some form of information declassification or downgrading. Examples are: releasing the average salary from a secret database of salaries for statistical purposes, a password checking program, releasing an encrypted secret, etc. This calls for relaxed notions of noninterference where primitives for information declassification are adopted. At the same time we are also interested in mobile code. Currently a large amount of code is distributed over the Internet largely in the form of bytecode [LY99] or similar low-level languages. Direct verification of security properties of bytecode allows the code consumer to be assured that its security policy is upheld. It should also be stressed that most efforts on information flow target high-level languages yet compilation can produce a program which does not necessarily satisfy the same security properties. Analysis at the bytecode

* Work partially supported by LIFIA and Instituto Tecnológico de Buenos Aires

level has the benefit of independence of the mechanism (compiled, handwritten, code generated) that generated it.

An important issue that arises in mobile code (or any code) that provides a declassification mechanism is whether it can be exploited affecting the data that is declassified or whether declassification happens at all. If this does not arise we declare our declassification mechanism *robust*. We briefly illustrate that with an example from [MSZ04] translated to bytecode.

```

1. load  $x_L$ 
2. if 6
3. load  $z_H$ 
4. declassify L
5. store  $y_L$ 
6. ...

```

This program releases private information held in z_H to a publicly readable variable y_L if the variable x_L is true, and otherwise has no effect. Here, **declassify** explicitly indicates a release from secret to public of the level of the topmost element on the stack. As a consequence, this program does not enjoy noninterference. However, we can observe something more on the reliability of the declassification mechanism. Indeed, an attacker can affect whether information is declassified at all (assuming she can change the value of x_L) or can affect the information that is declassified (assuming she can change the value of z_H). This program is therefore declared not to be robust under any of these two assumptions.

The starting point of this paper is a core fragment of bytecode that includes a declassification instruction **declassify** κ , where κ is a security level (see Sec. 3). The effect of this instruction is to declassify the top element of the operand stack of the JVM run-time system. Our language allows flow sensitivity [HS06] w.r.t. the security level of the variables (i.e. local variables may be reused with different security levels, as in standard bytecode). Under an attacker that observes and manipulates data, we address the requirements that programs must meet in order for robustness to be enjoyed.

Contributions. We study the notion of robust declassification of a low-level language such as bytecode and reveal subtle ways in which, in low-level code, the standard notion of robust declassification fails. Particularly, this fails in the presence of flow-sensitive types and low-level stack manipulation primitives. We propose a type system for enforcing robust declassification in bytecode and prove that this system is sound.

Structure of the paper. Sec. 2 introduces motivating examples. It is followed by the attacker model and the bytecode language. The type system is presented in Sec. 4. Sec. 5 addresses noninterference for confidentiality and robust declassification. Finally, we conclude and suggest further avenues for research. For further details and full proofs please consult [BB09].

2 Motivating examples

In general, an active attacker may change system behavior by injecting new code in the program. Such attacker-controlled low-integrity computation may be interspersed with trusted high-integrity code. To distinguish the two, the high-integrity code is represented as a program in which some statements are missing, replaced by holes ($\bullet$). The idea is that the holes are places where the attacker can insert arbitrary low-integrity code. There may be multiple holes (possibly none) in the high-integrity code, represented by the notation $\overrightarrow{\bullet}$. Such expressions are called *high-integrity contexts*, written $B[\overrightarrow{\bullet}]$, and formally defined as programs in which the set of bytecode instructions is extended with a distinguished symbol “ $\bullet$ ”. These holes can be replaced by a vector of attacker fragments $\overrightarrow{a}$ to obtain a complete method $B[\overrightarrow{a}]$. An attacker is thus a vector of such code fragments. A passive attacker is an attack vector that fills all holes with the low-integrity code from the original program. An active attacker is any attack vector that fills some hole in a way that changes the original program behavior.

One way an attacker can cause information leak is to directly violate confidentiality and integrity (eg. using `declassify` in its own code). However, one is interested in how an attacker can exploit insecure information flow in the trusted code. Therefore, we restrict our attention to attacks that are *active* in this sense (cf. Def. 2). We discuss some examples³.

1. <code>load x</code>	1. $[\bullet]$	1. $[\bullet]$	1. $[\bullet]$
2. <code>load y</code>	2. <code>load x</code>	2. <code>load x</code>	2. <code>load x</code>
3. $[\bullet]$	3. <code>declassify L</code>	3. <code>declassify L</code>	3. <code>if 5</code>
4. <code>declassify L</code>	4. ...	4. <code>return</code>	4. <code>declassify L</code>
5. ...			5. ...
(a)	(b)	(c)	(d)

In (a), assuming x and y are high-integrity at line 4, the value of y is declassified. If the attacker consists of a `pop` instruction, then declassification of the value of x is forced. Likewise the attacker could push an element on the stack rather than remove it. In this case it is the newly pushed element that is declassified. Let us consider another example. In (b) the security level of x at 2 is assumed to be confidential. If the attacker inserts `load y; goto 3` where y holds some high confidentiality data, then by affecting the control-flow y is now declassified. Finally, if the code is that of (c), then by inserting the code `load y; return` declassification of x is avoided.

One final consideration is that we must also restrict variable reuse in low-integrity contexts. Consider the code excerpt of (d) where x is assumed high-integrity at 2 and y low-integrity at 2. Line 4 attempts to declassify depending on the value of x (a high-integrity value). Suppose the attacker reuses x with a low-integrity value, such as in `load y; store x`. Then in line 4 a declassification is attempted depending on the value of the variable y (a low-integrity

³ In these examples, for expository purposes, the argument of `declassify` is L. However these labels are actually pairs of levels (cf. Sec. 3).

value). Therefore the attacker can manipulate the condition in line 3 which decides whether declassification is performed or not. This situation is avoided by restricting attackers from manipulating high-integrity variables such as x above.

Our definition of *active attack* thus requires attacks to not include **return** nor **declassify** instructions. Likewise, we require all jumps to be local to the attackers code. The size of the stack before the attacker code is executed should be the same after it is executed. Finally, high-integrity variables and stack elements should not be modifiable. These observations are made precise in Def. 2 and the typing schemes for bytecode (Sec. 4).

3 Attacker Model and the Bytecode Language

Since robustness of a system depends on the power an attacker has to affect the execution of the system we follow [MSZ04] and introduce *integrity* labels in our model. High-integrity information and code are trusted and assumed not to be affected by the attacker while low-integrity information and code are untrusted and assumed to be under the control of the attacker. Both confidentiality and integrity labels are merged into a single lattice of *security levels*.

We assume given a lattice of confidentiality levels $\mathcal{L}_C$ with ordering $\preceq_C$ and a lattice of integrity levels $\mathcal{L}_I$ with ordering $\preceq_I$. We use $\perp_C$ and $\top_C$ for the bottom and top elements of $\mathcal{L}_C$ and similarly for those of $\mathcal{L}_I$. We write l_C for elements of the former and l_I for elements of the latter and $\sqcup_C$ and $\sqcup_I$, resp., for the induced join. A *security level* is a pair (l_C, l_I) . These pairs are elements of a lattice $\mathcal{L}$ and are typically denoted using letters $\kappa, \lambda, \kappa_i, \lambda_i, \dots$. The partial order $\preceq$ on $\mathcal{L}$ is defined as: $(l_C, l_I) \preceq (l'_C, l'_I)$ iff $l_C \preceq_C l'_C$ and $l_I \preceq_I l'_I$. The join on security levels is: $(l_C, l_I) \sqcup (l'_C, l'_I) = (l_C \sqcup_C l'_C, l_I \sqcup_I l'_I)$. The projections of the first and second component of a security level are $C((l_C, l_I)) = l_C$ and $I((l_C, l_I)) = l_I$. Also, we occasionally write C_κ for $C(\kappa)$ and I_κ for $I(\kappa)$.

The power of an attacker is described by a security level A , where $C(A)$ is the confidentiality level of data the attacker is expected to be able to read, and $I(A)$ is the integrity level of data or code that the attacker is expected to be able to affect. Thus, the robustness of a system is w.r.t. the attacker security level A .

The attacker security level A determines both high and low-confidentiality ($H_C = \{\kappa \mid C(\kappa) \not\preceq C(A)\}$ and $L_C = \{\kappa \mid C(\kappa) \preceq C(A)\}$) areas and high and low-integrity areas ($H_I = \{\kappa \mid I(A) \not\preceq I(\kappa)\}$ and $L_I = \{\kappa \mid I(A) \preceq I(\kappa)\}$) in the security lattice $\mathcal{L}$. We can identify four key areas of the lattice: (1) $LH \simeq L_C \cap H_I$; (2) $HH \simeq H_C \cap H_I$; (3) $LL \simeq L_C \cap L_I$; and (4) $HL \simeq H_C \cap L_I$. The LL region is under complete control of the attacker: she can read and modify data in this region. Data in the LH region may be read but not modified by the attacker. The attacker may not read nor modify data in the region HH . Finally, data in the HL region may not be inspected but can be modified by the attacker.

The Bytecode Language Let $\mathbb{N}$ stand for the natural numbers and $\mathbb{X}$ for the set of local variables whose elements we refer to by x, y . A *program* B is

a sequence of the following bytecode instructions where op is $+$ or $\times$, $x \in \mathbb{X}$, $j \in \mathbb{N}$:

push j	push j on stack	pop	pop from stack
load x	load value of x on stack	store x	store top of stack in x
prim op	primitive operation	ifeq j	conditional jump
goto j	unconditional jump	return	return
declassify κ	declassifies top of stack		

These instructions are standard except for **declassify** κ which *declassifies* the security level of value on the top of the operand stack. It does not have any effect on the state of the program, as described below. We write $\text{Dom}(B)$ for the set of *program points* of B and $B(i)$, with $i \in 1..n$ and n the length of B , for the i^{th} instruction of B .

We write $\mathbb{V}$ for the set of *values* whose elements are denoted by v . Values are taken to be the set of integers. *Machine states* are tuples $\langle i, \alpha, \sigma \rangle$, where $i \in \mathbb{N}$ is the program counter that points to the next instruction to be executed; α , (local variable array) is a mapping from local variables to values; σ (stack) is an operand stack. A machine state *for* B is one in which the program counter points to an element in $\text{Dom}(B)$. If $\langle i, \alpha, \sigma \rangle$, then we define $\text{pc}(s) = i$.

The small-step operational semantics is standard (cf. Appendix). We write $s_1 \longrightarrow_B s_2$ if both s_1, s_2 are states for B and s_2 results by one step of reduction from s_1 . In the sequel we work with states for some program B which shall be clear from the context and thus simply speak of states without explicit reference to B . Expressions of the form $\langle v \rangle$ are referred to as *final states*. A non-final state s may either reduce to another non-final state or to a final state. Initial states take the form $\langle 1, \alpha, \epsilon \rangle$, where ϵ denotes the empty stack. We write $\rightarrow_B$ for the reflexive-transitive closure of $\longrightarrow_B$. The trace $\text{Tr}_B(\langle i, \alpha, \sigma \rangle)$ of the execution from state $\langle i, \alpha, \sigma \rangle$ ($i \in \text{Dom}(B)$) is the sequence $\langle i, \alpha, \sigma \rangle \longrightarrow_B \langle i_1, \alpha_1, \sigma_1 \rangle \longrightarrow_B \langle i_2, \alpha_2, \sigma_2 \rangle \longrightarrow_B \langle i_3, \alpha_3, \sigma_3 \rangle \longrightarrow_B \dots$.

4 Type System

A *method type* is an expression of the form $(x_1 : \kappa_1, \dots, x_n : \kappa_n, \kappa_r)$, abbreviated $(\mathbf{x} : \kappa, \kappa_r)$, where $x_1, \dots, x_n$ are the formal parameters together with their security levels and κ_r is the return value security level. A *method* $M[\vec{\bullet}]$ is a pair $((\mathbf{x} : \kappa, \kappa_r), B[\vec{\bullet}])$, where $B[\vec{\bullet}]$ is a program referred to as the method's body. Methods $M[\vec{\bullet}]$ are typed by means of a *typing judgment*: $\mathcal{V}, \mathcal{S}, \mathcal{A} \triangleright M[\vec{\bullet}]$. Each of $\mathcal{V}, \mathcal{S}$ and $\mathcal{A}$ are called *typing contexts*. Typing contexts supply information needed to type each instruction of $M[\vec{\bullet}]$. As such they constitute families of either functions $(\mathcal{V}, \mathcal{S})$ or security levels $(\mathcal{A})$ indexed by a finite set of instruction addresses. We write $\mathcal{V}_i$ to refer to the i^{th} element of the family and similarly with the others. The first typing context assigns a *variable array type* V to each instruction (of $M[\vec{\bullet}]$). A variable array type is a function assigning security labels to each local variable. Thus $\mathcal{V}_i$ indicates the types of the local variables at instruction i . The $\mathcal{S}$ typing context assigns a *stack type* S to each instruction.

A stack type is a function assigning security labels to numbers from 1 to n , where n is assumed to be the length of the stack. $\mathcal{A}$ indicates the level of each instruction. If $\mathcal{A}(i)$ is high at some program point because $C(\mathcal{A}(i)) \in H_C$, then an attacker might learn secrets from the fact that execution reached that point. If $\mathcal{A}(i)$ is high because $L(\mathcal{A}(i)) \in L_I$, the attacker might be able to affect whether control reaches that point.

Two further ingredients are required before formulating the type system. The first is a notion of subtyping for variable array and stack types.

$$\frac{\forall x \in \mathbb{X}. V'(x) \preceq V''(x)}{V' \leq V''} \quad \frac{\begin{array}{c} |S'| = |S''| = n \\ \forall j \in 1..n. \begin{cases} S'(j) \preceq S''(j) \text{ if } \kappa \preceq S'(j), \\ S'(j) = S''(j) \text{ otherwise} \end{cases} \end{array}}{S' \leq_\kappa S''}$$

Frame types are compared pointwise using the ordering on security labels. We write $|S|$ for the length of S . $S' \leq_\kappa S''$ asserts that stack type S' is a subtype of S'' at level κ . Stack types may only be compared if they are of the same size. Furthermore, we allow depth subtyping at position j provided that $S'(j)$ is at least κ . The other ingredient is the availability of *control dependence regions* [BR05,BPR07].

Control Dependence Region High-level languages have control-flow constructs that explicitly state dependency. For example, from `while x begin c1; c2 end` one easily deduces that both `c1` and `c2` depend on `x`. Given that such constructs are absent from bytecode, as in most low-level languages, and that they may be the source of unwanted information flows, our type system requires this information to be supplied. Let $\text{Dom}(B)^\sharp$ denote the set of program points of $B[\vec{\bullet}]$ where branching instructions occur (i.e. $\text{Dom}(B)^\sharp = \{k \in \text{Dom}(B) \mid B[\vec{\bullet}](k) = \text{if } i\}$). We assume given two functions ($\wp$ below denotes the powerset operator): (1) **region**: $\text{Dom}(B)^\sharp \rightarrow \wp(\text{Dom}(B))$ and (2) **jun**: $\text{Dom}(B)^\sharp \rightarrow \text{Dom}(B)$.

The first computes the *control dependence region*, an over-approximation of the range of branching instructions and the second the unique junction point of a branching instruction at a given program point. We only require some abstract properties of these functions to hold [BPR07]. We need the notion of successor relation to formulate these properties.

Definition 1. *The successor relation $\mapsto \subseteq \text{Dom}(B) \times \text{Dom}(B)$ of a method $B[\vec{\bullet}]$ is defined by the clauses: (1) If $B[\vec{\bullet}](i) = \text{goto } k$, then $i \mapsto k$; (2) If $B[\vec{\bullet}](i) = \text{if } k$, then $i \mapsto i + 1$ and $i \mapsto k$; (3) If $B[\vec{\bullet}](i) = \text{return}$, then $i \not\mapsto$ (i has no successors); and (4) Otherwise, $i \mapsto i + 1$.*

A small set of requirements, the *safe over approximation property* or *SOAP*, on **region** and **jun** suffice for the proof of noninterference. They state some basic properties on how the successor relation, **region** and **jun** are related.

Property 1. Let $i \in \text{Dom}(B)^\sharp$. (1) If $i \mapsto i'$, then $i' \in \text{region}(i)$. (2) If $i' \mapsto i''$ and $i' \in \text{region}(i)$, then $i'' \in \text{region}(i)$ or $i'' = \text{jun}(i)$. (3) If $i \mapsto^* i'$ and $i' \in$

$\text{Dom}(B)^\# \cap \text{region}(i)$, then $\text{region}(i') \subseteq \text{region}(i)$. (4) If $\text{jun}(i)$ is defined, then $\text{jun}(i) \notin \text{region}(i)$ and $\forall i'$ such that $i \mapsto^* i'$, either $i' \mapsto^* \text{jun}(i)$ or $\text{jun}(i) \mapsto^* i'$.

Intuitively, successors i' of a branching instruction i should be in the same region. As for successors i'' of an instruction i' with $i' \in \text{region}(i)$, either i'' should be in the region associated to i or be a junction point for this region. Regarding the third item we write $\mapsto^*$ for the reflexive, transitive closure of $\mapsto$ and say that i' is an *indirect successor* of i when $i \mapsto^* i'$. This item reads as follows: if i' is the program point of a branching instruction that is an indirect successor of a branching instruction i that has not left the region of i , then the region of i' is wholly inside that of i . The final item states that the junction point of a region are not part of this region and, moreover, all indirect successors of the program point i of a branching instruction either come before or after (in terms of the indirect successor relation) the junction point of the region of i .

In what follows we shall adopt two assumptions that, without sacrificing expressiveness, simplify our analysis. First we assume a method body $B[\vec{\bullet}]$ has a unique **return** instruction. Second, that predecessors of a junction point are always **goto** instructions (i.e. $i \in \text{region}(k)$ and $i \mapsto i'$ and $i' = \text{jun}(k)$ implies $B(i) = \text{goto } i'$). Note that this last assumption also helps us guarantee Property 1(1).

4.1 Typing Schemes

We assume throughout this work that we are dealing with valid JVM bytecode in the sense that it passes the bytecode verifier. For example, in an instruction such as **prim** + we do not check whether the operands are indeed numbers. A program $B[\vec{\bullet}]$ is *well-typed* under method type $(\mathbf{x} : \kappa, \kappa_r)$ if there exist typing contexts $\mathcal{V}$, $\mathcal{S}$ and $\mathcal{A}$ such that the type judgment $\mathcal{V}, \mathcal{S}, \mathcal{A} \triangleright ((\mathbf{x} : \kappa, \kappa_r), B[\vec{\bullet}])$, defined below, holds.

$$\frac{\forall x_j \in \mathbf{x}. \mathcal{V}_1(x_j) = \kappa_j \quad \forall i \in \text{Dom}(B). \mathcal{V}, \mathcal{S}, \mathcal{A}, i \triangleright ((\mathbf{x} : \kappa, \kappa_r), B[\vec{\bullet}])}{\mathcal{V}, \mathcal{S}, \mathcal{A} \triangleright ((\mathbf{x} : \kappa, \kappa_r), B[\vec{\bullet}])}$$

The first requirement is that the variable array type $\mathcal{V}_1$ provide security levels for all parameters and must agree with the ones assigned to each of them by the declaration $\mathbf{x} : \kappa$. Also, all program points in $B[\vec{\bullet}]$ must type as the second requirement indicates. A sample of the typing schemes defining this second requirement are given in Fig. 1 (cf. appendix for the full set) and are described below. Note that there may be more than one set of typing contexts $\mathcal{V}$, $\mathcal{S}$ and $\mathcal{A}$ such that a program is well-typed. For reasons of technical convenience we assume, without loss of generality, that the chosen $\mathcal{A}$ is minimal such that $\forall i \in \text{Dom}(B). \mathcal{A}(1) \preceq \mathcal{A}(i)$. A *method* $((\mathbf{x} : \kappa, \kappa_r), B[\vec{\bullet}])$ is *well-typed* if $B[\vec{\bullet}]$ is a well-typed program under $(\mathbf{x} : \kappa, \kappa_r)$ with the additional assumption that the stack type is empty, $\mathcal{S}_1 = \epsilon$.

We now briefly describe the typing schemes. They introduce a set of constraints between the frames type and stacks type at different instructions in the program. We write $\mathcal{S}_{i+1}(0)$ for the topmost element of the stack type $\mathcal{S}_{i+1}$ and

$$\begin{array}{c}
\text{(T-IF)} \\
\frac{
\begin{array}{l}
B[\bullet](i) = \text{if } i' \\
\mathcal{V}_i \leq \mathcal{V}_{i+1}, \mathcal{V}_{i'} \\
\mathcal{S}_i \leq_{\mathcal{A}(i)} \kappa \cdot \mathcal{S}_{i+1} = \kappa \cdot \mathcal{S}_{i'} \\
\mathcal{S}_i(0) \in \mathbb{L}_I, \text{ if } \mathcal{A}(i) \in \mathbb{L}_I \\
\forall k \in \text{region}(i). \kappa \preceq \mathcal{A}(k) \\
i+1, i' \in \text{Dom}(B)
\end{array}
}{\mathcal{V}, \mathcal{S}, \mathcal{A}, i \triangleright ((x : \kappa, \kappa_r), B[\bullet])}
\end{array}
\quad
\begin{array}{c}
\text{(T-STR)} \\
\frac{
\begin{array}{l}
B[\bullet](i) = \text{store } x \\
\mathcal{S}_i \leq_{\mathcal{A}(i)} \mathcal{V}_{i+1}(x) \cdot \mathcal{S}_{i+1} \\
\mathcal{V}_i \setminus x \leq \mathcal{V}_{i+1} \setminus x \\
\mathcal{A}(i) \preceq \mathcal{V}_{i+1}(x) \\
\mathcal{S}_i(0), \mathcal{V}_i(x) \in \mathbb{L}_I, \text{ if } \mathcal{A}(i) \in \mathbb{L}_I \\
i+1 \in \text{Dom}(B)
\end{array}
}{\mathcal{V}, \mathcal{S}, \mathcal{A}, i \triangleright ((x : \kappa, \kappa_r), B[\bullet])}
\end{array}$$

$$\begin{array}{c}
\text{(T-DECLASSIFY)} \\
\frac{
\begin{array}{l}
B[\bullet](i) = \text{declassify } \kappa \\
(\mathcal{S}_i \setminus 0) \leq_{\mathcal{A}(i)} (\mathcal{S}_{i+1} \setminus 0) \\
\mathcal{A}(i) \sqcup \kappa \preceq \mathcal{S}_{i+1}(0) \\
l(\mathcal{S}_i(0)) = l(\kappa) \\
l(\mathcal{A}(i)), l(\mathcal{S}_i(0)) \in \mathbb{H}_I \\
\mathcal{V}_i \leq \mathcal{V}_{i+1} \\
i+1 \in \text{Dom}(B)
\end{array}
}{\mathcal{V}, \mathcal{S}, \mathcal{A}, i \triangleright ((x : \kappa, \kappa_r), B[\bullet])}
\end{array}
\quad
\begin{array}{c}
\text{(T-HOLE)} \\
\frac{
\begin{array}{l}
B[\bullet](i) = [\bullet] \\
\mathcal{A}(i) \in \mathbb{L}_C \\
\mathcal{S}_i \leq_{(\perp_C, l(\mathcal{A}))} \mathcal{S}_{i+1} \\
\mathcal{V}_i \leq \mathcal{V}_{i+1} \\
i+1 \in \text{Dom}(B)
\end{array}
}{\mathcal{V}, \mathcal{S}, \mathcal{A}, i \triangleright ((x : \kappa, \kappa_r), B[\bullet])}
\end{array}$$

Fig. 1. Typing Schemes

$\mathcal{S}_{i+1} \setminus 0$ for $\mathcal{S}_{i+1}$ without the topmost element. Subtyping (rather than equality) for variable array and stack types are required given that instruction $i+1$ may have other predecessors apart from i . T-STORE, since instruction **store** x does change local variables, $\mathcal{V}_i \setminus x$ must be a subtype of $\mathcal{V}_{i+1} \setminus x$ as stated by the third line. Furthermore, the value of local variable x is modified by stack top, then $\mathcal{S}_i \leq \mathcal{V}_{i+1} \cdot \mathcal{S}_{i+1}$ must hold and $\cdot$. The four line states that the security level of the x value in follow instruction must be greater than or equal to the security level of current address. Regarding T-IF, we assume that **if** i' instructions have always two branches (in this case i' and $i+1$). T-IF requires $\mathcal{V}_{i'} = \mathcal{V}_{i+1}$ must hold. Since control jumps to i' or $i+1$, $\mathcal{V}_i$ must be a subtype of $\mathcal{V}_{i'}$ and $\mathcal{V}_{i+1}$ as stated by the second line. $\mathcal{S}_i$ must be a subtype of $\kappa \cdot \mathcal{S}_{i'}$ and $\kappa \cdot \mathcal{S}_{i+1}$. Moreover, the security level of **region**(i) must be greater than or equal to the security level κ of the value inspected at i . T-DECLASSIFY states that only high-integrity data is allowed to be declassified and that declassification might only occur at a high-integrity security environment. Because the possible flow origins is within the high-integrity area of the lattice, the attacker (who can only affect the low-integrity area of the lattice) cannot influence uses of the declassification mechanism and therefore cannot exploit it. Regarding T-HOLE, the second condition deserves a comment. For robustness, it is important that holes not be placed at program points where the type environment is high-confidentiality, because then the attacker would be able to directly observe implicit flows to the hole and could therefore cause information to leak even without a **declassify**.

As a final comment, note that these schemes disable high integrity variable reuse and popping of high integrity values off the stack in low integrity contexts. These constraints are necessary for proving that the type system guarantee robustness.

Now that the type system is in place and our motivating examples have been introduced we can formalize the notion of an active attack.

Definition 2 (Active attack). A program a is an active attack under method type $(x : \kappa, \kappa_r)$ if a does not contain **declassify** or **return** instructions and there exist typing contexts $\mathcal{V}, \mathcal{S}$ and $\mathcal{A}$ such that the following judgment holds:

$$\frac{\forall x_j \in x. \mathcal{V}_1(x_j) = \kappa_j \quad \forall i \in \text{Dom}(a). \mathcal{V}, \mathcal{S}, \mathcal{A}, i \triangleright ((x : \kappa, \kappa_r), a) \quad \mathcal{A}(1) = (\perp_C, \mathbf{l}_A) \quad |\mathcal{S}_1| = |\mathcal{S}_{exit}|}{\mathcal{V}, \mathcal{S}, \mathcal{A} \triangleright ((x : \kappa, \kappa_r), a)}$$

The index *exit* in $\mathcal{S}_{exit}$ refers to the program point $n + 1$ where n is the size of a . We assume that execution always ends at this program point. Also, as in the definition of well-typed programs, we assume the chosen $\mathcal{A}$ verifies $\forall i \in \text{Dom}(a). \mathcal{A}(1) \preceq \mathcal{A}(i)$.

5 Soundness

The type system enforces two interesting properties: noninterference and robust declassification. Noninterference states that any two (terminating) runs of a well-typed method, that does not use **declassify**, starting from initial states that coincide on public input data produce final states in which public output data are identical. This relation on states, called *indistinguishability*, is formalized in Sec. 5.1 and we then give the structure of the proof of noninterference in Sec. 5.2. Robust declassification states that an attacker may not manipulate the declassification mechanism to leak more information than intended. This property applies to programs that may contain **declassify** instructions. Sec. 5.3 introduces the precise definition and proves the main result of this paper.

5.1 Indistinguishability

Indistinguishability for states at some level λ is formulated by considering each of its components at a time (values, local variable assignments and stacks). States are declared indistinguishable depending on what may be observed and this, in turn, depends on the level κ of the observer. For further discussion on these notions please consult [BB08, BB09].

Definition 3 (Value indistinguishability). Given values v_1, v_2 and security levels κ, λ we define $v_1 \sim_\kappa^\lambda v_2$ (“values v_1 and v_2 are indistinguishable at level λ w.r.t. observer level κ ”) to hold if either (1) $\kappa \preceq \lambda$ and $v_1 = v_2$; or (2) $\kappa \not\preceq \lambda$.

Definition 4 (Local variable assignment indistinguishability). Let α_1, α_2 be local variable assignments, V_1, V_2 frame types and κ, λ security levels. We write $\alpha_1 \sim_{(V_1, V_2), \kappa}^\lambda \alpha_2$ (or $\alpha_1 \sim_{V_1, \kappa}^\lambda \alpha_2$ when $V_1 = V_2$ and $\kappa \preceq \lambda$).

- Low-indist. assignments ($\kappa \preceq \lambda$ and $V_1 = V_2$). α_1 and α_2 are considered low-indist. at level λ if for all $x \in \mathbb{X}$, $\alpha_1(x) \sim_{V_1(x)}^\lambda \alpha_2(x)$.
- High-indist. assignments ($\kappa \not\preceq \lambda$). α_1 and α_2 are considered high-indist. at level λ if for all $x \in \mathbb{X}$, $\alpha_1(x) \sim_{V_1(x) \sqcup V_2(x)}^\lambda \alpha_2(x)$.

Definition 5 (Stack indistinguishability). Let σ, σ' be stacks, S, S' stack types and λ, κ security levels. We write $\sigma \sim_{(S, S'), \kappa}^\lambda \sigma'$ (or $\sigma \sim_{S, \kappa}^\lambda \sigma'$, when $S = S'$ and $\kappa \preceq \lambda$).

– Low-indist. stacks ($\kappa \preceq \lambda$).

$$\frac{\kappa \preceq \lambda}{\epsilon \sim_{\epsilon, \kappa}^\lambda \epsilon} \quad \frac{\sigma \sim_{S, \kappa}^\lambda \sigma' \quad v \sim_{\kappa'}^\lambda v' \quad \kappa \preceq \lambda}{v \cdot \sigma \sim_{\kappa' \cdot S, \kappa}^\lambda v' \cdot \sigma'}$$

– High-indist. stacks ($\kappa \not\preceq \lambda$).

$$\frac{\sigma \sim_{S, \kappa'}^\lambda \sigma' \quad \kappa' \preceq \lambda \quad \kappa \not\preceq \lambda}{\sigma \sim_{(S, S), \kappa}^\lambda \sigma'} \quad \frac{\sigma \sim_{(S, S'), \kappa}^\lambda \sigma' \quad \kappa', \kappa \not\preceq \lambda}{v \cdot \sigma \sim_{(\kappa' \cdot S, S'), \kappa}^\lambda v' \cdot \sigma'} \quad \frac{\sigma \sim_{(S, S'), \kappa}^\lambda \sigma' \quad \kappa', \kappa \not\preceq \lambda}{\sigma \sim_{(S, \kappa' \cdot S'), \kappa}^\lambda v \cdot \sigma'}$$

Two states are indistinguishable if they are either both executing in a high region (in which case we say they are *high-indistinguishable*) or they are both executing the same instruction in a low region (*low-indistinguishable*). Furthermore, in any of these cases the frame types and operand stacks should also be indistinguishable.

Definition 6 (Machine state indistinguishability). Given security level λ , $\langle i, \alpha, \sigma \rangle \sim^\lambda \langle i', \alpha', \sigma' \rangle$ holds iff (1) $\mathcal{A}(i), \mathcal{A}(i') \not\preceq \lambda$ or $\mathcal{A}(i) = \mathcal{A}(i') \preceq \lambda$ and $i = i'$; (2) $\alpha \sim_{(\mathcal{V}_i, \mathcal{V}_{i'}), \mathcal{A}(i)}^\lambda \alpha'$; and (3) $\sigma \sim_{(S_i, S_{i'}), \mathcal{A}(i)}^\lambda \sigma'$.

5.2 Noninterference

Henceforth we assume $M[\vec{a}]$ to be well-typed: $\mathcal{V}, \mathcal{S}, \mathcal{A} \triangleright ((\mathbf{x} : \kappa, \kappa_r), B[\vec{a}])$. We let $s_1 = \langle i, \alpha_1, \sigma_1 \rangle$ and $s_2 = \langle i, \alpha_2, \sigma_2 \rangle$ be states for $M[\vec{a}]$. Furthermore, let λ be a security level.

Definition 7. $M[\vec{a}]$ satisfies noninterference, written $\text{NI}(M[\vec{a}])$, if for every $\kappa \preceq \mathbf{A}$, α_1, α_2 variable arrays and v_1, v_2 values: (1) $\langle 1, \alpha_1, \epsilon \rangle \longrightarrow^* \langle v_1 \rangle$, (2) $\langle 1, \alpha_2, \epsilon \rangle \longrightarrow^* \langle v_2 \rangle$ and (3) $\alpha_1 \sim_{\mathcal{V}_1, \kappa}^{\mathbf{A}} \alpha_2$ imply $v_1 \sim_{\kappa_r}^{\mathbf{A}} v_2$.

Our notion of noninterference assumes the attacker has the ability to see the initial low variables value, and its final result (variant of termination-insensitive information flow). We do not address other covert channels (e.g. termination, timing, or abstraction-violation attacks) in this type system.

Proposition 1. All well-typed methods $M[\vec{a}]$, without *declassify* instructions, satisfy $\text{NI}(M[\vec{a}])$.

Prop. 1 is a consequence of a stronger property (Prop. 2) that we state below. This latter property is used in our result on robustness (Prop. 3). First we introduce some definitions. The λ -projection of a trace t , written $t|_\lambda$, where $t = \langle i_0, \alpha_{i_0}, \sigma_{i_0} \rangle \longrightarrow_{B[\vec{a}]} \langle i_1, \alpha_1, \sigma_1 \rangle \longrightarrow_{B[\vec{a}]} \dots$, is the sequence of states $\langle i_0, \alpha_{j_0}, \sigma_{j_0} \rangle \langle j_1, \alpha_{j_1}, \sigma_{j_1} \rangle \langle j_2, \alpha_{j_2}, \sigma_{j_2} \rangle \langle j_3, \alpha_{j_3}, \sigma_{j_3} \rangle \dots$ s.t. $0 < j_1 < j_2 < \dots$ and for all i_k , if $\mathcal{A}(i_k) \preceq \lambda$ then $i_k = j_n$, for some n . We define two finite traces t_1

and t_2 of the same length n to be indistinguishable, written $t_1 \sim^\lambda t_2$, iff their λ -projections are indistinguishable $t_1|_\lambda \sim^\lambda t_2|_\lambda$. This latter property holds iff for all $k \in 0..n$, $\langle i_k, \alpha_{i_k}, \sigma_{i_k} \rangle \sim^\lambda \langle j_k, \alpha_{j_k}, \sigma_{j_k} \rangle$ where $t_1|_\lambda = \langle i_0, \alpha_{i_0}, \sigma_{i_0} \rangle \cdots \langle i_n, \alpha_{i_n}, \sigma_{i_n} \rangle$ and $t_2|_\lambda = \langle j_0, \alpha_{j_0}, \sigma_{j_0} \rangle \cdots \langle j_n, \alpha_{j_n}, \sigma_{j_n} \rangle$.

Proposition 2. *Suppose $M[\vec{a}]$ has no **declassify** instructions and s_1 and s_2 are initial states such that $s_1 \sim^\lambda s_2$. Then $\text{Tr}_{B[\vec{a}]}(s_1) \sim^\lambda \text{Tr}_{B[\vec{a}]}(s_2)$.*

The proof relies on three *unwinding lemmas*.

Lemma 1. *Suppose (1) $\mathcal{A}(i) \preceq \lambda$; (2) $s_1 \longrightarrow_{B[\vec{a}]} s'_1$; (3) $s_2 \longrightarrow_{B[\vec{a}]} s'_2$; and (4) $s_1 \sim^\lambda s_2$. Moreover, assume $B[\vec{a}]$ does not contain **declassify**. Then (1) either $s'_1 = \langle i'_1, \alpha'_{i'_1}, \sigma'_{i'_1} \rangle$, $s'_2 = \langle i'_2, \alpha'_{i'_2}, \sigma'_{i'_2} \rangle$ and $s'_1 \sim^\lambda s'_2$; or (2) $s'_1 = \langle v'_1 \rangle$, $s'_2 = \langle v'_2 \rangle$ and $v'_1 \sim^\lambda_{\kappa_r} v'_2$.*

Lemma 2. *Let $i_1, i_2 \in \text{region}(k)$ for some k such that $\forall k' \in \text{region}(k) : \mathcal{A}(k') \not\preceq \lambda$. Furthermore, suppose: (1) $s_1 \sim^\lambda s'_1$; (2) $s_1 \longrightarrow_{B[\vec{a}]} s_2$; (3) $s_2 = \langle i'_1, \alpha'_{i'_1}, \sigma'_{i'_1} \rangle$; (4) and $i'_1 \in \text{region}(k)$. Then $s_2 \sim^\lambda s'_1$.*

Lemma 3. *Let $i_1, i_2 \in \text{region}(k)$ for some k such that $\forall k' \in \text{region}(k) : \mathcal{A}(k') \not\preceq \lambda$. Suppose: (1) $s_1 \longrightarrow_{B[\vec{a}]} s'_1$ and $s_2 \longrightarrow_{B[\vec{a}]} s'_2$; (2) $s_1 \sim^\lambda s_2$, f ; and (3) $s'_1 = \langle i'_1, \alpha'_{i'_1}, \sigma'_{i'_1} \rangle$ and $s'_2 = \langle i'_2, \alpha'_{i'_2}, \sigma'_{i'_2} \rangle$; (4) and $\mathcal{A}(i'_1) = \mathcal{A}(i'_2)$ and $\mathcal{A}(i'_1) \preceq \lambda$. Then $i'_1 = \text{jun}(k) = i'_2$ and $s'_1 \sim^\lambda s'_2$.*

Remark 1. Noninterference may be proved for any program which is well-typed rather than just well-typed methods: the requirement that execution begin at a state where the stack is empty is not necessary for the result to hold.

5.3 Robustness

Definition 8 (Robustness). *Let $M[\vec{\bullet}] = ((x : \kappa, \kappa_r), B[\vec{\bullet}])$ be a well-typed method. We say that it satisfies robustness with respect to active attacks at level A if for every s_1, s'_1 initial states and $\vec{a}_1, \vec{a}_2$ active attacks:*

$$\text{Tr}_{B[\vec{a}_1]}(s_1) \sim^{(C(A), \top_I)} \text{Tr}_{B[\vec{a}_1]}(s'_1) \text{ implies } \text{Tr}_{B[\vec{a}_2]}(s_1) \sim^{(C(A), \top_I)} \text{Tr}_{B[\vec{a}_2]}(s'_1)$$

Robust declassification holds if the attacker's observations from the execution of $M[\vec{a}_2]$ may not reveal any secrets apart from what the attacker already knows from observations about the execution of $M[\vec{a}_1]$. The main result of this work is therefore:

Proposition 3. *All well-typed methods satisfy robustness.*

In order to address the proof two auxiliary results are required. The first (Lem. 4) states that an active attack cannot consult or modify high-integrity data on the stack and the second (Lem. 8) extends this observation to the states of a trace.

Lemma 4. *An active attack: (1) cannot consult or modify high-integrity data on the stack; and (2) satisfies noninterference.*

The proof of the second item follows immediately from Prop. 2, the first one is by case analysis on the instructions of the attack. Regarding the second auxiliary result, it's proof proceeds on the basis of three unwinding lemmas. First a definition.

Definition 9. *Let $s_1 = \langle i_1, \alpha_1, \sigma_1 \rangle$ and $s_2 = \langle i_2, \alpha_2, \sigma_2 \rangle$. We define:*

1. $H_I(\alpha_1) = H_I(\alpha_2)$ iff $\forall x \in \mathbb{X}. \mathcal{V}_{i_1}(x), \mathcal{V}_{i_2}(x) \in H_I \Rightarrow \alpha_1(x) = \alpha_2(x)$;
2. $H_I(\sigma_1) = H_I(\sigma_2)$ iff $\forall k \in 0..min(|\sigma_1|, |\sigma_2|) - 1. \mathcal{S}_{i_1}(k), \mathcal{S}_{i_2}(k) \in H_I \Rightarrow \sigma_1(k) = \sigma_2(k)$;
3. $H_I(s_1) = H_I(s_2)$ iff $H_I(\alpha_1) = H_I(\alpha_2)$, $H_I(\sigma_1) = H_I(\sigma_2)$ and $(\mathcal{A}(i_1), \mathcal{A}(i_2) \in H_I \Rightarrow \mathcal{A}(i_1) = \mathcal{A}(i_2))$.

The three unwinding lemmas follow together with the statement of Lem. 8.

Lemma 5. *Suppose (1) $\mathcal{A}(s_1) \in H_I$; (2) $s_1 \xrightarrow{B[\vec{a}_1]} s'_1$; (3) $s_2 \xrightarrow{B[\vec{a}_2]} s'_2$; and (4) $H_I(s_1) = H_I(s_2)$. Then $H_I(s'_1) = H_I(s'_2)$.*

Lemma 6. *Let $pc(s_1), pc(s'_1), pc(s_2) \in L_I$. Furthermore, suppose: (1) $H_I(s_1) = H_I(s'_1)$; (2) $s_1 \xrightarrow{B[\vec{a}_1]} s_2$; (3) and $pc(s_2) \in \text{region}(k)$. Then $H_I(s_2) = H_I(s'_1)$.*

Lemma 7. *Let $pc(s_1), pc(s_2) \in L_I$ and $pc(s'_1), pc(s'_2) \in H_I$. Furthermore, suppose: (1) $s_1 \xrightarrow{B[\vec{a}_1]} s'_1$ and $s_2 \xrightarrow{B[\vec{a}_2]} s'_2$; (2) $H_I(s_1) = H_I(s_2)$; and (3) and $pc(s'_1) = pc(s'_2)$. Then $H_I(s'_1) = H_I(s'_2)$.*

Lemma 8 (Preservation of equality of high integrity data). *Let $M[\vec{\bullet}] = ((\mathbf{x} : \kappa, \kappa_r), B[\vec{\bullet}])$ be a well-typed method and $\vec{a}_1, \vec{a}_2$ active attacks. Let κ s.t. for all $\kappa' \in H_I : \kappa' \preceq \kappa$, s_i in $\text{Tr}_{B[\vec{a}_1]}(s_1)|_{(\top_C, \kappa)}$, s'_i in $\text{Tr}_{B[\vec{a}_2]}(s'_1)|_{(\top_C, \kappa)}$, and $pc(s_i), pc(s'_i) \in H_I$. If $H_I(s_1) = H_I(s'_1)$, then $H_I(s_i) = H_I(s'_i)$.*

We now deliver the promised result, namely the proof of Prop. 3. Consider any two execution paths of $M[\vec{a}_2] = ((\mathbf{x} : \kappa, \kappa_r), B[\vec{a}_2])$:

$$\begin{array}{l} s_1 \xrightarrow{B[\vec{a}_2]} s_2 \xrightarrow{B[\vec{a}_2]} \cdots \xrightarrow{B[\vec{a}_2]} s_n \\ s'_1 \xrightarrow{B[\vec{a}_2]} s'_2 \xrightarrow{B[\vec{a}_2]} \cdots \xrightarrow{B[\vec{a}_2]} s'_m \end{array}$$

where (1) $s_1 = \langle 1, \alpha_1, \sigma_1 \rangle$ and $s'_1 = \langle 1, \alpha'_1, \sigma'_1 \rangle$ with $\sigma_1 = \sigma'_1 = \epsilon$, (2) $s_n = \langle v \rangle$ and $s'_m = \langle v' \rangle$, and (3) $s_1 \sim^{(C(A), \top_I)} s'_1$ and suppose $\mathcal{A}(1) \in L_C$. If $H_I = \emptyset$ then declassification is disallowed by the typing rules and the result follows from Prop. 2. Otherwise we proceed as follows. Starting from s_1 and s'_1 repeatedly apply Lem. 1 until it is no longer possible. Let $s_{j'}$ and $s'_{j'}$ be the states that are reached. Note $s_{j'} \sim^{(C(A), \top_I)} s'_{j'}$. If $j = n$, then the result holds immediately since the full trace has been attained. Otherwise, $j < n$ and we have two cases to treat (we write $\mathcal{A}(s)$ for $\mathcal{A}(i)$, where $s = \langle i, \alpha, \sigma \rangle$):

1. Case $\mathcal{A}(s_j), \mathcal{A}(s'_j) \preceq (\mathbf{C}(\mathbf{A}), \top_I)$, $s_j = \langle i, \alpha, v \cdot \sigma \rangle$ and $s'_j = \langle i', \alpha', v' \cdot \sigma' \rangle$ and $B[\vec{a}_2](i) = \text{declassify } \kappa$. From $s_j \sim^{(\mathbf{C}(\mathbf{A}), \top_I)} s'_j$ and $\mathcal{A}(s_j), \mathcal{A}(s'_j) \preceq (\mathbf{C}(\mathbf{A}), \top_I)$ we deduce $i = i'$. Also, from the semantics $s_{j+1} = \langle i+1, \alpha, v \cdot \sigma \rangle$ and $s'_{j+1} = \langle i+1, \alpha', v' \cdot \sigma' \rangle$. Finally, by typability of $B[\vec{\bullet}]$ all hypothesis of T-DECLASSIFY are assumed.

We wish to prove $s_{j+1} \sim^{(\mathbf{C}(\mathbf{A}), \top_I)} s'_{j+1}$. The only non-trivial case is $v \cdot \sigma \sim_{\mathcal{S}_{i+1}, \mathcal{A}(i+1)}^{(\mathbf{C}(\mathbf{A}), \top_I)} v' \cdot \sigma'$. For this we first check that $\sigma \sim_{\mathcal{S}_{i+1} \setminus 0, \mathcal{A}(i+1)}^{(\mathbf{C}(\mathbf{A}), \top_I)} \sigma'$. This fact is obtained by reasoning as follows:

- If $\mathcal{A}(i+1) \preceq (\mathbf{C}(\mathbf{A}), \top_I)$, then the result follows from $\sigma \sim_{\mathcal{S}_i \setminus 0, \mathcal{A}(i)}^{(\mathbf{C}(\mathbf{A}), \top_I)} \sigma'$, $(\mathcal{S}_i \setminus 0) \leq_{\mathcal{A}(i)} (\mathcal{S}_{i+1} \setminus 0)$ the fact that stack indist. is preserved by subtyping.
- If $\mathcal{A}(i+1) \not\preceq (\mathbf{C}(\mathbf{A}), \top_I)$, the result follows from the previous item and H-LOW.

Second we must check that $v \sim_{\mathcal{S}_{i+1}(0)}^{(\mathbf{C}(\mathbf{A}), \top_I)} v'$. By $s_j \sim^{(\mathbf{C}(\mathbf{A}), \top_I)} s'_j$, we know that $v \sim_{\mathcal{S}_i(0)}^{(\mathbf{C}(\mathbf{A}), \top_I)} v'$. We consider two cases depending on whether $\mathcal{S}_i(0) \preceq (\mathbf{C}(\mathbf{A}), \top_I)$ or $\mathcal{S}_i(0) \not\preceq (\mathbf{C}(\mathbf{A}), \top_I)$. Suppose $\mathcal{S}_i(0) \preceq (\mathbf{C}(\mathbf{A}), \top_I)$.

- if $\mathcal{S}_{i+1}(0) \preceq (\mathbf{C}(\mathbf{A}), \top_I)$, $v \sim_{\mathcal{S}_{i+1}(0)}^{(\mathbf{C}(\mathbf{A}), \top_I)} v'$ follows from $s_j \sim^{(\mathbf{C}(\mathbf{A}), \top_I)} s'_j$ and value indist.;
- otherwise, we conclude directly by value indist.

Suppose now $\mathcal{S}_i(0) \not\preceq (\mathbf{C}(\mathbf{A}), \top_I)$. If $v = v'$ then $v \sim_{\mathcal{S}_{i+1}(0)}^{(\mathbf{C}(\mathbf{A}), \top_I)} v'$ is immediate. But, if $v \neq v'$ and $\mathcal{S}_{i+1}(0) \preceq (\mathbf{C}(\mathbf{A}), \top_I)$, since declassification has occurred, indistinguishability is not a priori guaranteed. Since v, v' are high-integrity data and active attackers cannot modify them (Prop. 4(i)), these same values are at the same program point in the execution of $B[\vec{a}_1]$ (Lem. 8). Then these same data, v, v' , are also declassified by $B[\vec{a}_1]$ at this program point (since **declassify** instructions do not belong to $B[\vec{\bullet}]$). By Lem. 8, if high integrity values not are equal at the same program point, then there exist high integrity data on initial states s.t. they are not equal either. If $v \neq v'$, then the given assumption $\text{Tr}_{B[\vec{a}_1]}(s_1) \sim^{(\mathbf{C}(\mathbf{A}), \top_I)} \text{Tr}_{B[\vec{a}_1]}(s'_1)$ would fail. Hence it must be the case that $v = v'$.

2. Case $\mathcal{A}(s_j), \mathcal{A}(s'_j) \not\preceq (\mathbf{C}(\mathbf{A}), \top_I)$. As a consequence, there exists a program point k in $\text{Dom}(B)^\#$ such that k was the value of the program counter for s_{j-1} and s'_{j-1} . Furthermore, let $\text{pc}(s_j) = g$ and $\text{pc}(s'_j) = g'$ then we have $k \mapsto g$ and $k \mapsto g'$ by the definition of the Successor Relation. By the SOAP properties (1(i)), both $g, g' \in \text{region}(k)$. Furthermore, by T-IF, $\forall d \in \text{region}(k). \mathcal{S}_k(0) \preceq \mathcal{A}(d)$. In particular, $\mathcal{S}_k(0) \preceq \mathcal{A}(s_j), \mathcal{A}(s'_j)$. Therefore, by the minimality assumption in the definition of well-typed programs, we may assume $\mathcal{S}_k(0) \not\preceq (\mathbf{C}(\mathbf{A}), \top_I)$. Hence, $\forall d \in \text{region}(k). \mathcal{A}(d) \not\preceq (\mathbf{C}(\mathbf{A}), \top_I)$. Therefore the hypothesis of Lem. 2 is satisfied.

Now, we repeatedly apply Lem. 2 in both branches of executions until it is no longer possible. Let us call s_{h_1} and s'_{h_2} the reached states. By transitivity and symmetry of indistinguishability of states, $s_{h_1} \sim^{(\mathbf{C}(\mathbf{A}), \top_I)} s'_{h_2}$. By Lem. 2, $\mathcal{A}(s_{h_1}), \mathcal{A}(s'_{h_2}) \not\preceq (\mathbf{C}(\mathbf{A}), \top_I)$. Also, if $h_1 \mapsto h'_1$ and $h_2 \mapsto h'_2$,

then $\mathcal{A}(h'_1), \mathcal{A}(h'_2) \preceq (\mathbf{C}(\mathbf{A}), \top_I)$. We are therefore at a junction point, $h'_1 = \text{jun}(k) = h'_2$. Finally, by Lem. 3, $s_{h'_1} \sim^{(\mathbf{C}(\mathbf{A}), \top_I)} s'_{h'_2}$.

We repeat the argument, namely applying Lem. 1 until no longer possible and then analyzing cases 1 and case 2. Given that the derivations are finite, eventually case 1 is reached.

6 Related work

A brief description of related literature on type based information flow analysis (IFA) for declassification and for low-level languages follows. Zdancewic and Myers [ZM01] introduce robust declassification (RD) in an attempt to capture the desirable restrictions on declassification (as discussed in this paper), but did not propose a method for determining when a program satisfies RD. Zdancewic [Zda03] shows that a simple change to a security type system can enforce it: extend the lattice of security labels to include integrity constraints as well as confidentiality constraints and then require that the decision to perform declassification have high integrity. Myers et al [MSZ04] introduce a type system for enforcing RD and also, they show how to support upgrading (endorsing) data integrity. Barthe et al [BR05] define a simple JVM bytecode type system for IFA. They prove preservation of noninterference for a compilation function from a core imperative language to bytecode. [BPR07] extends the type system to handle objects and method calls. And present a type-preserving compilation for a Java-like language with objects and exceptions. All these works assume fields have a fixed security level. Less importantly, they do not allow variable reuse and have a less precise management of information flow of the operand stack. [BB08] proposes a core bytecode language together with a flow-sensitive type system that performs IFA in a setting where the security level assigned to fields may vary at different points of the program where their host object is created. Rezk et al [BCR08] provide a modular method for achieving sound type systems for declassification from sound type systems for noninterference, and instantiate this method to a sequential fragment of the JVM. They consider a declassification policy called delimited non-disclosure that combines the *what* and *where* dimensions of declassification.

7 Conclusion

We present a type system for ensuring secure information flow in a core JVM-like language that includes a mechanism for performing downgrading of confidential information. It is proved that the type system enforces robustness of the declassification mechanism in the sense of [ZM01]: attackers may not affect what information is released or whether information is released at all. The restriction of the attackers ability to modify data is handled by enriching the confidentiality lattice with integrity levels.

In presence of flow-sensitive type systems we have showed that variable reuse endows the attacker with further means to affect declassification than those available in high-level languages in which variables are assigned a fixed type during their entire lifetime. Additional means of enhancing the ability of an attacker to declassify information in stack-based languages (e.g. bytecode) is by manipulating the stack, as exemplified in Sec. 2. Also, in unstructured low-level languages, we are required to restrict the use of jumps.

Our current efforts are geared toward enriching the core bytecode language while maintaining our results. Also, the inclusion of a primitive for *endorsement* (for upgrading the integrity of data) as studied elsewhere [MSZ04] in the setting of high-level languages would be interesting. Note that although confidentiality and integrity are dual properties, the treatment of `declassify` and `endorse` should not be symmetric for otherwise the attacker would be given total control over the data.

Acknowledgements. To the referees for helping improve this work considerably.

References

- [BB08] Francisco Bavera and Eduardo Bonelli. Type-based information flow analysis for bytecode languages with variable object field policies. In *SAC'08, Proceedings of the 23rd Annual ACM Symposium on Applied Computing. Software Verification Track*, 2008.
- [BB09] Francisco Bavera and Eduardo Bonelli. www.lifia.info.unlp.edu.ar/~eduardo/publications/robustLong.pdf, 2009.
- [BCR08] G. Barthe, S. Cavadini, and T. Rezk. Tractable enforcement of declassification policies. *Computer Security Foundations Symposium, IEEE*, 0:83–97, 2008.
- [BPR07] Gilles Barthe, David Pichardie, and Tamara Rezk. A Certified Lightweight Non-Interference Java Bytecode Verifier. In *Proc. of 16th European Symposium on Programming (ESOP'07)*, Lecture Notes in Computer Science. Springer-Verlag, 2007.
- [BR05] Gilles Barthe and Tamara Rezk. Non-interference for a JVM-like language. In *TLDI '05: Proceedings of the 2005 ACM SIGPLAN international workshop on Types in languages design and implementation*, pages 103–112, New York, NY, USA, 2005. ACM Press.
- [HS06] S. Hunt and D. Sands. On flow-sensitive security types. In *POPL'06, Proceedings of the 33rd Annual ACM SIGPLAN - SIGACT Symposium on Principles of Programming Languages*, January 2006.
- [LY99] Tim Lindholm and Frank Yellin. *The Java(TM) Virtual Machine Specification*. Addison Wesley, 1999.
- [MSZ04] Andrew C. Myers, Andrei Sabelfeld, and Steve Zdancewic. Enforcing robust declassification. In *CSFW*, pages 172–186. IEEE Computer Society, 2004.
- [SM03] A. Sabelfeld and A. Myers. Language-based information-flow security. *IEEE Journal on Selected Areas in Communications*, 21(1), 2003.
- [Zda03] S. Zdancewic. A type system for robust declassification, 2003.
- [ZM01] Steve Zdancewic and Andrew C. Myers. Robust declassification. In *CSFW-14 2001*, pages 5–. IEEE Computer Society, 2001.

Técnicas Anti-Forenses en Informática: Ingeniería Reversa Aplicada a TimeStomp

Armando Botero, Iván Camero y Jeimy Cano

Departamento de Ingeniería de Sistemas, Pontificia Universidad Javeriana,
Carrera 7 No. 40 – 62, Bogotá, Colombia
{armando.botero, icamero, j.cano}@javeriana.edu.co

Resumen. Cada vez más las técnicas de evasión y las vulnerabilidades materializadas por los atacantes son más creativas y sofisticadas. En este contexto y conscientes del reto propio que esto implica para las investigaciones forenses en informática, se presenta en este documento un análisis de una de las herramientas anti-forenses conocidas como lo es timestomp, la cual es analizada en sus detalles e impactos sobre el sistema de archivo NTFS. El documento concluye con evaluación del funcionamiento detallado de timestomp para lo cual se utilizan técnicas de ingeniería reversa ilustrando los puntos clave para su detección y rastreo en NTFS.

Palabras Clave: Métodos Anti-Forenses, Timestomp, Computación Forense, NTFS, Ingeniería Reversa.

1. INTRODUCCIÓN

En la actualidad, para los investigadores de cómputo forense, se presentan retos cada vez más exigentes en cuanto al rastreo y detección de un atacante o “inquieto” [2], dada la alta creatividad de los intrusos en sus técnicas de evasión, las cuales cada vez son más sofisticadas y efectivas en los sistemas informáticos.

El reconocimiento de las vulnerabilidades en las herramientas utilizadas para adelantar procedimientos de informática forense, ha generado la aparición de las llamadas técnicas anti-forense que se definen como: “*cualquier intento de comprometer la disponibilidad de la evidencia para un proceso forense*” [1]. Estas técnicas buscan manipular el material más sensible de una investigación al destruir, ocultar, eliminar y falsificar la evidencia.

Para efectos de este artículo se revisarán las características de la herramienta timestomp, que se especializa en destruir y falsificar evidencia digital aprovechándose de las vulnerabilidades del sistema de archivos NTFS, atacando los atributos de tiempo MACE (Modificado, Accedido, Creado y Entry Modify) de cada archivo [10].

De otra parte, considerando que un método para determinar aspectos de fondo y observar el comportamiento en una aplicación de software o uno de sus componentes es el de la ingeniería reversa (IR) [19], se buscará identificar deficiencias en Timestomp para determinar los posibles rastros de la aplicación de esta herramienta, en sistemas Windows XP.

Para ilustrar el uso y funcionamiento de Timestomp, se presenta un ejemplo práctico, donde se detalla cómo opera dicha herramienta con los Atributos MACE y en general con la MFT (Master File Table). Luego, se realiza el análisis del código fuente de dicha herramienta basada en la ingeniería reversa aplicada sobre la misma.

2. DEFINICIÓN DE TÉCNICAS ANTI-FORENSES

Las herramientas o técnicas anti-forenses se definen según (Harris, 2006) como *“cualquier intento de comprometer la disponibilidad de la evidencia para un proceso forense.”*[4] Del mismo modo si se profundiza un poco más en éste concepto y se desarrolla en términos más técnicos se genera la siguiente definición: “Cualquier intento exitoso efectuado por un individuo o proceso que impacte de manera negativa la identificación, la disponibilidad, la confiabilidad y la relevancia de la evidencia digital en un proceso forense” [6]

Estas técnicas proporcionan a los atacantes una ventaja inusual sobre los investigadores en cómputo forense, ya que al hacerse efectivas sobre la evidencia digital, pueden comprometer fácilmente la confianza y claridad de la misma en un proceso.

Así mismo, sugiere a los investigadores observar con un mayor detalle las evidencias digitales encontradas en una escena del crimen, lo que exige replantear los protocolos para investigaciones pasadas y futuras.

3. CLASIFICACIÓN DE MÉTODOS ANTI-FORENSES

A medida que se explora y se investiga más sobre las técnicas anti-forenses se han generado varias clasificaciones y del mismo modo se han definido varios métodos. Para efectos de este trabajo se tomará la clasificación planteada por (Harris 2006) a saber [7]:

- ✓ Destrucción de la evidencia.
- ✓ Ocultar la evidencia.
- ✓ Eliminación de las fuentes de la evidencia.
- ✓ Falsificación de la evidencia.

La sofisticación y complejidad de cada uno de estos métodos demuestra que los personajes interesados en su creación y ejecución -llamados normalmente intrusos- realizan muchas más cosas y acciones que lo que indican los manuales de los proveedores de software o hardware. [2]

A continuación se establece una aproximación a cada método propuesto por Harris de las herramientas anti-forenses:

✓ *Destrucción de la evidencia:*

El principal objetivo de esta técnica es evitar que la evidencia sea encontrada por los investigadores y en caso de que estos la encuentren, disminuir sustancialmente el uso que se le puede dar a dicha evidencia en la investigación formal. Este método no busca que la evidencia sea inaccesible si no que sea irrecuperable. [7]

Esto implica que se deben destruir, dismantelar o en su defecto modificar todas las pruebas útiles para una investigación (Harris, 2006) [4]. Así como en la vida real cuando ocurre un crimen y el criminal quiere destruir todo rastro o evidencia se vale de una serie de herramientas que le facilitan este objetivo.

Existen dos niveles de destrucción de la evidencia [7]:

- ✓ Nivel Físico: A través de campos magnéticos.
- ✓ Nivel Lógico: Busca reinicializar el medio, cambiar la composición de los datos, sobrescribir los datos o eliminar la referencia a los datos.

Existe una variedad de herramientas para la destrucción de evidencia de las cuales se pueden valer los intrusos para realizar este método anti-forense. Un ejemplo de herramientas son: Wipe, Shred, PGP Secure Delete, Evidence Eliminator y Sswap. [7]

✓ *Ocultamiento de la Evidencia:*

Este método tiene como principal objetivo hacer inaccesible la evidencia para el investigador. No busca manipular, destruir o modificar la evidencia sino hacerla lo menos visible para el investigador. [4]

Esta técnica puede llegar a ser muy eficiente de ser bien ejecutada pero conlleva muchos riesgos para el atacante o intruso, puesto que, al no modificar la evidencia de ser encontrada puede ser válida en una investigación formal y por lo tanto servir para la incriminación e identificación del autor de dicho ataque.

Este método puede valerse de las limitaciones del software forense y del investigador atacando sus puntos ciegos o no usuales de búsqueda de alguna anomalía. [4]

Una de las herramientas utilizadas por los atacantes es la esteganografía la cual versa sobre técnicas que permiten la ocultación de mensajes u objetos, dentro de otros, llamados portadores, de modo que no se perciba su existencia. [8] En el mercado se pueden encontrar muchos instrumentos fáciles de usar, de bajo costo que pueden ayudar a realizar esta técnica anti-forense, como por ejemplo StegoArchive [9].

✓ *Eliminación de la fuentes de la evidencia:*

Este método tiene como principal objetivo neutralizar la fuente de la evidencia, por lo que no es necesario destruir las pruebas puesto que no han llegado a ser creadas. Por ejemplo, en el mundo real cuando un criminal utiliza guantes de goma para utilizar un arma lo que está haciendo es neutralizando y evitando dejar huellas dactilares en el arma. Así mismo en el mundo digital esta neutralización de las fuentes de la evidencia aplica. [4]

Una de las acciones que los atacantes pueden llevar a cabo para realizar este método anti-forense es la desactivación de los log de auditoría del sistema que esté atacando.

✓ *Falsificación de la evidencia:*

Esta método busca engañar y crear falsas pruebas para los investigadores forenses logrando así cubrir a el verdadero autor, incriminando a terceros y por consiguiente desviar la investigación con lo cual sería imposible resolverla de manera correcta.

El ejercicio de este método se vale en una edición selectiva de las pruebas creando evidencias incorrectas y falsas que corrompen y dañan la validez de dichas pruebas en una investigación forense formal, por lo cual no podrán ser tomadas en cuenta como evidencias. [4]

4. FUNDAMENTOS DEL NTFS

Revisando la historia del sistema de archivos NTFS, encontramos que es un sistema de archivos diseñado e implementado por Microsoft, el cual surge como una necesidad para solucionar las fallas de seguridad, desempeño y confiabilidad que el sistema de archivos

FAT poseía [13]. Dichos atributos se optimizan en NTFS al manejar la mayoría de los datos como archivos, de esta manera se hace más sencillo el control de la partición, ya que se tiene un bloque de información de control almacenado en archivos con metadata desde el momento en que la partición es creada, lo que le permite al sistema operativo identificar y localizar cualquier archivo de manera más eficiente [14].

La estructura que maneja un volumen de este sistema de archivos se ilustra en la siguiente imagen.



Figura 1 Estructura del NTFS [5]

- ✓ Cabe resaltar que no todos los datos son archivos dentro de la partición. Por ejemplo, el Partition Boot Sector (PBS) no los maneja, ya que dicho fragmento en la estructura, es el encargado de hacer las operaciones del sistema de archivos.

Para realizar las operaciones del sistema, el PBS se divide en dos sectores; el BIOS Parameter Block (BPB) y el Volume Boot Code (VBC). El BPB es el encargado de describir el formato que tiene la partición y la estructura de datos de metadata y archivos que maneja la capa física del volumen. Así mismo, posee el Boot Code, el cual se encarga de comunicarle al sistema operativo cuales son los recursos con los que la máquina cuenta [16]. Sabiendo las características físicas del computador, el VBC carga el sistema operativo con el código que es único para cada uno.

- ✓ La Master File Table (MFT) actúa como una base de datos relacional en la cual las filas son archivos de historiales y las columnas con archivos de atributos. Todos los archivos de una partición NTFS deben tener por lo menos una ocurrencia dentro de la MFT [12]. Los primeros dieciséis registros de tabla son usados para describirse. A partir del diecisieteavo registro comienzan todos los registros de la partición. El tamaño de los registros se asocia con el tamaño del cluster del volumen, sin embargo, tienen un mínimo de 1024 bytes y un máximo de 4096 bytes, así, si un registro tiene 512 bytes, el tamaño que se le asigna es de 1024 [15]. El ejemplo más claro de este tipo de atributo es el nombre del archivo o su *Time Stamp*. La clasificación de los registros se da por la información que tienen y los tamaños dados, en este contexto se hacen llamar atributos, que pueden ser residentes y no residentes. Anteriormente, se cito un ejemplo de lo que pasa cuando el tamaño de un registro es menor que el mínimo establecido; ese tipo de registros y los que son menores o iguales al máximo establecido, se

consideran residentes ya que se pueden almacenar en una sola tabla. Por otro lado, si un atributo llega a ser mayor de valor máximo estipulado, los clusters restantes se almacenan en una tabla adicional; este tipo se hace llamar atributos no residentes.

Cada registro posee una lista de atributos que se caracteriza no sólo como residentes o no, sino también por su tipo. Los tipos que concretamente atañen la investigación del artículo son:

- Standard Information Attribute (SIA): contiene información sobre modo de acceso, timestamps (marcas de tiempo) y cuenta de acoplamiento
- File Name (FN): posee información sobre nombre del archivo, un atributo repetible para los nombres corto y largo de un archivo.

Una lista detallada de todos los tipos se puede encontrar en [12].

- ✓ Los System Files o archivos de sistema son los archivos formales en los que se almacena información en forma de metadatos [16]. Se encuentran dentro de la MFT y guardan datos que la MFT no.
- ✓ El File Area contiene una copia de la MFT para efectos de recuperación de los datos en caso de problemas con la copia original [17].

5. TIMESTOMP

Timestomp es una herramienta anti-forense que sirve para leer y escribir los registros de tiempo o *time-stamps* de los archivos en NTFS (*New Technology File System*), desarrollada por *Metasploit Anti-Forensics Project* y distribuida de forma gratuita. Junto a herramientas como *slacker*, para ocultar archivos y *samjuicer* para obtener contraseñas de Windows NT/2000/XP/2003 conforman el *Metasploit Anti-Forensic Investigation Arsenal* (MAFIA) [10].

Los registros de tiempo o *time-stamps* de NTFS sirven para almacenar información de cuándo fue modificado (M), accedido (A), creado (C) y *entry modified* (E) un archivo en el sistema NTFS, lo cual se conoce como MACE. La primera sigla, M, se refiere a la fecha y hora del último cambio que se hizo sobre el atributo Datos de la *Master File Table* (MFT) de NTFS, lo que normalmente se conoce como “el archivo” [11], la segunda, A, se refiere a la última vez que el archivo se vio envuelto en una actividad, la tercera C, hace referencia a la fecha y hora en que fue creado, y el último, *entry modified* se refiere al tiempo en que fue modificado por última vez cualquier atributo del archivo dentro de la MFT, como su nombre, metadatos, datos, etc.

Según la clasificación que aparece en [10] y por las características ofrecidas por timestomp, esta es una herramienta anti forense que destruye y falsifica información.

Con *timestomp* se consigue evitar que un analista forense obtenga una línea de tiempo de sucesos en un sistema, dificultando la correlación de eventos y desacreditando la evidencia digital. Todo lo anterior perturba la etapa de análisis de datos dentro del proceso digital forense.

6. INGENIERÍA REVERSA EN APLICACIONES DE SOFTWARE

En ciencias de la computación, la ingeniería reversa (IR) se define como un proceso de análisis de un software o hardware para poder identificar sus componentes y cómo se relacionan entre sí y así crear una abstracción diferente o mayor, a la que el sistema presentaba originalmente [18]. Ya que se pretende aplicar alguna metodología de IR a Timestomp, en esta sección presentará de manera breve lo relacionado con la IR en las aplicaciones.

Definir una metodología específica para llevar a cabo la IR en el software no es labor sencilla, ya que es un proceso relativo a las entradas y lo que se espera como salida. Dentro de las entradas encontramos [19]:

- ✓ Código fuente
- ✓ Documentación
- ✓ Material de Pruebas
- ✓ Diagramas de diseño y arquitectura.
- ✓ Manuales de usuario, instalación y mantenimiento.
- ✓ Entrevistas con los desarrolladores del sistema a analizar.

Como posibles salidas encontramos [20]:

- ✓ Nuevos diagramas.
- ✓ Re-Ingeniería en el código fuente.
- ✓ Nueva documentación
- ✓ Bases de datos modificadas.
- ✓ Informes analíticos sobre falencias o fallos encontrados.

Para delimitar el problema de tener una amplia gama de posibles mezclas de entradas y salidas, se usará como entrada el código fuente de Timestomp para obtener como salida nueva documentación que pueda mostrar el funcionamiento de esta herramienta desde una perspectiva de la informática forense.

Para nuestro caso de estudio, se hará una exploración detallada y una descomposición en subrutinas, que busca encontrar en qué funciones se generan rastros que son el resultado

del uso de esta herramienta por un atacante, rastros con los cuales se pretende determinar la ejecución de una técnica anti-forense.

En conclusión, la salida que se espera al culminar el proceso de IR, es un informe analítico que refleje cómo TimeStomp dejó algún rastro en los atributos MACE modificados.

7. ANÁLISIS DE TIMESTOMP CON IR

Cómo se describió anteriormente TimeStomp se vale de la alteración de los atributos MACE, los cuales se ubican en la MTF (Master File Table) de un sistema de archivos NTFS. Esta herramienta se enfoca específicamente en los Standard Information Attribute (SIA) de la MTF, donde se encuentran los Timestamp (estampillas de tiempo) que hacen referencia al momento de tiempo actual de un acontecimiento registrado por un computador [21].

Sin embargo, estos no son los únicos registros de tiempos existentes, puesto que hay otros almacenados en el File Name Attribute (FN) [11], [12] y que no son modificados por la herramienta *TimeStomp* lo que sugiere una posible debilidad de esta aplicación. La diferencia entre los registros de tiempo del SIA y los de FN radica en que en SIA la información MACE se modifica cada vez que llevamos a cabo alguna acción sobre un archivo como accederlo o modificarlo, en FN la información solamente se modifica cuando creamos o movemos al archivo de una ruta a otra [10]. Por lo tanto, las fechas y horas MACE registradas en el atributo FN - en la mayoría de los casos - va a ser más antigua a las registradas en el atributo SIA ya que las operaciones que modifican éste atributo se realizan con mayor frecuencia con respecto a los que cambian el FN.

Siguiendo con el análisis de la herramienta y dejando la anterior afirmación como inquietud a resolver más adelante, se tiene que la estructura general de los SIA es representada en la tabla 1, donde los atributos más importantes para el análisis que se está efectuando son: File Creation Time, File Alteration Time y File Read File; que son los que usa TimeStomp.

0x00	8	<u>File Creation Time</u>
0x08	8	<u>File Alteration Time</u>
0x10	8	MFT Change
0x18	8	<u>File Read Time</u>
0x20	4	DOS File Permissions
0x24	4	Maximum number of versions
0x28	4	Version number
0x2C	4	Class ID
0x30	4	2K Owner ID

Tabla 1 Estructura de SIA 22

Para poder manipular y modificar los atributos arriba mencionados, TimeStomp utiliza principalmente las funciones NtQueryInformationFile(), NtSetInformationFile() de la librería ntdll.dll que proporciona Windows, las cuales son una colección de comandos que pueden ser compartidos por diferentes programas en una misma plataforma [23].

Dichas funciones se utilizan para modificar (NtSetInformationFile()) y para consultar NtQueryInformationFile() los File Standard Information donde se encuentran los atributos MACE.

La utilización de estas funciones se evidencia en el código fuente de TimeStomp, específicamente en sus principales rutinas **SetFileMACE** y **RetrieveFileBasicInformation** [26]. Las cuales cargan la librería ntdll y hacen el respectivo llamado a las funciones que realizan el trabajo de modificar los atributos MACE mencionadas anteriormente.

Una de las principales ventajas de timestomp radica en el uso de funciones de la librería ntdll.dll, ya que esta puede ser invocada por cualquier programa en modo usuario, es decir sin necesidad de privilegios [24], como lo muestra la figura 2. Además la utilización y llamado a estas funciones no queda registrada por el sistema, lo que implica que no se genera evidencia del uso de dichas funciones.

Asimismo al valerse de la ntdll.dll, no hace necesario que timestomp inyecte nuevas librerías dll o APIs, porque las que utiliza son proporcionadas por el mismo sistema comprometido.

Estas ventajas conllevan a la disminución de la visibilidad que generaría la utilización de timestomp, a los ojos de una investigación de cómputo forense. Además se puede observar que el desarrollador de esta herramienta, posee un nivel medio de sofisticación y conocimientos técnicos puesto que el uso de la librería ntdll está prácticamente sin documentación [24].

```
ntdll = LoadLibrary("ntdll.dll");
if (ntdll == NULL) {
    return 0;
}
```

Figura 2. Invocación Librería ntdll.dll [26]

A continuación se realizará un estudio detallado de las funciones del código fuente de la herramienta.

8.1. Estructura general de la clase timestomp.c

Las estructuras declaradas dentro del código y las funciones dentro de él son las siguientes:

```
DWORD ParseDateTimeInput(char *inputstring, SYSTEMTIME *systemtime);
HANDLE RetrieveFileBasicInformation(char *filename, FILE_BASIC_INFORMATION *fbi);
DWORD ConvertLocalTimeToLargeInteger(SYSTEMTIME localsystemtime, LARGE_INTEGER *largeinteger);
DWORD ConvertLargeIntegerToLocalTime(SYSTEMTIME *localsystemtime, LARGE_INTEGER largeinteger);
DWORD SetFileMACE(HANDLE file, FILE_BASIC_INFORMATION fbi);
DWORD TheCraigOption(char *directoryname);
```

Figura 3. Funciones de TimeStomp [26]

```

typedef struct _IO_STATUS_BLOCK {
    union {
        NTSTATUS Status;
        PVOID Pointer;
    };
    ULONG_PTR Information;
} IO_STATUS_BLOCK, *PIO_STATUS_BLOCK;

typedef enum _FILE_INFORMATION_CLASS {
    FileBasicInformation = 4,
    FileStandardInformation = 5,
    FilePositionInformation = 14,
    FileEndOfFileInformation = 20,
} FILE_INFORMATION_CLASS, *PFILE_INFORMATION_CLASS;

typedef struct _FILE_BASIC_INFORMATION {
    LARGE_INTEGER CreationTime;
    LARGE_INTEGER LastAccessTime;
    LARGE_INTEGER LastWriteTime;
    LARGE_INTEGER ChangeTime;
    ULONG FileAttributes;
} FILE_BASIC_INFORMATION, *PFILE_BASIC_INFORMATION;
// Created
// Accessed
// Modified
// Entry Modified

```

Figura 4. Estructuras de TimeStomp [26]

En las siguientes sub-secciones se mostrará como las funciones *SetFileMACE*, *RetrieveFileBasicInformation* y *TheCraigOption* interactúan con el sistema. Las funciones restantes al no acceder relevantemente a la MFT se describirán brevemente a continuación.

- *ParseDateTimeInput*: esta función convierte una cadena de caracteres al tiempo manejado por el sistema operativo (SYSTEMTIME) en el formato adecuado, retorna 1 si la conversión fue exitosa y 0 en el caso contrario.
- *ConvertLocalTimeToLargeInteger*: ya que los timestamps de un archivo son almacenados como tiempo universal coordinado o UTC, para hacer operaciones sobre estos datos es necesario convertirlos a un formato que lo permita. Esta función se encarga de transformar los UTC a un LARGE_INTEGER para realizar operaciones sobre ellos.
- *ConvertLargeIntegerToLocalTime*: realiza la operación inversa a la anterior.

8.2. SetFileMACE

- ✓ Aspectos importantes vistos desde la informática forense: *SetFileMACE()* es la función clave de TimeStomp ya que es la encargada de realizar la modificación de los atributos MACE principal objetivo de esta herramienta. Al analizar su funcionamiento, se observa que las funciones *LoadLibrary()*, *GetProcAddress()* y *FreeLibrary()* proporcionadas por el API de Windows [27] y *NtSetInformationFile()* que proporciona *ntdll.dll* no dejan ningún registro o evidencia de la utilización o llamados por parte de *SetFileMACE()* a estas;

además Timestamp al utilizar librerías propias del sistema operativo y el sistema de archivos, evita inyectar dll's diferentes que podrían dejar algún rastro y evidenciar de esta herramienta.

8.3. RetrieveFileBasicInformation

- ✓ Aspectos relevantes vistos desde la informática forense: El principal objetivo de esta función es almacenar los datos retornados NtQueryInformationFile() en la estructura FILE_BASIC_INFORMATION, los cuales son tomados directamente de los SIA de la MFT. Sin embargo, esto no se puede tomar como evidencia ya que esta función es directamente tomada de la librería ntdll.dll y no deja rastro alguno de su uso.

8.4. TheCraigOption

- ✓ Aspectos relevantes vistos desde la informática forense: Esta función trabaja principalmente con la estructura WIN32_FIND_DATA [28], que contiene toda la información importante de un archivo, de la que hace parte los atributos MACE. Asimismo valiéndose de esta estructura utiliza sus funciones, tales como FindFirstFile(), FindNextFile() y FindClose() para lograr el objetivo de eliminar los valores MACE en un análisis hecho con la herramienta de investigación forense EnCase. La utilización de estas funciones no deja rastro alguno, así que es imposible para EnCase conocer y darse cuenta de la utilización de estas, además está atacando los atributos de SIA, que son precisamente los que valida y utiliza para sus informes y generación de líneas de tiempo la herramienta EnCase.

8. VULNERABILIDADES DE TIMESTOMP

Timestamp presenta una debilidad que puede ser aprovechada por los analistas forenses para evidenciar se ha usado este programa y conforme a esto responder a las circunstancias apropiadamente.

En la figura 6 se muestra una imagen tomada con la herramienta EnCase, en la cual se ilustra una MFT al crear un archivo cualquiera. En ella se ven los valores almacenados en el SIA y el FN del archivo, junto con sus respectivos atributos MACE. Posterior a la creación del archivo, se accedió al archivo y se realizaron una serie de modificaciones con el fin de ilustrar que dichos cambios se almacenan únicamente en el SIA, más no en el FN.

Es por esto, que el principal defecto de timestamp radica en la modificación de la información MACE del SIA y no del FN dentro de la MFT, por tanto en condiciones normales, las fechas y horas registradas en SIA deben ser mayores o iguales a las

registradas en FN, entonces es posible asegurar que han cambiado las fechas del SIA y sospechar que han usado timestomp, al no ver el cambio en el FN.

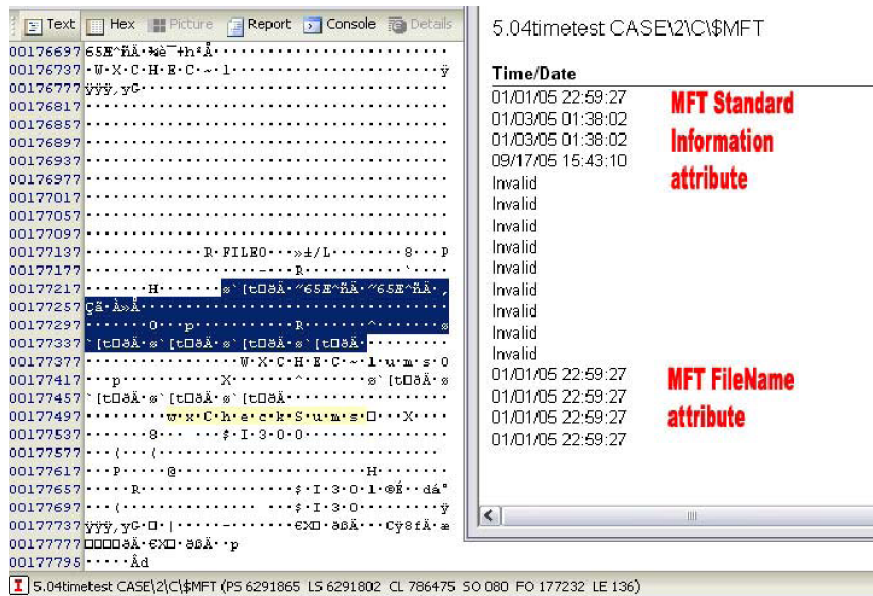


Figura 6. Estructuras de TimeStomp [29]

Una potencial solución a esta debilidad de timestomp consiste en modificar los valores FN con una herramienta que permita alterar datos “crudos” como WinHex [10], de tal forma se evita la inconsistencia en los datos del tiempo.

9. CONCLUSIONES

- ✓ A medida que las técnicas y tecnología que utilizan los atacantes informáticos evolucionan, la administración de justicia requiere mayores elementos de análisis e investigación, para identificar y determinar quien cometió el ataque.
- ✓ El trabajo de la computación forense se hace cada vez más complejo ya que con la utilización de las técnicas anti-forenses se está atacando y vulnerando tanto la evidencia como las herramientas utilizadas.
- ✓ A la fecha no existe un modelo certificado y unificado con el que los investigadores forenses cuenten para la identificación y análisis que permita determinar que se están utilizando herramientas anti-forenses.

- ✓ Timestamp utiliza funciones de la librería ntdll.dll las cuales no dejan ningún registro o rastro de su utilización, lo que dificulta el comprobar si un archivo fue intervenido con esta herramienta o no.
- ✓ Las herramientas de cómputo forense podrían evaluar y analizar los atributos FN y compararlos con los SIA, para realizar las líneas de tiempo y así minimizar las inconsistencias que se puedan presentar.

10. REFERENCIAS

1. Arckoff R., Addison H. (2007), Management F-Law. How Organizations Really Work.
2. Jeimy. J. Cano. (2007). Inseguridad informática y computación anti-forense: Dos conceptos emergentes en seguridad de la información. <http://www.virusprot.com/Archivos/Antifore07.pdf>
3. Jeimy. J. Cano. (2007). Introducción a las técnicas anti forenses: Conceptos e implicaciones para investigadores. http://www.acis.org.co/fileadmin/Base_de_Conocimiento/VII_JornadaSeguridad/VIIJNSI_JCano.pdf
4. R.Harris. (2006). Arriving at an anti-forensics consensus: Examining how to define and control the anti-forensics problem. <http://dfrrs.org/2006/proceedings/6-harris.pdf>
5. López, Oscar; Amaya, Haver; León Ricardo; Acosta Beatriz. (2002). Informática Forense: Generalidades, aspectos técnicos y herramientas. http://www.criptored.upm.es/guienteoria/gt_m180b.htm
6. Jeimy. J. Cano. (2007). Introducción a la informática forense. http://www.acis.org.co/fileadmin/Revista_96/dos.pdf
7. Andres R. Almanza. (2007). Ciencias Anti-forense. Un nuevo reto para las organizaciones. http://www.acis.org.co/fileadmin/Base_de_Conocimiento/VII_JornadaSeguridad/VIIJNSI_AAlmanza.pdf
8. Info Seguridad: Esteganografía. <http://www.infoseguridad0.es/Estenografia.htm>
9. Stegoarchive. What is Steganography?. <http://www.stegoarchive.com/>
10. Metasploit Anti-Forensics Project. http://www.metasploit.com/data/antiforensics/BlueHat-Metasploit_AntiForensics.ppt
11. MicrosoftTech. NTFS TimeStamps. <http://blogs.technet.com/ganand/archive/2008/02/19/ntfs-time-stamps-file-created-in-1601-modified-in-1801-and-accessed-in-2008.aspx>
12. MicrosoftTech. How NTFS Works. <http://technet.microsoft.com/en-us/library/cc781134.aspx>
13. PCGuide. Overview and History of the NTFS. <http://www.pcguides.com/ref/hdd/file/ntfs/ver.htm>

14. Ariza, A., Ruiz, J., Análisis de Metadatos en archivos Office y Adobe. (2008). http://www.criptored.upm.es/guiateoria/gt_m142g1.htm
15. PCGuide. NTFS file Attributes. http://www.pcguides.com/ref/hdd/file/ntfs/files_Attr.htm
16. PCGuide. NTFS System (Metadata) Files. http://www.pcguides.com/ref/hdd/file/ntfs/arch_Files.htm
17. Microsoft Corporation. How NTFS works. <http://technet.microsoft.com/en-us/library/cc781134.aspx>
18. Chikofsky, E., Cross, J., Reverse Engineering and Design Recovery: A Taxonomy, IEEE Software, pp. 13-17, 1990.
19. Galen Lab. Reverse Engineering. <http://calla.ics.uci.edu/reveng/>
20. Electronic Design. Reverse Engineering. <http://electronicdesign.com/Articles/Index.cfm?AD=1&ArticleID=11966>
21. Müller H., Jahnke J., Smith D., Storey M., Tilley S., Wong K.. Reverse Engineering. A Roadmap. (2005). <http://www.cs.ucl.ac.uk/staff/A.Finkelstein/fose/finalmuller.pdf>
21. What is: The Leading TI encyclopedia and learning Center. What is TimeStamp?. http://whatis.techtarget.com/definition/0,,sid9_gci817089,00.html
22. Santa Clara University, School of Engineering. NTFS File System. http://www.cse.scu.edu/~tschwarz/coen152_05/PPtPre/NTFSFS.ppt
23. Help with PCs. DLL. <http://www.helpwithpcs.com/jargon/dll.htm>
24. Win32 API Obscurity for I/O Blocking and Intrusion Prevention (2005). <http://www.ddj.com/security/184406098>
25. Metasploit Anti-Forensics Project. http://www.metasploit.com/data/antiforensics/BH2005-Catch_Me_If_You_Can.ppt
26. Metasploit Anti-Forensics Project. Timestamp codigo fuente. <http://trac.metasploit.com/browser/framework3/trunk/external/source/meterpreter/source/extensions/priv/server/timestamp.c?rev=6357>
27. Microsoft Developer Network. Windows API. <http://msdn.microsoft.com/en-us/library/aa383750.aspx>
28. Microsoft Developer Network. WIN32_FIND_DATA. [http://msdn.microsoft.com/en-us/library/aa365740\(VS.85\).aspx](http://msdn.microsoft.com/en-us/library/aa365740(VS.85).aspx)
29. Black, G., Evidence of Folder Renaming using MFT Standard Information Attribute and FileName Attribute. (2005). http://www.geoffblack.com/forensics/Evidence_of_Folder_Renaming.pdf

Aplicar el Modelo de Amenazas para incluir la Seguridad en el Modelado de Sistemas

Marta Castellaro, Susana Romaniz, Juan Carlos Ramos, Carlos Feck, Ivana Gaspoz
Argentina, Universidad Tecnológica Nacional, Facultad Regional Santa Fe
mcastell@frsf.utn.edu.ar - sromaniz@frsf.utn.edu.ar - jramos@frsf.utn.edu.ar
Lavalse 610 – 3000 Santa Fe -Argentina, Tel, Fax: +54 .0342.4690348

Resumen. Hasta hace poco, la seguridad era concebida como una propiedad deseable del software y formaba parte del conjunto de atributos a considerar para determinar la calidad del mismo. El aumento de ataques a aplicaciones vulnerables ha conducido a un reconocimiento gradual que las protecciones a nivel de infraestructura (hoy de uso común para proteger sistemas accesibles por Internet) no son suficientes para tal objetivo. Se habla de “producir software seguro” y esto es concebido como una disciplina, que si bien se presenta menos madura que otras relacionadas con la calidad, cuenta con iniciativas, enfoques, metodologías y tecnologías emergentes. Este trabajo reúne algunos resultados de análisis respecto a terminología y metodologías sobre desarrollo de software seguro, y herramientas para instrumentarlo en las actividades de modelado y diseño de aplicaciones, cuando el equipo de desarrollo está constituido por profesionales no expertos en seguridad.

Palabras claves. Seguridad. Modelado de amenazas. Diseño de Software Seguro.

1.Introducción

Hasta hace poco tiempo, la seguridad era concebida con una propiedad deseable para el aseguramiento del software y formaba parte del conjunto de atributos a considerar para determinar la calidad de un software. El aumento de ataques de aplicaciones vulnerables ha conducido a un reconocimiento gradual respecto a que las protecciones a nivel de infraestructura (hoy de uso común para proteger sistemas accesibles por Internet) no son suficientes para tal objetivo. Se reconoce que las protecciones y mitigaciones de seguridad de las aplicaciones deben ser especificadas a nivel del sistema y su arquitectura, e implementadas durante el desarrollo de las aplicaciones y su operación.

En los últimos años esto ha evolucionado, y la seguridad del software es considerada como una característica del software que puede ser consistentemente demostrada [1]. La garantía de producir software seguro es concebida como una disciplina, y si bien se presenta menos madura que otras relacionadas con la calidad, se cuenta con iniciativas, enfoques, metodologías y tecnologías emergentes. Nuevos conceptos aparecen asociados al software en riesgo: vulnerabilidades, amenazas, ataques y ello conlleva a una taxonomía específica,

clasificaciones y definiciones de patrones.

Existen diferentes definiciones acerca de ‘Aseguramiento del Software (Software Assurance)’ establecidas por organismos reconocidos (CNSS-Committee on National Security Systems; NASA-National Aeronautics and Space Administration; DHS-Department of Homeland Security; NIST-National Institute of Standards and Technology), pero todas ellas presentan un criterio común: “La seguridad se presenta en todo el ciclo de vida del software. Un software seguro es diseñado, implementado, configurado y operado de manera que cumpla con las propiedades esenciales: continúe operando en presencia de ataques, aisle o limite los daños y se recupere lo antes posible”.

2. Ingeniería de Sistemas Seguros

El software es típicamente un elemento o parte de un sistema más grande, sea este un sistema intensivo en software, o un sistema compuesto por elementos de hardware y software. Por ello la Ingeniería de Sistemas ha evolucionado desarrollando publicaciones y técnicas sólidas para sistemas, orientados a la construcción de sistemas seguros. Todo ello ha conformado la “Ingeniería de Sistemas Seguros”, una disciplina que trata sobre la construcción de sistemas que deben permanecer funcionando como se espera ante la maldad, el error, o el azar. Como toda disciplina, se enfoca en los instrumentos, procesos, y métodos que se emplean para diseñar, poner en práctica, y probar sistemas completos, y adaptar sistemas existentes a su ambiente.

Las actividades de ingeniería de sistemas involucradas en la construcción de sistemas seguros, son: (1) El desarrollo de requerimientos de seguridad de sistemas; (2) El desarrollo de diseños de sistemas seguros; (3) La integración de componentes de subsistema; (4) Las pruebas de sistemas y subsistemas. En este trabajo nos centramos en las dos primeras actividades, incluyendo en el diseño las actividades de planificación de pruebas, por lo que luego las discutiremos en más detalle.

2.1 Desarrollo de requerimientos de software seguro

Se reconoce que muchos problemas de seguridad de software provienen de la especificación inadecuada o inexacta de los requerimientos o del desajuste entre la interpretación de los mismos durante la reconstrucción y su intención real.

Los requerimientos para las funcionalidades de seguridad en sistemas intensivos de software a menudo son confundidos con requerimientos para el software seguro.

La primera categoría incluye las funciones que ponen en práctica una política de seguridad (funciones de control de acceso, identificación, autenticación y autorización, y las funciones que realizan el cifrado y el descifrado, y la gestión de claves) y también las funciones que previenen la violación de las propiedades de seguridad del sistema o la información que procesa, como el acceso no autorizado, la modificación, la negación de servicio, descubrimiento, etc. Se conocen también como **‘requerimientos de servicios de seguridad’**.

La segunda categoría de requerimientos afecta directamente a la probabilidad que el software en sí mismo sea seguro y, en general, son no funcionales (o propiedades). En conjunto, aseguran que el sistema ha de permanecer funcionando correctamente incluso cuando se encuentre bajo amenaza; están orientados a la reducción o eliminación de vulnerabilidades en el software y están vinculados principalmente al plan de desarrollo de software y al control de gestión del proyecto. Se conocen como **‘objetivos de seguridad’** [2].

Durante la obtención de requerimientos se debe recolectar “información relativa a la seguridad” tal como: (a) determinar y valorar los **activos**, definiendo qué se va a proteger (activos tangibles o intangibles) y determinando su valor (económico y no económico); (b) determinar los **actores**, dueños de los datos; (c) determinar los **casos de uso**, roles de cada actor; (d) determinar los requerimientos **legales y de negocios** (restricciones legales y de negocios, requerimientos de auditoría y no repudio, políticas de seguridad); (e) determinar las **amenazas** sobre los activos identificados, su probabilidad de ocurrencia y la política de manejo; (f) definir la **arquitectura** del sistema, entendiendo cómo se integran los mecanismos de seguridad al sistema, cómo se maneja la confianza, y cómo se comportará el sistema frente a un ataque (se debe especificar un modelo a alto nivel de amenazas y referencias a los documentos que describen los procedimientos de seguridad; también se deben enumerar las medidas de seguridad a implementar).

Durante el análisis de requerimientos se pueden identificar diversas características que derivan en los requerimientos de seguridad del software, por ejemplo: (a) arquitectura de la aplicación; (b) plataforma a emplear; (c) requisitos de cumplimiento de normativas y marcos regulatorios; (d) tipo de conectividad; (e) tipos de datos (confidenciales, públicos, etc.) que se almacenan o transmiten; (f) tipo de almacenamiento de datos (local o remoto, centralizado o distribuido, etc.); (g) roles de usuarios necesarios para la aplicación; (h) tipo de acceso a los datos por parte de cada rol; (i) acciones sobre el sistema que puede hacer cada rol (cambiar la configuración, arrancar o detener servicios, etc.); (j) tipo de registro que el sistema debe generar (de acceso a recursos, de cambio de configuraciones, de cambio de privilegios, etc.).

Luego es necesario ‘identificar’ los requerimientos de seguridad. Los usuarios pueden no ser totalmente conscientes de los riesgos de seguridad y las vulnerabilidades asociadas con su sistema. Para esta tarea, los ingenieros de sistemas en conjunto con los usuarios, pueden realizar un análisis descendente de los posibles fallos de seguridad que podrían causar riesgo a la organización, así como definir exigencias para identificar vulnerabilidades.

Una técnica que favorece este análisis descendente es el análisis de **árbol de amenaza** (a veces llamado árbol de ataque). En este análisis, se coloca el objetivo del atacante en lo alto del árbol y luego el analista identifica posibles alternativas para alcanzar dicho objetivo. Para cada alternativa, el analista recurrentemente puede añadir alternativas para alcanzar los sub-objetivos que componen el objetivo principal. Este proceso se repite para cada objetivo de atacante. Examinando los nodos de nivel más bajos del árbol de ataque resultante, el analista puede identificar las técnicas posibles para violar la seguridad del sistema, y así se pueden especificar como requerimientos de seguridad las prevenciones para estas técnicas.

Otras técnicas que apoyan el desarrollo de los requerimientos de seguridad son el **modelado de amenazas** y el desarrollo de **casos de mal uso y casos de abuso** [3].

Los resultados del análisis de requerimientos de seguridad del sistema se deben usar como la base para los escenarios de casos de prueba de seguridad que van a ser empleados durante la integración o pruebas de la aceptación.

2.2 Desarrollo de diseños de sistemas seguros

Respecto al modelado, en esta etapa se realiza una aproximación para descubrir y aprender sobre los requerimientos del software, lo que proporciona un modo de prever los funcionamientos y las interacciones del software propuesto dentro de su ambiente. Por

lo tanto, el desarrollo de software seguro requiere modelar las posibles amenazas explícitamente. Existen herramientas y técnicas que facilitan el modelado de amenazas, ataques y vulnerabilidades.

En el diseño de sistemas seguros, se deben incorporar varios rasgos claves de diseño para orientar la identificación de vulnerabilidades típicas de sistemas. Son ejemplos ilustrativos: el diseño de protocolo de seguridad, el diseño de control de acceso y contraseña, el control de publicaciones en sistemas distribuidos, el control de concurrencia, la tolerancia a fallos, y la recuperación de fallos; algunos están asociados a aspectos funcionales, pero existen otros que no están directamente relacionados con la funcionalidad de seguridad.

Para el diseño detallado se pueden identificar patrones de diseño. Así como existen patrones de diseño para problemas recurrentes de sistemas de información, existen también patrones de diseño de seguridad, que son esencialmente buenas prácticas puestas en forma de plantilla.

Los ‘**patrones de ataques**’ se estudian y documentan para sistemas operativos, dispositivos de conectividad, sistemas de gestión de datos, usos y servicios de web, si bien aún no han sido establecidos para ataques específicos. Se emplean en la educación de desarrolladores, en la realización de evaluaciones de riesgo y la definición de modelos de amenaza para sistemas de software [4]. Hay numerosos trabajos destinados a enumerar patrones de ataques donde el objetivo es el software (más que los datos o las redes) [5].

2.3 Desarrollo de pruebas de sistemas seguros

En la referencia [6] se recomienda aumentar el equipo de pruebas de sistemas mediante la inclusión de pruebas independientes por un tercero desinteresado, dado que los atacantes no están en general influenciados por el conocimiento de diseño de sistemas o de mecanismos de protección de seguridad.

Las pruebas para descubrir defectos de diseño son difíciles de desarrollar. El grupo de pruebas (independiente o no) debe ser capaz de construir casos de prueba basados en el entendimiento de la psicología de los atacantes y el conocimiento de software típico, hardware, y otros tipos de defectos de sistemas. Las fuentes adicionales de información para el desarrollo de casos de prueba incluyen: (a) casos de mal uso y casos de abuso; (b) reportes del análisis de árbol de amenaza; (c) modelos de amenaza; (d) modelos de política de seguridad; (e) objetivos de seguridad; (f) requerimientos de seguridad del sistema.

La prueba de resistencia del diseño del sistema ante ataques debe incluir, como mínimo: (a) probar la capacidad del sistema para resistir a adivinación de contraseña, máscaras, etc.; (b) probar defectos transitorios, como una combinación insólita o la secuencia de acontecimientos, degradación del ambiente de operaciones (saturación temporal de la red, pérdida de control, cambios ambientales), o la pérdida temporal de sincronización inducida entre los componentes de un sistema; (c) pruebas “creativas”.

3. Amenazas-Ataques

3.1 Categorías y Fuentes de Amenazas

Los ataques orientados a explotar vulnerabilidades de software han aumentado exponencialmente con la coincidencia de proliferación de sistemas intensivos de software, servicios accesibles a través de la Internet, dispositivos como teléfonos móviles y sistemas de

posicionamiento global. Se espera que todos estos sistemas funcionen continuamente, sin interrupciones. Las amenazas a este funcionamiento son muchas y de muy variadas fuentes. A continuación, en la Tabla 1 se presenta una categorización simple de las mismas, y luego un conjunto inicial de fuentes identificadas.

Tabla 1. Categorías de amenazas.

Categoría de amenaza	Descripción	Propiedad comprometida
Sabotaje	La ejecución del software se suspende o termina, o su funcionamiento se degrada, o el ejecutable se suprime o destruye	Disponibilidad
Cambio de versión	El software ejecutable se modifica intencionadamente (cambio o corrupción) o se substituye por parte no autorizada, o se inserta en el ejecutable lógica no autorizada (código malicioso)	Integridad
Intercepción	Una entidad no autorizada accede al software o a una función restringida dentro del mismo	Control de acceso
Descubrimiento	Se revelan aspectos tecnológicos del software y detalles de puesta en práctica empleando ingeniería reversa (por ej.: descompilación, desmontaje)	Confidencialidad

Las “fuentes de amenazas” al software se pueden agrupar en las siguientes categorías generales: (a) atacantes externos; (b) personas maliciosas que intencionadamente abusan del software; (c) usuarios, no maliciosos, que intencionadamente emplean mal el software, sea porque encuentran limitaciones del uso correcto para obtener lo que están buscando, o simplemente por curiosidad sobre cómo el software podría responder a ciertas entradas o acciones; (d) usuarios benignos que por casualidad (generalmente un malentendido) incluyen una entrada o realizan una acción que duplica un patrón de ataque (incidentes accidentales), y si bien son involuntarios, su resultado es el mismo que el de ataques intencionales.

3.2 El Modelado de Amenazas (‘Threat Modeling’ - TM)

La tarea de identificar los riesgos y las amenazas en el software requiere un esquema estructurado y repetible, que mejore el entendimiento de los involucrados en el desarrollo, ayude a identificar problemas de diseño e integración de componentes, y sirva como input para el equipo de pruebas. El TM constituye un framework específico para el proceso de análisis de riesgo estructurado, que permite identificar las amenazas de una aplicación y cuantificar los riesgos a los que la misma estará expuesta.

El **proceso de TM** consta de las siguientes etapas [7]: (1) conformar un grupo de análisis de riesgo; (2) descomponer la aplicación e identificar componentes claves; (3) determinar las amenazas a cada componente de la aplicación; (4) asignar un valor a cada amenaza; (5) decidir cómo responder a las amenazas; (6) identificar las técnicas y tecnologías necesarias para mitigar los riesgos identificados.

Si bien este proceso constituye un esquema estructurado, se reconoce [8] que el TM se presenta como un conjunto complejo y multidimensional de compensaciones que podrían ser hechas para encontrar un juego de objetivos implícitos o explícitos.

Desde **distintas dimensiones** pueden observarse diferentes enfoques o empleos:

- *Respecto al término ‘amenaza’*: se utiliza tanto para referirse a los posibles “ataques” intencionales, como al “riesgo” que se corre cuando alguien se equivoca.
- *Respecto a las acciones*: mediante el modelado de amenazas se puede referir a una técnica para obtener requerimientos (qué amenazas se van a modelar) o a técnicas de análisis y diseño (cómo se analizan y reflejan esas amenazas en el modelo)
- *Respecto al foco*: el modelado puede ser ‘centrado en el activo’ (implica algún nivel de evaluación de riesgo, aproximación o clasificación), ‘centrado en el atacante’ (implica la clasificación de riesgo o intenta estimar recursos, capacidades o motivaciones, y conlleva a determinar escenarios de amenazas) o ‘centrado en el software’ (análisis de software, redes de organización o sistemas y conlleva al modelado de protocolos y de amenazas de red).
- *Respecto a los recursos humanos*: el modelado de amenaza puede ser hecho por expertos de seguridad, en colaboración de ingenieros con expertos, o por ingenieros sin expertos disponibles.

Se concluye que el modelado de amenaza abarca una amplia variedad de actividades en la definición de requerimientos de seguridad y análisis de diseños de seguridad. No hay un único modo ‘mejor’ o ‘correcto’ de modelado de amenaza.

El objetivo de nuestra investigación es la incorporación de la seguridad como atributo transversal, especialmente en el desarrollo de aplicaciones típicas, por equipos de trabajo pequeños, de profesionales no expertos en ataques. Por ello, nuestro trabajo de análisis realizó un recorte (basado en estos aspectos) definiendo como marco: a) las amenazas se refieren a riesgos; b) se enfoca en las etapas de análisis y diseño; c) está centrado en los activos; d) es ejecutado por equipos de trabajo que conocen la aplicación a desarrollar pero no cuentan con expertos.

Además, nos hemos centrado inicialmente en las fases de requerimientos, diseño arquitectónico y diseño detallado, que son etapas tempranas y donde consideramos que los profesionales no expertos deben contar con apoyo para contemplar la seguridad.

4. Seguridad y los procesos de ciclo de vida del software

Una metodología de desarrollo de software que incorpore mejoras en el sentido de incluir la seguridad, debe proporcionar un marco integrado, o en algunos casos la dirección fase-por-fase para promover el desarrollo de la seguridad de software en todas las fases de ciclo de vida. La integración de actividades de seguridad en los procesos de ciclo de vida de la ingeniería de software implica la adición de prácticas de seguridad (ejemplo: pruebas de penetración) y de principios (ejemplo: un diseño que hace cumplir la menor parte de privilegio), a las actividades y prácticas (como la ingeniería de requerimientos, el modelado, y pruebas a base de modelo) en cada fase del ciclo de vida de desarrollo de software tradicional. Si bien ha habido una investigación activa al respecto, no existe una metodología de la ingeniería de software, que permita asegurar que la seguridad existe en el desarrollo de sistemas de software en gran escala. Las propuestas corrientes indican que la seguridad debería ser considerada desde las etapas tempranas del ciclo de vida de software [9].

Entendemos por ‘Ciclo de Vida del Desarrollo de Software’ (**SDLC** – por sus siglas en inglés: *Software Development Life Cycle*) al proceso de crear sistemas de infor-

mación, mediante una conjunto de modelos y metodologías específicos, que describe las etapas de análisis de requerimientos, diseño, implementación, prueba y mantenimiento. Del seguimiento de este proceso debería resultar un sistema de información de alta calidad, acorde a los requerimientos del cliente.

4.1 Metodologías y Técnicas para Seguridad

Han aparecido metodologías que modifican actividades tradicionales de los SDLCs, o insertan nuevas actividades, con el objetivo de reducir el número de debilidades y vulnerabilidades en software y su creciente dependencia ante amenazas [10]. Algunas metodologías han sido usadas satisfactoriamente en proyectos de desarrollo de productos o en proyectos pilotos académicos. Se pueden destacar: (1) *Microsoft Trustworthy Computing SDL* [11]; (2) *Oracle Software Security Assurance Process* [12]; (3) *Comprehensive, Lightweight Application Security Process -CLASP-* [13]; (4) *Seven Touchpoints for Software Security* [14]; y (5) *Team Software Process for Secure Software Development - TSP-Secure -* [15].

4.2 Microsoft Trustworthy Computing SDL

Para la experiencia que se describe en este trabajo se ha seleccionado la metodología *Microsoft Trustworthy Computing SDL* - MTC-SDL -, establecida formalmente por Microsoft para su proceso de desarrollo de software mejorado en seguridad, diseñado de manera que pueda ser empleada en un sentido amplio para diferentes desarrollos y diferentes tecnologías. El método SDL se aplica a lo largo de seis fases, trazando un mapa de tareas relevantes de seguridad y entregables en el SDLC existente: requerimientos, diseño, implementación y desarrollo, verificación, puesta en producción, soporte y mantenimiento. Se lo ha seleccionado por su completitud, por los materiales de guía y formación publicados, por su adaptabilidad a distinto tipo de proyectos y por su independencia tecnológica. Propone que el diseño sea conducido por el Modelado de Amenazas, que permite una descripción textual detallada y una descripción gráfica de las amenazas significativas para el sistema que se está modelando; este modelo captura los caminos a partir de los cuales las funciones del software y la arquitectura pueden ser apuntadas e identifican a los agentes de amenaza potenciales. En este trabajo nos hemos enfocado en dos técnicas:

1. ***STRIDE/DREAD Model-V1-2004***: ayuda a definir potenciales escenarios de amenazas desde la perspectiva del atacante, y se complementa con una metodología de cálculo de riesgo que encapsula respuestas para las potenciales preguntas sobre riesgo. Ayuda a ponderar y priorizar la importancia de sus contramedidas y mitigaciones [16].
2. ***Microsoft Threat Analysis and Modeling***: simplifica y cambia la perspectiva del atacante al defensor, considerando que el usuario se identifica estrechamente con amenazas, más que con ataques, y que el defensor puede entender mejor las amenazas al sistema que el atacante. Para su implementación se han generado herramientas, de cuales hemos seleccionado para realizar el análisis:
 - ***Microsoft Threat Analysis & Modeling v2.1 (2006) - TAM***: Herramienta creada para facilitar la creación y asimilación de modelos de amenazas. Permite a los no expertos en seguridad generar un modelo basado en información ya conocida, presente

en el desarrollo de todo sistema: requerimientos de negocios y arquitectura de la aplicación. A partir de la información de estos modelos, la herramienta identifica automáticamente las amenazas ha ser consideradas y permite generar un conjunto de artefactos de seguridad de valor: matrices de control de acceso, diagramas de flujo de datos basados en casos de uso, superficies de ataque, y reportes variados.

- **Threat Modeling Tool (2009) - TMT:** Herramienta que permite crear documentos de modelos de amenazas para aplicaciones, organizando la información relevante del modelo (puntos de entrada, activos, niveles de confianza, DFDs, amenazas y árboles de amenaza y vulnerabilidades) dentro de una vista simple e integrada. Permite exportar la documentación como XML, HTML y MHT.

Ambos son procesos iterativos, iniciando con un modelo de amenaza de alto nivel y con progresos de diseño detallado en las fases subsecuentes del ciclo de vida. Una de las versiones incorpora bibliotecas de ataque predefinidas que describen mitigaciones eficaces a cada tipo de ataque asociado con cada amenaza, autogenerando modelos de amenaza basados en un contexto definido de aplicación. El modelo entonces traza un mapa de aquellos modelos de amenaza a contramedidas relevantes.

5. Aplicación del MTC-SDL en el modelado de un caso

Para mostrar la aplicación del MTC-SDL y los aportes del mismo en las actividades de análisis y diseño, se ha seleccionado una aplicación web típica, a la que se la han realizado recortes a fin de poder exhibir resultados y mostrarlos de manera visible en las capturas de pantallas. El caso considerado en este análisis consiste en un Sistema Proveedor de Servicios de distribución de contenidos académicos (textos, apuntes de cátedra, artículos de revistas científicas, investigaciones, etc.), en formato digital con opciones de impresión. El sistema está basado en una arquitectura de tres capas, donde el cliente debe contar con un navegador de Internet ('browser') para poder interactuar con el mismo. También se requiere de espacio disponible en la Internet para montar dicho sitio y una aplicación de base de datos. Se debe contar con una base de datos de los clientes y de sus respectivas cuentas, y con una base de datos de publicaciones (libros y documentos) disponibles.

En el diagrama de componentes de esta aplicación se identifican dos Servidores que se comunican: un Servidor de Bases de Datos (que comprende la Base de Datos de Publicaciones y la Base de Datos de Clientes y Cuentas) y un Servidor Web.

En este análisis nos centraremos en la fase de requerimientos, identificando requerimientos funcionales principales y requerimientos de seguridad (funcionales y no funcionales), dejando a un lado otro tipo de requerimientos; y en la fase de diseño y modelado de amenazas de una porción del sistema completo de la librería on-line, específicamente la sección de 'Autenticación de un Usuario'.

En primer lugar, se presentan resultados obtenidos al emplear la herramienta TAM como ayuda para documentar los requerimientos de la aplicación y su arquitectura, para luego poder utilizarla en futuras etapas del desarrollo de software. Luego, se presentan alternativas usando la herramienta TMT.

5.1 Aplicando TAM

Para comprender el contexto de la aplicación, se deben definir los elementos individua-

les, así como la interacción entre ellos para definir el contexto de la aplicación. Esto hace posible la identificación de amenazas potenciales y la decisión de estrategias de seguridad efectivas. Al definir el contexto de la aplicación, es necesario definir los requerimientos y luego la arquitectura. Los primeros consisten en objetivos de negocios, roles de usuarios, datos y casos de uso, mientras que la arquitectura consiste en componentes, roles de servicios, dependencias externas y llamadas.

También se pueden asignar prioridades a cada una de las amenazas y especificar cada una de las respuestas, para así conformar la base de datos de amenazas.

A partir de los requerimientos y la descripción de la arquitectura, la herramienta trata de identificar de manera automática las amenazas, al tiempo que produce una serie de elementos como son: (1) matrices de acceso a datos; (2) casos de uso; (3) diagramas de flujos de datos, de llamada y de confianza; (4) superficie de ataque; y (5) informes.

A continuación se muestran capturas de pantalla obtenidas en distintos momentos de trabajo realizado sobre el caso de análisis con la herramienta TAM.

La Figura 1 muestra resultados del primer paso de descripción y modelado de la aplicación a desarrollar. En el panel de la izquierda puede observarse el árbol con el que esta aplicación organiza los elementos del modelo (roles de usuario - **a** -, almacenes de datos - **b** -, componentes - **c** -, dependencias externas y casos de uso - **d** -). A la derecha se encuentra el formato de la definición de un componente seleccionado ('Servidor Web'), donde el analista de requerimientos puede indicar datos básicos (nombre, descripción, roles asociados) así como también aspectos de 'alto nivel' relacionados a la implementación (tipo de servicio y tecnologías). Además, se pueden seleccionar (de una lista ofrecida como menú) las relevancias - **e** - del componente (atributo que define su comportamiento); que permite vincular los 'ataques' con los 'componentes' a través de librerías de ataques específicas y constituye el medio por el cual se comienza a incorporar la seguridad en la etapa temprana de relevamiento.

Luego de definir los componentes, se pasa a especificar los requerimientos mediante Casos de Uso, los cuales consisten de un conjunto de llamadas (calls). Cada llamada está compuesta por una entidad llamante (un rol de usuario o de servicio o un componente), y un componente que recibe la misma. Además, se permite especificar los datos enviados y recibidos, y el efecto sobre ellos. La definición precisa de las llamadas es muy importante, ya que la herramienta genera a partir de ellas una 'propuesta' de amenazas a evaluar, respecto a Confidencialidad, Integridad y Disponibilidad.

Para mostrar los aportes de la herramienta, como ejemplo analizamos el caso de uso 'Autenticación de un Usuario'. El proceso de autenticación lo inicia el usuario anónimo que llega al sitio y desea identificarse frente al mismo, para lo cual deberá proveer sus credenciales (en este caso, nombre de usuario y contraseña).

Definidas las llamadas, el caso de uso se visualiza como indica la Figura 2. Según se observa a la derecha, hay opciones para obtener una vista con el formato habitual de flujo de datos, o solicitar que se exhiban los aspectos de seguridad. Esta información se genera automáticamente a partir de los datos definidos anteriormente.

Una vez identificados los componentes claves de la aplicación, se debe examinar cuáles pueden ser las amenazas a las que está sujeto cada uno, para lo cual la metodología propone utilizar el método *STRIDE per element* para reconocer y categorizar las amenazas asociadas a la aplicación. La herramienta permite generar automáticamente las amenazas, que se clasifican en tres categorías: (a) amenazas de confidencialidad, (b) amenazas de integri-

dad, y (c) amenazas de disponibilidad. Las amenazas se pueden seleccionar de librerías predefinidas, o también crear librerías nuevas. Es posible entonces, definir automáticamente los ataques potenciales que se pueden usar para una amenaza dada. En la Figura 3 se presenta una vista de la definición de la amenaza ‘Revelación no autorizada de las credenciales’, identificada como una amenaza de Confidencialidad para el caso de uso seleccionado.

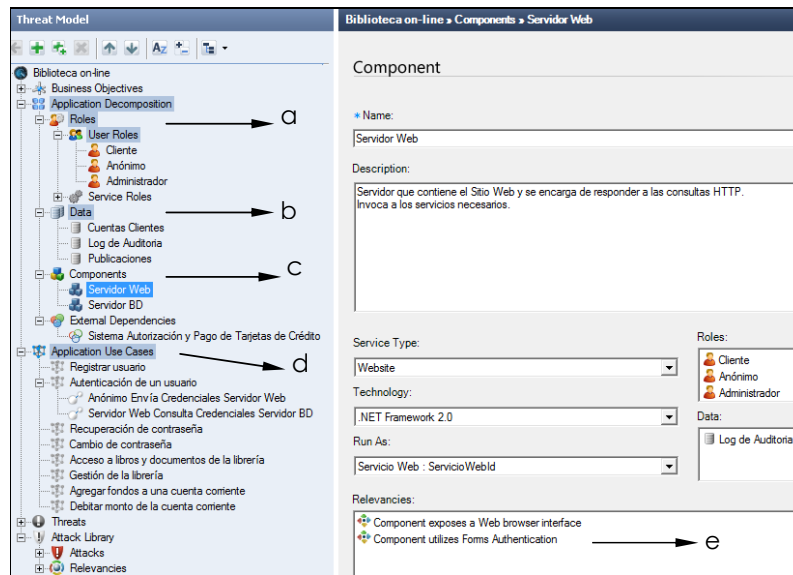


Fig. 1. Modelo-Componentes.

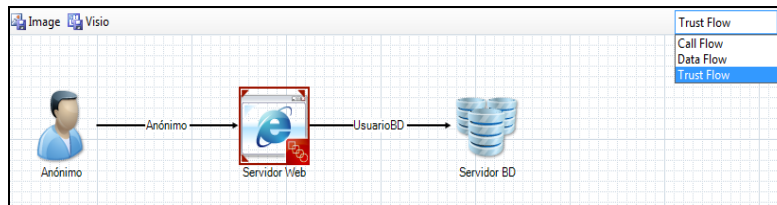


Fig. 2. Caso de Uso: ‘Autenticación de un Usuario’

Además, al identificar una amenaza es necesario ‘ponderarla’ para poder: (1) establecer el riesgo de cada una de ellas, (2) planificar prioridades y establecer contramedidas, así como fijar criterios de aceptación de riesgos. La herramienta permite utilizar el **Método DREAD** que pondera a través de cinco propiedades evaluadas por separado (daño potencial, reproducibilidad, facilidad de explosión, usuarios afectados, facilidad de descu-

brimiento), y luego obtener el valor de riesgo de la amenaza como un promedio. En la Figura 3, ya se ha presionado el botón del ítem ‘Calcular Riesgo’ y se puede observar la ventana que se abre para cuantificar los atributos y realizar el cálculo. En cuanto a las contramedidas, las que figuran en esta vista son las asociadas automáticamente por la aplicación y están relacionadas a los componentes por las relevancias, pero el usuario puede manualmente agregar otras contramedidas de la librería que crea necesarias.

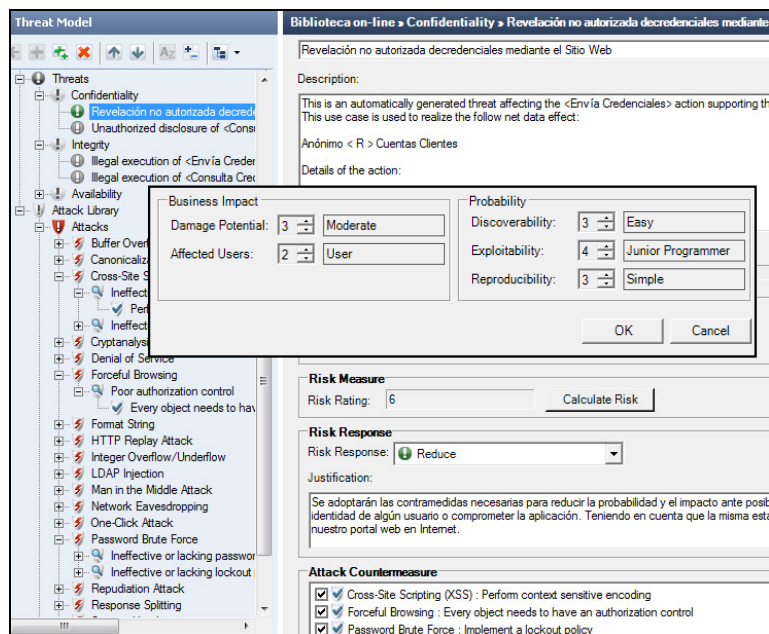


Fig. 3. Generación de la amenaza ‘Contraseña débil’.

Es de destacar que la identificación de amenazas y la asignación de los valores para calcular el riesgo ponderado están sujetos a la capacidad del analista. El aporte de la herramienta, al disponer de librerías e ir ofreciendo ítems para seleccionar, facilita la tarea cuando se trata de un analista no experto, pero no suple su decisión, por lo que resulta conveniente la toma de conciencia y la capacitación de los equipos para trabajar en el desarrollo de software seguro. Se debe observar que hay espacios para indicar descripción y justificación, que permite documentar las decisiones y así facilitar la trazabilidad.

Con la información consignada al generar una amenaza, es posible obtener también el correspondiente Árbol de Ataques, donde la raíz es la amenaza que se seleccionó y cada rama constituye un camino para su potencial explosión. La Figura 4 constituye una vista parcial del árbol de amenaza generado automáticamente para la amenaza ‘Revelación no autorizada de las credenciales’(se presenta más de 10 posibles ataques en el primer nivel de árbol y más de 25 contramedidas en el último).

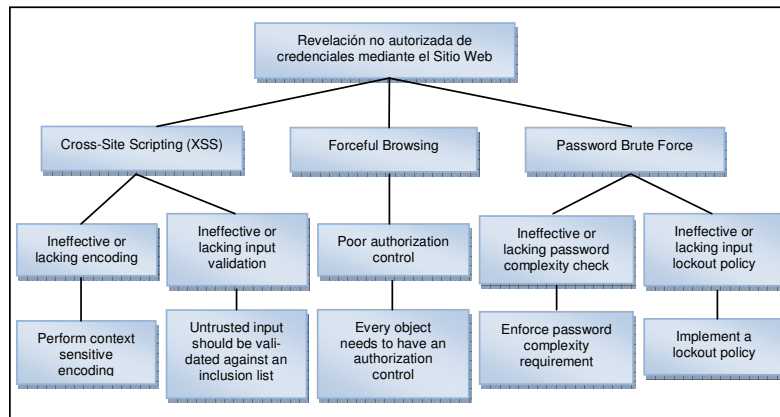


Fig. 4. Árbol de ataques de la amenaza ‘Revelación no autorizada de las credenciales’

Una vez identificadas las amenazas, el paso siguiente es proponer contramedidas efectivas. Como se observa en el cuadro izquierdo de la Figura 3, al seleccionar una amenaza se presenta al final de la pantalla una lista de posibles contramedidas para mitigar ese potencial ataque. En este caso, se ha seleccionado la contramedida de realizar controles de acceso sobre todos los objetos (“Every object needs to have an authorization control”) y se despliega automáticamente un cuadro que la describe, como en la Figura 5.

Countermeasure	
* Name:	Every object needs to have an authorization control
☒ Core countermeasure	
Description:	Authorization should be resource centric. Every privileged resource should be checked for authorization before granting access.
How to implement:	1. Implement proper Access control mechanism on resources. 2. Remove unnecessary files, temp files, back up files etc., 3. Ensure proper authorization checking in the application when user accesses admin pages, performs sensitive operations etc.
Code Snippet 1:	<pre> if HttpContext.Current.User.IsAuthenticated Then lblUserName.Text = HttpContext.Current.User.Identity.Name If HttpContext.Current.User.IsInRole("Manager") Then 'enable manager-level features </pre>

Fig. 5. Mitigación de riesgos/Contramedidas.

Es posible también **generar reportes**, destinados a diferentes miembros del equipo, que constituyen documentación de aporte para las etapas siguientes del ciclo. Estos pueden ser reportes destinados a: responsables de riesgos (amenazas y respuesta a las mismas), el equipo de diseño, el equipo de desarrollo (implementaciones sugeridas para las contramedidas), el equipo de testing (procedimientos o casos para testear los componentes, según los ataques a los que están expuestos y las amenazas definidas en

el modelo) y para el equipo de operaciones. Estos reportes pueden “customizarse” y proporcionan información resultante del modelo de amenazas, resumida y organizada, que orienta a diferentes actividades del proceso de desarrollo del software.

5.2 Aplicación de TMT

Una alternativa de modelado puede ser el uso de DFD, una representación gráfica que agiliza el proceso de modelado de requerimientos y al mismo tiempo es bien conocido por los profesionales de la ingeniería de software. La herramienta TMT da soporte para ello, con adaptaciones de manera que: los componentes se representan como Burbujas de Proceso, los roles de usuarios y dependencias externas como Entidades, y los almacenes de datos en forma directa. Además de los componentes típicos se han agregado los límites de confianza. A fin de determinar cuáles son las amenazas es preciso conocer: (a) cuáles son los puntos de entrada, (b) los niveles de confianza, y (c) los activos de mayor interés. En nuestro caso de análisis, el Diagrama del Modelo, enriquecido con los límites de confianza de datos y de equipos, tiene la vista que se muestra en la Figura 6.

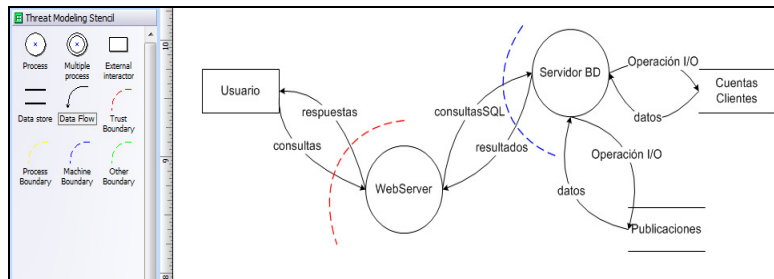


Fig. 6. DFD con límites de confianza.

Elemento	Spoofing Identity	Tampering with Data	Repudiation	Information Disclosure	Denial of Service	Elevation of Privilege
Entidad Externa	*		*			
Proceso	*	*	*	*	*	*
Almacén de datos		*	X	*	*	
Flujo de datos		*		*	*	

Fig. 7. Matriz de STRIDE.

A partir del diagrama, la herramienta produce un análisis automático por cada elemento de la lista de componentes definidos en el diseño, mediante el método STRIDE, tomando como base la siguiente matriz de ‘Identificación STRIDE de amenazas por tipo de elemento’ (Ver Figura 7).

Para el diagrama de la Figura 6, si se selecciona que se genere el Modelo de Análisis, resulta una vista como la que se muestra en la Figura 8, donde a la izquierda se tiene un

árbol completo de los elementos que caracterizan el sistema (entidades, procesos, almacenes y flujos), y para cada uno de ellos una expansión (en un nivel) de las categorías de amenazas provistas por STRIDE. Seleccionando un ítem (hoja del árbol) a la derecha se presenta automáticamente un conjunto de preguntas que el analista puede formularse como ayuda para identificar las amenazas. Además, dispone de una plantilla para poder definir una amenaza e indicar el impacto y la mitigación, de manera textual descriptiva. Si bien los detalles de la amenaza son menos precisos, la ayuda para su identificación es de nivel básico para un analista no experto y permite documentar las cuestiones de seguridad desde el inicio y de una forma simple que pueda ser comunicada al equipo.

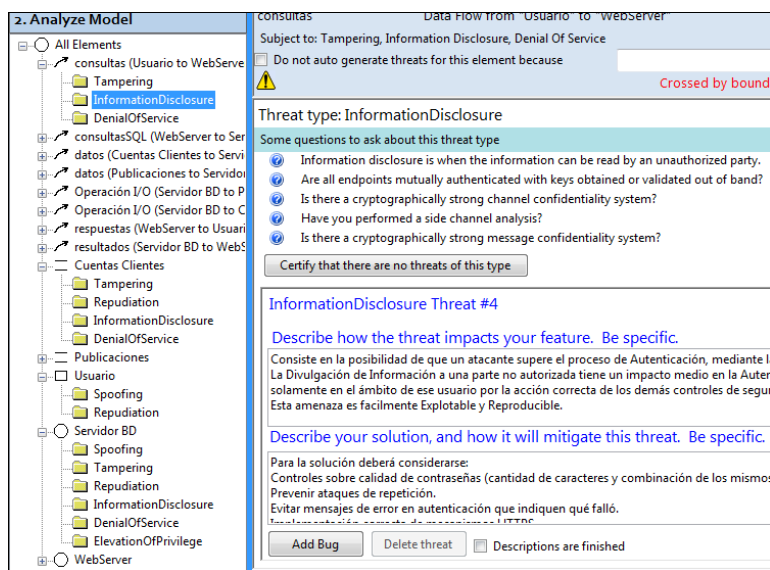


Fig. 8. Amenaza en TM.

6. Conclusiones

El éxito de implementar un modelado de amenazas pasa por realizar sucesivas iteraciones durante la evolución del proyecto y favorecer la reutilización de componentes que puedan servirnos para posteriores mejoras o para su uso en otros proyectos que tengan cierta similitud (en especial aplicaciones web). Las revisiones y mejoras del modelo se deben ir incorporando a medida que se produzcan cambios en el entorno de la aplicación o del sistema, su diseño, implementación, configuración y, por supuesto, cuando se modifiquen o aparezcan nuevos casos de uso. La incorporación de medidas de seguridad en el desarrollo de software puede no parecer rentable a corto plazo, pero las malas prácticas terminan demostrando que a medio-largo plazo esta percepción es errónea. Por ello, el uso de este enfoque viene a desempeñar un papel fundamental en todo ciclo de

vida de desarrollo que aspire a considerarse seguro.

Cuanto antes la comunidad incorpore metodologías y técnicas como las analizadas en este trabajo, mayor será su impacto. Si bien la disciplina de Ingeniería de Software Seguro está en amplia evolución y crece la comunidad de expertos, es necesario que los equipos actuales de diferentes tipos de proyectos comiencen a incorporarlo y la formación de recursos atienda estas cuestiones.

Referencias

1. Goertzel K., Winograd T., McKinley H., Oh L., Colon M., McGibbon T., Fedchak E., Vienneau R. "Software Security Assurance -State-of-the-Art Report (SOAR)", Information Assurance Technology Analysis Center (IATAC). 2007.
2. Microsoft Corporation. "Security Engineering Explained". Remond, WA; Microsoft Corporation Patterns and Practices Developer Center. USA. 2005.
<http://msdn2.microsoft.com/en-us/library/ms998382.aspx>
3. Hope P. and McGraw G. and Anton A. "Misuse and Abuse Cases: Getting Past the Positive". IEEE Security and Privacy. 2004. <http://www.cigital.com/papers/download/bsi2-misuse.pdf>
4. US CERT, Attack Patterns. US CERT.
<https://buildsecurityin.us-cert.gov/daisy/bsi/articles/knowledge/attack.html>
5. Hoglund G., McGraw G. "Exploiting Software: How to Break Code". Addison-Wesley. 2004.
6. Herrmann D. "A Practical Guide to Security Engineering and Information Assurance". Auerbach Publications. 2002.
7. Milano P. "Seguridad en el ciclo de vida del desarrollo de software". Argentina.
www.cybsec.com/upload/cybsec_Tendencias2007_Seguridad_SDLc.pdf
8. Shostack A. "Experiences Threat Modeling at Microsoft". 2008.
<http://blogs.msdn.com/sdl/archive/2008/10/08/>
9. Mouratidis H. "Integrating Security and Software Engineering". Idea Group Publishing. Inc. 2007.
10. N. Davis. "Secure Software Development Life Cycle Processes". US-CERT. 2006.
<https://buildsecurityin.us-cert.gov/daisy/bsi/326.html>
11. Lipner S., Howard M. "El ciclo de vida de desarrollo de seguridad de Trustworthy Computing". Unidad tecnológica y empresarial de seguridad-Microsoft Corporation, 2004 Annual Computer Security Applications Conference, copatrocinada por IEEE. USA. 2004.
12. R. Shores. "Oracle Software Security Assurance".
<http://www.oracle.com/security/software-security-assurance.html>
13. CLASP. <http://www.securesoftware.com/CLASP>
14. McGraw, G. "Software Security: Building Security In". Addison-Wesley. 2006.
15. Over, J. "TSP for Secure Systems Development". Presentation at CMU SEI. USA.
<http://www.sei.cmu.edu/tsp/tsp-secure-presentation/>
16. Meier J., Mackman A., Dunner M., Vasireddy S., Escamilla R., Murukan A. "Improving Web Application Security: Threats and Countermeasures". Microsoft Corporation. 2003.
<http://msdn.microsoft.com/en-us/library/aa302419.aspx>

Sobre el número de funciones bent obtenidas a partir de funciones de máximo peso*

Joan-Josep Climent¹, Francisco J. García² y Verónica Requena^{1, **}

¹ Departament de Ciència de la Computació i Intel·ligència Artificial.
Universitat d'Alacant.

Ap. correus 99, E-03080 Alacant, España.
jcliment@ua.es, vrequena@ua.es

² Departament de Fonaments de l'Anàlisi Econòmica.
Universitat d'Alacant.

Ap. correus 99, E-03080 Alacant, España.
francisco.garcia@ua.es

Resumen Dada una función bent $f(x)$ de n variables definimos sus funciones de máximo y mínimo peso como las funciones booleanas $f^+(x)$ y $f^-(x)$ cuyos soportes son los conjuntos $M^+ = \{a \in \mathbb{Z}_2^n \mid w(f \oplus l_a) = 2^{n-1} + 2^{\frac{n}{2}-1}\}$ y $M^- = \{a \in \mathbb{Z}_2^n \mid w(f \oplus l_a) = 2^{n-1} - 2^{\frac{n}{2}-1}\}$ respectivamente, donde $w(f \oplus l_a)$ denota el peso de Hamming de la función booleana $f(x) \oplus l_a(x)$ y $l_a(x)$ es la función lineal definida por $a \in \mathbb{Z}_2^n$. Combinando los 4 minterms de 2 variables con las funciones de máximo o mínimo peso de 4 funciones bent de n variables obtenemos una función bent de $n+2$ variables. Dado que no resulta fácil contar el número de funciones bent obtenidas con dicha construcción, proporcionamos una cota inferior de dicho número.

Palabras clave: Función booleana, función lineal, no linealidad, función bent, minterm, soporte, función de máximo peso, base de Gauss-Jordan

1 Introducción

Las funciones booleanas se utilizan en diferentes aplicaciones criptográficas tales como cifradores en bloque, cifradores en flujo, funciones hash [7,10,25], en teoría de códigos [5,22], entre otras. Con objeto de reducir al máximo las posibilidades de éxito del criptoanálisis, tanto diferencial como lineal, necesitamos funciones booleanas que posean buenas propiedades criptográficas. Para un número par de variables, las funciones bent son funciones booleanas que poseen la máxima no linealidad posible así como la menor autocorrelación posible [30,32],

* Este trabajo ha sido parcialmente subvencionado por el proyecto MTM2008-06674-C02-01 del Ministerio de Ciencia e Innovación del Gobierno de España.

** La investigación de esta autora ha sido financiada con una ayuda del Vicerrectorado de Investigación, Desarrollo e Innovación de la Universitat d'Alacant destinada a la formación de doctores.

lo cual las hace especialmente apropiadas para resistir los criptoanálisis anteriores. Por ejemplo, la implementación de una S-box necesita funciones booleanas no lineales para resistir ataques tales como el criptoanálisis diferencial [1,20,27].

El origen de las funciones bent se remonta a un artículo teórico de McFarland [24] sobre conjuntos de diferencias en grupos finitos no cíclicos. Posteriormente Dillon [17] sistematizó y extendió las ideas de McFarland proporcionando una gran cantidad de propiedades. El nombre *bent* con el que se conocen estas funciones se debe a Rothaus [29]; desde entonces estas funciones han sido objeto de estudio en muchas áreas como se desprende de la abundante literatura al respecto (véase por ejemplo [2,3,4,8,11,12,18,19,21,23,26,28] y las referencias en ellas incluidas).

Hay diferentes métodos para obtener funciones bent, muchos de ellos basados en la forma normal algebraica de una función booleana y en la transformada de Fourier (o Walsh) (véase, por ejemplo, [2,11,32]). Sin embargo, para un entero par n , nosotros [14,15] usamos la representación clásica de las funciones booleanas a través de minterms para construir funciones bent de $n + 2$ variables a partir de algunas funciones bent de n variables.

Tanto el uso de la forma normal algebraica como el de la tabla de verdad (equivalentemente, la expresión como suma de minterms), tienen sus ventajas y desventajas. Por ejemplo, la forma normal algebraica de una función booleana $f(\mathbf{x})$ de n variables proporciona directamente su grado y, si es mayor que $n/2$ se puede asegurar que $f(\mathbf{x})$ no es una función bent (véase [29]); sin embargo, no conocemos la cardinalidad de su soporte (es decir, el número de minterms). Por otro lado, si conocemos la tabla de verdad de $f(\mathbf{x})$, sabemos si su soporte tiene el número necesario de minterms o no para ser una función bent, pero no conocemos su grado.

Cabe destacar que no existe un método que permita generar todas las funciones bent, excepto para algunos casos particulares. Por ejemplo, para $n = 2$ solamente hay 8 funciones bent; para $n = 4$ sólo existen 896 funciones bent y para $n = 6$, Chang [13] probó que el número de funciones bent es 5 425 430 528. Sin embargo, su clasificación o determinación para $n \geq 8$ continúa siendo un problema abierto. Como comentamos al final de la sección 2, nosotros estamos interesados en la obtención del número de funciones bent distintas que podemos construir y no en el número de clases de funciones afínmente equivalentes.

El resto del artículo está organizado como sigue. En la sección 2 introducimos algunos conceptos básicos, la notación que utilizaremos a lo largo del artículo y algunos resultados de [14] necesarios para el seguimiento de este artículo, que puede ser considerado como la segunda parte de aquel. En la sección 3 introducimos los resultados que nos proporcionarán una cota inferior del número de funciones bent construidas de acuerdo con la construcción general introducida en [14]. Finalmente en la sección 4 presentamos algunas conclusiones.

2 Preliminares

Consideremos el cuerpo binario $\mathbb{Z}_2$ con la adición módulo 2 (denotada por $\oplus$) y la multiplicación módulo 2 (denotada por yuxtaposición). Para cualquier entero positivo n , sabemos que $\mathbb{Z}_2^n$ es un espacio vectorial sobre $\mathbb{Z}_2$ con la adición $\oplus$ dada por

$$\mathbf{a} \oplus \mathbf{b} = (a_1 \oplus b_1, a_2 \oplus b_2, \dots, a_n \oplus b_n)$$

para $\mathbf{a} = (a_1, a_2, \dots, a_n)$ y $\mathbf{b} = (b_1, b_2, \dots, b_n)$ en $\mathbb{Z}_2^n$. En $\mathbb{Z}_2^n$ consideramos también el producto interno

$$\langle \mathbf{a}, \mathbf{b} \rangle = a_1 b_1 \oplus a_2 b_2 \oplus \dots \oplus a_n b_n$$

de $\mathbf{a}$ y $\mathbf{b}$. Denotamos por $\mathbf{e}_i = (e_1^{(i)}, e_2^{(i)}, \dots, e_{n-1}^{(i)}, e_n^{(i)}) \in \mathbb{Z}_2^n$ la expansión binaria de n dígitos de i , es decir

$$e_1^{(i)} 2^{n-1} + e_2^{(i)} 2^{n-2} + \dots + e_{n-1}^{(i)} 2^1 + e_n^{(i)} 2^0 = i.$$

Con esta representación, podemos identificar el vector $\mathbf{e}_i$ con el entero i y, en consecuencia, podemos identificar $\mathbb{Z}_2^n$ con $\mathbb{Z}_{2^n}$.

Decimos que el conjunto $\{\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_k\} \subseteq \mathbb{Z}_2^n$ es una **base de Gauss-Jordan** de cardinalidad k si la matriz cuyas filas son $\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_k$ está en forma escalonada reducida (vease [9,16]).

Una función booleana de n variables es una aplicación $f : \mathbb{Z}_2^n \longrightarrow \mathbb{Z}_2$. El conjunto $\mathcal{B}_n$ de todas las funciones booleanas de n variables es un espacio vectorial sobre $\mathbb{Z}_2$ con la adición usual de funciones $\oplus$ dada por

$$(f \oplus g)(\mathbf{x}) = f(\mathbf{x}) \oplus g(\mathbf{x}), \quad \text{para } f, g \in \mathcal{B}_n.$$

Si $f \in \mathcal{B}_n$, llamamos **tabla de verdad** de f a la $(0, 1)$ -secuencia de longitud 2^n dada por

$$\boldsymbol{\xi}_f = (f(\mathbf{e}_0), f(\mathbf{e}_1), \dots, f(\mathbf{e}_{2^n-1})).$$

La tabla de verdad de una función booleana queda perfectamente determinada a partir de sus minterms. Un **minterm** en las variables $x_1, x_2, \dots, x_n$ es la función booleana dada por

$$m_{(u_1, u_2, \dots, u_n)}(x_1, x_2, \dots, x_n) = (1 \oplus u_1 \oplus x_1)(1 \oplus u_2 \oplus x_2) \cdots (1 \oplus u_n \oplus x_n).$$

Para $i = 0, 1, 2, \dots, 2^n - 1$, es evidente que $m_{\mathbf{e}_i}(\mathbf{x}) = 1$ si y sólo si $\mathbf{x} = \mathbf{e}_i$. Cuando no haya lugar a confusión, escribiremos $m_i(\mathbf{x})$ en lugar de $m_{\mathbf{e}_i}(\mathbf{x})$. Por tanto, la tabla de verdad

$$(m_i(\mathbf{e}_0), m_i(\mathbf{e}_1), \dots, m_i(\mathbf{e}_{2^n-1}))$$

de $m_i(\mathbf{x})$ tiene un 1 en la i -ésima posición y 0 en las restantes. En consecuencia,

$$\bigoplus_{i=0}^{2^n-1} m_i(\mathbf{x}) = 1. \quad (1)$$

También, como $m_i(\mathbf{x}) = m_j(\mathbf{x})$ si y sólo si $i = j$, podemos identificar el minterm $m_i(\mathbf{x})$ con el entero i (o con el vector $\mathbf{e}_i$ según convenga).

Ahora, para cualquier $f \in \mathcal{B}_n$ es fácil comprobar que

$$f(\mathbf{x}) = \bigoplus_{i=0}^{2^n-1} f(\mathbf{e}_i) m_i(\mathbf{x}) \quad (2)$$

y como la igualdad

$$\bigoplus_{i=0}^{2^n-1} a_i m_i(\mathbf{x}) = 0$$

implica $a_i = 0$ para $i = 0, 1, 2, \dots, 2^n - 1$, podemos afirmar que el conjunto $\{m_0, m_1, m_2, \dots, m_{2^n-1}\}$ es una base de $\mathcal{B}_n$.

Llamamos **soporte** de f al conjunto

$$M = \{\mathbf{a} \in \mathbb{Z}_2^n \mid f(\mathbf{a}) = 1\} \quad \text{o} \quad M = \{i \in \mathbb{Z}_{2^n} \mid f(\mathbf{e}_i) = 1\}$$

de acuerdo con la expresión (2) y la identificación de $\mathbb{Z}_2^n$ con $\mathbb{Z}_{2^n}$. Así, M es el conjunto de minterms de $f(\mathbf{x})$. De esta forma, podemos escribir la expresión (2) como

$$f(\mathbf{x}) = \bigoplus_{i \in M} m_i(\mathbf{x})$$

donde M es un subconjunto de $\mathbb{Z}_2^n$ o de $\mathbb{Z}_{2^n}$, según convenga.

El **peso de Hamming**, o simplemente **peso** de una $(0, 1)$ -secuencia α , que denotamos por $w(\alpha)$, es el número de 1 de α . El peso de Hamming de una función booleana $f(\mathbf{x})$, que denotamos por $w(f)$, es el peso de Hamming de su tabla de verdad ξ_f ; es decir, $w(f) = w(\xi_f)$ y, en consecuencia, $w(f)$ es el número de minterms en la expresión de $f(\mathbf{x})$ como suma de minterms, o equivalentemente, el cardinal del soporte de $f(\mathbf{x})$.

Decimos que $f \in \mathcal{B}_n$ es una **función afín** si

$$f(\mathbf{x}) = l_{\mathbf{a}}(\mathbf{x}) \oplus b$$

donde $\mathbf{a} \in \mathbb{Z}_2^n$, $b \in \mathbb{Z}_2$ y $l_{\mathbf{a}}(\mathbf{x}) = \langle \mathbf{a}, \mathbf{x} \rangle$. Si $b = 0$, decimos que f es una **función lineal**.

Definimos la **no linealidad** de una función $f \in \mathcal{B}_n$ como

$$\text{NL}(f) = \min\{d(f, \varphi) \mid \varphi \in \mathcal{A}_n\}$$

donde $\mathcal{A}_n \subseteq \mathcal{B}_n$ es el conjunto de todas las funciones afines y la distancia $d(f, g)$, para $f, g \in \mathcal{B}_n$, está definida como $d(f, g) = w(f \oplus g)$. La no linealidad de f está acotada superiormente (véase [32]) por

$$\text{NL}(f) \leq 2^{n-1} - 2^{\frac{n}{2}-1}.$$

Llamamos **funciones bent** a las funciones booleanas que alcanzan la máxima no linealidad (véase [32]). Por tanto, las funciones bent solamente existen para n par.

El resultado siguiente (véase [31,32]), que enunciamos para futuras referencias, proporciona una caracterización de las funciones bent.

Teorema 1. Sea $f(\mathbf{x})$ una función booleana de n variables. $f(\mathbf{x})$ es una función bent si y sólo si el número de 1 de la tabla de verdad de la función booleana $f(\mathbf{x}) \oplus l_{\mathbf{a}}(\mathbf{x})$ es $2^{n-1} \pm 2^{\frac{n}{2}-1}$ para todo $\mathbf{a} \in \mathbb{Z}_2^n$.

Como consecuencia del teorema anterior, si $f(\mathbf{x})$ es una función bent, entonces el número de 1 de su tabla de verdad es $2^{n-1} \pm 2^{\frac{n}{2}-1}$, o equivalentemente, $w(f) = 2^{n-1} \pm 2^{\frac{n}{2}-1}$. Además, $1 \oplus f(\mathbf{x})$ y $f(\mathbf{x}) \oplus l_{\mathbf{a}}(\mathbf{x})$ son funciones bent para todo $\mathbf{a} \in \mathbb{Z}_2^n$.

Supongamos que $f(\mathbf{x})$ es una función bent de n variables, entonces, de acuerdo con el teorema 1, podemos afirmar que $w(f \oplus l_{\mathbf{a}}) = 2^{n-1} \pm 2^{\frac{n}{2}-1}$ para todo $\mathbf{a} \in \mathbb{Z}_2^n$, lo cual motiva la definición siguiente.

Definición 1. Sea $f(\mathbf{x})$ una función bent de n variables. Llamamos **función de máximo peso** de $f(\mathbf{x})$ a la función

$$f^+(\mathbf{x}) = \bigoplus_{\mathbf{a} \in M^+} m_{\mathbf{a}}(\mathbf{x})$$

donde

$$M^+ = \{\mathbf{a} \in \mathbb{Z}_2^n \mid w(f \oplus l_{\mathbf{a}}) = 2^{n-1} + 2^{\frac{n}{2}-1}\}$$

es el **conjunto de máximo peso** de $f(\mathbf{x})$.

Análogamente, llamamos **función de mínimo peso** de $f(\mathbf{x})$ a la función

$$f^-(\mathbf{x}) = \bigoplus_{\mathbf{a} \in M^-} m_{\mathbf{a}}(\mathbf{x})$$

donde

$$M^- = \{\mathbf{a} \in \mathbb{Z}_2^n \mid w(f \oplus l_{\mathbf{a}}) = 2^{n-1} - 2^{\frac{n}{2}-1}\}$$

es el **conjunto de mínimo peso** de $f(\mathbf{x})$.

Notemos que al ser $f(\mathbf{x})$ una función bent, por el teorema 1 y la expresión (1) tenemos que

$$f^-(\mathbf{x}) = 1 \oplus f^+(\mathbf{x}), \quad (3)$$

es decir, la función de mínimo peso de $f(\mathbf{x})$ es la función complementaria de la función de máximo peso de $f(\mathbf{x})$.

Notemos también que, en general, $f^+ \neq f \neq f^-$,

$$w(f^+) \neq 2^{n-1} + 2^{\frac{n}{2}-1} \quad \text{y} \quad w(f^-) \neq 2^{n-1} - 2^{\frac{n}{2}-1}$$

como ponemos de manifiesto en la tabla 1 que muestra la relación entre f , f^+ y f^- cuando f recorre el conjunto formado por las ocho funciones bent de 2 variables.

Para una función bent $f(\mathbf{x})$ de n variables, en [14] introdujimos la función $f^+(\mathbf{x})$ como $f^*(\mathbf{x})$, la llamamos función dual de $f(\mathbf{x})$ y probamos que también era una función bent. Los resultados siguientes recogen, para futuras referencias, los resultados más importantes de dicho artículo.

f	f^+	f^-
m_0	$m_1 \oplus m_2 \oplus m_3$	m_0
m_1	m_2	$m_0 \oplus m_1 \oplus m_2$
m_2	m_1	$m_0 \oplus m_2 \oplus m_3$
m_3	m_3	$m_0 \oplus m_1 \oplus m_3$
$m_0 \oplus m_1 \oplus m_2$	$m_0 \oplus m_1 \oplus m_2$	m_3
$m_0 \oplus m_1 \oplus m_3$	$m_0 \oplus m_2 \oplus m_3$	m_1
$m_0 \oplus m_2 \oplus m_3$	$m_0 \oplus m_1 \oplus m_3$	m_2
$m_1 \oplus m_2 \oplus m_3$	m_0	$m_1 \oplus m_2 \oplus m_3$

Tabla 1. Funciones de máximo y mínimo peso de las ocho funciones bent de 2 variables

Teorema 2 (Teorema 2 de [14]). Si $f(\mathbf{x})$ es una función bent de n variables, entonces su función de máximo peso $f^+(\mathbf{x})$ también es una función bent de n variables.

Teorema 3.

1. (Corolario 1 de [14]). Sea $f(\mathbf{x})$ una función bent de n variables. Si $g(\mathbf{x}) = 1 \oplus f(\mathbf{x})$, entonces $g^+(\mathbf{x}) = 1 \oplus f^+(\mathbf{x})$.
2. (Corolario 2 de [14]). Si $f(\mathbf{x})$ es una función bent de n variables, entonces $f^{++}(\mathbf{x}) = f(\mathbf{x})$.
3. (Corolario 3 de [14]). Sean $f(\mathbf{x})$ y $g(\mathbf{x})$ funciones bent de n variables. Si $f(\mathbf{x}) \neq g(\mathbf{x})$, entonces $f^+(\mathbf{x}) \neq g^+(\mathbf{x})$.

Teorema 4 (Teorema 3 de [14]). Sean $f_0(\mathbf{x})$, $f_1(\mathbf{x})$, $f_2(\mathbf{x})$ y $f_3(\mathbf{x})$ cuatro funciones bent de n variables (no necesariamente distintas) tales que

$$f_0(\mathbf{x}) \oplus f_1(\mathbf{x}) \oplus f_2(\mathbf{x}) \oplus f_3(\mathbf{x}) = 1. \quad (4)$$

Si (i_0, i_1, i_2, i_3) es una permutación de $(0, 1, 2, 3)$ e $\mathbf{y} = (y_1, y_2)$ es un vector de dos variables, entonces

$$F(\mathbf{y}, \mathbf{x}) = m_{i_0}(\mathbf{y}) f_0^+(\mathbf{x}) \oplus m_{i_1}(\mathbf{y}) f_1^+(\mathbf{x}) \oplus m_{i_2}(\mathbf{y}) f_2^+(\mathbf{x}) \oplus m_{i_3}(\mathbf{y}) f_3^+(\mathbf{x})$$

es una función bent de $n + 2$ variables.

Notemos que como consecuencia de la expresión (3), los resultados anteriores son también válidos si cambiamos las funciones de peso máximo por las correspondientes funciones de peso mínimo.

Determinar todas las cuaternas $(f_0(\mathbf{x}), f_1(\mathbf{x}), f_2(\mathbf{x}), f_3(\mathbf{x}))$ de funciones bent de n variables que satisfacen la expresión (4) no es fácil. Sin embargo, si $f(\mathbf{x})$ y $g(\mathbf{x})$ son funciones bent de n variables, entonces las cuaternas

$$(f(\mathbf{x}), f(\mathbf{x}), f(\mathbf{x}), 1 \oplus f(\mathbf{x})) \quad (5)$$

$$(f(\mathbf{x}), f(\mathbf{x}), g(\mathbf{x}), 1 \oplus g(\mathbf{x})) \quad (6)$$

satisfacen, evidentemente, la expresión (4) con lo que tenemos el resultado siguiente.

Teorema 5.

1. (**Corolario 4 de [14]**). Si $f(\mathbf{x})$ es una función bent n variables e $i \in \{0, 1, 2, 3\}$, entonces

$$A_f(\mathbf{y}, \mathbf{x}) = f^+(\mathbf{x}) \oplus m_i(\mathbf{y})$$

es una función bent de $n + 2$ variables.

2. (**Corolario 5 de [14]**). Si $f(\mathbf{x})$ y $g(\mathbf{x})$ son funciones bent de n variables e (i_0, i_1, i_2, i_3) es una permutación de $(0, 1, 2, 3)$, entonces

$$B_{f,g}(\mathbf{y}, \mathbf{x}) = (m_{i_0}(\mathbf{y}) \oplus m_{i_1}(\mathbf{y})) f^+(\mathbf{x}) \oplus m_{i_2}(\mathbf{y}) g^+(\mathbf{x}) \oplus m_{i_3}(\mathbf{y}) (1 \oplus g^+(\mathbf{x}))$$

es una función bent de $n + 2$ variables.

Ahora, si en el teorema 5.2 exigimos que $g(\mathbf{x}) \neq f(\mathbf{x})$ y $g(\mathbf{x}) \neq 1 \oplus f(\mathbf{x})$, entonces todas las funciones bent construidas de acuerdo con el teorema 5 son distintas entre sí. Por tanto, si denotamos por ν_n el número de funciones bent de n variables, tenemos que

$$6\nu_n^2 - 8\nu_n \quad (7)$$

es el número de funciones bent de $n + 2$ variables que podemos construir de acuerdo con el teorema 5 (vease [14, página 15]).

Fuera de los casos particulares proporcionados por el teorema 5 resulta difícil contar cuántas cuaternas $(f_0(\mathbf{x}), f_1(\mathbf{x}), f_2(\mathbf{x}), f_3(\mathbf{x}))$ de funciones bent de n variables satisfacen la expresión (4), excluidas las correspondientes a las expresiones (5) y (6) y, por tanto, resulta difícil contar cuántas funciones bent de $n + 2$ variables distintas proporciona el teorema 4. En la sección siguiente establcere-mos una cota inferior de dicho número que, a partir de ahora, denotaremos por ω_n .

Antes de pasar a la sección siguiente, recordemos que dos funciones booleanas $f(\mathbf{x})$ y $g(\mathbf{x})$ son **afínmente equivalentes** si existe una matriz invertible A de tamaño $n \times n$ y un vector $\mathbf{a} \in \mathbb{Z}_2^n$ tales que $g(\mathbf{x}) = f(\mathbf{x}A \oplus \mathbf{a})$ y en tal caso (véase por ejemplo [6]) $g(\mathbf{x})$ es bent si y sólo si $f(\mathbf{x})$ lo es. En consecuencia, muchos autores trabajan en el problema de *encontrar el número de clases de funciones afínmente equivalentes y sus representantes*. Sin embargo, nosotros estamos interesados en el problema de *obtener el número de funciones bent distintas que existen o que podemos construir*, ya que el hecho de que dos funciones bent sean afínmente equivalentes no implica que sean distintas como ponemos de manifiesto en el ejemplo siguiente.

Ejemplo 1. Consideremos la función bent

$$f(\mathbf{x}) = m_0(\mathbf{x}) \oplus m_1(\mathbf{x}) \oplus m_2(\mathbf{x}) \oplus m_4(\mathbf{x}) \oplus m_8(\mathbf{x}) \oplus m_{15}(\mathbf{x})$$

de 4 variables, la matriz invertible

$$A = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}$$

y el vector $\mathbf{a} = (0, 0, 0, 1)$. Es fácil comprobar que las funciones $f(\mathbf{x}A \oplus \mathbf{a})$ y $f(\mathbf{x})$ tienen la misma tabla de verdad y, en consecuencia, que son iguales.

3 Resultados principales

Sean $f(\mathbf{x})$ y $g(\mathbf{x})$ funciones bent de n variables, supongamos que $\mathbf{a}, \mathbf{b} \in \mathbb{Z}_2^n$ y consideremos las cuaternas de funciones bent

$$(f(\mathbf{x}), f(\mathbf{x}) \oplus l_{\mathbf{a}}(\mathbf{x}), g(\mathbf{x}) \oplus l_{\mathbf{b}}(\mathbf{x}), 1 \oplus g(\mathbf{x}) \oplus l_{\mathbf{a} \oplus \mathbf{b}}(\mathbf{x})). \quad (8)$$

Como $l_{\mathbf{a} \oplus \mathbf{b}}(\mathbf{x}) = l_{\mathbf{a}}(\mathbf{x}) \oplus l_{\mathbf{b}}(\mathbf{x})$, es evidente que dicha cuaterna satisface la expresión (4). Antes de continuar, notemos que si tomamos $\mathbf{a} = \mathbf{b} = \mathbf{0}$, entonces las cuaternas (5) y (6) son un caso particular de la cuaterna anterior cuando $g(\mathbf{x}) = f(\mathbf{x})$ y $g(\mathbf{x}) \neq f(\mathbf{x})$ respectivamente. Por tanto, sólo necesitamos considerar los dos casos siguientes (justificaremos esta afirmación más adelante).

Teorema 6. Sean $\mathbf{a}, \mathbf{b} \in \mathbb{Z}_2^n \setminus \{\mathbf{0}\}$ con $\mathbf{a} \neq \mathbf{b}$. Si $f(\mathbf{x})$ es una función bent de n variables e (i_0, i_1, i_2, i_3) es una permutación de $(0, 1, 2, 3)$, entonces

$$\begin{aligned} C_{f, \mathbf{a}, \mathbf{b}}(\mathbf{y}, \mathbf{x}) &= m_{i_0}(\mathbf{y})f^+(\mathbf{x}) \oplus m_{i_1}(\mathbf{y})(f \oplus l_{\mathbf{a}})^+(\mathbf{x}) \\ &\quad \oplus m_{i_2}(\mathbf{y})(f \oplus l_{\mathbf{b}})^+(\mathbf{x}) \oplus m_{i_3}(\mathbf{y}) \left(1 \oplus (f \oplus l_{\mathbf{a} \oplus \mathbf{b}})^+(\mathbf{x})\right) \end{aligned}$$

es una función bent de $n + 2$ variables.

Teorema 7. Sean $\mathbf{a}, \mathbf{b} \in \mathbb{Z}_2^n \setminus \{\mathbf{0}\}$ con $\mathbf{a} \neq \mathbf{b}$. Si $f(\mathbf{x})$ y $g(\mathbf{x})$ son funciones bent de n variables tales que $f(\mathbf{x}) \neq g(\mathbf{x})$ e (i_0, i_1, i_2, i_3) es una permutación de $(0, 1, 2, 3)$, entonces

$$\begin{aligned} D_{f, g, \mathbf{a}, \mathbf{b}}(\mathbf{y}, \mathbf{x}) &= m_{i_0}(\mathbf{y})f^+(\mathbf{x}) \oplus m_{i_1}(\mathbf{y})(f \oplus l_{\mathbf{a}})^+(\mathbf{x}) \\ &\quad \oplus m_{i_2}(\mathbf{y})(g \oplus l_{\mathbf{b}})^+(\mathbf{x}) \oplus m_{i_3}(\mathbf{y}) \left(1 \oplus (g \oplus l_{\mathbf{a} \oplus \mathbf{b}})^+(\mathbf{x})\right) \end{aligned}$$

es una función bent de $n + 2$ variables.

Notemos que no todas las funciones bent proporcionadas por el teorema 6 son distintas entre sí como ponemos de manifiesto en el ejemplo siguiente.

Ejemplo 2. Supongamos que $n = 2$, consideremos los vectores $\mathbf{a} = \mathbf{1} = (0, 1)$ y $\mathbf{b} = \mathbf{2} = (1, 0)$, la función bent $f(\mathbf{x}) = m_1(\mathbf{x}) \oplus m_2(\mathbf{x}) \oplus m_3(\mathbf{x})$ y la permutación $(0, 2, 1, 3)$ de $(0, 1, 2, 3)$. Entonces

$$l_1(\mathbf{x}) = m_1(\mathbf{x}) \oplus m_3(\mathbf{x}), \quad l_2(\mathbf{x}) = m_2(\mathbf{x}) \oplus m_3(\mathbf{x}) \text{ y } l_{\mathbf{1} \oplus \mathbf{2}}(\mathbf{x}) = m_1(\mathbf{x}) \oplus m_2(\mathbf{x})$$

y, de acuerdo con el teorema 6 y la tabla 1, tenemos que

$$\begin{aligned} C_{f, \mathbf{a}, \mathbf{b}}(\mathbf{y}, \mathbf{x}) &= m_0(\mathbf{y})f^+(\mathbf{x}) \oplus m_2(\mathbf{y})(f \oplus l_1)^+(\mathbf{x}) \\ &\quad \oplus m_1(\mathbf{y})(f \oplus l_2)^+(\mathbf{x}) \oplus m_3(\mathbf{y}) \left(1 \oplus (f \oplus l_3)^+(\mathbf{x})\right) \end{aligned}$$

$$= m_0(\mathbf{y})m_0(\mathbf{x}) \oplus m_2(\mathbf{y})m_2(\mathbf{x}) \oplus m_1(\mathbf{y})m_1(\mathbf{x}) \oplus m_3(\mathbf{y})(1 \oplus m_3(\mathbf{x})).$$

Por otro lado, si consideramos los vectores $\mathbf{u} = \mathbf{1} = (0, 1)$ y $\mathbf{v} = \mathbf{3} = (1, 1)$, la función bent $g(\mathbf{x}) = m_2(\mathbf{x})$, y la permutación $(1, 0, 2, 3)$ de $(0, 1, 2, 3)$, entonces, procediendo como en el caso anterior tenemos que

$$\begin{aligned} C_{g,\mathbf{u},\mathbf{v}}(\mathbf{y}, \mathbf{x}) &= m_1(\mathbf{y})g^+(\mathbf{x}) \oplus m_0(\mathbf{y})(g \oplus l_1)^+(\mathbf{x}) \\ &\quad \oplus m_2(\mathbf{y})(g \oplus l_3)^+(\mathbf{x}) \oplus m_3(\mathbf{y})(1 \oplus (g \oplus l_2)^+(\mathbf{x})) \\ &= m_1(\mathbf{y})m_1(\mathbf{x}) \oplus m_0(\mathbf{y})m_0(\mathbf{x}) \oplus m_2(\mathbf{y})m_2(\mathbf{x}) \oplus m_3(\mathbf{y})(1 \oplus m_3(\mathbf{x})) \end{aligned}$$

que evidentemente coincide con $C_{f,\mathbf{a},\mathbf{b}}(\mathbf{y}, \mathbf{x})$.

Notemos que en el ejemplo anterior $\{\mathbf{1}, \mathbf{2}\}$ y $\{\mathbf{1}, \mathbf{3}\}$ son bases del mismo subespacio vectorial $\{\mathbf{0}, \mathbf{1}, \mathbf{2}, \mathbf{3}\}$ de $\mathbb{Z}_2^n$. Con el fin de evitar esta situación, consideraremos solamente vectores $\mathbf{a}, \mathbf{b} \in \mathbb{Z}_2^n$ tales que $\{\mathbf{a}, \mathbf{b}\}$ es una base de Gauss-Jordan de $\mathbb{Z}_2^n$ de cardinalidad 2. El resultado siguiente establece que las funciones bent construidas de acuerdo con el teorema 6 son distintas dos a dos si $\{\mathbf{a}, \mathbf{b}\}$ es una base de Gauss-Jordan de $\mathbb{Z}_2^n$ de cardinalidad 2.

Lema 1. Sean $f(\mathbf{x})$ y $p(\mathbf{x})$ dos funciones bent de n variables. Supongamos que $C_{f,\mathbf{a},\mathbf{b}}(\mathbf{y}, \mathbf{x})$ es la función bent de $n + 2$ variables construida de acuerdo con el teorema 6 utilizando $f(\mathbf{x})$, la base de Gauss-Jordan $\{\mathbf{a}, \mathbf{b}\}$ de $\mathbb{Z}_2^n$ de cardinalidad 2 y la permutación (i_0, i_1, i_2, i_3) de $(0, 1, 2, 3)$. Supongamos también que $C_{p,\mathbf{u},\mathbf{v}}(\mathbf{y}, \mathbf{x})$ es la función bent de $n + 2$ variables construida de acuerdo con el teorema 6 utilizando $p(\mathbf{x})$, la base de Gauss-Jordan $\{\mathbf{u}, \mathbf{v}\}$ de $\mathbb{Z}_2^n$ de cardinalidad 2 y la permutación (j_0, j_1, j_2, j_3) de $(0, 1, 2, 3)$. Si $f(\mathbf{x}) \neq p(\mathbf{x})$, entonces $C_{f,\mathbf{a},\mathbf{b}}(\mathbf{y}, \mathbf{x}) \neq C_{p,\mathbf{u},\mathbf{v}}(\mathbf{y}, \mathbf{x})$.

Demostración. Si ξ y η son las tablas de verdad de $f(\mathbf{x})$ y $p(\mathbf{x})$ respectivamente, entonces las tablas de verdad de $C_{f,\mathbf{a},\mathbf{b}}(\mathbf{y}, \mathbf{x})$ y $C_{p,\mathbf{u},\mathbf{v}}(\mathbf{y}, \mathbf{x})$ tienen cuatro bloques (no necesariamente en dicho orden y no necesariamente el mismo orden para las dos):

$$\begin{array}{llll} C_{f,\mathbf{a},\mathbf{b}}: & \xi^+ & (\xi \oplus \Lambda_{\mathbf{a}})^+ & (\xi \oplus \Lambda_{\mathbf{b}})^+ & \mathbf{1} \oplus (\xi \oplus \Lambda_{\mathbf{a} \oplus \mathbf{b}})^+ \\ C_{p,\mathbf{u},\mathbf{v}}: & \eta^+ & (\eta \oplus \Lambda_{\mathbf{u}})^+ & (\eta \oplus \Lambda_{\mathbf{v}})^+ & \mathbf{1} \oplus (\eta \oplus \Lambda_{\mathbf{u} \oplus \mathbf{v}})^+ \end{array}$$

donde $\Lambda_{\mathbf{a}}, \Lambda_{\mathbf{b}}, \Lambda_{\mathbf{a} \oplus \mathbf{b}}, \Lambda_{\mathbf{u}}, \Lambda_{\mathbf{v}}$ y $\Lambda_{\mathbf{u} \oplus \mathbf{v}}$ son las tablas de verdad de las funciones lineales $l_{\mathbf{a}}(\mathbf{x}), l_{\mathbf{b}}(\mathbf{x}), l_{\mathbf{a} \oplus \mathbf{b}}(\mathbf{x}), l_{\mathbf{u}}(\mathbf{x}), l_{\mathbf{v}}(\mathbf{x})$ y $l_{\mathbf{u} \oplus \mathbf{v}}(\mathbf{x})$ respectivamente, y $\mathbf{1}$ es la tabla de verdad de la función constante 1.

Si $C_{f,\mathbf{a},\mathbf{b}}(\mathbf{y}, \mathbf{x}) = C_{p,\mathbf{u},\mathbf{v}}(\mathbf{y}, \mathbf{x})$, entonces los cuatro bloques de la segunda fila son una permutación de los cuatro bloques de la primera fila. Ahora bien, si consideramos los $4!$ casos correspondientes a dichas permutaciones obtenemos, utilizando el teorema 3, que $f(\mathbf{x}) = p(\mathbf{x})$, o que $l_{\mathbf{c}}(\mathbf{x}) = 1$ para algún $\mathbf{c} \in \mathbb{Z}_2^n$ que depende de los vectores $\mathbf{a}, \mathbf{b}, \mathbf{u}$ y $\mathbf{v}$, o que

$$(\mathbf{u}, \mathbf{v}) \in \{(\mathbf{a}, \mathbf{a} \oplus \mathbf{b}), (\mathbf{b}, \mathbf{a} \oplus \mathbf{b}), (\mathbf{a} \oplus \mathbf{b}, \mathbf{a}), (\mathbf{a} \oplus \mathbf{b}, \mathbf{b})\}. \quad (9)$$

En cualquier caso tenemos una contradicción ya que $f(\mathbf{x}) \neq p(\mathbf{x})$ por hipótesis, $l_{\mathbf{c}}(\mathbf{x}) \neq 1$ para todo $\mathbf{c} \in \mathbb{Z}_2^n$, y si se satisface la relación (9), entonces $\{\mathbf{a}, \mathbf{b}\}$ y $\{\mathbf{u}, \mathbf{v}\}$ no pueden ser simultáneamente bases de Gauss-Jordan de cardinalidad 2. En consecuencia, $C_{f,\mathbf{a},\mathbf{b}}(\mathbf{y}, \mathbf{x}) \neq C_{p,\mathbf{u},\mathbf{v}}(\mathbf{y}, \mathbf{x})$. $\square$

Ahora, como consecuencia inmediata del lema anterior tenemos el resultado siguiente que establece el número de funciones bent de $n + 2$ variables distintas que podemos construir de acuerdo con el teorema 6.

Teorema 8. *Si ν_n es el número de funciones bent de n variables, entonces el número de funciones bent de $n + 2$ variables distintas que podemos construir utilizando el teorema 6 es*

$$\frac{4!}{3} \binom{2^n - 1}{2} \nu_n. \quad (10)$$

Demostración. De acuerdo con el lema 1, utilizando el teorema 6 podemos construir $4! \nu_n N(n, 2)$ funciones bent de $n + 2$ variables, donde $N(n, 2)$ es el número de bases de Gauss-Jordan de $\mathbb{Z}_2^n$ de cardinalidad 2. Ahora bien, como cada subespacio vectorial de dimensión 2 de $\mathbb{Z}_2^n$ tiene una única base de Gauss-Jordan de cardinalidad 2, tenemos que $N(n, 2)$ coincide con el número de subespacios vectoriales de $\mathbb{Z}_2^n$ de dimensión 2; por tanto (véase [33, página 46])

$$N(n, 2) = \frac{(2^n - 1)(2^n - 2)}{(2^2 - 1)(2^2 - 2)} = \frac{1}{3} \binom{2^n - 1}{2}$$

con lo que la expresión (10) representa el número de funciones bent de $n + 2$ variables distintas que proporciona el teorema 6. $\square$

Igual que ocurría con el teorema 6, tampoco todas las funciones bent construidas de acuerdo con el teorema 7 son distintas entre sí como ponemos de manifiesto en el ejemplo siguiente.

Ejemplo 3. Supongamos que $n = 2$, consideremos los vectores $\mathbf{a} = \mathbf{1} = (0, 1)$ y $\mathbf{b} = \mathbf{2} = (1, 0)$, las funciones bent $f(\mathbf{x}) = m_0(\mathbf{x})$ y $g(\mathbf{x}) = 1 \oplus m_3(\mathbf{x})$, y la permutación $(0, 1, 2, 3)$ de $(0, 1, 2, 3)$. De acuerdo con el teorema 7 y la tabla 1 (véase también el ejemplo 2 para las funciones $l_1(\mathbf{x})$, $l_2(\mathbf{x})$ y $l_3(\mathbf{x})$), tenemos que

$$\begin{aligned} D_{f,g,\mathbf{a},\mathbf{b}}(\mathbf{y}, \mathbf{x}) &= m_0(\mathbf{y})f^+(\mathbf{x}) \oplus m_2(\mathbf{y})(f \oplus l_1)^+(\mathbf{x}) \\ &\quad \oplus m_1(\mathbf{y})(g \oplus l_2)^+(\mathbf{x}) \oplus m_3(\mathbf{y})(1 \oplus (g \oplus l_3)^+(\mathbf{x})) \\ &= m_0(\mathbf{y})(m_1(\mathbf{x}) \oplus m_2(\mathbf{x}) \oplus m_3(\mathbf{x})) \\ &\quad \oplus (m_1(\mathbf{y}) \oplus m_2(\mathbf{y}))(m_0(\mathbf{x}) \oplus m_2(\mathbf{x}) \oplus m_3(\mathbf{x})) \oplus m_3(\mathbf{y})m_0(\mathbf{x}). \end{aligned}$$

Por otro lado, si consideramos los vectores $\mathbf{u} = \mathbf{1} = (0, 1)$ y $\mathbf{v} = \mathbf{3} = (1, 1)$, las funciones bent $p(\mathbf{x}) = m_0(\mathbf{x}) \oplus m_1(\mathbf{x}) \oplus m_3(\mathbf{x})$ y $q(\mathbf{x}) = m_3(\mathbf{x})$, y la permutación $(1, 0, 3, 2)$ de $(0, 1, 2, 3)$, entonces, procediendo como en el caso anterior, tenemos que

$$D_{p,q,\mathbf{u},\mathbf{v}}(\mathbf{y}, \mathbf{x}) = m_1(\mathbf{y})p^+(\mathbf{x}) \oplus m_0(\mathbf{y})(p \oplus l_1)^+(\mathbf{x})$$

$$\begin{aligned}
& \oplus m_3(\mathbf{y})(q \oplus l_3)^+(\mathbf{x}) \oplus m_2(\mathbf{y}) \left(1 \oplus (q \oplus l_2)^+(\mathbf{x}) \right) \\
& = m_0(\mathbf{y}) (m_1(\mathbf{x}) \oplus m_2(\mathbf{x}) \oplus m_3(\mathbf{x})) \\
& \oplus (m_1(\mathbf{y}) \oplus m_2(\mathbf{y})) (m_0(\mathbf{x}) \oplus m_2(\mathbf{x}) \oplus m_3(\mathbf{x})) \oplus m_3(\mathbf{y})m_0(\mathbf{x})
\end{aligned}$$

que evidentemente coincide con $D_{f,g,\mathbf{a},\mathbf{b}}(\mathbf{y}, \mathbf{x})$.

Notemos que en el ejemplo anterior se satisfacen las igualdades

$$g(\mathbf{x}) = f(\mathbf{x}) \oplus l_3(\mathbf{x}) \quad \text{y} \quad q(\mathbf{x}) = p(\mathbf{x}) \oplus l_2(\mathbf{x}) \oplus 1.$$

Por tanto, para evitar estas situaciones, en la construcción de las funciones $D_{f,g,\mathbf{a},\mathbf{b}}(\mathbf{y}, \mathbf{x})$ proporcionadas por el teorema 7 supondremos siempre que

$$g(\mathbf{x}) \neq f(\mathbf{x}) \oplus l_c(\mathbf{x}) \oplus c, \quad \text{para todo } (c, c) \in \mathbb{Z}_2^n \times \mathbb{Z}_2.$$

El resultado siguiente, cuya demostración es similar a la del lema 1 y que por tanto omitimos, establece que las funciones bent construidas de acuerdo con el teorema 7 son distintas dos a dos cuando las funciones $f(\mathbf{x})$ y $g(\mathbf{x})$ satisfacen la desigualdad anterior.

Lema 2. *Supongamos que $f(\mathbf{x})$, $g(\mathbf{x})$, $p(\mathbf{x})$ y $q(\mathbf{x})$ son funciones bent de n variables tales que*

$$g(\mathbf{x}) \neq f(\mathbf{x}) \oplus l_c(\mathbf{x}) \oplus c \text{ y } q(\mathbf{x}) \neq p(\mathbf{x}) \oplus l_c(\mathbf{x}) \oplus c, \text{ para todo } (c, c) \in \mathbb{Z}_2^n \times \mathbb{Z}_2.$$

Supongamos que $D_{f,g,\mathbf{a},\mathbf{b}}(\mathbf{y}, \mathbf{x})$ es la función bent construida de acuerdo con el teorema 7 utilizando las funciones bent $f(\mathbf{x})$ y $g(\mathbf{x})$, los vectores $\mathbf{a}$ y $\mathbf{b}$ de $\mathbb{Z}_2^n$ (con $\mathbf{a} \neq \mathbf{b}$) y la permutación (i_0, i_1, i_2, i_3) de $(0, 1, 2, 3)$. Supongamos también que $D_{p,q,\mathbf{u},\mathbf{v}}(\mathbf{y}, \mathbf{x})$ es la función bent construida de acuerdo con el teorema 7 utilizando las funciones bent $p(\mathbf{x})$ y $q(\mathbf{x})$, los vectores $\mathbf{u}$ y $\mathbf{v}$ de $\mathbb{Z}_2^n$ (con $\mathbf{u} \neq \mathbf{v}$) y la permutación (j_0, j_1, j_2, j_3) de $(0, 1, 2, 3)$. Si $f(\mathbf{x}) \neq p(\mathbf{x})$ y $f(\mathbf{x}) \neq p(\mathbf{x}) \oplus l_{\mathbf{u}}(\mathbf{x})$, entonces $D_{f,g,\mathbf{a},\mathbf{b}}(\mathbf{y}, \mathbf{x}) \neq D_{p,q,\mathbf{u},\mathbf{v}}(\mathbf{y}, \mathbf{x})$.

Ahora, como consecuencia inmediata del lema anterior tenemos el resultado siguiente que establece el número de funciones bent de $n + 2$ variables distintas que podemos construir de acuerdo con el teorema 7.

Teorema 9. *Si ν_n es el número de funciones bent de n variables, entonces el número de funciones bent de $n + 2$ variables distintas que podemos construir utilizando el teorema 7 es*

$$4! \binom{2^n - 1}{2} \frac{\nu_n}{2^n} \left(\frac{\nu_n}{2^{n+1}} - 1 \right). \quad (11)$$

Demostración. Como consecuencia del lema 2 podemos elegir $f(\mathbf{x})$ de $\nu_n/2^n$ formas distintas y, una vez fijada $f(\mathbf{x})$, podemos elegir $g(\mathbf{x})$ de $\nu_n/2^{n+1} - 1$ formas distintas. Por otro lado, puesto que podemos elegir los vectores $\mathbf{a}$ y $\mathbf{b}$ de $\binom{2^n - 1}{2}$ formas distintas y dado que hay $4!$ permutaciones distintas de $(0, 1, 2, 3)$, tenemos que la expresión (11) representa el número de funciones bent de $n + 2$ variables distintas que proporciona el teorema 7. $\square$

El resultado siguiente, cuya demostración es análoga a la de los lemas 1 y 2 y que por tanto omitimos, establece que ninguna de las funciones bent construidas de acuerdo con el teorema 6 coincide con ninguna de las funciones bent proporcionadas por el teorema 7 y viceversa.

Lema 3. *Supongamos que $f(\mathbf{x})$, $p(\mathbf{x})$ y $q(\mathbf{x})$ son funciones bent de n variables tales que*

$$q(\mathbf{x}) \neq p(\mathbf{x}) \oplus l_{\mathbf{c}}(\mathbf{x}) \oplus c \quad \text{para todo } (\mathbf{c}, c) \in \mathbb{Z}_2^n \times \mathbb{Z}_2.$$

Supongamos que $C_{f,\mathbf{a},\mathbf{b}}(\mathbf{y}, \mathbf{x})$ es la función bent construida de acuerdo con el teorema 6 utilizando la función bent $f(\mathbf{x})$, la base de Gauss-Jordan $\{\mathbf{a}, \mathbf{b}\}$ de $\mathbb{Z}_2^n$ de cardinalidad 2 y la permutación (i_0, i_1, i_2, i_3) de $(0, 1, 2, 3)$. Supongamos también que $D_{p,q,\mathbf{u},\mathbf{v}}(\mathbf{y}, \mathbf{x})$ es la función bent construida de acuerdo con el teorema 7 utilizando las funciones bent $p(\mathbf{x})$ y $q(\mathbf{x})$, los vectores $\mathbf{u}$ y $\mathbf{v}$ de $\mathbb{Z}_2^n$ (con $\mathbf{u} \neq \mathbf{v}$) y la permutación (j_0, j_1, j_2, j_3) de $(0, 1, 2, 3)$. Entonces $C_{f,\mathbf{a},\mathbf{b}}(\mathbf{y}, \mathbf{x}) \neq D_{p,q,\mathbf{u},\mathbf{v}}(\mathbf{y}, \mathbf{x})$.

Ahora, como consecuencia del lema 3 y de los teoremas 8 y 9 podemos enunciar el resultado siguiente que establece el número de funciones bent de $n + 2$ variables distintas que podemos construir de acuerdo con los teoremas 6 y 7.

Corolario 1. *Si ν_n es el número de funciones bent de n variables, entonces el número de funciones bent de $n + 2$ variables distintas que podemos construir utilizando los teoremas 6 y 7 es*

$$4! \binom{2^n - 1}{2} \nu_n \left(\frac{1}{3} + \frac{1}{2^n} \left(\frac{\nu_n}{2^{n+1}} - 1 \right) \right).$$

Demostración. Basta sumar las expresiones (10) y (11) para obtener el resultado ya que, por el lema 3, las funciones bent construidas de acuerdo con los teoremas 6 y 7 son distintas entre sí. $\square$

Finalmente, tal como habíamos anunciado al inicio de la sección, cualquier otra posible elección de los vectores $\mathbf{a}$ y $\mathbf{b}$, o de las funciones $f(\mathbf{x})$ y $g(\mathbf{x})$, puede ser reducida a uno de los casos considerados en los teoremas 5, 6 y 7 (junto con las condiciones adicionales del lema 2). Por ejemplo:

- Si $\mathbf{a} = \mathbf{0}$ y $\mathbf{b} \neq \mathbf{0}$, entonces la 4-tupla de la expresión (8) se convierte en

$$(f(\mathbf{x}), f(\mathbf{x}), g(\mathbf{x}) \oplus l_{\mathbf{b}}(\mathbf{x}), 1 \oplus g(\mathbf{x}) \oplus l_{\mathbf{b}}(\mathbf{x}))$$

que corresponde al teorema 5.1 si $f(\mathbf{x}) = g(\mathbf{x}) \oplus l_{\mathbf{b}}(\mathbf{x})$, o al teorema 5.2 si $f(\mathbf{x}) \neq g(\mathbf{x}) \oplus l_{\mathbf{b}}(\mathbf{x})$.

- Si $\mathbf{a} \neq \mathbf{0}$ y $\mathbf{b} = \mathbf{0}$, entonces la 4-tupla de la expresión (8) se convierte en

$$(f(\mathbf{x}), f(\mathbf{x}) \oplus l_{\mathbf{a}}(\mathbf{x}), g(\mathbf{x}), 1 \oplus g(\mathbf{x}) \oplus l_{\mathbf{a}}(\mathbf{x}))$$

que corresponde al teorema 5.1 si $g(\mathbf{x}) = f(\mathbf{x})$, $g(\mathbf{x}) = 1 \oplus f(\mathbf{x})$, $g(\mathbf{x}) = f(\mathbf{x}) \oplus l_{\mathbf{a}}(\mathbf{x})$ o $g(\mathbf{x}) = 1 \oplus f(\mathbf{x}) \oplus l_{\mathbf{a}}(\mathbf{x})$; o al teorema 7 (junto con las condiciones adicionales del lema 2) si $g(\mathbf{x}) \neq f(\mathbf{x})$, $g(\mathbf{x}) \neq 1 \oplus f(\mathbf{x})$, $g(\mathbf{x}) \neq f(\mathbf{x}) \oplus l_{\mathbf{a}}(\mathbf{x})$ o $g(\mathbf{x}) \neq 1 \oplus f(\mathbf{x}) \oplus l_{\mathbf{a}}(\mathbf{x})$.

Mediante un razonamiento análogo al utilizado en los lemas 1, 2 y 3, podemos probar que ninguna de las funciones bent construida de acuerdo con el teorema 5 puede ser obtenida por los teoremas 6 y 7 y viceversa. Por tanto, como consecuencia del corolario 1 tenemos el resultado siguiente.

Corolario 2. *Si ν_n es el número de funciones bent de n variables y ω_n es el número de 4-tuplas de funciones bent que satisfacen la ecuación (4), excluyendo las correspondientes a los casos contemplados en el teorema 5, entonces*

$$\omega_n \geq \binom{2^n - 1}{2} \nu_n \left(\frac{1}{3} + \frac{1}{2^n} \left(\frac{\nu_n}{2^{n+1}} - 1 \right) \right).$$

Finalmente, de la expresión (7) y el corolario anterior tenemos que el teorema 4 proporciona, al menos,

$$6\nu_n^2 + \left(-8 + 24 \binom{2^n - 1}{2} \left(\frac{1}{3} + \frac{1}{2^n} \left(\frac{\nu_n}{2^{n+1}} - 1 \right) \right) \right) \nu_n$$

funciones bent de $n+2$ variables distintas. Así, nuestra construcción proporciona 512 funciones bent de 4 variables, 9372608 funciones bent de 6 variables y 345031196824408177152 funciones bent de 8 variables. Como no conocemos el valor de ν_8 , no podemos determinar el número de funciones bent que podemos construir.

4 Conclusiones

En [14] presentamos un método general para obtener funciones bent de $n+2$ variables a partir de 4 funciones bent $f_0(\mathbf{x})$, $f_1(\mathbf{x})$, $f_2(\mathbf{x})$ y $f_3(\mathbf{x})$ de n variables que satisfacen la condición expresada por la ecuación (4). En este artículo hemos introducido una familia de cuaternas de funciones bent de n variables que satisfacen dicha condición y establecido el número de funciones bent de $n+2$ variables que podemos construir a partir de dichas cuaternas, obteniendo, en consecuencia, una cota inferior del número de funciones bent que se pueden obtener a partir del método general introducido en [14].

Referencias

1. C. M. ADAMS. Constructing symmetric ciphers using the CAST design procedure. *Designs, Codes and Cryptography*, **12**: 283–316 (1997).
2. C. M. ADAMS y S. E. TAVARES. Generating and counting binary bent sequences. *IEEE Transactions on Information Theory*, **35**(6): 1170–1173 (1990).
3. C. M. ADAMS y S. E. TAVARES. Generating bent sequences. *Discrete Applied Mathematics*, **39**: 155–159 (1992).
4. T. P. BERGER, A. CANTEAUT, P. CHARPIN y Y. LAIGLE-CHAPUY. On almost perfect nonlinear functions over $\mathbb{F}_2^n$. *IEEE Transactions on Information Theory*, **52**(9): 4160–4170 (2006).

5. Y. BORISSOV, A. BRAEKEN, S. NIKOVA y B. PRENEEL. On the covering radii of binary Reed-Muller codes in the set of resilient Boolean functions. *IEEE Transactions on Information Theory*, **51**(3): 1182–1189 (2005).
6. A. BRAEKEN, Y. BORISSOV, S. NIKOVA y B. PRENEEL. Classification of Boolean functions of 6 variables or less with respect to some cryptographic properties. En L. CAIRES, G. F. ITALIANO, L. MONTEIRO, C. PALAMIDESSI y M. YUNG (editores), *Automata, Languages and Programming*, volumen 3580 de *Lecture Notes in Computer Science*, páginas 324–334. Springer-Verlag, Berlin, 2005.
7. A. BRAEKEN, V. NIKOV, S. NIKOVA y B. PRENEEL. On Boolean functions with generalized cryptographic properties. En A. CANTEAUT y K. VISWANATHAN (editores), *Progress in Cryptology – INDOCRYPT 2004*, volumen 3348 de *Lecture Notes in Computer Science*, páginas 120–135. Springer-Verlag, Berlin, 2004.
8. A. CANTEAUT y P. CHARPIN. Decomposing bent functions. *IEEE Transactions on Information Theory*, **49**(8): 2004–2019 (2003).
9. A. CANTEAUT, M. DAUM, H. DOBBERTIN y G. LEANDER. Finding nonnormal bent functions. *Discrete Applied Mathematics*, **154**: 202–218 (2006).
10. C. CARLET y Y. TARANNIKOV. Covering sequences of Boolean functions and their cryptographic significance. *Designs, Codes and Cryptography*, **25**: 263–279 (2002).
11. C. CARLET. On bent and highly nonlinear balanced/resilient functions and their algebraic immunities. En M. FOSSORIER, H. IMAI, S. LIN y A. POLI (editores), *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes (AAECC-16)*, volumen 3857 de *Lecture Notes in Computer Science*, páginas 1–28. Springer-Verlag, Berlin, 2006.
12. C. CARLET y P. GUILLOT. An alternate characterization of the bentness of binary functions, with uniqueness. *Designs, Codes and Cryptography*, **14**: 133–140 (1998).
13. D. K. CHANG. Binary bent sequences of order 64. *Utilitas Mathematica*, **52**: 141–151 (1997).
14. J.-J. CLIMENT, F. J. GARCÍA y V. REQUENA. Construcción de funciones bent de $n+2$ variables a partir de las funciones duales de funciones bent de n variables. En A. CASTRO, J. LIPORACE y J. RAMÍO (editores), *Anales del IV Congreso Iberoamericano de Seguridad Informática (CIBSI 2007)*, páginas 3–17. Universidad Católica de Salta, 2007.
15. J.-J. CLIMENT, F. J. GARCÍA y V. REQUENA. On the construction of bent functions of $n+2$ variables from bent functions of n variables. *Advances in Mathematics of Communications*, **2**(4): 421–431 (2008).
16. M. DAUM, H. DOBBERTIN y G. LEANDER. An algorithm for checking normality of Boolean functions. En *Proceedings of the 2003 International Workshop on Coding and Cryptography (WCC 2003)*, páginas 133–142. marzo 2003.
17. J. F. DILLON. *Elementary Hadamard Difference Sets*. Tesis Doctoral, University of Maryland, 1974.
18. H. DOBBERTIN, G. LEANDER, A. CANTEAUT, C. CARLET, P. FELKE y P. GABORIT. Construction of bent functions via Niho power functions. *Journal of Combinatorial Theory (Series A)*, **113**(5): 779–798 (2004).
19. J. FULLER, E. DAWSON y W. MILLAN. Evolutionary generation of bent functions for cryptography. En *Proceedings of the 2003 Congress on Evolutionary Computation*, volumen 2, páginas 1655–1661. IEEE, 2003.
20. K. C. GUPTA y P. SARKAR. Improved construction of nonlinear resilient S-boxes. *IEEE Transactions on Information Theory*, **51**(1): 339–348 (2005).
21. P. V. KUMAR, R. A. SCHOLTZ y L. R. WELCH. Generalized bent functions and their properties. *Journal of Combinatorial Theory (Series A)*, **40**: 90–107 (1985).

22. K. KUROSAWA, T. IWATA y T. YOSHIWARA. New covering radius of Reed-Muller codes for t -resilient functions. *IEEE Transactions on Information Theory*, **50(3)**: 468–475 (2004).
23. A. LEMPEL y M. COHN. Maximal families of bent sequences. *IEEE Transactions on Information Theory*, **28(6)**: 865–868 (1982).
24. R. L. MCFARLAND. A family of difference sets in non-cyclic groups. *Journal of Combinatorial Theory (Series A)*, **15**: 1–10 (1973).
25. W. MEIER y O. STAFFELBACH. Nonlinearity criteria for cryptographic functions. En J. QUISQUATER y J. VANDEWALLE (editores), *Advances in Cryptology – EUROCRYPT’89*, volumen 434 de *Lecture Notes in Computer Science*, páginas 549–562. Springer-Verlag, Berlin, 1990.
26. K. NYBERG. Constructions of bent functions and difference sets. En I. B. DAMGÅRD (editor), *Advances in Cryptology – EUROCRYPT ’90*, volumen 473 de *Lecture Notes in Computer Science*, páginas 151–160. Springer-Verlag, Berlin, 1991.
27. K. NYBERG. Perfect nonlinear S-boxes. En D. W. DAVIES (editor), *Advances in Cryptology – EUROCRYPT ’91*, volumen 547 de *Lecture Notes in Computer Science*, páginas 378–386. Springer-Verlag, Berlin, 1991.
28. J. D. OLSEN, R. A. SCHOLTZ y L. R. WELCH. Bent-function sequences. *IEEE Transactions on Information Theory*, **28(6)**: 858–864 (1982).
29. O. S. ROTHUS. On “bent” functions. *Journal of Combinatorial Theory (Series A)*, **20**: 300–305 (1976).
30. P. SARKAR y S. MAITRA. Construction of nonlinear Boolean functions with important cryptographic properties. En B. PRENEEL (editor), *Advances in Cryptology – EUROCRYPT 2000*, volumen 1807 de *Lecture Notes in Computer Science*, páginas 485–506. Springer-Verlag, Berlin, 2000.
31. J. SEBERRY y X.-M. ZHANG. Constructions of bent functions from two known bent functions. *Australasian Journal of Combinatorics*, **9**: 21–35 (1994).
32. J. SEBERRY, X.-M. ZHANG y Y. ZHENG. Nonlinearity and propagation characteristics of balanced Boolean functions. *Information and Computation*, **119**: 1–13 (1995).
33. S. A. VANSTONE y P. C. VAN OORSCHOT. *An Introduction to Error Correcting Codes with Applications*. Kluwer Academic Publishers, Boston, MA, 2000.

Um IDS Cooperativo para Redes de Acesso de Banda Larga

Tiago Cruz¹, Thiago Leite¹, Patrício Batista¹, Rui Vilão¹,
Paulo Simões¹, Fernando Bastos², Edmundo Monteiro¹

¹ CISUC – DEI, Universidade de Coimbra,
Dep. Eng. Informática, Polo II da Universidade de Coimbra,
3030-290 Coimbra, Portugal

² PT Inovação SA,
Rua Eng. José Ferreira Pinto Basto, 3810-106 Aveiro, Portugal
tjcruz@dei.uc.pt, {thiago, pmbento, rpvilao}@student.dei.uc.pt,
psimoes@dei.uc.pt, fbastos@ptinovacao.pt, edmundo@dei.uc.pt

Resumo. O crescimento do número de clientes servidos por redes de acesso de banda larga (cabo, xDSL) acarreta um novo conjunto de preocupações ao nível da segurança, com potenciais consequências para os operadores de telecomunicações, para os seus clientes e para terceiros. O elevado número de clientes domésticos e tecnicamente impreparados que são actualmente servidos por conexões de elevado débito e natureza permanente constitui um cenário de risco para o qual o modelo tradicional de segurança dos operadores, centrado na sua infra-estrutura interna, é incapaz de dar resposta. Como alternativa a este cenário, propõe-se um modelo baseado no conceito de *segurança partilhada*, envolvendo a estreita cooperação entre os recursos de rede do operador e dos próprios clientes, procurando tirar partido do posicionamento que as *gateways* domésticas possuem no contexto das infra-estruturas de banda larga – como mediadores de fronteira entre as redes do operador e do cliente – para implementar um IDS/IPS (*Intrusion Detection System/Intrusion Protection System*) distribuído e escalável.

Palavras-chave: arquitecturas de segurança para serviços de banda larga, IDS distribuídos, redes domésticas.

1 Introdução

As redes de acesso de banda larga, na sua forma actual, representam um risco de segurança significativo para o *Internet Service Provider* (ISP) devido a quatro factores distintos: os elevados débitos disponíveis para cada um dos clientes; a massificação generalizada deste tipo de acesso, resultando num elevado número de clientes servidos por cada ISP; o carácter tendencialmente permanente das ligações (xDSL, Cabo, fibra óptica); e o facto de grande parte destes clientes não terem conhecimentos técnicos suficientes para garantir a segurança da sua rede doméstica. Ainda que parte dos riscos actuais existisse já anteriormente, o carácter intermitente das ligações *dial-up* clássicas e os reduzidos débitos disponíveis tornavam mais

simples aos ISP a tarefa de detectar e controlar situações de risco para as suas redes, para os seus clientes ou para terceiros.

A recente convergência dos serviços de voz, dados e televisão num mesmo canal de acesso (serviços *triple play*), aliada à profusão de serviços e aplicações com implicações nos modelos de uso tradicionais (aplicações *peer-to-peer* (P2P [1]), *instant messaging*, etc.) veio agravar ainda mais o problema da segurança em ambientes de banda larga, com repercussões a vários níveis. Em primeiro lugar, os clientes, por falta de sensibilidade tecnológica têm uma crescente dificuldade para lidar com o problema de segurança ao nível das suas próprias redes domésticas, ficando estas mais vulneráveis a ataques externos que poderão posteriormente comprometer a segurança da própria rede do operador. Adicionalmente, a sucessiva adição de serviços e aplicações torna cada vez mais difícil a detecção e resolução de ataques de segurança, principalmente quando esta tarefa é confiada a sistemas centralizados na rede do operador, de limitada escalabilidade. Por último, o impacto de quebras ou limitações de serviço de conectividade IP é cada vez maior, pois os clientes esperam que os serviços tradicionais agora suportados sobre o canal de banda larga mantenham parâmetros de qualidade e fiabilidade não inferiores às experiências anteriores com meios convencionais de acesso a telefone e televisão.

A atitude tradicional dos ISP tem sido considerar que a segurança da rede do cliente está fora da sua esfera de influência, devendo ser administrada autonomamente pelo cliente. Em geral, os operadores consideram que a sua esfera de influência termina no seu equipamento de fronteira (tais como, por exemplo, os DSLAM – *Digital Subscriber Line Access Multiplexers* - em infra-estruturas xDSL), sendo exclusiva responsabilidade do cliente a gestão dos seus equipamentos de fronteira – que designaremos por *home gateways* – e de tudo o que esteja para lá desses equipamentos. Esta atitude está aliás alinhada com a perspectiva cultural da maioria dos utilizadores, uma vez que estes não aceitariam de bom grado a interferência do operador – implícita ou explícita – na sua rede doméstica.

As redes *triple play* começam a alterar parcialmente esta perspectiva, passando a ser necessárias e aceites algumas intervenções do operador no interior da rede do cliente, nomeadamente para administrar remotamente *set-top boxes* (STB [2]) de TV e *gateways* de serviço telefónico. Mesmo para além desse contexto específico, a profusão de novos dispositivos nas redes domésticas, a oferta de novos serviços (IPTV, *video on demand*, telefone, televigilância, *online backup*...) e a mudança dos modelos de tráfego (com um peso cada vez maior de tráfego P2P) tornam necessário reavaliar estes pressupostos.

Actualmente, o modelo de segurança adoptado pelos ISPs é baseado num número relativamente reduzido de barreiras de tráfego (*firewalls*, analisadores de tráfego e outros mecanismos de defesa e prevenção) posicionadas em locais estratégicos da sua rede. Como consequência do aumento de clientes, do débito das ligações de cada cliente e de novas aplicações *bandwidth intensive* essas barreiras necessitam de lidar com volumes de tráfego cada vez mais elevados e com mecanismos de análise de tráfego cada vez mais complexos, com sérias implicações ao nível da escalabilidade e dos custos destas barreiras.

Em alternativa a esse modelo, propõe-se o aproveitamento do posicionamento específico que as *gateways* domésticas possuem no contexto das infra-estruturas de banda larga – como mecanismos que fazem a mediação entre as fronteiras da rede do

operador e dos clientes – para implementar um IDS/IPS (*Intrusion Detection System/Intrusion Protection System*) largamente distribuído. Caso o operador possa usar essas *gateways* domésticas como primeiro ponto de defesa da sua própria rede, poderá implementar mecanismos de segurança mais sofisticados, mais baratos, mais escaláveis e mais granulares. Em paralelo, os tradicionais utilizadores domésticos (com reduzidos conhecimentos técnicos) beneficiarão também com esta gestão partilhada das suas *gateways*, passando a ter as suas redes domésticas potencialmente mais protegidas.

O projecto S3P (Segurança em Ambientes *Triple-Play* [3]) – conduzido pelo Grupo de Comunicações e Telemática do Centro de Informática e Sistemas da Universidade de Coimbra e pela PT Inovação – tem precisamente por objectivo a definição, implementação e avaliação de uma arquitectura de segurança baseada nesse pressuposto de melhor articulação entre as redes domésticas e a infra-estrutura do operador, enquadrando a *gateway* doméstica como um dispositivo útil, simultaneamente, ao ISP e ao cliente. Nesta comunicação são apresentados os principais aspectos da arquitectura definida no S3P, de acordo com a seguinte organização: a Secção 2 discute em maior detalhe o contexto do projecto (ambientes *Triple Play* e tendências da indústria), a Secção 3 analisa as implicações ao nível ético da solução proposta. A Secção 4 debruça-se sobre os principais aspectos da solução proposta. As Secções 5 e 6 apresentam a arquitectura da plataforma S3P em detalhe (na perspectiva da *gateway* doméstica e do operador, respectivamente). A Secção 7 constitui um sumário do trabalho prévio realizado em projectos similares. A Secção 8 apresenta as conclusões e perspectivas de futuros desenvolvimentos do projecto.

2 Motivação

Tal como foi já mencionado na secção anterior, nos ambientes de banda larga a segurança da rede doméstica do cliente é tradicionalmente da sua inteira responsabilidade. Ainda que isto seja aceitável para clientes tecnicamente qualificados, coloca riscos consideráveis no caso da esmagadora maioria dos clientes domésticos, cuja capacidade para instalar e gerir mecanismos de segurança é nula ou bastante reduzida. Esta situação afecta em primeiro lugar o próprio cliente, mas acarreta também consequências para o operador. Por um lado tem um cliente potencialmente menos satisfeito (degradação de serviços prestados a esse cliente, incidentes graves de segurança na esfera do cliente). Por outro lado, caso a rede do cliente seja comprometida poderá ser usada para atacar outros clientes do ISP, o próprio operador ou terceiros. Nesse cenário, os elevados débitos oferecidos pelas actuais redes de acesso, a profusão de aplicações P2P e a convergência para cenários totalmente suportados sobre IP aumentam substancialmente os riscos, tanto para o cliente (numa perspectiva de intrusão na sua rede e acesso a dados confidenciais) como para o operador, que passa a estar muito mais vulnerável a ataques concertados de DoS e a situações de uso abusivo da sua infra-estrutura de rede.

Esta tendência tem-se reflectido na indústria, com a entrada na rede doméstica de equipamentos do operador (em especial *set-top boxes*) e com a presente tendência de normalização e convergência. Seja por iniciativas como a HGI [4] e o *Broadband*

Forum [5] seja por produtos como o *Windows Home Server* [6], passará a ser possível contar na rede de cada cliente com um conjunto homogêneo de serviços de segurança e administração remota, capazes de monitorizar a rede interna do cliente (se este assim o desejar), bem como a ligação entre a rede doméstica e a rede do operador. Diversas propostas técnicas produzidas pelo *Broadband Forum* e pela HGI apontam neste sentido, com a proposta de interfaces normalizados para configurações de serviços de segurança e operações de manutenção remota [4][7][8] em dispositivos localizados na rede do cliente.

Em conjunto, estas tendências abrem caminho para uma revisão da segurança das redes domésticas e das redes de acesso subjacentes, sendo que por um lado é necessário identificar e caracterizar as novas ameaças de segurança associadas a este cenário e por outro é necessário investigar e avaliar novas abordagens à forma de lidar com a rede doméstica do cliente.

3. Modelos de Segurança e Questões Éticas

Apesar de concebidas para proteger o ISP e/ou o cliente, praticamente todas as medidas de segurança têm efeitos secundários adversos, afectando um ou ambos os lados – a limitação de largura de banda e o *traffic shaping* por exemplo, têm sido alvo de polémica devido às suas implicações ao nível ético e legal. Estes efeitos colaterais tornam difícil alcançar o equilíbrio adequado entre a segurança e a privacidade em ambientes de banda larga.

No modelo tradicional, o ISP possui um raio de acção limitado (estando o seu perímetro de influência limitado ao nível da rede de acesso), recorrendo a barreiras de tráfego e sensores colocados em posições estratégicas para detectar e conter potenciais ataques. Uma análise simples a este modelo demonstra que este se baseia em dois pressupostos base:

- os clientes não abdicam da sua privacidade e não toleram interferências externas com os seus equipamentos e redes – sendo portanto responsáveis pela sua própria infra-estrutura
- os ISPs devem assegurar níveis adequados de segurança nas suas próprias infra-estruturas, ao mesmo tempo que restringem o seu raio de acção ao perímetro das redes de acesso

No cenário moderno das redes de banda larga, estes dois pressupostos são em larga medida incompatíveis entre si, uma vez que a maioria das ameaças é proveniente de equipamentos e redes de clientes que se encontram comprometidos. A título de exemplo, é frequente os ataques DDoS (*Distributed Denial of Service* [9]) serem provenientes de *botnets* [10] constituídas por aglomerados de computadores domésticos comprometidos por software malicioso, que atacam de modo concertado e coordenado – os próprios *spammers* utilizam técnicas similares para inundar os servidores de correio electrónico. No passado, estes ataques podiam ser mitigados com o recurso a medidas de contenção de tráfego activadas ao nível das barreiras

colocadas no interior da rede do ISP. Contudo, estas medidas são em parte ineficazes devido à mudança que se registou na natureza dos ataques: em vez de recorrer a um número relativamente limitado de hospedeiros comprometidos que geram fluxos de elevado débito no sentido de inundar um alvo, actualmente o número de equipamentos envolvidos é significativamente maior (envolvendo as redes de vários ISPs em vários países), caracterizando-se os fluxos de ataque pelo seu reduzido - quase insignificante - débito, em termos de tráfego gerado individualmente por cada máquina comprometida. No sentido de dar resposta a estes cenários, torna-se inevitável permitir aos ISPs que alarguem o seu raio de acção para além das restrições actualmente impostas.

A visão tradicional dos serviços de acesso de banda larga – nos quais o fornecedor apenas oferece serviços de conectividade, dando completa autonomia ao utilizador na forma de organizar a sua rede doméstica – perde algum sentido com *Triple Play* e outros serviços de valor acrescentado que dependem directamente de equipamento a colocar em casa do cliente. Na maior parte dos casos estes serviços exigem a instalação de equipamentos fornecidos especificamente pelo operador (*set-top boxes*, telefones IP, centrais de alarme...), quer por questões de compatibilidade técnica quer por estratégias comerciais (por exemplo capacidade de garantir níveis de *Digital Rights Management* apropriados para conteúdos multimédia). A crescente aceitação desses dispositivos abre caminho para uma redefinição da fronteira entre cliente e operador que permita uma melhor articulação entre a rede de acesso e a rede doméstica, sem com isso deixar de garantir a autonomia, liberdade de escolha e privacidade do cliente.

3.1 Um Novo Modelo de Segurança num Novo Cenário

Com os clientes a aceitarem voluntariamente a interferência, até certo grau, por parte dos ISPs nas suas próprias redes e equipamentos, um dos pressupostos base do modelo tradicional perde a sua relevância. Simultaneamente, as limitações do modelo de segurança tradicional tornam-no incapaz de beneficiar adequadamente do acréscimo de tolerância por parte dos subscritores dos serviços. Estas circunstâncias obrigam à adopção de uma abordagem diferente para as questões da segurança nos ambientes de banda larga.

No sentido de tirar partido da oportunidade da tolerância acrescida por parte dos utilizadores em relação à intervenção externa do ISP, propõe-se a introdução do conceito de *segurança assistida pelo operador*, baseado num modelo de gestão partilhada onde é permitido ao operador que alargue a sua influência até ao nível da *gateway* do cliente (denominada CPE – *Customer Premises Equipment*, no contexto deste projecto, ainda que o termo possua um significado mais amplo) podendo opcionalmente ir mais longe (ao nível da rede do cliente) se tal for autorizado:

- a possibilidade por parte do ISP de aceder ao CPE cria as condições necessárias para a implementação de um novo modelo de segurança baseado numa abordagem distribuída onde é possível posicionar mecanismos “inteligentes” de monitorização e segurança ao nível do CPE (na fronteira estratégica entre as redes do ISP e do cliente, em vez de centralizados ao

nível da infra-estrutura do ISP, com benefícios em termos de escalabilidade) sem interferir com a rede do cliente. Adicionalmente, nos casos em que sejam usadas exclusivamente para monitorizar o tráfego entre o ISP e o cliente, não reduzem a privacidade do cliente: o ISP poderia sempre proceder a uma monitorização semelhante dentro da sua rede, ainda que com custos substancialmente mais elevados.

- se o cliente assim o permitir, o CPE pode estender o raio de acção ao nível da sua rede interna, procurando localizar e identificar um amplo leque de vulnerabilidades, podendo mesmo recorrer a mecanismos nativos dos sistemas operativos para aceder a informações sobre as actualizações e parametrizações locais de segurança.

De certo modo, esta proposta constitui uma mudança de paradigma no que diz respeito aos modelos de segurança nas redes de banda larga, passando-se de uma arquitectura centralizada e pouco escalável para uma solução distribuída, baseada num modelo de *gestão partilhada* (Tabela 1).

Tabela 1: Abordagem Tradicional vs. Gestão Partilhada.

Abordagem Tradicional	Abordagem Proposta: Gestão partilhada
ISP restrito à sua própria rede.	Utilizadores definem o grau de influência do ISP. Alcance mínimo conseguido pelo ISP estendido até à fronteira do CPE. Rede interna do cliente poderá ser monitorizada pelo CPE, se o cliente assim desejar e autorizar.
Barreiras estáticas dentro da infra-estrutura do ISP, actuando como limitadores de tráfego/largura de banda.	Mecanismos a nível do CPE permitem um controlo de tráfego granular.
Equipamento do CPE incorpora capacidades limitadas e a sua eficiência em termos de segurança depende largamente dos (hipotéticos) conhecimentos técnicos do cliente.	Integração do CPE na infra-estrutura de segurança, gestão partilhada entre o ISP e o cliente.
Monitorização de tráfego é exequível na infra-estrutura do ISP, ainda que com limitações em termos de escalabilidade, granularidade e custo.	Monitorização de tráfego ao nível do CPE.
Mecanismos de detecção e contenção disponíveis em caso de um ataque distribuído são limitados, lentos e ineficazes.	Capacidade de detecção de ataques em execução antes de se espalharem, através de pedidos locais ao CPE. Capacidades embebidas no CPE permitem a activação de contra-medidas rápidas e de uma forma efectiva.

4. Abordagem Proposta

4.1 Modelo de Gestão

Como resposta à crescente dificuldade em escalar soluções de segurança clássicas do lado do operador, o Projecto S3P propõe um modelo de segurança distribuído, aproveitando as capacidades de processamento e gestão remota das *home gateways*, transferindo parte das funções de segurança para o equipamento do cliente. As *home gateways* são actualmente dispositivos com capacidade computacional bastante razoável, já estão disponíveis sem custos adicionais e estão posicionadas num ponto privilegiado da rede (mediação da rede de um único cliente com a rede de acesso do operador). Podem assim ser usadas para filtrar o tráfego de rede (em ambos os sentidos), enviar informação relevante para o operador e/ou para o cliente (alarmes de segurança, padrões de utilização, etc.) e implementar medidas de protecção (por exemplo bloqueio selectivo de tráfego em resposta a eventuais ataques).

O Projecto S3P propõe assim a criação de uma estrutura descentralizada em que as *gateways* domésticas actuam na linha da frente da protecção das redes internas, de modo a conter os efeitos de um eventual ataque a uma rede doméstica ou à rede do operador, ou mesmo evitá-lo de todo. A Figura 1 ilustra esta abordagem.

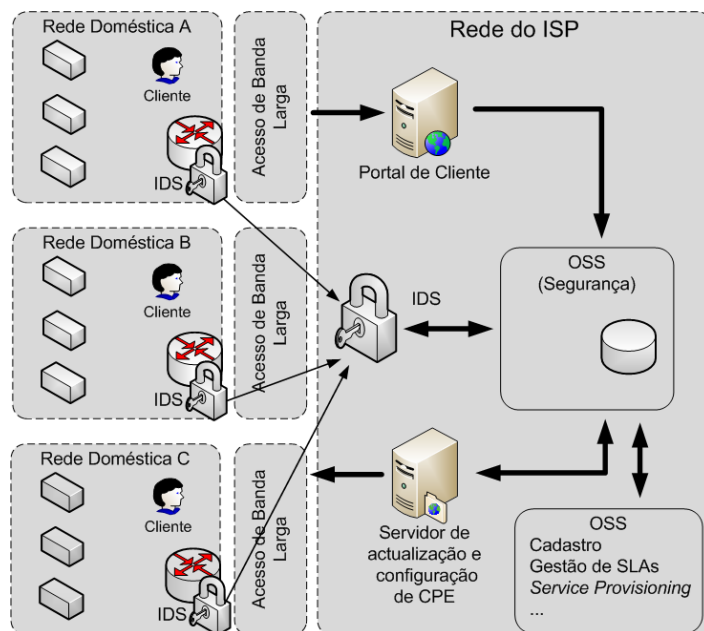


Figura 1: Modelo Genérico da Solução Proposta.

Na arquitectura S3P as *gateways* domésticas passam a funcionar de forma coordenada. Para além de realizarem funções de monitorização e prevenção de ataques através de meios próprios (com base em configurações previamente definidas pelo operador), podem também notificar o IDS do operador de determinados eventos e exercer acções de controlo de tráfego com base em instruções do IDS central.

Continuarão obviamente a existir clientes cujas *gateways* não colaborem com o IDS do ISP e que existe o risco de ter *gateways* comprometidas dentro da estrutura, pelo que o grau de confiança depositado pelo ISP em cada *gateway* doméstica nunca pode ser absoluto. A plataforma do operador terá pois de ter flexibilidade suficiente para suportar simultaneamente clientes com *gateways* cooperantes, clientes com *gateways* comprometidas e clientes sem *gateways* integradas. Apesar disso, do ponto de vista global, os potenciais ganhos de granularidade e de escala são consideráveis.

A arquitectura proposta não corresponde apenas ao “aproveitamento” da *gateway* doméstica pelo IDS do operador, tentando também aumentar efectivamente a articulação entre a rede do cliente e o ISP. Para o efeito as políticas de segurança adoptadas pelo IDS distribuído tomam em consideração o perfil do utilizador (cadastro, serviços contratados, etc.) e também permitem ao utilizador algum grau de personalização, por meio de um portal de cliente onde este pode por exemplo solicitar suporte explícito para algumas aplicações ou especificar perfis de uso mais detalhados que possam ser repercutidos no funcionamento do sistema (por exemplo controlo parental de conteúdos Web, bloqueio de acesso a servidores SMTP não previamente discriminados, etc.).

A arquitectura prevê o uso de entidades (agentes) presentes ao nível do ISP e CPE (Figura 2). Estas entidades actuarão sobre a análise do tráfego de dados transmitido, além de outros dados que possam ser obtidos a partir das bases de dados do ISP.

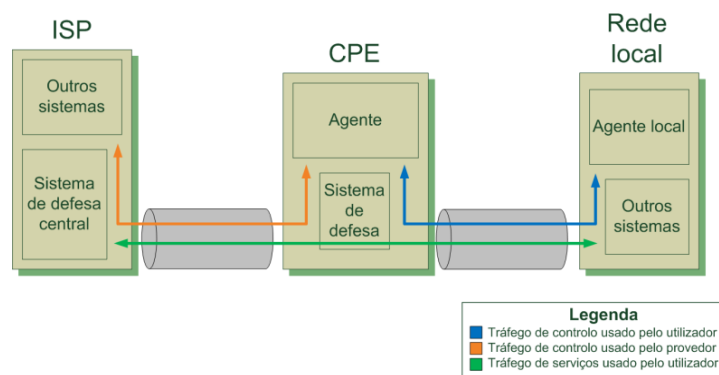


Figura 2: Âmbitos de influência no Projecto S3P

Conforme transparece da Figura 1, o modelo proposto não é completamente descentralizado, visto continuar a existir uma infra-estrutura de gestão do lado do operador que coordena os vários intervenientes neste processo, orquestrando a sua operação com base na correlação das informações recolhidas na rede do próprio operador e nos diversos CPEs.

A existência de um modelo de gestão de perfis de utilizadores permite que seja relativamente simples ao operador disseminar novas regras ou configurações para grupos alargados de utilizadores, em função dos seus perfis específicos e dos equipamentos instalados. A gestão de configurações (Figura 3) é realizada recorrendo ao servidor de auto-configuração (ACS, ou *Auto-configuration Server*), que realiza a distribuição de actualizações de software dos CPEs e a adição de novos serviços. A comunicação entre o ACS e os CPEs é assegurada por uma pilha protocolar especialmente desenvolvida para o efeito, que implementa a especificação TR-069 do *BroadBand Forum* (ou CWMP – *CPE WAN Management Protocol*), com o intuito de permitir transacções seguras entre o operador e os CPEs geridos.



Figura 3: Relação entre ACS e CPEs

4.2 Mecanismos de Segurança e Tratamento de Eventos na Arquitectura S3P

A ideia de proporcionar um papel mais activo às *gateways* domésticas não é propriamente novidade. Ferramentas como *firewalls* e mecanismos de gestão de *QoS* fazem hoje parte da maioria desses equipamentos, podendo ser configurados pelos utilizadores por meio de interfaces *Web*. No entanto, esta abordagem à questão é limitada por um conjunto de factores:

- A responsabilidade de configurar estas ferramentas é do utilizador, que frequentemente não tem a preparação técnica adequada para o efeito.
- A *gateway* funciona de forma isolada, não havendo por exemplo correlação de ataques com outros utilizadores do mesmo ISP ou com serviços específicos do operador ou da rede local. Um ataque realizado de modo

distribuído, como é característico das *botnets*, não é detectável através da análise de uma rede isolada.

- A capacidade destas ferramentas é relativamente limitada, podendo não ser suficiente para os ataques cada vez mais sofisticados a que hoje se assiste. Uma ferramenta, por mais flexível e poderosa que seja, não é efectiva se não for acompanhada por um conjunto de regras e mecanismos adaptáveis à altura das necessidades.

Estes três factores são minimizados no projecto S3P dos seguintes modos:

- Pela maior sofisticação dos mecanismos locais de segurança, incluindo filtros de pacotes, *proxies*, detecção e prevenção de intrusos, detecção de *portscans* [11] e vários outros mecanismos habitualmente reservados para redes de maior dimensão.
- Pela capacidade que o operador tem de correlacionar incidentes ocorridos em clientes distintos e activar mecanismos de resposta coordenados ao nível da sua rede e de todos os seus clientes.
- E pelo facto de as configurações de segurança de cada CPE serem geridas pelo operador, ainda que tendo em conta as preferências do cliente.

A detecção e o correcto tratamento de incidentes de segurança (que designaremos genericamente por *eventos*) é uma peça fundamental da arquitectura proposta. A geração e o tratamento de eventos ocorrem a dois níveis distintos:

- Ao nível do ISP. Os eventos recebidos dos diversos CPEs são correlacionados pelo motor de eventos do ISP, permitindo assim detectar, por exemplo ataques concertados a/de vários clientes do ISP. O ISP pode reagir a esses eventos tomando medidas preventivas na sua própria rede ou alterando as configurações dos CPEs (Figura 4).
- Ao nível local (CPE). Por razões de eficiência e escalabilidade, e de modo a permitir uma elevada granularidade no processo de monitorização, o CPE está dotado de um motor local de correlação de eventos (eventos esses captados pelas ferramentas de análise de tráfego, nomeadamente o sistema de detecção de intrusão, *portscans* e registos da actuação de outras ferramentas, tais como os bloqueios realizados pelo *firewall*, *proxy* e sistema de prevenção de intrusões). Todos os eventos são processados pelo motor local de correlação, podendo despoletar contra-medidas de natureza local, baseadas na aplicação de regras e procedimentos ao nível dos mecanismos de segurança do próprio CPE e/ou notificações de eventos para o ISP.

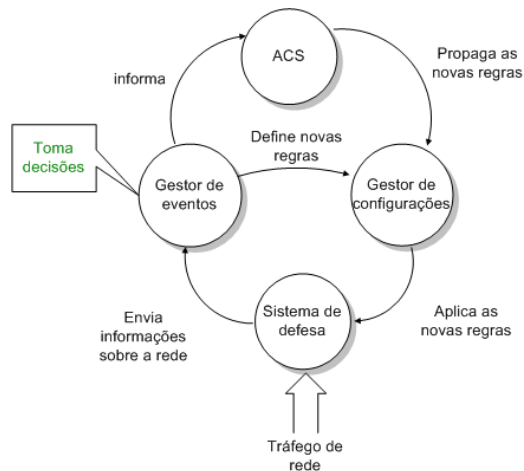


Figura 4: Modelo operacional do processo de tomada de decisões ao nível do ISP

Em termos gerais a plataforma proposta funciona como um sistema de detecção e prevenção de intrusões distribuído, abrangendo a rede do operador e os pontos de entrada/saída da rede dos clientes, sendo capaz de lidar com dois níveis de operação: CPE/microscópico e ISP/macrocópico. A título de exemplo, consideremos um conjunto de redes cliente que acabam de ser atacadas por um *trojan* [12] e estão a cooperar num ataque DDoS sincronizado. Cada CPE poderá, ao detectar actividades anómalas, alertar o operador através de um evento. Do lado do operador o mecanismo de correlação, ao detectar um padrão global, efectua a distribuição de novas regras de segurança (por exemplo, restringindo o uso de um determinado conjunto de portos TCP/IP ao nível do CPE) de modo a prevenir este ataque nos restantes clientes. É nesta óptica que se destaca a importância da gestão de eventos distribuída no projecto S3P.

5. Arquitectura S3P ao Nível do CPE

A Figura 5 apresenta a arquitectura da plataforma S3P na perspectiva do CPE. Os três módulos nucleares do CPE correspondem ao sistema de defesa (*Defense System*), ao motor de gestão de eventos (*Event Management*) e à gestão de configuração (*Configuration Management*). Entre os módulos de suporte inclui-se o gestor de falhas (*Failure Management*), destinado a gerir o funcionamento do próprio CPE (avarias de hardware, perdas de configurações, etc.) e o monitor da rede do cliente (*Customer Network Monitoring*), um módulo que poderá mais tarde ser usado para monitorizar a rede doméstica do utilizador.

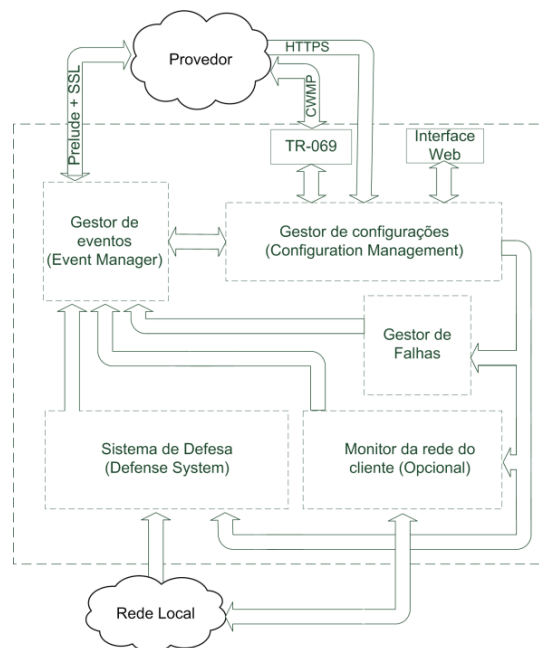


Figura 5: Arquitectura do S3P (vertente CPE)

O *Defense System* agrega os mecanismos activos de protecção do ambiente e análise passiva do tráfego que circula pelo CPE. A sua configuração (regras, listas de acesso, etc.) é controlada pelo ISP com recurso aos mecanismos anteriormente apresentados. O *Defense System* inclui os seguintes componentes: *firewall*, filtragem *Web* (*proxy* e controlo parental), sistema de detecção e prevenção de intrusão (IDS/IPS), detector de *portscans*, entre outros. Alguns destes componentes terão funções passivas (detectores de *portscans*, IDS), limitando-se a gerar eventos para tratamento pelo motor local. Outros terão também funções activas, podendo a sua configuração ser alterada dinamicamente, por decisão local ou do operador, em reacção a incidentes de segurança.

O gestor de eventos do CPE assenta na plataforma *Prelude IDS* [13], que inclui sofisticados mecanismos programáveis para processamento e correlação de eventos [14]. Nesta ferramenta os eventos são provenientes de sensores (agentes simples) que recolhem informação proveniente de diversas fontes e a partir daí constroem mensagens IDMEF (*Intrusion Detection Message Exchange Format*) [15] que posteriormente são enviadas para o módulo de gestão de eventos através de uma ligação segura e mantidas numa base de dados local.

O Gestor de Configurações assegura a gestão das configurações do CPE (serviços instalados, configurações activas). As actualizações podem ser despoletadas remotamente pelo ISP ou pelo próprio CPE (alteração de configuração decidida localmente para reacção a incidente de segurança).

6. Arquitectura S3P do Ponto de Vista do ISP

A Figura 6 apresenta a arquitectura da plataforma, do lado da infra-estrutura do operador. Os principais componentes correspondem ao gestor de perfis (*Profile Management*), ao gestor de eventos de segurança (*Security Event Management*) e ao gestor dos CPEs (*CPE Management*).

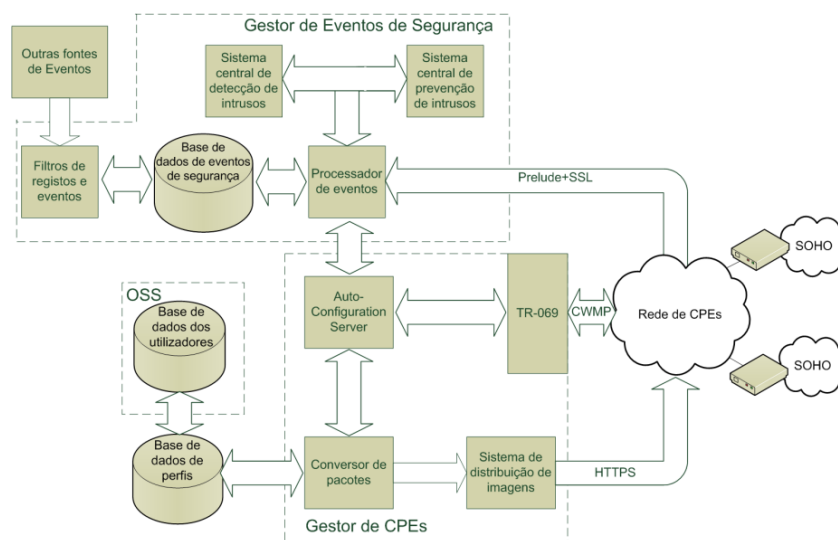


Figura 6: Arquitectura S3P (componentes do lado do ISP)

O gestor de eventos de segurança corresponde a um sistema correlacionador de eventos de segurança – alimentado pelos gestores de eventos de cada CPE e também por eventos detectados por sensores instalados na própria rede do operador – que permite correlacionar acontecimentos ocorridos em pontos distintos da rede e accionar respostas orquestradas a esses acontecimentos – por exemplo bloqueio de tráfego ao nível da rede do operador e/ou reconfigurações de *firewalls* dos CPEs.

A gestão dos CPE lida essencialmente com a configuração remota dos CPE (distribuição das aplicações e das configurações que devem aplicadas por cada CPE) e na monitorização do funcionamento do CPE (detectando e reagindo a falhas de funcionamento). Estas ferramentas servem para gestão de configuração, numa perspectiva genérica (inventário, *updates*, etc.) e são também o mecanismo de actuação remota que o ISP usa para alterar dinamicamente a forma de funcionamento dos CPE em resposta a incidentes de segurança.

A gestão de perfis assegura a manutenção de uma base de dados com perfis de equipamentos (fabricantes das CPE, modelos e versões instalados em cada cliente) e de utilizadores (serviços contratados ao ISP, preferências definidas no portal de cliente, etc.). Estes perfis são essenciais para definir que configurações devem ser enviadas para cada CPE.

7. Trabalho Relacionado

Apesar da relevância que assume actualmente a protecção de redes de acesso de banda larga, e da profusão de propostas de plataformas de IDS distribuídas, não existem, tanto quanto é do nosso conhecimento, outras propostas de integração de dispositivos domésticos em plataformas de segurança geridas de modo concertado por ISPs. Deste modo, os trabalhos mencionados nesta secção são distintos da plataforma S3P, ainda que com alguns pontos de contacto.

Koutepas [16] e Wan [17] [18] apresentaram um conjunto de propostas para lidar com ataques DDoS, baseadas em sistemas de detecção locais, posicionados estrategicamente ao longo da Internet, constituindo um sistema distribuído de detecção de ataques. Ao nível interno deste DIDS, os alertas são comunicados com recurso a mensagens IDMEF – uma vez detectado um ataque, são activados um conjunto de mecanismos de limitação de débito de tráfego para conter a ameaça.

Ioannidis et al [19], propõem uma solução para o problema dos ataques DDoS, utilizando *routers* quer para a detecção quer para a resposta aos ataques. Uma vez detectado um ataque DDoS, o sistema utiliza filtros específicos para bloquear o tráfego ao nível dos *routers*. Entre *routers* cooperantes, os eventos são comunicados com recurso ao protocolo *Pushback* desenvolvido para o efeito – deste modo, as fontes dos ataques vão sendo alvo de um cerco progressivo, ao mesmo tempo que são subjugadas ao uso controlado da largura de banda disponível.

8. Conclusão

Esta arquitectura distingue-se por aproveitar activamente a gateway doméstica – enquanto dispositivo de fronteira entre a rede de acesso e a rede doméstica – para criar uma plataforma distribuída de segurança, com ganhos para o operador (maior escalabilidade e granularidade, menores custos com sistemas centralizados na sua própria rede) e para o cliente. Ainda que esta abordagem pareça ir contra a visão tradicional do serviço Internet – com a fronteira na rede de acesso do ISP – ela ajusta-se bem aos recentes desenvolvimentos com a introdução de redes *Triple Play* e com a adopção pelos fabricantes de normas como o TR-069 para gestão remota de CPEs.

Está actualmente em curso a validação deste protótipo numa rede piloto da PT Inovação, de modo a que se possam recolher dados para um trabalho mais extenso de validação da escalabilidade da plataforma, por meio de medições experimentais e de métodos analíticos.

Agradecimentos. Este trabalho de investigação é parcialmente financiado pela Fundação para a Ciência e Tecnologia, através do projecto DOMUS (POSC/EIA/61076/2004) e pela PT Inovação, através do projecto S3P.

Referências

- Schollmeier, R., "A Definition of Peer-to-Peer Networking for the Classification of Peer-to-Peer Architectures and Applications", Proceedings of the First International Conference on Peer-to-Peer Computing, IEEE, 2002
- Allen, D., "The Ultimate Set Top Box (DVR) to Rule Them All?" (online), PC World, <http://www.networkworld.com/news/2009/072409-the-ultimate-set-top-box.html>, 2009
- Projecto S3P, <http://s3p.dei.uc.pt>
- HGI (2008), "Home Gateway Technical Requirements: Residential Profile v1.01", Home Gateway Initiative (online), http://www.homegatewayinitiative.org/publis/HGI_V1.01.pdf, 2008.
- Broadband-forum, <http://www.broadband-forum.org>
- WHS (2007), "Windows Home Server" (online), Microsoft Corporation, <http://www.microsoft.com/windows/products/winfamily/windowshomeserver>
- Broadband-forum (2006), "Functional Requirements for Broadband Residential Gateway Devices (TR-124) issue 1.0" (online), Broadband Forum, <http://www.broadband-forum.org/technical/download/TR-124.pdf>, Dezembro 2006.
- Broadband-forum (2007), "CPE WAN Management Protocol (TR-069) v1.1", <http://www.broadband-forum.org/technical/download/TR-069Amendment2.pdf>, Dezembro 2007
- Advanced Networking Management Lab, "Distributed Denial of Service Attacks(DDoS)" (online) Resources, Indiana University, <http://anml.iu.edu/ddos/types.html>
- Olmann G., "A Botnet by Any Other Name" (online), SecurityFocus, <http://www.securityfocus.com/columnists/501>, Maio 2009
- Jamieson S., "The Ethics and Legality of Port Scanning", SANS Institute GSEC Practical Assignment v1.2f" (online), http://www.sans.org/reading_room/whitepapers/legal/the_ethics_and_legality_of_port_scanning_71?show=71.php&cat=legal, Outubro 2001
- CERT (1999), "CERT Advisory CA-1999-02 Trojan Horses" (online), <http://www.cert.org/advisories/CA-1999-02.html>, Março 1999
- Vandoorselaere, Yoann (2008), "Prelude Universal SIM: State of the Art" (online), *Libre Software Meeting* 2008, http://www.prelude-ids.com/fileadmin/templates/pdf/RMLL_2008.pdf, Mont-de-Marsan, France, Julho 2008
- Chifflier, Pierre and Tricaud, Sébastien (2008), "Intrusion Detection Systems Correlation: a Weapon of Mass Investigation" (online), CanSecWest 2008, <http://www.prelude-ids.com/fileadmin/templates/pdf/correlation-womi-cansec2008.pdf>, Vancouver, Canada, Março 2008
- Debar, H. et al (2007), "The Intrusion Detection Message Exchange Format (IDMEF)", RFC 4765, Março 2007
- Koutepas, G., Stamatelopoulos, F., Maglaris, B., "Distributed Management Architecture for Cooperative Detection and Reaction to DDoS Attacks", Journal of Network and Systems Management, Vol. 12, No. 1, Março 2004
- Wan, K. and Chang R., "Engineering of a global defense infrastructure for DDoS attacks", Proc. of IEEE International Conference on Networking, Agosto 2002.
- Wan, K., "An infrastructure to defend against distributed denial-of-service attack", M.Sc. Thesis, Hong Kong Polytechnic University, Junho 2001.
- Ioannidis, J. and Bellovin S., "Implementing pushback: Router-based defense against DDoS attacks", Network and Distributed System Security Symposium 2002, San Diego, California, Fevereiro 2002.

An Extended Reference Monitor for Security and Safety

Eduardo B. Fernandez¹, Michael VanHilst¹, David laRed Martinez²
and Sergio Mujica³

¹ Department of Computer Science and Engineering, Florida Atlantic University
777 Glades Road

Boca Raton, Florida 33431 USA

² Departamento de Informática, Facultad de Ciencias Exactas y Naturales y Agrimensura -
Universidad Nacional del Nordeste

Avenida Libertad 5460

3400 Corrientes, Argentina

³ Facultad de Ingeniería y Ciencias, Universidad Adolfo Ibanez

Diagonal Las Torres 2640

Peñalolén - Santiago, Chile

{ed, mike}@cse.fau.edu, lrmdavid@exa.unne.edu.ar, sergio.mujica@udp.cl

Abstract. We present here a unified way to handle the combined enforcement of safety and security through the use of patterns, in particular through the concept of Reference Monitor. A Reference Monitor is an abstract mechanism that applies authorization rules to decide access. We first define an Extended Reference Monitor with reified decisions and then apply the idea to a dual monitor that can handle security and safety.

Keywords: Reference monitor, security, authorization, safety.

1 Introduction

Although it is a non-functional aspect, security is a fundamental objective of any system where there are assets that may be targets of attacks because of their economic value or potential for disruptive impact [1]. Security is the protection against intentional attacks and until recently most studies of security have focused on the protection of information assets based on their monetary and business value. Recent terrorist threats have brought interest to the protection of physical assets as well. Correspondingly, the studies of threats to infrastructure have focused on safety. Safety is the freedom from unacceptable risks, including threats to human lives, the environment, or to costly facilities. Safety is often expressed with assertions on how to avoid unsafe conditions, e.g. an elevator must not open its doors when moving. These assertions are usually related to specific states of a control system. Safety is often confused with reliability or with availability. Some perceive safety as providing “zero defect” or “failure free” software, others consider it a matter of providing uninterrupted service. While reliability and availability are clearly important, they are

not sufficient to provide safety. It is also not enough to apply the same security measures used in IT systems to critical systems. While superficially there are many commonalities, delving deeper one finds many differences [2]. These differences imply the need for a specialized use of mechanisms and development methodologies. However, there are commonalities and it is important to identify them to produce systems that can enforce more than one type of requirement and to save development work. This paper is an attempt in that direction.

Hazards are states or conditions of the system, that when combined with some state of the environment, lead to mishaps. A mishap is the occurrence of an unwanted event that leads to human or environment harm. Leveson categorizes hazards into three groups [3]: The system is not available. The system generates an incorrect output. The system misses a hard deadline. Security attacks can deny service, illegally modify system state, or introduce artificial delays; that is, security attacks can directly affect safety through all three hazard categories. On the other hand, an unsafe condition can cause security problems. All of these commonalities point to the need to deal with security and safety in a unified way. We need to produce systems where security and safety are developed together. Our long-term objective is the development of a complete methodology for critical systems but we are initially concentrating on some specific aspects. Safety requires reliability and methods to improve reliability are also of interest to us. Because of the changing nature of these systems, we also consider flexibility as an added requirement. Security and safety concerns have been mostly developed separately, although a few works try to combine them, e.g. [4,5,6].

The conceptual control models of safety-critical systems, both static and dynamic, are defined at the application level. It is here also where the security policies of the institution should be applied. At this level the semantics of the application are well understood. We can also apply here institutional policies and policies that correspond to regulations. Other non-functional aspects are also specified here, e.g., the required degree of reliability and safety assertions. The lower levels enforce the restrictions defined at the higher levels. Each level has its own security mechanism and should participate in enforcing the constraints [7]. For example, a DBMS enforces the authorizations in the application by restricting access to database items; this restriction is propagated down to control access to the files where this data resides [8].

An important development in software is the concept of *pattern*. A pattern is a solution to a recurrent problem in a given context [9]. A pattern embodies the knowledge and experience of software developers and can be reused in new applications. Well-thought patterns also implicitly apply good design principles. The typical solution provided by a pattern comes in the form of a class diagram complemented with some sequence diagrams and possibly activity or state diagrams. This level of detail and precision allows designers to use them as guidelines and to users to understand the use of the mechanism they represent.

We present here a unified way to handle the combined enforcement of safety and security through the use of patterns, in particular through the concept of Reference Monitor. A Reference Monitor is an abstract mechanism that applies authorization rules to decide access [10]. Concrete realizations of the reference Monitor may exist at each architectural level. In order to define this Reference Monitor precisely we provide UML class and sequence diagrams to describe its structure and behavior.

Section 2 describes some background, while Section 3 presents our extensions to the idea of Reference Monitor. We apply those extensions in Section 4 to define a Reference Monitor that can also handle safety. We then discuss related work and we finish with some conclusions and ideas for future work.

2 Threats and Hazards

In previous work we introduced an approach for finding security requirements based on misuse activities (actions) [11]. This method starts from the activity diagram of a use case (or a sequence of use cases). Each activity is analyzed to see how it could be subverted to produce a misuse of information. This analysis results in a set of threats. We then consider which policies can stop or mitigate these threats. We recently extended that approach to consider the type of misuse (confidentiality, integrity,...) that can happen in each activity, the role of the attacker, and the context for the threat. This extended analysis results in a finer and more systematic way to find threats and we can identify now more threats [12]. We can apply this approach to control systems to identify security threats and safety hazards as well as the effect of errors.

Mishaps happen because a dangerous situation was not considered in the requirements, or because there was a failure in some unit of the system, or because there was an intentional action taken against vulnerable elements of the system. Using our systematic approach to find threats we can also find hazards. Hazards must be prevented in the design of the software for the system. In other words, safety is dependent on the correctness, reliability, and security of the system. Conversely, safety features or restrictions (e.g. emergency access) or reliability failures can lead to security breaches. However, high reliability and security do not guarantee safety.

Once misuse actions have been uncovered, appropriate security policies must be selected to stop and/or mitigate them. This activity could be aided by some comprehensive security policy list. However, this selection should result in a minimum set of mechanisms instead of mechanisms piled up because they might be useful. We discuss these issues in [11] and [12]. Through examples, we have found that with a good catalog of policies we can stop or mitigate most threats or hazards.

Policies lead to specific countermeasures. For example, authorization requirements can be enforced through the abstract concept of Reference Monitor. A Reference Monitor intercepts requests for resources (protection objects) and decides if the request is valid according to the authorization rules.

3 The Extended Reference Monitor

We proposed a pattern for a Reference Monitor in [13]. We present here a pattern for an Extended Reference Monitor that allows a more precise definition of the decision made by this mechanism. Specifically, we modify the Reference Monitor pattern to include Decision as a separate class, i.e. the decision is reified and has attributes. This addition allows manipulation of decisions as well as defining

conditions or other additional restrictions. We present only some sections of the complete pattern, following the template used in [9].

Intent: Enforce authorizations when a subject requests a protection object and provide the subject with a decision which can have conditions.

Context: A multiprocessing environment where subjects request protection objects to perform their functions and access resources maybe bsd on conditions.

Problem: Authorization can be defined according to different models, e.g. access matrix or Role-Based Access Control. If we don't enforce the defined authorizations it is the same as not having them, subjects can perform all type of illegal actions. Any user could read any file for example. How can we control the subjects' actions?

Decisions can be sometimes more complex than a Boolean response, e.g. when a user wants to access some database items, she may be authorized to access a subset of the data she requests. In this case, the decision is not Boolean (yes/no) but only some items are released. Also, many times there is a need to store decisions for possible reuse.

Intent: Enforce authorizations when a subject requests a protection object and provide the subject with a decision which can have conditions.

Context: A multiprocessing environment where subjects request protection objects to perform their functions and access resources maybe bsd on conditions.

Problem: Authorization can be defined according to different models, e.g. access matrix or Role-Based Access Control. If we don't enforce the defined authorizations it is the same as not having them, subjects can perform all type of illegal actions. Any user could read any file for example. How can we control the subjects' actions?

Decisions can be sometimes more complex than a Boolean response, e.g. when a user wants to access some database items, she may be authorized to access a subset of the data she requests. In this case, the decision is not Boolean (yes/no) but only some items are released. Also, many times there is a need to store decisions for possible reuse.

Solution: Define an abstract process that intercepts all requests for resources, checks them for compliance with authorizations, makes decisions based on these authorization rules and stores those decisions including their attributes.

Fig. 1 shows a class diagram showing a Reference Monitor with a reified decision. In this figure Set_of_Authorization_Rules denotes a collection of Authorization rules organized in a convenient way and corresponding to some security model. The Actual Subject (Client) can be a process which makes a Request for a Protection Object with the intention of applying to it some access type. The Reference Monitor makes a Decision according to the authorization rules. Examples of Concrete Reference Monitors are file permission systems, memory managers, or other resource controllers. Fig. 2 shows a sequence diagram showing how checking is performed.

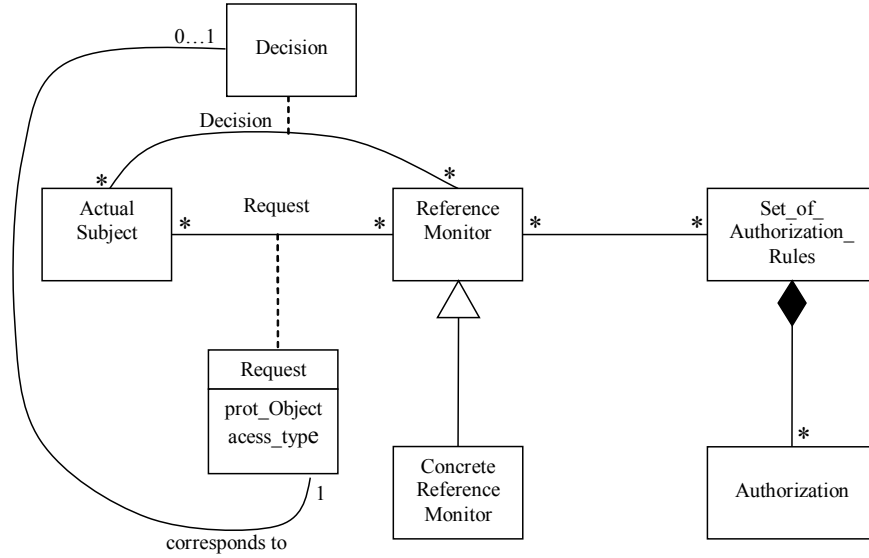


Fig. 1. Class diagram for the modified Reference Monitor.

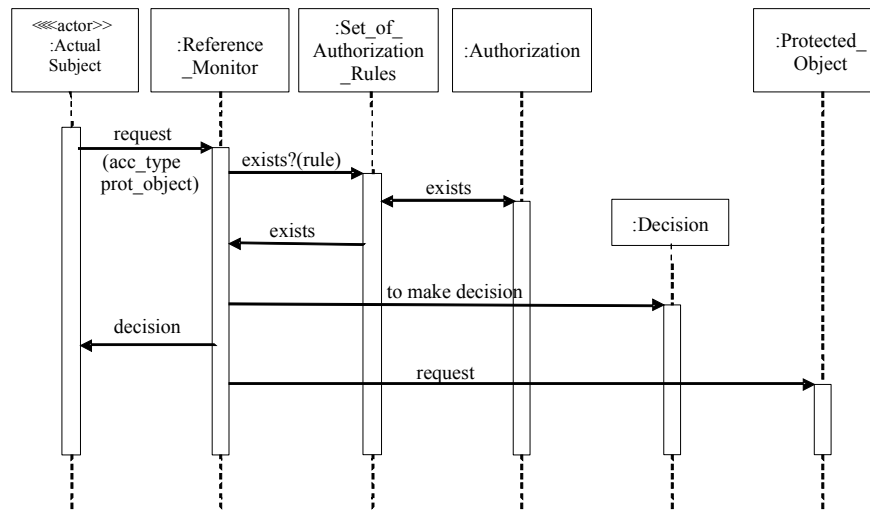


Fig. 2. Sequence diagram for enforcing security of requests.

4 Safety Reference Monitor

We see security conditions and safety assertions as similar (from the point of view of their enforcement). To enforce the constraints of the combined security/safety model we are proposing an Extended Reference Monitor which intercepts and verifies client requests for authorization and safety compliance. A UML class model for an Extended Reference Monitor is shown in Fig. 3. The Client sends requests to access an Object in a specific way (access type). The extended Reference Monitor separates the requests into security and/or safety requests. The Authorization Monitor has a set of authorization rules and enforces security constraints such as: "Only an authorized individual or role can change a valve setting". The safety Monitor enforces safety assertions: An operator may be authorized to send some command to a nuclear reactor but the conditions (state) are such that this command would produce an unsafe state, so the Safety Monitor rejects this command.

While the Authorization Monitor needs to control any types of accesses in the requests, the Safety Monitor only needs to consider requests that may change the state of the system. Both authorization rules and safety assertions need access to the state of the system in order to decide the validity of requests. We can classify different types of access constraints:

- Modal, based on current or temporal factors:
 - the current operating mode
 - access only if situation critical or non-critical
 - access only for off-line (test) operation
 - access only when resource hungry logging is active
- Schedule, based on schedule or process phase
 - only during quiet periods
 - only at scheduled time in process sequence
 - only when needed
- Discretionary, depending on the current load
 - access only when not busy relative to time constraints
 - access only when solar power is available
- Validated, involving active analysis
 - a semantic or pragmatic constraint analyzes the consequences of a request in the current context and performs checks against acceptable ranges.
- Preemptable, access could be cut off based on other control needs:
 - monitor plays a continuous and active role in fulfilling safety needs
 - access suspended while system performs schedule dependent task
 - access times out

These, and other constraints, can be modeled as conditions on the safety assertions and can be conveniently described in UML extended with OCL [14].

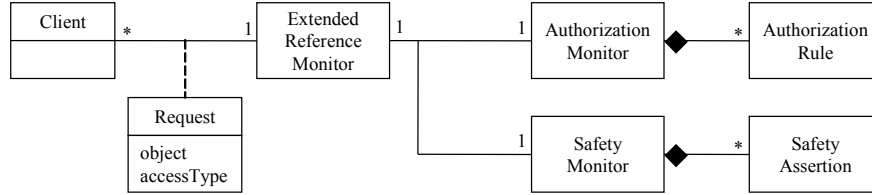


Fig. 3. Class diagram of the Extended Reference Monitor

5 Related Work

A systematic approach to safety-critical systems is proposed in [15]. However, they did not consider data aspects, they worked mostly with dynamic aspects described by statecharts. They don't consider security either. M. Tichy has studied pattern-based synthesis of fault-tolerant embedded systems [16]. His objective was to automate the application of fault tolerance into embedded systems. For this purpose, he developed a formal model based on Petri nets. In order to define the points where to apply fault tolerance mechanisms he considers the possible faults of the system. This last objective is similar to ours, but we analyze safety and security threats instead of reliability threats. L. Grunske focused on validation of safety properties: If a software architecture does not satisfy its requirements, it is modified by adding patterns that improve its safety properties [17]. His approach is based on transformational patterns that effectively perform model refactoring [18]. Y. Choi [19] converts use cases scenarios into activity diagrams (as we do in [20]), which in turn become Finite State Machines modeled using RSML-e, a formal language that has a model checker where hazard conditions can be identified.

Nonfunctional aspects such as security or safety have a strong effect on the resulting system architecture. A variety of approaches for software architecture design have been proposed [21]. However, all of them have relatively little to say about non-functional aspects, they mostly provide generic recommendations, with little detail on how to apply them. We follow mostly an approach based on RUP [22], but we will consider some of the other approaches if they have specific aspects that appear useful.

None of these studies explicitly considers the use of security patterns or provides a general methodology for secure or safe design. None of these approaches attempts to combine security and safety either. However, the work of Choi is very relevant and we are adapting some of her results to evaluate security threats. Closer in objectives are attempts for secure software development.

A general methodology for developing security-critical software which in particular can be used to develop secure DBMSs has been proposed in [23]. It makes use of an extension of the Unified Modeling Language (UML), UMLSec, to include security-relevant information. The approach is supported by extensive automated

tool-support for performing a security analysis of the UMLsec models against security requirements and has been used in a variety of industrial projects [24]. This model is clearly applicable to support our work.

Mouratidis and his group use a special methodology, Tropos, to model security. Their work started modeling requirements [25]. They have also looked at other stages; for example how to test security along the lifecycle. Instead of UML they use special diagrams and they do not use patterns. They have recently added models using UMLSec [26].

Only the last two of these approaches provide a relatively complete development methodology, the others consider only specific aspects. None of them considers safety or reliability aspects. We believe that there is a need for a complete development approach with a strong conceptual basis that combines security and safety. Our work is a step in that direction.

6 Conclusion

Many critical systems are data-driven systems. An air traffic control system that must keep track of many aircraft positions, a patient drug-delivery system where drug dosages and times are based on stored data, a building hosting a nuclear reactor where the movement of people must be restricted, are examples of this type of systems. Security is especially critical in these systems because an illegal modification of the related data could affect the safety of many people. Our Extended Reference Monitor can be the basis of an access control model that combines the specification and enforcement of security and safety for both information and physical structures [27]. We are implementing a preliminary version now. A byproduct of this work has been a new course on critical infrastructure systems [28].

A particularly interesting implementation approach is the use of components in a service-oriented architecture developed using an MDE approach [29], where the Extended Reference Monitor can be implemented using XACML. An important result of a systematic lifecycle for safety-oriented designs is a procedure for certification [30]. We intend to analyze how patterns can help for this purpose. Another aspect of future work is the integration of other security patterns [31] and fault-tolerance patterns [32]. The project started at Florida Atlantic University in the USA but it now includes participation from Argentina and Chile.

References

1. McMillan, R.: Critical Infrastructure Often Under Cyberattack. IDG News Service, November 11, 2008
2. Naedele, M.: Addressing IT Security for Critical Control Systems. In: 40th Hawaii International Conference on System Sciences, p. 115. IEEE, New York (2007).
3. N. Leveson, N., Safeware: System Safety and Computers. Addison-Wesley, Reading (1995)
4. Lautieri, S., Cooper, D., Jackson, D.: SafSec: Commonalities Between Safety and Security Assurance, In: Redmill, F., Anderson, T. (eds.) *Constituents of Modern Systems Thinking*:

- Proceedings of the Thirteenth Safety Critical Systems Symposium, pp. 65--78. Springer, London (2005)
5. Ligatti, J., Bauer, L., Walker, D.: Edit Automata: Enforcement Mechanisms for Run-time Security Policies. *International Journal of Information Security*, 4(1-2) 2--16. Springer (2005)
 6. Littlewood, B., Strigini, L.: Redundancy and Diversity in Security. In: 9th European Symposium On Research in Computer Security, pp. 42--428. Springer (2004).
 7. Fernandez, E.B., France, R.B.: Formal Specification of Real-Time Dependable Systems. In: 1st IEEE International Conference on Engineering of Complex Computer Systems, pp. 342--348. IEEE (1995)
 8. Fernandez, E.B.: Coordination of Security Levels for Internet Architectures. In: 10th International Workshop on Database and Expert Systems Applications, pp. 837--841. IEEE (1999)
 9. Buschmann, F., Meunier, R., Rohnert, H., Sommerlad, P., Stal, M.: *Pattern-Oriented Software Architecture Volume 1: A System of Patterns*. Wiley, West Sussex (1996).
 10. Gollmann, D.: *Computer Security*, 2nd Edition. Wiley, West Sussex (2006).
 11. Fernandez, E.B., VanHilst, M., Larrondo-Petrie, M.M., Huang, S.: Defining Security Requirements through Misuse Actions, In: Ochoa, S.F., Roman, G.-C. (eds.) *Advanced Software Engineering: Expanding the Frontiers of Software Technology*, pp. 123--137. Springer (2006)
 12. Braz, F.A., Fernandez, E.B., VanHilst, M.: Eliciting Security Requirements Through Misuse Activities. In: 19th International Workshop on Database and Expert Systems Applications, pp. 328--333. IEEE (2008)
 13. Fernandez, E.B.: Patterns for Operating Systems Access Control. In: 9th Pattern Languages of Programs Conference. Hillside Group (2002)
 14. Warmer J.B., Kleppe, A.G.: *The Object Constraint Language: Getting Your Models Ready for MDA*, 2nd Edition, Addison-Wesley, Boston (2003)
 15. Leveson, N.G., Heimdahl, M.P.E., Hildreth, H., Reese, J.D.: Requirements Specification for Process-Control Systems, *IEEE Trans. on Soft. Eng.*, 20(9), 684--707, (1994)
 16. Tichy, M.: Pattern-Based Synthesis of Fault-Tolerant Embedded Systems. In: *Doctoral Symposium of the Fourteenth ACM SIGSOFT Symposium on Foundations of Software Engineering*, pp. 1--18. ACM, New York (2006)
 17. Grunske, L.: Transformational Patterns for the Improvement of Safety Properties in Architectural Specification. In: 2nd Nordic Conference on Pattern Languages of Programs. Hillside Group (2003)
 18. Fowler, M.: *Refactoring: Improving the Design of Existing Code*, Addison-Wesley, Upper Saddle River (1999)
 19. Choi, Y.: Early Safety Analysis: from Use Cases to - Component-based Software Development. *Journal of Object Technology*. 6(8), 185--203 (2007)
 20. Fernandez, E.B., Larrondo-Petrie, M.M., Sorgente, T., VanHilst, M.: A Methodology to Develop Secure Systems Using Patterns, Chapter 5. In: Mouratidis, H., Giorgini, P. (eds.) *Integrating Security and Software Engineering: Advances and Future Vision*, pp. 107-126. IDEA Group, Hershey (2006)
 21. Hofmeister, C., Kruchten, P., Nord, R.L., Obbink, H., Ran, A., America, P.: A General Model of Software Architecture Design Derived from Five Industrial Approaches. *Journal of Systems and Software*, 80(1), pp. 106--126 (2007)
 22. Jacobson, I., Booch, G., Rumbaugh, J.: *The Unified Software Development Process*. Addison-Wesley, Upper Saddle River (1999)
 23. Jurjens, J.: *Secure Systems Development with UML*. Springer, Berlin (2004).
 24. Best, B., Jurjens, J., Nuseibeh, B.: Model-based Security Engineering of Distributed Information Systems Using UMLsec. In: 29th International Conference on Software Engineering (ICSE 2007), pp. 581--590. ACM, New York (2007)

25. Mouratidis, H., Giorgini, P., Manson, G.: Integrating Security and Systems Engineering: Towards the Modeling of Secure Information Systems. In: 15th Conference on Advanced Information Systems 2003. LNCS, vol. 2681, pp. 63--78. Springer, Heidelberg (2003)
26. Mouratidis, H., Jürjens, J., Fox, J.: Towards a Comprehensive Framework for Secure Systems Development. In: 18th Conference on Advanced Information Systems 2006. LNCS, vol. 4001, pp. 48--62. Springer, Heidelberg (2006)
27. Fernandez, E.B., Ballesteros, J., Desouza-Doucet, A.C., Larrondo-Petrie, M.M.: Security Patterns for Physical Access Control Systems. In: S. Barker, S., Ahn, G.J. (eds.), Data and Applications Security XXI, LNCS, vol. 4602, pp. 259--274, Springer (2007)
28. Fernandez E.B., Larrondo-Petrie, M.M.: A Course on Security for Critical Infrastructure Systems. In: 6th Latin American and Caribbean Conference for Engineering and Technology, (2008)
29. Delessy N., Fernandez, E.B.: A Pattern-Driven Security Process for SOA Applications. In: 3rd International Conference on Availability, Reliability, and Security, pp. 416--421. IEEE (2008)
30. Heimdahl, M.: Safety and Software Intensive Systems: Challenges Old and New. In: International Conference on Software Engineering 2007: Future of Software Engineering, pp. 137--152. ACM Press (2007).
31. Fernandez, E.B.: Security Patterns and a Methodology to Apply Them. In: Spanoudakis, G., Maña, A. (eds.) Security and Dependability for Ambient Intelligence. Springer (2009)
32. Buckley, I., Fernandez, E.B.: A Survey of Fault Tolerance Patterns. Department of Computer Science and Engineering, Florida Atlantic University (2008)

Detección y limitaciones de ataques clásicos con Honeynets virtuales

Hugo Fernández, Jorge Sznek, Eduardo Grosclaude
hugofernandeznqn@gmail.com, jsznez@uncoma.edu.ar,
oso@uncoma.edu.ar

Universidad Nacional del Comahue
Neuquén, Argentina

Resumen Las honeynets surgen como una herramienta de seguridad diseñada para ser sondeada, atacada y comprometida por hipotéticos intrusos. Se componen de entornos de redes, conjuntos de aplicaciones de análisis y monitoreo, y dispositivos de almacenamiento de eventos. Luego de realizada la instalación y configuración de todos estos componentes, la honeynet queda dispuesta para recibir ataques, con la intención de mantener un ambiente controlado para el estudio de los eventos ocurridos. Luego, mediante el análisis de esos eventos, es posible comprender los objetivos, tácticas e intereses de los atacantes sobre el entorno propuesto.

En el presente trabajo se han implementado dos honeynets virtuales con sus correspondientes herramientas de seguridad sobre diferentes topologías de red, a los efectos de estudiar en cada caso un conjunto de ataques previamente seleccionado. Como resultado de la experimentación, se ha logrado estudiar el impacto de cada uno de los ataques en cada una de las honeynets desplegadas, analizando lo ocurrido para finalmente proponer metodologías de prevención o mitigación de las vulnerabilidades encontradas.

PALABRAS CLAVE: honeypots, honeynets, virtualización, NIDS, HIDS.

1. Introducción

Las redes y sus aplicaciones, y en particular Internet, han introducido nuevas posibilidades que también implican riesgos. Éstos surgen a partir de las *vulnerabilidades* que poseen los sistemas. La existencia de vulnerabilidades implica *amenazas*, cuya concreción son los *ataques*.

Las vulnerabilidades pueden ser aprovechadas, con diversos fines, por muchas clases de *atacantes*: expertos o legos; interesados en el recurso de información que piensan comprometer, o motivados por intenciones en contra de la organización que atacan. En los últimos años, la frecuencia de aparición de ataques ha crecido considerablemente [13]. Este hecho, unido a las vulnerabilidades, descubiertas o latentes, en todo tipo de sistemas operativos y aplicaciones, convierte a cualquier

organización en una víctima potencial. Este panorama plantea la necesidad de disponer de instrumentos que permitan descubrir y analizar los *agujeros* de seguridad que pueda presentar un sistema, así como las técnicas y herramientas utilizadas por los posibles atacantes.

Para prevenir o mitigar cualquier tipo de amenaza es necesario conocer y comprender las vulnerabilidades constitutivas del entorno. Una de las metodologías para esto es crear un ambiente de red controlado pero a la vez lo suficientemente atractivo para los atacantes, que permita detectar comportamientos maliciosos, para estudiarlos, entenderlos y actuar en consecuencia, ya sea de una manera proactiva o reactiva, sin perjudicar el ambiente de producción de la organización. Este es el fundamento de las *honeynets* [5].

Estos ambientes pueden construirse en base a sistemas total o parcialmente compuestos por elementos virtuales. La idea de virtualización ha sido aplicada, desde hace bastante tiempo, a diferentes aspectos y ámbitos de la informática, desde sistemas computacionales completos hasta capacidades o componentes individuales. El tema en común de todas las tecnologías de virtualización es la de ocultar los detalles técnicos a través de la encapsulación. La virtualización crea una interfaz externa que esconde una implementación subyacente mediante la combinación de recursos en locaciones físicas diferentes, o mediante la simplificación del sistema de control. Un reciente desarrollo de nuevas plataformas y tecnologías de virtualización han hecho que se vuelva a prestar atención a este maduro concepto.

El presente trabajo comprende el diseño, implementación y despliegue de dos *honeynets* virtuales, la selección de cada una de las aplicaciones que la conforman y el conjunto de ataques a las cuales serán sometidas. El objetivo consiste en contrastar la información suministrada por cada uno de los despliegues, analizar dicha información, brindar una explicación relativa a lo ocurrido a partir de la información recolectada y proponer las distintas maneras de mitigar cada uno de estos ataques.

En la sección siguiente nos referiremos al modelo de *honeynet* empleado en nuestro trabajo; en la sección 3 describiremos las topologías experimentales desplegadas; en la sección 4 se enumerará el software utilizado en la implementación. En la sección 5 describiremos los ataques realizados, detallando para cada uno:

- Su objetivo
- Las precondiciones que lo hacen posible
- El software utilizado en el ataque
- El desarrollo del ataque
- Resultados obtenidos y análisis
- Contramedidas para detectar o neutralizar el ataque

Finalmente, en la sección 6 daremos las conclusiones globales del trabajo y algunas propuestas de trabajos futuros relacionados.

2. Honeynets

La implementación de honeynets debe tener en cuenta dos aspectos importantes:

- Control de datos: Define el grado de vulnerabilidad de los sistemas expuestos para que sea apetecible para el atacante y también el grado de restricción para no permitir la pérdida total del control.
- Captura de datos: Define qué herramientas y métodos se van a utilizar para el monitoreo y registro de toda la actividad efectuada dentro de la honeynet.

El control de datos tiene dos objetivos: el primero, es presentar un escenario lo suficientemente concreto para dar a los intrusos la sensación de que poseen el control del ataque, y, fundamentalmente, para que actúen como suelen hacerlo al momento de comprometer otras máquinas víctimas. El segundo, es lograr que el ambiente generado garantice la confinación del daño que pueda ser ocasionado al ambiente de la honeynet.

La captura de datos es una funcionalidad que permite recolectar y almacenar la evidencia tanto de la actividad de la red como de las máquinas intervinientes en los ataques a los que fueron expuestas. No se trata solamente de almacenar el log o la traza del tráfico de red, sino que es una combinación de monitoreo de la actividad, observación del *modus operandi* del atacante y capacidad de reconocer los distintos perfiles de ataques, para interpretarlos mejor.

2.1. Arquitecturas.

Podemos diferenciar dos clases de honeynets: las de *primera generación* y las de *segunda generación*.

Honeynets de primera generación

Su arquitectura está compuesta de un firewall, responsable del control de datos, un sistema de detección de intrusos (Intrusion Detection System, o IDS), responsable de la captura de datos, un servidor centralizado de logs y un conjunto de máquinas víctimas o sistemas señuelos (*honeypots*). El IDS puede especializarse en eventos locales al host donde está instalado (HIDS) o eventos propios de la red (NIDS) [6].

Honeynets de segunda generación

En las honeynets de primera generación, no se contaba con la posibilidad de monitorear a los atacantes por un período de tiempo lo suficientemente largo sin que éstos detectaran la presencia de la propia honeynet. Esto se debió a limitaciones inherentes a la propia implementación: primero, las restricciones en el número de conexiones salientes desde la honeynet, y segundo, el excesivo uso de comunicaciones sobre la capa de red (capa 3 del modelo OSI [15]) en

dispositivos críticos como firewall, IDS y servidores de logs. Al revelarse su existencia, se incrementaba notablemente el riesgo de convertirse en objetivos de ataque.

La arquitectura de una honeynet de segunda generación está compuesta por un equipo denominado *honeywall*, que cumple el papel de firewall e IDS simultáneamente, y un conjunto de honeypots.

2.2. Honeynets virtuales

Las honeynets virtuales son un concepto relativamente nuevo. La idea consiste en combinar todos los elementos físicos de una honeynet dentro de una única computadora, utilizando para ello software de virtualización.

Las principales ventajas de las honeynets virtuales son la gran reducción en los costos, la portabilidad y la facilidad tanto en su despliegue como en el mantenimiento de toda la infraestructura, ya que todo se encuentra integrado en un único sistema. Por otro lado, presentan ciertos problemas.

- Limitaciones inherentes: Limitados tipos de sistemas operativos que se pueden desplegar debido al hardware subyacente y al software de virtualización utilizado.
- Incremento en los riesgos: Los atacantes podrían comprometer la seguridad de la máquina física, obteniendo todo el control de la honeynet, y corromper todos los mecanismos tanto de control como de captura de información.
- Riesgo de *fingerprinting*: fingerprinting es la habilidad para identificar la honeynet en forma remota o local. Las honeynets virtuales poseen firmas que las hacen únicas (como resultado de la virtualización). Los atacantes podrían indentificar dichas firmas y detectar de esta manera el propósito de nuestra honeynet.

3. Topologías experimentales

Debido a la necesidad de generar diferentes escenarios para llevar a cabo el desarrollo de la experimentación, seleccionamos dos tipos de topologías distintas, tomando como modelos aquellas de uso más común en ámbitos hogareños y de la pequeña y mediana organización. En ambos casos la base física será una única computadora, y el resto de los dispositivos y conexiones serán emulados por virtualización.

Topología virtual I

Es una red donde todos los equipos involucrados, ya sean virtuales o físicos, e incluyendo estaciones honeypot y atacante, están sobre el mismo dominio de broadcast. Comparten un mismo espacio IP y todas están sobre el dominio de alcance de protocolos como ARP o DHCP. La topología experimental refleja el caso real en el cual la red sometida a ataques es una LAN y el atacante es un miembro de la misma, posiblemente una máquina comprometida por un troyano (Figura 1).

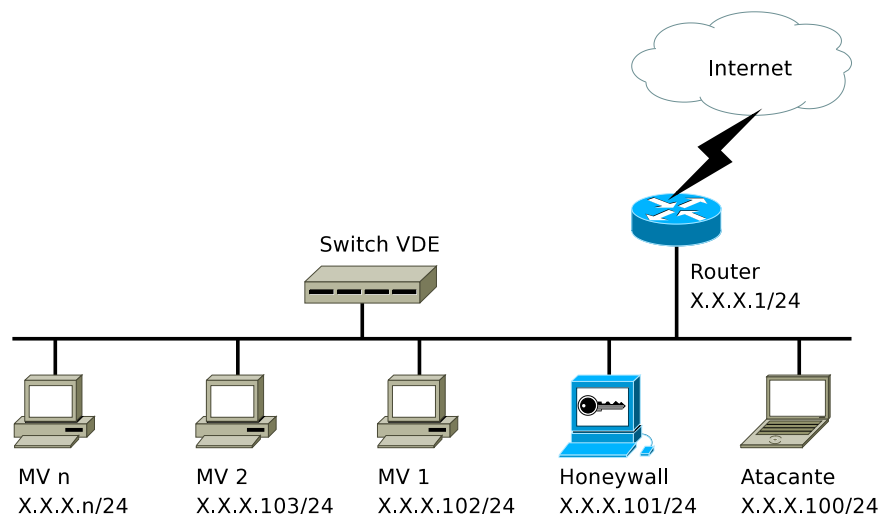


Figura 1. Topología I, dominio de broadcast único

Topología virtual II

Existe un elemento de conmutación que hace traducción de direcciones (NAT) y el espacio IP de los honeypots está separado del espacio IP del atacante. El router intermedio contiene los broadcasts, de modo que la difusión de protocolos como ARP y DHCP está restringida al espacio local de los honeypots. La topología refleja el caso en que el atacante se sitúa fuera de la red que intenta atacar (Figura 2).

4. Software utilizado en la experimentación

Para lograr la captura de la información acerca de los ataques o anomalías que se producen en la red trampa se utilizó un conjunto de programas, cada uno de éstos instalado estratégicamente en los nodos de interés a lo largo de la red.

- QEMU: VMM o software para administrar las máquinas virtuales [4].
- VDE-SWITCH: para efectuar la intercomunicación entre las máquinas virtuales y la máquina física [14].
- SNORT: para ser usado como IDS de red (NIDS) [3].
- OSSEC: para ser usado como IDS de host (HIDS)[10].

Además se han utilizado las siguientes herramientas:

- MYSQL: motor de base de datos utilizado para almacenar toda la actividad registrada por SNORT [16].
- APACHE: servidor web para presentar los resultados [2].

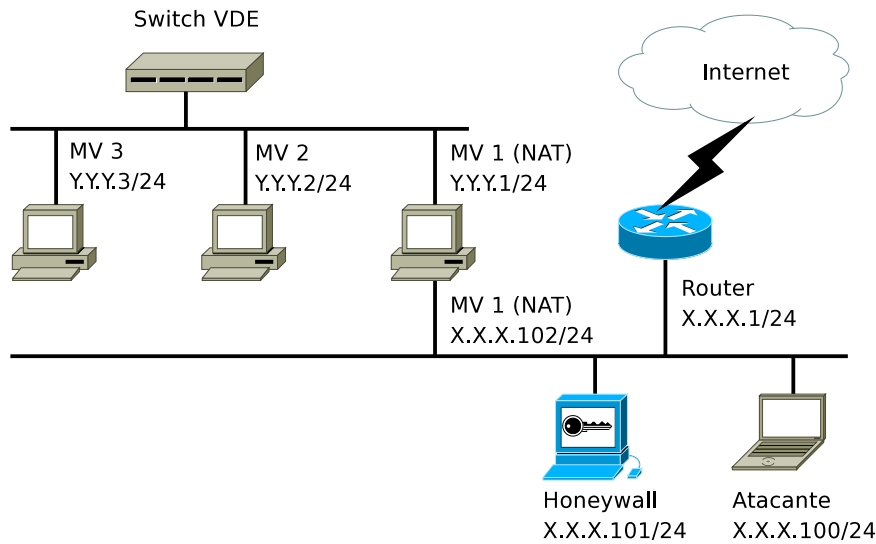


Figura 2. Topología II, traducción de direcciones

- SNORT REPORT: aplicación web PHP para poder visualizar en tiempo real todo lo registrado por Snort en el motor de base de datos, mediante consultas programadas, mostrando gráficos acerca de lo sucedido [17].

Las motivaciones que llevaron a la elección de QEMU sobre otras herramientas de virtualización similares fueron:

- Software libre, licenciado bajo la General Public License (GPL).
- Proyecto en desarrollo y con gran cantidad de documentación.
- Se puede implementar en distintos sistemas operativos y sobre distintas plataformas de hardware.
- Emula completamente el hardware subyacente, logrando una transparencia que no es alcanzada por otros productos de virtualización.
- Presenta performance aceptable con el módulo qemu (módulo del Kernel Linux que acelera la emulación de arquitecturas i386)
- Manejo conveniente de imágenes de disco, en varios formatos alternativos.

5. Ataques, experimentación y resultados

En esta sección se explicarán los cuatro ataques desarrollados: ArpSpoofing, Fuerza Bruta, Flooding y Keylogging.

5.1. ArpSpoofing

Descripción El protocolo ARP (Protocolo de Resolución de Direcciones) se encarga de la resolución de la dirección de hardware que corresponde a una

determinada dirección IP [1]. Ciertas características del protocolo ARP facilitan su uso de manera ilegítima para recibir tráfico ajeno. En particular, nos resultan relevantes las siguientes:

- Ausencia de autenticación en el protocolo. Es decir, una máquina modificará su comportamiento de acuerdo a los paquetes ARP recibidos, sin poder determinar de ningún modo la autenticidad de los mismos.
- Es posible -aunque evitable- modificar los contenidos de una caché ARP tan sólo con construir y enviar una petición o respuesta adecuada.
- Una máquina puede actualizar las caches ARP del resto de los sistemas de la red en cualquier momento.
- Como el protocolo ARP trabaja a nivel de capa de enlace de datos, esta técnica sólo puede ser utilizada en el ámbito de un dominio de broadcast.

El atacante aprovechará estas características en la técnica del *envenenamiento ARP*, o *ARP spoofing* [18], para denegar el servicio de la máquina remota, recibiendo tráfico ajeno en una red construida con switches. Mediante respuestas ARP, el atacante modificará el contenido de la caché de la víctima de forma que para la dirección IP de su interlocutor se corresponda la dirección MAC real del atacante. La idea consiste en “envenenar” la caché ARP de la máquina víctima con la dirección MAC del atacante, logrando de esta manera un ataque DoS ya que las peticiones de la víctima nunca serán respondidas.

Precondiciones Las precondiciones del ataque son las siguientes:

- La máquina atacante, conociendo las direcciones IP de los dos nodos cuyas comunicaciones quiere intervenir, resuelve mediante ARP las direcciones MAC que les corresponden.
- La máquina víctima está encendida y sin problemas de conectividad.
- Tanto el host víctima como el host atacante están dentro del mismo dominio de broadcast.

Debido a esta última precondición, el ataque estudiado no puede aplicarse en la topología 2 pues existe un router intermedio que separa la red donde se encuentra el host víctima de la del atacante.

Software utilizado La distribución Linux usada para realizar el ataque es BACKTRACK 2 versión Live, con software ARPSPOOF.

Ataque Para producir el “envenenamiento” de la caché ARP de la máquina víctima (IP 192.168.2.104) con el número MAC del atacante para todas las peticiones al gateway local (IP 192.168.2.1), el atacante inicia el ataque ejecutando el comando *arpspoof -t 192.168.2.104 192.168.2.1*.

Como consecuencia se provoca una denegación de servicio pues la máquina víctima no puede desempeñar su trabajo con normalidad. Por ejemplo, las aplicaciones que corren en ella no pueden navegar por internet porque al realizar peticiones al gateway, éste dirige el tráfico hacia la máquina atacante debido a la modificación producida en la caché ARP del host víctima.

Resultados y análisis El reporte de SNORT muestra el mensaje “*ICMP destination Unreachable - Communication with Destination Host is Administratively prohibited*”, identificando completamente al atacante por su número de IP y su FQDN (Fully Qualified Domain Name). Un análisis más exhaustivo permite ver los detalles del suceso ocurrido a nivel de paquetes de red. A través del NIDS se pudo obtener información relevante sobre el ataque:

- Fecha y hora del evento de seguridad
- IP y FQDN del atacante.
- Reporte de detalles acerca del ataque a nivel de paquetes de red.

Por otro lado, el HIDS instalado en la máquina víctima no detectó nada que pudiera brindar indicios sobre la ocurrencia del ataque, ya que se trata de un ataque de red.

Contramedidas y elementos para la detección La defensa más completa contra el ARP Spoofing es activar la vinculación por MAC en el switch, que consiste en no permitir el cambio de la dirección MAC asociada a un puerto una vez que ha sido establecida. Los cambios legítimos podrán ser realizados por el administrador caso por caso.

Otra defensa es utilizar tablas estáticas. La cache ARP puede tener entradas estáticas de modo que los paquetes falsificados sean ignorados. Sin embargo, esta técnica no resulta muy práctica para redes corporativas pues las mismas están integradas por una gran cantidad de equipos.

En pruebas realizadas sobre sistemas operativos Windows se encontró que se siguen aceptando entradas ARP falsas y se siguen utilizando entradas dinámicas en lugar de las estáticas. Estos resultados vuelven nula la prevención mediante el uso de rutas estáticas.

Además, es recomendable utilizar métodos por software para detectar estos ataques, tales como:

- Arpwatch: programa que monitorea una red de computadoras desde la actividad ARP, generando un archivo de logs de los pares de direcciones (IP, MAC) a lo largo de un período de tiempo. La principal y más importante razón de monitorear la actividad ARP es detectar ARP spoofing [8].
- Ettercap: es un programa que posee la capacidad de interceptar segmentos de tráfico de red para luego efectuar un análisis de dicha captura [11].

5.2. Fuerza Bruta

Descripción El objetivo de este ataque es ingresar al sistema víctima con credenciales de administrador (*root*) a través de SSH v2.0. Para ello se apelará a la metodología de fuerza bruta, basada en diccionario, para obtener la contraseña del administrador. Una vez ingresado, el atacante desarrolla alguna acción maliciosa tal como instalar algún troyano con el cual garantizar el acceso en caso de que se cambie la contraseña, modificar la contraseña para denegarle el acceso al

usuario real, abrir puertos para que un segundo atacante pueda ejecutar alguna otra acción maliciosa, etc.

El mecanismo de autenticación basado en usuario y contraseña es vulnerable a la impostura. Un atacante que conozca, o conjeture, la combinación (usuario, password) correcta, obtendrá acceso al sistema sin otra condición. Un ataque muy común que hace uso de las vulnerabilidades de autenticación son los ataques basados en fuerza bruta (brute force attack), que consiste en bombardear al servidor con nombres de usuarios y contraseñas aleatoriamente generados. Una manera más “inteligente” de realizar ataques de fuerza bruta es mediante el uso de diccionarios de contraseñas, pues en muchas ocasiones las contraseñas utilizadas para los usuarios administrador son palabras comunes, secuencias de números, secuencias de teclas del teclado, etc. El país, cultura, idioma, etc., del sistema víctima cuentan en la elección de diferentes diccionarios.

Los ataques por fuerza bruta están muy extendidos debido a que con un simple script se puede testear cientos o miles de servidores de forma automática sin necesidad de intervenir manualmente en el proceso.

Precondiciones Las precondiciones del ataque son las siguientes:

- El archivo de configuración de SSH permitirá una cantidad de 29 reintentos de inicio de sesión.
- Se ubicó la clave correcta en la posición 20 del extracto del archivo de contraseñas de JOHN THE RIPPER para poder efectuar un inicio de sesión exitoso luego de varios intentos fallidos.
- Ya que se ataca al usuario administrador de la máquina víctima, se supone que dicho usuario tiene acceso mediante SSH a ese sistema.

Software utilizado

- La distribución Linux usada para realizar el ataque es BACKTRACK 2, versión Live.
- Para escanear la red y localizar posibles víctimas se usó el programa NMAP [7].
- Para el ataque se utilizó el programa MEDUSA [9].
- El diccionario usado pertenece a un extracto del programa JOHN THE RIPPER [12](Incluido por defecto en la distribución).
- El servidor SSH de la víctima está incluido por defecto en la versión de FEDORA CORE 5 instalada en la máquina víctima.

NMAP es una utilidad para explorar y auditar redes informáticas. Utiliza los paquetes IP para investigar qué direcciones de red están asignadas a los hosts, qué servicios (nombre y versión de aplicación) están ofreciendo, qué sistemas operativos (y cuáles versiones) están en funcionamiento, qué tipo de filtros de paquetes o cortafuegos están en uso, y muchas otras características. Este programa fue diseñado para explorar rápidamente grandes redes.

MEDUSA es un programa que tiene como finalidad el acceso por fuerza bruta a una máquina víctima. Está elaborado en forma modular y con funcionamiento en paralelo, resultando muy veloz. JOHN THE RIPPER es un veloz y efectivo detector de debilidades y descriptador de contraseñas. Actualmente disponible para muchos tipos de Unix, Windows, DOS, BeOS, y OpenVMS.

Ataque Lo que se señala a continuación es válido para ambas topologías.

- El atacante inicia su ataque utilizando el programa MEDUSA y el archivo de contraseñas de JOHN THE RIPPER, mediante la ejecución del comando *medusa -h 192.168.2.104 -u root -r3 -P /root/Desktop/listapwds.txt -M ssh*.
- Durante el ataque, el programa recorre automáticamente los datos contenidos en el archivo de contraseñas y se conecta vía SSH para probar cada una de ellas, utilizando el usuario administrador (*root*). Finalmente, obtiene acceso al sistema en el reintento número 20 y, como consecuencia, el sistema víctima es vulnerado y se obtiene el control de la máquina con todos los privilegios de super usuario.

Resultados y análisis El NIDS utiliza un sniffer para realizar las capturas de los paquetes que viajan por la red y los analiza según reglas incorporadas. Debido a que las sesiones generadas a través de SSH están cifradas, el NIDS no logra encontrar ninguna anomalía ya que no puede efectuar ningún análisis posible para este caso.

El HIDS captura las secuencias de reintento para la validación de usuario y muestra los intentos del atacante para ingresar al sistema.

Los intentos de acceso quedaron registrados en los logs del sistema operativo (/var/log/secure).

Finalmente, se puede concluir que el ataque es detectado una vez que se comienza a interactuar con la máquina víctima, independientemente de la topología de red en la que ésta se encuentre.

Contramedidas Algunas medidas que se pueden tomar para disminuir el riesgo de penetración pueden ser:

- Nunca permitir el acceso remoto al usuario *root* (administrador). Es preferible el acceso con un usuario no privilegiado y una vez dentro del sistema, adquirir privilegios de administrador.
- Utilizar SSH sólo dentro de la red interna, bloqueando el acceso al puerto 22 mediante un firewall.
- Utilizar contraseñas difíciles de conjeturar. Por ejemplo: más de 8 caracteres, mezcla de letras (mayúsculas y minúsculas), números, símbolos, etc. Preferentemente considerar algún esquema de contraseñas dinámicas Ej.: S/Key o SecurID.
- Activar la caducidad de las contraseñas de manera de renovarlas con frecuencia.

5.3. Flooding

Descripción Las organizaciones hacen uso extensivo de los servicios de Internet a través de su propia infraestructura, con el objetivo de mejorar sus operaciones o reducir sus costos. Ya sea publicando páginas web, utilizando un servidor de correo electrónico propio o brindando acceso remoto para teletrabajo, se crean canales abiertos de comunicación. Estos canales o vías de ingreso virtual pueden ser utilizados en forma maliciosa atacando su disponibilidad.

Este ataque tiene como principal objetivo saturar las conexiones al puerto SSH (port 22) abriendo el mayor número de sockets posibles en la víctima, llegando a provocar de esta manera una denegación de servicio.

Software utilizado

- La distribución Linux usada para realizar el ataque es BACKTRACK 2, versión Live.
- Para el ataque se utilizó OCTOPUS, programa que satura las conexiones a un puerto específico, dirigiendo su ataque al puerto 22 (SSH).
- Se utilizó el compilador GCC versión 4.2.1 para generar código ejecutable.

Precondiciones Dado que se trata de un ataque desde un host remoto (perteneciente a otra red), no corresponde su aplicación en la Topología I.

- La máquina víctima debe estar encendida y funcionando en red.
- El puerto que se ataca debe estar abierto y visible para el atacante.

Ataque El atacante inicia el ataque ejecutando el comando

octopusInf 192.168.2.102 22

Con esto se pretende inundar de conexiones el puerto 22 de la máquina cuya IP es 192.168.2.102.

Durante el ataque se abren n conexiones al puerto 22 utilizando para ello n puertos del atacante, provocando la denegación de servicio del puerto que es atacado, impidiendo el acceso a este servicio a cualquier usuario, y provocando una gran lentitud en el desempeño de la máquina virtual.

Resultados y análisis De los datos obtenidos por las herramientas dispuestas en la honeynet, se puede concluir que el NIDS no aporta datos sobre el ataque, porque el patrón de tramas que circula por la red virtual es considerado como válido, pues corresponde a intentos de establecer una conexión con un servicio.

Por otro lado, cuando llegan las peticiones el HIDS detecta que el usuario que está intentando establecer la conexión con el servicio no está ingresando los datos de identificación (usuario y contraseña). Este no es un dato menor, ya que si tantas conexiones en tan poco tiempo acusan ese problema es muy probable que se esté frente a un ataque.

Contra medidas

- Utilizar un firewall para restringir el acceso al puerto para el uso externo o bien cerrarlo.
- Nunca utilizar el servicio con usuario *root* (administrador).
- Configurar el IDS para tomar medidas frente a esta clase de ataques. En este caso OSSEC provee la posibilidad de efectuar “respuestas activas”, utilizando para ello un demonio llamado *ossec-excd*. Por ejemplo permite efectuar: denegación de acceso al host; ejecutar determinadas reglas de firewall (cerrar el servicio para determinada dirección IP), etc.
- Utilizar OSSEC configurando el demonio *ossec-maild*, para que frente a algún problema alerte al administrador mediante el envío de un mail en el momento que ocurre el problema.
- No utilizar el servicio.

5.4. Linux Key Logger

El ataque tiene como principal objetivo capturar toda la información que la víctima ingrese por el teclado con el fin de obtener información sensible, dándole al atacante la posibilidad de realizar posteriormente ataques más específicos, como pueden ser: acceder a cuentas de correo electrónico, conocer las páginas que visita el usuario, acceder al *home banking* utilizando usuario y contraseña legítimos, entre otros.

Descripción Un *keylogger* es una herramienta que se encarga de registrar las pulsaciones que se realizan sobre el teclado, para guardarlas en un archivo o enviarlas a través de la red. El registro de lo que se teclea se puede hacer tanto por hardware como por software. Los sistemas comerciales disponibles incluyen dispositivos que pueden conectarse al cable del teclado y al teclado mismo. La escritura de aplicaciones para realizar keylogging es simple y, como cualquier programa computacional, puede ser distribuido masivamente a través de un troyano o como parte de un gusano.

A veces se afirma que la utilización de “teclado virtual” evita la captura de las pulsaciones del teclado al requerir únicamente clicks del mouse. Sin embargo, ya que los eventos de mensajes del teclado deben ser enviados al programa externo para que se escriba el texto, en principio cualquier keylogger podría registrar lo escrito mediante un teclado virtual. Por otro lado, existe malware que graba videos del movimiento del teclado mientras opera en un teclado virtual, lo cual lo inutiliza completamente como medida de prevención.

Software utilizado Para realizar el ataque se utilizó el software LINUX KEY LOGGER (LKL), que es un capturador de teclado que funciona en espacio de usuario bajo el sistema operativo Linux.

LKL captura las pulsaciones del teclado ocultando el puerto 0x60 y crea un archivo de log con todos los datos recolectados, previa traducción a código ASCII

utilizando un archivo de mapa del teclado. Además ofrece la facilidad de enviar por e-mail al atacante los datos recolectados, luego de llegar a algún tamaño de archivo previamente configurado.

Precondiciones

- El atacante es capaz de instalar y configurar en la máquina víctima un programa para que se ejecute el programa LKL cada vez que el usuario ingresa al sistema.
- El usuario víctima puede enviar correo electrónico.
- El usuario víctima usa su computadora en forma habitual y además navega por Internet visitando páginas interesantes para el atacante, como aquellas de *home banking* o webmail.

Estas precondiciones son válidas para ambas topologías.

Ataque El ataque se desarrolla utilizando el programa LKL y el archivo de mapeos de código ASCII KEYMAP. El comando ejecutado es:

```
lkl -l -k /root/Desktop/lkl/keymaps/it_km -o log.file -m atacante@mail.com;
```

donde:

- l determina que se haga log del puerto del teclado (0x60).
- k establece la ruta del “keymap” que viene con la aplicación.
- o especifica el nombre y extensión del archivo log que se va a generar.
- m especifica la dirección de mail donde se enviarán los datos recogidos por la aplicación.

Por defecto, la aplicación envía datos al atacante cada 1 KB de información recopilada y continúa su funcionamiento.

Resultados y análisis SNORT no detecta ningún tipo de anomalía aparente, debido a que el único tráfico de red es la salida de un mail. OSSEC tampoco encuentra ningún patrón irregular que permita deducir algún evento de seguridad.

De los datos obtenidos por las herramientas dispuestas en la honeynet, se puede concluir que el NIDS no aporta datos sobre el ataque, pues el patrón de tramas que circula por la red virtual es considerado como válido por el IDS; y efectivamente lo es, ya que corresponde simplemente al envío de mensajes de email de un usuario válido.

Con respecto al HIDS, las herramientas de detección instaladas y la disposición de las topologías no fueron suficientes para detectar el *daemon* que está a la escucha del puerto del teclado; por lo tanto, no pudo ser identificado el ataque.

Finalmente, se puede concluir que para este conjunto de herramientas, los resultados son equivalentes en ambas topologías y no son suficientes para detectar el ataque.

Contramedidas Algunas medidas que se pueden tomar para detectar la existencia de un keylogger son:

- Acceder a un programa monitor de procesos para ver lo que está siendo ejecutado por la computadora en ese momento. En forma complementaria, es aconsejable contar con software anti-rootkit para detectar procesos ocultos.
- Utilizar software anti-spyware o anti-keylogging, que detectan a los keyloggers conocidos mediante el uso de firmas.

Existen algunas medidas que dificultan o impiden el ataque:

- Usar un firewall para evitar la transmisión del material registrado sobre la red.
- Ciertas prácticas permiten “engañar” a los keylogger, evitando la captura de la información al usar la secuencia de copiar/pegar texto.

6. Conclusiones y trabajo futuro

El resultado concreto de este trabajo es el diseño e implementación de honeynets virtuales. Para lograrlo, se ha aplicado un conjunto de herramientas de seguridad a dos diferentes topologías. Para demostrar el funcionamiento de cada instancia de honeynet, se ha preparado un conjunto de ataques que fueron finalmente ejecutados en cada topología seleccionada.

El resultado de la experimentación permitió:

- Visualizar la explotación de las vulnerabilidades expuestas.
- Confirmar que los IDS detectaron la mayoría de los patrones de ataques.
- Encontrar información sobre el atacante (Ej. IP y FQDN).
- Confirmar que utilizar herramientas que muestren los eventos de seguridad ocurridos de una manera comprensible y ordenada nos ayuda a implementar distintos tipos de solución para cada problema encontrado.
- Conocer cuáles son las vulnerabilidades del sistema y establecer a futuro políticas de seguridad que minimicen los riesgos para que alguno de estos ataques sea llevado a cabo.
- Descubrir que las herramientas dispuestas no fueron suficientes para la detección de la ejecución del ataque en ninguna de las topologías propuestas. (Ej: Durante el ataque utilizando linux key-logger).
- Confirmar que el uso de distintas topologías de red tiene una gran incidencia en cuanto a la seguridad. (Ej: Para la topología II el ataque de envenenamiento de caché arp no es implementable por cuestiones de funcionamiento de la propia topología.)

El resultado de la presente experiencia puede ser tomado como punto de partida para realizar otros casos de estudio utilizando honeynets. Algunos de estos casos de estudio podrían ser:

- Exponer una arquitectura honeynet a Internet, con el objetivo de visualizar, estudiar, clasificar y documentar los ataques encontrados, ya sean éstos ataques conocidos o no.

- Implementar una arquitectura honeynet dentro de alguna organización pequeña o mediana replicando el funcionamiento de la infraestructura existente, con todos los servicios necesarios y/o existentes: Ej: Servidor de correo electrónico, controladores de dominio, servidores web, DNS, DHCP, etc.
- Realizar un conjunto de ataques que explote las vulnerabilidades existentes, para finalmente proponer y documentar un conjunto de políticas de seguridad que ayude a mejorar la seguridad en dicha red.
- Implementar una arquitectura honeynet distribuida y enviar toda la información recolectada, utilizando una red privada virtual, al servidor de base de datos central que almacena toda la información de lo ocurrido en cada uno de los honeypots remotos. De esta manera, analizar las distintas tendencias de los atacantes en cada sitio remoto.

A modo de conclusión general, el diseño e implementación de honeynets no es un problema trivial y existen múltiples maneras de realizarlo. Diversas elecciones definen el tipo de honeynet a implementar (primera generación, segunda generación, honeynets virtuales, honeynets distribuidas, etc.), la definición de los grados de libertad y restricciones que se puedan aplicar para cada despliegue (i.e. conjunto de herramientas y políticas de seguridad que deban ser respetadas), y por supuesto, el conjunto de servicios que serán expuestos.

Referencias

1. Plummer David C. Rfc-826: An ethernet address resolution protocol - or - converting network protocol addresses to 48.bit ethernet address for transmission on ethernet hardware. 1982.
2. Apache Community. The apache software foundation. <http://www.apache.org>, 1999.
3. Snort Community. Snort (ids/ips). <http://www.snort.org>, 1998.
4. Bellard Fabrice. Qemu. <http://www.qemu.org>, 2005.
5. López de Vergara J. Gallego E. Honeynets: Aprendiendo del atacante, 2004.
6. D. Gonzalez Gomez. Sistemas de detección de intrusiones versión 1.01, cap. 4. <http://www.dgonzalez.net/pub/ids/html/index.htm>, 2003.
7. Lyon Gordon. Nmap. <http://nmap.org/>, 1997.
8. LBNL's Network Research Group. arpswatch. <http://www-nrg.ee.lbl.gov/>.
9. JoMo-Kun. Medusa. <http://www.foofus.net/jmk/medusa/medusa.html>.
10. Cid Daniel B.; Panda Partha; Baig Bilal; Rockburn Lynn. Ossec: Open source host-based intrusion detection system. <http://www.ossec.net>, 2008.
11. Valleri Marco Ornaghi Alberto. Ettercap. <http://ettercap.sourceforge.net>, 2001.
12. Openwall Project. John the ripper. <http://www.openwall.com/john/>.
13. The Honeynet Project. Know your enemy - learning about security threats, 2002.
14. Davoli Renzo. Virtual square. <http://wiki.virtualsquare.org>, 2006.
15. Tanenbaum Andrew S. *Redes de computadoras*. Pearson Educación, 2003.
16. Inc. Sun Microsystems. Mysql: Open source database. <http://www.mysql.com/>, 1995.
17. Symmetrix Technologies. Snort report. <http://www.symmetrixtech.com>, 2007.
18. Volobuev Yuri. *Redir games with ARP and ICMP*. 1997.

UN MODELO COMPACTO DE CRIPTOGRAFÍA ASIMÉTRICA EMPLEANDO ANILLOS NO CONMUTATIVOS

Juan Pedro Hecht

Carrera de Especialización en Criptografía y Seguridad Teleinformática – EST (IESE) *
Carrera de Especialización y Maestría en Seguridad Informática – UBA **

* Escuela Superior Técnica – Instituto de Enseñanza Superior del Ejército (IESE)
Av. Cabildo 15 - (C1426AAA) - Buenos Aires – Argentina

** Facultad de Cs. Exactas y Naturales/Facultad de Ingeniería/Facultad de Cs. Económicas
Universidad de Buenos Aires – Argentina
phecht@dc.uba.ar
<http://cripto.iese.edu.ar/>

Resumen. La criptografía asimétrica (AC) ha estado presente desde los trabajos pioneros de Diffie, Pohlig, Merkle y Hellman. Prácticamente todos los protocolos criptográficos AC están vinculados a funciones trampa de una vía. Hoy día las primitivas de mayor difusión están vinculadas al problema de la factorización de enteros o al problema del logaritmo discreto. Ambos están íntimamente vinculados y pertenecen en general a operaciones aritméticas en estructuras algebraicas conmutativas (grupos, anillos y campos). Una desventaja reside en que se conocen ataques de complejidad subexponencial (y tal vez existan métodos de resolución en tiempo polinómico) y ataques eficientes a través de algoritmos cuánticos, de modo que hay serios motivos para desconfiar de los mismos. Para reemplazar de raíz a la AC convencional (y su aplicación en criptografía de clave pública), ha surgido en años recientes un interés creciente en la aplicación de estructuras algebraicas no conmutativas. Con este objetivo en mente, se desarrollaron sistemas AC usando grupos de distinta naturaleza (trenzas, Thompson, nilpotentes, simétricos, policíclicos) y recientemente anillos de polinomios matriciales. Hay que enfatizar que todos usan, de una forma u otra, bibliotecas de precisión múltiple de manera que se complica su aplicación en plataformas de baja capacidad computacional. En este trabajo presentamos una solución (la definimos como *modelo compacto*) y que consiste en la aplicación de polinomios matriciales en un formato particular. Estas matrices de orden cuatro poseen elementos de 8 bits y todas las operaciones se llevan a cabo en aritmética Z_{256} de manera que pueden ser manejadas por procesadores de 16 bits (tarjetas inteligentes, teléfonos celulares, llaves criptográficas USB, etc.). Tal como se ilustra en este trabajo, su simplicidad no afecta a la seguridad de empleo, siempre que se tomen los recaudos del caso. Este enfoque puede servir para avanzar en el desarrollo de nuevas soluciones *compactas* de AC.

Palabras clave: criptografía asimétrica, álgebra no conmutativa, anillos.

1 ANTECEDENTES

En 1976, Diffie y Hellman concibieron su histórico algoritmo de intercambio de claves y Pohlig-Hellman un algoritmo asimétrico de encriptación [1]. Esa brillante idea fue absorbida y perfeccionada por Rabin, Shamir y Adleman con el desarrollo de la criptografía de clave pública (PKC) [2], la que maduró luego en la plataforma tecnológica PKI (*public-key infrastructure*). El fundamento de la criptografía asimétrica (AC) consiste en el empleo de funciones trampa de una vía [1]. Hoy día, las primitivas más exitosas de la AC están basadas en problemas computacionalmente difíciles (de complejidad NP o al menos cuando no se conoce algoritmos de ataque de complejidad P) que con una pieza de información secreta se transforman en resolubles (complejidad P) [3]. En particular el énfasis ha sido puesto en el problema de la factorización de enteros positivos (IFP) y el problema del logaritmo discreto (DLP) [1] en *grupos conmutativos* de dimensiones suficientemente elevadas acorde con la tecnología de ataque estado-del-arte.

Resumiendo, los anillos más empleados son $\mathbb{Z}_n$ y su extensión $\mathbb{Z}_n[x]$. Así fueron desarrollados protocolos de amplia difusión como RSA [2], Rabin-Williams [4], el sistema LUC [5] o versiones RSA basadas en curvas elípticas KMOV [6]. Otro ejemplo adecuado es la familia ElGamal incluyendo su versión básica [7], criptosistemas de curvas elípticas (ECC) e hiperelípticas (HECC) [3], el intercambio de claves Diffie-Hellman [8], el estándar de firma digital DSS que emplea ElGamal [1] o el esquema McCurley [9], todos basados en la dificultad de resolver logaritmos discretos en un campo finito generado por un primo de tamaño suficiente y por supuesto incluidos en un *grupo conmutativo*. En particular para ECC, no existen siquiera algoritmos de tiempo subexponencial para poder atacarlo.

Tradicionalmente, los fundamentos de la AC y la PKC se hallaban mas en el ámbito de la teoría de números que en el espacio del álgebra abstracta [3], [10]. En computación cuántica [11] tanto IFP y DLP, como DLP sobre curvas elípticas (ECDLP) pueden ser eficazmente atacados gracias a los algoritmos de Shor [12], Kitaev [13] y Proos-Zalka [14]. A pesar que una computadora cuántica realmente eficaz a estos fines esté tal vez unas décadas más allá del horizonte actual, la mera noción de esa debilidad crea desconfianza en los protocolos vigentes [15]. Es por este motivo que en los últimos años se está enfatizando la búsqueda de nuevos algoritmos de naturaleza más algebraica que numérica [15]. Resumamos las principales líneas exploradas:

- En 1985, Wagner [16] propuso un protocolo PKC basado en el problema indecible de las palabras en *grupos* y *semigrupos*. Esta idea fue perfeccionada por otros investigadores [17].
- En 1999, Anshel [18] propuso un protocolo PKC basado en la dificultad de resolver ecuaciones en *grupos no conmutativos*. En su trabajo propusieron incluso el uso de la teoría de trenzas como alternativa viable.
- En 2000, Ko [19] presenta por primera vez un protocolo PKC fundamentado en el empleo de la teoría de trenzas. Este tema alcanzó una popularidad inmediata [18], [20]. Sin embargo, desde 2001 a 2003, el criptoanálisis exitoso frenó ese en-

tusiasmo [21]. Incluso algunos autores anunciaron la muerte prematura de la PKC basada en *grupos* de trenzas [21]. Seguramente no está dicha la última palabra sobre este tema [22].

- En 2001, Paeng [23] publica un esquema PKC basado en *grupos finitos no conmutativos*. El método emplea el problema DLP en el *automorfismo* definido por la operación de *conjugación*. Específicamente se trata de la dificultad de resolver la búsqueda del *elemento conjugador* (CSP). Este sistema ha sido ulteriormente perfeccionado como protocolo MOR [24].
- Simultáneamente Magliveras [10] presenta nuevos esquemas usando funciones trampa de una vía sobre *grupos finitos*. Así presenta dos sistemas PKC basados en la dificultad de factorizar productos en *grupos finitos*, denominados MST1 y MST2. En 2002, González Vasco [25] demuestra que generalizando lo necesario, los conceptos de factorización empleados, se obtiene una plataforma general que unifica criptosistemas disímiles como MST1, MST2, MOR, ElGamal y teoría de trenzas.
- En 2002, Grigoriev y Ponomarenko [26] desarrollaron criptosistemas *homomorficos* en *anillos conmutativos* de *matrices enteras* y los extendieron [27] a *semi-grupos finitos* basándose en la dificultad del problema de la *pertenencia* en *grupos* de *matrices* Z_n^* .
- En 2004, Eick y Kahrobaei [28] proponen generar sistemas PKC usando *grupos policíclicos*, una generalización natural de los grupos cíclicos pero de estructura más compleja. Esta línea puede llegar a ser muy promisoría en el futuro.
- En 2004 y 2005, Shpilrain y Ushakov [29], [30] sugieren el uso del *grupo de Thompson* para desarrollos PKC. Se trata de un *grupo no-abeliano infinito* con representación finita. Allí se plantea por primera vez la dificultad del problema de la *descomposición simétrica* (SDP) del que hablaremos más adelante en este trabajo y que es una generalización del problema CSP.
- En 2007, Cao, Dong y Wang [31] presentan un interesante esquema general de PKC basándose en la dificultad de resolver SDP en un *anillo no conmutativo* de *polinomios matriciales* (PSDP). Esto conduce directamente a protocolos de intercambio de claves Diffie-Hellman generalizados (GDHP) y a criptosistemas ElGamal generalizados.
- En 2008, Mahalanobis [32] propone con igual finalidad el empleo de presentaciones finitas de *grupos nilpotentes de clase 2*.
- En 2008, Thomas y Lal [33] presentaron por primera vez un criptosistema genérico de firma digital asegurada (UDS o *undeniable digital signature*) embebido en un protocolo de prueba de conocimiento cero (ZKP). Se basaron en la dificultad de los problemas CSP y SDP. En su desarrollo proponen el uso de *álgebra no conmutativa* pero no definen ninguna estructura en particular aunque sugieren como alternativas el empleo de *grupos de trenzas*, *grupos policíclicos* o el *grupo de Thompson*.
- En 2008, Anjaneyulu, Reddy y Reddy [34] aplicaron la teoría desarrollada por Cao [31] generando un protocolo UDS basándose en el problema DSDP.

Sintetizando, existe una fuerte tendencia al reemplazo de AC y PKC basado en *sistemas algebraicos numéricos y conmutativos* por otros de naturaleza *no conmutativa*. Podemos resumir que en cada caso se buscan funciones trampa de una vía que condu-

cen a generalizaciones del problema DLP. Este problema de *logaritmo discreto generalizado* (GDLP) en el marco de una *estructura algebraica no conmutativa* (NCAS) es especialmente atractivo por dos motivos: a) no existen algoritmos de tiempo sub-exponencial para atacarlos, léase Index Calculus [1] y b) aún (hoy día) son inmunes al ataque con computadoras cuánticas [11]. Sin embargo hay que poner mucha atención en la clase de *estructura y operación* que se elige, sirva como ejemplo la fugaz moda de la llamada “criptografía de trenzas” [35].

2 OBJETIVO DEL PRESENTE TRABAJO

Hay ciertos inconvenientes que subsisten en la criptografía NCAS, uno de los cuales es la necesidad de empleo de bibliotecas de precisión extendida y el uso de *hardware* que permita usarlas con fluidez [1]. Un desafío abierto es hallar un modelo NCAS que sea criptográficamente seguro pero capaz de emplear recursos computacionales escasos, lo que permitirá una eficaz operación en plataformas de baja potencia. Estos modelos que definiremos como **compactos**, emplearán procesadores de 16 bits, en contenedores inteligentes como *smartcards*, teléfonos celulares o llaves inteligentes USB.

Este trabajo, inspirado en el desarrollo de Cao [31], explora como alternativa el uso de un modelo compacto de NCAS empleando el *anillo no conmutativo* de matrices $M_4[\mathbb{Z}_{256}]$ que no requieren el empleo de bibliotecas numéricas de precisión extendida, independientemente de su extensión espacial (memoria ocupada = 16 bytes por matriz) dado que todas las operaciones numéricas se efectuarán dentro del anillo numérico $\mathbb{Z}_{256}$. Tal como lo propuso Cao en su trabajo, la dificultad de ataque al protocolo requiere operar con un módulo RSA ($n=p.q$, primos fuertes) y por ende no sólo requiere bibliotecas de precisión extendida sino que además queda expuesto a los ataques del problema IFP. Esto se corrige con el modelo que aquí se presenta. Como ejemplo, se aplicará la estructura no conmutativa en un protocolo Diffie-Hellman generalizado (GDHP), aunque queda claro por lo ya descrito en la bibliografía que no existe inconveniente en trasladarlo a un criptosistema ElGamal, a un esquema de firma digital [31] o tal vez a un protocolo ZKP [33]. El cardinal del espacio de matrices propuesto es $|M_4[\mathbb{Z}_{256}]| = 2^{128}$ pero como se verá más adelante su dimensión no incide en la seguridad del protocolo.

3 FUNDAMENTOS ALGEBRAICOS Y COMPUTACIONALES

3.1. Problemas complejos en grupos (y semigrupos) no conmutativos

En un *grupo no conmutativo* G , dos elementos (x, y) son *conjugados*, representado como $x \sim y$, si $y = z^{-1}x z$ para algún $z \in G$. Aquí a z o a z^{-1} se lo denomina

el elemento *conjugador*. Así se definen problemas que a su manera, representan instancias algebraicas del problema generalizado del *logaritmo discreto* (GDLP) .

(CSP) Problema de la búsqueda del conjugador: dados $(x, y) \in G \times G$, hallar $z \in G$ tal que $y = z^{-1} x z$.

(DP) Problema de la descomposición: dados $(x, y) \in G \times G$ y $S \subseteq G$, hallar $(z_1, z_2) \in S$ tal que $y = z_1 x z_2$.

(SDP) Problema de la descomposición simétrica: dados $(x, y) \in G \times G$, y $(m, n) \in \mathbb{Z}$, hallar $z \in G$ tal que $y = z^m x z^n$.

(GSDP) Problema de la descomposición simétrica generalizada: dados $(x, y) \in G \times G$ y $S \subseteq G$ y $(m, n) \in \mathbb{Z}$, hallar $z \in S$ tal que $y = z^m x z^n$.

Acorde a los conocimientos presentes, todos los problemas aquí aplicados sobre *grupos no conmutativos* generales son lo suficientemente difíciles como para ser empleados con seguridad criptográfica [31]. Están ordenados por grado de complejidad creciente. En particular GSDP es al menos tan complejo como SDP. Concluimos que no se conoce (y casi seguramente no exista) algoritmo probabilístico de tiempo P capaz de resolver GSDP si la escala dimensional es la adecuada. Dado que GSDP puede ser considerado una instancia del GDLP, podemos agregar:

(CDH) Problema computacional Diffie-Hellman en el grupo no conmutativo G (con respecto a algún subgrupo abeliano S): Computar $x^{z_1 z_2}$ ($= x^{z_2 z_1}$) para un dado (x, x^{z_1}, x^{z_2}) tal que $x \in G$, $(z_1, z_2) \in S$ y $S \subseteq G$.

Hoy día no hay ninguna idea acerca de cómo extraer z_1 (o z_2) de los datos provistos [34] y eso es alentador si se lo compara con el empleo tradicional de *grupos numéricos conmutativos*. Las mismas consideraciones son válidas si G fuese un *semigrupo no conmutativo*, basta con restringir las condiciones de los exponentes a valores enteros positivos $(m, n) \in \mathbb{Z}_{>0}$ o $(z_1, z_2) \in \mathbb{Z}_{>0}$.

3.2. Problemas complejos en anillos no conmutativos

Para ejemplificar los siguientes problemas computacionalmente complejos trabajaremos sobre *anillos no conmutativos de polinomios con coeficientes enteros positivos*. Sea R un anillo $(R, +, 0)$ y $(R, \cdot, 1)$, donde la suma forma un *grupo abeliano* y el producto forma un *semigrupo no abeliano*.

Definición 1. El producto escalar se define para $k \in \mathbb{Z}_{>0}$ y $r \in R$ y vale

$$(k)r \triangleq \langle r + \dots + r \rangle_{k \text{ veces}} \quad (1)$$

Cuando $k \in \mathbb{Z}_{<0}$ vale

$$(k)r \triangleq (-k)(-r) = \langle (-r) + \dots + (-r) \rangle_{k \text{ veces}} \quad (2)$$

Para $k = 0$ es natural que $(k)r = 0$. Similarmente, vale $r^k = \langle r \cdot \dots \cdot r \rangle_{k \text{ veces}}$.

Teorema 1. $(a)r^m.(b)r^n = (ab)r^{m+n} = (b)r^n.(a)r^m$, $\forall a, b, m, n \in \mathbb{Z}$ y $\forall r \in R$, lo que se demuestra elementalmente aplicando la *propiedad asociativa del anillo* y la *propiedad conmutativa de la suma*.

Lema 1. Nótese que en general $(a)r.(b)s \neq (b)s.(a)r$ si $r \neq s$ dada la *no conmutatividad del semigrupo producto*. Esta distinción es crucial para el modelo desarrollado.

Definición 2. Se define el anillo de polinomios de coeficientes enteros positivos para todo elemento $r \in R$

$$f(r) = \sum_{i=0}^n (a_i)r^i = (a_0).1 + (a_1)r + \dots + (a_n)r^n \quad \text{donde } \forall i \mid a_i \in \mathbb{Z}_{>0} \quad (3)$$

Aquí r puede considerarse como una simple variable (x) y el conjunto de los polinomios $f(r)$ puede considerarse como la extensión polinómica del anillo $\mathbb{Z}_{>0}[r]$ para todo polinomio con coeficientes enteros positivos $f(r)$ y todo $r \in R$.

Teorema 2. Dados dos polinomios cualesquiera de la extensión del anillo $f(r), h(r) \in \mathbb{Z}_{>0}[r]$, se demuestra

$$f(r).h(r) = h(r).f(r) \quad (4)$$

Es elemental la demostración usando el Teorema 1. y la *propiedad distributiva del producto* respecto de la *suma*.

Lema 2. Nótese que en general $f(r).h(s) \neq h(s).f(r)$ si $r \neq s$ dada la *no conmutatividad del semigrupo producto*. También esta distinción resulta importante para fundamentar al modelo desarrollado.

Definición 3. Dado un *anillo no conmutativo* $(R, +, \cdot)$ y cualquier elemento $a \in R$ elegido al azar, definimos un *subanillo* $P_a \subseteq R$ por

$$P_a \triangleq \{ f(a) : f(x) \in \mathbb{Z}_{>0}[x] \} \quad (5)$$

Es decir P_a abarca al conjunto de todos los polinomios de la extensión aplicados sobre la misma variable a .

Ahora podemos definir nuevas versiones de los problemas SDP y CDH definidos en el apartado 3.1.

(PSD) Problema de la descomposición simétrica sobre el anillo no conmutativo de polinomios con coeficientes enteros positivos R : dados $(a, x, y) \in R^3$, y $(m, n) \in \mathbb{Z}$, hallar $z \in P_a$ tal que $y = z^m x z^n$.

(PDH) Problema computacional Diffie-Hellman en el anillo no conmutativo de polinomios con coeficientes enteros positivos R : Computar $x^{z_1 z_2}$ ($= x^{z_2 z_1}$) para un dado (a, x, x^{z_1}, x^{z_2}) tal que $a \neq x$, $(a, x) \in R^2$ y $(z_1, z_2) \in P_a$.

Nótese que P_a es precisamente el *subanillo conmutativo* necesario para poder implementar el protocolo de intercambio de claves Diffie-Hellman que se presenta

en el siguiente punto. No importa que instancias polinómicas $f(a)$, $h(a)$ se elijan, su producto será independiente del orden. Pero también es cierto que esto deja de ser válido si el producto de esos mismos polinomios se aplica a distintos elementos $f(a)$, $h(b)$, ver el Teorema 2. y su Lema. Resulta evidente que si valen todos los antecedentes previamente expuestos, ambos problemas (PSD y PDH) son computacionalmente intratables, o sea no existe (o al menos no se conoce) un algoritmo probabilístico de tiempo P que pueda resolverlos con precisión no despreciable con respecto a la escala dimensional del problema [3] (lo que se consigue con cardinales $|R|$ y $|P_a|$ suficientemente grandes como para no ser resuelto con los recursos computacionales vigentes).

4 PROTOCOLO DE INTERCAMBIO DE CLAVES DIFFIE-HELLMAN USANDO ANILLOS NO CONMUTATIVOS

Se han expuesto los fundamentos necesarios y ahora se está en condiciones de presentar un protocolo (*ejemplo*) de intercambio de claves Diffie-Hellman empleando anillos no conmutativos de polinomios matriciales con coeficientes enteros positivos.

- (a) ALICE envía dos números enteros positivos aleatorios $(m, n) \in \mathbb{Z}_{16} > 0$ y dos elementos aleatorios $(a, b) \in R$ a BOB a través de un canal público e inseguro. Cada elemento aleatorio es una matriz $M_4[\mathbb{Z}_{256}]$ de orden 4 y valores en $\mathbb{Z}_{256}$.
- (b) ALICE elige un polinomio entero al azar no nulo $f(x) \in \mathbb{Z}_{16}[x]$ tal que $f(a) \neq 0$ y sus coeficientes y exponentes sean enteros positivos ($\in \mathbb{Z}_{16} > 0$) y lo define como su clave privada.
- (c) BOB elige un polinomio entero al azar no nulo $h(x) \in \mathbb{Z}_{16}[x]$ tal que $h(a) \neq 0$ y sus coeficientes y exponentes sean enteros positivos ($\in \mathbb{Z}_{16} > 0$) y lo define como su clave privada.
- (d) ALICE computa $r_A = f(a)^m \cdot b$. $f(a)^n$ y se lo envía por igual canal a BOB.
- (e) BOB computa $r_B = h(a)^m \cdot b$. $h(a)^n$ y se lo envía por igual canal a ALICE.
- (f) ALICE computa la clave de sesión compartida $K_A = f(a)^m \cdot r_B \cdot f(a)^n$
- (g) BOB computa la clave de sesión compartida $K_B = h(a)^m \cdot r_A \cdot h(a)^n$

Observar que $K_A = K_B$ dado que se verifica

$$K_A = f(a)^m \cdot h(a)^m \cdot b \cdot h(a)^n \cdot f(a)^n = h(a)^m \cdot f(a)^m \cdot b \cdot f(a)^n \cdot h(a)^n = K_B \quad (6)$$

En la práctica los pasos (a) (b) (d) pueden ser simultáneos y requieren un solo mensaje por parte de ALICE. Luego los pasos (c)(e) son simultáneos y BOB envía un único mensaje. Finalmente (f)(g) son autónomos. Hay que advertir dos detalles acerca del muestreo y enmascaramiento de los datos transmitidos:

- (1) La selección de los elementos (a,b) debe ser aleatoria e uniforme dentro de su espacio. Dado que se trata de matrices numéricas con 16 valores del intervalo $[0 - 255]$ no resultará difícil llevar adelante la selección. La misma condición debe regir para el resto de los parámetros del sistema. Algunas de estas matrices modulares resultan ser *nilpotentes*, por lo cual deben ser descartadas por ALICE porque generan claves de sesión débiles.
- (2) Una medida excepcional a considerar consiste en el enmascaramiento de los datos en tránsito. Si se pretendiese armar un (micro) entorno de espacio reducido con el objeto de incrementar la velocidad en una plataforma de recursos escasos (lo cual es en principio desaconsejable) será conveniente enmascarar los datos en tránsito para dificultar el ataque [21], [36].

5 CONSIDERACIONES ACERCA DE LA SEGURIDAD COMPUTACIONAL DEL PROTOCOLO PROPUESTO

Queda claro que apoyándose en la dificultad computacional de los problemas PSD y PDH y las claves privadas de tamaño *adecuado*, el protocolo será potencialmente inmune a los ataques.

Se debe resaltar que incrementar el orden de los parámetros (m,n) o de las matrices (a,b) no aporta seguridad al protocolo dado que son elementos públicos. La seguridad reside exclusivamente en el cardinal del espacio de las claves privadas, es decir en el grado y los coeficientes de los polinomios que elijan las partes. Si se considera que un *ataque de fuerza bruta* debe trabajar simultáneamente con ambas claves privadas, la complejidad de dicho ataque será proporcional al cardinal de pares de polinomios privados.

Grado de los polinomios y sus coeficientes. Considerando que hay $O(m^2)$ polinomios enteros $\mathbb{Z}_m[x]$ de grado $\leq m$ y coeficientes en $\mathbb{Z}_m$, al elegir $\mathbb{Z}_{16}[x]$ en el protocolo ejemplo existen $O(2^{16})$ pares de polinomios. Obviamente no se puede usar este ejemplo en aplicaciones prácticas, pero incrementando el espacio $\mathbb{Z}_m[x]$ a valores *razonables* para transacciones de vida útil *breve*, por ejemplo con polinomios $\mathbb{Z}_{256}[x]$, el número de pares de claves privadas se incrementa a $O(2^{32})$. En aplicaciones corrientes no sería aventurado pensar en operar con claves privadas del orden $\mathbb{Z}_{4096}[x]$ en cuyo caso la complejidad del ataque será $O(2^{48})$ para lograr un nivel de seguridad computacional aceptable.

A pesar de ello se prefirió presentar al protocolo con claves privadas $\mathbb{Z}_{16}[x]$ con el solo objeto de mostrarlo aquí como ejemplo.

6 UN EJEMPLO DEL PROTOCOLO

- (a) Todas las operaciones numéricas se efectúan en $\mathbb{Z}_{256}$. ALICE envía a BOB ($m=12$, $n=7$) y los elementos

$$a = \begin{pmatrix} 17 & 3 & 128 & 45 \\ 255 & 12 & 1 & 65 \\ 136 & 90 & 28 & 13 \\ 201 & 166 & 8 & 28 \end{pmatrix}; b = \begin{pmatrix} 244 & 25 & 67 & 117 \\ 9 & 25 & 0 & 59 \\ 11 & 2 & 195 & 219 \\ 104 & 38 & 121 & 44 \end{pmatrix}$$

- (b) ALICE elige su clave privada $f(x) = 11x^{14} + 6x^9 + 3x^7 + 13$

- (c) BOB elige su clave privada $h(x) = x^{15} + 15x^{12} + 3x^8 + 11x^5 + 2x + 7$

- (d) ALICE computa su *token* y se lo envía a BOB (I es la matriz identidad orden 4)

$$f(a) = 11 \begin{pmatrix} 17 & 3 & 128 & 45 \\ 255 & 12 & 1 & 65 \\ 136 & 90 & 28 & 13 \\ 201 & 166 & 8 & 28 \end{pmatrix}^{14} + 6 \begin{pmatrix} 17 & 3 & 128 & 45 \\ 255 & 12 & 1 & 65 \\ 136 & 90 & 28 & 13 \\ 201 & 166 & 8 & 28 \end{pmatrix}^9 + 3 \begin{pmatrix} 17 & 3 & 128 & 45 \\ 255 & 12 & 1 & 65 \\ 136 & 90 & 28 & 13 \\ 201 & 166 & 8 & 28 \end{pmatrix}^7 + 13I = \begin{pmatrix} 101 & 70 & 91 & 29 \\ 163 & 208 & 97 & 193 \\ 111 & 46 & 127 & 167 \\ 230 & 1 & 68 & 106 \end{pmatrix}$$

$$r_a = \begin{pmatrix} 101 & 70 & 91 & 29 \\ 163 & 208 & 97 & 193 \\ 111 & 46 & 127 & 167 \\ 230 & 1 & 68 & 106 \end{pmatrix}^{12} \begin{pmatrix} 244 & 25 & 67 & 117 \\ 9 & 25 & 0 & 59 \\ 11 & 2 & 195 & 219 \\ 104 & 38 & 121 & 44 \end{pmatrix} \begin{pmatrix} 101 & 70 & 91 & 29 \\ 163 & 208 & 97 & 193 \\ 111 & 46 & 127 & 167 \\ 230 & 1 & 68 & 106 \end{pmatrix}^7 = \begin{pmatrix} 214 & 107 & 138 & 118 \\ 4 & 243 & 148 & 80 \\ 12 & 233 & 60 & 176 \\ 73 & 202 & 107 & 91 \end{pmatrix}$$

- (e) BOB computa su *token* y se lo envía a ALICE

$$h(a) = \begin{pmatrix} 17 & 3 & 128 & 45 \\ 255 & 12 & 1 & 65 \\ 136 & 90 & 28 & 13 \\ 201 & 166 & 8 & 28 \end{pmatrix}^{15} + 15 \begin{pmatrix} 17 & 3 & 128 & 45 \\ 255 & 12 & 1 & 65 \\ 136 & 90 & 28 & 13 \\ 201 & 166 & 8 & 28 \end{pmatrix}^{12} + 3 \begin{pmatrix} 17 & 3 & 128 & 45 \\ 255 & 12 & 1 & 65 \\ 136 & 90 & 28 & 13 \\ 201 & 166 & 8 & 28 \end{pmatrix}^8 +$$

$$+ 11 \begin{pmatrix} 17 & 3 & 128 & 45 \\ 255 & 12 & 1 & 65 \\ 136 & 90 & 28 & 13 \\ 201 & 166 & 8 & 28 \end{pmatrix}^5 + 2 \begin{pmatrix} 17 & 3 & 128 & 45 \\ 255 & 12 & 1 & 65 \\ 136 & 90 & 28 & 13 \\ 201 & 166 & 8 & 28 \end{pmatrix} + 7I = \begin{pmatrix} 197 & 216 & 148 & 122 \\ 234 & 241 & 102 & 222 \\ 64 & 94 & 251 & 214 \\ 164 & 8 & 90 & 197 \end{pmatrix}$$

$$r_b = \begin{pmatrix} 197 & 216 & 148 & 122 \\ 234 & 241 & 102 & 222 \\ 64 & 94 & 251 & 214 \\ 164 & 8 & 90 & 197 \end{pmatrix}^{12} \begin{pmatrix} 244 & 25 & 67 & 117 \\ 9 & 25 & 0 & 59 \\ 11 & 2 & 195 & 219 \\ 104 & 38 & 121 & 44 \end{pmatrix} \begin{pmatrix} 197 & 216 & 148 & 122 \\ 234 & 241 & 102 & 222 \\ 64 & 94 & 251 & 214 \\ 164 & 8 & 90 & 197 \end{pmatrix}^7 = \begin{pmatrix} 242 & 43 & 53 & 189 \\ 199 & 73 & 72 & 195 \\ 51 & 96 & 51 & 87 \\ 80 & 140 & 255 & 10 \end{pmatrix}$$

(f) ALICE computa la clave de sesión compartida

$$K_A = \begin{pmatrix} 101 & 70 & 91 & 29 \\ 163 & 208 & 97 & 193 \\ 111 & 46 & 127 & 167 \\ 230 & 1 & 68 & 106 \end{pmatrix}^{12} \begin{pmatrix} 242 & 43 & 53 & 189 \\ 199 & 73 & 72 & 195 \\ 51 & 96 & 51 & 87 \\ 80 & 140 & 255 & 10 \end{pmatrix} \begin{pmatrix} 101 & 70 & 91 & 29 \\ 163 & 208 & 97 & 193 \\ 111 & 46 & 127 & 167 \\ 230 & 1 & 68 & 106 \end{pmatrix}^7 = \begin{pmatrix} 192 & 211 & 72 & 228 \\ 166 & 175 & 90 & 150 \\ 82 & 221 & 174 & 162 \\ 65 & 136 & 163 & 107 \end{pmatrix}$$

(g) BOB computa la clave de sesión compartida

$$K_B = \begin{pmatrix} 197 & 216 & 148 & 122 \\ 234 & 241 & 102 & 222 \\ 64 & 94 & 251 & 214 \\ 164 & 8 & 90 & 197 \end{pmatrix}^{12} \begin{pmatrix} 214 & 107 & 138 & 118 \\ 4 & 243 & 148 & 80 \\ 12 & 233 & 60 & 176 \\ 73 & 202 & 107 & 91 \end{pmatrix} \begin{pmatrix} 197 & 216 & 148 & 122 \\ 234 & 241 & 102 & 222 \\ 64 & 94 & 251 & 214 \\ 164 & 8 & 90 & 197 \end{pmatrix}^7 = \begin{pmatrix} 192 & 211 & 72 & 228 \\ 166 & 175 & 90 & 150 \\ 82 & 221 & 174 & 162 \\ 65 & 136 & 163 & 107 \end{pmatrix}$$

Resulta evidente la obtención de una clave de sesión común, sin emplear aritmética de precisión extendida. Falta definir cómo se traslada la matriz resultante a la clave numérica, aunque es trivial trasladar la matriz a un vector de 16 bytes (128 bits). Hay $16!$ formas de concatenar los bytes de la clave común, sin contar que la misma puede finalmente someterse a un *hashing* estándar. Estos detalles también pueden ser protocolizados en las aplicaciones concretas.

7 APLICACIÓN PRÁCTICA: SEGURIDAD Y TIEMPOS

Habiendo definido de la complejidad computacional del ataque de fuerza bruta, se puede asumir que existe una relación entre los cardinales de los espacios de búsqueda de claves privadas (los polinomios), los tiempos de cómputo y los potenciales rangos de aplicación del protocolo aquí presentado.

Resulta de sumo interés obtener estimaciones del tiempo que lleva generar las claves compartidas, lo que puede llegar a ser limitante para las aplicaciones prácticas. En la Fig. 1. se presenta un gráfico que ilustra la relación entre la dimensión de las claves privadas y una estimación pesimista del tiempo de procesamiento. En este caso se usó un código fuente no compilado y no optimizado en tiempos en un ambiente prototipo, sugiriendo que bajo condiciones adecuadas estos tiempos podrán ser significativamente reducidos.

En la Fig. 2. se presenta otro gráfico que ilustra la relación que existe entre el grado del polinomio usado y el cardinal del espacio de claves. Como es trivial de observar, el cardinal del espacio crece exponencialmente. Considerando los costos computacionales crecientes, se presentan posibles bandas de seguridad de compromiso tiempo-seguridad con las potenciales aplicaciones de uso. La comunicación celular encriptada se refiere a tráfico no almacenado.

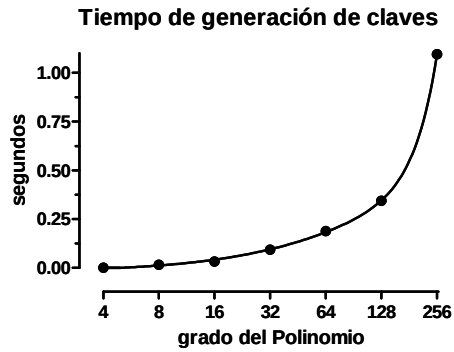


Fig. 1. Tiempos pesimistas de la generación de claves. En este gráfico se observa la curva de crecimiento temporal de la generación de claves según el protocolo presentado. Los valores corresponden a código no compilado y no optimizado en *Mathematica 7.0* corriendo en un equipo WinXP CPU Intel CoreDuo 1.6 GHz y 2 Gb RAM.

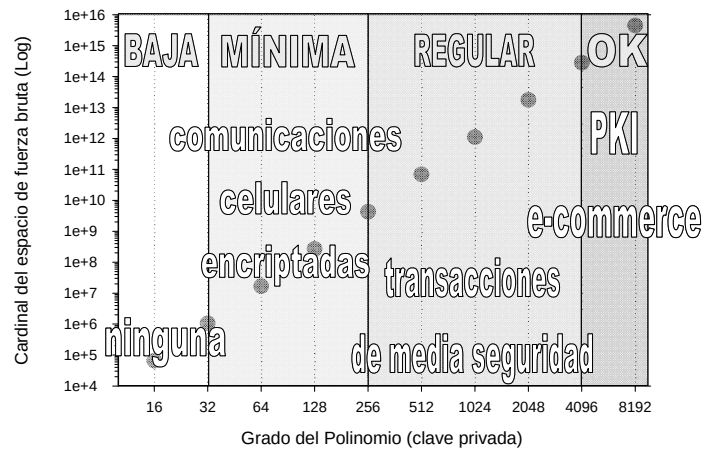


Fig. 2. Bandas potenciales de seguridad y aplicabilidad práctica. Aquí se observan hipotéticas bandas de seguridad y aplicación práctica, definidas por la relación entre el grado del polinomio usado como clave privada y el cardinal de la exploración sistemática. Si los cómputos del ataque procediesen a razón de 10^6 tanteos por segundo, la banda baja se explora en menos de 1 segundo, la banda mínima en menos de 3 horas, la banda regular en menos de tres años y la banda segura en el orden de 3 siglos.

8 CONCLUSIONES

En la última década se han desarrollado algoritmos y protocolos criptográficos basados en *estructuras algebraicas no conmutativas* (NCAS). Lo atractivo de estas soluciones es que no están basadas en la dificultad de la resolución de ciertos problemas que operan en *estructuras conmutativas* como la factorización de enteros (IFT) o el logaritmo discreto (DL) y para los cuales ya existen ataques de complejidad subexponencial en el tiempo. Por otra parte, la eventual aparición de computadoras cuánticas amenaza con destruir los criptosistemas tradicionales, lo que en estos sistemas NCAS no parece ser viable. Por ese motivo se han apelado a las más diversas estructuras como grupos de trenzas, grupos de Thompson, grupos policíclicos, grupos nilpotentes, grupos simétricos, anillos matriciales, etc. La lista seguramente no está agotada y se verán aparecer nuevos y seguramente promisorios enfoques. Lo interesante del caso es que una vez descubierto algún protocolo de criptografía asimétrica (AC) o de clave pública (PKC) casi automáticamente se resuelven temas vinculados a la confidencialidad, autenticación, pruebas de conocimiento cero e intercambio de claves.

Hasta ahora no hay antecedentes bibliográficos acerca de sistemas NCAS compactos. En este trabajo se presenta por primera vez un protocolo de esa clase diseñado específicamente para ser operado en plataformas computacionalmente pobres, es decir que no requiera bibliotecas de precisión extendida y que puedan resolverse cómodamente usando aritmética entera de 16 bits, todo ello sin resignar seguridad criptográfica. Obviamente en ambientes de mayor envergadura su operación puede llegar a competir favorablemente con otros protocolos de seguridad equivalente.

Sintetizando, entendemos que el modelo compacto aquí presentado puede brindar una alternativa útil tanto por su seguridad criptográfica como por su bajo costo computacional que lo hace apto para implementaciones muy reducidas.

8 REFERENCIAS

1. Menezes A.J., van Oorschot P.C., Vanstone S.A.: Handbook of Applied Cryptography. CRC Press (1997)
2. Rivest R.L. Shamir A., Adleman L.: A method for obtaining digital signatures and public key cryptosystems, Communications of the ACM., 21:1, 120-126 (1978)
3. Koblitz N.: Algebraic aspects of cryptography, Springer (1998)
4. Williams H.C.: A modification of the RSA public key encryption procedure, IEEE Transactions on Information Theory, 26:6, 726-729 (1980)
5. Smith P. Lennon M.: LUC - a new public key system, Proceedings of the IFIP- IX International conference on information security, North-Holland, 93, 103-117 (1993)
6. Koyama K., Maurer U., Okamoto T., Vanston S.: New public key schemes based on elliptic curves over the ring $\mathbb{Z}_n$, Crypto'91, (ed) J. Feigenbaum, Springer (1992)
7. ElGamal T.: A public key cryptosystem and a signature scheme based on discrete logarithms, IEEE Transactions on information theory, 31, 469-472 (1985)
8. Diffie W. Hellman M.E.: New directions in cryptography, IEEE Transactions on information theory, 22, 644-654 (1976)

9. McCurley K.: A key distribution system equivalent to factoring, *Journal of Cryptology*, 1, 95-100 (1988)
10. Magliveras S.S., Stinson D.R., van Trung T.: New approaches to designing public key cryptosystems using one-way functions and trapdoors in finite groups, Technical Report CORR, 2000-2049 (2000)
11. Nielsen M.A., Chuang I.L.: Quantum computation and quantum information. Cambridge University Press (2000)
12. Shor P.: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer, *SIAM J. Comput.*, 5, 1484-1509 (1997)
13. Kitaev A.: Quantum measurements and the abelian stabilizer problem, Preprint arXiv/quant-ph., 9511026 (1995)
14. Proos J., Zalka C.: Shor's discrete logarithm quantum algorithm for elliptic curves, *Quantum information and computation*, 3, 317-344 (2003)
15. Lee E.: Braid groups in cryptography, *IEICE Trans. Fundamentals*, E87-A:5, 986-992 (2004)
16. Wagner N.R., Magyarik M.R.: A public-key cryptosystem based on the word problem, *Crypto'84* (ed) Blakley G.R. Chaum D. Springer Verlag (1985)
17. Birget J., Magliveras S.S., Sramka M.: On public-key cryptosystems based on combinatorial group theory, *Cryptology ePrint archive report 2005/070* (2005)
18. Anshel I., Anshel M., Goldfeld D.: An algebraic method for public-key cryptography, *Math. Research Letters*, 6, 287-291 (1999)
19. Ko K.H., Lee S.J., Cheon J.H., Han J.W.: New public-key cryptosystem using braid groups, *Crypto 2000*. 166-183, Springer Verlag (2000)
20. Thomas T., Lal A.K.: Group signatures schemes using braid groups, Preprint arXiv/cs., 0602063 (2006)
21. Dehornoy P.: Braid based cryptography, *Contemporary mathematics*, 360, 5-33 (2004)
22. Bohli J.M., Glas B., Steinwandt R.: Towards provable secure group key agreement building on group theory, *Cryptology ePrint archive, report 2006/79* (2006)
23. Paeng S.H., Ha K.C., Kim J.H., Chee S., Park C.: New public-key cryptosystem using finite non-abelian groups, *Crypto 2001*, (ed) J. Kilian, Springer Verlag, 470-485 (2001)
24. Paeng S.H., Kwon D., Ha K. C., Kim J.H.: Improved public-key cryptosystem using finite non-abelian groups, *Cryptology ePrint archive, Report 2001/066* (2001)
25. González Vasco M. I., Martínez C., Steinwandt R.: Towards a uniform description of several group based cryptographic primitives, *Designs, Codes and Cryptography*, 33, 215-226 (2004)
26. Grigoriev D., Ponomarenko I.: On non-abelian homomorphic public-key cryptosystems, Preprint arXiv/cs, 0207079 (2002)
27. Grigoriev D., Ponomarenko I.: Homomorphic public-key cryptosystems over groups and rings, Preprint arXiv/cs, 0309010 (2003)
28. Eick B., Kahrobaei D.: Polycyclic groups: a new platform for cryptography, Preprint arXiv/math.gr, 0411077 (2004)
29. Shpilrain V., Ushakov A.: A new key exchange protocol based on the decomposition problem, Preprint arXiv/math.gr, Vol. 0512140 (2005)
30. Shpilrain V., Ushakov A.: Thompson's group and public-key cryptography, Preprint arXiv/math.gr, 0505487 (2005)
31. Cao Z., Xiaolei D., Wang L.: New public-key cryptosystems using polynomials over non-commutative rings, Preprint arXiv/cr, eprint.iacr.org/2007/009.pdf (2007)
32. Mahalanobis A.: The Diffie-Hellman key exchange protocol and non-abelian groups, Preprint arXiv/math.gr, 0602282v3 (2007)
33. Thomas T., Lal A.K.: A Zero-Knowledge undeniable signature scheme in non-abelian group setting, *International Journal of Network Security*, 6:3, 265-269 (2008)

34. Anjaneyulu G.S., Reddy P.V., Reddy U.M.: Secured digital signature scheme using polynomials over non-commutative division semirings, IJCSNS International Journal of computer science and network security, 8:8, 278-284 (2008)
35. Mahlborg K.: An overview of braid group cryptography, www.math.wisc.edu/~boston/mahlburg.pdf (2004)
36. Shpilrain V., Zapata G.: Combinatorial group theory and public-key cryptography, Preprint arXiv/math.gr, 0410068 (2004)

Watermarking in the Encrypted Domain

Jordi Herrera-Joancomartí[†], David Megías[‡]

[†] Departament d'Enginyeria de la Informació i les Comunicacions
Universitat Autònoma de Barcelona

[‡] Estudis d'Informàtica, Multimèdia i Telecomunicacions
Universitat Oberta de Catalunya
jherrera@deic.uab.cat, dmegias@uoc.edu

Abstract. In this paper we discuss the possibilities of insertion and detection of watermarks in the encrypted domain. We review the literature of computing in the encrypted domain and we discuss the application of such techniques in a watermarking scenario to obtain secure protocols for zero-knowledge detection and asymmetric fingerprinting schemes.

Keywords Watermarking, fingerprinting, copyright protection, encryption.

1 Introduction

Encryption functions are those that transform a message (or in general an object, an image or a file) into a piece of nonsense information $I_k = \mathcal{E}_k(I)$. Usually, the only goal of such encrypted information I_k is to allow to transfer or store original data so as to ensure that the original information cannot be derived from its encrypted version. However, what will be the result if the encrypted data is modified? In other words, can we operate in the encrypted domain? The answer depends on which operations we want to perform and if we want to transform such operations back in the clear domain. Obviously, some general transformations can be performed on encrypted data providing the transformation itself has a “meaning” in the encrypted domain. For instance, encrypted data can be compressed¹ in order to save space. Here the compression function, $\mathcal{C}$, has a complete meaning (to save space) in the encrypted domain. Such transformation can be performed on encrypted data provided the decompression function is applied before the decryption process, that is

$$\mathcal{D}_{k^{-1}}(\mathcal{C}^{-1}(\mathcal{C}(\mathcal{E}_k(I)))) = I$$

But if we want to compress the encrypted data in order to obtain a compressed data that can be decrypted to obtain a compressed version of the clear information, then the involved functions must hold the following equation:

$$\mathcal{C}^{-1}(\mathcal{D}_{k^{-1}}(\mathcal{C}(\mathcal{E}_k(I)))) = \mathcal{D}_{k^{-1}}(\mathcal{C}^{-1}(\mathcal{C}(\mathcal{E}_k(I))))$$

¹ Although cryptosystems are normally applied on already compressed data for masking the statistical properties of the clear text, such example may help to illustrate an hypothetical scenario.

and, of course, such condition does not hold for all possible functions. From now on, we will refer to *computing in the encrypted domain* only to those operations that “survive” the decryption process and can be inverted after that.

In general, computing in the encrypted domain could be performed depending on the cryptosystem used and also depending the operations performed in the encrypted domain.

On the other hand, the interest on computing in the encrypted domain is based on the fact that original data can be “meaningfully” modified by anyone without having access to the original information. As we will show in next sections, such property is very useful in some secure protocols.

2 Homomorphic Encryption

A first approach towards computing in the encrypted domain is to find out cryptosystems that allow some basic algebraic operations in the encrypted domain.

Privacy homomorphisms were first proposed by Rivest *et al* in [1]. Privacy homomorphisms were defined as encryption functions which permit encrypted data to be operated on without preliminary decryption of the operands. In this seminal paper, five simple examples of privacy homomorphisms were presented, the most relevant one was the RSA, which is multiplicative, *i.e.*, it allows multiplication on encrypted data:

$$\mathcal{E}(a \cdot b) = (a \cdot b)^e \bmod n = (a^e \bmod n) \cdot (b^e \bmod n) = \mathcal{E}(a) \cdot \mathcal{E}(b)$$

In [2] an additive and multiplicative privacy homomorphism was presented. The proposal allows addition, subtraction and multiplication to be carried out directly on encrypted data, but it does not allow computation of multiplicative inverses; in other words, it is a ring privacy homomorphism, but not a field privacy homomorphism.

Another well known privacy homomorphism is the one proposed by Paillier [3] used as a building block of different applications.

Security of privacy homomorphisms has also been studied. It is known that if a privacy homomorphism preserves order when encrypting, it is insecure against a ciphertext-only attack; if it is additive, it is insecure against a chosen-cleartext attack [4]. With the exception of RSA, all the examples given in [1] were subsequently broken in [5] using ciphertext-only or known-cleartext attacks. On the other hand, the privacy homomorphism presented in [2] is secure against a known-cleartext attack.

2.1 Homomorphic encryption applications

As we have pointed out, computing in the encrypted domain can be applied in some secure protocols. In particular, homomorphic encryption has been used successfully in different environments as a building block.

For instance, in [6] it is shown that some security properties of a secret sharing scheme can be proven using the homomorphic properties of the secret sharing function.

On the other side, as an extension of a secret sharing, multiparty computation schemes [7] and verifiable signature sharing [8] also use homomorphic encryption as a building block.

Regarding more specific applications, electronic voting systems also use homomorphic encryption. In this case, some properties of homomorphic encryption are used to verify the tally of the ballots without revealing what those ballots are, since homomorphic encryptions allow to have the sum of a group of encrypted values verified without revealing those encrypted values (see [9] for an exhaustive bibliography on voting schemes using homomorphic encryption).

On the other hand, electronic auctions also use homomorphic encryption [10]. For instance, in [11] a sealed-bid auction using homomorphic encryption is proposed and in [12] a proposed sealed-bid auction is based on circuit evaluation using homomorphic encryption.

Homomorphic encryption is also used in the context of databases [13–15] in order to allow database computation while maintaining a certain level of privacy of the records.

In mobile agent technology, homomorphic encryption is also applied. For instance, a partial solution for the problem of protecting agents against malicious hosts is proposed in [16] based on computing with encrypted functions, an extension of computing with encrypted data.

Finally, in [17], the authors take advantage of the homomorphic properties of the RSA to define a publicly verifiable watermarking scheme in which the verification process does not reveal the embedded mark.

3 Cryptographic computation on encrypted data

In addition to the basic algebraic operations described in the previous section, more complex computation can be performed on encrypted data. In particular, encryption and digital signature can be performed.

3.1 Encryption of encrypted data

Recalling again the idea of computing on encrypted domain described in Section 1 in the sense that operations in the encrypted domain can be inverted in the clear domain, cryptosystems which allow encryption in the encrypted domain fall into the category of commutative cryptosystems when both encryption functions are the same.

A commutative encryption function $\mathcal{E}_k(\cdot)$, with a secret key k , has the property that:

$$\mathcal{E}_{k_1}(\mathcal{E}_{k_2}(m)) = \mathcal{E}_{k_2}(\mathcal{E}_{k_1}(m))$$

for all keys k_1, k_2 .

Obviously, most symmetric encryption functions are not commutative while RSA is a commutative encryption scheme, since

$$\mathcal{E}_{k_1}(\mathcal{E}_{k_2}(m)) = \mathcal{E}_{k_1}(m^{k_2} \bmod n) = m^{k_2 \cdot k_1} \bmod n = m^{k_1 \cdot k_2} \bmod n = \mathcal{E}_{k_2}(\mathcal{E}_{k_1}(m))$$

Commutative encryption schemes are useful in different areas. For instance, a fair coin flipping protocol and a bit commitment scheme can be defined using commutative encryption [18].

3.2 Digital signature of encrypted data

The interest of digitally sign encrypted data has been discussed in the literature. In [19, 20] the authors argue that it is there is no sense to sign encrypted data since non-repudiation cannot be ensured because the signer does not know what he/she signs. However, at it is pointed out in [21], there are some applications where privacy is more important than non-repudiation.

Blind signatures: a particular approach. The concept of blind signature defined by Chaum in [22] can be seen as an special case of digital signature on encrypted data. In a blind signature scheme, the signer signs some data without knowing which data he has signed. In this case it is assumed that the encrypted data will be always signed and then decrypted before the signature validation, that is given an encryption function $\mathcal{E}$, a decryption function $\mathcal{D}$, a signature function $\mathcal{S}$ and a signature verification function $\mathcal{V}$, the only possible function composition will be

$$\mathcal{V}(\mathcal{D}(\mathcal{S}(\mathcal{E}(m)))) = m$$

In this especial case, $\mathcal{E}$ is a blinding function rather than an encryption function, since the composition with the “decryption” or unblinding counterpart holds $\mathcal{D}(\mathcal{E}(m)) \neq m$.

Assuming such principles, the blinding function can be defined as

$$\mathcal{E}(m) = m \cdot r^e \bmod n = c$$

where $k = r^e$ is the private key with $r \in_R \mathbb{Z}_n$ and e the public key of the signature function $\mathcal{S}$ defined by

$$\mathcal{S}(c) = c^d \bmod n = m^d \cdot r^{ed} \bmod n = m^d \cdot r \bmod n = c'$$

since $d \cdot e \equiv 1 \bmod \phi(n)$.

The unblinding function is defined by

$$\mathcal{D}(c') = c' \cdot r^{-1} = m^d = s$$

and the signature verification function is

$$\mathcal{V}(s) = s^e \bmod n = (m^d)^e \bmod n = m$$

With these definitions, it is possible to sign in the blinded domain and verify in the clear domain, that is

$$\mathcal{V}(\mathcal{D}(\mathcal{S}(\mathcal{E}(m)))) = m.$$

The first application of blind signatures was an anonymous payment system [23] proposed also by D. Chaum.

4 Watermarking in the encrypted domain

In the previous sections, we have presented different approaches to compute in the encrypted domain. In this section, we want to pay attention to the possibility of watermarking in the encrypted domain, that is to say, we want to study when or how encryption and decryption functions can commute with the watermarking operations (typically, mark and verify).

As we mention above, computing in the encrypted domain allows the generation of secure protocols to solve new problems. In the same way, watermarking over encrypted data may produce secure protocols to solve problems like publicly verifiable watermarking schemes or asymmetric fingerprinting schemes.

A watermarking scheme can be defined with two functions, the embedding or marking function $\mathcal{M}$ and the extraction or verification function $\mathcal{V}$. Roughly speaking, the marking function takes an image² I as an input together with the mark m to be embedded and produces the marked image $I^* = \mathcal{M}(I, m)$. On the other side, the verification function takes the marked (and possibly distorted) image $\hat{I}$ and gives the mark embedded in the image $\mathcal{V}(\hat{I}) = m$.

Again, we denote the encryption function $\mathcal{E}$ with a encrypting key k that produce a encrypted image $I_k = \mathcal{E}_k(I)$. The decryption function $\mathcal{D}$ produces the clear image $I = \mathcal{D}_{k^{-1}}(I_k)$ given the decrypting key k^{-1} and the encrypted image I_k .

Depending on where each function is applied different properties can be satisfied:

Prop. 1 The marking function $\mathcal{M}$ can be executed in an encrypted image I_k to embed a mark m .

Prop. 2 The verification function $\mathcal{V}$ can be able to reconstruct a mark in the encrypted domain when it has been embedded in the encrypted domain.

Prop. 3 The verification function $\mathcal{V}$ can be able to reconstruct a mark in the encrypted domain when it has been embedded in the clear domain.

Prop. 4 The decryption function does not affect the mark integrity in terms of mark reconstruction process.

Property 1 ensures that when the marking function is performed over an encrypted image, the result $I_k^* = \mathcal{M}(I_k, m)$ will have some sense. This point is

² For simplicity, we speak about images. However main ideas of the discussion proposed can be applied to digital objects different than images, such as audio files.

relevant since the marking schemes in the literature are based on image characteristics that may disappear when the image is encrypted.

The second property ensures that the mark and verification process can be performed entirely in the encrypted domain

$$\mathcal{V}(\mathcal{M}(\mathcal{E}_k(I), m)) = m.$$

The third property means that the encryption function does not affect the mark integrity in terms of mark reconstruction process and it holds if

$$\mathcal{D}_{k^{-1}}(\mathcal{V}(\mathcal{E}_k(\mathcal{M}(I, m)))) = m.$$

Notice that properties 2 and 3 are equivalent in case the marking function and the encryption function commute

$$\mathcal{M}(\mathcal{E}_k(I), m) = \mathcal{E}_k(\mathcal{M}(I, m)) = I_k^*.$$

The last property implies that

$$\mathcal{V}(\mathcal{D}(\mathcal{M}(\mathcal{E}_k(I), m))) = m.$$

At first glance, it could be difficult to determine encryption/decryption functions and watermarking algorithms that hold all the proposed properties. For instance, as it has been pointed out, Property 1 fails for the vast majority of marking schemes in the literature since to achieve imperceptibility they are based on image characteristics that disappear when the image is encrypted.

However, in some scenarios, an interesting secure protocol can be defined only with one of the proposed properties. A clear example is the verification protocol proposed in [17] that allows to demonstrate the presence of a watermarking in an image without revealing the mark, thus producing the first approach to a zero-knowledge watermarking verification protocol. In this case, the watermarking verification protocol proposed is based on any linear and additive watermarking algorithm in which the watermarking verification can be performed by mark correlation (for instance the spread spectrum technique proposed by Cox et al. [24]). The encryption algorithm used is the RSA. When using such watermarking techniques together with this encryption algorithm Property 3 holds due to the homomorphic properties of the RSA algorithm together with the multiplicative operations performed in the verification process. This example shows the possibilities of watermarking in the encrypted domain, even if only one of the properties holds.

If we assume functions $\mathcal{M}$, $\mathcal{V}$, $\mathcal{E}$, $\mathcal{D}$ hold the properties stated above we can define new protocols with some interesting properties as it is shown in the next subsection.

4.1 An asymmetric fingerprinting scheme

In contrast to watermarking (which allows ownership protection), fingerprinting is a technique which allows to track redistributors of electronic information. In

a fingerprinting scheme, each copy image is marked with a different mark that allows buyer identification. Usually, it is assumed that two or more dishonest buyers can only locate and delete marks by comparing their copies (marking assumption, [25]).

Classical fingerprinting schemes [26, 25] are symmetrical in the sense that both the seller and the buyer know the fingerprinted copy. Even if the seller succeeds in identifying a dishonest buyer, her previous knowledge of the fingerprinted copies prevents her from using them as a proof of redistribution in front of third parties. In [27], the concept of asymmetric fingerprinting was introduced, whereby only the buyer knows the fingerprinted copy. Different asymmetric fingerprinting proposals can be found in the literature [27–33].

In [29, 27] the authors use multiparty computation [34] to ensure that the seller cannot obtain knowledge about the marked copy. Since multiparty computation is computationally complex, some proposals are focused on avoiding such a technique. For instance, in [30] the cryptographic tool Committed Oblivious Transfer [35] is used instead of multiparty computation. Such scheme has been improved in [32] and [33].

Watermarking in the encrypted domain can offer new possibilities towards the construction of new asymmetric fingerprinting schemes. As an example of a possible application, consider the following scenario.

Suppose functions $\mathcal{M}$, $\mathcal{V}$, $\mathcal{E}$, $\mathcal{D}$ hold the four properties stated above. Suppose the watermarking scheme used is asymmetric in the sense that a different secret information is needed for embedding and verifying the mark [36]. Finally, suppose the function $\mathcal{E}$ is a commutative encryption function in the sense described in Section 3.

With these assumptions, we can define an asymmetric fingerprinting scheme based on a protocol between the seller, S , and the buyer, B that allows to embed a mark into an image in a way that:

- only B obtains the marked image,
- only S knows the original image.

Protocol 1 (The marking protocol)

1. S , encrypts the original image I using an encryption algorithm $\mathcal{E}$ and a secret key k_1 . $I_{k_1} = \mathcal{E}_{k_1}(I)$. Then S sends the encrypted image I_{k_1} and the mark m to be embedded to B .
2. B embeds the mark m into the encrypted image I_{k_1} using the marking function $\mathcal{M}$. The marked image is denoted by $I_{k_1}^* = \mathcal{M}(I_{k_1}, m)$. Then B encrypts $I_{k_1}^*$ using a key k_2 and sends the result $I_{k_1 k_2}^* = \mathcal{E}_{k_2}(I_{k_1}^*)$ to S .
3. S verifies that B has embedded the mark, that is, using the verification function $\mathcal{V}$, S extracts the mark m and checks that $\mathcal{V}(I_{k_1 k_2}^*) = m$.
4. S removes the first encryption with k_1^{-1} over the image $\mathcal{D}_{k_1^{-1}}(I_{k_1 k_2}^*) = I_{k_2}^*$ and sends the result to B .
5. B removes the second encryption with k_2^{-1} over the image $\mathcal{D}_{k_2^{-1}}(I_{k_2}^*) = I^*$ and obtains the marked image.

Notice that although B is the one that marks the image, he does not obtain the original image since he only obtains the encrypted version I_{k_1} and the marked one I^* . On the other hand, the scheme proposed is asymmetric (in the sense of fingerprinting) since S does not obtain the marked image I^* .

The correctness of the protocol is based on the assumptions stated before. Step 2 uses Property 1: the marking function can be executed in the encrypted domain. Correctness of Step 3 is based on Property 3: the encryption function does not affect the mark integrity in terms of mark reconstruction. Step 4 is based on the assumption that the encryption function is a commutative encryption function and also on Property 4: the decryption function does not affect the mark integrity in terms of mark reconstruction. Finally, Step 5 is also based on Property 4. On the other hand, the asymmetry assumed for the watermarking scheme allows B to embed the mark in a way that S should not be able to reproduce, although S can verify that the mark is embedded into the marked image. Otherwise, the knowledge of the original image and the mark would allow S to obtain the marked image.

Of course, this is a high level overview of the protocol and we have made many assumptions on the properties of functions $\mathcal{M}$, $\mathcal{V}$, $\mathcal{E}$, $\mathcal{D}$. However, this protocol illustrates how watermarking in the encrypted domain can be applied to obtain new protocols with new properties.

5 Conclusions and further research

In this paper we have discussed the possibilities that watermarking in the encrypted domain offers. We have reviewed the most relevant literature that deals with the problem of computing in the encrypted domain. We have given some examples of privacy homomorphisms which preserve some algebraic operations and other cryptosystems that allow cryptographic operations on encrypted data. We have introduced the concept of watermarking in the encrypted domain. We have defined the four main properties needed in the interaction of the main functions encryption/decryption and marking/verifying. We have shown that some properties have been already used to obtain an approach to a zero-knowledge watermarking verification protocol where the verifier does not obtain the embedded mark after the verification process. Finally we have outlined a high level asymmetric fingerprinting scheme based on watermarking in the encrypted domain.

5.1 Further research

Further research should be directed to obtain functions $\mathcal{M}$, $\mathcal{V}$, $\mathcal{E}$, $\mathcal{D}$ with the aforementioned properties. Different strategies can be used to reach this objective. First of all, a deep study of the existing watermarking schemes and cryptosystems must be carried out in order to determine if functions with those properties already exist. A first approach has shown that Property 3 holds for some specific schemes [17]. Secondly, new watermarking schemes and ad-hoc

cryptosystems can be proposed. Here we mean by ad-hoc cryptosystem, a encryption function that encrypts an object producing a same object type as encrypted output (for instance, a clear image that is encrypted giving an encrypted image). Such ad-hoc encryption for images has been studied in the literature [37–40] but the objective of such work has been sometimes undervalued since detractors argue that an image can be considered as a file and can be encrypted using any standard cryptosystem. However, if you need to compute in the encrypted domain you may need the encrypted object be an image and then ad-hoc cryptosystems may have a new development field.

On the other hand, combined watermarking/cryptosystem functions can be defined in order to obtain some of the needed properties. For instance, different functions can be defined providing a function composition holds

$$\mathcal{V}(\mathcal{D}(\mathcal{M}(\mathcal{E}_k(I), m))) = m$$

In this case, the “encryption” function may not be the inverse of the ”decryption” counterpart but this is not an issue if the application order of functions are determined a priori (see for instance, blind signatures -Section 3-).

Furthermore, if such functions can be defined, a detailed study about their security must be carried out before they can be used within a secure protocol.

Acknowledgements

This work was partially supported by the Spanish MCYT and the FEDER funds under grant TSI2007-65406-C03-03 E-AEGIS and CONSOLIDER CSD2007-00004 ”ARES”, funded by the Spanish Ministry of Science and Education.

References

1. Rivest, R.L., Adleman, L., Dertouzos, M.L.: On data banks and privacy homomorphisms. In Foundations of Secure Computation, New-York, Academic Press (1978) 169–179
2. Domingo-Ferrer, J.: A provably secure additive and multiplicative privacy homomorphism. In Information Security, Berlin, Springer-Verlag (2002) 471–483 Lecture Notes in Computer Science Volume 2433.
3. Paillier, P.: Public-key cryptosystems based on composite degree residuosity classes. In Cryptology - Eurocrypt ’99, Berlin, Springer-Verlag (1999) 223–238 Lecture Notes in Computer Science Volume 1592.
4. Ahituv, N., Lapid, Y., Neumann, S.: Processing encrypted data. Commun. ACM **30** (1987) 777–780
5. Brickell, E.F., Yacobi, Y.: On privacy homomorphisms. In Advances in Cryptology - EuroCrypt ’87, Berlin, Springer-Verlag (1987) 117–126 Lecture Notes in Computer Science Volume 304.
6. Cohen Benaloh, J.: Secret sharing homomorphisms: keeping shares of a secret secret. In Advances in Cryptology - Crypto ’86, Berlin, Springer-Verlag (1986) 251–260 Lecture Notes in Computer Science Volume 263.

7. Cramer, R., Damgård, I., Nielsen, J.B.: Multiparty computation from threshold homomorphic encryption. In: EUROCRYPT '01: Proceedings of the International Conference on the Theory and Application of Cryptographic Techniques, Springer-Verlag (2001) 280–299
8. Burmester, M.V.D.: Homomorphisms of secret sharing schemes: a tool for verifiable signature sharing. In: Advances in Cryptology - EuroCrypt '96, Berlin, Springer-Verlag (1996) 96–106 Lecture Notes in Computer Science Volume 1070.
9. Peng, K., Aditya, R., Boyd, C., Dawson, E., Lee, B.: Multiplicative homomorphic e-voting. In: Advances in Cryptology - Indocrypt '04, Berlin, Springer-Verlag (2004) 61–72 Lecture Notes in Computer Science Volume 3348.
10. Yokoo, M., Suzuki, K.: Secure multi-agent dynamic programming based on homomorphic encryption and its application to combinatorial auctions. In: Proceedings of the first international joint conference on Autonomous agents and multiagent systems, ACM Press (2002) 112–119
11. Cachin, C.: Efficient private bidding and auctions with an oblivious third party. In: Proceedings of 6th ACM Conference on Computer and Communications Security, ACM Press (1999) 120–127
12. Baudron, O., Stern, J.: Non-interactive private auctions. In: FC '01: Proceedings of the 5th International Conference on Financial Cryptography, Springer-Verlag (2002) 364–378
13. Chor, B., Kushilevitz, E., Goldreich, O., Sudan, M.: Private information retrieval. *J. ACM* **45** (1998) 965–981
14. Hacigumus, H., Iyer, B.R., Li, C., Mehrotra, S.: Executing SQL over encrypted data in the database service provider model. In: SIGMOD Conference. (2002)
15. Freedman, M., Nissim, K., Pinkas, B.: Efficient private matching and set intersection. In: Advances in Cryptology - Eurocrypt '04, Berlin, Springer-Verlag (2004) 1–19 Lecture Notes in Computer Science Volume 3027.
16. Sander, T., Tschudin, C.: Protecting mobile agent against malicious hosts. In: Mobile Agents and Security. LNCS 1419, Springer-Verlag (1998) 44–60
17. Gopalakrishnan, K., Memon, N., Vora, P.L.: Protocols for watermark verification. *IEEE MultiMedia* **8** (2001) 66–70
18. Schneier, B.: Applied Cryptography. John Wiley & Sons, Inc. (1996)
19. Abadi, M., Needham, R.: Prudent engineering practice for cryptographic protocols. *IEEE Trans. Softw. Eng.* **22** (1996) 6–15
20. Anderson, R., Needham, R.: Robustness principles for public key protocols. In: Advances in Cryptology - Crypto '95, Berlin, Springer-Verlag (1995) 236–247 Lecture Notes in Computer Science Volume 963.
21. Syverson, P.: Limitations on design principles for public key protocols. In: SP '96: Proceedings of the 1996 IEEE Symposium on Security and Privacy, IEEE Computer Society (1996) 62
22. Chaum, D.: Blind signatures for untraceable payments. In: Advances in Cryptology: Proceedings of Crypto '82, New York, USA, Plenum Publishing (1982) 199–204
23. Chaum, D.: Security without identification: Transaction systems to make big brother obsolete. *Communications of the ACM* **28** (1985) 1030–1044
24. Cox, I.J., Kilian, J., Leighton, T., Shamoon, T.: Secure spread spectrum watermarking for multimedia. *IEEE Transactions on Image Processing* **6** (1997) 1673–1687
25. Boneh, D., Shaw, J.: Collusion-secure fingerprinting for digital data. In: Advances in Cryptology-CRYPTO'95. LNCS 963, Springer-Verlag (1995) 452–465

26. Blakley, G.R., Meadows, C., Purdy, G.B.: Fingerprinting long forgiving messages. In: *Advances in Cryptology-CRYPTO'85*. LNCS 218, Springer-Verlag (1986) 180–189
27. Pfiztmann, B., Schunter, M.: Asymmetric fingerprinting. In: *Advances in Cryptology-EUROCRYPT'96*. LNCS 1070, Springer-Verlag (1996) 85–95
28. Pfiztmann, B., Waidner, M.: Asymmetric fingerprinting for larger collusions. In: *Proceedings of the 4th ACM Conference on Computer and Communications Security*. (1997) 151–160
29. Domingo-Ferrer, J.: Anonymous fingerprinting of electronic information with automatic identification of redistributors. *Electronics Letters* **34** (1998) 1303–1304
30. Domingo-Ferrer, J.: Anonymous fingerprinting based on committed oblivious transfer. In *Public key cryptography, PKC'99*. LNCS 1560, Springer-Verlag (1999) 43–52
31. Pfiztmann, B., Sadeghi, A.: Coin-based anonymous fingerprinting. In: *Advances in Cryptology - EUROCRYPT'99*, Berlin, Springer-Verlag (1999) 150–164 *Lecture Notes in Computer Science* Volume 1592.
32. Sadeghi, A.R.: How to break a semi-anonymous fingerprinting scheme. In *Information Hiding - IH'01*, Berlin, Springer-Verlag (2001) 384–394 *Lecture Notes in Computer Science* Volume 2137.
33. Choi, J.G., Park, J.H., Kwon, K.R.: Analysis of cot-based fingerprinting schemes: New approach to design practical and secure fingerprinting scheme. In *Information Hiding - IH'04*, Berlin, Springer-Verlag (2004) 253–265 *Lecture Notes in Computer Science* Volume 3200.
34. Chaum, D., Damgaard, I.B., van de Graaf, J.: Multiparty computations ensuring privacy of each party's input and correctness of the result. In: *Advances in Cryptology - CRYPTO'87*. LNCS 293, Springer-Verlag (1988) 87–119
35. Crépeau, C., van de Graaf, J., Tapp, A.: Committed oblivious transfer and private multi-party computation. In: *Advances in Cryptology-CRYPTO'95*. LNCS 963, Springer-Verlag (1995) 110–123
36. Furon, T., Duhamel, P.: An asymmetric public detection watermarking technique. In: *IH '99: Proceedings of the Third International Workshop on Information Hiding*, Springer-Verlag (2000) 88–100
37. Fridrich, J.: Image encryption based on chaotic maps. In: *Proceedings of the IEEE Conf. on Systems, Man, and Cybernetics*. Volume 2., IEEE Press (1997) 1105–1110
38. Scharinger, J.: Fast encryption of image data using chaotic Kolmogorov flows. *JEI* **7** (1998) 318–325
39. Chuang, T., Lin, J.: New approach to image encryption. *JEI* **7** (1998) 350–356
40. Chang, C.C., Hwang, M.S., Chen, T.S.: A new encryption algorithm for image cryptosystems. *J. Syst. Softw.* **58** (2001) 83–91

La Protección de Datos y el diseño de Tratamientos de datos personales. Especificaciones funcionales necesarias.

Resumen: La legislación española de protección de datos establece una serie de obligaciones relacionadas con el ejercicio de derechos por parte de los ciudadanos, cuya única forma de cumplimiento es posible conseguirla introduciendo en las especificaciones funcionales de los sistemas de información que realizan los tratamientos de datos personales, módulos específicos de ayuda. Estos requerimientos funcionales de los sistemas de tratamiento van más allá de la implantación de las medidas de seguridad que se han desarrollado reglamentariamente para los tratamientos de datos personales, de forma automatizada o no.

Palabras clave: Dato personal, Derechos reconocidos en materia de Protección de Datos, Calidad de los datos, Consentimiento y su revocación, Oposición al tratamiento, Acceso, Rectificación y Cancelación, Cesiones de datos.

1 Introducción

La Ley Orgánica 15/1999, de 13 de diciembre, que lleva a cabo la transposición al ordenamiento jurídico español de la Directiva Comunitaria 95/46/CEE establece como su objeto garantizar y proteger, en lo que concierne al tratamiento de los datos personales, las libertades públicas y los derechos fundamentales de las personas físicas, y especialmente de su honor e intimidad personal y familiar.

El Tribunal Constitucional Español, en su sentencia 292/2000, ha matizado que el derecho fundamental a la Protección de Datos, a diferencia del derecho a la intimidad, con quien comparte el objetivo de ofrecer una eficaz protección constitucional de la vida privada personal y familiar, atribuye a su titular un haz de facultades que consiste en su mayor parte en el poder jurídico de imponer a terceros la realización u omisión de determinados comportamientos, ... La peculiaridad de este derecho fundamental a la Protección de Datos respecto de aquel derecho fundamental tan afín como es el de la intimidad radica, pues, en su distinta función, lo que apareja, por consiguiente, que también su objeto y contenido difieran.

Esta matización es fundamental para realizar una correcta interpretación de los preceptos contenidos en la LOPD. Con frecuencia, las normas que regulan determinados tratamientos de datos de carácter personal, hacen coincidir la necesidad de cumplir los preceptos impuestos por la normativa de Protección de Datos con la necesaria garantía de confidencialidad en el tratamiento de los datos, como vía para preservar el derecho a la intimidad del sujeto titular de los mismos. Realmente, conforme a lo establecido por el Tribunal Constitucional en la citada sentencia 292/2000, el contenido del derecho fundamental a la Protección de Datos consiste en un poder de disposición y de control sobre los datos personales que faculta a la

persona para decidir cuáles de esos datos proporcionar a un tercero, sea el Estado o un particular, o cuáles puede este tercero recabar, y que también permite al individuo saber quién posee esos datos personales y para qué, pudiendo oponerse a su posesión o uso.

Visto lo anterior, para preservar el derecho fundamental a la Protección de Datos no basta con garantizar la confidencialidad en el tratamiento de los datos, es necesario permitir al titular de los mismos ejercer ese poder de control sobre los mismos, dando cumplimiento a los principios que establece la LOPD en su título II y facilitando el ejercicio de los derechos que se regulan en su título III.

El Reglamento de desarrollo de la LOPD, aprobado mediante Real Decreto 1720/2007, de 21 de diciembre, ha concretado, en muchos supuestos, determinadas obligaciones mediante el establecimiento de determinados procedimientos que deben ser seguidos por el responsable del Tratamiento, así como marcando perentorios plazos para el cumplimiento de determinadas obligaciones. Sólo con una previsión de las obligaciones impuestas e incorporando sistemas de ayuda a su cumplimiento en los diseños funcionales de los tratamientos de datos que se vayan a implantar, es posible dar un cumplimiento óptimo a las mismas y garantizar a los ciudadanos que se protege, efectivamente, su derecho de control sobre sus propios datos de carácter personal.

2 Calidad de los datos

El artículo 4 de la LOPD establece que los datos de carácter personal serán exactos y puestos al día de forma que respondan con veracidad a la situación actual del afectado. Si los datos de carácter personal registrados resultaran ser inexactos, en todo o en parte, o incompletos serán cancelados y sustituidos de oficio por los correspondientes datos rectificados o completados, sin perjuicio de las facultades que a los afectados reconoce el artículo 16 (Derecho de Rectificación y Cancelación).

El Reglamento de desarrollo de la LOPD establece que los datos deberán ser cancelados o rectificados en el plazo de diez días desde que se tuviese conocimiento de la inexactitud, salvo que la legislación aplicable al fichero establezca un procedimiento o un plazo específico para ello. Cuando los datos hubieran sido comunicados previamente, el responsable del fichero o tratamiento deberá notificar al cesionario, en el plazo de diez días, la rectificación o cancelación efectuada, siempre que el cesionario sea conocido. En el plazo de diez días desde la recepción de la notificación, el cesionario que mantuviera el tratamiento de los datos, deberá proceder a la rectificación o cancelación notificada.

Conforme a los anteriores mandatos establecidos por la legislación, los tratamientos de datos personales deberán incluir módulos de seguimiento de las

cesiones que se realicen, al objeto de poder generar las solicitudes de rectificación o cancelación que deban enviarse a los cesionarios que hayan recibido datos del responsable del tratamiento. Además, deberán preverse sistemas de control de los breves plazos establecidos por la normativa para realizar las cancelaciones o supresiones y las notificaciones correspondientes a los cesionarios receptores de los datos.

También el artículo 4 de la LOPD establece que los datos de carácter personal serán cancelados cuando hayan dejado de ser necesarios o pertinentes para la finalidad para la cual hubieran sido recabados o registrados.

El Reglamento de desarrollo de la LOPD indica, no obstante, que los datos podrán conservarse durante el tiempo en que pueda exigirse algún tipo de responsabilidad derivada de una relación jurídica o de la ejecución de un contrato o de la aplicación de medidas precontractuales solicitadas por el interesado.

Una vez concluido el tratamiento para el que se recogieron los datos, o el plazo de conservación obligatoria que puedan establecer otras normas legales de aplicación, los datos deberían ser objeto de cancelación. No obstante, los datos podrán conservarse una llegado el momento de destrucción de los mismos, en tanto en cuanto no concluyan los plazos de prescripción de exigencia de cualquier tipo de responsabilidad. Conforme a estos requerimientos, los sistemas deberán incorporar prestaciones que controlen las fechas en que se han concluido los tratamientos, al objeto de poder contabilizar los plazos necesarios de conservación. Una vez transcurridos estos, el sistema deberá proceder a la destrucción de los datos conservados.

Durante los plazos de conservación correspondientes a los plazos de prescripción de la posibilidad de exigir responsabilidades de algún tipo, los datos deberán estar diferenciados de aquellos que estén siendo sometidos a tratamiento en ese momento. Durante ese período de conservación a efectos de prueba, los datos no podrán ser objeto de ningún tipo de tratamiento para ninguna finalidad distinta de su mera conservación. Esta diferenciación obliga a prever en los sistemas de tratamiento los posibles estados que puedan corresponder a cada tipo de dato en cada momento.

También en el artículo 4 de la LOPD establece que los datos no serán conservados en forma que permita la identificación del interesado durante un período superior al necesario para los fines en base a los cuales hubieran sido recabados o registrados. Reglamentariamente se determinará el procedimiento por el que, por excepción, atendidos los valores históricos, estadísticos o científicos de acuerdo con la legislación específica, se decida el mantenimiento íntegro de determinados datos.

Con carácter general, los datos que deseen conservarse para estas finalidades deberán ser sometidos a un procedimiento previo de disociación irreversible. Los sistemas que vayan a realizar los tratamientos estadísticos o científicos deberán estar

adaptados al tratamiento de este tipo de datos anonimizados.

Concluye el artículo 4 de la LOPD indicando que, los datos de carácter personal serán almacenados de forma que permitan el ejercicio del derecho de acceso, salvo que sean legalmente cancelados.

Este mandato condiciona la forma en que se almacenará la información, especialmente cuando se utilicen tratamientos no automatizados o sólo parcialmente automatizados, dado que no podrá utilizarse como argumento para dificultar el ejercicio del derecho de acceso por el volumen de información que componga nuestro archivo o por el sistema de almacenamiento empleado. Como veremos al tratar de forma específica el ejercicio del derecho de acceso por parte del interesado, el responsable dispone del plazo máximo de un mes para dar respuesta a la petición formulada por el titular de los datos.

3 Consentimiento de afectado, revocación y oposición al tratamiento.

Conforme al artículo 6 de la LOPD, el tratamiento de los datos de carácter personal requerirá el consentimiento inequívoco del afectado, salvo que la ley disponga otra cosa.

Para que el consentimiento otorgado por el interesado tenga el carácter de inequívoco, será necesario haberle informado, con carácter previo a la recogida de sus datos: de la existencia de un fichero o tratamiento de datos de carácter personal, de la finalidad de la recogida de éstos y de los destinatarios de la información; del carácter obligatorio o facultativo de su respuesta a las preguntas que le sean planteadas; de las consecuencias de la obtención de los datos o de la negativa a suministrarlos; de la posibilidad de ejercitar los derechos de acceso, rectificación, cancelación y oposición; de la identidad y dirección del responsable del tratamiento o, en su caso, de su representante.

El Reglamento de desarrollo de la LOPD establece ahora que corresponde al responsable del tratamiento la prueba de la existencia del consentimiento del afectado por cualquier medio admitido en derecho. Además, el cumplimiento del deber de información previo a la obtención del consentimiento, deberá llevarse a cabo a través de un medio que permita acreditar su cumplimiento, debiendo conservarse mientras persista el tratamiento de los datos del afectado. En concreto, el Reglamento indica que para el almacenamiento de los soportes, el responsable del fichero o tratamiento podrá utilizar medios informáticos o telemáticos. En particular, podrá proceder al escaneado de la documentación en soporte papel, siempre y cuando se garantice que en dicha automatización no ha mediado alteración alguna de los soportes originales.

Esta obligación de conservación de los soportes originales en los que conste el consentimiento del interesado y el cumplimiento del deber de información, o sus

documentos electrónicos debidamente autenticados debe preverse en el diseño del sistema de almacenamiento de la información que de soporte al tratamiento de los datos personales. Estos documentos deberán ser conservados en tanto se conserven los datos personales, bien por ser objeto todavía de tratamiento, bien se conserven bloqueados a efectos de prueba hasta la conclusión de los plazos de prescripción correspondientes. Cuando se proceda a la destrucción definitiva de los datos de carácter personal deberá procederse igualmente a la destrucción de los documentos acreditativos del cumplimiento del deber de información y de la obtención del consentimiento del interesado.

Cuando la información se facilite por medios electrónicos o se obtenga el consentimiento del interesado de la misma forma, deberá conservarse igualmente algún medio de prueba admitido en derecho de tal situación. Si la información o el consentimiento se obtienen en el transcurso de una conversación telefónica o mediante el intercambio de mensajes de cualquier tipo, deberá conservarse copia de los mismos, debidamente autenticados.

El Artículo 6 de la LOPD también establece que cuando los datos de carácter personal no hayan sido recabados del interesado, éste deberá ser informado de forma expresa, precisa e inequívoca, por el responsable del fichero o su representante, dentro de los tres meses siguientes al momento del registro de los datos, salvo que ya hubiera sido informado con anterioridad.

Este mandato será de aplicación en los supuestos en que los datos se hayan obtenido por medio de una cesión o comunicación realizada por un tercero que, como se analiza en el siguiente apartado de este documento, está sometida a una serie de condicionantes para que se considere legítima. Este supuesto de obligación de informar al interesado estará sometido a los mismos criterios que se establecen para el supuesto en que los datos se hayan recabado del interesado directamente, debiendo realizarse por un medio que permita acreditar el cumplimiento del deber de informar, acreditando además su realización dentro del período de tres meses establecido por la norma, y la necesidad de conservación del medio por el que se haya realizado, a efectos de prueba, mientras dure el tratamiento de los datos o los plazos de conservación necesarios.

El consentimiento a que se refiere el artículo 6.1 de la LOPD podrá ser revocado cuando exista causa justificada para ello y no se le atribuyan efectos retroactivos.

El Reglamento de desarrollo de la LOPD establece que el afectado podrá revocar su consentimiento a través de un medio sencillo, gratuito y que no implique ingreso alguno para el responsable del fichero o tratamiento. En particular, se considerará ajustado al reglamento el procedimiento en que tal negativa pueda efectuarse, entre otros, mediante un envío prefranqueado al responsable del tratamiento o la llamada a un número telefónico gratuito o a los servicios de atención al público que el mismo hubiera establecido.

Continúa el Reglamento indicando que no se considerarán conformes a lo dispuesto en la LOPD, los supuestos en que el responsable establezca como medio para que el interesado pueda manifestar su negativa al tratamiento el envío de cartas certificadas o envíos semejantes, la utilización de servicios de telecomunicaciones que implique una tarificación adicional al afectado o cualesquiera otros medios que impliquen un coste adicional al interesado.

La posibilidad de ejercicio de la facultad de revocar el consentimiento otorgado por el interesado para el tratamiento de sus datos obliga a prever, en el diseño de los sistemas de tratamiento las funcionalidades necesarias y los procedimientos para que el titular de los datos pueda ejercer su derecho, teniendo en cuenta las premisas concretas establecidas por el Reglamento de desarrollo de la LOPD.

El responsable cesará en el tratamiento de los datos en el plazo máximo de diez días a contar desde el de la recepción de la revocación del consentimiento, sin perjuicio de su obligación de bloquear los datos conforme a lo dispuesto en la LOPD. Cuando el interesado hubiera solicitado del responsable del tratamiento la confirmación del cese en el tratamiento de sus datos, éste deberá responder expresamente a la solicitud. Si los datos hubieran sido cedidos previamente, el responsable del tratamiento, una vez revocado el consentimiento, deberá comunicarlo a los cesionarios, también en el plazo de diez días, para que estos, cesen en el tratamiento de los datos en el caso de que aún lo mantuvieran.

Ante una petición de revocación del consentimiento por parte del interesado para el tratamiento de sus datos, el responsable o encargado del tratamiento debe haber previsto en su Sistema de información la posibilidad de proceder a la supresión o al bloqueo de sus datos, conservándolos de forma diferenciada del resto de los datos en tanto no se cumplan los plazos de prescripción de las posibles acciones legales que éste pudiera entablar. Debe preverse, además, el procedimiento de notificación al interesado de que se ha procedido a excluir sus datos del tratamiento, cuando éste así lo haya solicitado y, por último, debe haberse dotado al sistema de un seguimiento de las cesiones de datos realizadas a terceros, al objeto de poder contactar con ellos para manifestarle la revocación del consentimiento realizada por el interesado, al objeto de que también ellos excluyan sus datos de los tratamientos que estén realizando.

En los casos en que no sea necesario el consentimiento del afectado para el tratamiento de los datos de carácter personal, y siempre que una ley no disponga lo contrario, éste podrá oponerse a su tratamiento cuando existan motivos fundados y legítimos relativos a una concreta situación personal. En tal supuesto, el responsable del fichero excluirá del tratamiento los datos relativos al afectado.

Este derecho tiene un efecto similar al analizado previamente de revocación del consentimiento, pero en aquellos supuestos en que éste no haya sido necesario para poder recabar y tratar los datos del interesado. Las consecuencias del ejercicio de este derecho son similares a las del derecho de revocación, excluir los datos del interesado del tratamiento que se esté realizando.

El Reglamento de desarrollo de la LOPD establece los supuestos en que es factible el ejercicio de este derecho: a) Cuando no sea necesario su consentimiento para el tratamiento, como consecuencia de la concurrencia de un motivo legítimo y fundado, referido a su concreta situación personal, que lo justifique, siempre que una ley no disponga lo contrario; b) Cuando se trate de ficheros que tengan por finalidad la realización de actividades de publicidad y prospección comercial, cualquiera que sea la empresa responsable de su creación; c) Cuando el tratamiento tenga por finalidad la adopción de una decisión referida al afectado y basada únicamente en un tratamiento automatizado de sus datos de carácter personal, en los términos previstos para este supuesto en la LOPD.

El procedimiento de ejercicio de éste derecho y los plazos previstos para su resolución son los mismos que se establecen para el ejercicio del derecho de revocación del consentimiento, con los mismos efectos y consecuencias. A efectos de planificación de funcionalidades del sistema de información con el que estemos tratando los datos del interesado, nuevamente, deberemos prever la posibilidad del ejercicio de este derecho, la cancelación o bloqueo de los datos del mismo y la remisión de la solicitud del interesado a todos aquellos terceros a los que se hayan cedido o comunicado sus datos.

En el supuesto de que el ciudadano ejerza su derecho de oposición y el responsable o encargado del tratamiento constata que no dispone de datos de su titularidad, sigue existiendo la obligación de comunicarle tal hecho al interesado.

Una última novedad en materia introducida por el Reglamento de desarrollo de la LOPD es la posibilidad de que los menores, mayores de catorce años, puedan otorgar su consentimiento para el tratamiento de sus datos, salvo en aquellos casos en que la Ley exija para su prestación la asistencia de los titulares de la patria potestad o tutela.

Cuando se vayan a tratar datos de menores, mayores de 14 años, basándose en el consentimiento prestado por ellos mismos, no se podrán recabar, a través del menor, información sobre los demás miembros del grupo familiar, o sobre las características del mismo, como los datos relativos a la actividad profesional de los progenitores, información económica, datos sociológicos o cualesquiera otros, sin el consentimiento de los titulares de tales datos. No obstante, podrán recabarse los datos de identidad y dirección del padre, madre o tutor con la única finalidad de recabar la autorización prevista anteriormente.

El diseño de tratamientos de datos de menores debe prever la edad de los sujetos sobre los que se vayan a tratar los datos, diferenciando, en función de la misma, el procedimiento y la persona que deberá prestar el consentimiento para el tratamiento de los mismos. En este sentido el Reglamento de desarrollo de la LOPD establece que corresponderá al responsable del fichero o tratamiento articular los procedimientos que garanticen que se ha comprobado de modo efectivo la edad del menor y la autenticidad del consentimiento prestado en su caso, por los padres, tutores o representantes legales del mismo.

Además, el cumplimiento de la obligación de informar con carácter previo a la recogida de los datos, en el caso de tratarse de datos de menores, deberá realizarse expresándose en un lenguaje que sea fácilmente comprensible por aquellos.

4 Comunicación de datos

Los datos de carácter personal objeto del tratamiento sólo podrán ser comunicados a un tercero para el cumplimiento de fines directamente relacionados con las funciones legítimas del cedente y del cesionario con el consentimiento previo del interesado.

Atendiendo a la regulación analizada para la obtención del consentimiento del interesado para el tratamiento de sus datos, entiendo que aplicable a los supuestos de necesidad de obtención del consentimiento para su cesión o comunicación a terceros, éste deberá obtenerse de forma que quede constancia de la prestación del mismo, debiendo conservarse la prueba que acredite su consecución en tanto dure el tratamiento de los datos o se conserven bloqueados.

Como hemos visto en los apartados anteriores de este documento y reiteraremos en los posteriores, el responsable o encargado del tratamiento deberá prever en sus sistemas la trazabilidad de las cesiones o comunicaciones de datos que realice, dada la necesidad, como se ha visto, de contactar con los cesionarios en los supuestos de que el interesado ejercite cualquier de los derechos que le reconoce la normativa vigente en materia de Protección de Datos.

5 Derecho de acceso

El interesado tendrá derecho a solicitar y obtener gratuitamente información de sus datos de carácter personal sometidos a tratamiento, el origen de dichos datos, así como las comunicaciones realizadas o que se prevén hacer de los mismos.

El alcance del ejercicio del derecho de acceso se establece en el artículo 27.2 del Reglamento de desarrollo de la LOPD: En virtud del derecho de acceso el afectado podrá obtener del responsable del tratamiento información relativa a datos concretos, a datos incluidos en un determinado fichero o a la totalidad de sus datos sometidos a tratamiento. Este ámbito de aplicación del ejercicio del derecho debe analizarse en relación con la obligación impuesta en la LOPD para el almacenamiento de los datos, en virtud de la cual, los datos de carácter personal serán almacenados de forma que permitan el ejercicio del derecho de acceso, salvo que sean legalmente cancelados.

Analizados los preceptos anteriores desde el punto de vista del interesado, aún cuando su solicitud de acceso se refiera a la totalidad de datos sometidos a tratamiento por el interesado, en uno o varios sistemas, deberá haberse previsto tal posibilidad, no pudiendo alegarse, en ningún caso, que no es posible facilitar el ejercicio del derecho

solicitado por la posible complejidad de los sistemas o tratamientos que se están realizando.

La información podrá obtenerse mediante la mera consulta de los datos por medio de su visualización, o la indicación de los datos que son objeto de tratamiento mediante escrito, copia, telecopia o fotocopia, certificada o no, en forma legible o inteligible, sin utilizar claves o códigos que requieran el uso de dispositivos mecánicos específicos. El Reglamento de desarrollo de la LOPD establece que será el interesado que realiza la solicitud del ejercicio de su derecho, el que podrá optar por el medio por el cual desea que se le facilite la información. No obstante, podrán restringirse los medios por los que se facilitara la información al interesado, en función de la configuración o implantación material del fichero o de la naturaleza del tratamiento, siempre que el que se ofrezca al afectado sea gratuito y asegure la comunicación escrita si éste así lo exige.

Como medida para evitar posibles abusos que pudieran producirse por los interesados en el ejercicio de este derecho, el Reglamento de desarrollo de la LOPD prevé que: a) Si el responsable ofreciera un determinado sistema para hacer efectivo el derecho de acceso y el afectado lo rechazase, aquél no responderá por los posibles riesgos que para la seguridad de la información pudieran derivarse de la elección; b) Si el responsable ofreciera un procedimiento para hacer efectivo el derecho de acceso y el afectado exigiese que el mismo se materializase a través de un procedimiento que implique un coste desproporcionado, surtiendo el mismo efecto y garantizando la misma seguridad el procedimiento ofrecido por el responsable, serán de cuenta del afectado los gastos derivados de su elección.

En relación a la información que se debe facilitar al interesado, además de los datos objeto de tratamiento, debe contener la procedencia de los mismos y las comunicaciones o cesiones de datos que se hayan realizado o se prevea realizar. Para dar cumplimiento a estas obligaciones, nuevamente se manifiesta la necesidad de implantar en los sistemas de tratamiento las suficientes funcionalidades que garanticen la trazabilidad respecto al origen de los datos objeto de tratamiento, así como de las cesiones o comunicaciones realizadas a terceros.

El ejercicio del derecho de acceso se realizará sobre los datos que están siendo objeto de tratamiento por parte del responsable o encargado del fichero. Dada la definición de tratamiento de datos contenida en las normas legales, se considera que existe tratamiento aún cuando nos limitemos exclusivamente a la conservación de los datos, por lo que, el ejercicio del derecho de acceso debe entenderse referido incluso a los datos que se conserven, bloqueados, para la posible atención de responsabilidades que pudieran plantearse.

El derecho de acceso sólo podrá ser ejercitado a intervalos no inferiores a doce meses, salvo que el interesado acredite un interés legítimo al efecto, en cuyo caso podrán ejercitarlo antes.

Esta limitación en el plazo de tiempo en el que puede reiterarse la solicitud del ejercicio del derecho de acceso por parte del interesado, obliga a éste a establecer sistemas de control y seguimiento de las solicitudes que pueda recibir, al objeto de poder denegar aquellas que incumplan el plazo establecido.

6 Derecho de rectificación y cancelación

Serán rectificadas o canceladas, en su caso, los datos de carácter personal cuyo tratamiento no se ajuste a lo dispuesto en la presente ley y, en particular, cuando tales datos resulten inexactos o incompletos. El responsable del tratamiento tendrá la obligación de hacer efectivo el derecho de rectificación o cancelación del interesado en el plazo de diez días.

La solicitud de rectificación deberá indicar a que datos se refiere y la corrección que haya de realizarse y deberá ir acompañada de la documentación justificativa de los solicitado. En la solicitud de cancelación, el interesado deberá indicar a qué datos se refiere, aportando al efecto la documentación que lo justifique, en su caso.

Los plazos establecidos reglamentariamente para resolver las solicitudes de ejercicio de derechos de rectificación o cancelación son de diez días. Si los datos rectificados o cancelados hubieran sido cedidos previamente, el responsable del fichero deberá comunicar la rectificación o cancelación efectuada al cesionario, en idéntico plazo, para que éste, también en el plazo de diez días contados desde la recepción de dicha comunicación, proceda, asimismo, a rectificar o cancelar los datos.

La cancelación dará lugar al bloqueo de los datos, conservándose únicamente a disposición de las Administraciones Públicas, Jueces y Tribunales, para la atención de las posibles responsabilidades nacidas del tratamiento, durante el plazo de prescripción de éstas. Cumplido dicho plazo deberá procederse a la supresión.

7 Conclusiones

Tal como se ha podido constatar a través del análisis de los preceptos legales, el cumplimiento de las obligaciones impuestas por los mismos sólo podrá realizarse cuando se haya previsto en el diseño de las funcionalidades de los sistemas de tratamiento las medidas que permitan y agilicen el cumplimiento de las mismas.

En concreto, y a modo de resumen, deberán preverse las siguientes funcionalidades:

- a) Diseño y establecimiento de vías o modos de cumplir el deber de información al interesado con carácter previo a la recogida de sus datos, o con posterioridad en el caso de que los datos no se obtengan directamente del interesado.

En el supuesto de que los datos correspondan a menores, mayores de 14 años, la información que se le facilite deberá estar adaptada a su capacidad de comprensión.

Deberá establecerse el procedimiento y la forma en que se conservarán los medios de prueba que acrediten el cumplimiento de la obligación de informar.

- b) Diseño y procedimentación de la forma de obtener el consentimiento del interesado cuando éste sea necesario para el tratamiento de sus datos o la cesión o comunicación de los mismos a terceros.

En el caso de que los titulares de los datos a recoger sean menores, mayores de 14 años, y los datos no afecten al grupo familiar, en la obtención del consentimiento deberá acreditarse que se ha verificado la edad del menor.

Cuando se haya obtenido a través del menor la identidad y dirección de sus padres o tutores, al objeto de obtener también su consentimiento, deberá verificarse la autenticidad del consentimiento de estos obtenido a través del menor.

Deberá establecerse el procedimiento y la forma en que se conservarán los medios de prueba que acrediten la obtención del consentimiento del interesado para la recogida y tratamiento de sus datos, o para la cesión o comunicación de los mismos a terceros.

- c) El almacenamiento de los datos deberá realizarse de forma que se facilite el ejercicio de los derechos por parte del interesado. Deberá preverse que el alcance del ejercicio de los derechos efectuados puede alcanzar a todos los tratamientos realizados por el responsable o encargado del fichero.
- d) Deberá preverse, en el diseño de los sistemas de almacenamiento de la información, la posibilidad de conservación de los datos debidamente bloqueados, excluidos de cualquier tratamiento realizado por el responsable, a los solos efectos de posibles reclamaciones de responsabilidades mientras no concluyan los plazos de prescripción de las correspondientes acciones.

Además, para el supuesto de datos bloqueados, deberá establecer algún sistema de control sobre la conclusión de los plazos de prescripción de las posibles acciones de exigencia de responsabilidades, para, a su cumplimiento, proceder al borrado definitivo de los datos de los interesados.

- e) Diseño de funcionalidades que garanticen y faciliten realizar la trazabilidad de los datos objeto de tratamiento, asegurando poder conocer en cualquier momento el origen de los datos así como las cesiones o comunicaciones de los mismos realizadas a terceros.

- f) Implantar funcionalidades que permitan controlar y realizar un seguimiento de las solicitudes de ejercicio de derechos recibidas, al objeto de controlar los plazos para su resolución (10 días en todos los supuestos excepto en el caso del ejercicio del derecho de acceso que es un mes), y de la posible denegación del ejercicio del derecho de acceso por no haber transcurrido un año desde el último ejercicio realizado.

Por último, y aunque no han sido objeto de éste trabajo, por supuesto los sistemas de tratamiento de datos de carácter personal, sean informatizados o no, deberán prever todas las medidas de seguridad establecidas en el Reglamento de desarrollo de la LOPD.

8 Referencias

- a. Ley Orgánica 15/1999, de 13 de diciembre, de Protección de Datos de Carácter Personal.
- b. Directiva 95/46/CE del Parlamento Europeo y del Consejo de 24 de octubre de 1995 relativa a la protección de las personas físicas en lo que respecta al tratamiento de datos personales y a la libre circulación de estos datos.
- c. Real Decreto 1720/2007, de 21 de diciembre, por el que se aprueba el Reglamento de desarrollo de la Ley Orgánica 15/1999.

Una marca de agua inteligente aplicada al dinero electrónico

Patricia Jaimes^{1*}, Gabriel Hermosillo², Roberto Gómez¹

¹ITESM-CEM/Dpto. Ciencias Computacionales, Edo. de Mexico, Mexico.

²University of Lille- LIFL- INRIA Project ADAM, Villeneuve d'Ascq, France

¹{jpatricia,rogomez}@itesm.mx, ²{gabriel.hermosillo}@inria.fr

Resumen El uso de las marcas de agua se ha incrementado, principalmente por la necesidad de proteger los derechos de autor, detener copias ilegales o medir la integridad de los datos de ciertos archivos. Es bien sabido que se puede insertar código ejecutable en imágenes, pero hasta ahora solamente se ha estudiado como una amenaza de seguridad para el usuario. Nosotros proponemos utilizar esta característica de manera segura para expandir las aplicaciones actuales de las marcas de agua, dándoles flexibilidad a través del código ejecutable. Presentamos el modelo de marca de agua inteligente para resolver problemas de incompatibilidad de funciones y demostramos cómo se puede aplicar este modelo a un escenario de dinero electrónico. En dicho escenario el beneficiario puede manejar diferentes implementaciones de dinero electrónico mediante una aplicación estándar. Como parte de este escenario, también proponemos una máquina expendedora de dinero electrónico para ofrecer una opción de pago a los usuarios que no tienen cuenta bancaria.

Palabras clave: marca de agua en imágenes, dinero electrónico, billete electrónico, código ejecutable oculto, esteganografía.

1. Introducción

La venta y el intercambio de archivos multimedia (software, imágenes, vídeo, audio) se ha incrementado en los últimos años. Una de las técnicas para brindar seguridad a dichos archivos son las marcas de agua. Estas se han utilizado principalmente para proteger los derechos de autor y detectar copias ilegales, lo cual implica el uso de marcas de agua de tipo robusto. El objetivo de esta categoría es no perder la información oculta ante ataques como distorsiones geométricas, cambios de compresión, resolución, entre algunos otros.

Lo opuesto a las marcas de agua robustas son las de tipo frágil, su objetivo es que la información oculta se destruya ante cualquier ataque, también se usan

* El trabajo de este autor fue realizado en el Institut National de Recherche en Informatique et en Automatique (INRIA), Lille, Nord Europe, Francia.

para medir la integridad de los datos, detectar cambios y autenticar imágenes [1]. Una de sus principales aplicaciones es en la medicina, por ejemplo se puede ocultar la información del paciente evitando distorsionar la región de interés de la imagen (electrocardiograma, rayos-x, etc). Nuestra propuesta esta basada en este tipo de marca de agua, de esta forma el dinero electrónico se vuelve inválido ante cualquier modificación.

Las marcas de agua son una rama de la esteganografía, su diferencia reside en el uso que se les asigne. La esteganografía esconde información independientemente del objeto que se utilice para esconder los datos y debe ser estadísticamente indetectable, en cambio en las marcas de agua, es de importancia el objeto en donde se insertarán los datos a ocultar y normalmente es de conocimiento público que el objeto contiene una marca. En nuestro caso al aplicar una marca de agua al dinero electrónico es de importancia la imagen que se usa para esconder los datos, ya que para el usuario final es la representación de dinero físico y al mismo tiempo facilita la portabilidad de los datos. Lo anterior puede ser de gran utilidad para incentivar la adopción del dinero electrónico en los usuarios finales.

Existen diversas clasificaciones de marcas de agua, dependiendo del tipo de archivos multimedia (vídeo, audio, imágenes), la técnica de inserción (sustitución, adición, transformación del dominio) [2], la percepción y pérdida de información (calidad perceptual, métricas basadas en píxeles) [3], la necesidad de la cubierta original para extraer el contenido (ciego, semi ciego, no ciego) [1], el objeto portador (tarjeta inteligente, software), el nivel de anonimato en caso de usar una huella digital (simétrica, pública, anónima) [4], la cantidad de información insertada [3], la integridad contra ataques maliciosos (frágil, semi frágil, robusta) [5], y el propósito de la aplicación (control de copias, propiedad intelectual, monitoreo de transmisiones, imágenes médicas) [2].

Nuestra propuesta es una marca de agua inteligente, la cual contiene una imagen con código ejecutable y mecanismos de seguridad que mantienen la integridad y autenticidad de la imagen. Esta marca puede ser utilizada en problemas de incompatibilidad de funciones, como es el caso de los modelos de dinero electrónico o las diferentes reglas de negocio en los boletos de avión. Ejemplificamos el uso de nuestro modelo aplicándolo a un escenario de dinero electrónico, dentro del cual proponemos una nueva opción de pago que no necesita de cuenta bancaria por parte del usuario.

El dinero electrónico, también conocido como *e-cash* o *e-money*, es solamente uno de los muchos métodos de pago en el mercado y su principal ventaja es el nivel de anonimato del usuario. Uno de los problemas con los modelos de dinero electrónico es la incompatibilidad entre ellos, ya que su técnica para generar y cobrar el dinero es diferente. Podemos encontrar un ejemplo de esto en Japón, donde existen modelos incompatibles de dinero electrónico [6,7,8,9], por lo tanto los comerciantes necesitan tener diferentes sistemas para poder recibir los pagos de cada modelo y los clientes deben poner atención en comprar solamente en tiendas donde su modelo de dinero electrónico es aceptado [10]. Con la marca de agua inteligente, los comerciantes necesitarán tener una única aplicación están-

dar que manejará cualquier tipo de dinero electrónico y los emisores de dinero podrán implementar cualquier modelo.

Otro uso de la marca de agua inteligente puede ser en los boletos de avión, en donde se puede tener un mostrador compartido entre todas las aerolíneas en el cual se reciban los boletos electrónicos, los cuales contienen la marca de agua inteligente con la información relacionada al vuelo. La aplicación estándar se encargará de validar, extraer y ejecutar el código que contiene la lógica para interactuar con el usuario y realizar la comunicación necesaria con la línea aérea. El modelo de marca de agua inteligente simplifica la interacción entre diferentes entidades y expande las aplicaciones actuales de las marcas de agua, dándoles flexibilidad a través del comportamiento configurable en el código ejecutable.

El resto de este artículo está organizado de la siguiente manera: La sección 2 presenta la motivación y algunos trabajos relacionados. La sección 3 muestra nuestro modelo de marca de agua inteligente. La sección 4 muestra la aplicación del modelo en un escenario de dinero electrónico y nuestra propuesta de una máquina expendedora de dinero electrónico. Finalmente, la sección 5 concluye y discute el trabajo futuro.

2. Motivación y trabajos relacionados

El sistema de dinero electrónico ideal, descrito por Okamoto [11], debe ser anónimo, independiente de cualquier condición física, capaz de prevenir copias no autorizadas, divisible, transferible a otros usuarios y no debe requerir en todo momento de una conexión. El dinero electrónico intenta ser el equivalente al dinero físico. En los últimos años se han propuesto diversos modelos de dinero electrónico, sin embargo solamente algunos se han comercializado, esto se debe a que el usuario final no adopta este tipo de pago o en ocasiones no satisface las leyes del país. En algunos otros casos, el modelo suele ser incompatible con las aplicaciones de pago existentes o simplemente no ha sido implementado. Algunos ejemplos de dinero electrónico que se utiliza actualmente son: Chipknip (Holanda) [12], Proton (Bélgica) [13], Geldkarte (Alemania) [14], Moneo (Francia) [15], Felica (Japón) [6] y Suica (Japón) [7].

Un proyecto de dinero electrónico que utiliza código ejecutable y un certificado es X-Cash [16]. Tanto el código como el certificado son enviados a través de la red, con el fin de encontrar en algún lugar la mejor opción para el objeto y el precio que el usuario busca. Sus autores lo llegan a comparar con un tipo de virus amigable, el cual está basado en un paradigma de agente móvil. El cliente paga al vendedor con un X-Cash firmado que contiene los elementos para comprar y la cantidad de dinero a ser pagado. El vendedor puede entonces intercambiar el X-Cash por dinero físico en el banco del cliente. El banco ejecuta el programa de X-Cash, utiliza los elementos como entradas y determina la cantidad de dinero que se le debe al vendedor. En este modelo todos los elementos se tratan de manera separada y no están insertados en ningún archivo.

Una de las características de nuestra propuesta es reunir toda la información en un solo archivo, lo cual puede otorgar al dinero electrónico mayor portabilidad.

Lo anterior ataca el problema de adopción por parte del usuario, ya que ayudaría a crear una fácil abstracción del dinero físico.

Algunos trabajos de investigación han encontrado una amenaza de seguridad cuando se esconde código ejecutable dentro de un archivo [17,18,19]. Esta vulnerabilidad puede ser explotada si previamente se engaña al usuario para que instale un software que se encargue de extraer el código ejecutable en la computadora del usuario. En algunos casos, el código ejecutable puede insertarse a sí mismo en nuevos archivos sin que el usuario se de cuenta. Algunos ejemplos de este tipo de virus son Perrun y Trojan Frog [20,21], los cuales fueron creados para demostrar que un antivirus no puede detectar el código ejecutable insertado en las imágenes. Otra manera de ejecutar código insertado en una imagen es explotando una antigua vulnerabilidad de desbordamiento de búfer en el procesamiento de imágenes JPEG dentro de la interfaz de dispositivo gráfico (GDI por sus siglas en inglés) de Microsoft[22]. En este caso no se necesita instalar previamente un software que extraiga el código, ya que un atacante puede utilizar esta vulnerabilidad si guarda correctamente código ejecutable y algunas direcciones de memoria en una imagen JPEG, por lo tanto cuando un usuario abra la imagen, el GDI ayudará a ejecutar el código insertado.

Nuestra propuesta es una marca de agua inteligente que contiene código ejecutable insertado en una imagen, además de una firma digital e información sobre el objeto (en el contexto de dinero electrónico la información son los datos, como nombre del banco, número de identificación, etc). Este tipo de marca de agua se puede aplicar sobre diferentes contextos, por ejemplo, billetes de lotería, billetes de avión, o sobre distintos modelos de dinero electrónico. En este último caso el emisor de dinero puede implementar el código insertado que contendrá la lógica del modelo. El usuario final únicamente tendrá una imagen, la cual puede ser fácilmente relacionada con el dinero físico y nunca tendría que manejar los datos en crudo del dinero electrónico, ni las aplicaciones para interpretarlos. Además, las entidades emisoras de dinero electrónico podrían implementar comportamientos especiales en algunos billetes, como premios, encuestas, estadísticas, etc.

Dentro del escenario de aplicación de nuestro modelo, proponemos un nuevo enfoque para el dinero electrónico basado en las necesidades de los países en desarrollo, donde en los últimos años los pagos a través de Internet sin tener una cuenta bancaria han creado un reto. En un estudio reciente, las Naciones Unidas reportaron que la mayoría de las transferencias electrónicas de dinero en los países en desarrollo se realizan a través de giro postal o telefonía móvil, dado que mucha gente no tiene una cuenta bancaria [23]. Esto significa que los pagos son prácticamente anónimos, pues en los países en desarrollo los teléfonos móviles no están ligados a una persona y los giros postales pueden tener cualquier dirección. Un estudio mexicano realizado en el 2008 muestra que el comercio electrónico ha crecido 78 % en el último año y podría ser mejorado si la gente tuviera más opciones de pago [24]. El mismo estudio mostró que el 50 % de la gente que tiene una tarjeta de crédito nunca ha comprado en Internet, debido a que se sienten inseguros al dar su información bancaria. Al utilizar nuestro modelo de dinero electrónico, los usuarios serán capaces de comprar en Internet sin tener

que exponer su información personal y financiera e incluso sin necesidad de tener una cuenta bancaria.

3. El modelo de marca de agua inteligente

Nuestra propuesta de una marca de agua inteligente puede ser utilizada para resolver problemas de incompatibilidad de funciones. Como se muestra en la Fig. 1, en este tipo de problemas se tienen varios distribuidores y cada uno puede crear distintos tipos de objetos. Estos objetos son conjuntos de datos que toman valor y sentido cuando son procesados por el software correcto, dichos objetos son enviados al usuario, quien necesita una aplicación diferente para utilizar cada objeto.

El distribuidor necesita proveer al usuario de la aplicación correspondiente o renovar la actual para cada versión y tipo de objeto. Lo anterior representa un problema para el usuario, pues necesita manejar un número creciente de aplicaciones para poder aceptar los diferentes objetos.

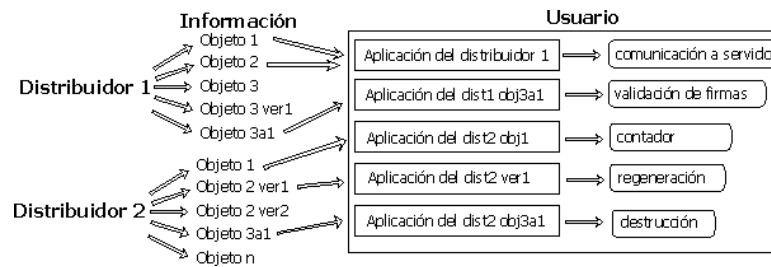


Figura 1. Problemas de incompatibilidad de funciones

Es cierto que si todos los datos que actualmente se intercambian se enviaran junto con la aplicación que los procesa se crearían archivos de gran cantidad de bytes, inadmisibles para el usuario y sin sentido ya que se podrían reutilizar. Sin embargo existen situaciones en donde las aplicaciones son pequeñas y los diferentes objetos se pueden agrupar bajo un mismo tema, los cuales tienen una gran variedad de versiones, en donde se añaden y disminuyen constantemente los objetos. También puede ser que los usuarios cambien con frecuencia y que muchos solo utilicen el software sola una vez. Lo anterior además de consumir tiempo puede ser inseguro, ya que los usuarios no verifican que el software que instalan sea íntegro y que provenga de una entidad válida. Por lo tanto en este tipo de casos conviene que los datos se encuentren acompañados de su aplicación.

Un ejemplo se puede observar en el dinero electrónico ya que su objetivo final es que sea transferible entre usuarios finales tal como lo es el dinero físico. Debido a que existen muchos modelos de dinero electrónico es probable que un usuario casual no vuelva a cobrar un billete del mismo modelo y versión dentro

de mucho tiempo. Para el caso de comercios pueden tener previamente instalado el software necesario para manejar cada modelo de billete electrónico y de esta forma evitar extraer el código ejecutable de cada billete, además con la ventaja de poder recibir nuevos modelos o cambios de versiones hasta que se realice la instalación del nuevo software fijo.

Nuestra propuesta de marca de agua inteligente puede ayudar a integrar en un solo archivo un conjunto de datos con su aplicación de manera portable y segura para el usuario. Primero explicaremos la creación de la marca de agua inteligente y luego detallaremos la aplicación estándar encargada de manejar cualquier tipo de objeto basado en dicha marca.

La meta común de cada distribuidor D es crear objetos O , los cuales contienen distinta información *info* (i.e. un número único, el nombre del distribuidor, etc.) y una o varias firmas digitales. En nuestro modelo, en lugar de obtener O , obtendremos una marca de agua inteligente $I_{marcaInte}$, como se puede observar en la Fig. 2. Cada D desarrolla su propio código ejecutable C , el cual contiene la lógica de la aplicación para manejar los datos de *info* y puede realizar diversas tareas, como comunicarse con D , generar un nuevo objeto, destruir un objeto, interactuar con un usuario U , manejar una marca de agua o una huella digital adicional, etc.

D utiliza una función de inserción $E()$ la cual introduce C dentro de la imagen original I_{orig} , y como resultado obtenemos una imagen que contiene el código ejecutable a la que le llamaremos I_{marca} . Lo anterior se expresa como $I_{marca} = E(I_{orig}, C)$. La función $E()$ dependerá del tamaño del código, del objetivo de la aplicación y del desempeño deseado. D no necesita crear una I_{marca} por cada objeto, porque puede reutilizar una existente, como se explicará más adelante. Cada D tiene su propio certificado K , creado por una Autoridad Certificadora (AC) utilizando sus llaves asimétricas (sk_{dist}, pk_{dist}).

D generará una firma digital F_{dw} utilizando su llave privada sk_{dist} , los datos *info*, la I_{marca} y la función Sig en la cual se implementan los algoritmos. Dicha firma digital se obtiene a partir de $F_{dw} = Sig_{sk_{dist}}(info || I_{marca})$. Esto le confirmará a todo usuario U la integridad de C y de *info*, mientras que también le asegurará que la I_{marca} fue creada por un D válido, utilizando el K de D para comprobar la firma. Finalmente, la marca de agua inteligente $I_{marcaInte}$ se obtiene utilizando una función de unión $J()$, la cual se expresa $I_{marcaInte} = J(F_{dw} || info || I_{marca})$.

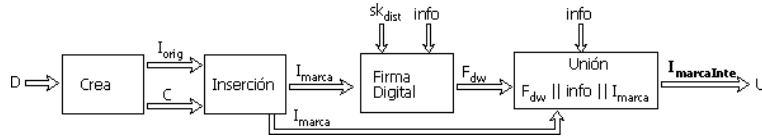


Figura 2. Generación de la marca de agua inteligente

Los distribuidores necesitan definir las implementaciones de las funciones $E()$ y $J()$, así como el tamaño y el orden de la información común que se va a utilizar en el campo *info*, en el cual también se puede agregar información específica de cada distribuidor que no requiere ser acordada por el conjunto de distribuidores. Los datos *info* pueden ser diferentes para cada $I_{marcaInte}$, dependiendo de la aplicación, pero la I_{marca} puede ser la misma para todos, ya que lógica del código ejecutable insertado no cambia a menos de que se trate de un nuevo tipo o versión de $I_{marcaInte}$. Esto quiere decir que D solamente necesitará crear una I_{marca} para crear cualquier número de $I_{marcaInte}$. La función $J()$ puede ser utilizada tanto por D como por U , pero lo mejor es que sea utilizada por U , ya que esto permite distribuir el poder de cómputo necesario para la función, en lugar de centralizar todo en D .

Una vez creada, la $I_{marcaInte}$ puede ser utilizada por U para interactuar con cualquier otro usuario U , a quien en este caso llamamos U_2 . Para lograr esta interacción, U_2 necesita tener una única aplicación estándar A que sea compatible con los estándares definidos por el grupo de distribuidores D . Para utilizar la información contenida en la $I_{marcaInte}$, la aplicación A emplea la función de separación $S()$, expresada como $S(I_{marcaInte}) = info, F_{dw}, I_{marca}$, de la cual se obtienen los elementos contenidos en la imagen. Enseguida A verifica la firma digital F_{dw} aplicando la llave pública del distribuidor pk_{dist} en la función $Verifica_{pk_{dist}}(F_{dw}, info || I_{marca})$, y si la verificación es exitosa entonces U_2 puede confiar en la integridad de la información y puede estar seguro de que el objeto fue creado por un D autorizado. La aplicación estándar A puede contener diversas implementaciones de F_{dw} , dependiendo en los estándares definidos por el grupo de D . Para extraer el código C incrustado en I_{marca} , U_2 deberá utilizar la función de extracción $X(I_{marca}) = C$, para luego ejecutar C utilizando los datos de *info* como entrada. La Fig. 3 muestra cómo funciona la aplicación estándar.

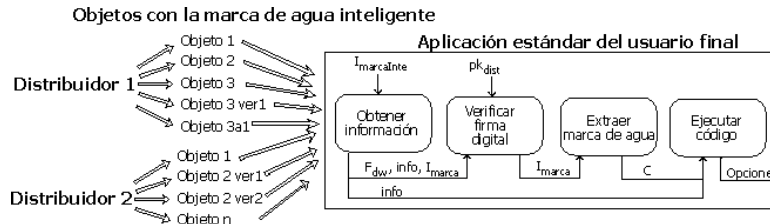


Figura 3. Una aplicación estándar para la marca de agua inteligente

4. Aplicando el modelo de marca de agua inteligente al dinero electrónico

Para probar nuestro modelo de marca de agua inteligente, utilizaremos un escenario de dinero electrónico en el que participan tres diferentes bancos, llamados B_1 , B_2 y B_3 , quienes tienen cada uno su propia implementación de dinero electrónico. Para utilizar la marca de agua inteligente, los bancos han elegido el formato de imagen TIFF (Tagged Image File Format), para crear los archivos de dinero electrónico [25]. Para la implementación elegimos este tipo de archivo sin embargo la marca de agua inteligente también se podría implementar en archivos de tipo PNG, JFIF o cualquier archivo que pueda contener una sección de datos extra a la imagen.

En las etiquetas del archivo TIFF almacenamos la firma digital y los datos, que son: la denominación, el nombre del banco, el tamaño del código ejecutable y la información de la imagen. Cada banco puede agregar la información necesaria en la sección de etiquetas según el modelo de dinero electrónico que elija. En el siguiente segmento, de acuerdo con la especificación de los archivos TIFF, deberá estar la imagen que en este caso es de tipo JPEG, ya que permite modificar su calidad y tamaño de acuerdo a la necesidad de cada banco, esta imagen es la que contiene el código ejecutable.

En nuestro escenario, los bancos acordaron en usar la técnica F5 para insertar y extraer el código en la imagen [26], ya que dicha técnica opera sobre los coeficientes de la transformada discreta del coseno (DCT por sus siglas en inglés), no tiene pérdidas en los datos ocultos y distribuye la información a lo largo de toda la imagen disminuyendo las distorsiones. Este escenario es solo un ejemplo de aplicación y podrían utilizar JSteg o cualquier otra técnica esteganográfica [27].

La anterior técnica la estamos utilizando como una marca de agua frágil, ya que se desea que ante cualquier modificación de la imagen (rotación, translación, recorte, etc.) el dinero electrónico sea inválido. Estamos utilizando el modelo de marca de agua inteligente únicamente con una función de inserción, pero los bancos pueden estipular más funciones siempre y cuando se entregue también la función de extracción en la aplicación estándar del usuario.

Para obtener la firma digital de cada billete, los bancos insertan solamente una vez el código ejecutable en la imagen JPEG y luego pueden reutilizarla al concatenarla con los datos únicos de cada billete y a partir de esto calcular su firma. El banco siempre envía la firma digital y los datos al usuario, en algunos casos también envía la imagen que contiene el código ejecutable dependiendo si el usuario no cuenta con una copia de la imagen. La aplicación estándar del usuario une toda la información y crea el archivo TIFF, en el cual se insertan los datos del billete, la firma digital y la imagen con el código. La Fig. 4 muestra la información contenida dentro de un archivo TIFF.

En la Fig.5 se muestra la interacción general de las entidades involucradas aplicando la marca de agua inteligente en la mayoría de modelos de dinero electrónico. El dinero electrónico creado por B_1 utiliza como modelo de dinero electrónico una firma parcialmente ciega, basada en el problema del logaritmo dis-

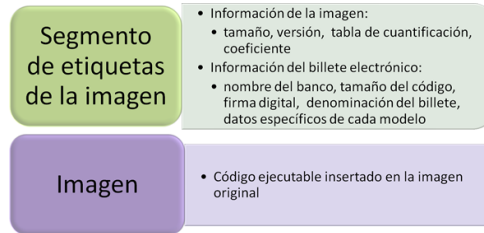


Figura 4. Campos del dinero electrónico con la marca de agua inteligente

creto [28]. En este modelo el usuario se registra la primera vez ante una tercera entidad de confianza, la cual crea un certificado digital con su llave pública. El usuario al momento de retirar un billete electrónico de su banco, envía una parte de los datos del billete cegados mediante una firma especial y otra en claro. Al aplicar la marca de agua inteligente el banco tiene lista la imagen con el código oculto mediante F5 y los datos. El banco firma ciegamente los datos de acuerdo a este modelo y envía la firma digital general de la marca de agua inteligente, la cual como acordaron los bancos, se calcula con SHA y RSA. El usuario recibe la información y la une en el archivo TIFF por medio de la misma aplicación con la que realizó la petición de billetes. Finalmente este archivo es el billete electrónico.

Posteriormente el usuario compra un bien al beneficiario P y le paga con el billete, además entrega también su certificado. P tiene instalada la aplicación estándar la cual separa la información del archivo TIFF y obtiene los datos de las etiquetas y la imagen JPEG. Luego la aplicación localiza el certificado digital que contiene la llave pública del banco y valida la firma digital. Si la firma es válida extrae el código de la imagen mediante el algoritmo de extracción F5.

El código implementa el modelo de cobro elegido por el banco B_1 , el cual valida el certificado del usuario mediante la llave pública del banco y genera un reto para el usuario a partir del certificado. Si el usuario contesta correctamente su pago es aceptado porque significa que realmente es el dueño del billete. Finalmente, el código ejecutable le pregunta a P si desea depositar el dinero en su cuenta o si desea hacerlo más tarde. Esto es posible ya que este es un modelo sin conexión, si el usuario llegara a gastar otra vez este billete, se termina su anonimidad y podría ser acusado de robo. En el momento en que P decide realizar el depósito, la aplicación envía el dinero electrónico al banco de P , donde es validado nuevamente y depositado.

Como segunda opción en este escenario proponemos el uso del billete electrónico condicional [29], el cual sería creado por B_2 . Para ejemplificarlo más fácilmente podemos tener un comercio de apuestas el cual realiza una petición de billetes al banco. El banco genera el billete bajo la marca de agua inteligente por lo tanto ya tiene imágenes con código oculto mediante el algoritmo F5 y crea los datos necesarios, que ya hemos mencionado (tamaño, versión, nombre

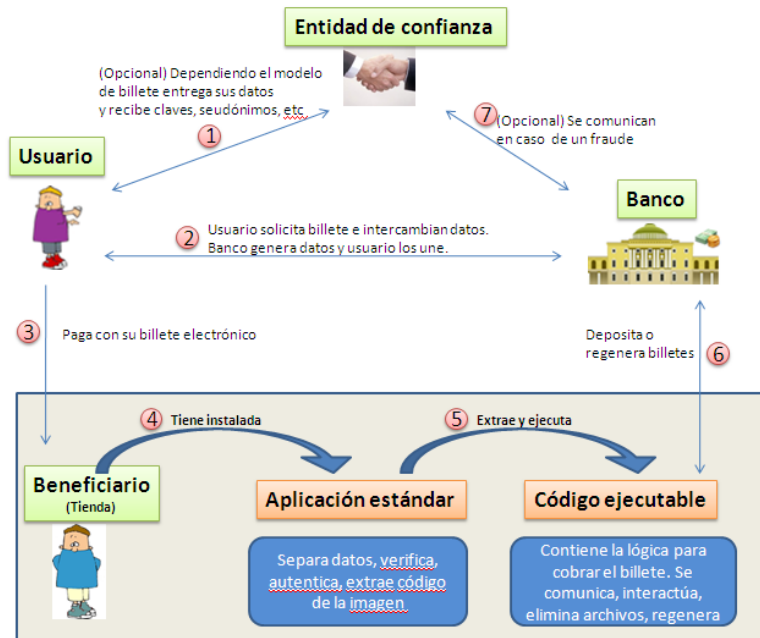


Figura 5. Aplicación de la marca de agua inteligente de manera general en los modelos de dinero electrónico

del banco, tamaño del código, denominación, etc). El banco firma usando SHA y RSA y envía lo anterior al comercio de apuestas.

La aplicación que realizó la petición del billete del comercio de apuestas une los datos en el archivo TIFF además de aumentar cierta información propia y un secreto según lo indicado en este modelo. Como ejemplo el comercio de apuestas tiene identificado cierto tipo de secretos para el equipo de fútbol 1 y otros para el equipo de fútbol 2. El comercio entrega a los clientes los billetes según su equipo favorito. Los clientes deben tener instalada la aplicación estándar para cobrar el dinero. Si los clientes intentan cobrar el billete sin el secreto correcto, solo podrán comprobar que el billete es auténtico mediante la la firma digital, sin embargo cuando se ejecute el código oculto no podrán cobrar el billete porque no tienen el secreto requerido. Cuando se conoce el resultado del encuentro de fútbol el comercio publica el secreto de los billetes del equipo ganador. Por lo tanto ahora los clientes al procesar el billete con la aplicación estándar pueden introducir el secreto cuando se extrae y ejecuta el código oculto. Al coincidir el secreto, el código realiza una conexión segura con el banco y pide la cuenta del usuario para depositar el billete.

Como podemos ver hasta ahora el modelo de marca de agua inteligente se puede incorporar a los anteriores modelos de dinero electrónico. Como tercera opción en este escenario, nosotros proponemos una máquina expendedora de dinero electrónico, la cual también está basada en el modelo de marca de agua inteligente. Esta solución puede ayudar a incrementar el comercio electrónico en países en desarrollo, donde hay mucha gente que quiere realizar compras en Internet pero que no cuenta con una cuenta bancaria, lo cual es un requisito para la mayoría de las formas de pago en línea. Esta tercera opción se implementó de forma práctica en el lenguaje JAVA.

En este caso, el usuario introduce el dinero físico en la máquina donde B_3 distribuye el dinero, la cual solicita un nuevo billete electrónico a un servidor utilizando una conexión segura. En nuestra implementación simulamos la máquina expendedora con un software, en el cual el usuario selecciona las opciones de dinero físico. El servidor que representa al banco, tiene implementados diferentes servicios web, una base de datos para guardar la información de cada billete y previamente oculta el código ejecutable en ciertas imágenes. Dicho código se encuentra compactado en un archivo JAR, el cual contiene la lógica de cobro del billete y ofrece distintas opciones al usuario, esta imagen se encuentra guardada en la máquina expendedora y se puede actualizar según lo indique el banco.

El servidor al recibir una petición de nuevo billete, le responde a la máquina expendedora con los datos del billete (el nombre del banco, el tamaño del código ejecutable, la denominación, la versión de la implementación, un número único de identificación) y la firma digital de dichos datos. La firma se implemento utilizando SHA y RSA. La máquina expendedora como ya tiene previamente guardadas las imágenes de las diferentes denominaciones con el código ejecutable insertado, solo realiza la copia de una imagen y la agrega al archivo TIFF junto con los datos y firma del billete que recibe. Finalmente el usuario guarda el dinero electrónico en algún dispositivo de almacenamiento, como una memoria USB o en su teléfono celular. En la Fig. 6 se muestra la máquina expendedora en el escenario de dinero electrónico.

Cuando el usuario entrega este tipo de dinero electrónico a P , este último también puede utilizar la máquina expendedora para cambiar dinero electrónico por dinero físico. En este caso, la máquina expendedora contactará al servidor para verificar el dinero electrónico y registrarlo como cobrado.

Otra opción que P tiene para cobrar el dinero, es utilizar la aplicación estándar, la cual en nuestra implementación lee y muestra al usuario la información de las etiquetas del archivo TIFF que se encuentran identificadas por números y se obtiene la imagen JPEG, para finalmente verificar la firma digital. La aplicación estándar tiene una sección para dar de alta o baja certificados digitales los cuales hayan sido creados por una autoridad bancaria y a partir de estos certificados se obtiene la llave pública necesaria para verificar la firma.

Si la firma es válida y el usuario decide cobrar el billete, se extrae el código ejecutable de la imagen JPEG por medio de la técnica F5 y se ejecuta el código. En este punto la lógica de cobro que da sentido a los datos del billete es la que se

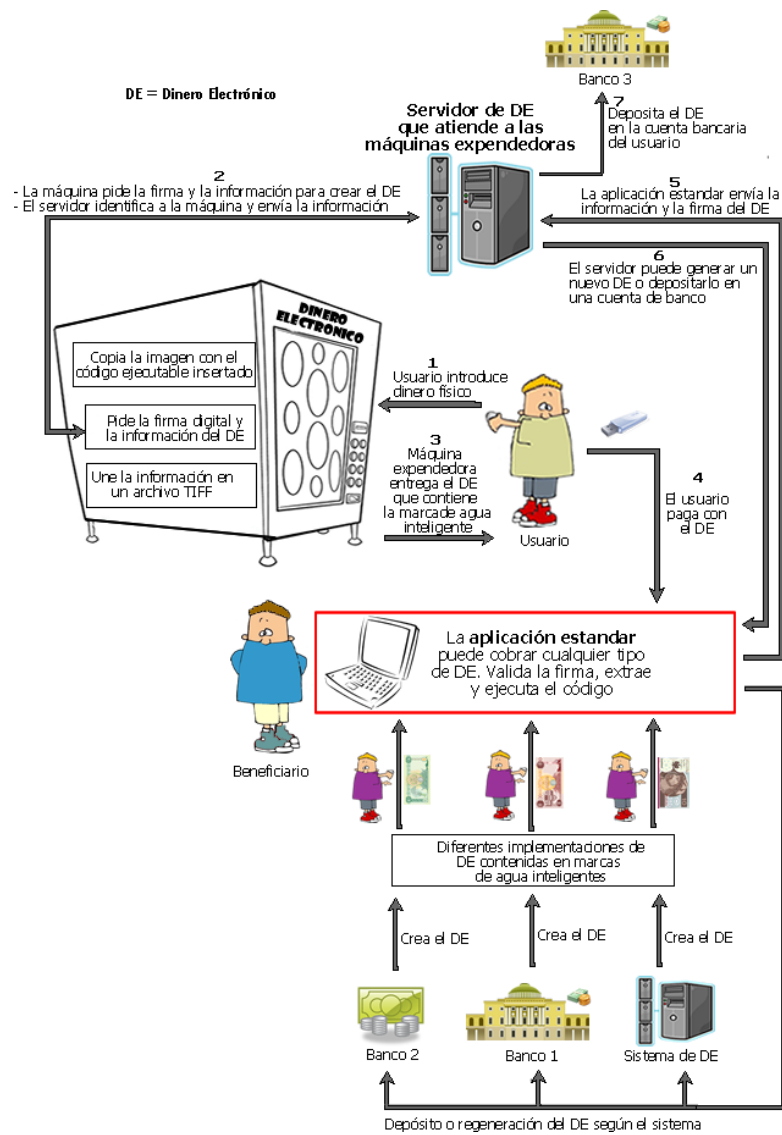


Figura 6. Máquina expendedora de dinero electrónico

encuentra en el código ejecutable, el cual ofrece al usuario dos opciones: generar un nuevo billete o depositarlo en una cuenta bancaria.

En ambos casos el código se conecta al servicio web ofrecido por el banco y envía los datos y la firma digital. El servidor verifica la autenticidad de los datos y revisa que el identificador único del billete no haya sido cobrado previamente. En caso de la opción de generar un nuevo billete, el servidor regresa un nuevo segmento de datos y una firma digital, los cuales los recibe el código en ejecución y se utilizan para generar un nuevo archivo TIFF de la misma forma que lo hizo la máquina expendedora. Dicho billete volverá a tener la función de dinero electrónico y puede usarse para pagar a otro usuario. En caso de elegir depositarlo en una cuenta bancaria, la implementación del código solicita a P los datos de la cuenta y envía la petición al banco correspondiente.

Se realizaron pruebas instalando la máquina expendedora y la aplicación estándar en diferentes máquinas, se regeneraron entre ellas varios billetes de diferentes denominaciones y se depositaron a diferentes cuentas los billetes. Se realizaron modificaciones a los billetes, las cuales fueron detectadas por la aplicación estándar al avisar que la firma digital no era válida. También se hicieron pruebas con diferentes códigos ejecutables ocultos los cuales algunos ofrecían premios a los usuarios. El código ejecutable en el archivo JAR midió aproximadamente 23Kb dependiendo de las opciones que se ofrecieron y las imágenes usadas fueron de 474 x 308 píxeles.

5. Conclusiones

Hemos presentado un nuevo modelo llamado marca de agua inteligente, el cual simplifica la interacción entre distintas entidades y expande las aplicaciones actuales de las marcas de agua, dándoles flexibilidad a través de la implementación del código ejecutable. La marca de agua inteligente es segura para los usuarios, y el código es únicamente ejecutado cuando la aplicación estándar instalada en la computadora del usuario valida que el archivo es íntegro y que fue creado por un distribuidor válido. La función de inserción es realizada solamente una vez por el distribuidor para crear tantos objetos como desee y la función de unión puede ser realizada por el usuario, distribuyendo así el poder de cómputo necesario.

Aplicamos este concepto para resolver incompatibilidades entre diferentes implementaciones de dinero electrónico, de esta forma los beneficiarios pueden aceptar, de manera transparente, cualquier implementación creada bajo el mismo modelo de marca de agua inteligente. Otra de sus ventajas es agrupar toda la información de los modelos de dinero electrónico en un solo archivo brindando portabilidad y una fácil adopción de la tecnología al usuario.

Los diferentes distribuidores de dinero electrónico deben acordar previamente el tipo de archivo y datos comunes que utilizarán. Sólo necesita ser instalada una única aplicación estándar en la máquina del beneficiario, para poder manejar todos los archivos de dinero electrónico que los usuarios entreguen. Para mostrar el uso del modelo de marca de agua inteligente, lo aplicamos en un escenario que

contiene tres implementaciones distintas de dinero electrónico. La última de estas es nuestra propuesta de una máquina expendedora de dinero electrónico, la cual es una opción para evitar el uso de una cuenta bancaria y así ayudar a cubrir las necesidades que tienen los países en desarrollo en el comercio electrónico. La marca de agua inteligente puede simplificar la necesidad de tener diferentes aplicaciones, es segura para los usuarios y cada distribuidor puede implementar libremente sus propias reglas y lógica en el código ejecutable de la imagen.

Como trabajo futuro, proponemos la búsqueda de más aplicaciones de la marca de agua inteligente, para demostrar que nuestro modelo puede ser aplicado en varios escenarios. Se pueden realizar nuevas investigaciones para mejorar las medidas de seguridad que deben ser agregadas a la marca de agua inteligente y así realizar las validaciones de una forma más rápida. Se pueden implementar prácticamente más modelos de dinero electrónico y finalmente, se podrían comparar distintos algoritmos esteganográficos para medir la cantidad de información que logran ocultar, para así obtener el mayor provecho del espacio para el código ejecutable.

Agradecimientos

Al equipo ADAM en el INRIA Lille Nord Europe (<http://adam.lille.inria.fr>), por la ayuda y el apoyo recibidos para la realización de este artículo.

Referencias

1. Petitcolas, F.A.P.: Digital Watermarking. In: Digital Rights Management: Technological, Economic, Legal and Political Aspect. Volume 2770 of LNCS. Springer Berlin - Heidelberg (2003) 81–92
2. Suhail, M., Chun-Shien, L.: Digital Watermarking for Protection of Intellectual Property. In: Multimedia Security: Steganography and Digital Watermarking Techniques for Protection of Intellectual Property. Idea Group Publishing, Covent Garden, London (2004) viii, 255 p.
3. Kutter, M., Petitcolas, F.A.P.: A fair benchmark for image watermarking systems. In Wong, P.W., Delp, E.J., eds.: Electronic Imaging '99, Security and Watermarking of Multimedia Contents. Volume 3657., San Jose, California, U.S.A (1999) 226–239
4. Wang, Y., Lü, S., Liu, Z.: A simple anonymous fingerprinting scheme based on blind signature. In: Information and Communications Security, 5th International Conference, ICICS 2003, Huhehaote, China, October 10-13, 2003. (2003) 260–268
5. Cox, I.: Digital Watermarking: Principles and Practice. Morgan Kaufmann Publishers Inc. (2001)
6. Felica Networks Inc: Mobile felica. <http://www.felicanetworks.co.jp>
7. JR East: Mobile suica. <http://www.jreast.co.jp/e/press/20071201/index.html>
8. AEON Group: Waon. <http://www.waon.com/>
9. Seven and I Holdings Co: Nanaco. <http://www.nanaco-net.jp/>
10. Callas, J., Desmedt, Y., Nagy, D., Otsuka, A., Quisquater, J.J., Yung, M.: Real electronic cash versus academic electronic cash versus paper cash (panel report). (2008) 307–313

11. Okamoto, T., Ohta, K.: Universal electronic cash. In: CRYPTO '91: Proceedings of the 11th Annual International Cryptology Conference on Advances in Cryptology, London, UK, Springer-Verlag (1992) 324–337
12. Netherlands: Chipknip. <http://www.chipknip.nl>
13. Electronic Payment Certification Institute: Proton. <http://www.epci.be>
14. EURO Kartensysteme GmbH: Geldkarte. <http://www.geldkarte.de>
15. BMS Exploitation: Moneo. <http://www.moneo.net>
16. Jakobsson, M., Juels, A.: X-cash: Executable digital cash. In: FC '98: Proceedings of the Second International Conference on Financial Cryptography, London, UK, Springer-Verlag (1998) 16–27
17. Cochran, J.T.: “Steganographic Computer Warfare”. Master’s thesis, Air Force University. Wright-Patterson Air Force Base, Ohio, United States. (2000)
18. Gómez, R., Ramírez, G.: Using digital images to spread executable code on internet. In: 6th International Workshop on Innovative Internet Community Systems, I2CS, Neuchâtel, Switzerland (2006)
19. Rogers, M.: Steganographic trojans. In: DEF-CON X, Las Vegas, Nevada (2002)
20. McAfee: Trojan frog on the loose. <http://www.avertlabs.com/research/blog/?p=36> (2006)
21. Erdelyi, G.: W32.perrun. <http://www.f-secure.com/v-descs/perrun.shtml> (2002)
22. Microsoft TechNet: Microsoft security bulletin ms04-028, buffer overrun in jpeg processing (gdi+) could allow code execution. <http://www.microsoft.com/technet/security/bulletin/MS04-028.mspx> (2004)
23. Tran-Nguyen, A.N.: E-Banking and e-payments: implications for developing and transition economies. In: UNCTAD Information Economy Report 2007-2008. United Nations Publication, New York and Geneva (2007) xxxvi, 348 p.
24. Menendez, P.: Estudio de comercio electrónico 2008. <http://www.amipci.org.mx/estudios.php> (2008)
25. AdobeDevelopers: Tiff revision 6.0. <http://partners.adobe.com/public/developer/en/tiff/TIFF6.pdf> (1992)
26. Westfeld, A.: F5-a steganographic algorithm: High capacity despite better steganalysis. In: 4th International Workshop on Information Hiding, Springer-Verlag (2001) 289–302
27. Upham, D.: Jpeg-jsteg-v4. <http://www.funet.fi/pub/crypt/steganography/jpeg-jsteg-v4.diff.gz> (2008)
28. Miyazaki, S., Sakurai, K.: A more efficient untraceable e-cash system with partially blind signatures based on the discrete logarithm problem. In: FC '98: Proceedings of the Second International Conference on Financial Cryptography, London, UK, Springer-Verlag (1998) 296–308
29. Shi, L., Carbutar, B., Sion, R.: Conditional e-cash. In Dietrich, S., Dhamija, R., eds.: Financial Cryptography. Volume 4886 of Lecture Notes in Computer Science., Springer (2007) 15–28

Aumento de la Fiabilidad de la Evidencia en un Protocolo de Intercambio Justo mediante la División del Entorno de Firma

Jorge L. Hernández-Ardieta, Ana I. González-Tablas, Benjamín Ramos
Álvarez, y Arturo Ribagorda Garnacho

Departamento de Informática, Grupo SeTI, Universidad Carlos III de Madrid
Avda. de la Universidad 30, 28911 – Leganes, España.
jlopez.ha@gmail.com, aigonzal@inf.uc3m.es, benja1@inf.uc3m.es,
arturo@inf.uc3m.es

Resumen El respaldo legal de la firma electrónica unido a su reconocimiento como evidencia de no repudio por parte de los estándares internacionales hace que la seguridad del proceso de creación de firmas sea una cuestión de suma importancia. Sin embargo, numerosos estudios demuestran que existe una gran variedad de ataques a entornos de creación de firmas, lo cual socava la fiabilidad de la firma electrónica como evidencia de no repudio y evidencia en procedimientos legales. En este artículo se presenta un protocolo en el cual se aumenta considerablemente la fiabilidad de la evidencia generada aun cuando el firmante emplea un entorno de creación de firmas inseguro. El protocolo se ha diseñado tomando como base un protocolo de intercambio justo presentado con anterioridad, en el cual se asegura que ningún participante obtiene una ventaja respecto al otro durante la transacción.

Palabras clave: Firma electrónica, evidencia, no repudio, protocolo de intercambio justo, ataques.

1. Introducción

Se han hecho muchos esfuerzos por promocionar el comercio electrónico, especialmente en relación a la mejora de su seguridad. Quizá el más relevante ha sido el apoyo que los Gobiernos y la Industria han dado a la firma electrónica. Una firma electrónica se considera como datos en formato electrónico que se adjuntan o asocian a otros datos electrónicos y que actúan como método de autenticación [16]. Sin embargo, múltiples legislaciones consideran la firma electrónica legalmente equivalente a la firma manuscrita, asignándola la propiedad de evidencia en procedimientos legales [16,21,13,45]. Por otra parte, los estándares internacionales también consideran a la firma electrónica como evidencia de no repudio en transacciones electrónicas [29,30], tanto cuando se aplica en transacciones de comercio electrónico como de otra índole. Es obvio que la firma electrónica se considera un elemento clave, y por tanto su seguridad es una cuestión crítica que debe analizarse con rigurosidad.

Aunque las legislaciones actuales son neutras desde un punto de vista tecnológico, el hecho es que los requisitos establecidos sólo se cumplen actualmente por las firmas digitales basadas en criptografía asimétrica [38,14] y soportadas por una Infraestructura de Clave Pública (PKI) [11] la cual permita establecer confianza en las identidades digitales empleadas. Como resultado, cuando una firma digital se verifica correctamente, el firmante, cuya identidad está vinculada a la clave pública correspondiente, no podría rechazar haber creado dicha firma, y, por ende, no podría negar su participación ni compromiso adquirido en la transacción dada.

Sin embargo, los entornos donde se producen estas firmas, especialmente aquellos empleados por los usuarios finales, no son todo los seguros que debieran ser, dada la cantidad de amenazas que se ciernen sobre ellos [25]. Ciertos ataques se centran en comprometer la clave privada de firma con el fin de poder posteriormente generar firmas electrónicas sin el consentimiento ni conocimiento del firmante [12,22,43,34]. Otros ataques tienen por objetivo engañar al firmante de forma que éste firme unos datos diferentes a los deseados sin detectarlo [41,42]. Numerosos resultados muestran que es difícil asegurar la fiabilidad de una firma electrónica si se emplean documentos o datos con formatos complejos, como por ejemplo aquellos que permiten la inclusión de macros o gráficos [3,31,32]. Este tipo de ataques, que nosotros denominamos ataques semánticos, anulan los mecanismos WYSIWYS [40] (*What You See Is What You Sign*), mediante los cuales se intenta ofrecer al firmante un último paso de verificación de los datos inmediatamente anterior a la generación firma.

Si no se puede proveer al firmante de medios seguros para la creación de firmas, éste no debería ser responsable del compromiso adquirido en documentos firmados en su nombre (ej. orden de compra, contrato, correo electrónico, etc.). El firmante podría fácilmente probar que la clave privada pudo haber sido comprometida o que el documento que intentaba firmar pudo haber sido modificado justo antes de realizar la firma. El firmante podría probar ante el juez que existe una duda razonable respecto a la seguridad del entorno de firma, y por tanto la validez de la firma se vería puesta en cuestión. Ya existe jurisprudencia al respecto. Recientemente se ha producido un fallo en favor del demandado al ser incapaz la entidad demandante de demostrar un cierto grado de seguridad en el proceso de creación de la firma repudiada [2]. No obstante, la actual legislación en materia de firma electrónica traslada al firmante la carga de prueba [35], siendo éste quien debe probar la falsedad de la firma. Este contexto sin duda le sitúa en un escenario de incertidumbre, provocando una falta de confianza en la tecnología y sus procesos.

El protocolo que se presenta en este artículo tiene por objetivo incrementar la fiabilidad de las firmas electrónicas, reforzando su propiedad de evidencia de no repudio así como su capacidad de evidencia en procedimientos legales. El efecto buscado es aumentar la confianza de los usuarios en esta tecnología, lo cual fomentará el uso del comercio electrónico, entre otros modelos de negocio. Este protocolo toma como punto de partida el protocolo de intercambio justo

OFEPSP [24], el cual asegura la equidad en la transacción en la que tanto origen como receptor participan.

La seguridad del protocolo aquí propuesto se ha verificado formalmente [27] tomando como referencia el modelo de intruso Dolev-Yao [15], en el cual el atacante tiene absoluto control sobre la red pero no puede realizar criptoanálisis. En este sentido el atacante puede intercambiar, reenviar, reutilizar o eliminar cualquier mensaje intercambiado entre los participantes. Las propiedades de seguridad verificadas incluyen la correcta autenticación de los participantes así como el aseguramiento de la integridad de los mensajes intercambiados.

El artículo se ha estructurado como sigue. La siguiente sección 2 proporciona una breve descripción de OFEPSP. Las modificaciones de diseño realizadas se detallan en la sección 3. La sección 4 incluye los resultados del análisis formal de seguridad del protocolo. Finalmente se concluye el artículo en la sección 5.

2. OFEPSP

OFEPSP [24] es un protocolo orientado a transacciones electrónicas donde dos participantes pueden intercambiar información de forma equitativa y segura. El origen de la transacción envía cierto mensaje firmado electrónicamente mientras que el receptor responde con una prueba de recepción de dicho mensaje. Al emplear firmas electrónicas durante el proceso, ambos participantes adquieren determinado compromiso en la transacción: el origen no puede repudiar haber enviado el mensaje y el receptor no puede repudiar su recepción. Como protocolo de intercambio justo, OFEPSP asegura que ningún participante obtiene una ventaja respecto al otro en la transacción. Por lo tanto, o bien ambos participantes obtienen la información esperada de la otra parte o ninguno obtiene información relevante del otro. Al ser un protocolo optimista, existe una Tercera Parte de Confianza (TTP) que interviene en el protocolo con el fin de resolver situaciones de desequilibrio, pero únicamente cuando se produce un error en el protocolo (p. ej. se pierde un mensaje por fallo en la red) o cuando alguna de las partes se comporta de forma maliciosa.

La mayoría de los protocolos de este tipo emplean cifrado simétrico para evitar que un participante desvele información útil al otro sin que éste haya adquirido cierto compromiso en la transacción [33]. Por el contrario, OFEPSP está diseñado en base a políticas de firma [18,39]. Una política de firma es un documento que recoge las reglas y requisitos para la creación y validación de firmas electrónicas, y bajo las cuales una firma electrónica se considera válida en un contexto transaccional concreto. Las políticas de firma se definen tanto en formato entendible por el ser humano como en formato estructurado, como XML y ASN.1, permitiendo una generación y validación automática de firmas. Una referencia a la política de firma empleada por el firmante se incluye como atributo firmado en las firmas generadas, permitiendo al verificador comprobar si dicha firma cumple con los requisitos impuestos por la política de firma referenciada.

OFEPSP emplea una política de firma para establecer los pasos y condiciones a seguir por los participantes de la transacción (origen, receptor, TTP) de forma

que se asegure la justeza de la misma. Gracias a este diseño, OFEPSP permite al origen evaluar las condiciones que dictaminarán la transacción. De esta forma, el origen se convierte en un participante activo que puede decidir si confía o no en la entidad que emite dicha política así como aceptar o no los términos que en ella se establecen. Por el otro lado, el receptor publica las condiciones a cumplir por cualquier origen que quiera realizar una transacción con él. Así pues, OFEPSP tiene una clara orientación a escenarios de comercio electrónico donde estas características aportan un valor añadido.

A día de hoy los organismos de estandarización ETSI e IETF han publicado un conjunto de informes técnicos no estándares y una RFC Experimental, respectivamente, en los cuales se define la estructura de una política de firma en formato ASN.1 [39,20] y XML [17]. Sin embargo, estas propuestas sólo permiten establecer los requisitos para una única firma electrónica. No se contempla la gestión de múltiples firmas electrónicas en cuanto a sus requisitos individuales ni las relaciones entre ellas. Por ello, protocolos en los cuales la evidencia se componga de múltiples firmas electrónicas, como es el caso de los protocolos de intercambio y no repudio justos, no quedan cubiertos. ETSI publicó un informe técnico en el que se recogían los requisitos de alto nivel que debería cumplir una política de firmas para cubrir este objetivo [19].

En [26] proponemos un framework completo para la definición y uso de políticas de firmas extendidas, cubriendo así la necesidad establecida por ETSI. De esta forma, se puede definir cualquier árbol de firmas que sea necesario para otorgar validez a la transacción. El árbol de firmas incluye por tanto las firmas necesarias que deben estar presentes para que los compromisos adquiridos por los participantes de la transacción sean vinculantes, así como los requisitos particulares de cada firma y las relaciones espacio-temporales entre ellas: orden de generación, dependencia respecto a otras firmas, etc. Así mismo, el framework define el algoritmo de generación y validación de firmas de acuerdo a la política de firmas extendida requerida. Gracias a esta nueva propuesta, protocolos como OFEPSP o el que detallamos en la sección 3 pueden ser fácilmente implementados.

OFEPSP se compone de un protocolo principal y un protocolo de recuperación. El protocolo principal permite a origen y receptor interactuar durante una serie de etapas hasta que se genera la evidencia final llamada NRA (*Non-Repudiation evidence of Acknowledgment*). Durante la ejecución del protocolo principal se generan dos evidencias intermedias no vinculantes, pero de las cuales depende el NRA. Así pues, el origen genera la firma no vinculante NRO (*Non-Repudiation evidence of Origin*), mientras que el receptor genera el NRR (*Non-Repudiation evidence of Receipt*). NRO y NRR se generan sobre los datos de la transacción, mientras que el NRA se genera sobre la firma NRR. De esta manera se compone un árbol de firmas con dos firmas en el primer nivel (NRO y NRR) y una tercera firma (NRA) en el segundo nivel, dependiente de NRR. La evidencia vinculante, así definida por la política de firma, consiste en el árbol de firmas completo.

Esta evidencia es la que realmente ata a los participantes respecto a los compromisos adquiridos en la transacción. Por tanto, hasta que el árbol de firmas no se genere correctamente, los participantes no adquieren ningún compromiso vinculante. Una vez generado, el origen no puede repudiar el envío del mensaje y el receptor no puede rechazar su recepción. Estos compromisos pueden extenderse a otros más particulares dependiendo del tipo de compromiso necesitado en el contexto de negocio en el cual tiene lugar la transacción, y que se configura y personaliza a nivel de política de firma. Por ejemplo, la evidencia puede implicar no sólo el no repudio de origen y no repudio de recepción para origen y receptor, respectivamente, sino también un compromiso de aceptación de contenido por ambas partes.

Por último, el protocolo de recuperación permite al receptor obtener el NRA en caso que se produzca algún error en la comunicación o si el origen se comporta maliciosamente.

3. Protocolo de Intercambio Justo con División del Entorno de Creación de Firma

Esta sección detalla el nuevo diseño que aumenta la fiabilidad de la evidencia vinculante aun cuando se empleen entornos de firma inseguros. El nuevo diseño añade además un nuevo subprotocolo de aborto que permite al origen abortar la transacción en caso que detecte que ésta es una transacción no deseada.

La siguiente sección 3.1 introduce el diseño ideado así como las suposiciones del entorno de partida. La notación empleada en la definición del protocolo se explica en la sección 3.2. Finalmente se describe el protocolo principal y ambos subprotocolos.

3.1. Aproximación al Diseño

La principal novedad del protocolo aquí propuesto radica en el empleo por parte del origen de dos entornos diferentes para la generación de las firmas electrónicas que componen la evidencia del protocolo.

Por un lado, el origen debe firmar la información a enviar al receptor mediante el Entorno de Creación de Firmas (ECF). ECF debe disponer de una aplicación de firma electrónica que permita al origen generar firmas electrónicas basadas en criptografía asimétrica, empleando un dispositivo de creación de firmas hardware (SCDev1) - ej. una tarjeta inteligente. SCDev1 almacenará de forma segura una clave privada de firma (Key1).

Por otro lado, proponemos que el origen confirme la transacción haciendo uso de un segundo entorno, llamado Entorno de Confirmación de Transacción (ECT), y un SCDev diferente (SCDev2), que contenga una clave privada de firma distinta (Key2).

En [28] hemos demostrado formalmente cómo la división del entorno de creación de firmas mejora sustancialmente la fiabilidad de la evidencia resultante. En particular, el uso de dos entornos y dos SCDev-Key diferentes asegura que

un atacante no obtendrá ningún beneficio en caso de comprometer la seguridad del ECF o ECT. Por ello, la seguridad del protocolo es sustancialmente mayor que la de los actuales sistemas de firma electrónica, los cuales se basan en un único entorno.

El mensaje a firmar por el origen debe ser normalizado de acuerdo a una plantilla, la cual restringe el formato del mensaje (ASCII) así como el contenido del mismo. La plantilla debe ser definida por el receptor de acuerdo a las necesidades de la transacción. El mensaje m enviado por el origen será posteriormente procesado por el receptor teniendo en cuenta la plantilla empleada. De esta forma la semántica de la información se mantiene a lo largo de toda la transacción, evitando los ataques semánticos.

Un ejemplo de plantilla para la restricción de la semántica de la información generada en un proceso de negocio es el estándar UBL desarrollado por OASIS [37]. UBL define un conjunto de componentes de negocio reutilizables para la composición de documentos que se generan e intercambian en una transacción dada. Por ejemplo, se definen componentes para facturas, pedidos y recibos. Así mismo, UBL se integra perfectamente en el estándar ebXML de OASIS [36], el cual define el framework y el protocolo para el intercambio de mensajes genéricos de negocio. El protocolo aquí propuesto puede por tanto adaptarse fácilmente para soportar UBL como modelo de información cuando la transacción se englobe en un contexto de comercio electrónico.

El diseño y seguridad del protocolo dependen de las siguientes consideraciones:

1. El origen será normalmente un usuario final. Tanto ECF como ECT se consideran entornos inseguros. Ello es debido a que el usuario empleará plataformas habitualmente poco seguras, es decir, plataformas donde la probabilidad de sufrir un ataque o verse infectadas por malware es alta. Un ejemplo puede ser un ordenador personal/corporativo que actúe como ECF y un dispositivo móvil con capacidades criptográficas que actúe como ECT.
2. La seguridad de tanto ECF como ECT puede verse comprometida. Sin embargo, la probabilidad de un ataque distribuido a ECF y ECT por malware diferente, y que colabore con el fin de generar una evidencia vinculante fraudulenta, se demuestra sustancialmente menor que en caso de emplear un único entorno [28].
3. El entorno empleado por el receptor para generar las firmas electrónicas correspondientes se considera seguro. En este sentido, dicho entorno opera conforme a sus especificaciones, lo cual no implica que estas especificaciones no puedan ser perjudiciales para el origen (p. ej. el receptor intenta estafar al origen en un proceso de compra on-line). Por ello, el receptor no es de confianza aunque se descarta la posibilidad de que su entorno sufra un ataque que comprometa la seguridad del protocolo. Este requisito aplica a escenarios donde el receptor es una corporación u organización más que un usuario final, como en transacciones de comercio electrónico.

4. La información a intercambiar debe estar en formato ASCII sin formato (texto plano). Este requisito es fácilmente alcanzable en contextos como sistemas de correo electrónico o comercio electrónico.

3.2. Notación Básica

A continuación se detalla la notación empleada en la definición del protocolo:

SP Política de Firma empleada en el protocolo
 $X \rightarrow Y : m$ Entidad X envía un mensaje m a la Entidad Y
 $X \leftarrow Z : SP$ Entidad X solicita SP a la entidad Z
 $S_X(m)$ Firma electrónica generada por la entidad X sobre el mensaje m
 $S_X(m|SP)$ Firma electrónica generada por la entidad X sobre el mensaje m en base a la política de firma SP
 $POO = S_O(m, \ell, tpl\_id|SP)$
 Prueba de Origen (*Proof of Origin*) respecto al mensaje m .
 $POR = S_R(m, \ell, tpl\_id|SP)$
 Prueba de Recepción (*Proof of Receipt*) respecto al mensaje m .
 $NRO = S_O(POR|SP)$
 Evidencia de no repudio de origen (*Non-Repudiation evidence of Origin*) respecto al mensaje m generada sobre POR.
 $NRR = S_R(NRO|SP)$
 Evidencia de no repudio de recepción (*Non-Repudiation evidence of Receipt*) respecto al mensaje m generada sobre NRO.
 $NRA_O = S_O(NRR|SP)$
 Evidencia de no repudio de aprobación (*Non-Repudiation evidence of Acknowledgment*) generada sobre NRR. Esta es la firma que termina de componer la evidencia vinculante del protocolo.
 $NRA_{TTP} = S_{TTP}(NRR|SP)$
 Evidencia de no repudio de aprobación generada por el Tercero de Confianza (TTP) sobre NRR. Esta firma compone igualmente la evidencia vinculante del protocolo, pero cuando se ejecuta el subprotocolo de recuperación.

Cada ejecución del protocolo se identifica mediante un identificador único ℓ . Las entidades deben emplear la plantilla para fijar la semántica de la información a enviar por el origen. Esta plantilla debe referenciarse en cada mensaje haciendo uso de su identificador tpl_id .

Al igual que en OFEPSP, la evidencia vinculante consiste en el árbol de firmas completo, que en este caso se compone de las firmas electrónicas POO y POR en el primer nivel, y las firmas NRO, NRR y NRA en niveles consecutivos y dependientes secuencialmente.

3.3. Protocolo Principal

En el protocolo principal el origen inicia la transacción al enviar el mensaje firmado al receptor empleando ECF. Posteriormente, el origen confirmará dicha transacción empleando ECT. El receptor intercambiará diferentes firmas

electrónicas con tanto ECF como ECT hasta que la firma electrónica (NRA) que completa el árbol es generada. A continuación se detalla el protocolo principal empleando la notación anteriormente descrita:

$$(1) \quad O_{ECF} \leftarrow R : tpl [tpl\_id], S_R (tpl [tpl\_id])$$

Primero, el origen (O) solicita al receptor (R) la plantilla identificada por tpl_id (1) haciendo uso de ECF.

$$(2) \quad O_{ECF} \leftarrow TTP-SP : SP, S_{TTP-SP} (SP)$$

En el siguiente paso (2), el origen obtiene la política de firma SP necesaria para interactuar con el receptor. Una vez que el origen dispone de la plantilla y la política de firma, puede generar el mensaje y POO.

$$(3) \quad O_{ECF} \rightarrow R : m, \ell, tpl\_id, POO$$

Posteriormente (3), el origen inicia el protocolo al enviar el mensaje m , el identificador único de protocolo ℓ , el identificador de la plantilla y POO.

$$(4) \quad R \leftarrow TTP-SP : SP, S_{TTP-SP} (SP)$$

$$(5) \quad R \rightarrow O_{ECT} : m, \ell, tpl\_id, POO, POR$$

El receptor recupera la política de firma (4), si no dispone de ella, y valida el POO recibido. Después genera y envía el POR al origen, pero en este caso al ECT (5). El paso (4) puede evitarse, mejorando la eficiencia del protocolo, si el receptor accede al TTP-SP una única vez y en ejecuciones posteriores del protocolo emplea la copia local.

$$(6) \quad O_{ECT} \rightarrow R : NRO$$

El origen debe generar el NRO si y sólo si la información recibida en el paso (5) corresponde a una transacción deseada, y POO y POR son correctamente verificados.

$$(7) \quad R \rightarrow O_{ECF} : NRR$$

Una vez que el origen ha confirmado la transacción mediante ECT, el receptor envía el NRR al entorno ECF del origen (7).

$$(8) \quad O_{ECF} \rightarrow R : NRA_O$$

En el último paso (8) el origen completa la transacción enviando el NRA al receptor.

Aunque no se ha mostrado, las firmas POO, POR, NRO, NRR y NRA deben contener el correspondiente sello temporal. El sello de tiempo es una prueba proporcionada por una Autoridad de Sellado de Tiempo (TSA) de que ciertos datos existían antes de un momento determinado. El procedimiento de obtención del sello temporal debe hacerse conforme al estándar [1], implicando así la participación de una TSA. La política de firma recogerá las condiciones de dependencia temporal y secuencial entre todas las firmas del protocolo. El verificador

podrá comprobar que dichas condiciones se cumplen tomando como referencia temporal los sellos de tiempo incluidos en las firmas.

3.4. Subprotocolo de Recuperación

Este subprotocolo permite al receptor obtener la evidencia NRA en caso que ocurra un error en la transacción o el origen se comporte de forma maliciosa. Este subprotocolo debe ejecutarse si el receptor no es capaz de obtener el NRA por parte del origen, e implica los siguientes pasos:

$$(1) \quad R \rightarrow TTP : H(m, \ell, tpl\_id), \ell, POO, POR, NRO, NRR$$

Si el protocolo ha sido abortado por el origen, entonces

$$(2a) \quad TTP \rightarrow R : S_{TTP}(S_O(abort, \ell|SP)|SP)$$

en caso contrario

$$(2b) \quad TTP \leftarrow TTP-SP : SP$$

$$(3b) \quad TTP \rightarrow R, O_{ECF}, O_{ECT} : NRA_{TTP}$$

En (1) el receptor envía al TTP las firmas POO, POR, NRO y NRR. Con el objetivo de proteger la privacidad de los participantes, sólo se envía el resumen (*hash*) de la información firmada en POO y POR, esto es, el mensaje, el identificador único correspondiente a la presente ejecución y la referencia a la plantilla. Aun así, el TTP es capaz de verificar POO y POR empleando directamente dicho resumen, siempre y cuando el esquema de firma digital se base en criptografía de clave pública (p. ej. RSA, DSA, ECDSA). TTP debe descifrar POO y POR - empleando la correspondiente clave pública -, obteniendo el resumen de los datos firmados, y comparar dicho valor con el proporcionado en $H(m, \ell, tpl\_id)$. ℓ se envía en (1) para permitir al TTP recuperar y actualizar la información correspondiente a dicha transacción.

Si el protocolo había sido abortado previamente, TTP simplemente reenvía la evidencia de aborto al receptor (2a). En caso contrario, TTP genera NRA_{TTP} teniendo en cuenta la política de firma referenciada - (2b) y (3b) -, pero sólo en la primera solicitud. Al mantener una copia local de dicha información, puede reutilizarla en posteriores solicitudes, aumentando el rendimiento.

Es importante resaltar que las firmas electrónicas generadas durante el protocolo (POO, POR, NRO, NRR y NRA) deben ser generadas de acuerdo a estándares internacionales de firma [24]. De esta forma, se incluye una referencia a la política de firma empleada como atributo firmado en la firma electrónica, permitiendo al TTP conocer y obtener dicha política. Así mismo, este mecanismo impide a un atacante sustituir la política de firma referenciada.

3.5. Subprotocolo de Aborto

El subprotocolo de aborto permite al origen abortar el protocolo en caso que detecte que una transacción no deseada se ha iniciado en su nombre, si sospecha del comportamiento del receptor u ocurre algún error durante el protocolo que impida su completitud. Este subprotocolo se compone de los siguientes pasos:

- (1) $O_{ECF|ECT} \rightarrow TTP : abort, \ell, S_O(abort, \ell|SP)$
Si el protocolo ha sido recuperado por el receptor, entonces
- (2a) $TTP \rightarrow O_{ECF|ECT} : NRA_{TTP}$
en caso contrario
- (2b) $TTP \leftarrow TTP-SP : SP$
- (3b) $TTP \rightarrow O_{ECF|ECT} : S_{TTP}(S_O(abort, \ell|SP)|SP)$

Por cuestiones de eficiencia, $S_{TTP}(S_O(abort, \ell|SP)|SP)$ sólo se genera la primera vez (2b) y (3b), siendo reutilizado en posteriores ocasiones. Por otra parte, si el protocolo ha sido recuperado por el receptor con anterioridad (2a), TTP simplemente reenvía al origen la copia local de NRA_{TTP} .

4. Análisis de Seguridad del Protocolo

La validación formal de protocolos de seguridad es de suma importancia, especialmente antes de que el mercado pueda tomarlos como referencia, e implementaciones de dichos protocolos se implanten en sistemas comerciales. La experiencia ha demostrado que una verificación heurística - por lo general siempre incompleta - y no formal de las propiedades de seguridad que un protocolo dice cumplir suponen una incertidumbre en cuanto a su seguridad real. Numerosos protocolos actuales adolecen de vulnerabilidades descubiertas tras su estandarización. La causa es que una validación no formal siempre obvia vectores de ataque potenciales, permitiendo a un atacante aprovecharse de vulnerabilidades no descubiertas durante el diseño del protocolo.

Los métodos de razonamiento automático pueden aplicarse para analizar y verificar formalmente las propiedades de seguridad establecidas en la especificación de un protocolo. Estos métodos generalmente buscan un contra-ejemplo a lo estipulado en la especificación. En caso de no encontrarse, se puede considerar que la especificación del protocolo implementa las propiedades establecidas.

El protocolo propuesto en la sección 3 se ha validado formalmente empleando las herramientas de validación automática AVISPA (*Automated Validation of Internet Security Protocols and Applications*) y SPAN (*Security Protocol Animator for AVISPA*).

AVISPA [4,6] proporciona un conjunto de aplicaciones para la construcción y análisis formal de modelos de protocolos de seguridad. AVISPA incorpora cuatro motores de razonamiento automático: OFMC (*On the-Fly Model-Checker*) [8], CL-AtSe (*Constraint-Logic-based model-checker*) [44], SATMC (*SAT-based Model-Checker*) [5], y TA4SP (*Tree Automata based Automatic Approximations*

for the Analysis of Security Protocols) [9]. Por otra parte, SPAN [23] proporciona una interfaz gráfica de usuario que permite al diseñador del protocolo interactuar con las capacidades de AVISPA de forma sencilla.

Estas herramientas necesitan la especificación del protocolo para poder analizarlo. AVISPA permite especificar la dinámica del protocolo así como las propiedades de seguridad que se deseen verificar empleando el lenguaje de especificación de protocolos HPSL (*High Level Protocol Specification Language*) [7,10].

Así mismo, para poder analizar las propiedades de seguridad es necesario establecer el modelo de atacante que se tomará como referencia. AVISPA utiliza el modelo estándar de intruso de Dolev-Yao (DY) [15], en el cual el atacante tiene absoluto control sobre la red pero no puede realizar criptoanálisis. En este sentido el atacante puede intercambiar, reenviar, reutilizar o eliminar cualquier mensaje intercambiado entre los participantes.

En [27] se detalla el procedimiento seguido para validar el protocolo propuesto en la sección 3 empleando AVISPA y SPAN, así como los resultados obtenidos. No obstante, a continuación se describe brevemente las conclusiones más importantes de dicha validación.

La metodología que se ha seguido se resume en tres etapas:

1. Especificación del protocolo mediante HPSL.
2. Verificación de la corrección de la especificación.
3. Finalmente, la validación de la seguridad del protocolo.

De la etapa de especificación HPSL cabe destacar la definición de las propiedades de seguridad a verificar. AVISPA soporta tres tipos de objetivos o propiedades de seguridad: secreto, autenticación fuerte y autenticación débil. Ambos objetivos de autenticación aseguran la integridad del mensaje autenticado. La diferencia entre los dos tipos de autenticación es que una autenticación débil no protege frente a ataques por replicación. La propiedad estipulada para el protocolo es la autenticación débil de los participantes respecto a todas las firmas electrónicas generadas (POO, POR, NRO, NRR, NRA). Es importante mencionar que AVISPA no soporta de forma explícita la propiedad de justeza o equidad, por lo que, en esta primera aproximación, no ha podido verificarse dicha propiedad del protocolo.

Antes de verificar la seguridad del protocolo en base a su especificación, es importante comprobar que ésta efectivamente implementa el comportamiento esperado, especialmente si se ha empleado un lenguaje de bajo nivel como HPSL. Para ello se han ejecutado diversas utilidades ofrecidas por AVISPA y SPAN, garantizando la corrección sintáctica y cumplimiento funcional de la especificación.

Por último, en la etapa de validación se han definido múltiples escenarios de análisis que han sido utilizados por los motores OFMC y CL-AtSe para verificar las propiedades de seguridad indicadas. El resultado obtenido ha sido satisfactorio en todos los casos, lo cual ofrece las garantías buscadas en cuanto a la seguridad del protocolo. Los motores SATMC y TA4SP no pudieron emplearse debido a ciertas limitaciones actuales de la herramienta, y que les impedían analizar un protocolo de las características del propuesto.

5. Conclusiones y Trabajo Futuro

En este artículo se ha presentado un protocolo novedoso que permite mejorar de forma considerable la fiabilidad de la evidencia generada durante una transacción gobernada por un protocolo de intercambio justo. Esta evidencia se compone de múltiples firmas electrónicas, las cuales deben cumplir los requisitos impuestos por una política de firma determinada. El diseño del protocolo obliga al origen de la transacción a emplear dos entornos de creación de firmas diferentes, con diferentes claves privadas de firma en cada uno de ellos. Así mismo, la información intercambiada en la transacción debe ceñirse a una plantilla definida por el receptor, restringiendo la semántica de los datos.

De esta forma, la mayoría de los ataques conocidos no pueden violar la seguridad del protocolo, al necesitar comprometer no sólo ambos entornos sino también ejecutar un ataque colaborativo que permita generar evidencias de forma fraudulenta. La probabilidad de este tipo de ataques es sustancialmente menor que la probabilidad que tienen los actuales sistemas de firma implantados en el mercado de sufrir cualquiera de los ataques existentes.

Por otra parte, se ha presentado los principales resultados de un trabajo previo en el cual se verificó formalmente el protocolo propuesto. La verificación se llevó a cabo mediante herramientas de razonamiento automático. Teniendo en cuenta los resultados del análisis, podemos garantizar que las propiedades de seguridad estipuladas en el diseño del protocolo efectivamente se cumplen.

El objetivo inmediato es completar la verificación formal incluyendo la propiedad de equidad. Para ello se deberá especificar dicha propiedad mediante fórmulas objetivo y predicados especiales. Por otra parte, se va a proponer al grupo de trabajo de PKIX de IETF la redacción de una RFC Experimental que recoja el framework completo de políticas de firma extendidas mencionado en el artículo.

Agradecimientos. Este trabajo ha sido parcialmente realizado en el marco del proyecto SEGUR@, subvencionado por CDTI, Ministerio de Industria, Turismo y Comercio de España, dentro del programa CENIT, con referencia CENIT-2007 2004. (<https://www.cenitsegura.es>)

Referencias

1. Adams, C., Cain, P., Pinkas, D., Zuccherato, R.: RFC 3161 - Internet X.509 Public Key Infrastructure Time-Stamp Protocol (TSP). Internet Engineering Task Force (2001)
2. Adjunct Law Prof Blog - A Member of the Law Professor Blogs Network: Employer did not show electronic signature on arbitration pact was valid. Blog, 3rd March 2009, <http://lawprofessors.typepad.com/adjunctprofs/2009/03/employer-did-no.html>
3. Alsaid, A., Mitchel, J. C.: Dynamic content attacks on digital signatures. Information Management & Computer Security 13 (4), 328 - 336 (2005)

4. Armando, A., Basin, D., Boichut, Y., Chevalier, Y., Compagna, L., Cuellar, J., Hankes Drielsma P., Heám, P.-C., Kouchnarenko, O., Mantovani, J., Mödersheim, S., von Oheimb, D., Rusinowitch, M., Santos Santiago, J., Turuani, M., Viganò, L., and Vigneron, L.: The AVISPA Tool for the automated validation of internet security protocols and applications. In K. Etessami and S. Rajamani, editors, 17th International Conference on Computer Aided Verification, CAV'2005, volume 3576 of Lecture Notes in Computer Science, pages 281–285, Edinburgh, Scotland. Springer (2005)
5. Armando, A., and Compagna, L.: SATMC: a SAT-based model checker for security protocols. In Proc. of the 9th European Conference on Logics in Artificial Intelligence (JELIA'04), LNAI, 3229, 730–733, Lisbon, Portugal. Springer-Verlag (2004)
6. AVISPA: Automated validation of internet security protocols and applications (2003) FET Open Project IST-2001-39252. <http://www.avispa-project.org/>
7. AVISPA. Deliverable 2.1: The High-Level Protocol Specification Language. Available at <http://www.avispa-project.org/publications.html> (2003)
8. Basin, D. A., Sebastian, M., and Viganò, L.: Ofmc: A symbolic model checker for security protocols. *Int. J. Inf. Sec.*, 4(3):181–208 (2005)
9. Boichut, Y., Heam, P.-C., Kouchnarenko, O., and Oehl, F.: Improvements on the Genet and Klay Technique to Automatically Verify Security Protocols. In Proc. of Automated Verification of Infinite States Systems (AVIS'04), 1 - 11. ENTCS (2004)
10. Chevalier, Y., Compagna, L., Cuellar, J., Hankes Drielsma, P., Mantovani, J., S. Mödersheim, and Vigneron, L.: A High Level Protocol Specification Language for Industrial Security-Sensitive Protocols. In Proc. SAPS'04. Austrian Computer Society, 2004.
11. Cooper, D., Santesson, S., Farrel, S., Boeyen, S., Housley, R., Polk, W.: RFC 5280 - Internet X.509 Public Key Infrastructure. Certificate and Certificate Revocation List (CRL) Profile. Internet Engineering Task Force (2008)
12. Dasgupta, P., Chatha, K., Gupta, S. K. S.: Vulnerabilities of PKI based Smart-cards. Proceedings of the IEEE Military Communications Conference 2007 (MIL-COM 2007), 1-5. ISBN:978-1-4244-1513-7 (2007)
13. Department of Justice. Government of Canada. Personal Information Protection and Electronic Documents Act. April 13th (2000)
14. Diffie, W., Hellman, M.: New Directions in Cryptography. *IEEE Transactions of Information Theory*, 22 (6), 644-654. ISSN: 0018-9448 (1976)
15. Dolev, D., and Yao, A.: On the Security of Public-Key Protocols. *IEEE Transactions on Information Theory*, 2(29) (1983)
16. European Directive 1999/93/CE of the European Parliament and of the Council of 13 December 1999 on a Community framework for electronic signatures (1999)
17. European Telecommunications Standards Institute: ETSI TR 102 038 v1.1.1 - TC Security - Electronic Signatures and Infrastructures (ESI); XML format for signature policies (2002)
18. European Telecommunications Standards Institute: ETSI TR 102 041 v1.1.1 - Signatures policies report (2002)
19. European Telecommunications Standards Institute: ETSI TR 102 045 v1.1.1 - Electronic signatures and infrastructures (ESI); Signature policy for extended business model (2003)
20. European Telecommunications Standards Institute: ETSI TR 102 272 v1.1.1 - Electronic Signatures and Infrastructures (ESI); ASN.1 format for signature policies (2003)

21. Federal Trade Commission, Department of Commerce. United States of America: Electronic Signatures in Global and National Commerce Act. Available at www.senate.gov/search/index.html under S.761 in the 106th Congress (2000)
22. Girard, P., Giraud, J-L.: Software attacks on smart cards. Information Security Technical Report, 8 (1), 55 - 66 (2003)
23. Glouche, Y., Genet, T., Heen, O., and Courtay, O.: A Security Protocol Animator Tool for AVISPA. In ARTIST2 Workshop on Security Specification and Verification of Embedded Systems, Pisa (2006)
24. Hernandez-Ardieta, J. L., Gonzalez-Tablas, A. I., Alvarez, B. R.: An Optimistic Fair Exchange Protocol based on Signature Policies. Computers & Security, 27 (7-8), 309 - 322. ISSN 0167-4048. Ed. Elsevier (2008)
25. Hernández-Ardieta, J. L., González-Tablas, A. I., Ramos, B.: Repudio de firmas electrónicas en Infraestructuras de Clave Pública. Actas de la X Reunión sobre Criptología y Seguridad Informática (X RECSI 2008). Salamanca (2008)
26. Hernandez-Ardieta, J. L., Gonzalez-Tablas, A. I., Ramos, B., Ribagorda, A.: Extended Electronic Signature Policies. 2nd ACM International Conference on Security of Information and Networks (SIN 2009). ACM Press. North Cyprus (2009)
27. Hernandez-Ardieta, J. L., Gonzalez-Tablas, A. I., Ramos, B.: Formal Validation of OFEPSP+ with AVISPA. Joint Workshop on Automated Reasoning for Security Protocol Analysis and Issues in the Theory of Security. Lecture Notes in Computer Science, Springer-Verlag. York (2009)
28. Hernandez-Ardieta, J. L., Gonzalez-Tablas, A. I., Ramos, B., Ribagorda, A.: On the Need to Divide the Signature Creation Environment. International Conference on Security and Cryptography (SECRYPT 2009). Milan (2009)
29. ISO/IEC DIS 13888-1. Information technology - Security techniques - Non repudiation - Part 1: General model. ISO/IEC JTC1/SC27 N1503 (1996)
30. ISO/IEC 13888-3 Information technology - Security techniques - Non repudiation - Part 3: Mechanisms Using Asymmetric Techniques. ISO/IEC (1997)
31. Jsang, A., Povey, D., Ho, A.: What You See is Not Always What You Sign. In the proceedings of the Australian UNIX User Group. Melbourne (2002)
32. Kain, K.: Electronic Documents and Digital Signatures. Master Thesis (2003)
33. Kremer, S., Markowitch, O., Zhou, J.: An intensive survey of fair non-repudiation protocols. Computer Communications 25, 1601 - 1621 (2002)
34. Marchesini, J., Smith, S.W., Zhao, M.: Keyjacking: the surprising insecurity of client-side SSL. Computers & Security, 24 (2), 109-123 (2005)
35. McCullagh, A., Caelli, W.: Non-repudiation in the digital Environment. First Monday, 5 (8). Available at <http://firstmonday.org/issues/issue5.8/mccullagh/index.html> (2000)
36. Organization for the Advancement of Structured Information Standards: ebXML Business Process Specification Schema Technical Specification v2.0.4 (2006)
37. Organization for the Advancement of Structured Information Standards: Universal Business Language (UBL) v2.0 (2008)
38. Rivest, R. L., Shamir, A., Adleman, L.: A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM, 21 (2), 120-126. ISSN:0001-0782 (1978)
39. Ross, J., Pinkas, D., Pope, N.: RFC 3125 - Electronic Signature Policies. International Engineering Task Force (2001)
40. Scheibelhofer, K.: What You See Is What You Sign - Trustworthy Display of XML Documents for Signing and Verification. Proceedings of the IFIP TC6/TC11 International Conference on Communications and Multimedia Security Issues of the New Century, 192, ISBN:0-7923-7365-0 (2001)

41. Spalka, A., Cremers, A. B., Langweg, H.: Protecting the Creation of Digital Signatures with Trusted Computing Platform Technology Against Attacks by Trojan Horse Programs. Proceedings of IFIP TC11 16th International Conference on Information Security. Kluwer, Boston (2001)
42. Spalka, A., Cremers, A. B., Langweg, H.: Trojan Horse Attacks on Software for Electronic Signatures. *Informatica* 26 (2), 191 - 203 (2002)
43. Tiri, K.: Side-channel Attack Pitfalls. Proceedings of the 44th Annual ACM IEEE Design Automation Conference, 15-20. ISBN:0738-100X (2007)
44. Turuani, M.: The CL-Atse Protocol Analyser. In F. Pfenning, editor, Proc. of 17th International Conference on Rewriting Techniques and Applications, RTA, Lecture Notes in Computer Science, Seattle (WA), Aug. Springer (2006)
45. United Nations. UNCITRAL Model Law on Electronic Signatures with Guide to Enactment (2001)

Análisis Formal del Estándar NIST para Modelos RBAC

Carlos D. Luna¹ and Cristian D. Rosa²

¹ Instituto de Computación, Universidad de la República, Uruguay
cluna@fing.edu.uy

² FCEIA, Universidad Nacional de Rosario, Argentina
crosa@fceia.unr.edu.ar

Institución Actual: LORIA-INRIA, Equipe AlGorille, Batiment B, BP 239, 54506
Vandoeuvre-lès-Nancy, Cedex, France

Resumen *Role Based Access Control* (RBAC) es un enfoque basado en el concepto de rol para implementar políticas de control de acceso. Actualmente es considerado uno de los modelos más generales, debido a su neutralidad respecto a las políticas de control de acceso y a su flexibilidad. A causa de la existencia de múltiples variantes de modelos e implementaciones RBAC, el National Institute of Standards and Technology (NIST) realizó una propuesta de estandarización, definiendo un modelo de referencia. Si bien la especificación se desarrolla en un lenguaje de especificación formal (en lenguaje Z), la propuesta no incluye un análisis de su corrección, ni la verificación de propiedades de seguridad. Este artículo presenta un análisis formal de los modelos básico (*Core RBAC*) y jerárquico (*Hierarchical RBAC*) del estándar NIST, que pone en evidencia la existencia de imprecisiones, omisiones y errores en la especificación de referencia. A partir de este análisis el trabajo introduce una nueva especificación formal de *Core RBAC* y *Hierarchical RBAC* que subsana los problemas encontrados. En torno a la nueva formalización, desarrollada en el Cálculo de Construcciones Inductivas, se demuestran algunas propiedades relevantes de seguridad inherentes a los modelos RBAC, usando el asistente de pruebas Coq.

Palabras clave: Control de Acceso Basado en Roles, Especificación y Verificación Formal, Teoría de Tipos, Coq.

1. Introducción

El concepto del rol como entidad central en el control de acceso se origina con los sistemas multiusuarios, en la década del 70 [11]. La utilización de grupos de usuarios en UNIX, entre otros sistemas, y el agrupamiento por privilegios en sistemas de administración de bases de datos son las raíces de los sistemas actuales [2, 13, 14]. *Role Based Access Control* (RBAC) es un enfoque basado en el concepto de rol para implementar políticas de control de acceso. Actualmente es considerado uno de los modelos más generales, debido a su neutralidad respecto a las políticas de control de acceso y a su flexibilidad, que le permiten simular

enfoques alternativos como DAC y MAC. Los modelos RBAC han demostrado su efectividad en ámbitos muy variados, como en organizaciones gubernamentales [5] y en complejas aplicaciones web [6]. En RBAC los permisos son asociados con roles y los usuarios son asignados a roles apropiados, esto simplifica en gran medida la administración de los permisos. Los roles son creados según responsabilidades y niveles de autoridad en el contexto de una organización. Un rol es la construcción semántica sobre la cual se formulan las políticas de control de acceso. La colección de usuarios y permisos vinculados por un rol es transitoria.

Los modelos e implementaciones RBAC existentes son relativamente similares en los conceptos fundamentales, pero difieren, significativamente en algunos casos, en los detalles. Las diferencias no siempre resultan obvias, debido a que suelen utilizarse diferentes terminologías para describir los mismos conceptos. Para dar solución a problemas de alcance y terminología, el National Institute of Standards and Technology (NIST) realizó una propuesta de estandarización que unifica las implementaciones comerciales más relevantes y los principales modelos desarrollados en ámbitos académicos [4, 10]. El estándar referido, adoptado en 2004 como un estándar ANSI/INCITS, propone un modelo de referencia, que define la terminología general e introduce los elementos básicos de RBAC, y una especificación funcional en lenguaje Z, que describe los requerimientos de los comandos utilizados para interactuar con el sistema.

Este artículo presenta un análisis formal de los modelos básico (*Core RBAC*) y jerárquico (*Hierarchical RBAC*) del estándar NIST, que pone en evidencia la existencia de imprecisiones, omisiones y errores en la especificación propuesta, en mayor medida debidos a decisiones libradas a criterios de implementación. A partir de este análisis el trabajo introduce una nueva especificación formal de los modelos RBAC considerados que subsana los problemas encontrados. En torno a la nueva formalización, desarrollada en el Cálculo de Construcciones Inductivas (CCI), se demuestran importantes propiedades de seguridad inherentes a los modelos RBAC, usando el asistente de pruebas Coq [3, 12]. Si bien existen trabajos previos que analizan el estándar NIST referido, tales como [1, 7, 8], los mismos no plantean una nueva especificación ni verifican formalmente propiedades de seguridad del modelo. Asimismo, el presente trabajo, a diferencia de los anteriores, utiliza un lenguaje (una teoría de tipos) que permite derivar especificaciones funcionales correctas por construcción. En particular, como ventaja adicional de este enfoque, un modelo ejecutable podría utilizarse en una técnica complementaria de la verificación formal, como es el testing de caja negra, donde oficie de oráculo frente a casos de pruebas derivados de la especificación.

Una versión completa de este trabajo es el reporte [9]. La nueva especificación formal introducida en este artículo, incluyendo las propiedades probadas usando Coq, están disponibles en <http://www.fceia.unr.edu.ar/~crosa>.

El resto de este artículo está organizado como sigue. La sección 2 introduce el modelo RBAC propuesto por NIST. En la sección 3 se presenta un análisis de las decisiones de diseño libradas a criterios de implementación por parte del estándar y se justifican las elecciones tomadas en este trabajo, que dan lugar a una nueva especificación formal de *Core RBAC* y *Hierarchical RBAC*. La sección 4 describe

algunos errores encontrados en la especificación del estándar NIST. Luego, en la sección 5 se introducen aspectos relevantes de la nueva formalización desarrollada en el CCI y se mencionan algunas propiedades verificadas. Finalmente, la sección 6 exhibe las conclusiones y los trabajos futuros.

2. Modelo RBAC

Esta sección presenta el modelo de referencia RBAC propuesto por NIST [4]. Primero se introducen generalidades de RBAC, su organización modular y las funcionalidades provistas por cada componente. Posteriormente se describen, con mayor detalle, los componentes que serán analizados en este trabajo.

2.1. Descripción General del Modelo

El modelo de referencia NIST y su especificación funcional están divididos en cuatro componentes o modelos: *Core RBAC*, *Hierarchical RBAC*, *Static Separation of Duty Relations* (SSD) y *Dynamic Separation of Duty Relations* (DSD). Cada uno de estos componentes extiende la funcionalidad del componente anterior.

Core RBAC abarca los aspectos esenciales de RBAC. Dentro de los conceptos básicos de RBAC tenemos que: los usuarios son asignados a roles, los permisos son asociados a roles y los usuarios adquieren permisos, siendo miembros de roles. Las asignaciones usuario-rol y permiso-rol permiten que un usuario pertenezca a varios roles simultáneamente y que un rol posea múltiples usuarios. Asimismo, un permiso puede ser asociado a varios roles y un rol puede tener asociados múltiples permisos. *Core RBAC* incluye también el concepto de sesión, que permite la activación y desactivación selectiva de roles, con el objetivo de minimizar el número de permisos ejercidos simultáneamente, limitándose a los necesarios para cada operación. Las decisiones de acceso se realizan sobre el conjunto de permisos asociados con los roles activos de cada sesión.

Hierarchical RBAC agrega al modelo anterior los elementos necesarios para el soporte de jerarquías de roles. Una jerarquía es un orden parcial en el conjunto de roles que define una relación de herencia entre éstos. Los roles ascendientes en la jerarquía adquieren los permisos de sus descendientes y los roles descendientes adquieren los usuarios pertenecientes a sus ascendientes.

Los componentes SSD y DSD definen relaciones que se utilizan para hacer cumplir políticas de conflicto de intereses. Un conflicto de interés puede darse cuando un usuario obtiene autorización para ejercer permisos asociados con roles conflictivos. Por ejemplo, si un usuario tiene asignados un rol que le permite gestionar una compra y un rol que lo autoriza a aprobar dicha operación. Ambos componentes permiten especificar políticas de conflicto de intereses de manera centralizada y luego imponerlas uniformemente en el sistema. La diferencia entre SSD y DSD reside en el contexto en el cual la política es impuesta. Mientras que SSD restringe las asignaciones usuario-rol, DSD limita la disponibilidad de permisos, imponiendo restricciones en los roles que pueden ser activados simultáneamente en las sesiones.

2.2. Core RBAC

Como se mencionó previamente, el modelo RBAC está esencialmente definido en términos de usuarios individuales asignados a roles y de permisos asociados a roles. Se puede considerar al rol como la manera de describir relaciones “muchos a muchos” entre usuarios individuales y permisos. Por razones de simplicidad, un usuario está definido, en general, como un ser humano. Un rol es un trabajo o función, en el contexto de una organización, con alguna semántica relacionada a la autoridad y responsabilidad conferida a los usuarios asignados a dicho rol. Un permiso es una autorización para realizar una operación sobre un objeto determinado. Una operación es una acción que, al ser invocada, ejecuta una tarea para un usuario. En tanto que un objeto es una identidad que contiene o recibe información.

Core RBAC define cinco conjuntos de elementos de datos básicos: usuarios (USERS), roles (ROLES), objetos (OBS), operaciones (OPS) y permisos (PRMS). La figura 1 esquematiza la relación que existe entre los elementos que componen el modelo (la relación *Role Hierarchy* se describe en la sección 2.3).

Para que un usuario pueda ejercer los permisos para los cuales está autorizado, es necesario que éste active los roles que tienen asociados tales permisos. Para tal fin, *Core RBAC* define un conjunto de sesiones (SESSIONS), donde cada sesión es un mapeo entre un usuario y un subconjunto de roles activos. Cuando una sesión es creada debe proveerse un conjunto inicial de roles activos (posiblemente vacío), que posteriormente puede ser actualizado por el sistema. La utilización de sesiones incrementa la granularidad del control de acceso a los recursos y fortalece la aplicación del *principio del menor privilegio*. Las decisiones relacionadas a control de acceso son tomadas por usuario y sesión. Es decir, un usuario está autorizado para realizar una determinada operación sobre cierto objeto en una sesión, si existe algún rol activo en dicha sesión que tiene asignado el permiso correspondiente.

Core RBAC especifica los requerimientos funcionales de los comandos utilizados para administrar los elementos básicos del modelo. Estos comandos contemplan la creación y eliminación de usuarios, roles y permisos, la asignación de usuarios a roles y la asignación de permisos a roles. Asimismo, el modelo básico incluye comandos para instrumentar el control de acceso, realizar la gestión de sesiones e inspeccionar el estado del sistema.

2.3. Hierarchical RBAC

Las jerarquías de roles son una manera natural de estructurar los roles con el objetivo de reflejar las líneas de autoridad y las responsabilidades en una organización. El componente *Hierarchical RBAC* incorpora al modelo básico el soporte para instrumentar jerarquías de roles. Para esto se agrega a *Core RBAC* una relación *RH* (*Role Hierarchy*) entre roles, que en la figura 1 se denota con el uso de líneas de puntos.

La relación de herencia se define en término de permisos y usuarios. Intuitivamente, que r_1 “herede de” r_2 tiene como consecuencia que todos los permisos

de r_2 son también permisos de r_1 y que todos los usuarios de r_1 son asimismo usuarios de r_2 . Formalmente, $RH \subseteq ROLES \times ROLES$ es un orden parcial sobre el conjunto $ROLES$, llamado relación de herencia, que se denota con el símbolo $\succeq$. Luego, se cumple que: $r_1 \succeq r_2 \Rightarrow perms(r_2) \subseteq perms(r_1)$ y $users(r_1) \subseteq users(r_2)$, donde $perms$ y $users$ tienen una semántica formal que resulta intuitiva en el contexto previo. *Hierarchical RBAC* define los requerimientos funcionales de los comandos de gestión e inspección de la jerarquía, y especifica cambios en los requerimientos de algunos comandos de *Core RBAC*, que deben contemplar la existencia de una jerarquía de roles.

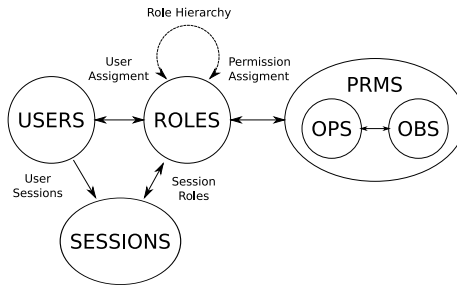


Figura 1. Elementos del Modelo RBAC

3. Análisis de las Decisiones de Diseño

El modelo RBAC propuesto por NIST, introducido en la sección 2, propone en múltiples partes a lo largo de su especificación decisiones de diseño que quedan libradas a criterios de implementación. A continuación incluimos un análisis de alternativas válidas para estas decisiones, según el estándar, que en ciertos casos dan lugar a potenciales fallas de seguridad, si no se toman recaudos apropiados. En cada caso planteamos restricciones a las decisiones de diseño, que serán consideradas luego en la formulación de una nueva especificación en la sección 5.

3.1. Operaciones sobre la Jerarquía de Roles

La jerarquía de roles está definida como un orden parcial sobre el conjunto de los roles. Esto es así ya que la transitividad y la antisimetría modelan de manera precisa el comportamiento de una relación de herencia.

Eliminación de Herencias Indirectas. Es una decisión de implementación permitir, o no, la eliminación de herencias indirectas, también llamadas implícitas o transitivas.

La figura 2 muestra una jerarquía de roles, donde $A \succeq B$, $B \succeq C$ y $B \succeq D$ se relacionan y representan de manera directa (explícita), y $A \succeq C$ y $A \succeq D$ se vinculan y representan indirectamente. Si se permitiese eliminar, por ejemplo,

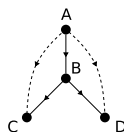


Figura 2. Jerarquía de Ejemplo

la herencia indirecta $A \succeq C$ se podría proceder de dos formas:

1. Eliminar $A \succeq C$, pero entonces se tendría que: $A \succeq B$ y $B \succeq C \not\Rightarrow A \succeq C$. Por lo tanto, $\succeq$ resultaría no transitiva y no sería un orden parcial. De esta manera se perderían las propiedades deseables de la jerarquía.
2. Para preservar el orden parcial se podría eliminar $B \succeq C$ o $A \succeq B$. Esto por un lado representa una elección no determinista y por el otro, la decisión que se tome podría llegar a afectar a roles intermedios, generándose de esta manera resultados impredecibles.

A partir del análisis previo, en este trabajo se considera que la eliminación de herencias debería restringirse únicamente a las que son de relación directa, es decir, de herencia inmediata.

Conservación de Herencias Indirectas. El estándar NIST considera que es una decisión de implementación la conservación o no de herencias indirectas inducidas por una herencia inmediata que se elimina. Por ejemplo, dada la jerarquía descrita en la figura 2, si se elimina $A \succeq B$, la conservación de $A \succeq C$ y $A \succeq D$ es opcional.

Existen dos alternativas para implementar esta funcionalidad, conservando las propiedades de orden parcial de $\succeq$. La primera consiste en exigir que todas las relaciones entre roles sean inmediatas. Para lograr esto, cada vez que se vinculan dos roles resulta necesario agregar de manera explícita todas las nuevas relaciones transitivas inducidas por dicho vínculo. Este enfoque subutiliza el hecho de que $\succeq$ es un orden parcial, ya que no se hace uso de la transitividad. La segunda alternativa es transformar en herencias inmediatas todas las relaciones inducidas por transitividad a partir de la relación eliminada. Este comportamiento complica el control de la jerarquía. Por ejemplo, si elimina una relación e inmediatamente se la vuelve a incorporar, no necesariamente la jerarquía resultante es igual a la inicial.

Como conclusión, en este trabajo se considera que las herencias indirectas relacionadas a través de un rol eliminado no deberían ser conservadas.

3.2. Manejo de Sesiones

Activación Explícita de Roles. Otra decisión librada a criterio de implementación se relaciona con la manera de proceder ante la creación de una sesión o la activación de un rol en una sesión existente, cuando se cuenta con una jerarquía de roles. El sistema puede activar sólo el rol o los roles indicados, o bien puede además activar automáticamente todos los roles heredados a través de éstos.

Si se tiene como objetivo preservar el *principio del menor privilegio*, es entonces muy importante que la implementación asegure que toda activación de roles sea explícita, ya que de esta manera se reduce el número de roles activos en todo momento.

Terminación de Sesiones al Eliminar un Usuario. Es una decisión de diseño establecer qué sucede con las sesiones de usuarios que son eliminados. El sistema puede terminar las sesiones inmediatamente o esperar a que éstas finalicen normalmente.

Si las sesiones de un usuario no se terminan al ser éste eliminado y no se modifica la semántica dada a algunos comandos, el sistema podría evolucionar a un estado vulnerable a una potencial falla de seguridad. Por ejemplo, considérese un estado del sistema donde se elimina cierto usuario y se preservan sus sesiones activas. Si posteriormente se crea el mismo usuario y se le asigna un conjunto de roles diferente, dicho usuario podría tener ahora sesiones con roles activos para los cuales posiblemente ya no esté autorizado. Esta consituye una grave falla de seguridad.

En este trabajo se considera que es necesario culminar de manera inmediata todas las sesiones de un usuario cuando éste es eliminado. De esta manera se evita cualquier inconsistencia entre el conjunto de sesiones y el conjunto de usuarios.

Terminación de Sesiones al Eliminar o Desasignar un Rol. Al igual que en el caso anterior, es también una decisión de implementación determinar qué sucede con las sesiones que tienen activo un rol, cuando éste es eliminado o desasignado a un usuario. Las opciones son: esperar hasta que las sesiones terminen normalmente, eliminar el rol de la lista de roles activos de las sesiones, o terminar inmediatamente dichas sesiones.

En el primer caso, el estado del sistema luego de desasignar un rol a un usuario no satisface la condición de integridad que debe existir entre los conjuntos de roles activos de las sesiones pertenecientes al usuario y el conjunto de los roles asignados al mismo. Esta falta de integridad implica una falla de seguridad, debido a que el usuario puede seguir ejerciendo permisos asociados a un rol para el cual ya no está autorizado. Adicionalmente, si se agregaran luego nuevos permisos a dicho rol, los mismos podrían ser ejercidos igualmente por el usuario.

Para este trabajo resultan válidas las dos últimas opciones, aunque en la propuesta de una nueva especificación se establece que las sesiones involucradas deberían ser finalizadas inmediatamente, en concordancia con la restricción formulada para el caso anterior.

4. Algunos Problemas de la Especificación

El estándar propuesto por NIST utiliza el lenguaje de especificación Z para formalizar los requerimientos funcionales de los comandos de interacción con el sistema. Si bien este lenguaje dota a la especificación de un grado de precisión y de claridad notable, un análisis riguroso de los requerimientos, resultado de aplicar herramientas de verificación, puso de manifiesto algunas impresiones e incluso contradicciones en la especificación de la semántica de varios comandos. A continuación se describen algunos de los problemas encontrados. Un análisis completo está disponible en [9].

Los inconvenientes detectados en la especificación están relacionados con el componente *Hierarchical RBAC*, debido especialmente a la forma en que está definida la relación de herencia entre roles. En la sección “A.2 Requirements for Hierarchical RBAC” del estándar se define la relación de herencia a través de dos relaciones, una de herencia inmediata entre roles ($\gg$) y otra que corresponde a la clausura reflexo-transitiva de la anterior ($\geq$).

Un problema reside en los esquemas Z de los comandos *AddInheritance* y *DeleteInheritance*, que asumen un vínculo entre las relaciones $\gg$ y $\geq$ que no está especificado [4]. *AddInheritance* está definido de la siguiente manera:

“...This command establishes a new immediate inheritance relationship $r_{asc} \gg r_{desc}$ between existing roles r_{asc}, r_{desc} ...”

$AddInheritance(r_{asc}, r_{desc} : NAME) \triangleleft$
 $r_{asc} \in ROLES; r_{desc} \in ROLES; \neg(r_{asc} \gg r_{desc}); \neg(r_{desc} \geq r_{asc})$
 $\geq' = \geq \cup \{r, q : ROLES \mid r \geq r_{asc} \wedge r_{desc} \geq q \bullet r \mapsto q\} \triangleright$

Claramente se observa que la poscondición del esquema no actualiza la relación $\gg$. Esto es, la especificación formal no coincide con la descripción que acompaña al esquema. Algo similar sucede con el comando *DeleteInheritance*, que tiene por definición y esquema:

“...This command deletes an immediate inheritance relationship $r_{asc} \gg r_{desc}$... in this definition, implied relationships are preserved after deletion...”

$DeleteInheritance(r_{asc}, r_{desc} : NAME) \triangleleft$
 $r_{asc} \in ROLES; r_{desc} \in ROLES; r_{asc} \ll r_{desc}$
 $\geq' = (\gg \setminus \{r_{asc} \mapsto r_{desc}\})^* \triangleright$

Nuevamente la poscondición no especifica que sucede con la relación $\gg$. En el esquema anterior existe también otro problema. La descripción informal del comando aclara que las relaciones implícitas (transitivas) son preservadas, pero la semántica de *DeleteInheritance* no expresa esto. A fin de ejemplificar esta última observación, considérense las siguientes relaciones:

$\gg = \{A \mapsto r_{asc}, r_{asc} \mapsto r_{desc}, r_{desc} \mapsto B\}$
 $\geq = \gg \cup \{A \mapsto r_{desc}, r_{asc} \mapsto B, A \mapsto B\}$

Si se ejecuta el comando *DeleteInheritance* r_asc r_desc , según su especificación, se tiene:

$$\begin{aligned}\geq' &= (\gg \setminus \{r\_asc \mapsto r\_desc\})^* \\ &= (\{A \mapsto r\_asc, r\_desc \mapsto B\})^* \\ &= \{A \mapsto r\_asc, r\_desc \mapsto B\}\end{aligned}$$

donde en la nueva relación $\geq'$ se perdieron las herencias implícitas.

Continuando con los comandos relacionados con la jerarquía de roles, los esquemas que especifican la semántica de *AddAscendant* y *AddDescendant* (que permiten agregar herencias inmediatas creando uno de los roles involucrados) poseen una contradicción en sus precondiciones. *AddAscendant* está especificado como sigue,

$$\begin{aligned}&AddAscendant(r\_asc, r\_desc : NAME) \triangleleft \\ &AddRole(r\_asc) \\ &AddInheritance(r\_asc, r\_desc) \triangleright\end{aligned}$$

done *AddRole* permite agregar un rol en el sistema. Si se expanden los esquemas incluidos en la especificación, se obtiene como parte de la precondición la siguiente contradicción:

$$\begin{aligned}&AddAscendant(r\_asc, r\_desc : NAME) \triangleleft \\ &r\_asc \notin ROLES; \\ &r\_asc \in ROLES; \dots \triangleright\end{aligned}$$

Todo parece indicar que el origen de dicha contradicción proviene de asumir un concepto de *secuencialidad* entre los estados de los esquemas incluidos, que es inexistente en $\mathbb{Z}$.

Por último y también relacionando con una funcionalidad incluida en *Hierarchical RBAC*, el estándar especifica que al eliminar un rol la jerarquía de roles no se modifica. Si bien esto no es afirmado explícitamente, al comienzo de la sección A.2a.1 se expresa que: “All functions of Section A.1.1 remain valid. In addition, this section defines the following new specific functions”. La sección A.1.1 incluye la definición de *DeleteRole*, que por lo tanto no cambia en presencia de la jerarquía de roles. No obstante, esto se puede transformar en una potencial falla de seguridad, ya que el sistema podría transicionar a un estado en donde no haya integridad entre los roles en la jerarquía y el conjunto de roles del sistema. Por ejemplo, se pueden activar todos los roles heredados a través de un rol eliminado, a pesar de que éste ya no exista. Peor aún, si este rol es nuevamente creado en el sistema, estaría relacionado con sus antiguos herederos, agregando un comportamiento no especificado sobre el comando *AddRole*.

En este trabajo se considera que deberían eliminarse todas las relaciones de la jerarquía de roles en las cuales un rol eliminado esté involucrado.

5. Formalización en el CCI

Esta sección introduce una nueva especificación de los modelos *Core RBAC* y *Hierarchical RBAC*. La formalización, desarrollada en el CCI, respeta el estándar NIST bajo las consideraciones planteadas en las secciones 3 y 4. Por razones de espacio muchos detalles serán omitidos. El lector interesado puede referirse a [9] por más información.

5.1. Notación

Para los conectivos lógicos y la igualdad se utiliza la notación estándar ($\wedge, \vee, \rightarrow, \neg, \exists, \forall, =$). Un tipo registro se define como sigue,

$$R \stackrel{\text{def}}{=} \llbracket \text{field}_0 : A_0, \dots, \text{field}_n : A_n \rrbracket$$

donde R es un tipo inductivo no recursivo con un único constructor y n funciones de proyección $\text{field}_i : R \rightarrow A_i$.

La notación para conjuntos de tipo A es *set* A . El tipo inductivo *set* codifica a los conjuntos como listas. Para los constructores de tipos inductivos de la forma

$$I : t_0 \dots t_n \stackrel{\text{def}}{=} C_i (a_0 : k_0) \dots (a_j : k_j) : I (b_0 : t_0) \dots (b_n : t_n) \quad \forall i, j, n \in \mathbb{N}_0$$

se usa alternativamente la notación:

$$C_i \frac{a_0 \dots a_j}{I b_0 \dots b_n}$$

donde las variables libres se consideran cuantificadas universalmente y sus tipos definidos por el contexto.

5.2. Formalización de Componentes

Preludio. Primero se introducen algunos tipos que serán utilizados en el resto de la especificación.

User, *Role*, *Op*, *Ob* y *Session* representan los tipos de los usuarios, los roles, las operaciones, los objetos y las sesiones, respectivamente. Estos tipos constituyen elementos básicos del modelo RBAC.

Un permiso describe la posibilidad de realizar una operación sobre un objeto determinado. Formalmente es un elemento del siguiente tipo registro:

$$\text{Perm} \stackrel{\text{def}}{=} \llbracket \text{op}_0 : \text{Op}, \text{ob}_0 : \text{Ob} \rrbracket$$

donde op_0 representa una operación sobre el objeto ob_0 .

Los permisos pueden estar asociados a múltiples roles. Estas asociaciones están definidas por el tipo:

$$\text{PA}_{\text{elem}} \stackrel{\text{def}}{=} \llbracket r_1 : \text{Role}, p_1 : \text{Perm} \rrbracket$$

De manera análoga, los usuarios pueden estar asignados a uno o más roles. El siguiente tipo registro formaliza esta relación:

$$\text{UA}_{\text{elem}} \stackrel{\text{def}}{=} \llbracket u_0 : \text{User}, r_0 : \text{Role} \rrbracket$$

Jerarquía de Roles. La relación de jerarquía entre roles es un orden parcial sobre el conjunto de roles del sistema. Para simplificar la manipulación de esta jerarquía se consideran dos relaciones: una de herencia inmediata o directa,

$$(\ll) : Role \rightarrow Role \rightarrow Prop$$

y su respectiva clausura reflexo-transitiva, definida inductivamente como sigue:

$$(\preceq) : Role \rightarrow Role \rightarrow Prop$$

$$\text{rt.refl} \frac{}{x \preceq x} \quad \text{rt.step} \frac{x \ll y}{x \preceq y} \quad \text{rt.trans} \frac{x \preceq y \quad y \preceq z}{x \preceq z}$$

La jerarquía de roles está vinculada con los usuarios del sistema a través de la relación *authorizedUser*. Un usuario está autorizado para un rol si éste está asignado directamente a él o bien si está asignado a un rol que hereda del mismo. Formalmente,

$$\text{authorizedUser} (u:User) (r:Role) \stackrel{\text{def}}{=} \exists r':Role, \text{role\_of\_user } r' u \wedge r \preceq r'$$

donde *role_of_user* es el predicado característico para la asignación de roles a usuarios.

Sesiones. De acuerdo a la descripción dada en la sección 2.3, una sesión es un mapeo entre los usuarios y sus roles activos. Un usuario puede tener más de una sesión establecida de manera simultánea. El estado de las sesiones del sistema está dividido en dos partes. La primera es un conjunto de identificadores válidos de sesión de tipo *Session*. La segunda parte es una función que mapea cada sesión con el usuario dueño de la misma y con el conjunto de sus roles activos, esto es, $se : Session \rightarrow SE_{elem}$, donde SE_{elem} es el tipo:

$$SE_{elem} \stackrel{\text{def}}{=} \llbracket \text{usr}:User, \text{rl}:set \text{ Role} \rrbracket$$

siendo el campo *usr* el usuario dueño de la sesión y *rl* el conjunto de roles activos de la misma.

Estado. El estado del sistema se describe a través del siguiente tipo registro:

$$\begin{aligned} State \stackrel{\text{def}}{=} \llbracket & ob:set \text{ Ob}, op:set \text{ Op}, p:set \text{ Perm}, u:set \text{ User}, r:set \text{ Role}, \\ & ua:set \text{ UA}_{elem}, pa:set \text{ PA}_{elem}, (\ll):Role \rightarrow Role \rightarrow Prop, \\ & seID:set \text{ Session}, se:Session \rightarrow SE_{elem} \rrbracket \end{aligned}$$

donde *ob* es el conjunto de objetos sobre el cual se pueden realizar operaciones; *op* es el conjunto de estas operaciones; *p* es el conjunto de permisos; *u* es el conjunto de usuarios válidos; *r* es el conjunto de roles; *ua* es el conjunto de asignaciones usuario-rol; *pa* es el conjunto de asignaciones permiso-rol; $\ll$ es la relación de herencia inmediata entre roles; *seID* es el conjunto de identificadores de sesión de la sesiones establecidas en el sistema; y *se* es la función, descrita previamente, que mapea cada identificador de sesión con su dueño y roles activos.

Propiedades de Validez de los Estados. No todo elemento de tipo *State* es un estado válido, para ello es necesario que sus componentes verifiquen ciertas condiciones. Para caracterizarlas formalmente se definen propiedades de validez, que todo estado RBAC $s : State$ debe cumplir. A continuación se detallan sólo algunas de las más relevantes.

“Toda sesión tiene un dueño, que además es miembro del conjunto de usuarios”

$$\begin{aligned} existsSessionOwner\ s &\stackrel{\text{def}}{=} \forall\ sid:Session, \\ isSession\ s\ sid &\rightarrow \exists\ u:User, isUser\ s\ u \wedge session\_owner\ s\ u\ sid \end{aligned}$$

“El usuario dueño de una sesión es único”

$$\begin{aligned} uniqueSessionOwner\ s &\stackrel{\text{def}}{=} \forall\ (sid:Session)(u_1\ u_2:User), isSession\ s\ sid \\ \wedge\ session\_owner\ s\ u_1\ sid \wedge session\_owner\ s\ u_2\ sid &\rightarrow u_1 = u_2 \end{aligned}$$

“Los roles activos de una sesión son un subconjunto del conjunto de roles para los que el usuario dueño de la misma está autorizado”

$$\begin{aligned} activeSessionRoles\ s &\stackrel{\text{def}}{=} \forall\ (sid:Session)(u:User)(r:Role), isSession\ s\ sid \\ \wedge\ session\_owner\ s\ u\ sid \wedge role\_of\_session\ s\ r\ sid &\rightarrow authorized\_user\ s\ u\ r \end{aligned}$$

“La relación entre roles ($\preceq$) que modela la jerarquía de roles es un orden parcial”

$$\begin{aligned} isOrder\ (\preceq) &\stackrel{\text{def}}{=} \forall\ x, x \preceq x \\ &\wedge \forall\ x\ y, x \preceq y \wedge y \preceq x \rightarrow x = y \\ &\wedge \forall\ x\ y\ z, x \preceq y \wedge y \preceq z \rightarrow x \preceq z \end{aligned}$$

“Los roles pertenecientes a la jerarquía de roles son un subconjunto del conjunto de roles del sistema”

$$H\_integrity\ s \stackrel{\text{def}}{=} \forall\ r_1\ r_2:Role, r_1 \ll r_2 \rightarrow isRole\ s\ r_1 \wedge isRole\ s\ r_2$$

Finalmente, un estado s es válido si cumple la condición *Valid* [9]:

$$\begin{aligned} Valid\ s &\stackrel{\text{def}}{=} existsSessionOwner\ s \wedge uniqueSessionOwner\ s \\ &\wedge activeSessionRoles\ s \wedge isOrder\ (\preceq)\ s \wedge UA\_integrity\ s \\ &\wedge Perm\_integrity\ s \wedge PA\_integrity\ s \wedge H\_integrity\ s \end{aligned}$$

5.3. Comandos

El estado del sistema evoluciona mediante la ejecución de comandos, cuya sintaxis está definida por el tipo inductivo no recursivo *Funcs*, y cuya su semántica está especificada mediante *pre* y *pos* condiciones. Para cada comando (más de 20 fueron considerados), su precondition es un predicado sobre el estado, y su

postcondición es un predicado que relaciona al estado previo y al posterior a la ejecución del mismo, con una posible respuesta del sistema (de tipo *Answer*). Ambas relaciones están definidas inductivamente, utilizando un constructor para cada comando. Por razones de espacio sólo se indican sus tipos a continuación:

$$Pre : State \rightarrow Funcs \rightarrow Prop$$

$$Pos : State \rightarrow State \rightarrow Answer \rightarrow Funcs \rightarrow Prop$$

Errores. Cuando se intenta ejecutar un comando cuya precondition no se satisface, el sistema responde con un código de error de tipo *ErrorCode*. Para cada comando f se define la relación *ErrorMsg* f , que vincula un estado sobre el cual no se verifica la precondition de f con un código de error.

Ejecución de los Comandos. La ejecución de un comando f en un estado s produce un nuevo estado s' y una respuesta ans , donde la relación entre el estado previo, el nuevo y la respuesta está determinada por la postcondición de f . Se define el tipo $exec : State \rightarrow Funcs \rightarrow Answer \rightarrow State \rightarrow Prop$, donde:

$$\begin{array}{c} \text{exec\_pre} \frac{Pre\ s\ f \quad Pos\ s\ s'\ ans\ f}{exec\ s\ f\ ans\ s'} \\ \\ \text{exec\_npre} \frac{\neg (Pre\ s\ f) \quad \exists ec, s = s' \wedge ErrorMsg\ s\ f\ ec \wedge ans = error\ ec}{exec\ s\ f\ ans\ s'} \end{array}$$

Notar que si $Pre\ s\ f$ no se cumple entonces el estado s no cambia y la respuesta del sistema es el mensaje de error determinado por la relación *ErrorMsg* f .

5.4. Verificación

La verificación de la especificación se realizó en dos etapas. Primero se probó la invarianza de la validez de los estados respecto a la ejecución de los comandos, esto es, que la semántica de los comandos preserva la propiedad *Valid* (teorema 1). Luego se demostraron propiedades relevantes para los modelos formalizados. A modo de ejemplo se incluyen 2 lemas. Para ver todas las propiedades analizadas, junto con sus pruebas, referirse a [9].

Teorema 1 $\forall (s : State)(f : Func), Valid\ s \wedge Pre\ s\ f \wedge Pos\ s\ s'\ ok\ f \rightarrow Valid\ s'$

La invarianza de *Valid* garantiza que al ejecutarse cualquier comando se obtiene un estado en el cual se cumple que cada una de las sesiones establecidas poseen un único usuario válido dueño de las mismas. Es importante además demostrar que este usuario es el mismo que en el estado previo a la ejecución del comando, siempre y cuando éste no termine la sesión. La propiedad de intransferibilidad de las sesiones se demuestra en el siguiente lema:

Lema 1 $\forall (u : User)(sid : Session)(f : Funcs), isSession\ s\ sid \wedge session\_owner\ s\ u\ sid \wedge exec\ s\ f\ ok\ s' \wedge isSession\ s'\ sid \rightarrow session\_owner\ s'\ u\ sid$

Una propiedad fundamental que debe cumplir la formalización respecto a la jerarquía de roles es que ésta modele correctamente el concepto de herencia. El segundo lema establece que, no puede suceder que un usuario asignado a roles de mayor jerarquía que otro no esté autorizado para un rol cuando el segundo sí lo está.

Lema 2 $\forall(u_1\ u_2:User)(r\ r_2:Role),$
 $(\forall(r_1:Role), role\_of\_user\ s\ r_1\ u_1 \rightarrow r_1 \preceq r_2) \wedge authorized\_user\ s\ u_2\ r_2 \rightarrow$
 $\neg(authorized\_user\ s\ u_1\ r \wedge \neg authorized\_user\ s\ u_2\ r)$

6. Conclusiones y Trabajos Futuros

El análisis realizado de los modelos RBAC considerados en el estándar NIST puso en evidencia la existencia de problemas en torno a la especificación propuesta, que por otra parte no incluye un análisis de su corrección, ni la verificación de propiedades de seguridad.

Este trabajo introdujo una nueva especificación formal de *Core RBAC* y *Hierarchical RBAC*, desarrollada en el CCI, que subsana los problemas encontrados. En particular, esta formalización posibilitó el estudio en profundidad de alternativas libradas a decisiones de implementación por parte del estándar. Se mostró que para administrar la jerarquía de roles, cualquier intento de manipular las herencias indirectas de manera explícita provoca, en algunos casos, la pérdida de la propiedad de orden parcial de la relación que modela la jerarquía y, en otros casos, conduce a la subutilización de la transitividad del orden. Asimismo, en este artículo se pusieron de manifiesto ambigüedades, imprecisiones y omisiones en los requerimientos funcionales de algunos comandos incluidos en el estándar NIST. Por ejemplo, en la especificación del comando *DeleteRole* se omite la presencia de una jerarquía de roles, hecho que permite al sistema transicionar a un estado inválido, donde existan relaciones de herencia entre roles que no pertenecen al conjunto de roles del sistema.

Contar con una especificación en el CCI permitió razonar de manera abstracta sobre la corrección de los modelos RBAC formalizados, usando Coq. Por un lado, se verificó que la semántica de los comandos conserva invariante la validez de los estados, propiedad que garantiza que el sistema no puede transicionar a un estado que no represente un posible estado RBAC. Por otro lado, se demostraron dos propiedades, a modo de ejemplo, relevantes para todo modelo RBAC: la intransferibilidad de las sesiones y el correcto comportamiento de la jerarquía de roles. Debido a limitaciones de espacio se omitieron muchos detalles sobre la nueva especificación introducida en este artículo, como así también propiedades verificadas, junto con sus pruebas. La formalización completa consta de 14 módulos (archivos) Coq con 1525 líneas en total, que incluyen 87 definiciones, 27 lemas y 8 teoremas. El lector interesado puede referirse a [9] por más información.

Si bien éste no es el primer artículo que analiza el estándar NIST referido, se diferencia de los anteriores en que plantea una nueva formalización, en una

teoría de tipos, que permite verificar formalmente propiedades de seguridad relevantes del modelo, usando un asistente de pruebas como Coq. Dado que el formalismo utilizado permite obtener especificaciones *certificadas* de sistemas, se plantea como trabajo futuro la extracción de un prototipo funcional correcto por construcción de la formalización de los modelos RBAC, que sirva como implementación de referencia, u oráculo, para desarrolladores e investigadores. Previo a esto, se propone desarrollar una extensión de la especificación en el CCI para incorporar los modelos RBAC SSD y DSD del estándar NIST. Finalmente, resulta también de interés extender la formalización para permitir razonar sobre secuencias de estados, contemplando, en particular, el análisis de propiedades temporales.

Referencias

- [1] A. E. Abdallah and E. J. Khayat. Formal z specifications of several flat role-based access control models. *Software Engineering Workshop, Annual IEEE/NASA Goddard*, 0:282–292, 2006.
- [2] R. Baldwin. Naming and grouping privileges to simplify security management in large databases. In *IEEE Symposium on Security and Privacy*, pages 116–132, 1990.
- [3] Y. Bertot and P. Castéran. *Interactive Theorem Proving and Program Development. CoqArt: The Calculus of Inductive Constructions*. Texts in Theoretical Computer Science. Springer-Verlag, 2004.
- [4] D. F. Ferraiolo, R. Sandhu, S. Gavrila, D. R. Kuhn, and R. Chandramouli. Proposed nist standard for role-based access control. *ACM Trans. Inf. Syst. Secur.*, 4(3):224–274, 2001.
- [5] J. Joshi, A. Ghafoor, W. G. Aref, and E. H. Spafford. Digital government security infrastructure design challenges. *Computer*, 34(2):66–72, 2001.
- [6] J. B. D. Joshi, W. G. Aref, A. Ghafoor, and E. H. Spafford. Security models for web-based applications. *Commun. ACM*, 44(2):38–44, 2001.
- [7] N. Li, J.-W. Byun, and E. Bertino. A critique of the ansi standard on role-based access control. *IEEE Security and Privacy*, 5(6):41–49, 2007.
- [8] N. Li and M. V. Tripunitara. Security analysis in role-based access control. *ACM Trans. Inf. Syst. Secur.*, 9(4):391–420, 2006.
- [9] C. D. Rosa. Especificación formal del modelo RBAC en el cálculo de construcciones inductivas. Master’s thesis, U.N.Rosario, Argentina, 2008.
- [10] R. Sandhu, D. Ferraiolo, and R. Kuhn. The nist model for role-based access control: Towards a unified standard. In *In Proceedings of the fifth ACM workshop on Role-based access control*, pages 47–63. ACM Press, 2000.
- [11] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman. Role-based access control models. *IEEE Computer*, 29(2):38–47, 1996.
- [12] The Coq Development Team. *The Coq Proof Assistant Reference Manual – Version V8.1*, 2006.
- [13] D. J. Thomsen. Role-based application design and enforcement. *Database Security, IV: Status and Prospects*, 1990.
- [14] T. C. Ting, S. A. Demurjian, and M.-Y. Hu. Requirements, capabilities, and functionalities of user-role based security for an object-oriented design model. In *Results of the IFIP WG 11.3 Workshop on Database Security V*, pages 275–296, Amsterdam, The Netherlands, 1992. North-Holland Publishing Co.

FACTOIDS: Modelos y Herramientas para el Análisis e Intercambio Seguro de Datos Colectados por Sensores

Carlos Martínez-Cagnazzo

Facultad de Ingeniería, Universidad de la República, Julio Herrera y Reissig 565, Montevideo, Uruguay

carlosm@fing.edu.uy

Resumen. Los *Computer Security Information Response Teams* (CSIRTs) son organizaciones de servicio que alojan equipos de trabajo altamente especializados que realizan respuesta a incidentes de seguridad ya sea a nivel de coordinación u operativa. Las actividades maliciosas en Internet no reconocen fronteras de ningún tipo por lo que para poder llevar adelante su tarea con la mayor efectividad posible, los CSIRTs deben establecer relaciones de confianza entre sí para poder compartir información. Para ser aceptables, estos intercambios de información deben además respetar las políticas de seguridad de la información de las organizaciones madre de los respectivos equipos de seguridad y para ser lo más efectivos posible deberían poder ser automatizables. Este trabajo presenta una introducción a este escenario de intercambio de información entre CSIRTs, para luego analizar algunas herramientas y arquitecturas relevantes encontradas en la literatura, realizar un análisis de requerimientos y proponer una arquitectura de alto nivel para intercambio de información automáticos entre CSIRTs a través de dominios administrativos de manera de respetar las políticas de SI de cada organización.

Palabras clave: CSIRTs, Logs, Information Exchange, FACTOIDS, IODEF, IDMEF, Sanitization, Anonymity, Intrusion Detection, Honeypots, Honeynets

1 Introducción

En cualquier instalación de red actual existen múltiples sensores que proporcionan datos sobre eventos de seguridad que pueden ser útiles como entrada para un grupo de respuesta a incidentes de seguridad (CSIRT). Estos sensores incluyen, pero no se limitan a: sistemas de detección de intrusos (IDS), firewalls, routers y diferentes clases de honeypots (1).

Típicamente un CSIRT opera y obtiene datos de una o más de estas fuentes. Sin embargo, la visibilidad que cada uno de los grupos tiene individualmente está limitada por los sensores que cada grupo dispone y administra, los que generalmente son

limitados en número y localizados bajo un único proveedor de servicios de Internet (ISP) En el caso de los sensores de tipo honeypot influye además el tamaño del rango de direcciones IP que el sensor utiliza y además el comportamiento observado puede tener componentes fuertemente locales.

Debido a esto el poder compartir estos datos entre grupos de trabajo permitiría a los investigadores tener una visión más global de los eventos que ocurren en la red y les permitiría extraer conclusiones de validez más general, así como también actuar mitigando amenazas en el caso de los CSIRTs.

Existen diversos obstáculos para que estos intercambios de datos entre grupos de trabajo sean posibles, entre ellos:

- Falta de normalización de los datos colectados por diferentes sensores
- Falta de buenas técnicas de representación interna de los datos y de almacenamiento de los mismos
- Protocolos de transporte de datos de seguridad entre grupos de trabajo
- Necesidad de sanitizar datos antes de compartirlos de acuerdo a las políticas de seguridad de la información del sitio origen de los datos
 - Dependiente del origen, tipo y destino de los datos
 - Teniendo en cuenta además posibles requisitos administrativos y/o legales del sitio origen de los datos
- Si bien distintos grupos tienen interés en consultar información colectada por otros, por un lado o todos los datos que colecta un grupo son de interés para otros, además es generalmente de interés el poder agregar o *minar* esta información de diferentes maneras.

Esto sumado a los importantes volúmenes de información que se manejan, hace necesaria una forma de *consultar* los datos colectados por un grupo por parte de otros de forma automatizada.

- Necesidad de contar con garantías de autenticidad e integridad de los datos obtenidos de otros grupos
- Necesidad de establecer *relaciones de confianza* entre grupos que habiliten el intercambio de información. En general estas relaciones de confianza pasan por el contacto institucional y la firma de alguna forma de “MoU” (*Memorandum of Understanding*), documento que establece en líneas generales las reglas que hacen posible la colaboración.

Estos MoU se establecen en escalas temporales que van de las semanas a meses, muchas veces excediendo el tiempo que los grupos disponen para responder con efectividad a un incidente.

Es de nuestro entender que la posibilidad de establecer estas relaciones de confianza de forma automatizada junto con las garantías de integridad de la información habilitarían nuevas áreas de cooperación así como permitiría una rápida respuesta a incidentes que abarcan grupos de trabajo que no han tenido contactos institucionales previos.

Existen en la literatura propuestas de arquitecturas para compartir información así como también múltiples trabajos que proponen protocolos, lenguajes o componentes que podrían ser utilizados en un sistema de este tipo.

Además de la comunicación entre grupos de trabajo en seguridad, la normalización de la representación de los datos colectados por sensores en sí misma permitiría la correlación de eventos y extracción de conclusiones de forma más ágil y sencilla.

En la segunda sección presentaremos los principales problemas que se deben resolver para habilitar el intercambio de información a través de fronteras administrativas, para luego continuar con referencias a sistemas que atacan problemas similares y terminar con una propuesta de arquitectura (FACTOIDS) que proponemos como modelo de referencia.

2 Aspectos generales y problemas a resolver para facilitar la comunicación entre organizaciones

2.1 Características generales de las fuentes de información de seguridad

En el entorno de red moderno los especialistas en seguridad cuentan con una multitud de fuentes de eventos de seguridad, como ser sistemas de detección de intrusos (IDSs), archivos de log de firewalls, archivos de log de distintos servidores (web, archivos, etc.) y fuentes *Netflow* (2) entre otros.

De particular interés para los equipos de respuesta a incidentes de seguridad (CSIRTs) resultan los *honeypots* y *honeynets*. De acuerdo a (1) y (3), un *honeypot* es un recurso cuyo valor reside en el uso no autorizado o ilícito del mismo. Los *honeypots* pueden ser de alta o baja interacción según el recurso expuesto sea virtualizado o consista en un sistema real. De acuerdo a (3) *honeynet* es un tipo de *honeypot* de *alta interacción*.

Como toda actividad detectada en un *honeypot* se asume maliciosa, estos dispositivos proveen fuentes de información extremadamente valiosa, permitiendo obtener inteligencia acerca de mecanismos de ataque, nuevas formas de propagación de *malware*, métodos de envío de correo *spam* (4),(5), etc.

Entre los distintos tipos de datos que podemos identificar en estas fuentes de eventos tenemos:

- Direcciones IP: Las direcciones IP aparecen como uno de los tipos de datos más frecuentes, ya que son un factor común a la mayoría de los eventos de seguridad.
- Encabezados de correo electrónico: Los mensajes de correo electrónico contienen *encabezados* (headers) que contienen información útil para estudiar el comportamiento de, por ejemplo, los envíos de correo no solicitado.
- Flujos de tráfico: Un flujo de tráfico es un conjunto de paquetes de red intercambiados entre dos puntos finales de comunicación. Cuando se envía un correo electrónico o se copia un archivo por red, la información se debe

paquetizar y estos paquetes fluyen entre dos procesos (los puntos finales de la comunicación), constituyendo así un *flujo*.

La forma más común en que un sensor almacena su información es lo que se conoce usualmente como archivo de *log*. Un archivo de log se puede abstraer como una serie temporal de líneas de texto.

Si bien existen algunos estándares de facto para archivos de log en algunos dominios específicos como el *Extended Log File Format* (6) del World Wide Web Consortium, en la mayoría de los casos la estructura de los mismos es específica de cada sensor y aplicación.

2.2 Necesidad de modelos de información para almacenamiento y transmisión

Si algo tienen en común todas estas fuentes de información es su heterogeneidad y falta de estructura uniforme. Por ello hace falta elaborar *modelos de información* que permitan almacenar y transmitir esta información con eficiencia y expresividad.

Existen múltiples aproximaciones a este problema, en su mayoría diseñadas para dominios de problema específicos. Entre ellos podemos mencionar a CVE (*Common Vulnerabilities and Exposures*) (7) (8), OVAL (*Open Vulnerability and Assessment Language*) (9) (10)

En particular, el *Incident Object Description Exchange Format* (IODEF) (11) es un modelo de información basado en XML definido por el IETF (*Internet Engineering Task Force*). De acuerdo a (11), IODEF es un formato diseñado para representar información de seguridad informática a intercambiar entre CSIRTs. Su principal objetivo es proveer a los grupos de respuesta a incidentes con información *normalizada y procesable automáticamente*.

IODEF define un esquema de documento XML y diferentes *clases* de información, como ser la clases “*Incident*” (Incidente), “*Contact*” (Contacto) y “*Report Time*” (Referencia temporal del reporte). IODEF es además un formato *extensible* y el mismo provee mecanismos para enriquecer el modelo.

IODEF no define como deberán lograrse las propiedades de confidencialidad, integridad y autenticidad, (11) explícitamente menciona que estas propiedades deberán ser garantizadas por los mecanismos de transporte o almacenamiento seleccionados.

```

<?xml version="1.0" encoding="UTF-8"?>
<!-- This example demonstrates a report for a very old worm (Code Red) -->
<IODEF-Document version="1.00" lang="en"
  xmlns="urn:ietf:params:xml:ns:iodef-1.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="urn:ietf:params:xml:schema:iodef-1.0">
  <Incident purpose="reporting">
    <IncidentID name="csirt.example.com">189493</IncidentID>
    <ReportTime>2001-09-13T23:19:24+00:00</ReportTime>
    <Description>Host sending out Code Red probes</Description>
    <!-- An administrative privilege was attempted, but failed -->
    <Assessment>
      <Impact completion="failed" type="admin"/>
    </Assessment>
    <Contact role="creator" type="organization">
      <ContactName>Example.com CSIRT</ContactName>
      <RegistryHandle registry="arin">example-com</RegistryHandle>
      <Email>contact@csirt.example.com</Email>
    </Contact>
  </Incident>
</IODEF-Document>

```

Fig. 1. Ejemplo de un documento IODEF tomado de la RFC 5070 (11)

2.3 Normalización de eventos

La estructura de los archivos de log y en general de la información obtenida de las diferentes fuentes de información de seguridad posee estructuras dispares, con diferentes campos de información ocupando diferentes lugares, posiciones o con diferentes formatos.

Por ello es necesario *normalizar* esta información obteniendo los campos de información relevantes para el modelo de información. El proceso de *normalización* es que nos permitirá obtener por ejemplo direcciones IP, o direcciones de correo electrónico a partir de archivos de log de estructuras dispares para almacenarlos o transmitirlos de acuerdo a un modelo de información.

2.4 Sanitización de la información de seguridad

Uno de los principales obstáculos para la automatización del flujo de información de seguridad a través de fronteras administrativas de diferentes organizaciones es el temor de que en el proceso de divulgación de información se produzcan fugas de información considerada sensible.

Entre estos temores es posible distinguir dos casos, a saber:

- La divulgación de datos sobre infraestructura crítica (servidores de DNS, servidores de correo internos, etc.) o información sobre sistemas comprometidos o vulnerables de la organización. En este caso, el temor principal es que de caer esta información en manos de una entidad maliciosa, la misma pueda ser utilizada para realizar ataques dirigidos con mayor posibilidad de éxito.

- La divulgación de información personal que pueda incumplir alguna normativa de protección de la privacidad ya sea de la organización que divulga los datos o del país donde esta se encuentra, o que pueda comprometer información personal. El temor principal en este caso, además de posibles consecuencias legales es el de esta información personal pueda ser utilizada por una entidad maliciosa para realizar ataques dirigidos de *phishing* (12) o similares intentos de fraude.

En todo proceso de divulgación de información se hace necesario entonces *sanitizar* la información antes de transmitirla, ya sea para ocultar información de infraestructura o para anonimizar información personal. El trabajo de Bishop y Crawford (13) presenta los principales problemas que ocurren al sanitizar información de capa de red; problemas que se pueden generalizar a otros tipos de información.

La técnica clásica de sanitización es la conocida como “*black marker*”¹ (marcador negro): la información considerada sensible bajo algún criterio es suprimida de la salida cambiándola por una etiqueta fija, por ejemplo el mensaje “IP 192.168.1.10” se transforma en “IP X.X.X.X”.

El problema principal con esta técnica es que elimina información sin preservar ninguna de las características del conjunto de datos original. En (14) se discute el impacto de la sanitización de flujos de eventos de seguridad en un entorno cooperativo de detección de intrusos.

En (13) se analiza el problema de sanitizar eventos de seguridad conteniendo direcciones IP como un problema de *mapeo* entre dos espacios a través de sustituciones guiadas por funciones o por tablas. En este trabajo también se establece que para que el analista pueda continuar obteniendo conclusiones útiles estos mapeos deben preservar la mayor cantidad de propiedades de los datos originales, como ser distancias, localidad, etc.

3 Referencias en la literatura

3.1 Sistemas de Gestión de Información de Seguridad (SIMs)

Los Sistemas de Gestión de Información o *Security Information Management Systems*, de los cuales el ejemplo mejor conocido es OSSIM (15), presentan en general arquitecturas distribuidas, basadas en el concepto de agregar información proveniente de múltiples sensores en una base de datos única para luego reprocesarla para obtener correlaciones y realizar detección de intrusos.

OSSIM presenta una arquitectura donde se identifican componentes de captura de eventos de seguridad, almacenamiento de información y un fuerte énfasis en la correlación de eventos para detección de intrusiones.

¹ El “*black marker*” es una analogía con las técnicas de eliminación de información de documentos, en los cuales ciertas líneas son borradas tapándolas con tinta negra.

El objetivo principal de estos sistemas es incrementar la capacidad de detección prematura de intrusiones y la capacidad de respuesta ante las mismas, presentando a un operador con información sumariada y aplicando métodos de correlación de eventos.

Si bien muchos de estos sistemas incluyen funcionalidades de *exportar* información de seguridad en diferentes formatos (IODEF por ejemplo), la comunicación automatizada a través de fronteras administrativas no es su objetivo principal.

3.2 AirCERT

AirCERT (16) fue un proyecto llevado adelante por el CERT/CC (*CERT Coordination Center*) de la Universidad de Carnegie-Mellon, cuyo objetivo explícitamente consistió en implementar un sistema *escalable y distribuido* con el objetivo de *compartir información de seguridad a través de dominios administrativos*.

AirCERT identifica varios componentes, incluyendo *sensores, normalizadores, colectores, herramientas de análisis y herramientas para representar información*. En el sitio web de AirCERT se encuentran implementaciones de algunos de estos componentes, aunque el proyecto AirCERT parece estar abandonado, no habiéndose encontrado actualizaciones posteriores a 2003-2004, y consultas realizadas por el autor en diferentes listas de correo no recibieron respuesta.

3.3 HIDEF / HIDEAS

HIDEF (*Data Exchange Format for Information Collected in Honeypots and Honeynets*) (17) propone un formato basado en XML como una extensión de IODEF que se propone resolver el problema de falta de interoperabilidad entre diferentes implementaciones de *honeypots* y *honeynets*.

HIDEAS (*Honeypots Information and Data Exchange and Analysis System*) (18) es una continuación de este mismo trabajo en el que se propone una arquitectura que intercambiando información en formatos como IODEF, IDMEF o HIDEAS habilita la correlación y posterior análisis y procesamiento de estos datos.

HIDEAS plantea que su objeto fundamental es el sensor *honeypot*. A partir de esta definición, se plantean requisitos funcionales que se plantean para HIDEAS se encuentran el almacenamiento y comunicación de información de seguridad de acuerdo, el soporte para sanitización de eventos colectados por *honeypots*. Se define que la seguridad en el transporte debe ser proporcionada por protocolos externos.

La hipótesis de que el sensor fundamental es el *honeypot* permite realizar hipótesis sobre los tipos de datos contenidos en los eventos capturados, como ser que en su mayoría serán eventos de red (direcciones IP, puertos TCP, etc.)

4 FACTOIDS

De acuerdo a lo expresado en la introducción, la visión de este trabajo es proponer una arquitectura general que brinde los servicios básicos identificados como requerimientos, es decir que permita recolectar datos de sensores heterogéneos, procesar estos datos para normalizarlos de acuerdo con un cierto modelo de información para luego compartirlos con una federación de sistemas de arquitectura similar, modelo que hemos llamado FACTOIDS.

La arquitectura propuesta incluye conceptos de OSSIM, AirCERT y HIDEAS. En particular, puede verse como una generalización de este último para incluir otras fuentes de información además del honeypot.

Los conceptos fundamentales que subyacen a la arquitectura son:

- La capacidad de *recolectar datos de sensores heterogéneos*, incluyendo pero no limitándose a honeypots, fuentes *Netflow*, archivos de log de firewalls y routers, etc.
- El *almacenamiento normalizado de estos datos* en alguna base de datos que soporte un *modelo de información bien especificado* y compatible con modelos estándar.
- El *soporte para realizar correlación de eventos y detección de intrusiones operando sobre datos propios y recibidos de otros sistemas similares*.
- El soporte para *sanitización de eventos* con la incorporación de *funciones de mapeo* adecuadas para los tipos de datos que el sistema maneja.
- La especificación de *interfaces bien definidas* con semánticas claras que permitan la interconexión sencilla de sistemas similares.
- El soporte para realizar *enriquecimiento contextual* de los eventos colectados. Existen casos donde los eventos colectados pueden enriquecerse con información adicional a la enviada por el sensor mismo, pero que debe obtenerse por otras vías más lentas o costosas que imposibiliten el hacerlo en línea.

Un ejemplo de esto es el agregar a cada dirección IP mencionada en un evento colectado la información de su mapeo reverso de DNS y su sistema autónomo. Ambos datos pueden obtenerse haciendo consultas a bases de datos en Internet, pero ambas pueden resultar uno o más órdenes de magnitud más lentas que la normalización y almacenamiento de eventos.

- El soporte para la *realización de consultas* por parte de sistemas externos apropiadamente autorizados.

En la mayoría de las arquitecturas similares encontradas en la literatura, el énfasis en la comunicación con sistemas externos está puesto en la *exportación* de eventos. Creemos que en muchos casos puede resultar más eficiente y ágil el permitir que los sistemas externos adecuadamente autorizados realicen *consultas* sobre los repositorios locales de información.

En el resto de esta sección presentaremos las interfaces y componentes de la arquitectura FACTOIDS.

4.1 Interfaces

4.1.1 Interfaz “A” – Recolección de datos de sensores

La interfaz “A” representa el punto de contacto entre sensores y la base de datos de eventos normalizada. Esta interfaz es de un lado dependiente del tipo de sensor, mientras que en su lado interno conecta a los sensores con la base de datos normalizada aplicando el modelo de datos identificado.

Por ello, la interfaz A se especializa según los diferentes sensores de los que se busque recibir datos. En el ejemplo del a figura tenemos:

1. **A1** : sistema de detección de intrusos
2. **A2** : firewall / router
3. **A3** : Honeypot
4. **A4** : Fuente Netflow

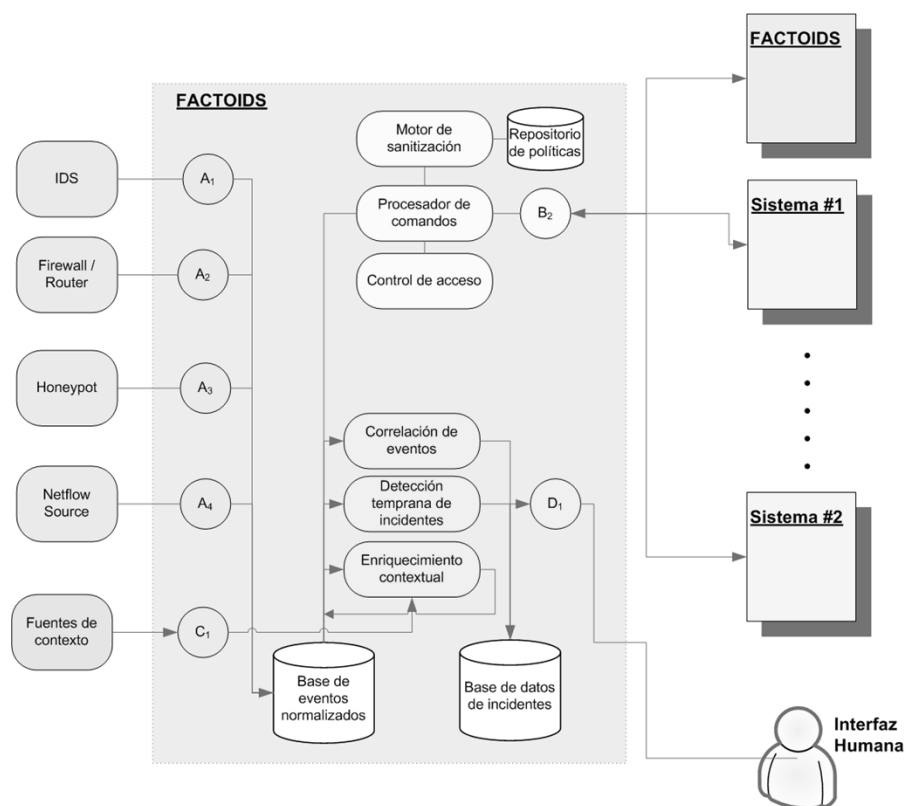


Fig. 2. FACTOIDS

4.1.2 Interfaz “B” – Intercambio de información con sistemas externos al grupo

La interfaz “B” es la interfaz que provee el punto de contacto entre fronteras administrativas, posiblemente con otros sistemas de arquitectura similar

Esta interfaz se alimenta por un lado de los datos ya normalizados y deberá implementar un conjunto de operaciones que permitan la realización consultas por parte de sistemas similares apropiadamente autorizados para acceder al repositorio local de información.

Es a este flujo saliente de información al que se le deben aplicar las políticas de seguridad de la información de la organización madre, las que se almacenarán en el repositorio de políticas.

4.1.3 Interfaz “C” – Interfaz para enriquecimiento contextual de eventos

Esta interfaz provee el punto de contacto con bases de datos de información externa como ser los mapeos IP-sistema autónomo (ASN) (19) o búsquedas reversas en DNS. Como se mencionó anteriormente, no es posible realizar estas búsquedas a nivel de las interfaces “A” por motivos de performance, y además es posible que no tenga sentido realizarlas para *todos* los eventos colectados si no para un subconjunto de los mismos.

4.1.4 Interfaz “D” – Interfaz humana

Interfaz con operadores humanos, donde se prestan funcionalidades como:

- Interfaz de administración
- Detección temprana de incidentes, mediante el motor de correlación
- Notificación de alarmas por consola u otras vías (e-mail, etc.)

4.2 Componentes

FACTOIDS define tres grupos de componentes básicos, a saber: Componentes de *Almacenamiento*, componentes de *Reprocesamiento* y componentes de *Control*.

4.2.1 Componentes de almacenamiento

El sistema necesita almacenar la información recolectada de los sensores, así como también permitir la operación de otros componentes que procesen y enriquezcan esta información.

La naturaleza heterogénea y el volumen de la información recolectada a partir de los sensores, así como la necesidad de procesarla con eficiencia, hace que los componentes de almacenamiento sean uno de los puntos más críticos del sistema.

Para este componente se debe definir un *modelo de información*, que estandarice la representación de los diferentes tipos de eventos enviados por los sensores.

4.2.2 Componentes de control

Los componentes de control agrupan las funcionalidades de aplicación de políticas de seguridad (incluyendo un repositorio adecuado para el almacenamiento de estas políticas), el procesamiento de comandos y respuestas recibidos de sistemas externos y de los operadores a través de la interfaz “B”.

El motor de políticas en particular deberá aplicar las políticas de sanitización definidas por la organización, para ello deberá contar con un *repositorio de políticas* y acceso a un conjunto de *funciones de mapeo* que le permitan implementar estas políticas para los diferentes tipos de datos y eventos colectados.

Los componentes de control deberán incluir módulos de *control de acceso* que permitan definir privilegios y establecer las relaciones de confianza con otros dominios administrativos.

4.2.3 Componentes de reprocesamiento

Los componentes de reprocesamiento implementan funcionalidades de reprocesamiento de los eventos recolectados a través de la interfaz “A”, incluyendo por ejemplo *motores de correlación* que permitan realizar detección temprana de incidentes, procesos de archivo y eliminación de información antigua.

5 Conclusiones y trabajo a futuro

El escenario de intercambio de información de seguridad entre CSIRTs, a través de fronteras administrativas encierra la promesa de permitir a investigadores, personal de respuesta a incidentes y otros grupos como ser equipos policiales el acceder a vastos repositorios de información.

Para obtener el máximo valor de esta información con agilidad y eficiencia, es necesario resolver ciertos problemas que incluyen el almacenamiento normalizado, transmisión en formatos estándar y acceso a través de interfaces bien definidas.

Si bien se han propuesto soluciones y arquitecturas con fines relativamente similares, creemos que es necesario definir una arquitectura lo suficientemente general, basada en todo el trabajo previo existente, que permita acomodar las necesidades de un público amplio.

A diferencia de AirCERT, el objetivo de este trabajo no es el de proveer una implementación exhaustiva de todos los componentes si no el de definir un marco que permita incorporar diferentes propuestas de modelos de información, funciones de sanitización, motores de correlación y otros eventos.

Como generalización de HIDEAS, FACTOIDS propone el incorporar fuentes de información adicionales a los honeypots, así como poner un énfasis en el control de acceso, privilegios de acceso y sanitización de eventos.

Como concepto adicional, FACTOIDS incorpora la noción de la bidireccionalidad de la interfaz “B” (interfaz con otros dominios administrativos), a través de la cual no solo sería posible exportar información sino también realizar consultas.

FACTOIDS es un trabajo en progreso y por ello hay varios aspectos todavía no definidos completamente. Una vez establecidas todas las interfaces del sistema, se pasará a definir detalladamente la semántica de las diferentes interfaces y continuar con la implementación de un prototipo basado en IODEF como modelo de información y proporcionando funciones de sanitización para los eventos de red.

6 Referencias

1. **Spitzner, Lance.** Honeypots - Definitions and Value of Honeypots. *Sitio web "Tracking Hackers"*. [Online] Marzo 2003. [Cited: Marzo 15, 2009.] <http://www.tracking-hackers.com/papers/honeypots.html>.
2. **Cisco Systems, Inc.** RFC 3954 - Cisco Systems NetFlow Services Export Version 9. *Internet Engineering Task Force*. [Online] 2009. [Cited: marzo 15, 2009.] <http://www.ietf.org/rfc/rfc3954.txt>.
3. **The HoneyNet Project.** Know Your Enemy: Honeynets. *The HoneyNet Project*. [Online] Mayo 31, 2006. [Cited: Marzo 15, 2009.] <http://old.honeynet.org/papers/honeynet/>.
4. **CERT.br: Computer Emergency Response Team - Brazil.** Spam Metrics: The SpamPots Project. *Sitio web CERT.br*. [Online] 2007. [Cited: marzo 15, 2009.] <http://www.cert.br/docs/palestras/certbr-icann2007.pdf>.
5. **Hoepers, Cristine and Steding-Jessen, Klaus.** New Developments in the SpamPots Project. *Sitio web CERT.br*. [Online] 2008. [Cited: marzo 15, 2009.] <http://www.cert.br/docs/palestras/certbr-national-csirts-meeting2008.pdf>.
6. **World Wide Web Consortium.** Extended Log File Format; W3C Working Draft WD-logfile-960323. *W3C Web Site*. [Online] 1996. [Cited: marzo 15, 2009.] <http://www.w3.org/TR/WD-logfile.html>.
7. **The Mitre Corporation.** *Making Security Measurable*. [Online] enero 14, 2009. [Cited: marzo 15, 2009.] <http://measurablesecurity.mitre.org/>.
8. **Towards a Common Enumeration of Vulnerabilities. Mann, David E. and Christey, Steven M.** West Lafayette, Indiana, USA : s.n., 1999. 2nd Workshop on Research with Security Vulnerability Databases, Purdue University.
9. **The Mitre Corporation.** About OVAL. *Mitre's OVAL Website*. [Online] marzo 6, 2008. [Cited: marzo 15, 2009.] <http://oval.mitre.org/oval/about/index.html>.
10. —. An Introduction to the OVAL Language. *Mitre's OVAL Website*. [Online] 2006. [Cited: marzo 15, 2009.] http://oval.mitre.org/oval/documents/docs-06/an_introduction_to_the_oval_language.pdf.
11. **Danyliw, R., Meijer, J. and Demchenko, Y.** RFC 5070 - The Incident Object Description Exchange Format. *Internet Engineering Task Force (IETF)*. [Online] Diciembre 2007. [Cited: marzo 15, 2009.]
12. **Why phishing works. Dhamija, Rachna, Tygar, J. D. and Hearst, M.** Montréal, Québec, Canada : ACM New York, NY, USA, 2006. ISBN:1-59593-372-7.
13. **Some Problems in Sanitizing Network Data. Bishop, M., et al., et al. s.l. :** IEEE Computer Society Washington, DC, USA, 2006. Proceedings of the 15th IEEE

International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises. ISBN ~ ISSN:1524-4547 , 0-7695-2623-3 .

14. *Impact of Sanitized Message Flows in a Cooperative Intrusion Warning System*. **Tolle, J. Jahnke, M. Felde, N.G. Martini, P.** Washington DC : IEEE, 2006. Military Communications Conference, 2006. MILCOM 2006. IEEE. ISBN: 1-4244-0617-X.

15. **AlienVault**. OSSIM: Open Source Security Information Management. *Sitio web de OSSIM*. [Online] [Cited: Abril 14, 2009.] <http://www.ossim.net/>.

16. **CERT Coordination Center**. AirCERT. *Sitio web de AirCERT*. [Online] [Cited: marzo 15, 2009.] <http://aircert.sourceforge.net/>.

17. *HIDEF: a Data Exchange Format for Information Collected in Honeypots and Honeynets*. **Hoepers, C, Vijaykumar, NL and Montes, A.** 2007.

18. **Hoepers, Cristine**. *Projeto E Implementacao De Uma Infra-Estrutura Para Troca E Analise De Informacoes De Honeypots E Honeynets*. INPE - Instituto Nacional de Pesquisas Espaciais. Sao Paulo : INPE, 2008. Tesis doctoral. INPE-0000-TDI/000.

19. **Team Cymru**. IP to ASN Mapping. *Team Cymru Website*. [Online] 2009. [Cited: marzo 15, 2009.] <http://www.team-cymru.org/Services/ip-to-asn.html>.

20. **Debar, H., Curry, D. and B.Feinstein**. RFC 4765. *The Intrusion Detection Message Exchange Format (IDMEF)*. [Online] marzo 2007. [Cited: marzo 15, 2009.] <http://www.ietf.org/rfc/rfc4765.txt>.

Autorización de Acceso en MIDP 3.0

Gustavo Mazeikis y Carlos Luna

Instituto de Computación, Facultad de Ingeniería, Universidad de la República, Uruguay
mazeikis@adinet.com.uy, cluna@fing.edu.uy

Resumen. En la Plataforma Java Micro Edition, el Perfil para Dispositivos de Información Móviles (MIDP) provee el ambiente de ejecución estándar para teléfonos móviles y asistentes de datos personales. La tercera versión del perfil, de reciente publicación, introduce una nueva dimensión en el modelo de seguridad de MIDP: la seguridad a nivel de aplicación. Para la segunda versión de MIDP, Zanella, Betarte y Luna proponen una especificación formal del modelo de seguridad en el Cálculo de Construcciones Inductivas, usando el asistente de pruebas Coq. Este artículo presenta una extensión conservativa de la especificación referida que incorpora la autorización de acceso de aplicaciones a recursos compartidos, introducida por la tercera versión de MIDP. El trabajo también propone un refinamiento de la especificación y un algoritmo funcional para el evento de autorización de acceso, cuyo comportamiento es formalmente verificado. Finalmente, se pone de manifiesto una debilidad del nuevo modelo de seguridad. Mediante un caso concreto se ilustra como una aplicación accede a un recurso compartido falsificando una credencial exigida. Para subsanar esta debilidad, se proponen tres medidas alternativas.

Palabras Clave: Dispositivos Móviles Interactivos, MIDP, Especificación y Verificación Formal, Teoría de Tipos, Coq.

1. Introducción

En los últimos años el uso de dispositivos portátiles, tales como teléfonos celulares y asistentes digitales personales, se ha popularizado a escala mundial. Este tipo de dispositivos tienen fines y características que son en esencia diferentes de los que tienen, por ejemplo, las computadoras portátiles o de escritorio.

En la actualidad, la gran mayoría de las plataformas tecnológicas para los dispositivos portátiles incorporan la tecnología Java para sistemas embebidos – *embedded systems* – denominada *Java Micro Edition* (JME) [1]. La característica fundamental de esta edición, al igual que en toda tecnología Java, es el uso de una máquina virtual, especialmente diseñada y adaptada para aprovechar al máximo los escasos recursos con los que cuentan los dispositivos portátiles. La tecnología JME proporciona mecanismos integrales para garantizar propiedades de seguridad de los dispositivos; en particular, para dispositivos móviles define un modelo de seguridad

que restringe el acceso de las aplicaciones a funciones consideradas potencialmente peligrosas.

JME plantea una arquitectura basada en dos capas por encima de la máquina virtual: la capa configuración y la capa perfil. La configuración *Connected Limited Device Configuration* (CLDC) es un conjunto mínimo de bibliotecas de clases para ser utilizadas en dispositivos con procesadores poco potentes, memoria de trabajo limitada y capacidad para establecer comunicaciones de bajo ancho de banda. Esta configuración se complementa con el perfil *Mobile Information Device Profile* (MIDP) para obtener un entorno de ejecución adaptado a dispositivos portátiles como teléfonos celulares y asistentes digitales personales. El perfil define, entre otras cosas, un modelo de seguridad para aplicaciones, el ciclo de vida de las mismas, los mecanismos de comunicación de red y una biblioteca para el acceso a funciones propias de la clase de dispositivos para la que fue concebido.

En la primera versión de MIDP (MIDP 1.0) [2] se plantea, y en la segunda versión de MIDP (MIDP 2.0) [3] se refina, un modelo de seguridad a nivel de plataforma basado en conjuntos de permisos, para acceder a las funciones del dispositivo. En la tercera versión (MIDP 3.0) [4] se introduce una nueva dimensión al modelo: la seguridad a nivel de aplicación, basada en autorizaciones de acceso, para que una aplicación pueda acceder a los recursos que comparte otra aplicación.

Para MIDP 2.0 Zanella, Betarte y Luna proponen en [5, 6] una especificación formal del modelo de seguridad en el Cálculo de Construcciones Inductivas [7], usando el asistente de pruebas Coq [8]. Esta especificación es una base formal para la verificación del modelo de seguridad y la comprensión de su complejidad.

En este artículo se propone una extensión conservativa de la especificación formal mencionada y un algoritmo certificado para evaluar la autorización de acceso a los recursos compartidos de una aplicación en MIDP 3.0. La extensión de la formalización está también implementada usando el asistente de pruebas Coq. La versión completa del código fuente y la documentación están disponibles en [9].

Otro aporte de este trabajo son observaciones al nuevo modelo de seguridad, las cuales surgen del estudio de la especificación informal. Una debilidad introducida en esta nueva versión se pone de manifiesto mediante un ejemplo de acceso, falsificando una credencial exigida. Se proponen medidas alternativas para subsanar la debilidad referida.

El artículo está organizado como sigue. En la sección 2 se describe el modelo de seguridad de MIDP a través de sus distintas versiones. En la sección 3 se resume la especificación formal del modelo de seguridad de MIDP 3.0. La sección 4 presenta el algoritmo certificado para autorización de acceso. La sección 5 plantea observaciones a la seguridad a nivel de aplicación de MIDP 3.0 y, finalmente, la sección 6 exhibe las conclusiones y los trabajos futuros.

Una versión preliminar de este trabajo es [10] y una versión extendida es el reporte técnico [11].

2. El Modelo de Seguridad de MIDP

El modelo de seguridad de MIDP ha evolucionado a través de sus distintas versiones. En las primeras dos versiones el principal objetivo fue la protección de funciones sensibles del dispositivo, mientras que en la tercera versión se busca proteger los recursos de una aplicación que pueden compartirse con otras aplicaciones. Estos dos niveles de seguridad se describen a continuación.

2.1 Seguridad a Nivel de Plataforma

En MIDP 1.0 toda aplicación que no haya sido preinstalada en el dispositivo (por ejemplo, por el fabricante), se ejecuta en un ambiente controlado denominado *sandbox* que prohíbe el acceso a aquellas funciones del dispositivo que se consideran potencialmente peligrosas.

En MIDP 2.0 se refina el modelo *sandbox* en uno menos restrictivo, basado en el concepto de *dominio de protección*. En los dispositivos compatibles con MIDP 2.0 cada aplicación instalada está ligada a un único dominio de protección. La política de seguridad es una matriz de control de acceso, donde los sujetos son las aplicaciones asociadas a cada dominio de protección y los objetos son las funciones del dispositivo que deben protegerse.

El dominio de protección otorga permisos de dos formas distintas a las aplicaciones que resultan ligadas al mismo. La primera forma es otorgando el permiso de manera incondicional. La segunda forma implica la participación del usuario mediante una modalidad de intervención: *blanket*, el permiso otorgado es válido mientras la aplicación está instalada; *session*, el permiso otorgado es válido mientras la aplicación está ejecutándose; y *oneshot*, el permiso es concedido sólo para ese uso. Por lo tanto, el uso de ciertos recursos sensibles involucra una consulta al usuario y en función de su respuesta se determina la disponibilidad del recurso.

El conjunto de permisos efectivamente otorgados a una aplicación se determina en función de su dominio de protección, de los permisos solicitados por la aplicación y de los permisos otorgados por el usuario. Una aplicación posee un determinado permiso cuando lo ha solicitado, y además su dominio de protección otorga el permiso incondicionalmente, o su dominio especifica un modo de interacción con el usuario para el permiso, y el usuario ha dado una autorización que continúa siendo válida.

El modelo establece dos clases de aplicaciones. Una aplicación no confiable es aquella para la que no se ha podido determinar el origen y la integridad. Toda aplicación no confiable, así como una desarrollada sobre MIDP 1.0, se asocia a un dominio de protección que provee un entorno de ejecución restringido. Una aplicación confiable es aquella para la que se ha podido determinar su origen e integridad. Cuando se verifica el origen de la aplicación, se determina también el dominio de protección al que se asocia.

2.2 Seguridad a Nivel de Aplicación

En MIDP 3.0 se permite que aplicaciones de diferentes dominios de seguridad compartan datos en formato *Record Management System* (RMS) y se comuniquen en tiempo de ejecución mediante eventos de aplicación, o mediante el protocolo *Inter-Midlet Communication* (IMC). La necesidad de autorizar el acceso a los recursos compartidos por una aplicación define una nueva dimensión en el modelo de seguridad de MIDP. La seguridad a nivel de aplicación, basada en el concepto de autorización de acceso, es complementaria de la seguridad a nivel de plataforma, basada en el concepto de permiso.

Los mecanismos para validar la identidad de las aplicaciones involucradas se basan en la Infraestructura de Clave Pública X.509 [3, 12]. Una aplicación se considera firmada si declara la firma del archivo de aplicación (realizada con la clave privada del vendedor) y el certificado de clave pública del vendedor. Los certificados raíz instalados en el dispositivo permiten verificar los certificados y las firmas declarados.

Para que una aplicación comparta recursos debe declarar autorizaciones de acceso, que se diferencian por las credenciales exigidas a las aplicaciones solicitantes. La declaración de autorización puede tomar algunas de las cuatro formas posibles: autorización para las aplicaciones del mismo dominio de seguridad; autorización para las aplicaciones firmadas con un certificado específico; autorización para las aplicaciones de un vendedor determinado, firmadas con un certificado específico; y autorización para las aplicaciones sin firmar, de un vendedor determinado.

Cuando otra aplicación pretende acceder a recursos compartidos, el dispositivo aplica un conjunto de reglas para determinar si está autorizada. Las reglas evalúan la declaración de la aplicación que comparte y las credenciales de la aplicación que solicita, como ser: el nombre de su dominio de protección y, el nombre, la firma y los certificados de clave pública de su vendedor. Si las credenciales de la aplicación solicitante se ajustan a las autorizaciones declaradas entonces se concede autorización de acceso a los recursos compartidos.

3. Especificación Formal del Modelo de Seguridad de MIDP 3.0

Esta sección describe una extensión de la formalización del modelo de seguridad de MIDP 2.0 [5, 6] con los conceptos relacionados a la seguridad a nivel de aplicación introducidos en MIDP 3.0. Se presenta la notación utilizada y los principales elementos de la especificación formal: la política de seguridad del dispositivo, las aplicaciones instaladas, el conjunto de estados válidos del dispositivo y los principales eventos vinculados con la seguridad. La conducta de los eventos se especifica mediante pre y pos condiciones. En la extensión se conservan las propiedades del modelo de seguridad demostradas en la formalización original. También se plantean y demuestran nuevas propiedades relacionadas con la autorización de acceso.

3.1 Notación

En la formalización se usa la notación estándar para la igualdad y los operadores lógicos ($\wedge, \vee, \neg, \rightarrow, \forall, \exists$). La definición de un tipo registro R con un único constructor mkR y funciones de proyección $campo_i: R \rightarrow A_i$, tiene la forma:

$$R =_{def} \{ campo_1: A_1, \dots, campo_n: A_n \} \quad (1)$$

La aplicación de las proyecciones sobre un registro r se abrevia con la notación punto ($campo_i r = r.campo_i$). Por cada $campo_i$ de un registro r se define una relación binaria $\equiv_{campo_i}$ entre objetos del tipo como sigue,

$$r_1 \equiv_{campo_i} r_2 =_{def} \forall j, j \neq i \rightarrow r_1.campo_j = r_2.campo_j \quad (2)$$

Se asume como predefinido el tipo inductivo paramétrico *option* T , con los constructores *None*: *option* T y *Some*: $T \rightarrow option\ T$.

3.2 Política de Seguridad y Aplicaciones

Se denota al conjunto de los dominios de protección del dispositivo con *Domain* y al conjunto de los permisos para cada función protegida del dispositivo con *Permission*. La política de seguridad define, para cada dominio de protección, los permisos que se otorgan a las aplicaciones ligadas al dominio, así como los permisos que pueden ser otorgados por el usuario en una de tres modalidades. *Mode* denota el conjunto enumerado de estas modalidades $\{oneshot, session, blanket\}$. La política de seguridad es una constante de tipo *Policy*.

$$Policy =_{def} \{ \begin{array}{l} allow: Domain \rightarrow Permission \rightarrow Prop, \\ user: Domain \rightarrow Permission \rightarrow Mode \rightarrow Prop \end{array} \} \quad (3)$$

Las aplicaciones MIDP, comúnmente llamadas MIDlets, suelen agruparse para su distribución en paquetes denominados suites. Se denota el conjunto de los identificadores válidos para MIDlet suites con *SuiteID*. Un tipo registro *Suite* representa una suite instalada, con campos para su identificador, el dominio de protección asociado y su descriptor.

$$Suite =_{def} \{ id: SuiteID, domain: Domain, descriptor: Descriptor \} \quad (4)$$

Una MIDlet suite expone, mediante un archivo descriptor, información relevante para el dispositivo. Entre otros atributos declara: el nombre de su vendedor, la firma y el certificado de la clave pública en caso de ser firmada, los permisos sobre los recursos sensibles que utiliza, y las autorizaciones de acceso sobre los recursos de aplicación que comparte.

El tipo registro *Descriptor* representa el archivo homónimo. *Vendor* denota los nombres de vendedores de suites, *Signature* representa la firma del archivo de la aplicación, y *Certificate* el conjunto de los certificados de clave pública. Los predicados *required* y *optional* denotan respectivamente la declaración de los permisos necesarios y opcionales. El predicado *domainAuthorization* denota la declaración de autorización de acceso por dominio de protección. La autorización de

acceso para un conjunto de suites firmadas con un certificado particular se establece con el predicado *signerAuthorization*. El predicado *vendorUnsignedAuthorization* hace lo propio para el conjunto de suites sin firmar de un vendedor, mientras que *vendorSignedAuthorization* denota la autorización de acceso al conjunto de suites, firmadas con un certificado específico de un vendedor.

$$\begin{aligned} \text{Descriptor} =_{\text{def}} \{ & \text{vendor: Vendor,} \\ & \text{jarSignature: option Signature,} \\ & \text{signerCertificate: option Certificate,} \\ & \text{required, optional: Permission} \rightarrow \text{Prop,} \\ & \text{domainAuthorization: Domain} \rightarrow \text{Prop,} \\ & \text{signerAuthorization: Certificate} \rightarrow \text{Prop,} \\ & \text{vendorUnsignedAuthorization: Vendor} \rightarrow \text{Prop,} \\ & \text{vendorSignedAuthorization: Vendor} \rightarrow \text{Certificate} \rightarrow \text{Prop} \} \end{aligned} \quad (5)$$

3.3 Estado del Dispositivo y Eventos

El estado del dispositivo se modela con el conjunto de suites instaladas, la información de la suite activa, en caso de que haya una, los permisos otorgados o revocados en modo permanente, y las autorizaciones de acceso concedidas o denegadas a las suites instaladas.

$$\begin{aligned} \text{State} =_{\text{def}} \{ & \text{suite : SuiteID} \rightarrow \text{Prop,} \\ & \text{session: option SessionInfo,} \\ & \text{granted, revoked: SuiteID} \rightarrow \text{Permission} \rightarrow \text{Prop,} \\ & \text{authorized, unauthorized: SuiteID} \rightarrow \text{SuiteID} \rightarrow \text{Prop} \} \end{aligned} \quad (6)$$

La información de la suite activa, denotada con *SessionInfo*, comprende el identificador de la suite, junto con los permisos otorgados o revocados a ella en la sesión.

$$\text{SessionInfo} =_{\text{def}} \{ \text{id : SuiteID, granted, revoked : Permission} \rightarrow \text{Prop} \} \quad (7)$$

La validez del estado del dispositivo se establece con el predicado *Valid*, como una conjunción de condiciones que deben verificarse. La compatibilidad entre los permisos ofrecidos por el dominio de protección y los permisos solicitados por la suite, se establece con el predicado *SuiteCompatible*. La unicidad de los identificadores de suites instaladas está dada con el predicado *UniqueSuiteID*. Por otra parte, la compatibilidad entre los permisos otorgados, en una sesión o en forma permanente, y los permisos solicitados se establece con *ValidSessionGranted* y *ValidGranted*, respectivamente. Finalmente, el predicado *ValidAuthorization* establece que una suite no puede estar autorizada y desautorizada por otra suite al mismo tiempo.

$$\begin{aligned} \text{Valid} = & \text{SuiteCompatible} \wedge \text{UniqueSuiteID} \wedge \text{CurrentInstalled} \wedge \\ & \text{ValidSessionGranted} \wedge \text{ValidGranted} \wedge \text{ValidAuthorization} \end{aligned} \quad (8)$$

Los eventos relacionados con la seguridad se modelan como constructores del tipo *Event* y se detallan en la tabla 1.

Tabla 1. Eventos relacionados con la seguridad

Nombre	Descripción	Tipo
<i>install</i>	<i>instala una suite</i>	$\text{SuiteID} \rightarrow \text{Descriptor} \rightarrow \text{Domain} \rightarrow \text{Event}$
<i>remove</i>	<i>remueve una suite</i>	$\text{SuiteID} \rightarrow \text{Event}$
<i>start</i>	<i>inicia una sesión</i>	$\text{SuiteID} \rightarrow \text{Event}$
<i>terminate</i>	<i>fin de una sesión</i>	Event
<i>request</i>	<i>solicita permiso</i>	$\text{Permission} \rightarrow \text{option UserAnswer} \rightarrow \text{Event}$
<i>authorization</i>	<i>solicita autorización</i>	$\text{SuiteID} \rightarrow \text{Event}$

El comportamiento de los eventos se especifica mediante pre y pos condiciones. Las pre condiciones están definidas en términos del estado del dispositivo, mientras que las pos condiciones se definen en términos del estado anterior, el estado posterior y una respuesta opcional del dispositivo. La respuesta del dispositivo se representa con el tipo *Response*, donde el valor *allowed* denota la aceptación y *denied* el rechazo.

El evento *authorization* modela la solicitud de autorización de acceso por parte de una suite a los recursos compartidos de la suite activa. Tiene como precondition que en el estado *s* exista una suite activa y que el identificador de la suite solicitante *idReq* corresponda al identificador de una suite instalada en el dispositivo.

$$\begin{aligned} \text{Pre } s \text{ (authorization idReq)} =_{\text{def}} & \forall \text{ses: SessionInfo,} \\ & s.\text{session} = \text{Some ses} \rightarrow \exists \text{msReq: Suite,} \\ & s.\text{suite msReq} \wedge \text{msReq.id} = \text{idReq} \end{aligned} \quad (9)$$

La poscondición establece que se autoriza el acceso a la suite solicitante, si no estaba previamente desautorizada y sus credenciales coinciden con alguna declaración de autorización de la suite activa. En este caso, el estado resulta modificado ya que la suite solicitante pasa a integrar el conjunto de suites autorizadas por la suite activa.

$$\begin{aligned} \text{Pos } s \text{ s' (Some allowed) (authorization idReq)} =_{\text{def}} & \forall \text{ses: SessionInfo,} \\ & s.\text{session} = \text{Some ses} \rightarrow \neg s.\text{unauthorized ses.id idReq} \wedge \\ & \text{AccessAuthorization } s \text{ ses.id idReq} \wedge s \equiv_{\text{authorized}} s' \wedge \\ & s'.\text{authorized ses.id idReq} \end{aligned} \quad (10)$$

El predicado *AccessAuthorization* representa la evaluación de las reglas de autorización de acceso en un estado del dispositivo. Se concede la autorización cuando se verifica al menos una de las siguientes condiciones.

La primera condición verifica que el dominio de protección de la suite solicitante *msReq* satisface el predicado *domainAuthorization* de la suite *msGrn* que comparte el recurso.

$$\begin{aligned} \text{IsDomainAuthorized} := & \\ & \text{msGrn.descriptor.domainAuthorization (msReq.domain)} \end{aligned} \quad (11)$$

La segunda condición establece que el vendedor y el certificado de la suite solicitante *msReq* satisfacen el predicado *vendorSignedAuthorization* de la suite *msGrn* que comparte el recurso.

$$\begin{aligned} \text{IsVendorSignedAuthorized} := & \\ & \text{msGrn.descriptor.vendorSignedAuthorization} \\ & \quad (\text{msReq.vendor msReq.signerCertificate}) \end{aligned} \quad (12)$$

La tercera condición verifica que el certificado de la suite solicitante *msReq* satisface el predicado *signerAuthorization* de la suite *msGrn* que comparte el recurso.

$$\begin{aligned} \text{IsSignerAuthorized} := & \\ & \text{msGrn.descriptor.signerAuthorization (msReq.signerCertificate)} \end{aligned} \quad (13)$$

Finalmente, la cuarta condición verifica que el nombre de vendedor de la suite solicitante *msReq* satisface el predicado *vendorUnsignedAuthorization* de la suite *msGrn* que comparte el recurso.

$$\begin{aligned} \text{IsVendorUnsignedAuthorized} := & \\ & \text{msGrn.descriptor.vendorUnsignedAuthorization (msReq.vendor)} \end{aligned} \quad (14)$$

El caso donde se niega el acceso a la suite solicitante, se omite por razones de espacio. Por la misma causa se omite el comportamiento de los restantes eventos. El lector interesado puede consultar [9, 11] por más información.

Un resultado interesante, demostrado en la formalización, es la invariancia de la validez del estado del dispositivo con respecto a la ejecución de cualquier evento. También se demuestra que cuando una suite es desautorizada, cualquier solicitud de acceso posterior de la misma suite es rechazada. Estas propiedades y sus demostraciones están disponibles en [9].

4. Prototipo Certificado para Autorización de Acceso

La representación del modelo de seguridad, descrita en la sección anterior, posee un alto nivel de abstracción. Esta característica no condiciona posibles implementaciones y resulta adecuada para razonar sobre las propiedades del modelo. Sin embargo, para extraer un prototipo del algoritmo para autorización de acceso y certificar que el mismo satisface la conducta especificada, se debe buscar una representación concreta que sea representable en un lenguaje funcional.

4.1 Representación concreta

Sin pérdida de generalidad ni inconsistencia se asume que todos los predicados utilizados para describir el estado del dispositivo y los eventos son decidibles y tienen un dominio finito. El criterio adoptado para la transformación segura de un predicado P decidible y definido sobre un conjunto finito S es el siguiente. Cuando el predicado P caracteriza una colección de elementos, se representa como una lista exhaustiva l de elementos de tipo S que satisfacen el predicado. Por otra parte, cuando el predicado P caracteriza una propiedad sobre un conjunto de elementos, se representa como una función F de S en un tipo isomorfo a $bool$, tal que: $\forall x \in S, P x \Leftrightarrow F x = true$.

A modo de ejemplo, la representación concreta del estado del dispositivo se presenta a continuación. El conjunto de suites instaladas se representa como una lista, mientras que las autorizaciones y desautorizaciones de acceso se representan con las correspondientes funciones sobre booleanos.

$$\begin{aligned} CState =_{def} \{ & \\ & suites : list CSuite, \\ & session : option CSessionInfo, \\ & granted : SuiteID \rightarrow Permission \rightarrow bool, \\ & revoked : SuiteID \rightarrow Permission \rightarrow bool, \\ & authorized : SuiteID \rightarrow SuiteID \rightarrow bool, \\ & unauthorized : SuiteID \rightarrow SuiteID \rightarrow bool \} \end{aligned} \quad (15)$$

4.2 Algoritmo de Autorización de Acceso

En la representación concreta se propone una implementación del algoritmo para autorización de acceso. El algoritmo *AlgAccessAuthorization* recibe como parámetros el identificador de la suite solicitante y el estado concreto actual, y retorna una pareja formada por un estado, eventualmente modificado, y la respuesta a la solicitud de autorización. A partir del estado actual el algoritmo obtiene la suite activa, que es la que dispone del recurso compartido; y a partir del identificador obtiene la suite solicitante. Luego, el algoritmo procede a comparar las credenciales de la suite solicitante con las declaraciones de autorización de la suite activa.

Este análisis se descompone en varios casos, según las condiciones evaluadas. Si hay correspondencia entre la declaración de la suite activa y las credenciales de la suite solicitante, la respuesta es solicitud concedida, sino es solicitud denegada. El estado del dispositivo es modificado en la medida que la suite solicitante pasa a formar parte de las suites autorizadas o desautorizadas por la suite activa [11].

AlgAccessAuthorization (sidReq: SuiteID)(cst: CState): CState * option Response :=

```

let msGrn:= getMIDletSuiteInSession cst in      (* suite activa en la sesión *)
let msReq:= getMIDletSuite sidReq cst.suites in  (* suite solicitante *)

match (isAuthorized cst msReq) with
| true      => (cst, Some allowed)      ((* autorización existente *))
| false     =>

match (isUnauthorized cst msReq) with
| true      => (cst, Some denied)      ((* desautorización existente *))
| false     =>

match (isDomainAuthorized cst msReq) with
| true      => (mkCState (cst.suites, cst.session, cst.granted, cst.revoked,
                        addMIDletSuiteAuthorized cst sidReq, cst.unauthorized )
                , Some allowed)      ((* autorización por Dominio *))
| false     =>

match (isSigned msReq) with
| true      =>      (* suite solicitante firmada *)
  match (isVendorSignedAuthorized cst msReq) with
  | true     => (mkCState (cst.suites, cst.session, cst.granted, cst.revoked,
                        addMIDletSuiteAuthorized cst sidReq, cst.unauthorized )
                , Some allowed)      ((* autorización por Vend. y Certificado *))
  | false    =>

  match (isSignerAuthorized cst msReq) with
  | true     => (mkCState (cst.suites, cst.session, cst.granted, cst.revoked,
                        addMIDletSuiteAuthorized cst sidReq, cst.unauthorized )
                , Some allowed)      ((* autorización por Certificado *))

  | false    => (mkCState (cst.suites, cst.session, cst.granted, cst.revoked,
                        cst.authorized, addMIDletSuiteUnauthorized cst sidReq )
                , Some denied)      ((* desautorización de suite firmada *))
  end end
| false     =>      (* suite solicitante no firmada *)
  match (isVendorUnsignedAuthorized cst msReq) with
  | true     => (mkCState (cst.suites, cst.session, cst.granted, cst.revoked,
                        addMIDletSuiteAuthorized cst sidReq, cst.unauthorized )
                , Some allowed)      ((* autorización por Vendedor *))

  | false    => (mkCState (cst.suites, cst.session, cst.granted, cst.revoked,
                        cst.authorized, addMIDletSuiteUnauthorized cst sidReq )
                , Some denied)      ((* desautorización de suite no firmada *))
  end
end end end end

```

4.3 Certificación del Algoritmo

Una propiedad importante que podemos verificar en la formalización concreta es la certificación del algoritmo de autorización de acceso. El algoritmo está certificado si a partir de un estado inicial válido, en el cual se cumple la precondition del evento *authorization*, se obtiene por la ejecución del algoritmo, un estado resultante que satisface la poscondición de dicho evento. Esta propiedad se formaliza a continuación.

$$\begin{aligned} \forall (cst\ cst': CState) (msReq: CSuite) (r: option Response), \\ CPre\_authorization\ cst\ msReq.id \rightarrow \\ AlgAccessAuthorization\ msReq.id\ cst = (cst', r) \rightarrow \\ CPos\_authorization\ cst\ cst'\ msReq.id\ r \end{aligned} \quad (16)$$

La demostración se sigue por la estructura del algoritmo y está basada en el análisis de casos de las condiciones evaluadas. El lector interesado puede encontrar la demostración completa en [9].

5. Debilidad del Sistema de Autorización de MIDP 3.0

En esta sección se revisan las diferencias entre los niveles de seguridad de MIDP 3.0. De la comparación surge una debilidad en la seguridad a nivel de aplicación. Esta situación se ilustra mediante un ejemplo concreto y se aportan sugerencias para su corrección.

5.1 Comparación entre Niveles

En la especificación informal de MIDP pueden apreciarse diferencias en la definición de los niveles de seguridad. Las diferencias están en la granularidad de la protección sobre los recursos, en la modalidad en que los permisos son otorgados y en los mecanismos para establecer la confianza en las aplicaciones.

A nivel de plataforma, el dominio de protección define permisos para cada uno de los recursos sensibles del dispositivo. Por el contrario, a nivel de aplicación la declaración de autorización de acceso no diferencia entre los múltiples recursos que una aplicación puede compartir. Esto implica que una vez que se otorga autorización de acceso, la misma permite acceder a todos los recursos compartidos por la aplicación.

En segundo lugar, aunque un permiso a nivel de plataforma puede tener distintas modalidades de otorgamiento (por única vez, por sesión o permanente), la autorización de acceso a nivel de aplicación sólo es otorgada en forma permanente.

Finalmente, el grado de confianza a nivel de plataforma está basado exclusivamente en la Infraestructura de Clave Pública X.509 [12], a través de firmas y certificados digitales. Sin embargo, a nivel de aplicación no todas las declaraciones de confianza se basan en dicho mecanismo. Existe una declaración de autorización de acceso que exige como credencial a la aplicación solicitante el nombre del vendedor de la aplicación. Este atributo de la aplicación no está protegido por firma o

certificado alguno y su autenticidad no es evaluada en ningún momento. Esto deja una brecha que puede ser explotada por una aplicación maliciosa para acceder a un recurso compartido, como se muestra en la siguiente sub sección.

5.2 Un Ejemplo de Acceso Indeseado

Este ejemplo ilustra un posible escenario donde una aplicación no confiable accede a un recurso compartido por otra aplicación, presentando una credencial falsa.

El *Record Management System* (RMS) es un recurso sensible del dispositivo que permite crear y acceder a información persistida en un almacén de registros. Independientemente del dominio de protección al que resulten ligadas, todas las aplicaciones tienen permisos sobre los métodos del RMS.

En MIDP 3.0 la aplicación propietaria de un almacén de registros puede compartirlo con otras aplicaciones y restringir el acceso al mismo usando el mecanismo de autorización. En la API del RMS, el método *openRecordStore* permite crear y acceder a un almacén de registros. Dependiendo de su propósito, cada aplicación emplea una signatura distinta del mismo método.

La aplicación propietaria del almacén de registros emplea la siguiente signatura para crearlo y compartirlo en forma restringida. Usando el valor *AUTHMODE_APPLEVEL* en el tercer parámetro del método, se activa el mecanismo de control de acceso basado en autorizaciones.

```
public static javax.microedition.rms.RecordStore openRecordStore ( (17)
    String recordStoreName, boolean createIfNecessary, int authmode,
    boolean writable)
```

En el ejemplo, se crea un almacén de cuentas bancarias llamado *BankAccountsDb* con seguridad a nivel de aplicación.

```
rsPrv = RecordStore.openRecordStore ( (18)
    “BankAccountsDb”, true, AUTHMODE_APPLEVEL, false);
```

Para restringir el acceso, la aplicación debe incluir autorizaciones de acceso en su descriptor. En el ejemplo, la *declaración de autorización de acceso por vendedor* autoriza a todas las aplicaciones realizadas por el mismo vendedor *TrustyVendor* que provee la aplicación.

```
MIDlet-Name: TrustyMIDlet (19)
MIDlet-Version: 1.0
MIDlet-Vendor: TrustyVendor
MIDlet-Access-Authorization-1: vendor;TrustyVendor
```

Por otra parte, una aplicación que pretende acceder al recurso compartido debe emplear la signatura indicada abajo del método *openRecordStore*. Requiere además conocer el nombre del recurso compartido y algunos datos de la aplicación proveedora: su vendedor y su nombre público.

```
public static javax.microedition.rms.RecordStore openRecordStore ( (20)
    String recordStoreName, String vendorName, String suiteName)
```

Supongamos que un tercero no autorizado pretende obtener la información de las cuentas bancarias. Para ello desarrolla una aplicación sin firmas ni certificados digitales, empleando como nombre de vendedor el mismo nombre del propietario del almacén de registros. Un fragmento de su descriptor se detalla a continuación.

```
MIDlet-Name: MaliciousMIDlet
MIDlet-Version: 1.0
MIDlet-Vendor: TrustyVendor
```

(21)

Al no estar firmada, la aplicación maliciosa es instalada y ligada al *Dominio de Protección para Terceros No Identificados*. Esto protege a la plataforma de que la aplicación acceda a los recursos sensibles del dispositivo. Sin embargo no evita que tenga acceso al almacén de cuentas bancarias compartidas.

El siguiente código Java le permite a la aplicación no confiable abrir el almacén de registros y obtener la información.

```
// abre el almacén compartido
rsCns =RecordStore.openRecordStore(
"BankAccountsDb","TrustyVendor","TrustyMIDlet");

// procede a leer los registros del almacén
int nextID = rs.getNextRecordID();
byte[] data = null;
for( int id = 0; id < nextID; ++id ){
    try {
        int size = rs.getRecordSize( id ); // obtiene el tamaño del registro
        if( data == null || data.length < size )
            data = new byte[ size ];
        rs.getRecord( id, data, 0 ); // obtiene el registro
        processRecord( rs, id, data, size ); // procesa el registro
    } catch( InvalidRecordIDException e ){
    } catch( RecordStoreException e ){
        handleError( rs, id, e ); }
}
```

(22)

5.3 Observaciones y Alternativas

El ejemplo anterior ilustra un escenario donde una aplicación explota una debilidad del modelo de seguridad para acceder a un almacén de registros compartidos. La debilidad se explica por dos factores: el tipo de credencial exigida y el aseguramiento de la compatibilidad hacia atrás.

En primer lugar, el tipo de declaración de autorización empleada exige como única credencial el nombre del vendedor de la aplicación. Sobre este atributo no hay un mecanismo que permita determinar su autenticidad, por lo cual es fácilmente modificable por una aplicación maliciosa. Sin embargo, los demás tipos de declaración exigen como credenciales el dominio de protección o la firma y el certificado de clave pública. A diferencia del primer caso, la autenticidad de estos atributos está asegurada.

En segundo lugar, MIDP 2.0 considera a la API del RMS como un recurso sensible a nivel de plataforma, aunque no tan crítico, ya que otorga permisos de uso a una aplicación no confiable. Para asegurar la compatibilidad hacia atrás, este permiso también es otorgado en MIDP 3.0 a una aplicación no confiable.

Como alternativa a esta situación se plantean dos sugerencias. La primera es eliminar la declaración de autorización de acceso por nombre de vendedor, que exige una credencial fácilmente alterable. La segunda es revisar los permisos otorgados a las aplicaciones no confiables, como es el caso de los permisos sobre el RMS. Lo que hasta MIDP 2.0 se consideraba un recurso local a nivel de plataforma, pasa a ser un recurso de aplicación que puede ser accedido por otras aplicaciones, al ser compartido.

Más allá de este ejemplo concreto, y considerando la comparación entre los niveles del modelo de seguridad, se plantea una tercera sugerencia. La misma tiene que ver con eliminar la diferencia de granularidad entre los permisos a nivel de plataforma y las autorizaciones a nivel de aplicación. Con una declaración de autorización de acceso donde se incluya el identificador del recurso compartido se establece un mejor control. Actualmente la declaración de autorización no discrimina sobre el recurso al que se brinda acceso.

6. Conclusiones y Trabajos Futuros

El perfil MIDP define el ambiente de ejecución estándar para aplicaciones Java que se ejecutan en teléfonos celulares. Su viabilidad y éxito como plataforma móvil para aplicaciones depende en gran medida de la madurez de su modelo de seguridad.

El primer abordaje formal del modelo de seguridad de MIDP 2.0 demuestra ser una poderosa herramienta para razonar sobre las propiedades que debe cumplir cualquier implementación del estándar. Los cambios introducidos con la reciente publicación de MIDP 3.0 motivaron el desarrollo de una extensión de la especificación formal citada.

La extensión propuesta incorpora la autorización de acceso a los recursos compartidos entre aplicaciones, conservando las propiedades verificadas en la formalización anterior. A partir de un refinamiento seguro de la formalización se desarrolló un algoritmo para evaluar la autorización de acceso a los recursos de una aplicación y se certificó con respecto a la especificación del evento de seguridad involucrado. El algoritmo puede utilizarse en una técnica complementaria a la verificación formal, como es el testing de caja negra, donde oficie de oráculo frente a casos de pruebas derivados de la especificación. Este constituye un aporte trascendente para la comunidad vinculada al desarrollo de tecnologías seguras para dispositivos móviles de última generación. La formalización completa consta de una decena de módulos (archivos) Coq con unas 6000 líneas, que incluyen 130 definiciones, y 148 lemas y teoremas.

Del análisis de la especificación informal de MIDP 3.0 surgen notorias diferencias entre los niveles del modelo de seguridad. Tanto en la granularidad de la definición de la protección, como en la modalidad de otorgamiento de los permisos y, especialmente, en los mecanismos para establecer confianza en las aplicaciones, estas diferencias contribuyen a una debilidad del modelo. Se establece, con un caso

concreto, como una aplicación obtiene acceso a un recurso compartido falsificando una credencial. En este trabajo se plantearon tres alternativas para contrarrestar la debilidad analizada.

Como trabajo futuro se plantea la obtención de un prototipo ejecutable de la especificación completa del modelo de seguridad de MIDP 3.0. Haciendo uso de las ventajas que un asistente de pruebas basado en teoría de tipos, como Coq, brinda tanto para verificar sistemas como para derivar implementaciones correctas por construcción, este prototipo certificado puede ser construido.

Referencias

1. Java Platform Micro Edition, <http://java.sun.com/javame/index.jsp>, último acceso: Abril de 2009.
2. JSR 37 Expert Group: Mobile Information Device Profile for Java Micro Edition. Version 1.0. Sun Microsystems Inc. (2000).
3. JSR 118 Expert Group. Mobile Information Device Profile for Java Micro Edition. Version 2.0. Sun Microsystems Inc. and Motorola Inc. (2002).
4. JSR 271 Expert Group. Mobile Information Device Profile for Java Micro Edition. Version 3.0, Public Review Specification, Motorola Inc. (2007).
5. S. Zanella. Especificación formal del modelo de seguridad de MIDP 2.0 en el Cálculo de Construcciones Inductivas. Master's thesis, UNR, Argentina (2006).
6. S. Zanella, G. Betarte, y C. Luna. A Formal Specification of the MIDP 2.0 Security Model. T. Dimitrakos et al. (Eds.): FAST 2006, LNCS 4691, pp. 220–234, 2007. Springer-Verlag Berlin Heidelberg (2007).
7. Y. Bertot y P. Castéran. Interactive Theorem Proving and Program Development. Coq'Art: The Calculus of Inductive Constructions. Texts in Theoretical Computer Science. Springer-Verlag, (2004).
8. The Coq Development Team, The Coq Proof Assistant Reference Manual Version V8.1 (2006).
9. G. Mazeikis. MIDP Security Model, Código Coq de la Especificación Formal, 2008, <http://midpsecurity.wetpaint.com/>, último acceso: Abril de 2009.
10. G. Mazeikis, C. Luna, G. Betarte. Formalización y Análisis del Modelo de Seguridad de MIDP 3.0. Seguridad a Nivel de Aplicación, First Chilean Workshop on Formal Methods, ChWFM 2008, págs 34–43, ISBN 978-956-319-507-1, Punta Arenas, Chile, Noviembre de 2008.
11. G. Mazeikis. Formalización y Análisis del Modelo de Seguridad de MIDP 3.0. Reporte Técnico (2009). Disponible en <http://midpsecurity.wetpaint.com/>, último acceso: Abril de 2009.
12. Internet X.509 Public Key Infrastructure, <http://www.ietf.org/rfc/rfc2459>, último acceso: Abril de 2009.

Diseño Básico de la Seguridad para un Servicio Nacional de Salud Pública en Venezuela

Mireya Morales¹ y Emilio Hernández¹

¹ Universidad Simón Bolívar, Departamento de Computación,
Apartado Postal 89.000, Caracas 1080, Teléfono: +58 (212) 9063261,
Fax: +58 (212) 9063243, Venezuela
{mprimera, emilio}@usb.ve

Resumen. En este artículo se propone una infraestructura de seguridad que permite el almacenamiento de los datos de un Sistema Nacional de Salud bajo un esquema de jerarquía de memoria de tres niveles. Esto implica que las funciones de seguridad se deben tomar en cuenta en todos los repositorios distribuidos y en todos los niveles. Para ello se consideran los requerimientos de autenticación, confidencialidad y control de acceso y se describen los mecanismos de seguridad soportados con técnicas de cifrado de clave privada y clave pública, certificados, y un mecanismo de control de acceso basado en roles.

Palabras claves: Seguridad de datos, *Grids* de salud, Sistemas de información de salud.

1 Introducción

En Venezuela se ha planteado la necesidad de usar tecnologías de información y de cómputo de alto rendimiento para el acceso a datos relacionados con salud, a nivel nacional. Este planteamiento ha dado origen a un proyecto, actualmente en progreso, que propone el diseño y despliegue de un sistema nacional de información de salud. En la primera fase de este proyecto, se planteó un modelo de referencia llamado SINAPSIS (Sistema de Información Nacional Público de Salud para la Inclusión Social), el cual servirá de soporte para el Sistema de Información de Salud Nacional [1]. Actualmente este proyecto se encuentra en la fase de despliegue de un prototipo en 16 centros de salud remotos, con un servidor centralizado. No obstante se espera que en el futuro el sistema completo tenga miles de puntos de acceso y esté organizado en tres niveles, que incluyen en el nivel superior una plataforma de servicios *grid* en la que estarán replicados todos los datos del sistema nacional de salud. En el desarrollo del diseño intervinieron instituciones como el Ministerio de Salud, el Ministerio de Telecomunicaciones e Informática, el Ministerio de Ciencia y Tecnología y la Universidad Simón Bolívar, entre otras instituciones venezolanas.

Hasta este momento en Venezuela no se había abordado el desarrollo de un sistema de información nacional de salud que preste su servicio a toda la población, de modo

que esta iniciativa corresponde a un hecho sin precedentes. Con la puesta en marcha del satélite venezolano VENESAT-1 se espera tener un avance significativo en la ejecución del proyecto. Frente a este panorama se hace necesaria la adecuada protección de los datos médicos de los pacientes, por lo tanto el balance entre el servicio que ofrecerá el sistema y la protección de los datos adquiere un carácter estratégico. El diseño de la seguridad ya no puede ser un asunto de implantación sino que la seguridad es un elemento de primer nivel cuyos requisitos deben ser considerados durante todo el proceso de desarrollo del sistema [2].

En algunos países de Europa el despliegue de nodos *grid* en centros de salud, tales como hospitales, laboratorios de investigaciones médicas, administraciones de salud pública, entre otros, es aún muy restringido [3]. Esto se debe principalmente a las limitaciones presentes en el *middleware* de los ambientes *grid*, que no ofrecen las funcionalidades necesarias para asegurar la manipulación confidencial de los datos médicos. En cuanto a la estandarización, se requiere la adopción de estándares internacionales y de mecanismos de interoperabilidad para el almacenamiento de información médica, lo que implica aspectos relacionados con asegurar consentimientos y anonimización.

En este artículo se plantea un análisis en términos de los requerimientos de seguridad: autenticación de usuarios, confidencialidad de los datos y control de acceso para el sistema planteado que soportará el servicio nacional de salud. También se propone el diseño de los mecanismos de seguridad para resolver tanto la autenticación como la confidencialidad y se plantea además la conveniencia de usar control de acceso basado en roles (RBAC o *Role-Based Access Control*) como un esquema en el manejo de la autorización y gestión en el control de acceso. Esto se debe a que RBAC ha sido uno de los métodos de control de acceso más usados en numerosas aplicaciones de base de datos, incluyendo las médicas [15]. Pero aparte de esta realidad, también se ha propuesto el uso de la variante de RBAC conocida como TRBAC (*Temporal Role-Based Access Control*) [6], con la finalidad de dar soporte a aquellas operaciones relacionadas con eventos dependientes del tiempo.

El resto de este artículo se estructura de este modo: en la sección 2 se discuten algunos trabajos relacionados con soluciones a los requerimientos planteados en este artículo; en la sección 3 se describe la plataforma que dará soporte al sistema nacional de salud y se discuten los requerimientos de seguridad, así como los mecanismos que se proponen para dar la solución; en la sección 4 se presentan los mecanismos de seguridad, se identifican los actores, y se describen los procesos de autenticación que son parte del funcionamiento del sistema; en la sección 5 se describen trabajos futuros y finalmente en la sección 6 las conclusiones.

2 Trabajos Previos Relacionados

Existen propuestas relacionadas con el uso de mecanismos de seguridad para la protección de los datos en sistemas de salud. Hembroff y Cai [16] introducen la seguridad relativa a la autenticación y a la autorización en un sistema de redes de salud. Ellos proponen un repositorio centralizado para almacenar datos con un formato HL7 [17], transferidos desde varias organizaciones de salud afiliadas a una

red común. La información de los pacientes contenida en los registros médicos electrónicos puede ser accedida en forma segura utilizando una interfaz *web* a través de la red por el personal debidamente autenticado y autorizado. Kiyomoto et al. [4] propone el desarrollo de una plataforma distribuida de información médica, utilizando un esquema P2P. Cada par (hospital) comparte la información entre ellos y se provee de un sistema de alta velocidad para compartir la información a bajo costo. En ese artículo se propone una combinación de técnicas existentes, tales como cifrado simétrico y asimétrico, certificados de atributos y gestión de claves. El mecanismo de seguridad es validado asumiendo una carga típica de procesamiento de las operaciones reales.

Park y Gyu [18] han propuesto un servicio de información inteligente y seguro para pacientes basado en el modelo CC-RBAC (*Content and Context based RBAC*), que considera un contexto temporal y espacial para hospitales de próxima generación con ambientes de computación ubicua. El modelo RBAC no puede reflejar varios elementos dinámicos de ambientes ubicuos; para resolver este asunto es necesario analizar los modelos TRBAC y GTRBAC (Generalized Temporal Role-Based Access Control)[7] y aplicarlos al esquema planteado. En ese trabajo también se presentan definiciones del tipo de contexto, de las restricciones de contexto y de las políticas de control de acceso. Además se presenta una definición formal de CC-RBAC y su arquitectura. En [15] se reporta la experiencia de la implementación de RBAC dentro de una base de datos médicos existentes. El propósito de este trabajo es evaluar la conveniencia de RBAC a nivel de implementación y no a nivel de aplicación. También se demuestra que las implementaciones RBAC son factibles en aplicaciones de base de datos para un sistema de salud nacional con jerarquía de roles. Franco et al. en [19] estudian el uso del sistema SELinux en sistemas de información de salud. La propuesta se centra en la gestión de las políticas de SELinux, lo que se considera como el principal factor que limita la adopción de SELinux. También se selecciona RBAC como la opción más adecuada, dado que el acceso a los recursos debe estar basado en la función del usuario dentro de la organización.

3 Requerimientos de Seguridad para SINAPSIS

Esta sección describe los mecanismos de seguridad propuestos, así como también los requerimientos de seguridad necesarios en SINAPSIS.

3.1 Descripción del Modelo de Referencia SINAPSIS

SINAPSIS es un modelo de referencia que define el desarrollo y el despliegue progresivo del sistema de información de salud nacional [1]. El diseño de SINAPSIS tiene dos aspectos ortogonales: su definición como una jerarquía de memoria y su definición como un conjunto de módulos de software que coleccionan datos y eventualmente mueven esos datos ascendentemente en la jerarquía de memoria, con el objeto de mantener una base de datos global actualizada.

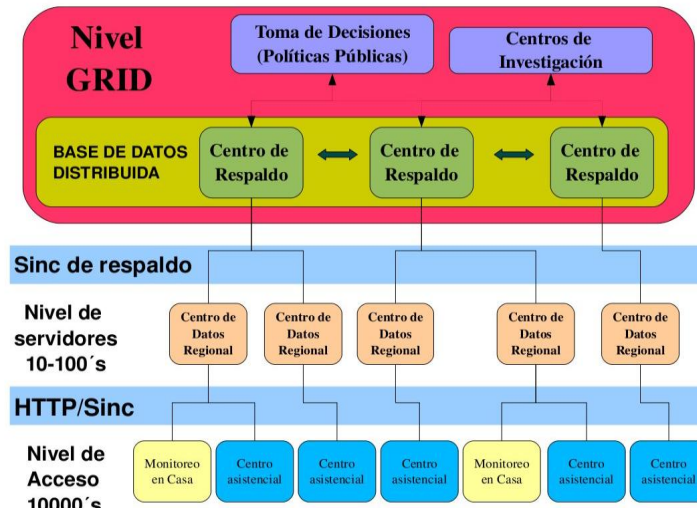


Fig. 1. Jerarquía de memoria y protocolos de sincronización.

La Fig. 1 muestra a Sinapsis como una jerarquía de memoria de tres niveles:

1. **Nivel de Acceso.** Este nivel está compuesto por los sistemas de usuario final, que permiten al personal médico interactuar con el sistema, así como también el monitoreo de sistemas que envían información de pacientes desde el hogar o desde dispositivos móviles.
2. **Nivel de Servidor Regional.** En este nivel se encuentran los centros de datos regionales, donde los datos son realmente almacenados. Este nivel en combinación con el nivel de acceso se asimila básicamente a un modelo cliente-servidor con características de tolerancia a fallos. Los servidores regionales pueden ser localizados dentro de los centros de salud, si tienen la capacidad de administrar un centro de datos pequeño. Para la mayoría de los centros de salud, el Ministerio de Salud proveerá conexiones de red para acceder al servidor regional más cercano. Los formatos de datos o el software cliente-servidor usados en los diferentes servidores regionales, puede variar de un sitio a otro.
3. **Nivel Grid de Respaldo.** En este nivel se encuentra una única base de datos, posiblemente distribuida, con la información de todos los usuarios del servicio de salud nacional, en un formato unificado. Los servidores regionales deben periódicamente sincronizar sus bases de datos con el nivel grid de respaldo, usando un protocolo de sincronización que ejecute transformaciones de formatos de datos si es necesario. En este nivel se deben proveer dos tipos de servicio a los servidores regionales:
 - **Servicio de sincronización:** para sincronizar los datos del servidor regional con el *grid* de respaldo
 - **Servicio de consulta:** es posible que un usuario vaya a un centro de salud lejano a su lugar de residencia, por ejemplo que tenga que ir

por una emergencia a un hospital en otra región. La historia médica de ese paciente se recupera desde el nivel de *grid* de respaldo, no desde el servidor regional donde están sus datos.

Niveles de Sincronización. Los niveles de comunicación deben tomar en cuenta la funcionalidad y la tolerancia a fallos. El flujo de datos entre el nivel de acceso y el nivel de servidor regional ocurrirá normalmente a través de una aplicación *web*. Sin embargo en algunos lugares en Venezuela, la comunicación debe ser vía enlace satelital, en cuyo caso la transmisión de la información se puede ver afectada por razones atmosféricas. En este sentido el sistema debe tener la posibilidad de trabajar en un modo independiente de la red, para lo que la aplicación *web* debe ser capaz de conectarse a una base de datos local y almacenar los datos temporalmente. Posteriormente, cuando se restablezca la conexión, se hará la transferencia de la información hacia el servidor regional. Eventualmente, el nivel de acceso también podría incluir sistemas relacionados con el monitoreo desde el hogar de los datos de los pacientes.

El flujo de datos entre los servidores regionales y el nivel *grid* de respaldo fluirá a través de protocolos de sincronización de las estructuras de datos usadas en ambos lados. El protocolo de intercambio y de formato de datos podría ser HL7 o una adaptación de este. Un aspecto importante es que en este diseño no se está considerando la definición de protocolos de comunicación entre los servidores regionales (no es una red P2P), lo que implica que si un paciente requiere ser atendido en un centro de salud lejos de su casa, el servidor regional asignado debe recuperar el registro médico del paciente desde el nivel de *grid* de respaldo.

3.2 Requerimientos de Seguridad

En esta sección se discuten los requerimientos que permitirán determinar los mecanismos de seguridad que deberán ser aplicados dentro de SINAPIS. Básicamente, las políticas y mecanismos de seguridad para el acceso a los datos médicos deben proveerse con mucho detalle en el nivel de servidores regionales (segundo nivel). En el nivel de *grid* de respaldo el acceso a la base de datos unificada será muchísimo más restringido, en esencia sólo los administradores de seguridad y administradores de bases de datos podrán hacerlo. Los servidores regionales podrán sincronizar sus datos con este nivel de *grid* de respaldo, pero casi ninguno de los roles de usuarios podrán acceder a los datos en el nivel de respaldo.

Los mecanismos de seguridad son en general combinaciones de funciones que sirven para mantener satisfactoriamente la seguridad de la información [4], por lo tanto la implementación del sistema nacional de salud pública propuesto requiere de la inclusión de los aspectos de seguridad necesarios para garantizar en lo posible la autenticación y el control de acceso a los datos de los pacientes. En la infraestructura planteada existe también la amenaza de que la comunicación entre los elementos de los niveles descritos pueda ser interceptada por algún usuario no autorizado, así como también que la información almacenada pueda ser adquirida ilegalmente por un intruso. En este caso debe haber una función que mantenga la seguridad de la información en forma adecuada.

En resumen, los requerimientos a ser considerados para esta infraestructura son:

1. La información transmitida por los medios de comunicación entre los diferentes niveles debe ser protegida, así como también la información que se encuentra almacenada en los repositorios de datos.
2. Se debe garantizar la autenticación de los usuarios; es decir verificar que un usuario es quien dice ser.
3. Las operaciones que el usuario ejecute sobre los datos de los pacientes deberán ser establecidas de acuerdo a los permisos asignados al rol que tiene el usuario, determinados en la política de seguridad.
4. Que las operaciones realizadas sobre los datos sean registradas para su posterior auditoria.

Para satisfacer el primer requerimiento es necesario utilizar técnicas de cifrado de la información, no sólo para la transmisión de los datos sino también para el almacenamiento [8]. Sin embargo, un aspecto crucial en la creación de un sistema seguro es la selección del método de gestión de claves que se usará en el cifrado [9].

El segundo requerimiento puede ser satisfecho a través del uso de un sistema de autenticación, en este caso se puede hacer uso de certificados de clave pública, ya que está planteado utilizar una Infraestructura de Clave Pública (Public Key Infrastructure, PKI). Con este esquema, se espera que se realice una autenticación mutua entre el solicitante y el servidor de autenticación.

Una vez autenticado el usuario, el tercer requerimiento será satisfecho a través de la implementación de RBAC, para permitir asignar la autorización correspondiente a un usuario, de acuerdo al rol que desempeñe dentro del sistema de salud y por lo tanto podrá ejecutar las operaciones sobre los datos médicos, de acuerdo a los permisos otorgados. Con el uso de RBAC, los roles representan funciones dentro de las organizaciones. Las autorizaciones son otorgadas a los roles y no a los usuarios [5], por lo tanto una vez que un usuario es autenticado y entra al sistema, puede activar un subconjunto de roles que le han sido otorgados, de acuerdo a la política de seguridad implementada.

A pesar de que RBAC ha alcanzado un buen nivel de madurez, existen algunos requerimientos no soportados por este modelo [6]. Uno de ellos está relacionado con la dimensión temporal de los roles, por ejemplo el caso de un médico que trabaje a tiempo parcial en un hospital, digamos en el horario comprendido entre las 9 a.m. y las 2 p.m. Si este médico está representado por un rol, entonces ese rol debería ser activado sólo en ese intervalo de tiempo. Para cumplir con ese tipo de requerimientos Bertino et al. [6] ha propuesto una extensión del modelo RBAC denominado Temporal-RBAC (TRBAC) que soporta restricciones para habilitar y deshabilitar roles periódicos, así como también dependencias temporales entre tales acciones. Estas dependencias son expresadas por medio de accionadores (*triggers*) de roles, es decir roles activados que son automáticamente ejecutados cuando ocurre una acción específica. También puede ser usado para restringir el conjunto de roles que un usuario particular puede activar en un instante de tiempo dado. El disparo de un *trigger* puede causar que un rol sea habilitado/deshabilitado inmediatamente o después de una cantidad de tiempo especificada explícitamente. Otro aspecto importante es que este mecanismo le permite al administrador de seguridad reaccionar frente a situaciones de emergencia, es decir, que podrá cambiar dinámicamente el estado de un rol y también el conjunto de usuarios asignados para activar ese rol

particular a través de la emisión de un requerimiento dinámico, tal es el caso de la activación del rol que corresponde al médico de emergencia. Lo más importante de TRBAC es que se pueden habilitar roles en períodos de tiempo y dependencias temporales entre roles que pueden ser expresadas a través de *triggers*. Por ejemplo un caso de uso de un *trigger* se visualiza con el hecho de que habilitar/deshabilitar el rol del médico nocturno implica con alta prioridad habilitar/deshabilitar el rol de enfermera nocturna. En conclusión, el tercer requerimiento será satisfecho utilizando un mecanismo de control de acceso soportado en RBAC y TRBAC.

El cuarto requerimiento debe ser satisfecho escribiendo un registro de todas las operaciones que se ejecuten sobre el sistema de salud. Es decir, a partir del momento de la autenticación de usuarios, asignación de roles y de las modificaciones de los datos de los pacientes. También deben registrarse las operaciones de los administradores de seguridad.

4 Diseño de los Mecanismos de Seguridad para la autenticación y la confidencialidad de los datos

Con base en los requerimientos discutidos en la sección anterior, los mecanismos de seguridad se han diseñado de la siguiente manera:

1. Los canales de comunicación en cada nivel serán cifrados utilizando una Infraestructura de Clave Pública (PKI) [10].
2. En la sección 3.1 se plantea que SINAPSIS define una jerarquía de memoria de tres niveles, esto implica que en el nivel de acceso es factible llevar a cabo un almacenamiento de los datos, esto depende de la capacidad tecnológica que haya en el centro asistencial, en el nivel de servidor regional seguro se hará almacenamiento de los datos y en el nivel *grid* de respaldo, es donde estarán los datos a nivel nacional de todos los centros regionales, lo que indica claramente que se requiere de la protección de los datos en los tres niveles. Por lo tanto, se propone el uso de un esquema de cifrado simétrico que se aplicará en el sitio donde se encuentran almacenados los datos. La ejecución de este esquema será a través de los mecanismos de cifrado que ofrezca el servidor de bases de datos que opere en cada nivel.
3. La autenticación de los usuarios se hará a través de la criptografía de clave pública, con el soporte de la infraestructura de clave pública. Este esquema de autenticación funcionará en los tres niveles de SINAPSIS.
4. El control de acceso a los datos médicos estará sujeto a la política de seguridad que se defina para el servicio nacional de salud pública; esta política deberá estar soportada legalmente en el marco de las leyes venezolanas, fundamentalmente en la Constitución de la República Bolivariana de Venezuela, en su artículo 28, Título III [11], en la Ley Especial Contra Delitos Informáticos artículos 20, 21 y 22 [12], capítulo III, , en la Ley de Mensajes de Datos y Firmas Electrónicas [13], en la Ley de Protección de Datos y Habeas Data, que en este momento se encuentra bajo la condición de anteproyecto. No obstante, se sugiere la creación de una Ley sobre Información de Salud cuyo contenido sea específico y pueda ser la

plataforma legal para la definición de las políticas de seguridad en el área de salud. Adicionalmente se espera que estas políticas estén soportadas bajo un estándar internacional, como por ejemplo ISO 27799 [20]. Otro aspecto importante a mencionar es que en el nivel de *grid* el acceso estará restringido al administrador de seguridad, o al grupo de administración de seguridad.

5. Los usuarios cuyos roles estén definidos y relacionados con funciones de Investigador y de Gestor de Datos Estadísticos para la definición de políticas públicas en salud (estos últimos serán en general empleados adscritos al Ministerio de Salud) deben tener acceso a datos consolidados, no individuales. Para los roles de Investigador y de Gestor de Datos Estadísticos, la información solicitada deberá ser sometida a un proceso de aprobación por parte de una autoridad superior que debe pertenecer al Ministerio de Salud. Posteriormente el administrador de seguridad deberá tener la competencia, establecida en un procedimiento de la política de seguridad, para garantizar la anonimización de los datos que no deben darse a conocer. Los datos autorizados deben ser extraídos de la base de datos y copiados en un medio físico diferente que será entregado a los solicitantes, nunca como una vista que acceda a la base de datos original, para evitar posibles rupturas de la protección de los datos.

4.1 Identificación de Actores

En esta sección se definen un conjunto inicial de los actores que tendrán acceso al sistema. Cada uno de estos actores tendrá privilegios bien definidos para el acceso a los datos, establecidos en la política de seguridad que soportará el sistema de salud. Los actores que en principio, tendrán interacción con el sistema nacional de salud pública son:

1. Pacientes y familiares. Normalmente acceden al sistema de salud para solicitar citas y eventualmente extraer información relacionadas con sus propios datos.
2. Personal médico, técnico y administrativo del sistema de salud: médico tratante, médico interconsultado, enfermera, personal administrativo, técnico de laboratorio, técnicos de imagenología (radiologías, tomografías, entre otras) otro profesional médico. Ellos tendrán contacto con el sistema, cuando quieran realizar operaciones de: agregar, modificar, eliminar o consultar datos vinculados con la historia médica de los de pacientes.
3. Personal técnico informático: administradores de seguridad (AS), administradores de redes y administradores de bases de datos. Tienen acceso a los datos por razones técnicas, pero deben firmar compromisos muy firmes relativos a mantener la confidencialidad de los datos. Los administradores de seguridad podrán realizar operaciones de crear usuarios, asignar roles a usuarios, asignar permisos a roles, establecer jerarquía de roles, crear restricciones, entre otras operaciones.
4. Funcionarios para toma de decisiones de políticas públicas: técnicos en estadística, personal de epidemiología, funcionarios del Ministerio de Salud, etc. No necesitan tener acceso a los datos directamente, se les pueden

suministrar conjuntos de datos consolidados por región, incidencia de enfermedades, uso de recursos hospitalarios, etc.

5. Investigadores: Tampoco necesitan tener acceso a los datos directamente, se les pueden suministrar conjuntos de datos consolidados similares a los que se entregan para tomas de decisiones en políticas públicas y adicionalmente conjuntos de datos no cuantificables, como por ejemplo imágenes médicas debidamente anonimizadas.

Metas del sistemas. Garantizar en lo posible, los requerimientos de seguridad: autenticación, confidencialidad, control de acceso, y auditoria de datos.

4.2 Procesos de autenticación

En esta sección se describen los procesos de autenticación para cada uno de los tres niveles de SINAPSIS.

Autenticación al nivel de acceso. En este nivel básicamente accede personal médico, técnico y administrativo del sistema de salud. Debe haber un mecanismo básico de autenticación, principalmente basado en los mecanismos de acceso (*login*) que tienen los equipos basados en Linux. No obstante, también se está estudiando la posibilidad del uso de una tarjeta inteligente que permita el acceso a las computadoras. En estos equipos puede haber datos temporales de pacientes que no han sido enviados al servidor regional debido a problemas de sincronización. Una vez que se subsanan las restricciones de comunicación, los datos recabados localmente se envían y se borran. Mientras esos datos estén almacenados localmente, una persona no autorizada podría acceder a ellos. Aunque este riesgo es limitado, circunscrito a un ámbito local, y con pocos datos en riesgo de estar comprometidos, debe proveerse el mecanismo básico de seguridad de acceso a los equipos.

Autenticación al nivel de servidor regional. El verdadero acceso a los datos no se realiza accediendo al equipo local, sino accediendo al servidor regional a través de alguna de las aplicaciones cliente-servidor que sean parte de SINAPSIS. En este nivel se ha planteado el uso de un esquema de autenticación mutua entre un usuario del sistema de salud y el servidor de autenticación, de modo que un usuario que ingrese al sistema de salud debe solicitar la creación de su par de claves pública y privada ante la autoridad de registro de la infraestructura de clave pública. Posteriormente le será otorgada la información, que contiene el par de claves, los datos personales del usuario y el rol que asumirá, de acuerdo al cargo que desempeña. También se emitirá un certificado de clave pública X.509, que será gestionado por la autoridad de certificación.

La red que implementa a SINAPSIS debe tener entre otros elementos una infraestructura de clave pública, es decir una autoridad de registro, una autoridad certificadora y un servidor de directorios ubicados en el nivel de servidor regional, un servidor de autenticación y un servidor de roles que estarán colocados también a este nivel.

Autenticación en los equipos de acceso al *grid* de respaldo. El *grid* de respaldo, que corresponde con el tercer nivel de SINAPSIS, debe ofrecer al menos los servicios de sincronización y de consulta, como se explica en la sección 3.1. En ambos casos, la conexión debe provenir de un servidor regional. Los servidores de sincronización y consulta del nivel de *grid* de respaldo sólo saben permitir conexiones desde los servidores regionales, y en los protocolos especificados, que en ningún caso es http, sino un protocolo basado en HL7 sobre un canal seguro. Las conexiones sobre ese canal seguro deben estar protegidas con certificados digitales y la autenticación debe ser mutua.

Autenticación dentro del *grid* de respaldo. La autenticación dentro del *grid* de respaldo será la usual dentro de un *grid* estándar: hay una doble verificación, una basada en certificados digitales asociadas a los equipos y otra basada en certificados digitales de los usuarios. Los roles de Investigador y Gestor de Datos Estadísticos pueden tener acceso a recursos del *grid*, pero no un acceso directo a la base de datos médicos. Los datos consolidados que serán usados por usuarios con estos roles deben extraerse de la base de datos y estar debidamente anonimizados.

5 Trabajos Futuros

Esta investigación debe continuar con la implementación de los mecanismos de seguridad propuestos, con la finalidad de proveer la protección de los datos médicos de pacientes en el sistema. En este momento se está trabajando en hacer una implementación del requerimiento de control de acceso a través de SELinux, debido a que este módulo incorpora RBAC. Se implementará alguna versión adaptada de TRBAC.

Se espera seguir desarrollando el resto de los requerimientos, como son: integridad, anonimización de datos, auditoria, no repudio, entre otros. Otro aspecto es el desarrollo del requerimiento de control de acceso aplicado al equipo de administradores de la red y administradores de la seguridad que estarán a cargo del servicio técnico de la plataforma del sistema de salud. Se aspira también que la gestión de los administradores del sistema de salud se haga con RBAC.

6 Conclusiones

En este trabajo se presenta un análisis de los mecanismos de seguridad para dar soporte a los requerimientos de: autenticación, confidencialidad y control de acceso en una infraestructura propuesta para un Sistema de Información Nacional de Salud (SINAPSIS). Cabe destacar que un aspecto importante es el hecho de que la seguridad ha sido tomada en cuenta como un elemento clave en el desarrollo de la infraestructura de la red que soportará el servicio de salud.

La arquitectura propuesta tiene requerimientos bien específicos desde el punto de vista de seguridad. Los esquemas de seguridad propuestos serán implementados en una infraestructura de sistema que está soportada en un diseño particular, como lo es

la jerarquía de memorias de tres niveles. Los niveles tienen diferentes requisitos de seguridad y el diseño de seguridad debe contemplar este hecho. En este trabajo se plantea una solución adaptada a cada nivel.

Se propone el uso del modelo TRBAC para implementar el control de acceso de los datos de los pacientes y se espera que esta implementación sea útil en general, no solamente para el caso específico del sistema de información de salud que se está diseñando en Venezuela.

Referencias

1. Hernández, E., Figueira, C.: Realistic Role of Grid Technologies in a National Health Service for Venezuela. First EELA-2 Conference, Bogotá, Colombia (2009)
2. Sanz, D., Aedo, I., Díaz, P.: Un Servicio Web de Políticas de Acceso Basadas en Roles para Hipermedia. IEEE Latin America Transactions, Vol. 4, No. 2, pp. 77--84 (2006)
3. SHARE Technology and Security Roadmap I, http://wiki.healthgrid.org/SHARE_Technology_and_Security_Roadmap_I
4. Kiyomoto, S., Tanaka, T., Miyake, Y., Mitamura, Y.: Development of Security Functions for a Medical Information Sharing Platform. System and Computers in Japan, Vol. 38, No. 11, pp. 49--62. Wiley Periodicals, Inc. (2007)
5. Ferraiolo, D., Sandhu, R., Gavrila, S., Kuhn, D., Chandramousli, R.: Proposed NIST Standard for Role-Based Access Control. ACM Transactions on Information and System Security, Vol. 4, No. 3, pp. 224--274 (2001)
6. Bertino, E., Andrea P., Ferrari, E.: TRBAC: A Temporal Role-Based Access Control Model. ACM Transactions on Information and System Security, Vol. 4, No. 3, pp. 191--223 (2001)
7. Joshi, J., Bertino, E., Latif, U., Ghafoor, A.: A Generalized Temporal Role-Based Access Control Model. IEEE Transactions on Knowledge and Data Engineering, Vol. 17, No. 1, pp. 4--23 (2005)
8. Blaze, M.: A Cryptographic File System for Unix. Proc. 1st ACM Conference on Communications and Computing Security, pp. 9--16 (1993)
9. Blaze, M.: Key Management in Encrypting File System. ISENIX Technical Conference, pp. 2--35, Boston (1994)
10. Nash, A., Duane, W., Joseph, C., Brink, D.: PKI Infraestructura de Claves Públicas. McGraw-Hill Interamericana, S.A. Colombia (2002)
11. Constitución de la República Bolivariana de Venezuela.: Gaceta Oficial No. 36.860. Oficio No. 0034-00 de la Asamblea Nacional Constituyente (2000)
12. Ley Especial Contra Delitos Informáticos. Gaceta Oficial No. 37.313 (2001)
13. Ley de Mensajes de Datos y Firmas Electrónicas. Gaceta Oficial No. 37.148 (2001)
14. Joon, P., Ravi, S., Gail-Joon A.: Role-Based Access Control on the Web. ACM Transactions on Information and System Security, Vol. 4, No. 1, pp. 37--1 (2001)
15. Role Based Access Control for a Medical Database, <http://www.wmin.ac.uk/cscs/pdf/ISTED SEA 2007 RBAC Slevin Macfie.pdf>
16. Hembroff, G., Cai, Y.: Secure Authentication and Authorization Design for Rural-Based Healthcare Institutions. Security and Communication Networks. Published online in Wiley InterScience, pp. 407--415 (2008)
17. HL7 Electronic Health Record Technical Committee, <http://www.hl7.org/EHR/> [2/1/2008]
18. Park, J., Gyu, D.: PIS-CC RBAC: Patient Information Service based on CC-RBAC in Next Generation Hospital considering Ubiquitous Intelligent Environment. International Conference on Multimedia and Ubiquitous Engineering. IEEE Computer Society (2007)

19. Franco, L., Sahama, T., Croll, P.: Security Enhanced Linux to Enforce Mandatory Access Control in Health Information System. In: Warren, J., Yu, P., Yearwood, J., (Eds) HDKM 2008, Vol. 80, Australia (2008)
20. Area Normas, http://iso27000.wik.is/Area_Normas

Estegoanálisis aplicado a la generación automática de estegotextos en lengua española.

Alfonso Muñoz Muñoz¹, Justo Carracedo Gallardo¹

¹ Universidad Politécnica de Madrid, E.U.I.T. Telecomunicación, DIATEL. Carretera de Valencia Km.7 – 28031. Madrid. España.
{amunoz, carracedo}@diatel.upm.es

Resumen. El presente artículo destaca las investigaciones iniciadas para analizar el potencial de la esteganografía lingüística aplicada a la lengua española, temática poco analizada para este idioma. La ocultación y detección de información enmascarada en lenguaje natural cobra una importante relevancia especialmente en entornos web 2.0 donde la información textual está más presente, si cabe, que en la “estática” red Internet, un ejemplo de ello son las redes sociales. El artículo resalta las ideas publicadas para la generación automática de estegotextos, estudiando las ventajas, inconvenientes y líneas futuras de trabajo derivadas de ellas. Se implementa una variante de un algoritmo de Peter Wayner (mimic function) en lenguaje JAVA y se analiza su utilidad esteganográfica en lengua española considerando diferentes criterios.

Palabras Claves: esteganografía lingüística, estegotextos, generación automática, CFG, NICETEXT, Peter Wayner, redes sociales.

1 Estructura del artículo.

El presente artículo está estructurado en los siguientes apartados. El segundo apartado es una breve introducción a la esteganografía y estegoanálisis, así como los trabajos previos de los autores en esta ciencia. El tercer apartado analiza el estado del arte de la esteganografía lingüística. El cuarto y quinto apartado se centra en el análisis de los mecanismos de generación automática de estegotextos. Se implementa uno de estos mecanismos y se analiza su utilidad y seguridad en lengua española. El último apartado resume los resultados obtenidos y conclusiones, avanzando posibles líneas de investigación futuras.

2 Introducción a la esteganografía. Trabajos Previos.

La esteganografía es la ciencia y el arte de ocultar una información dentro de otra, que haría la función de *tapadera o cubierta*, con la intención de que no se perciba ni siquiera la existencia de dicha información [1]. En teoría, sólo quienes conozcan

cierta información acerca de esa ocultación (un secreto) estarían en condiciones de descubrirla¹.

La ocultación de mensajes usando procedimientos esteganográficos puede tener fines legítimos o ilegítimos, que pueden ser beneficiosos para la proteger la privacidad de las comunicaciones o burlar censuras, o, por el contrario, ser vehículos para perpetrar actos criminales. Por estos motivos, en la presente década se está realizando una inversión importante en la detección de comunicaciones ocultas. El estegoanálisis es la ciencia y el arte que permite detectar esa información oculta. En general, existen dos tipos de ataques estegoanalíticos: Ataques activos y ataques pasivos. Los ataques activos se centran en la eliminación de la posible presencia de información enmascarada en un potencial estegomedio. Estas técnicas son utilizadas especialmente para atacar algoritmos de *watermarking*. Por otro lado, los ataques pasivos se centran en el estudio de los potenciales estegomédios² y la deducción de si almacenan información oculta. Los algoritmos estegoanalíticos más precisos (siglo XXI) son capaces no sólo de determinar el tamaño de la información oculta, sino de aplicar procedimientos de detección independientemente del conocimiento de la técnica de ocultación empleada, esto se denomina estegoanálisis a ciegas (blind steganalysis).³ No obstante si en el proceso de ocultación se toman las medidas de protección adecuadas, para el estegoanalista es inviable (según las publicaciones actuales) conseguir la extracción y recuperación de la información real enmascarada. En general, esta tarea difícil pertenecería a la ciencia del criptoanálisis.

2.1 Trabajo previo de los autores.

En 2007, como consecuencia del diseño de una arquitectura genérica de detección de información esteganográfica (2005), se publicó la herramienta libre Stegsecret (<http://stegsecret.sourceforge.net>) que demostraba que era viable la detección de información oculta con muchos de los programas esteganográficos disponibles en Internet (camouflage v1.2.1, inThePicture v2, JPEGXv2.1.1, técnicas EOF, etc.), así como se implementó algunas de las técnicas de estegoanálisis “genéricas” más difundidas hasta la fecha (ataques visuales, test chi-square y RS-attack). Esta investigación permitió observar los umbrales mínimos de precisión en la detección de algunos de los algoritmos de estegoanálisis más famosos. Es decir, si se oculta, debidamente, una cantidad mínima de bits los algoritmos de detección no son capaces de determinar si en un estegomedio dado hay información oculta o no. Este umbral depende de varios factores pero suele estar en torno a decenas de octetos. La idea es clara al ocultar menos información (si se utilizan técnicas que modifiquen el medio) el impacto será menor. En la actualidad, para conseguir esto se recurre al uso de

¹ En criptografía no se oculta la existencia del mensaje sino que se hace ilegible para quien no esté al tanto de un determinado secreto (la clave). Por este motivo, los mensajes que se procuran ocultar usando técnicas esteganográficas, habitualmente, son previamente cifrados.

² Se denomina estegomedio a la cubierta o medio original que se utiliza para ocultar una información.

³ El concepto de estegoanálisis a ciegas aprovecha el uso de diversos clasificadores (SVM, Fisher, etc) que mediante la definición de características propias de cada medio consiguen diferenciar entre cubiertas originales y potenciales estegomédios.

matrices de codificación⁴ y distribución de la información en varios portadores. Este último aspecto es especialmente interesante en las nuevas formas de comunicación colaborativa, donde puede dificultarse la detección de información esteganográfica mediante la distribución de información multiproveedor y multiportador. En el caso de las redes sociales, a estos factores se le suma la dificultad de automatizar el estegoanálisis de los datos publicados al existir mecanismos como: CAPTCHAS que dificultan el trabajo de software de rastreo, la existencia de grupos cerrados de usuarios, etc. La distribución multiproveedor y multiportador dificultaría la tarea si un proveedor de un servicio estuviera interesado en analizar la información almacenada por él. En este sentido se publicó en 2008 los artículos: *Herramienta DCST. Automatización de estegoanálisis en Redes Sociales y Detection of distributed steganographic information in social networks*. Para profundizar en algunos de los temas tratados véase [2].

La combinación interesante de esteganografía y redes colaborativas (por ejemplo, redes sociales, p2p o p2m) dirige nuestro interés actual. Entre las líneas de investigación los esfuerzos se centran en la ocultación y detección de información oculta en información textual en lengua española, aplicada especialmente a entornos colaborativos, para ello se está trabajando desde un punto de vista multidisciplinar, con especialistas en esteganografía y lingüistas con dominio de lengua española.

3 Estado del Arte. Esteganografía Lingüística.

Una de las técnicas más antiguas de ocultación de información, y posiblemente una excelente opción en determinadas situaciones en la actualidad, consiste en enmascarar información utilizando como estegomedio el lenguaje natural. La información textual, está presente en todo, Internet y las redes sociales es una buena muestra de ello. Este hecho hace que su utilización como estegomedio dificulte enormemente la capacidad de un estegoanalista en “separar el grano de la paja”⁵.

Las técnicas de ocultación basadas en información textual, en general, se basan en la utilización de textos existentes (se modifican) o la creación de textos de forma automática (“a medida”). La seguridad de estos estegotextos debe ser analizada desde diferentes puntos de vista, considerando ataques lingüísticos (sintáctico, semántico y de coherencia), por parte de máquinas y analistas, y ataques puramente estegonalíticos y estadísticos (análisis de entropía⁶, análisis de frecuencia de caracteres-palabras, ataques basados en conocimiento de cubierta original y cubierta modificada, etc). Estos últimos ataques pueden ser minimizados, en general para cualquier estegomedio, con algunos de los procedimientos ya comentados (matrices de codificación y distribución) u otros que compensen las perturbaciones introducidas en

⁴ El concepto de matriz de codificación hace referencia a procedimientos matemáticos que mejoran la relación información insertada-modificación de un estegomedio.

⁵ Algunas herramientas que ocultan información en texto son: *Nicetext, TextHide, TextSign, StegParty, Texto, Spammimic, Stegano, Steganosaurus, Mimicry Applet, Tyrannosaurus lex, Snowdrop*

⁶ Estudios mediante la ecuación de “entropía de Shannon” para detectar datos aleatorios (información cifrada) u otras alternativas que aproximan el valor pero son más eficientes en el procesamiento de datos, como la de Shamir y Van Someren [4].

un medio. También es útil la aplicación de ideas, más interesantes, como puede ser la negación plausible, como se verá en los siguientes apartados. En general, técnicas que necesitan una gran cantidad de información textual para ocultar un cantidad de información no muy abultada. Entre otras, porque el lenguaje natural, como estegomedio, es muy poco redundante (ruidoso) en comparación con estegomedios como son las imágenes o videos, lo cual hace más complicado la creación de algoritmos robustos de ocultación de información en lenguaje natural.

A lo largo de los siglos, no obstante, se han documentado múltiples formas de ocultación en: cartas, libros, telegramas, poesías, canciones, revistas, periódicos (por ejemplo, *newspaper code* en la época Victoriana o la verja de Cardano en el siglo XVI), en canales de mensajería instantánea (messenger, IRC), basado en el “ruido” de las traducciones automáticas, basados en lenguajes de marcado (HTML y XML), etc [2]. Es común ver su clasificación en términos de *open codes*⁷ y *semagrams*⁸, en terminología inglesa.

La década de los 90 del siglo pasado supuso el renacimiento de estos antiguos procedimientos gracias a la publicación de Peter Wayner en 1992 de las funciones *mimic* y otra serie de trabajos sobre el uso seguro de gramáticas en la generación de estegotextos [3]. Estas ideas ha evolucionado en la presente década y en la actualidad diferentes resultados esteganográficos en lenguaje natural se han obtenido para idiomas tan dispares como: inglés, ruso, mandarín, coreano, persa, etc. A pesar de esto, no existen estudios avanzados de esteganografía lingüística en lengua española, hasta lo que tenemos constancia. De la información consultada únicamente es destacable el artículo “*Using Selectional Preferences for Extending a Synonymous Paraphrasing Method in Steganography*” del mejicano Hiram Calvo y el ruso Igor. A. Bolshakov (2004) [3], aplicado a fragmentos en lengua castellana. En cualquier caso si se desea profundizar en esta temática para lengua española debe considerarse, en la actualidad, las siguientes líneas de investigación para ocultar información en lenguaje natural (textos):

1. **Generación automática de estegotextos.** Se analiza en detalle en los siguientes apartados.

2. **Generación de estegotextos basados en la modificación de textos existentes.** El mecanismo más socorrido, a lo largo de la historia, para ocultar información en lenguaje natural consiste en la modificación de textos o documentos existentes mediante diferentes procedimientos. El problema principal de estos mecanismos reside en la propia modificación, ya que si el texto original utilizado, para la ocultación, fuera accesible por un potencial estegoanalista, este hecho facilitaría un ataque clásico de comparación “texto en claro – texto codificado” que delataría la presencia de información oculta. La otra opción sería generar manualmente el texto original (o automáticamente), que no debería ser accesible, cuya modificación esteganográfica se distribuiría en el canal deseado, por ejemplo,

⁷ Los open codes genéricamente se refieren a textos de apariencia inocente, que ocultan información recuperable utilizando ciertas letras, palabras, frases del texto o comunicación. Métodos basados en esto son: Cues, Null Ciphers, Jargon Code, Grilles, etc [2].

⁸ Tipo de técnicas que consisten en la utilización (variación) de la estructura y formato de los elementos de un texto, aunque visibles, no por ello son fáciles de detectar.

mediante un artículo de opinión en un blog. En cualquier caso, cuando se desea modificar el contenido de un texto/documento con fines esteganográficos o de marcado digital debe considerarse:

a) Modificaciones léxicas. Estos procedimientos consisten en la ocultación de información mediante la sustitución/modificación de palabras. El método más analizado es la sustitución basada en el uso de sinónimos. Desde que esta idea fuera trabajada por Chapman y Davida en 1997 es considerada como una excelente opción y estudiada en diversos lenguajes [3]. El mayor problema con esta técnica es que, o no existen, o son muy pocos los sinónimos puros en una lengua, es decir, dos palabras que signifiquen exactamente lo mismo en cualquier contexto. Por este motivo, conseguir herramientas prácticas con estos principios (ya sea para ocultación de información en general o para marcado digital de textos) requiere de sofisticados mecanismos para determinar cual es la ambigüedad de una palabra dada en un contexto determinado, para saber si puede ser reemplazada o no por otra palabra. Para ello se requieren analizadores WSD⁹ robustos y estudios estadísticos que indiquen de los sinónimos disponibles para una palabra cuales son los más aconsejados.

b) Modificaciones Sintácticas y Semánticas. Los algoritmos de ocultación más robustos basados en esteganografía lingüística (se intuye) deberían ser capaces de aplicar modificaciones sintáctico-semánticas a un texto para ocultar información sin perder la coherencia y la semántica del texto. Esta investigación está completamente abierta en su aplicación a diferentes lenguas (algunos de nuestros estudios para lengua española van en esta dirección). Algunos recursos sintácticos documentados en diferentes lenguas (inglés, chino, coreano, turco, ruso, persa, etc) para ocultar información son: intercambio de voz activa/pasiva, desplazamiento en las posiciones de adverbios, orden en términos unidos por conjunciones (por ejemplo, listo y guapo o guapo y listo), etc [3]. Por otro lado, se sigue trabajando en mecanismos que aprovechándose de descripciones semánticas (ontologías aplicadas a la esteganografía) faciliten la ocultación de información considerando la semántica y coherencia de un texto. Un ejemplo sencillo consiste en la inserción de sentencias/términos semánticamente “vacíos” (no afectan al contexto), por ejemplo, en inglés la inserción de términos delante del sujeto de una sentencia como: *Basically, it seems that*, etc [3].

e) Traducciones a otros idiomas. La idea de estos procedimientos consiste en ocultar información basándose en la posibilidad de traducir una sentencia, de un lenguaje concreto, en varias sentencias “equivalentes” en un lenguaje destino, entre las cuales se puede elegir estableciendo un sistema binario de ocultación de información [3].

⁹ En lingüística computacional WSD (word sense disambiguation) hace referencia al proceso de identificar el sentido de una palabra en una frase o contexto determinado cuando dicha palabra tiene más de un significado.

f) Errores tipográficos y ortográficos. Abreviaturas y símbolos de puntuación. Estos mecanismos pueden parecer triviales desde un punto de vista lingüístico, sin embargo, pueden presentar utilidad esteganográfica si los textos creados con estas modificaciones se insertan en “canales” donde este tipo de errores sean frecuentes. Por ejemplo, incluir textos esteganográficos basados en faltas ortográficas en foros en Internet donde los textos escritos presentan muchos errores de este tipo (lo cual, por desgracia, es común). De la misma forma, se han documentado diferentes procedimientos para ocultación de información utilizando abreviaturas de palabras, por ejemplo, de manera ingeniosa en mensajes sms/mms [5]. En esta línea, los símbolos de puntuación (punto, coma, punto y coma, etc), más exactamente su colocación o no en zonas de un texto, también pueden ser utilizados para establecer sistemas binarios de ocultación. La creación de reglas generales para aplicar esta idea a los textos de un lenguaje dado no es nada sencillo. Por ejemplo, en turco una coma puede ser usada después de un sujeto si esta está relativamente distante del verbo. La inconsistencia de esta definición hace que su automatización sea difícil. [3].

g) Ocultación basada en formato. Este es el mecanismo más tradicional para ocultar información basado en el formato-estructura de un texto. Los recursos más utilizados son: uso de caracteres invisibles, separación entre líneas o palabras (por ejemplo, uso de espacios de tamaño variable entre palabras¹⁰) y codificación de información basada en sucesivos cambios del formato del texto (estilo de fuente, color, tamaño de letra, subrayado, negrita, cursiva, mayúsculas, etc) [3]. Estos mecanismos pueden favorecer la ocultación de una cantidad razonable de información pero no están exentos de problemas. A menudo, simples ataques activos (que anulan el formato) eliminan la información oculta. No obstante, en los últimos dos años, especialmente por el interés la comunidad científica china en el estegoanálisis de estegotextos, se han publicado estudios de interés. Véase por ejemplo [6][7].

En los siguientes apartados, dado que existe poca documentación al respecto, se analiza las ventajas e inconvenientes de alguno de los procedimientos de generación automática de estegotextos más conocidos en su aplicación a lengua española.

4 Generación automática de textos. Context-Free Grammar.

La generación automática de estegotextos permite la creación de textos originales que enmascaran una información oculta. Su ventaja fundamental reside en la posibilidad de crear estegotextos únicos para cada comunicación, de forma que se dificulte ataques basados en comparaciones texto original-estegotexto. En las últimas décadas los esfuerzos en este sentido se centran, principalmente, en la generación de estegotextos que imiten la gramática (sintaxis) y la estadística de un texto “típico” en una lengua concreta.

En la década de los 60 el excepcional lingüista A. Noam Chomsky postulo la gramática generativa. Esta gramática se definió como el conjunto de reglas innatas

¹⁰ El tamaño en dpi (dots per inch) que separa palabras o líneas en un texto puede ser configurado con sistemas tipográficos avanzados, como por ejemplo, TEX/LATEX.

que permite traducir combinaciones de ideas a combinaciones de palabras y en este sentido, **la gramática se convertía en un sistema combinatorio discreto que permite construir infinitas frases a partir de un número finito de elementos** mediante reglas diversas que pueden formalizarse mediante una gramática formal gobernada por normas de transformación. Según esta teoría de lenguaje formal una CFG (Context-Free Grammar) se define como una gramática en la que cada regla de producción es de la forma $v ::= w$, donde v es una variable y w es una cadena de símbolos no terminales y/o terminales [3]. En la década de los 90 Wayner [3] insistió en la posibilidad de utilizar estas construcciones (CFGs) para la generación automática de estegotextos, ya que implícitamente se generarían textos, que al menos, tendrían validez gramatical-sintáctica¹¹.

```
Variable_Inicio S ::= AB (.5) | AC (.5)
A ::= "Buenos Días" (.25) | "Buenas Tardes" (.25) | "Buenas noches" (.25) | "Hola" (.25)
B ::= "estimado amigo" C (.5) | "estimado compañero" C (.5)
C ::= "Juan" D (.25) | "Pedro" D (.25) | "Lucas" D (.25) | "Tomas" D (.25)
D ::= "quedamos algún día para" E (.5) | "dame tu número de teléfono para" E (.5)
E ::= "hablar" F (.5) | "charlar" F (.5)
F ::= Un saludo (1.0).
```

Fig. 1. Ejemplo de PCFG en lengua española en formato BNF. Ocultación máxima de 7 bits.

La ocultación de información se realiza mediante la selección de elementos concretos dentro de una regla específica, regla que es elegida mediante algún algoritmo de selección concreto. Aunque Wayner se esforzó en formalizar la construcción de CFGs seguras con utilidad esteganográfica y analizar su seguridad, es cierto que, su utilidad esteganográfica debe ser muy matizada. El primer problema, que parece insalvable, es que la gramática debe permanecer privada (emisor y receptor la deben conocer), ya que si no es así un atacante podría inferir fácilmente la información oculta. Este problema es mayor si la gramática es estática-manual y es comprometida, lo cual requiere un nuevo proceso tedioso y costoso de generación de la misma. La calidad del estegotexto depende claramente de la gramática, y si esta es estática-manual (pocas reglas) es más que probable la repetición de frases y términos en el estegotexto, facilitando a los estegoanalistas su trabajo. Aunque la gramática sea generada automáticamente de uno o más textos de referencia, conocidos por emisor y receptor, debe considerarse otros análisis al generar algoritmos esteganográficos basados en CFGs: a) Las palabras (términos) en una CFG se relacionan con sus vecinos en formas fijas. Aunque se añadan modelos estadísticos a las gramáticas (probabilistic context-free-grammars –PCFG–), para dificultar ataques de análisis, siempre existirán correlaciones mutuas si se quiere que el texto sea legible por un humano [3], b) ataques basados en estudio de terminales (información última de cada regla) ya que aunque las variaciones de texto creados puedan crecer sustancialmente con el tamaño de una gramática dada, el número de terminales está limitado por el tamaño de la gramática, lo cual significa que forzosamente, si el texto es lo

¹¹Las CFGs han jugado un papel nuclear en el diseño de lenguajes de programación y compiladores, así como en el análisis de la sintaxis del lenguaje natural. Una CFG se compone de terminales, variables y producciones.

suficientemente grande, combinaciones lineales de terminales se tienen que producir y por tanto repetir.

En la práctica resulta realmente complejo utilizar CFGs en herramientas públicas de manera robusta en la concepción actual. Un intento notorio, de los pocos destacables, fue el sistema NICETEXT, del que se pueden extraer ideas para nuevos diseños. Esta herramienta introdujo un sistema de generación automática de estegotextos [3] basado en PCFGs, que se pueden crear dinámicamente, y un procedimiento de sustitución basada en palabras categorizadas por contenido semántico. Su funcionamiento es sencillo: $NICETEXT_{D,S}(C) \rightarrow T$ y $SCRAMBLE_D(T) \rightarrow C$, siendo D un diccionario de palabras categorizadas por tipo y S (meta-style source) un conjunto de reglas de estilo de escritura basadas en las ideas de gramáticas libres de contexto. Estas gramáticas se eligen independientemente de la información a ocultar (basado en criterios estadísticos) y la información se oculta mediante la selección de las palabras concretas que correspondan a un término concreto de la regla definida. Entre las ventajas destacables de esta herramienta, se encuentra que el receptor no necesita la gramática para recuperar la información enmascarada, lo cual permite utilizar en caso extremo una gramática única por comunicación, de la riqueza que se desee, y que en el diccionario creado se utilizan palabras categorizadas semánticamente (extraída de algunas fuentes de texto y clasificada mediante analizadores morfológicos), lo que facilita la creación de fragmentos de textos con un cierto nivel semántico. A esto, se le une la posibilidad de utilizar principios de negación plausible aplicada a esteganografía lingüística [3]. En NICETEXT puede suceder que la información a ocultar no sea suficiente para completar todos los términos de una regla seleccionada de la gramática, por ello se necesita utilizar una información aleatoria para seleccionar el resto de palabras hasta completar la regla en curso. Como esta selección es intrínseca al sistema, se podría utilizar a su vez para ocultar información cifrada, de forma, que un analista no podría determinar si el texto generado de esa información aleatoria se debe al proceso natural del sistema (es plausible) o a otra información enmascarada (negación plausible).

Independientemente de las ventajas conceptuales aportadas y obviando defectos (sustituciones no válidas en contexto, anomalías entre el estilo de escritura seleccionado y el vocabulario empleado, etc) en 2008 la comunidad científica china publicó, una vez más, una serie avances en estegoanálisis lingüístico, en concreto atacando a herramientas como NICETEXT, TEXTO o basadas en cadenas de Markov, que demuestra las múltiples consideraciones a tener en cuenta. Sus ataques se basaron en la suposición que en un texto natural las palabras se distribuyen de manera no equitativa, es decir, algunas palabras se repiten frecuentemente en algunos lugares pero rara vez en otros. Basándose en esto calcularon una serie de estimadores basados en las localizaciones de las palabras en un texto bajo estudio (estimadores que luego entrenarían un Support Vector Machine). Sus resultados indican, a falta de ser contrastados con otros estudios independientes, que el ratio de detección excede del 90% para estegotextos de tamaño en torno a 5KB [10].

En resumen, las CFGs (en general, PCFGs) podrían ser utilizadas, al igual que en otros lenguajes, en lengua española (algunos ejemplos sencillos se han documentado [11]) con fines esteganográficos, pero dependiendo del volumen de información a ocultar (en torno a miles de octetos) no de un modo seguro en su concepción actual, lo cual tiraría por tierra el esfuerzo (costoso) de generar gramáticas de calidad. A

menudo, se teme que los estegotextos resultantes sufran de ataques que analicen la falta de semántica y retórica global de un estegotexto (o de sus fragmentos) que alerte la presencia de información oculta. Bien es cierto que en la actualidad las CFGs o PCFGs aplicadas a esteganografía deben ser muy depuradas, a un nivel más inicial aún, deben resistir ataques estadísticos basados en frecuencia de palabras, repetición y localizaciones de términos en un texto. Es conjeturable que en un futuro cercano estas ideas de generación automática de textos puedan ser mejoradas con los avances actuales en diferentes lenguas, por ejemplo en inglés, en sustituciones léxicas y transformaciones semánticas. Esta línea de investigación, que está en curso para lengua española, proporcionará sistemas más seguros a costa, se supone, de una menor capacidad de ocultación, frente a ataques estadísticos, sintácticos y, ahora sí, semánticos.

5 Análisis del algoritmo de imitado de Peter Wayner en lengua española. Herramienta Stelin.

Peter Wayner en la década de los 90 publicó un procedimiento de generación automática de estegotextos (T) basado en el imitado estadístico de una o más fuentes de textos (S) que es interesante analizar [3]. La idea es sencilla: Cójase una función de imitado f que modifique un fichero A de forma que asuma las propiedades estadísticas de otro fichero B. Es decir, si $p(t,A)$ es la probabilidad de que una cadena t suceda en A, entonces una función de imitado f , hace que la $p(t,f(A))$ sea aproximadamente $p(t,B)$ para toda cadena t de tamaño menor que n . La complejidad del modelo estadístico de imitado (análisis de frecuencia) depende, precisamente del orden estadístico n (orden de complejidad del algoritmo). Según esta idea, Wayner definió el siguiente algoritmo de imitado:

1. Constrúyase una lista de todas las diferentes combinaciones de n letras que ocurran en S y contabilizar el número de veces que ocurren en S.
2. Elegir una de ellas aleatoriamente que actuará de semilla inicial. Esto generará las primeras n letras de T (el estegotexto).
3. Repetir este punto hasta que se genere todo el texto deseado
 - a. Cójase las $n-1$ letras siguientes de T
 - b. Buscar en la tabla estadística (creada) todas las combinaciones de letras que comienzan con esas $n-1$ letras.
 - c. La última letra de esas combinaciones forma el conjunto de posibles elecciones para la siguiente letra que será añadida a T.
 - d. Elegir entre esas letras y usar la frecuencia de sus ocurrencias en S para “evaluar” cuál es la mejor elección.
 - e. Añadirla a T.

Por ejemplo, un primer orden de imitado genera caracteres aleatorios de acuerdo a su distribución estadística. En un segundo orden imita la distribución de parejas de caracteres de los textos S de entrenamiento, y así continuamente para mayor orden. El proceso de ocultación de información se realiza mediante la selección de las opciones

de la próxima letra a mostrar. Wayner justificó como esto se podría hacer, entre otras opciones, utilizando un árbol de Huffman, que basándose en las frecuencias de aparición de los caracteres (por ejemplo) les asignaría un código (código que se utilizará para ocultar una información). Si la selección de las ramas de este árbol (que imita la estadística a la fuente), es aleatoria el texto resultante imitará (o se aproximará) a la distribución estadística del texto fuente. Se supone (por la información publicada [3]) que para lengua inglesa, dependiendo del texto y del orden (texto de al menos decenas de KB y orden mayor que 8), pueden obtenerse algunos estegotextos con validez léxica y sintáctica, e incluso con apariencia semántica-estructural. En lengua española, por las pruebas iniciales realizadas, esta afirmación es difícil de mantener. La ocultación de unas pocas decenas de octetos producirá estegotextos con algún error léxico o gramatical. Entre otros motivos, porque la consideración del carácter como unidad atómica de entrenamiento para generar estegotextos puede producir léxico no válido en el contexto textual y esto no se soluciona al considerar textos más grandes (más posibilidades de elección) y orden de complejidad mayor (lo que implica un factor de expansión del estegotexto mayor).

“que no cese la producción y sus políticos. La seguridad y soberanía y la seguridad y soberanía alimentaria desde una posición de defensa campesinas de los poderes económicos, sociales, económicas y culturales. E) El rechazo social a las políticas de producir riqueza, el capitalista de la soberanía y la seguridad alimentaria en el mercado mundial, así como su reconocimiento recíproco como sujetos de derechos de la pobreza y la exclusión. Hoy no se producen los alimentos), homogeneizando culturas, criterios y técnicas productiva y culturales, manteniendo el control de los poderosos, el incremento de la des”

Fig. 2. Ocultación de 10 octetos de información (0xF0, 0x0F, 0xAA, 0x71, 0xF0, 0x0F, 0xAA, 0x71, 0xF0, 0x53) utilizando un texto¹² de 24KB y orden de complejidad 11. Expansión 1:61

La idea de Peter Wayner podría ser mejorada (o eso se piensa) si se considera un nivel de atomicidad de entrenamiento diferente, por ejemplo, la utilización de la palabra en lugar del carácter, dado que se espera que los resultados mejoren, al menos sintácticamente. Basado en estos criterios y en algunas mejoras adicionales, se implementada una herramienta de código libre para la generación automática de estegotextos. La herramienta Stelin, desarrollada en lenguaje JAVA, facilita el análisis del algoritmo de Peter Wayner y la variante propuesta en lengua española (<http://stelin.sourceforge.net>).

El algoritmo implementado en Stelin para imitar una fuente de texto de entrenamiento, considerando como nivel de atomicidad la palabra¹³, es el siguiente:

1. El proceso de generación se basa en el análisis de bloques de n

¹² El texto utilizado en el ejemplo es el artículo titulado: “Seguridad Alimentaria y sus condiciones de posibilidad”. Descargable de: <http://www.kaosenlared.net/noticia/seguridadalimentaria-condiciones-posibilidad>.

¹³ Otro nivel atómico sería posible: un párrafo, verso en un poema, etc. Esto condicionaría, en general, el factor de expansión (mayor) y la capacidad de ocultación (menor).

palabras extraídas del texto de entrenamiento mediante una ventana deslizante que se desplaza una posición para cada nuevo bloque. Es decir, el primer bloque tendrá los términos de 0 a n-1, el segundo bloque de 1 a n, y así sucesivamente.

2. N define el orden de complejidad del algoritmo. Lo que significa el número de palabras a considerar consecutivamente y por tanto, su aparición viene condicionada por la aparición de palabras que le preceden o suceden.

3. Las palabras se relacionan mediante nodos enlazados, en los que se contabiliza el número de veces que se han repetido en el texto de entrenamiento. Según esto, existirá una tabla raíz que almacenará todas las “palabras diferentes” que existan en el texto fuente.

Basado en los anterior, el algoritmo de generación de estegotextos, funcionaría, en general, de la siguiente manera:

a) Se selecciona una “palabra” aleatoriamente de la tabla raíz (otro criterio podría considerarse, con fines sintácticos, por ejemplo, hacer que el texto empezase por un artículo), de esta forma para un mismo texto de entrenamiento se podrían obtener diferentes estegotextos.

b) Si esta palabra no tiene sucesores (no apunta a otro nodo), se elije otro término de la tabla raíz (paso a). Si el nodo sucesor solo tiene una palabra, esta palabra se añade al estegotexto (no es posible ocultar información en este caso) y se elige el siguiente nodo disponible. Si el nodo sucesor tiene varias palabras posibles entre las que elegir se elije aquella cuya rama del árbol de Huffman, generado de las posibles palabras (y sus frecuencias), coincida con la información a ocultar, y se elige el siguiente nodo disponible.

c) Si se llega al último nodo (orden n=8, por ejemplo, 8 palabras consecutivas) se elige la última palabra seleccionada para el estegotexto y se vuelve al paso b). Este proceso se repite hasta que se genere el estegotexto que oculta la información deseada.

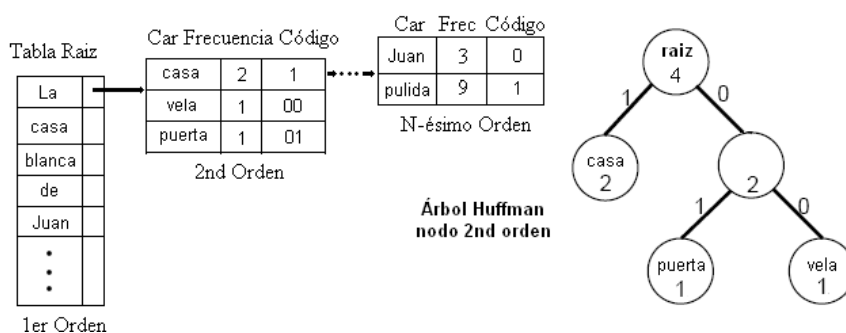


Fig. 3. Variante del algoritmo de P. Wayner . Nivel de Atomicidad = Palabra.

Esta variante, genera estegotextos de mayor tamaño (a mayor nivel de atomicidad es más probable que los elementos no tengan tantos diferentes sucesores), pero es más fácil obtener textos con validez léxica y sintáctica (no exentos de pequeños fallos), e incluso, en ocasiones, con apariencia semántica.

[INICIO TEXTO] *pesadilla. Está el sol en el ocaso. Suena el agua clara no mitiga, la amargura del tiempo de mentira, de infamia. A España toda, la luna llena, el ojo encandilado del búho insomne sueño mío! ¡Este frío de un amanecer en la tierra, y en este nuevo ejido sin duda, el amor a mujer el que llevó a un límite infranqueable la desubjetivación del sujeto. "¿Y cómo no intentar —dice Martín— devolver a mi oído, por la ventana de mi estancia, iluminada por esta luz invernal, —la tarde gris de plomo y azul de plata, con manchas de roja herrumbre, todo envuelto en luz violada. ¡Oh tierras de Alvargonzález, en el corazón de una tarde inmensa; mas falta el hilo entre los dos! Al borrarse la nieve, se alejaron los montes de la sierra. La tarde está cayendo frente a los caserones de sus lares; la tempestad llevarse los limos de una manera española, que fue casarse con una tarde clara y amplia como el hastío, cuando el eje del planeta se vence hacia el poeta admira y calla, el sabio mira y la noche azul ardía toda sembrada de estrellas. ¡Padre!, gritaron; al fondo de la laguna serena cayeron, y el eco ¡padre! repitió de peña denegrida, vuelve mi corazón a su faena, con el viento... ¡el viento de la tarde en su* **[FIN TEXTO]**

Fig. 4. Ocultación de 16 octetos (0xF0, 0x0F, 0xAA, 0x71, 0xF0, 0x0F, 0xAA, 0x71, 0xF0, 0x0F, 0xAA, 0x71, 0xF0, 0x0F, 0xAA, 0x71). Texto Fuente obra "Poesías Completas" de Antonio Machado (290KB texto plano) y orden de complejidad 9. Expansión 1:76.

En los textos mostrados en la Fig.2 y Fig.4, los estegotextos no finalizan necesariamente con una estructura puramente sintáctica. Esto puede solucionarse, por ejemplo, mostrando términos (ocultando una información aleatoria de relleno) hasta encontrar un fin de cadena (por ejemplo, un punto). Este hecho facilitaría a su vez la ocultación de una pequeña cantidad de información basado en principios de negación plausible. En cualquier caso, la selección de los textos de entrenamiento y orden de complejidad son vitales para la generación de estegotextos de calidad. Diferentes tipos de textos podrían ser considerados como fuente para ocultar información (poemas, novelas, artículos periodísticos, código de programación, etc). Si el texto fuente es más grande es más probable que existan diferentes alternativas que sucedan a una palabra y por tanto la capacidad de ocultación sea mayor (piénsese en lengua española por ejemplo en la presencia de preposiciones). Desde un punto de vista lingüístico debería, al menos, evitarse o filtrarse fragmentos de texto que claramente afecten a la coherencia en los estegotextos creados. Entre estos, índices, títulos, numeraciones (a), b), c), I, II, III), fechas, referencias, etc. Por otro lado, a falta de una mejor formalización, las pruebas realizadas indican, que en general, un orden 8 o superior proporciona unos resultados léxicos y sintácticos razonables. La cuestión está en determinar, si el esfuerzo merece la pena, si para un texto dado un orden N+1 es mejor (estadística y lingüísticamente) que un orden N o menos, ya que a mayor orden, es más probable, que el factor de expansión sea mayor y estegotexto generado (más grande) sea más "atacable". En cualquier caso, por las pruebas realizadas este procedimiento sólo permitiría la ocultación de una breve cantidad de información (decenas de octetos) de forma segura (siempre podrían distribuirse pequeñas informaciones en diferentes estegotextos). Esta cantidad de octetos permitiría el intercambio de breves mensajes de información, urls o claves criptográficas. Por ejemplo, una información de 16 octetos (128 bits) podría codificar, mediante un

alfabeto de 32 elementos (27 letras y 5 símbolos adicionales), hasta 25 letras. Por ejemplo, un mensaje de movilización como: “manifa ramblas a las ocho”.

5.1 Problemas estadísticos del algoritmo de Peter Wayner y variantes.

La seguridad de esta propuesta, como la de otras, debe ser analizada desde diferentes puntos de vista, no exclusivamente ataques lingüísticos. Un ejemplo de ello son los estudios estadísticos. En la práctica la aproximación estadística de la fuente de entrenamiento (texto) realizada por la idea de Wayner y variantes dependerá de varios factores, entre otros de la función de imitado utilizada. Por ejemplo, una posible codificación de tres caracteres (a,b,c) con probabilidades (0.89,0.07,0.04) utilizando el algoritmo Huffman sería (0,10,11). Si su función inversa es usada como función de imitado los caracteres aparecerían con frecuencia (0.5,0.25,0.25) lo cual dista de ser una aproximación estadística razonable. Esto es debido a que al utilizar un árbol binario para representar las opciones de los caracteres (u otro nivel de atomicidad) su distribución estadística de los mismos siempre será una potencia negativa de 2. Esta potencia depende de la distancia entre la raíz y la hoja correspondiente del árbol. Si existen muchos caracteres en el árbol su profundidad será mayor y la aproximación también (se han propuesto otras alternativas para aproximar mejor los valores de la fuente [3]). En general, utilizar un orden de complejidad mayor mejorará la aproximación estadística de la fuente. El límite del orden o del tamaño del texto fuente a procesar viene dado por los recursos hardware involucrados en las operaciones. Por este motivo, en la herramienta Stelin, en el caso de atomicidad basada en palabra, se implementan diferentes modos de actuación con el orden de complejidad, ya que para textos fuentes grandes es posible que el algoritmo desborde en memoria. Por ello, se puede configurar el orden concreto de almacenamiento de términos en cada nivel de recursividad del algoritmo, limitar el orden general de cada nivel a un valor fijo, etc.

La utilización de un nivel de atomicidad mayor que el carácter no tendría porque cumplir las aproximaciones estadísticas justificadas. Lo cierto es, que por los estudios en curso, al menos a nivel de carácter la variante implementada se aproxima a la distribución reflejada en la fuente de entrenamiento para órdenes grandes, 8 o más. (véase por ejemplo, Fig.5). No obstante, aparte de otros ataques, está por ver si estas propuestas son seguras frente a ataques basados en localización no equitativa de términos en un estegotexto generado [5]. En este caso concreto, por la información publicada, la propuesta estegoanalítica tendría cierto éxito en análisis sólo si los estegotextos son de al menos de unas cuantos miles de octetos, caso que no se da, por ejemplo, en Fig.2 y Fig.4.

Entre otros ataques sería interesante analizar qué sucedería si un atacante conociera el texto fuente de entrenamiento (que es secreto y compartido entre emisor y receptor) y el orden de complejidad utilizado para generar un estegotexto concreto. Si esto fuera así el atacante podría, reconstruyendo los árboles de Huffman correspondientes, decodificar cada palabra del estegotexto a un código concreto. Información binaria que podría ser analizada para descubrir la información enmascarada u observar la posibilidad de la existencia de una información cifrada (alta entropía). Por si esto sucediera, en primer lugar, la información a ocultar debería ser cifrada (en la

herramienta Stelin se utiliza un cifrador basado en el algoritmo AES-256 en modo contador -CTR mode). Además de esto, los ataques derivados de análisis de entropía y recuperación de información podrían dificultarse de diversas maneras, algunas de ellas comentadas ya en el artículo. En el caso concreto de la herramienta Stelin este hecho se dificulta aplicando ideas clásicas de cifradores basados en reducción de redundancia [12]. Stelin utiliza un generador PRNG (AES-256 en modo contador) que asigna a cada rama de cada árbol de Huffman (0 o 1) de forma aleatoria en función de una clave (no de forma fija a la rama derecha 0 o 1 y a la izquierda 1 o 0)

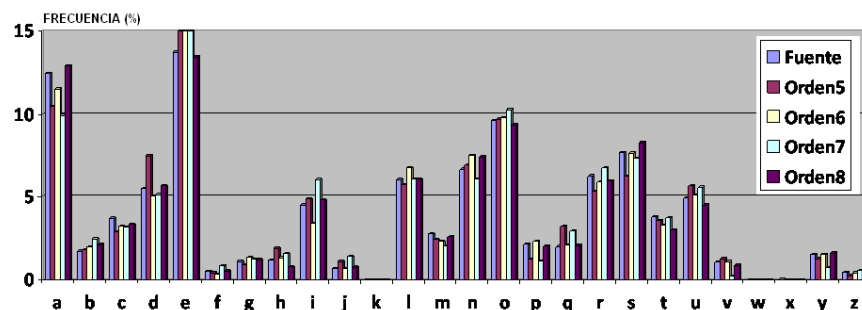


Fig. 5. Ejemplo de comparación de la distribución de frecuencias de caracteres de una fuente de texto de entrenamiento (los 13 primeros capítulos del Quijote con un total de 167.313 caracteres) y los estegotextos generados de ocultar 256 octetos (2048 bits) de información oculta a partir de dicha fuente (como nivel de atomicidad la palabra).

de forma que un atacante que conozca el texto de entrenamiento y el orden tendría dificultades en asignar un código concreto a una palabra del estegotexto determinado. Esto dificultaría, entre otras cosas, extraer la información oculta y aplicar análisis estadísticos a la misma (por ejemplo, análisis de entropía para revelar la presencia de información cifrada) [12].

6 Conclusiones. Trabajo Futuro.

El lenguaje natural es un medio poco redundante (ruidoso) en comparación con otros estegomédios más comunes (imágenes, vídeo, etc), esto hace que la ocultación de información de forma imperceptible, estadística y lingüísticamente, no sea tarea sencilla. Por ello, en el presente artículo se analizan las opciones actuales y futuras, y se profundiza en procedimientos de generación automática de estegotextos aplicado a lengua española, en concreto, PCFGs y una propuesta estadística de Peter Wayner.

La utilización de CFGs puede ser una solución de alta calidad (imperceptibilidad lingüística y estadística) en la creación de estegotextos en lengua española. Su problema principal reside en la automatización de la creación de estas gramáticas y en evitar la necesidad (por su criticidad) de que estas tengan que ser privadas y a la vez compartidas entre emisor y receptor de la comunicación enmascarada. Actualmente, la creación de herramientas públicas basadas en estos principios es cuestionable en

términos de seguridad estadística y lingüística. Una futura línea de investigación consiste en innovar en procedimientos públicos que utilizando gramáticas puedan hacer frente a los problemas analizados en el artículo.

Otra propuesta que se analiza en el artículo es un algoritmo de Peter Wayner que genera estegotextos imitando estadísticamente una fuente de texto. La propuesta original permite generar estegotextos en lengua española pero, por las pruebas iniciales, los errores léxicos y sintácticos se producen muy a menudo. Se analiza también una mejora a la idea original (con nivel de atomicidad la palabra) y se implementa en la herramienta libre Stelin (<http://stelin.sourceforge.net>). Esta variante mejora léxica y sintácticamente la generación de estegotextos en lengua española e incluso produce estegotextos con apariencia semántica. En la práctica aun con esta mejora resulta realmente complicado ocultar más de unas pocas decenas de octetos sin que el estegotexto resultante no tenga problemas, entre los más destacables, repeticiones de expresiones, semánticos o de coherencia global.

La nuevas líneas de investigación abiertas (las que se suponen más prometedoras) consisten en el análisis de las posibilidades sintácticas y semánticas del lenguaje español para la ocultación de información, así como el análisis de procedimientos robustos de sustitución léxica que puedan abrir nuevos caminos, por separado o unidos con las ideas de generación automática de estegotextos, para crear mecanismos robustos que oculten información en textos en lengua española para una información a ocultar de tamaño medio (miles de octetos).

Referencias

1. Carracedo, J.: Seguridad en Redes Telemáticas. Mc-Graw Hill InterAmericana de España. ISBN: 84-481-4157-1 (2004), páginas 123-131.
2. Muñoz, A.: My steganography investigation. <http://vototelematico.diatel.upm.es/alfonso>.
3. Bergmair, R.: A comprehensive Bibliography of Linguistic Steganography. SPIE International Conference on Security, Steganography, and Watermarking of Multimedia Contents, *volume 6505*, January 2007.
4. Shamir, A., Van Someren, N: Playing 'Hide and Seek' with Stored Keys. Lecture Notes in Computer Science. Springer Berlin. Volume 1648/1999. ISBN 978-3-540-66362-1.
5. Shirali-Shahreza, M., Shirali-Shahreza, M.H: Text Steganography in SMS. *ICCIT 2007*, Gyeongju, Korea, November 21-23, 2007, pp. 2260-2265.
6. Lingjun, L., Liusheng, H, Xinxin, Zhao., et al: A statistical attack on Kind of Word-Shift Text-Steganography. *IIH-MSP 2008*. Pages 1503-1507. 2008. ISBN:978-0-7695-3278-3
7. Lingyun, X., Xingming, S., Gang, L., Can, G: Research on Steganalysis fort text steganography based on font format. *IAS 2007*. Page 490-495. 2007. ISBN: 0-7695-2876-7.
8. Xin-guang, Sui., Hui, Luo., Zhong-liand, Zhu: A steganalysis method based on the distribution of Characters. *ICSP 2006*. 0-7803-9737-1. 2006 (IEEE)
9. Xin-guang, Sui., Hui, Luo., Zhong-liand, Zhu: A steganalysis method based on the distribution of first letters of words. *IIH-MSP 2006*. 0-7695-2745-0. IEEE
10. Zhi-li, Chen., Liu-Sheng, Huang., et al: .Effective Linguistic Steganography Detection. *IEEE 8th CIT Workshops*. 978-0-7695-3242-4. 2008.
11. Blasco, J., Hernandez, J., et al: Csteg: Talking in C code. In *Proceedings of SECRIPT International Conference*, pag. 399–406. INSTICC. Oporto. July 2008.
12. Hwang, M., A New Redundancy Reducing Cipher. *Informatica*, vol. 11, no. 4, pp. 435-440, Oct. 2000.

Metodología de Implantación de un SGSI en grupos empresariales de relación jerárquica

Gustavo Pallas y María Eugenia Corti

Grupo de Seguridad Informática, Instituto de Computación, Facultad de Ingeniería,
Universidad de la República
J. Herrera y Reissig 565, Montevideo, Uruguay
gpallas@adinet.com.uy
mcorti@fing.edu.uy
<http://www.fing.edu.uy/inco/grupos/gsi>

Resumen Un Sistema de Gestión de la Seguridad de la Información (SGSI), definido por la norma ISO/IEC 27001, no sólo debe considerar el contexto de la industria y características culturales de la Organización, sino que también debe ser sostenible en el tiempo, con capacidad de incorporar mejoras de forma incremental y continua, con un beneficio comprobable para la Organización. Para ello se requiere de una metodología bien definida que acompañe el dinamismo necesario de la empresa/Organización y de la industria y a su vez respete las estrategias empresariales y vinculación estructural.

Este artículo describe lineamientos metodológicos, de aplicación sistemática para el diseño, implantación, mantenimiento, gestión, monitoreo y evolución de un SGSI para una empresa o grupo empresarial atendiendo a su estructura de grupo multiempresa y diversidad en los niveles y dominios de seguridad requeridos.

Palabras clave: Sistema de Gestión de la Seguridad de la Información, Seguridad de la Información, ISO/IEC 27001.

1. Introducción

Cada vez existe mayor conciencia y consenso de la importancia de la Seguridad de la Información en empresas y Organizaciones, cualquiera sea el sector de la economía o rol en la sociedad que desempeñen. Sin embargo, existen estructuras empresariales que requieren que éstos temas sean analizados con una estrategia diferente, ya sea por la criticidad de la información que manejan, su dimensión o su estructura empresarial.

Grandes corporaciones o grupos empresariales, como pueden ser: empresas del sector financiero, salud, operadoras de telefonía, gubernamentales, etc., poseen una dimensión de la problemática de la Seguridad, diferente a las de una pequeña empresa. Deben tener en cuenta además, motivos legales, regulaciones y contratos, y muchas veces, la seguridad de la información es parte del propio negocio. Además de la dimensión, el factor estructural es en muchos casos un

elemento diferenciador que amerita que este tipo de empresas sea analizado de forma particular al momento de adoptar un SGSI.

La seguridad de la información, no es un activo a comprar, ni un fin en sí mismo, tampoco un estado a alcanzar haciendo una determinada inversión; debe gestionarse, deben existir metas concretas, criterios generales de evaluación y de decisión, y debe poder medirse. Es un sistema dinámico en constante evolución que debe ser evaluado y monitoreado, con métricas establecidas que permitan, comparar de manera consciente y objetiva escenarios diferentes y tomar decisiones con respecto a los riesgos que se afrontan y los recursos que se invierten. Todo esto se ve potenciado en un grupo empresarial.

Es necesario entonces lograr una metodología que conduzca a una solución eficaz y eficiente, desde el punto de vista técnico y económico, que provea los niveles de seguridad requeridos y brinde la confianza necesaria a las empresas, a los socios de negocios y a los usuarios. Esta metodología deberá considerar aspectos como las necesidades de los procesos del negocio con respecto a la información, aplicaciones y servicios, así como el uso eficiente de los recursos tecnológicos como soporte de estos procesos. Se requiere de un enfoque estratégica y económicamente proactivo y racional en la evaluación y tratamiento de riesgos, con un criterio amplio, considerando todos los aspectos y factores involucrados, pero concreto y consistente en las definiciones y decisiones más convenientes para la empresa con particular atención en la continuidad del negocio y los procesos estratégicos.

El presente trabajo aborda la problemática que se plantea al momento de implantar y gestionar un SGSI, tal como se define en la norma ISO/IEC 27001 [1], para una Organización perteneciente a un grupo empresarial.

Existen diversos trabajos, metodologías y herramientas que abordan la temática de la implementación y gestión de SGSI. Algunas de ellas como [2] son antecedentes incluso para la norma BS7799-3 [3] y por ende para la ISO/IEC 27005 [4]. Algunas declaran además de ser compatibles y cumplir los requerimientos de la norma ISO/IEC 27001, otras como [5] declaran tener objetivos diferentes a los de la norma pero igualmente ser compatibles con ésta, especialmente en la fase de Planificación, y en las otras fases, tener importantes aportes para los procesos y documentos requeridos. Por otra parte, en [6,7] se presenta una metodología de Análisis y Automatización de la Implantación de SGSI en empresas del tipo microempresas y PyMEs. Dicha metodología está enfocada “a cubrir a aquellas organizaciones que generalmente no poseen metodologías, prácticas, ni requerimientos de seguridad específicos o generales”.

Sin embargo, no hemos encontrado un trabajo específico que abarque las características y necesidades de una empresa como la que nos planteamos en el presente caso de estudio, es decir, la de un grupo empresarial constituido por una empresa principal y otra/s subordinada/s, cuyas necesidades son de naturaleza diferente a una PyME por ejemplo, debido entre otras cosas, a su estructura, dimensión y relacionamiento jerárquico.

Este artículo esta estructurado como se describe a continuación. La sección 2 describe y analiza los aspectos asociados a la implantación de un SGSI en

una estructura empresarial jerárquica. La sección 3 describe la metodología propuesta, concentrándose en la etapa de planificación. En la sección 4 se señalan las características deseables de un software de apoyo a la implementación. La sección 5 concluye.

2. Aspectos relacionados a la estructura empresarial jerárquica

Este trabajo considera la problemática que surge al tratar de implantar un SGSI en un grupo empresarial, particularmente en empresas con una relación de dependencia jerárquica, con una empresa principal y una o más empresas subordinadas que podrían funcionar como un área de negocio específica. Un ejemplo claro de este tipo de relación empresarial, es el de una empresa de Telecomunicaciones que se integra verticalmente con una empresa proveedora de servicios de Internet (ISP).

Si bien son dos empresas diferentes, existe una estructura empresarial en una relación jerárquica que se propaga o afecta a estrategias empresariales, infraestructura compartida, políticas, objetivos, recursos, etc. Esta relación no escapa a la Seguridad de la Información. Es así que, la empresa dependiente debe considerar lineamientos básicos de la empresa principal o dominante del grupo empresarial. Pero a su vez, como empresa en sí misma, tiene preocupaciones, objetivos, prioridades y riesgos propios e inherentes a su actividad, más allá de su incidencia en la empresa principal del grupo.

Pueden presentarse entonces la siguientes problemáticas:

- Los riesgos detectados y las prioridades percibidas, por la empresa principal, en la empresa dependiente, no coinciden con las prioridades y riesgos detectados por esta última. Esto puede darse porque la evaluación de prioridades local a la empresa menor, no siempre tiene en cuenta el impacto en la empresa principal, ya sea porque no están explícitamente alineados los objetivos empresariales o por falta de visión global. También puede suceder lo contrario, que globalmente no se consideren aspectos específicos o claves para la empresa cuando se baja el nivel de abstracción.
- Al existir infraestructuras compartidas, los servicios que involucran a ambas compañías pueden no estar explícitamente valuados y por ende las valoraciones de riesgo pueden no estar adecuadamente calibradas o pueden inducir a decisiones locales no alineadas con la conveniencia global.
- A veces los presupuestos de ambas empresas son independientes, y también los retornos de inversión, no se adecuan a políticas que son aplicadas por parte de la empresa principal.

Se hace necesario entonces armonizar o alinear los objetivos y prioridades en lo que a la Seguridad de la Información refiere y particularmente los SGSI de ambas empresas, de forma que, por un lado, se atiendan los lineamientos del grupo, los definidos por la empresa principal, sin ser percibidos como simples restricciones que deben cumplirse por mandato, sin un valor agregado.

Por otra parte, lograr la autonomía necesaria y la agilidad/capacidad de acción para la definición de un SGSI propio que permita gestionar la Seguridad de la Información y preservar los activos de acuerdo a las necesidades propias.

Es necesario, además, coordinar la asignación de recursos para atender prioridades del grupo y no de una sola de las empresas. Debe buscarse una metodología que permita que los SGSI de ambas empresas sean definidos, implantados y gestionados de forma cooperativa, de acuerdo a los intereses del grupo y que esta cooperación se dé de una forma eficaz y eficiente.

En la Figura 1 se ilustra esta necesidad de compatibilizar criterios en dos ejes, por un lado en el eje vertical, la jerarquía propia que responde al modelo general de seguridad establecido, hacia el interior de una empresa, donde el SGSI debe responder a los planes estratégicos y necesidades del negocio alineados con la misión y visión, respetando el marco legal vigente. Por otro lado en el plano horizontal, se refleja esta necesidad de armonizar y alinear las políticas de seguridad, hecho que se da con mayor énfasis en el SGSI de la empresa subordinada, que está a su vez comprendido o alcanzado, en su medida, por el SGSI de la empresa principal.

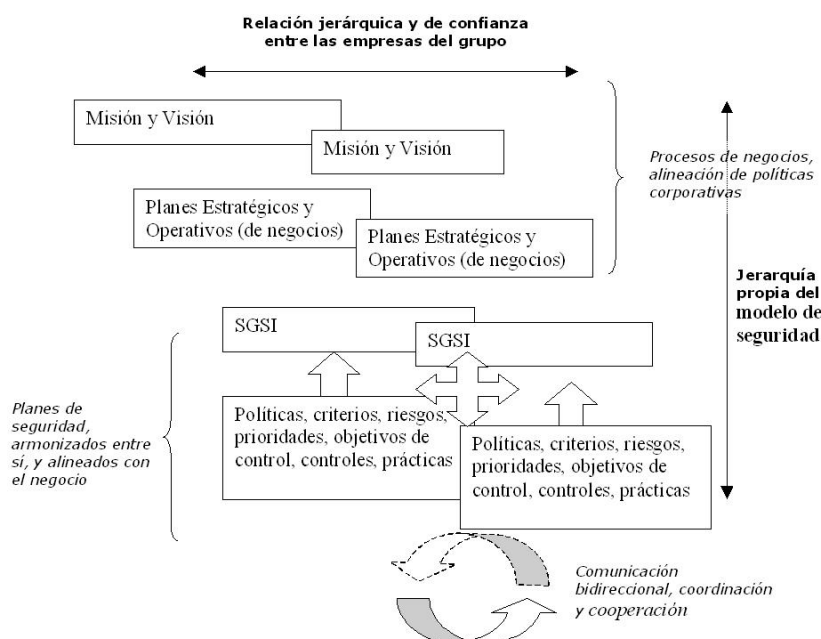


Figura 1. Posicionamiento jerárquico relativo del SGSI en un grupo empresarial

3. Metodología de implantación

En esta sección se describe la metodología propuesta para la implantación de un SGSI en un grupo empresarial con relación jerárquica, concentrándonos en la etapa de planificación del modelo PDCA (Plan Do Check Act), definido en la norma ISO/IEC 27001 [1]. Se presentan además diferentes alternativas de enfoque y estrategia y se discute su conveniencia o no.

3.1. Enfoque Centralizado, Distribuido o Mixto

La metodología definida busca ser eficiente, en recoger las directrices que vienen del SGSI de la empresa jerárquicamente mayor y a su vez, trasladar el *feedback* y las necesidades de la empresa subordinada. Por ello se hace necesario, definir y especificar la relación en términos de responsabilidades de Seguridad de la Información entre la empresa subordinada y la principal del grupo. Aquí pueden darse las siguientes alternativas:

Centralizado: Que exista un único SGSI y la Seguridad de la Información sea gestionada en forma central en el nivel más alto del grupo empresarial.

Distribuido: Que cada empresa tenga su propio SGSI y estos sean independientes.

Mixto: Un esquema mixto, donde existan dos SGSI pero relacionados e integrados. Donde se respete la naturaleza jerárquica estructural entre ambas empresas del grupo, pero también la autonomía operativa de cada una de ellas.

Un enfoque centralizado, si bien es más simple en su definición, se hace impracticable o muy complejo si la dimensión de la empresa, o grupo empresarial, es importante o de estructura más o menos compleja. La tarea de afrontar todos los riesgos y controles para todas las áreas o dominios de aplicación y todas las tecnologías parece ser muy ambiciosa e inabarcable de forma precisa, profunda y consistente.

Por otra parte, un enfoque totalmente distribuido e independiente, no parece responder a la naturaleza jerárquica y estructura empresarial. En este caso, si los planes estratégicos y objetivos no están precisamente alineados, pueden darse situaciones ambiguas, diferencias de percepción de prioridades en los riesgos o ineficiencia en la gestión y operativa de la Seguridad.

La metodología propone utilizar un enfoque mixto, de dirección centralizada pero con capacidad operativa distribuida. Un enfoque sistémico, con una visión de todo el conjunto, con sus relaciones jerárquicas y estratégicas de forma global, pero con la flexibilidad de afrontar las definiciones e implementaciones de forma más local y acotada, a los efectos de hacerlo manejable y atender las especificidades de sus ámbitos de aplicación, en términos del negocio, las tecnologías involucradas y los conocimientos requeridos.

3.2. Actividades de la etapa de Planificación

La Figura 2 ilustra la interacción y cooperación propuesta entre ambas empresas, para cada actividad del ciclo PDCA del SGSI. El objetivo es armonizar el SGSI de la empresa subordinada con el de la empresa principal del grupo.

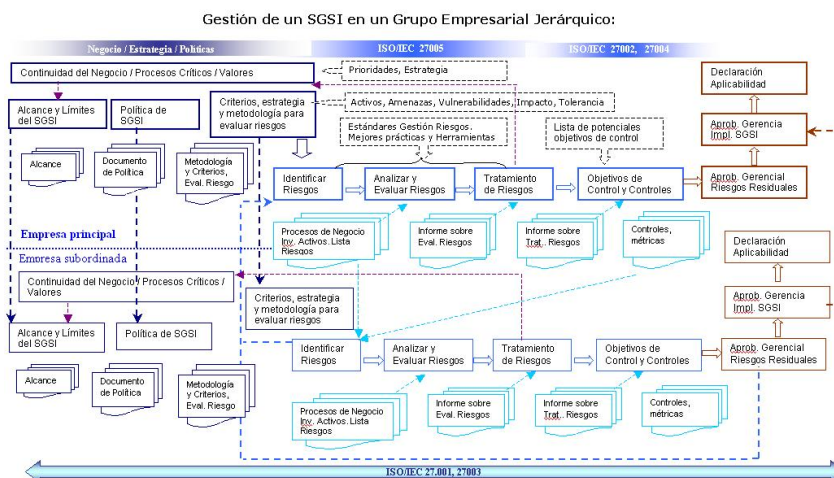


Figura 2. Propuesta metodológica

Alcance y Límites: La empresa principal, en general va a alcanzar a la empresa menor. Si esto es así, debe contemplarse el SGSI y política de la empresa principal a los efectos de ser más específicos en el detalle de políticas y controles si así se requiere y tratar las excepciones o propuestas que requieran de un tratamiento excepcional a lo que es la política general.

Política del SGSI: La empresa subordinada, deberá corresponderse o ser consistente con la Política del SGSI de la empresa principal o del grupo empresarial. Esto no quiere decir que no pueda ser más específica, de hecho muy probablemente lo será, pero deberá respetar la política general. La Política de Seguridad es una política más de la empresa, por lo tanto, la estructura del grupo empresarial y políticas generales serán las que determinen, con que autonomía y bajo que lineamientos se relacionarán ambas empresas.

Evaluación de riesgos: Si bien la evaluación de riesgos de ambas empresas puede ser independiente, si se establecen parámetros de correspondencia o fórmulas de conversión de uno a otro, a los efectos de favorecer la interoperabilidad y cooperación, es conveniente que compartan la estrategia, criterios, metodologías y eventualmente herramientas de análisis y gestión de riesgos.

3.3. Estrategia de evaluación de riesgos

Atendiendo a la necesidad de alinear las estrategias y políticas generales con respecto a la seguridad de la información, y a una conveniente relación costo/beneficio en lo global, lo más adecuado es un enfoque de alto nivel que guíe la estrategia de identificación, evaluación y priorización de riesgos.

En este enfoque global, no se debería entrar en detalles tecnológicos concretos ni particularidades del dominio de aplicación. Este análisis, debería en principio estar delegado a los especialistas del dominio o “dueños” del negocio/aplicación/información, quienes tomarán como insumo las políticas globales, los lineamientos y evaluación de riesgos con un enfoque sistémico, de alto nivel. Posteriormente, en función de los mismos, y de las necesidades concretas del dominio (su ámbito de aplicación) y conocimientos específicos, deberán hacerse las evaluaciones locales así como la definición de controles y la estrategia de implementación.

Sin embargo, atendiendo a la especificidad y complejidad de cada área o dominio de aplicación, un análisis racional con un criterio de efectividad y eficiencia, conlleva a delegar las evaluaciones y establecimientos de controles concretos así como la percepción local de riesgos adicionales (relacionados o no con los definidos a nivel global), permitiendo enriquecer la evaluación con una visión técnica o de mayor proximidad con los profesionales competentes y especializados. En este caso, puede ser necesario eventualmente, elevar determinada percepción de riesgos si es necesaria su consideración a un nivel superior. Tendríamos una implementación de controles y aporte (como retorno) del riesgo percibido *Bottom Up*.

Proponemos utilizar una estrategia de evaluación de riesgo combinada. *Top Down* en las definiciones estratégicas de alto nivel, con foco en el negocio e independiente de las particularidades de cada área y de aspectos tecnológicos concretos, y *Bottom Up* en la implementación de controles y retroalimentación del riesgo percibido por los especialistas de cada área.

De esta forma se suman los beneficios obtenidos por la adecuación a los planes empresariales globales y alineación con los objetivos del negocio y la seguridad de la información, con la efectividad y eficiencia de los aportes que deben ser realizados por las áreas que dominan la especificidad requerida para los temas de más bajo nivel (tecnológicos por ej.).

En [8] se presenta una estrategia de evaluación de riesgos “de dos etapas” que justamente combina una primer etapa de alto nivel con una etapa posterior de bajo nivel. Esta estrategia se adapta muy bien a las necesidades detectadas para el caso que nos compete. En el documento referido se presenta la evaluación de riesgos como un par de fases combinadas con distinto enfoque en cuanto a su granularidad y refinamiento sucesivo, afinando el nivel de detalle donde los activos y procesos de negocio lo ameriten.

3.4. Organización, estructura y relacionamiento

Para cada una de las etapas del SGSI, a lo largo del ciclo PDCA, debe establecerse qué roles serán necesarios y cómo se vincularán entre sí. Estos roles

no tienen por qué coincidir exactamente con el organigrama de la empresa ni del grupo empresarial, pueden incluso constituirse a los efectos de la seguridad de la información.

Quiénes deben participar en la definición del SGSI . En [9] se distingue entre tres niveles diferentes de gestión de la seguridad de la información:

Estratégico: - Dirige y Provee - Define los grandes lineamientos gerenciales para la seguridad de la información y política global del SGSI, coordina y aprueba los recursos.

Táctico: - Implementa y Optimiza - Diseño e Implementación del SGSI, establece objetivos concretos/específicos, gestiona los recursos.

Operacional: - Ejecuta y reporta - Intenta alcanzar los objetivos específicos mediante procesos técnicos.

Este enfoque sintetiza la necesidad de estructurar las definiciones de seguridad según su alcance y nivel jerárquico. También se adecua con el enfoque sistémico y la estrategia – multifase - elegida, por refinamiento desde lo global hacia dominios específicos.

Por otro lado, en [10] se presentan diferentes estructuras u organigramas posibles para la Seguridad de IT de organizaciones de diferentes dimensiones. En particular nos interesa la propuesta para organizaciones medianas y grandes. Allí se presenta la necesidad de tener un responsable de seguridad para cada orden o contexto jerárquico de la empresa, es decir: un responsable global de la seguridad de IT, un responsable de la seguridad de IT por Área, y un responsable de la seguridad por proyecto/ sistema. Adicionalmente se presenta la necesidad de un comité responsable de la coordinación de la seguridad de IT, y un equipo responsable de gerenciar la seguridad de IT. Este último coordina actividades globales a la organización, da lineamientos globales y gerencia las diferentes fases del sistema de gestión de seguridad, dando soporte a los oficiales de seguridad de los diferentes contextos (global, área, proyecto/ sistema).

Adicionalmente, la norma ISO/IEC WD 27003 [11] alineada también con la ISO/IEC (FDIS) 27005 [4], presenta las necesidades con respecto a la organización, roles requeridos y presenta también un ejemplo de organización posible.

Teniendo presente, estas recomendaciones, atendiendo a la estructura empresarial objeto de este trabajo y a las necesidades de coordinación propuestas hasta el momento entre las empresas, proponemos una estructura como la que se ilustra en la Figura 3 y cuyos roles se describen a continuación.

Dirección y Gerencia General: La alta Dirección y Gerencia es la encargada de aprobar la Política de Gestión de Seguridad de la Información, asignar los roles en materia de seguridad y coordinar el seguimiento de los planes de la Seguridad en la Organización.

Comité de Seguridad: Lleva un rol de liderazgo del SGSI, en sus lineamientos directrices y debe tener un rol ejecutivo para asegurar que se cumplan las actividades y etapas requeridas. Planifica y gestiona los recursos y el

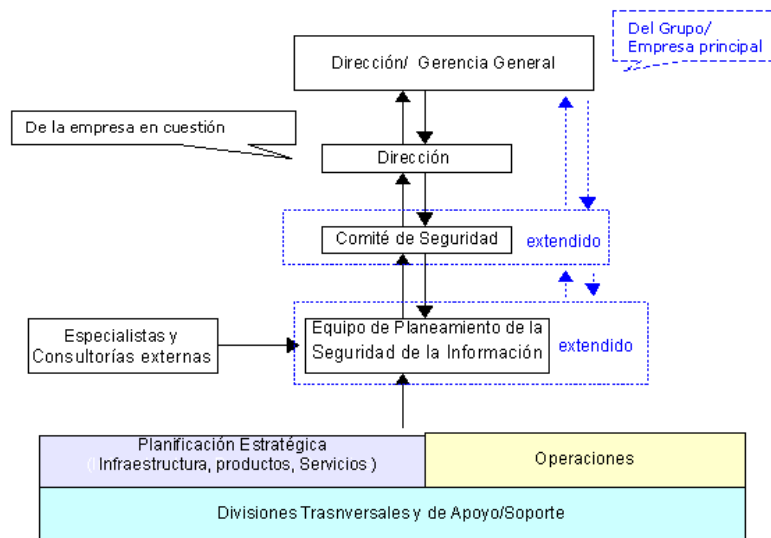


Figura 3. Esquema de Organización

ambiente para llevar adelante la gestión de riesgos y el establecimiento del SGSI. Planifica y define los documentos y contenidos del SGSI y gestiona la aprobación de la Dirección. Considera mejoras al SGSI en función de los resultados y métricas. Es deseable que estén representados aquí los intereses de seguridad de la información de todas las áreas funcionales, Divisiones y Departamentos, así como los procesos prioritarios. Debido a la estructura del grupo empresarial, y a las necesidades de alinear los SGSI de la empresa subordinada con el grupo empresarial, y que muchas veces comparten Divisiones de Apoyo o Soporte a las actividades sustantivas de la empresa, este Comité deberá integrarse tanto por profesionales de la empresa en cuestión como por responsables de la Gerencia de la Seguridad de la Información de la empresa principal del grupo (Comité de Seguridad extendido) y eventualmente de la Gerencia General si fuera necesario. Particularmente en lo que refiere a la decantación de las políticas corporativas, y los lineamientos estratégicos para arminizar ambos SGSI. Sin embargo, de acuerdo al alcance de los temas a tratar, y para darle la flexibilidad necesaria a la organización, este Comité podrá funcionar en su forma extendida o reducida según se acuerde y se requiera, y así sea convocado. De todas formas es deseable que el Comité se reúna en su modalidad extendida así sea menos frecuentemente, por ejemplo, al menos una o dos veces al año.

Equipo de Planeamiento de la Seguridad de la Información: Especialistas de Seguridad internos. Es deseable la participación y el asesoramiento profesional de especialistas de la Seguridad de la Información y sería de valor contar con ellos en la empresa (de forma interna). Deben tener buen conocimiento de los activos de información importantes de la empresa en el alcance del SGSI. Debe coordinar posibles conflictos de intereses entre Divisiones y Departamentos en lo que refiere a la Seguridad de la Información. Como se dijo anteriormente, el dimensionamiento y cantidad de personas asignadas en cada una de estas estructuras o grupos, dependerá del tamaño y tipo de la organización/ empresa. En el caso extremo por ejemplo, el Equipo de Planeamiento de la Seguridad de la Información, puede ser un Oficial/ Responsable de Seguridad que eventualmente convoca la participación de otras gerencias; manteniendo la responsabilidad sobre el primero que deberá ser un cargo independiente de las otras gerencias para preservar los intereses de cada rol y no comprometer las decisiones con diferentes intereses sobre la misma persona.

Asesoramiento y Vínculos externos: También es un aporte importante el intercambio, colaboración y asesoramiento de especialistas y grupos externos de Seguridad de la Información tanto de la industria como del sector académico (Consultoras, Facultad de Ingeniería, especialistas, etc.), de forma de seguir las tendencias de la industria, el uso de estándares, métodos y herramientas de forma de poder incorporar la evolución de la disciplina, la evolución de la industria e incorporar las lecciones aprendidas (propias y por otros). En ese sentido es muy importante el aporte y experiencias (propias y ajenas) de las organizaciones y los profesionales en torno a los incidentes de seguridad (CSIRT / CERT). En particular, los incidentes en organizaciones comparables a la empresa o grupo empresarial en cuestión, del mismo sector industrial. Debería promoverse un enfoque sistémico y multidisciplinario y no sesgado al área tecnológica o únicamente de IT.

Debe precisarse que algunas características de estos roles y estructuras, como ser su dimensionamiento, su grado de autonomía en sus funciones, su carácter de permanente o convocados a demanda, así como la frecuencia de su intervención, dependerán del tamaño y características de las empresas, su autonomía y verticalidad en la relación institucional. No obstante, es importante que la responsabilidad de la Seguridad recaiga sobre un rol que es independiente de los otros roles de gestión, a los efectos que no haya conflictos de intereses por compartir roles o responsabilidades ajenas a la seguridad por una misma persona, responsabilidades que no siempre están alineadas con la seguridad, con intereses diferentes.

3.5. Documentación

Tal como lo establece la norma ISO/IEC 27001, se requiere de un respaldo documentado de las decisiones tomadas así como una trazabilidad o correspondencia entre las decisiones de la gerencia de alto nivel en cuanto a objetivos, políticas, principios y lineamientos de la seguridad de la información, y las acciones

tomadas en referencia a la definición de procedimientos, criterios de evaluación de riesgos, registros, etc.

Los registros son la evidencia, de que los controles surgen de la evaluación de la mejor alternativa como consecuencia del análisis de riesgos y responden a objetivos de control fijados previamente. Los registros permiten hacer esta cadena de decisiones y definiciones, además de trazables, reproducibles y sirven además como referencia en la comunicación y coordinación con la empresa principal del grupo.

Estructura de la documentación Es deseable que un SGSI sea claro, preciso y fácil de entender. Por otra parte, debe comprender a todos los requisitos de la norma y todo lo que se defina en su alcance y la política misma.

Por todo lo expuesto en la necesidad de un enfoque combinado con necesidades de actualización con diferentes niveles de frecuencia y con contenidos de naturaleza diferente en su relevancia y detalles técnicos, resulta natural proponer una estructura de documentación, que se adapte a estas características. Un documento principal con los lineamientos básicos, dados por la Dirección, más estables en el tiempo y que guía la cobertura de todos los aspectos del SGSI y las etapas sucesivas, y una serie de documentos anexos, que eventualmente variarán con mayor frecuencia y requieren de análisis más detallado y de conocimiento específico del dominio.

Más allá de las revisiones inevitables, esta estructura propuesta tiene por objetivo lograr que el documento no pierda rápidamente vigencia, y lograr consistencia en la granularidad de la profundidad conceptual y niveles de abstracción o detalle en lo que a seguridad refiere. Se gana así en claridad y en la propia mantenibilidad del mismo.

El documento principal no debería contener detalles propios de cada dominio de aplicación, ni de instrumentación de políticas específicas ni tampoco procedimientos o detalles operativos. Es conveniente separar esta documentación específica en documentos de soporte al documento principal. De esta manera tendremos:

Un documento maestro o principal del SGSI

Documentos de soporte al SGSI:

Políticas técnicas (de dominio, alcance y cometido específico)

Procedimientos

Guías e instructivos de operación

En el caso general, el documento maestro o principal del SGSI de la empresa principal y el de la empresa subordinada coincidirán, de forma que compartirán los lineamientos macro de la política de la seguridad de la información. Eventualmente podrían ser documentos diferentes, pero en cualquier caso, el documento maestro del SGSI de la empresa subordinada deberá respetar y comprender al documento maestro de la empresa principal. El mismo podrá contener algún agregado o ser más específico dado que su dominio de aplicación es menor, pero

en ningún caso deberá incluir políticas específicas ni procedimientos y prácticas tecnológicas, que deberán ser documentos anexos y de soporte.

Entendemos que esta estructura brinda los siguiente beneficios:

- Simplifica la lectura y facilita la comprensión, separando los conceptos o lineamientos y política general del SGSI por un lado, de los detalles de políticas específicas en documentos diferentes.
- Separa los conceptos medulares, que varían menos frecuentemente, de las políticas y detalles de instrumentación específicos, de variación más frecuente.
- Flexibilidad a la hora de modificar, revisar el SGSI. Facilita su mantenimiento, revisión y auditoría.
- Permite complementar e integrar políticas específicas fácilmente sin alterar la estructura del documento principal.
- Facilita la capacitación, difusión y entendimiento.
- Favorece la estrategia *Top Down* en el nivel macro, alineándose con los planes de negocios y dirección de la empresa y a su vez la efectividad de definir los controles y políticas específicas en los niveles propios de su dominio (controles y procedimientos *Bottom up* en su instrumentación).

4. Características deseables de Software de Apoyo al SGSI

En esta sección, se describen las características del software que apoye el establecimiento, gestión y mejora de un SGSI en una empresa de las características que nos ocupa. No se especifican formalmente los requerimientos, sino tan sólo se brindan una lista de funcionalidades deseables de acuerdo a lo analizado en las secciones anteriores.

El software referido debería brindar funcionalidades que permitan:

- Heredar todos los lineamientos, metapolíticas y recomendaciones aplicables ya sean de la ISO/IEC (UNIT) así como aquellas más específicas del área o gobierno.
- Facilitar la armonización y eficiencia del proceso de acoplamiento e integración del SGSI de la empresa subordinada con el de la empresa principal sin duplicar costos, ni controles innecesariamente, y ayudar a establecer una correspondencia (y cumplimiento en lo que corresponda) entre las políticas establecidas para la empresa principal y la/s subordinada/s.
- La Gestión de Documentos con definición de roles, gestión de permisos de consulta y modificación, de forma que la documentación sea tratada como un activo más, preservando sus requerimientos de Confidencialidad, Integridad y Disponibilidad.
- Funcionalidades de workflow donde se puedan identificar roles, etapas de relevamiento, definición, comunicación y delegación, supervisión, escalamiento, etc., que comprenda la naturaleza jerárquica del grupo empresarial.

- Heredar clasificaciones de activos y riesgos realizadas por la empresa principal del grupo, que alcancen particularmente a activos y procesos dentro del dominio del SGSI de la empresa subordinada. Estas clasificaciones podrán ser ponderadas por parte de ésta última, en función de la jerarquía establecida y la criticidad percibida en forma corporativa (o por la empresa principal) o podrán tratarse como un mínimo aceptable (cota inferior) en el caso que las clasificaciones heredadas sean obligatorias. El software debería ser configurable, y tratar esto mediante coeficientes de ponderación, roles y permisos para cambiar ciertos valores asignados.
- Identificar y clasificar los procesos y sus activos asociados, valorizando los mismos en función de la escala establecida en cuanto a su necesidad de: confidencialidad, integridad y disponibilidad. Puede resultar interesante filtrar el grafo (o matriz) de dependencias para los procesos de negocios de determinada categoría (por ejemplo críticos o estratégicos, de alta prioridad).
- Analizar riesgos aplicando una estrategia combinada como la descrita. Es decir, una estrategia Top-Down en los lineamientos de alto nivel y corporativos que pueda refinarse de forma sucesiva, pero también la posibilidad de elevar ciertos riesgos detectados y percibidos a niveles técnicos y de dominios de aplicación específica que requieran ser considerados a un nivel superior.
- Enfocar el análisis y tratamiento de riesgos en cualquiera de los atributos interesantes para el análisis de seguridad: confidencialidad, integridad y disponibilidad. Es decir, posibilidad de filtrar o visualizar el grafo de dependencias y valoración de procesos y activos según el atributo de seguridad que se desea analizar. Esto puede realizarse si el software ofrece la posibilidad de visualizar layers o subgrafos a este nivel.
- Ayudar a detectar e identificar procesos y activos críticos así como ayudar a lograr eficiencia operativa y eficacia en el establecimiento de controles y salvaguardas atendiendo a las dependencias entre procesos y activos.
- Realizar revalorizaciones de activos de forma periódica
- Facilitar el establecimiento y comparación de escenarios de riesgos, ya sea con el transcurso del tiempo (evolución), o en etapa de planificación, evaluación y diseño de controles.
- Establecer vistas y trabajar por capas de abstracción, en el sentido de estar alineado con la Recomendación ITU-T X.805 [12], gestionando layers (Infraestructura, Servicios y aplicaciones), planos (gerencial, control y usuario final) y las ocho dimensiones de seguridad establecidas en la Recomendación.

5. Conclusiones

Un grupo empresarial, con una estructura de relación jerárquica o de subordinación, requiere de una metodología que permita gestionar la seguridad de la información y que la misma considere su estructura empresarial y su relacionamiento vertical. Es necesaria la cooperación de los SGSIs de las empresas, en todas las fases del ciclo PDCA, fundamentalmente en la fase de Planificación,

pero a su vez cada SGSI requiere de la flexibilidad y agilidad operativa suficiente para alcanzar los niveles de seguridad necesarios y los objetivos propuestos respetando los lineamientos corporativos.

Proponemos un enfoque mixto, de dirección centralizada pero con la autonomía necesaria a nivel de cada dominio y cada empresa, fundamentalmente en la gestión de controles y en la percepción del impacto de los riesgos locales. Esto permitirá aunar criterios y optimizar recursos cuando los riesgos deban afrontarse en forma conjunta. Entendemos que debe primar fundamentalmente un enfoque costo/ beneficio orientado a las necesidades de seguridad de la información del negocio, y que a los efectos de (re)dimensionar adecuadamente el alcance del SGSI de manera efectiva, se opta por una estrategia multifase, de refinamiento sucesivo, desde lo global a los dominios locales. Esta estrategia determinará en una primera etapa los procesos y activos críticos, y permitirá, en fases posteriores dedicar el esfuerzo y recursos a los activos y procesos que así lo ameritan, dotando así a la metodología de eficiencia además de eficacia. El redimensionamiento del alcance, es necesario para dar cumplimiento a la norma ISO/IEC 27001 en cuanto a la clasificación de todos los activos alcanzados por el SGSI. A los efectos de alinear el modelo de negocios, puede resultar de utilidad, la adopción de estándares, eventualmente específicos para el sector.

Atendiendo a la estructura empresarial, a su naturaleza de relacionamiento jerárquico entre las empresas, pero también a una estructura física muchas veces distribuida, sería muy importante poder disponer de un software que apoye la metodología definida, fomentando esta coordinación y cooperación, con procedimientos e hitos bien definidos, promoviendo los lineamientos directrices corporativos y una retroalimentación de las fuentes de seguridad de la información específicos de las diferentes áreas.

Un enfoque y estrategia de planificación *Top Down*, favorece la alineación de criterios, compatibiliza los modelos de negocio, los lineamientos de carácter estratégico y política de alto nivel de la seguridad de la información, permitiendo una fácil armonización de los SGSIs, separando lineamientos corporativos de políticas específicas de dominio. Esto es aplicable tanto a la metodología de evaluación de riesgos como a la estructura de la documentación.

En cada etapa del ciclo PDCA del SGSI cada una de las empresas del grupo debería tener la mayor información posible a los efectos de aplicar criterios y lineamientos corporativos o ajustar los parámetros adecuados para la percepción y estimación de riesgos, tanto locales como corporativos. Debería permitir escalar riesgos locales percibidos, así como heredar los riesgos percibidos y priorizados desde los estratos superiores, de manera de alinear los planes de Seguridad de la Información de la forma más conveniente en función de los recursos, objetivos concretos y condicionantes.

Referencias

1. International Organization for Standardization. Information technology - Security techniques - Information security management system - Requirements (ISO/IEC 27001:2005), 2005.

2. CRAMM. Information security toolkit. <http://www.cramm.com>, 04 2009.
3. British Standards Institute. Information Security Management Systems.Guidelines for information security risk management (BS 7799-3:2006), 2002.
4. International Organization for Standarization. Information technology - Security techniques - Information security risk management (ISO/IEC 27005:2008), 2005.
5. MEHARI. Information risk analysis and management methodology. <https://www.clusif.asso.fr/en/production/mehari/>, 04 2009.
6. María Eugenia Corti. *Análisis y automatización de la implantación de SGSI en Empresas Uruguayas*. Tesis de maestría, Universidad de la República, Facultad de Ingeniería, 2006.
7. G. Betarte M. E. Corti, R. de la Fuente. Hacia una Implementación Exitosa de un SGSI. In *Actas del 3º Congreso Iberoamericano de seguridad Informática (CIBSI'05)*, pages 457–471, 2005.
8. Krzysztof Lisek Andrzej Bialas. Integrated, business-oriented, two-stage risk analysis. In *Proceedings of the International Multiconference on Computer Science and Information Techonology*, pages 617–628. Institute of Innovations and Information Society, 2007.
9. ISM3. Information Security Management Maturity Model. <http://www.ism3.com/>, 04 2009.
10. Federal Office for Information security - Germany. BSI Standard 100-2 IT-Grundschutz Methodology, 04 2009.
11. International Organization for Standarization. Information technology - Security techniques - Information security management system implementation guidance (ISO/IEC FCD 27003), 2005.
12. Telecommunication Standardization Sector (ITU-T). Recommendation X.805 Security architecture for systems providing end-to-end communications. <http://www.itu.int/rec/T-REC-X.805-200310-I/en>, 2009.

Verificación Formal de la Equidad de un Protocolo de Firma de Contratos mediante Colored Petri Nets

M. Magdalena Payeras-Capellà, Macià Mut-Puigserver, Andreu P. Isern-Deyà,
Josep L. Ferrer-Gomila, Llorenç Huguet-Rotger

Universitat de les Illes Balears, Spain
{mpayeras, macia.mut, andreupere.isern, jlferrer, lhuguet}@uib.es

Resumen. Un protocolo de firma de contratos es un protocolo de intercambio equitativo de valores en el que las partes intercambian sus firmas sobre el contrato. Existen diversos protocolos de firma de contratos, usualmente acompañados de una verificación informal. En este artículo utilizamos un modelo basado en Colored Petri Nets para verificar formalmente la equidad de un protocolo de firma de contratos existente. En el análisis automático hemos modelado el protocolo y los roles de los firmantes, la tercera parte de confianza, firmantes maliciosos así como el rol de un usuario intruso. El análisis ha permitido detectar tres casos en los que el protocolo genera evidencias contradictorias y, finalmente, se ha definido el comportamiento de un árbitro que permite resolver el intercambio de forma equitativa incluso ante la presencia de evidencias contradictorias.

Palabras Clave: Protocolo de firma de contratos, Verificación formal, Petri Nets

1. Introducción

La firma de contratos es un caso particular de un protocolo de intercambio equitativo, en el que dos o más firmantes se comprometen a firmar un contrato previamente acordado, con la condición que a su finalización, todos los firmantes tengan el contrato firmado o ninguno de ellos lo tenga. Por tanto, un protocolo equitativo siempre debe proporcionar un trato igualitario a todos los participantes. Además, debe generar evidencias que prueben el comportamiento de los usuarios, para que, en caso de disputa, un árbitro externo pueda evaluar las evidencias y tomar una decisión sin ambigüedades.

Los protocolos de firma de contratos normalmente hacen uso de una Tercera Parte de Confianza (TTP) que ayuda a los usuarios a finalizar satisfactoriamente el intercambio. La TTP puede involucrarse en diferente medida en la ejecución del protocolo, actuando en cada paso, o bien solo interviniendo cuando los usuarios lo requieran. Usando este último tipo de TTP, se han definido varios protocolos eficientes y optimistas [1] que requieren solo tres mensajes para finalizar su ejecución. El protocolo de Micali [2] y el protocolo FPH [3] cumplen estas dos propiedades.

Con el objetivo de verificar formalmente un protocolo de intercambio equitativo, se han ido utilizando técnicas manuales, complicadas de aplicar y que deben ser adaptadas en cada caso, como el uso de Strand Spaces [4]. Recientemente, Sornkhom y Permpoontanalarp [6] han aplicado un método automatizado para analizar la seguridad del protocolo de Micali mediante el uso de Coloured Petri Nets (CPN) [7]. El método permite la demostración de vulnerabilidades del protocolo de Micali descubiertas anteriormente, además de detectar dos nuevos ataques al protocolo de Micali.

En este artículo vamos a aplicar un nuevo modelo de verificación formal y automatizado de protocolos de firma de contratos, basado en [6], que permitirá analizar con mayor rapidez y generalización las propiedades de los protocolos de intercambio equitativo. Como primer paso, mediante la aplicación de dicho modelo vamos a verificar formalmente la equidad del protocolo FPH.

El artículo se organiza de la siguiente forma: la sección 2 resume el protocolo FPH y sus características de seguridad. La sección 3 incluye la descripción del modelo de simulación utilizando CPNs. La sección 4 presenta el análisis de la equidad del protocolo y, finalmente, la sección 5 incluye las conclusiones y describe algunas líneas de trabajo futuro.

2. Protocolo de Firma de Contratos FPH

2.1. Propiedades Ideales del Protocolo de Firma de Contratos

Las soluciones prácticas para firma de contratos requieren de la existencia y la posible participación de una TTP. Para que el protocolo sea eficiente, generalmente se desean conseguir tres objetivos:

1. Reducir la participación de la TTP.
2. Reducir el número de mensajes intercambiados.
3. Minimizar la necesidad de operaciones costosas y de almacenamiento de gran volumen de información cuando la TTP deba actuar.

El primer objetivo es alcanzado en varias soluciones optimistas [1, 8, 9, 10], en donde la TTP solo actúa en caso de conflicto entre las partes para garantizar la equidad del protocolo. En referencia al número de mensajes intercambiados entre dos partes se establece que el mínimo son tres para las propuestas optimistas [1].

Además de los anteriores objetivos, un protocolo tiene que proporcionar evidencias a las partes implicadas que prueben, a la finalización del intercambio, si el contrato ha resultado firmado. Otras propiedades adicionales que deberían ser cumplidas por protocolos optimistas son [9, 10]:

- **Efectividad.** Si las partes implicadas actúan correctamente obtendrán los elementos deseados sin la participación de la TTP.
- **Equidad.** Ningún participante debe obtener una situación de ventaja en ninguna de las etapas de la ejecución del protocolo.

- **Asincronía.** Los participantes pueden decidir cuando finalizar la ejecución del protocolo, sin límites de tiempo.
- **No repudio.** Los participantes no pueden negar sus acciones.
- **Verificabilidad de la TTP.** Si la TTP actúa incorrectamente, todas las partes afectadas deben ser capaces de demostrarlo.

2.2 Descripción del Protocolo de Firma de Contratos FPH

El protocolo asume que Alice y Bob se han puesto de acuerdo en el texto del contrato C antes del inicio del intercambio. A continuación firman dicho contrato usando el protocolo. El canal utilizado para el envío de información entre los firmantes es un canal inseguro, por lo que no puede asumirse que los mensajes enviados a través de este canal llegan a su destinatario. El canal entre los firmantes y la TTP es un canal elástico, es decir, los mensajes llegaran en algún momento a su destinatario aunque el instante de llegada no puede ser predecido.

El iniciador, $A(lice)$, y el destinatario, $B(ob)$, intercambiarán las evidencias de no repudio directamente. Solo en el caso de que no puedan obtener el elemento esperado de la otra parte los firmantes contactaran con la $TTP(T)$, iniciando los subprotocolos *cancel* o *finish*. La notación y los elementos utilizados en la descripción del protocolo se resumen en la tabla siguiente:

NOTACIÓN	
X, Y	concatenación de dos mensajes X e Y
$H(X)$	Función de hash unidireccional resistente a colisiones del mensaje X
$S_i(X)$	Firma digital del mensaje X con la clave privada del usuario i (utilizando una función de hash, $H()$, para crear el resumen de X)
$i \rightarrow j: X$	El usuario i envía el mensaje X al usuario j
ELEMENTOS	
M	mensaje que contiene el contrato a firmar; el contrato especifica la identidad del iniciador, $A(lice)$, y la del destinatario, $B(ob)$
$h_A = S_A(M)$	Firma de A sobre el contrato M
$h_B = S_B(M)$	Firma de B sobre el contrato M
$ACK_A = S_A(h_B)$	Firma de A sobre h_B ; reconocimiento de que A sabe que el contrato ha sido firmado. Parte de la evidencia de no repudio (NR) para B
$ACK_T = S_T(h_B)$	Firma de la TTP sobre h_B ; reconocimiento equivalente al que A debería haber enviado
$h_{AT} = S_A[H(M), h_A]$	evidencia de que A ha solicitado la intervención de la TTP
$h_{BT} = S_B[H(M), h_A, h_B]$	evidencia de que B ha solicitado la intervención de la TTP

El subprotocolo de *intercambio* es el siguiente:

1. $A \rightarrow B$:	M, h_A
2. $B \rightarrow A$:	h_B
3. $A \rightarrow B$:	ACK_A

Si se completa la ejecución del protocolo, el iniciador A poseerá la evidencia de no repudio (NR), h_B , y el receptor B tendrá la evidencia de no repudio, h_A y ACK_A , por lo que el protocolo es efectivo. Si no es el caso, A o B , o ambos, deberán resolver la situación no equitativa iniciando los subprotocolos *cancel* o *finish*, respectivamente. Si A afirma que no ha recibido el mensaje 2 de B , A puede iniciar el subprotocolo de cancelación siguiente:

IF (<i>finished</i> = verdadero)	1'. $A \rightarrow T$:	$H(M), h_A, h_{AT}$
	2'. T :	recupera h_B
	3'. $T \rightarrow A$:	h_B, ACK_T
ELSE	2''. $T \rightarrow A$:	$S_T(\text{"cancelled"}, h_A)$
	3''. T :	almacena <i>cancelled</i> = verdadero

La TTP verificará la corrección de la información proporcionada por A . Si es errónea la TTP enviará un mensaje de error a A . Si es correcta, procederá de una de dos formas posibles. Si la variable *finished* es *verdadero*, significa que B había contactado previamente con la TTP (ver párrafo siguiente). La TTP ha proporcionado evidencia de NR a B , ACK_T , y ahora debe proporcionar la evidencia correspondiente a A . La TTP recupera el elemento almacenado, ACK_T , y lo envía a A , junto con un elemento que pruebe su implicación en la resolución de la disputa, h_B' . Si B no ha contactado con la TTP anteriormente, la TTP enviará un mensaje a A para cancelar el intercambio, y almacenará esta información (*cancelled* = *verdadero*) para satisfacer futuras peticiones de B . En cualquier caso, la situación final es equitativa.

Si B afirma que no ha recibido el mensaje 3, B puede iniciar el siguiente subprotocolo *finish*:

IF (<i>cancelled</i> = verdad)	2'. $B \rightarrow T$:	$H(M), h_A, h_B, h_{BT}$
	3'. $T \rightarrow B$:	$S_T(\text{"cancelled"}, h_B)$
ELSE	3''. $T \rightarrow B$:	ACK_T
	4''. T :	almacena <i>finished</i> = verdad. y h_B

La TTP también verificará la veracidad de la información proporcionada por B . Si ésta no es correcta la TTP enviará un mensaje de error a B . Si, en cambio, es correcta,

la TTP procederá de una de dos formas posibles. Si la variable *cancelled* es *verdadero*, significa que *A* ha contactado previamente con la TTP (ver párrafo anterior). La TTP ha enviado el mensaje a *A* para cancelar la transacción, y ahora debe enviar un mensaje similar a *B*. Si *A* no ha contactado con la TTP anteriormente, la TTP enviará el elemento de NR, ACK_T , a *B*. En este caso la TTP almacenará la evidencia NR, h_B , y asignará el valor *verdadero* a la variable *finished*, para satisfacer peticiones futuras de *A*. de nuevo, en cualquier circunstancia, el intercambio se encuentra en una situación equitativa.

Como conclusión, el protocolo es equitativo y no se han utilizado límites temporales, por lo que el protocolo es asíncrono.

2.3 Análisis Informal de la Equidad y el No Repudio en el Protocolo de Firma de Contratos FPH

Después de una ejecución completa del protocolo (con o sin la intervención de la TTP), pueden aparecer disputas entre los firmantes. Nos podemos encontrar con dos tipos de disputas: el repudio de *A* (*B* afirma que el contrato está firmado) y el repudio de *B* (*A* afirma que el contrato está firmado).

Un árbitro externo (que no forma parte del protocolo) tiene que evaluar las evidencias proporcionadas por las partes para resolver estos dos tipos de disputas. Como resultado, el árbitro determinará quien dice la verdad. El árbitro tiene que saber quien actúa como originador (información que aparece en el contrato).

En caso de repudio de *A*, *B* afirma que ha recibido la firma sobre el contrato *M* de *A* y debe proporcionar la siguiente información al árbitro: *M*, h_A y ACK_A o ACK_T . El árbitro comprobará si h_A es la firma de *A* sobre *M*, y si es correcta el árbitro asumirá que *A* ha enviado su firma a *B*. Entonces el árbitro comprobará si ACK_A es la firma de *A* sobre h_B , o comprobará si ACK_T es la firma de la TTP sobre h_B . Si se supera la verificación, el árbitro asumirá que *A* o la TTP había enviado un reconocimiento a *B*. Entonces el árbitro dará la razón a *B*. Si no es así, y una o ambas de las comprobaciones previas falla, el árbitro rechazará la demanda de *B*. Si la evidencia aportada por *B* prueba que tiene la razón, y *A* presenta el mensaje $PR_T[H(\text{"cancelled"}, h_A)]$, entonces la TTP o *A* actuaron de forma incorrecta.

En caso de repudio de *B*, *A* afirma que *B* ha firmado el contrato *M*. Debe proporcionar la siguiente información al árbitro: *M* y h_B . El árbitro comprobará si h_B es la firma de *B* sobre *M*, y si es correcta el árbitro asumirá que *B* ha recibido *M* y h_A , y que se ha comprometido a obtener el reconocimiento, ACK_A o ACK_T . Si las verificaciones previas fallan, el árbitro rechazará la demanda de *A*. Si la verificación es positiva, el árbitro deberá interrogar a *B*. Si *B* proporciona un mensaje *cancel*, significa que *B* ha contactado con la TTP, y que la TTP observó que *A* ha ejecutado el subprotocolo *cancel*. Por esta razón la TTP envía un mensaje *cancel* a *B*. Ahora se ha demostrado que *A* ha intentado engañar. Por tanto, el árbitro rechazará la demanda de *A*, y el árbitro dará la razón a *B*. Si *B* no puede proporcionar el mensaje *cancel*, el árbitro dará la razón a *A*. Como conclusión, el protocolo satisface los requisitos de equidad y no repudio.

Este análisis informal no cubre todas las situaciones derivadas de la ejecución del protocolo. Puede ser completado mediante una verificación formal del protocolo

(incluída en la sección 4) utilizando el modelo basado en Petri Nets descrito en la sección 3.

3. Descripción del Modelo Usado para la Verificación Formal de Protocolos de Intercambio Equitativo

3.1 Colored Petri Nets

CPN (Coloured Petri Nets) [7] es un lenguaje de modelado que combina las características y propiedades de las Petri Nets ordinarias con un lenguaje de programación de alto nivel llamado ML (Modeling Language) [7]. Las Petri Nets aportan las primitivas para la interacción de procesos, mientras que por su parte, el lenguaje de programación proporciona la definición de los tipos de datos y la posibilidad de manipularlos. CPN puede ser usado como un método formal de análisis de sistemas distribuidos y protocolos de comunicaciones. Una red CPN es un modelo ejecutable que representa los estados de un sistema a lo largo de una línea temporal. Una CPN contiene cuatro tipos de componentes:

- **Estados.** Representan el estado del sistema en un instante determinado. Los estados cambian a partir de la activación de las transiciones.
- **Transiciones.** Se corresponden con una acción que hace cambiar el estado del sistema.
- **Arcos.** Son los enlaces entre los estados y las transiciones.
- **Color sets.** Son testimonios que van pasando a través de los estados y las transiciones, y que tienen un determinado valor, llamado color.

El estado global del sistema modelado, para cada instante después de la generación de un evento, determina lo que se llama *marking*. Un *marking*, por tanto, es equivalente a una fotografía del estado del sistema tras cada evento que hace cambiar el mismo. Una de las herramientas que implementan CPN es CPNTools [7], la cual es usada en el desarrollo del presente trabajo. Se trata de una herramienta que permite construir CPN de forma rápida y visual. Una vez el modelo está diseñado, se puede someter a simulación, proceso mediante el cual se genera el *state space*. *State spaces* es el conjunto de *markings* del sistema entre el evento inicial y el final. Por tanto, mediante este procedimiento se obtiene una definición completa del comportamiento del sistema a lo largo de su ejecución.

3.2 Suposiciones Iniciales y Metodología

Para usar CPN con el objetivo de modelar un protocolo, hay que definir una serie de consideraciones generales: cada participante posee un identificador único, cada participante conoce, como paso previo a la ejecución del protocolo, las claves públicas del resto de partes firmantes, los algoritmos criptográficos usados son seguros, los mensajes enviados entre la TTP y cualquier participante siempre son entregados al destino sin modificaciones (canal elástico).

La metodología a seguir para construir el modelo y analizar la equidad del protocolo es la siguiente:

1. **Construir el modelo:** Declarar los *color sets* para representar los mensajes y los elementos del protocolo. Crear la red de alto nivel para modelar el conjunto del protocolo incluyendo los participantes. Crear la red a nivel de entidad para modelar el comportamiento interno de cada participante. Crear el nivel de proceso para cada nivel de entidad. Declarar funciones y variables que serán usadas en el modelo.
2. **Generar el *state space*:** definir el *marking* inicial para cada participante. Generar el *state space* del modelo usando CPNTools.
3. **Crear las funciones** para buscar estados de ataque.
4. **Extraer escenarios de ataque** mediante la evaluación de los *markings* obtenidos.

3.3 Descripción del Modelo

El modelo, basado en lo planteado por Sornkhom y Perpoontanalarp [10], define cuatro participantes: A(lice), B(ob), I(ntruso) y TTP. Mientras la TTP es estrictamente honesta, las otras partes pueden seguir un rol malicioso. *A* y *B*, mediante sus roles maliciosos (A_m y B_m , respectivamente), pueden parar el intercambio o contactar con la TTP en diferentes etapas del protocolo, distintas a las definidas por el mismo, para intentar engañar a la otra parte. *I* es un participante malicioso que puede actuar como observador, capturando y almacenando los mensajes en tránsito, pero además puede realizar otras tareas: eliminar, reenviar o modificar mensajes en tránsito transmitidos por cualquier participante.

Para modelar los eventos de parada del intercambio realizados por cualquier participante malicioso (esto es, A_m , B_m o *I*), el modelo define un mecanismo para informar sobre estos eventos a la otra parte involucrada. Cuando ocurre un evento de este tipo, inmediatamente se envía un mensaje desde el participante que realiza la eliminación del mensaje o la parada del intercambio a las otras partes implicadas. Con este procedimiento, se evita tener que usar nociones de tiempo (*timeout*) en el modelo. Cuando un mensaje de evento es recibido, el participante puede contactar con la TTP o bien parar el intercambio.

Otra consideración importante atañe al tipo de canal de comunicaciones usado para el intercambio de mensajes entre cualquier participante y la TTP, el cual por definición es elástico (ver 2.1).

Con los datos proporcionados, creamos un escenario para ser usado en el modelado de un protocolo utilizando diferentes sesiones de ataque, donde cada una de ellas puede involucrar un iniciador (*A* o A_m) y un receptor (*B* y B_m). Notar que *I* y la TTP están presentes implícitamente en cada sesión. Por tanto, se pueden generar cuatro sesiones distintas combinando los roles de los participantes: (*A*, *B*), (A_m , *B*), (*A*, B_m) y (A_m , B_m) donde (*X*, *Y*) se corresponde con el participante que es el iniciador (*X*) y el receptor (*Y*), respectivamente. En este artículo no se consideran sesiones en paralelo, donde las partes maliciosas pueden participar en múltiples sesiones concurrentes, dejando esta tarea para próximos trabajos.

La arquitectura del modelo está claramente definida en tres grandes bloques o niveles: alto nivel, nivel de entidad y nivel de proceso. Todos los mensajes enviados

El esquema del nivel alto (Figura 1) enseña la interacción entre todos los participantes involucrados en el protocolo, así como el flujo de mensajes entre los mismos. En el nivel alto se pueden consultar los contenidos de la base de datos de cada participante, las cuales contienen los mensajes de protocolo intercambiados. Finalmente, se puede observar y controlar el contenido de la sesión. La variable controla el rol de los participantes y cuáles son utilizados en la ejecución del protocolo (rol honesto o malicioso). Además, en la Figura 1 se puede observar como todos los mensajes son interceptados por I en su tránsito entre los participantes. La notación que aparece en la figura es la notación propia de las Colored Petri Nets, i aparece descrita en la documentación de las mismas.

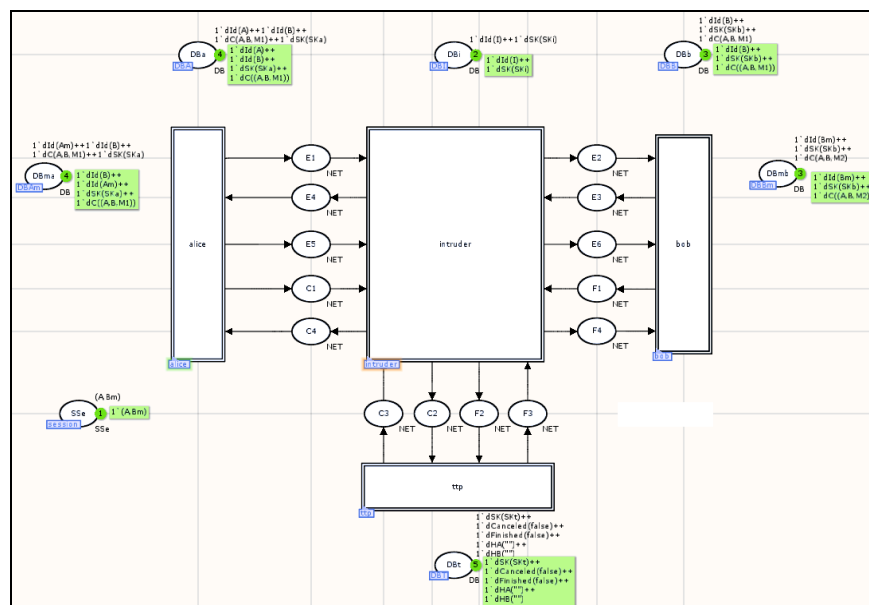


Figura 1. Esquema de Alto nivel.

El nivel de entidad muestra con mayor detalle el flujo de trabajo de los participantes. En la Figura 2(a) se puede ver el nivel de entidad de A y sus dos roles. Las transiciones TA_1 a TA_4 se corresponden a su rol honesto, mientras que las transiciones TAm_1 a TAm_4 pertenecen al rol malicioso. La primera transición de A , TA_1 y TAm_1 , genera el primer mensaje del protocolo y lo envía a B . Las transiciones TA_2 y TAm_2 reciben y verifican el segundo mensaje enviado por B y le retornan el tercer mensaje. TA_3 y TAm_3 tienen la responsabilidad de contactar con la TTP usando el subprotocolo de cancelación, y las ultimas transiciones, TA_4 y TAm_4 reciben la respuesta desde la TTP. Notar que la selección de las transiciones que tienen que ser ejecutadas se lleva a cabo a través de la configuración de la sesión.

El nivel de proceso implementa todas las acciones que ejecutan los participantes y especifica la forma en la que las entidades se relacionan. Las acciones ejecutadas por cada proceso son atómicas, esto es, solo un proceso puede ser ejecutado al mismo tiempo y debe terminar antes que el siguiente empiece su ejecución. Esto puede modelarse por un *token*. Este es único y compartido por todos los participantes. Es capturado por cada parte cuando un proceso empieza su ejecución y es liberado cuando el mismo proceso termina. Además, cada proceso está controlado por un mecanismo de control de flujo, mediante el cual un *token* es transferido a través de los participantes para cada paso del protocolo. Este *token* controla el orden en el que deben ser ejecutadas las acciones. Por ejemplo, el mecanismo controla que la generación de un mensaje se realice después de haberse ejecutado el proceso de verificación del mensaje previamente recibido.

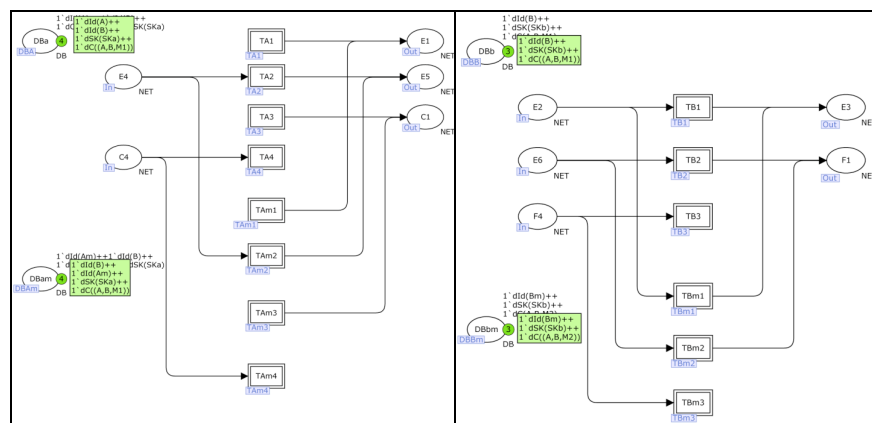


Fig. 2b. Nivel de entidad de B

3.4 Query functions

Para detectar y extraer escenarios de ataque a partir del *state space* generado por la utilidad, utilizamos un serie de funciones de búsqueda (Figura 3) para encontrar contenidos especiales en cada base de datos de los participantes. La función principal es *SearchCommitsTerminalNodes(ack,id)*, donde *ack* es el elemento o compromiso que se quiere buscar en la base de datos del participante identificador por *id*. La función está construida usando una función estándar de la utilidad CPN-Tools, llamada *PredNodes(p1.p2.p3)*. El primer parámetro de esta función es otra función anidada llamada *SearchCommits(ack,id)*, donde *ack* y *id* tienen el mismo uso que en

la función previa. Esta función es capaz de examinar los contenidos de la base de datos de *id* buscando el elemento *ack*. El segundo parámetro es usado para elegir solo los nodos que sean terminales, es decir, nodos hoja del árbol generado, los cuales contienen una ejecución completa del protocolo. El último parámetro, *NoLimit*, indica a la función que debe de examinar todos los nodos y devolver todos los resultados posibles.

La función principal es usada para analizar la equidad del protocolo, aplicándola contra las bases de datos de los participantes del intercambio. La función devuelve una lista de nodos terminales, que contienen el compromiso buscado. El análisis de esta lista permitirá determinar si el intercambio es o no equitativo.

```

fun SearchCommits(ack:DB, id:Id) : Node list
= PredAllNodes(
  fn n =>
    let
      val dba = Mark.Top'DBa 1 n
      val dbb = Mark.Top'DBb 1 n
      val dbi = Mark.Top'DBi 1 n
      val dbam = Mark.Top'DBma 1 n
      val dbbm = Mark.Top'DBmb 1 n
    in
      if (id=A) then
        cf(ack, dba) > 0
      else if (id=B) then
        cf(ack, dbb) > 0
      else if (id=Am) then
        cf(ack, dbam) > 0
      else if (id=Bm) then
        cf(ack, dbbm) > 0
      else (* id=I *)
        cf(ack, dbi) > 0
    end
  )
)

```

```

fun SearchCommitsTerminalNodes(ack:DB, id:Id) : Node list
= PredNodes ( (SearchCommits(ack,id)) ,
  fn n => (Terminal n) andalso (FullyProcessed n),
  NoLimit)

```

Figura 3. Funciones de búsqueda

4 Análisis Automático de la Equidad del Protocolo de Firma de Contratos FPH.

A continuación se describen algunas situaciones conflictivas del protocolo FPH en las que los firmantes disponen de evidencias contradictorias. Las evidencias generadas por este protocolo no son transferibles, y un árbitro debe contactar con ambos firmantes para resolver una disputa, conocer el estado final del intercambio y garantizar el no repudio. Esta propiedad ha sido descrita en [8] y recibe el nombre de *weak fairness*. En este análisis formal de la equidad del protocolo probaremos que el árbitro puede resolver todo tipo de situaciones conflictivas derivadas de la ejecución del protocolo.

En [4] se describió, junto al protocolo FPH, una situación conflictiva en la que A podía obtener evidencia NR de B (h_B) y un mensaje *cancel* de T , mientras B obtenía la evidencia NR de A (h_A , ACK_A). A puede conseguirlo, por ejemplo, invocando el subprotocolo *cancel* después de la finalización del subprotocolo de *intercambio*. Parece que A puede afirmar que el contrato está firmado (*signed*) o no firmado (*cancelled*), dependiendo de su utilidad, mientras que B posee la evidencia NR que probará que el contrato está firmado (*signed*). Hemos detectado esta situación en el análisis automático y la hemos llamado *caso 1*.

Además, gracias al modelo utilizado, hemos descubierto dos situaciones conflictivas más. La primera (a la que llamaremos *caso 2*) se produce cuando un A malicioso invoca el subprotocolo *cancel* después de la finalización del subprotocolo de *intercambio* (como en el *caso 1*) y después un B malicioso lo ejecuta también. Parece que tanto A como B pueden afirmar que el contrato está firmado (*signed*) o no firmado (*cancelled*), dependiendo de su voluntad.

La última situación conflictiva que hemos detectado (*caso 3*) se alcanza cuando el intercambio se interrumpe después del segundo paso. En este caso A dispone de la evidencia de NR de B mientras B no dispone de la evidencia NR de A . Ambas partes pueden contactar con la TTP . Si B contacta en primer lugar, la TTP le enviará la evidencia NR y el contrato estará firmado. Por otra parte, si A contacta en primer lugar, la TTP cancelará el intercambio y entonces A tendría la evidencia NR de B (h_B) y un mensaje de cancelación (*cancel*) de T , mientras B obtendría el mensaje de cancelación (*cancel*).

Las tres situaciones conflictivas se pueden detectar utilizando el modelo que recrea una sesión formada por un iniciador malicioso o bien dos usuarios maliciosos actuando como iniciador y receptor (A_m , B_m). De esta forma, todos los posibles comportamientos fraudulentos de ambas partes pueden ser contemplados. Usando las funciones (*query functions*) definidas sobre el modelo, como puede observarse en la figura 4, se ha realizado una búsqueda en las bases de datos de cada usuario para localizar las evidencias almacenadas. En este caso, se han buscado los mensajes segundo y tercero del subprotocolo de intercambio así como todas las respuestas recibidas de la TTP .

SearchCommitsTerminalNodes(dSOS(((A,B,M1),SKb), SKa), Am) val it = [211,467,488,506,607,610,613,614,615,616] : Node list	SearchCommitsTerminalNodes(dSOS(((A,B,M1),SKb), SKa), Bm) val it = [211,488,506,615,616] : Node list
SearchCommitsTerminalNodes(dSOS(((A,B,M1),SKb), SKt), Am) val it = [535,538,610,614,616] : Node list	SearchCommitsTerminalNodes(dSOS(((A,B,M1),SKb), SKt), Bm) val it = [467,506,535,538,610,614,616] : Node list
SearchCommitsTerminalNodes(dSMT([cancel,((A,B,M1),SKa)),SKt),Am) val it = [488,536,537,607,613,615,84] : Node list	SearchCommitsTerminalNodes(dSMT([cancel,((A,B,M1),SKb)),SKt),Bm) val it = [536,537,607,613,615] : Node list

Figura 4. Resultados de la aplicación de las funciones de búsqueda en la sesión (A_m , B_m).

Si analizamos la lista de nodos obtenidos en cada ejecución de la función, podemos construir la tabla siguiente con las tres situaciones previamente descritas. Por cada caso detectado, señalamos el estado del contrato (firmado o cancelado) así como los contactos de las partes, honestamente o de forma maliciosa, (A_m o B_m), con la TTP .

Caso	Nodo	A_m tiene NR	B_m tiene NR	A_m contacta con la TTP	B_m contacta con la TTP
1	488	Firmado Cancelado	Firmado	Si (malicioso)	No
2	615	Firmado Cancelado	Firmado Cancelado	Si (malicioso)	Si (malicioso)
3	607	Firmado Cancelado	Cancelado	Si (malicioso)	Si (honesto)
3	613	Firmado Cancelado	Cancelado	Si (malicioso)	Si (honesto)

Tabla 1. Lista de nodos correspondientes a las tres situaciones con evidencias contradictorias.

Como se puede observar en la tabla 2, la utilización del modelo basado en Petri Nets permite localizar las situaciones relativas a los tres casos conflictivos, en los que A_m y B_m disponen de evidencias contradictorias. Se han localizado cuatro posibles escenarios dado que el *caso 3* puede aparecer en dos ocasiones.

Podemos observar que el *caso 1* se produce en el nodo 488, cuando A_m obtiene la evidencia NR de B_m (entonces A dispone de la evidencia que demuestra que el contrato esta firmado (*signed*). Sin embargo A contacta con la TTP con el objetivo de cancelar el intercambio. Este es un comportamiento malicioso ya que A_m no debería contactar con la TTP para cancelar un intercambio que ya ha sido finalizado. La TTP envía a A_m el mensaje *cancel* y a partir de este momento A_m puede afirmar que el contrato está firmado (*signed*) o no firmado (*cancelled*). B_m recibe la evidencia NR de A_m y no necesita contactar con la TTP.

En el *caso 2*, correspondiente al nodo 615, el subprotocolo de intercambio finaliza satisfactoriamente para ambas partes, pero A_m contacta con la TTP, después de transferir la evidencia NR a B_m , para obtener el mensaje *cancel*. En el momento en el que B_m recibe la evidencia NR de A_m también contactará con la TTP y obtendrá un mensaje *cancel*. Por tanto, A_m y B_m se comportan incorrectamente ya que han contactado con la TTP en situaciones en las que no deberían hacerlo.

El *caso 3* se detecta en dos ocasiones durante la simulación del protocolo. En ambos escenarios A_m obtiene la evidencia NR de B_m pero B_m nunca recibe el tercer mensaje del subprotocolo de intercambio. En cada escenario, A_m ejecuta maliciosamente el subprotocolo de cancelación y como resultado recibe un mensaje *cancel* de la TTP. En el primer escenario, correspondiente al nodo 607, A_m decide, maliciosamente, detener el intercambio y por ello no envía el tercer mensaje a B_m . Por otra parte, el nodo 613 es el resultado de un evento de tipo *drop* sobre el tercer mensaje realizado por un usuario intruso, I . Desde el punto de vista de B_m , ambos escenarios son el mismo caso y por tanto contactará con la TTP para resolver la situación obteniendo un mensaje *cancel* para cada escenario.

Además de los casos conflictivos descritos anteriormente, existen otras situaciones, también detectadas por el modelo, en las que no se generan evidencias contradictorias, aunque las partes pueden disponer de pruebas repetidas al haber contactado con la TTP una vez finalizado el intercambio. La siguiente tabla muestra los escenarios en los que las pruebas generadas son complementarias pero no contradictorias:

Nodo	A_m tiene NR	B_m tiene NR	A_m contacta TTP	B_m contacta TTP
84	Cancelado	Ninguna	Si (honesto)	No
211	Firmado	Firmado	No	No
467	Firmado	Firmado	No	Si
506	Firmado	Firmado Firmado (TTP)	No	Si
535	Firmado (TTP)	Firmado (TTP)	Si (honesto)	Si (honesto)
536	Cancelado	Cancelado	Si (honesto)	Si (honesto)
537	Cancelado	Cancelado	Si (honesto)	Si (honesto)
538	Firmado (TTP)	Firmado (TTP)	Si (honesto)	Si (honesto)
610	Firmado Firmado (TTP)	Firmado (TTP)	Si (malicioso)	Si (honesto)
614	Firmado Firmado (TTP)	Firmado (TTP)	Si (malicioso)	Si (honesta)
616	Firmado Firmado (TTP)	Firmado Firmado (TTP)	Si (malicioso)	Si (malicioso)

Tabla 2. Escenarios especiales pero sin evidencias contradictorias

Los casos más interesantes de entre los listados en la tabla 2 son los nodos 84 y 616. En el nodo 84, B_m no tiene ningún elemento de A_m ya que un intruso I ha ejecutado un evento de *drop* sobre el primer mensaje. B_m no puede ejecutar el subprotocolo de finalización porque no dispone de ningún elemento válido recibido de A_m . Por otra parte, A_m resuelve la situación anormal ejecutando el subprotocolo de cancelación que le permitirá obtener de la TTP un mensaje de cancelación. El segundo nodo, 616, es el caso en el que A_m y B_m actúan maliciosamente contactando con la TTP cuando el subprotocolo de intercambio finaliza satisfactoriamente. Se trata de un caso similar al del nodo 615, pero esta vez B_m contacta en primer lugar con la TTP, obteniendo la correspondiente evidencia de NR. Entonces, si A_m intenta cancelar, la TTP envía una evidencia de NR que probará que el contrato está firmado.

Para resolver las situaciones conflictivas descritas anteriormente, un árbitro debe siempre contactar con ambas partes antes de tomar una decisión, y en caso de encontrar evidencias contradictorias, hemos determinado que debe seguir el siguiente procedimiento de actuación:

- *Caso 1:* A puede afirmar que el contrato está firmado o no (*signed* o *cancelled*), pero B posee evidencia de NR que probará que el contrato está firmado. Si A intenta usar el mensaje que prueba que el intercambio ha sido cancelado estará demostrando que es un impostor. Por esta razón el árbitro deberá dar la razón a B .
- *Caso 2:* Como en el caso anterior, un árbitro deberá contactar con ambas partes en caso de evidencias contradictorias. Si B puede aportar las evidencias de NR que prueben que el contrato está firmado, el árbitro puede afirmar que el contrato está firmado, y si B demuestra que el contrato se canceló, el árbitro dará la razón a B y afirmará que el intercambio fue cancelado y el contrato no ha sido firmado. De éste modo, debido al hecho que A es siempre la primera parte en hacer trampas, si el árbitro da la razón siempre a B , el protocolo desalentará a A a actuar fraudulentamente.

- *Caso 3:* En este caso, de nuevo *A* ha actuado fraudulentamente, y si el árbitro da la razón a *B*, afirmará que el contrato no está firmado ya que el intercambio fue cancelado.

Como conclusión, hemos detectado la situación conflictiva definida en [4] y hemos descubierto dos casos adicionales. Todos los casos son debidos al comportamiento fraudulento de *A*. Para resolver estas situaciones, un árbitro debe contactar siempre con ambas partes y en caso de conflicto siempre dará la razón a *B*. De este modo, el protocolo será equitativo en todos los casos y además las partes no se verán motivadas a comportarse fraudulentamente sabiendo que ello puede ser perjudicial para sus intereses.

5. Conclusiones y Trabajo Futuro

En este artículo se describe como hemos analizado formalmente, usando un mecanismo automático basado en Petri Nets, un protocolo de firma de contratos eficiente, el protocolo FPH [4], una de las soluciones existentes en la que las partes intercambian únicamente tres mensajes.

Para la evaluación del protocolo FPH se ha utilizado un modelo que asume que todos los firmantes pueden ser deshonestos e incluso que un intruso puede atacar también el intercambio. De esta forma el modelo puede ser utilizado para probar la resistencia del protocolo frente a ataques habituales en este tipo de protocolos.

Para la evaluación formal de la equidad del protocolo se han evaluado todas las posibles situaciones derivadas de la intervención de usuarios maliciosos. Se ha demostrado que en todos los casos el intercambio finaliza siempre proporcionando un trato justo a todos los firmantes.

En la realización de la demostración se ha detectado que existen tres casos en los que, aunque el intercambio sea equitativo, uno de los firmantes (o ambos) pueden poseer evidencias contradictorias (que permitan demostrar que el contrato está firmado o que no está firmado). Por esta razón, aunque el intercambio sea equitativo, no puede decirse que las pruebas generadas sean transferibles. Ambas partes deben ser interrogadas por un árbitro para que se conozca el estado final del intercambio.

Finalmente, hemos generado las reglas de comportamiento del árbitro para que se mantenga la equidad incluso cuando detecte la existencia de evidencias contradictorias.

La creación de un modelo de análisis automático ha permitido verificar una propiedad del protocolo. De modo similar el modelo puede utilizarse para detectar vulnerabilidades frente a ataques. Utilizando el modelo, en un futuro podrán analizarse otras propiedades del protocolo, como la verificabilidad de la tercera parte de confianza, y se podrán modelar protocolos más complejos como la versión multiparte del protocolo de firma de contratos. De forma paralela se seguirá trabajando en la mejora del modelo para permitir nuevos escenarios de ataque, como los ataques confabulados utilizando datos de sesiones diferentes, y para permitir más opciones de control sobre el comportamiento del usuario intruso.

Referencias

1. N. Asokan, M. Schunter and M. Waidner: "Optimistic protocols for fair exchange"; *4th ACM Conference on Computer and Communications Security*, pp. 7-17, 1997.
2. N. Asokan, V. Shoup and M. Waidner: "Asynchronous Protocols for Optimistic Fair Exchange"; *IEEE Symposium on Research in Security and Privacy*, pp. 86-99, 1998.
3. Bao, F., Wang, G., Zhou, J., and Zhu, Z.: "Analysis and Improvement of Micali's Fair Contract Signing Protocol", *Proceedings of The 9th Australasian Conference on Information Security and Privacy*, 2004: 176-187.
4. Ferrer-Gomila, J.L., Payeras-Capellà, M., Huguet-Rotger, L.: "Efficient Optimistic N-Party contract Signing Protocol", *Information Security Conference. 4th International Conference, ISC'01*, Lecture Notes in Computer Science 2200, pàgines 394-407, Springer Verlag, 2001.
5. Ferrer-Gomila, J.L., Payeras-Capellà, M., Huguet-Rotger, L.: "Optimality in Asynchronous Contract Signing Protocols", *Trust and Privacy in digital Bussiness, TrustBus'04*, Lecture Notes in Computer Science 3184, pàgines 200-208, Springer Verlag, 2004.
6. Garay, J.A.; Jakobsson, M., and MacKenzie, P.: "Abuse-Free Optimistic Contract Signing"; *CRYPTO'99*, LNCS 1666, Springer Verlag, pp. 449-466, 1999.
7. Jensen, K., Kristensen, L.M., and Wells, L., "Coloured Petri Nets and CPN Tools for Modeling and Validation of Concurrent Systems", *International Journal on Software Tools for Technology Transfer*, 9(3): 213-254, 2007.
8. Kremer, S., Markowitch, O., Zhou, J.: "An Intensive Survey of Fair Non-repudiation Protocols", *Computer Communications*, Vol. 25, 1606-1621, 2002
9. Micali, S.: "Simple and Fast Optimistic Protocols for Fair Electronic Exchange", *Proceedings of 21st Symposium on Principles of Distributed Computing*, 2003: 12-19.
10. Sornkhom, P. and Permpoontanalarp, Y.: Security analysis of Micali's fair contract signing protocol by using coloured Petri nets", *9th ACIS Int. Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing*, 329-334, 2008.
11. J. Zhou, R. Deng and F. Bao: "Some Remarks on a Fair Exchange Protocol"; *PKC 2000*, LNCS 1751, Springer Verlag, pp. 46-57, 2000.

Criptoanálisis del Generador Auto-Shrinking: Una Propuesta Práctica*

María Eugenia Pazo-Robles¹, Amparo Fúster-Sabater²

¹ ITBA, Instituto Tecnológico de Buenos Aires, Av E. Madero 399, Buenos Aires, Argentina
eugepazorobles@gmail.com

² Instituto de Física Aplicada, CSIC, Serrano 144, 28006 Madrid, España
amparo@iec.csic.es

Resumen. En este trabajo se presenta un ataque criptoanalítico sobre el generador auto-shrinking, un generador de secuencia cifrante bien conocido por su fácil implementación y sus buenas propiedades criptográficas. Se propone una variación práctica de la técnica criptoanalítica conocida como Guess-and-Determine, con supuestos claramente definidos y elaborados a lo largo del propio proceso. Se presentan resultados numéricos que mejoran el criptoanálisis planteado a este generador mediante la aplicación de la técnica conocida como Time Memory Trade Off (TMTO). Concretamente, se han logrado complejidades del orden de $O(2^{0.33L})$ en cuanto a cantidad de secuencia interceptada (siendo L la longitud del registro del generador), $O(L)$ en referencia a memoria consumida y una complejidad pre-computacional sumamente baja. Los programas de implementación del criptoanálisis y de resolución del sistema de ecuaciones han sido realizados en Matlab.

Palabras Clave: Generador auto-shrinking, criptoanálisis, cifrado en flujo, ecuaciones lineales.

1 Introducción

La criptografía de clave secreta se divide en dos grandes apartados: procedimientos de cifrado en flujo y procedimientos de cifrado en bloque. Los sistemas de cifrado en flujo son los más rápidos dentro de los métodos de cifrado, de ahí que en la actualidad se utilicen en numerosas aplicaciones prácticas. Prueba de ello son, por ejemplo, los algoritmos A5 (en su doble versión A5/1 y A5/2) que se utilizan en telefonía móvil GSM [1], el algoritmo E0 usado en especificaciones de Bluetooth [2] ó el algoritmo RC4 utilizado en Microsoft Word y Excel [3].

Un sistema de cifrado en flujo está compuesto por un algoritmo o generador de secuencia cifrante (conocido públicamente) y una clave de cifrado (conocida

* Trabajo financiado por CDTI (Centro para el Desarrollo Tecnológico Industrial, España) y las empresas INDRA, Unión Fenosa, Tecnobit, Visual Tools, Brainstorm, SAC y Technosafe en el marco del Proyecto Cenit-HESPERIA. También ha sido financiado por el Ministerio de Ciencia e Innovación (España) y el fondo europeo FEDER en el marco del Proyecto TIN2008-02236/TSI.

únicamente por los dos comunicantes). La clave corresponde normalmente al estado inicial del generador de secuencia. En el momento de su inicialización el generador toma la clave como semilla, posteriormente se ejecuta el algoritmo mediante un clock a la velocidad que necesite la aplicación y así se genera la secuencia cifrante.

Para cifrar, el emisor realiza una operación XOR, bit a bit, entre la secuencia cifrante y el texto claro, dando origen al texto cifrado que es el que se va a enviar por el canal de información. Para descifrar, el receptor genera la misma secuencia cifrante que suma bit a bit con el texto cifrado recibido y recupera así el texto claro original.

Muchos algoritmos de cifrado en flujo están basados en Registros de Desplazamiento con Realimentación Lineal [4], en inglés Linear Feedback Shift Registers (LFSRs), cuyas secuencias de salida, las *PN*-secuencias, se combinan entre sí mediante algún procedimiento o función no lineal. Entre los generadores de secuencias pseudoaleatorias más conocidos y mejor estudiados con aplicaciones criptográficas podemos señalar: generadores combinatoriales, filtros no lineales, generadores controlados por uno o varios relojes, generadores multi-velocidad, generadores con decimación irregular, etc. Todas estas estructuras producen a su salida secuencias cifrantes con una alta complejidad lineal, períodos muy largos y buenas propiedades estadísticas.

Los procedimientos de cifrado de flujo se utilizan para dar seguridad criptográfica a sistemas de comunicaciones con requerimientos de velocidad y sincronismo. Uno de estos procedimientos de cifrado de amplia difusión es el generador auto-shrinking. La Unión Europea, a través del Proyecto Stork [5], propuso a la comunidad científica internacional romper este generador mediante alguna técnica criptoanalítica que mejorase el ataque TMTO.

Siguiendo esta premisa, se ha emprendido el criptoanálisis de este generador mediante variaciones a la técnica del Guess and Determine [6]. Se ha llevado a cabo una personalización de dicha técnica, mediante lo que se consideran suposiciones bien definidas para romper este generador de secuencia cifrante.

Este trabajo está dividido en secciones y organizado de la siguiente manera. En la sección 2, se describe el generador auto-shrinking con sus correspondientes características y propiedades de pseudoaleatoriedad. La sección 3 está dedicada tanto a la técnica de criptoanálisis propuesta como a los correspondientes resultados de implementación. En la sección 4, el ataque desarrollado se compara con otras técnicas criptoanalíticas en uso, mientras que en la sección 5 se presentan las conclusiones y futuros trabajos a realizar.

2 Generador Auto-Shrinking

El generador auto-shrinking fue diseñado por Meier y Staffelbach [7] para su uso en aplicaciones criptográficas. Este generador es de fácil implementación y consiste en un único LFSR de L etapas y polinomio de realimentación primitivo de grado L [4]. El esquema general de un LFSR aparece representado en la Fig. 1.

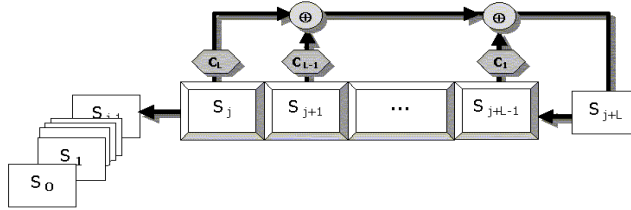


Fig.1 Registro de Desplazamiento con Realimentación Lineal

Este registro, que se nota, R_L , genera una única secuencia pseudoaleatoria, $\{s_n\}$, la cual se decima de forma irregular dando origen a la secuencia *auto-shrunk*, notada $\{z_n\}$, o secuencia de salida del generador. La regla de decimación es extremadamente simple,

se consideran pares (s_{2i}, s_{2i+1}) ($i = 0, 1, 2, \dots$) de bits consecutivos de $\{s_n\}$ tales que:

- Si $s_{2i} = 1$, entonces $z_i = s_{2i+1}$.
- Si $s_{2i} = 0$, entonces s_{2i+1} se descarta.

Es decir, si el primer bit del par considerado es un 1, entonces el segundo bit se inserta en la secuencia de salida. Por el contrario, si el primer bit del par considerado es un 0, entonces el segundo bit se rechaza. De esta manera, se van eliminando determinados bits de la secuencia $\{s_n\}$ y los que quedan constituyen la secuencia $\{z_n\}$ o secuencia auto-shrunk. La clave de este generador es el estado inicial del LFSR, aunque los autores recomiendan que el polinomio de realimentación forme también parte de la misma. De acuerdo con [7], el periodo, la complejidad lineal y las propiedades estadísticas de la secuencia $\{z_n\}$ son muy adecuados para su aplicación en criptografía. Asimismo y según [7], el polinomio característico $P(x)$ de la secuencia auto-shrunk $\{z_n\}$ viene dado por:

$$P(x) = (x + 1)^p, \quad (1)$$

donde el valor de p está acotado por:

$$2^{L-2} < p < 2^{L-1}. \quad (2)$$

Esto implica una relación de recurrencia lineal de la forma:

$$(E + 1)^p z_n = 0, \quad (3)$$

donde E es el operador desplazamiento que actúa sobre los términos de la secuencia $\{z_n\}$, es decir:

$$E z_n = z_{n+1}. \quad (4)$$

La ecuación (3) representa una ecuación en diferencias lineal con coeficientes binarios y constantes cuyo polinomio característico $P(x)$ dado por la ecuación (1) tiene una única raíz $\lambda = 1$ con multiplicidad p .

El periodo de esta secuencia auto-shrunk $\{z_n\}$ se denota por T y viene dado por:

$$T = 2^{L-1}. \quad (5)$$

La secuencia $\{s_n\}$, generada por el LFSR, puede considerarse formada por dos secuencias diferentes $\{c_n\}$ y $\{b_n\}$ que responden a los bits de subíndices pares e impares, respectivamente, de la secuencia $\{s_n\}$. Es decir:

$$c_i = s_{2i} \quad \forall i \geq 0 \quad (6)$$

$$b_i = s_{2i+1} \quad \forall i \geq 0 \quad (7)$$

A su vez, estas dos secuencias corresponden a la misma *PN*-secuencia $\{s_n\}$ generada por el LFSR pero desplazadas entre sí una distancia de 2^{L-1} bits. La existencia de dicho desplazamiento permite expresar una secuencia en función de la otra [6]. Así vemos que:

$$b_i = \sum_{j=0}^{L-1} h(x) c_j \quad (0 \leq i \leq L-1) \quad (8)$$

Donde el polinomio $h(x)$ [6] sirve para expresar unos coeficientes en función de los otros. A partir de esta relación, se puede armar una matriz de coeficientes H , que es la que usaremos para resolver el sistema de ecuaciones.

Esta matriz de coeficientes se construye para las sucesivos valores de b_i expresados en función de los coeficientes c_i específicos.

$$H = \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ \vdots \\ \vdots \\ \vdots \\ b_{L-1} \end{pmatrix} \quad (9)$$

3. Ataque Propuesto

El ataque consiste en generar o crear una serie de suposiciones para los $L/2+1$ bits iniciales de c_i y, a partir de ellos, determinar los restantes bits de c_i a la vez que los L bits de b_i . Juntos forman la condición inicial de la secuencia $\{s_n\}$. Romper el generador es equivalente a encontrar la condición inicial de la secuencia $\{s_n\}$ o, en su defecto, un estado inicial que continúe la secuencia lógica de bits $\{s_n\}$.

Para ello, se arman bloques de B bits de secuencia interceptada (suposiciones) y, para cada bloque, se comprueba si se rompe o no el criptosistema. En efecto, nos vamos desplazando dentro de estos B bits haciendo las comprobaciones pertinentes hasta encontrar la solución correcta.

Dado que es posible expresar los coeficientes de acuerdo con las ecuaciones (8) y (9), tenemos la posibilidad de plantear un sistema de ecuaciones lineales. Entonces y a

partir de esta matriz H y del polinomio característico del LFSR, se plantea el sistema de ecuaciones lineales con coeficientes binarios que habrá que resolver. No todas las suposiciones son aptas para la resolución del sistema, por lo que hay que seleccionarlás adecuadamente. Esta selección de suposiciones es lo que hace de nuestro criptoanálisis, un ataque efectivo y más eficaz que el que utiliza la técnica de Guess and Determine.

En la sección 3.3 se explica el proceso de selección de dichas suposiciones. En la sección 3.4 se describen los programas desarrollados para este criptoanálisis y la secuencia lógica que sigue el ataque y el diagrama de flujo detallado en la Fig. 2.

Definimos a continuación los parámetros que caracterizan el ataque criptoanalítico:

- T_s : Tiempo que se tarda en obtener todas las suposiciones posibles o las que se usaran en el ataque. Este parámetro no influye en el tiempo de procesamiento del criptoanálisis y depende del generador auto-shrinking a criptoanalizar (longitud y polinomio característico del LFSR).
- *Cantidad de Suposiciones*: Número de suposiciones. Para cada suposición, se conocen $L/2+1$ bits de secuencia $\{c_n\}$, de un total de L bits. El ataque consiste en probar cada una de estas suposiciones hasta romper el sistema. Es decir, hasta calcular el resto de los $L-L/2+1$ bits de $\{c_n\}$ y los L bits de $\{b_n\}$ con el fin de poder seguir generando bits de la secuencia $\{z_n\}$.
- N : Número de bloques en los que dividimos el período de secuencia de salida $\{z_n\}$ o secuencia auto-shrunk.
- B : Longitud del bloque en el que dividimos la secuencia $\{z_n\}$, es decir la cantidad de bits de secuencia interceptada que necesitamos para romper el generador.
- *Porcentaje de aciertos*: Número total de aciertos sobre el total de bloques estudiados de longitud B bits por bloque.
- *ECU*: Cantidad de bits supuestamente conocidos de c_i . Este valor es igual a $L/2+1$ para todo polinomio de L etapas a considerar.

La idea básica consiste en analizar, para valores pequeños de L , el número de aciertos sobre todo un período de la secuencia de salida, extrapolando luego estos resultados a generadores auto-shrinking en un rango de aplicación criptográfica. Este análisis intensivo puede llevarse a efecto hasta valores de $L \approx 20$, con longitudes mayores el período de la secuencia generada se hace relativamente grande.

Las pruebas hasta el momento se han realizado en Matlab, simulando diferentes generadores auto-shrinking con registros de desplazamiento de longitud L , donde L varía en el intervalo 10 - 20 etapas. A su vez, para cada una de estas longitudes se han simulado LFSRs con diferentes polinomios de realimentación.

3.1. Método propuesto

Para realizar este ataque habrá que plantear, a partir de una suposición y una cantidad de secuencia interceptada, un sistema booleano de ecuaciones lineales y, posteriormente, resolverlo. La resolución se lleva a cabo aplicando sencillamente el método de Gauss. Claramente, el sistema planteado no siempre va a tener solución. A

su vez hay que hacer notar que, en cada suposición considerada, sólo los bits c_i que toman el valor 1 aportan una ecuación al sistema. Por el contrario, los bits c_i con valor 0 no añaden nuevas ecuaciones.

A partir del programa desarrollado también es posible expresar los coeficientes c_i en función de los b_i y viceversa. Esta doble vía nos permite calcular una mayor cantidad de incógnitas y no sólo obtenerlas a partir de la resolución del sistema.

3.2. Tiempo, Memoria y Serie Cifrante

El tiempo de procesamiento máximo del ataque criptoanalítico es el que tarda el programa en procesar todas las suposiciones sobre todos los bits de cada bloque B . Pasamos ahora a exponer algunos resultados numéricos.

Para generadores con $L = 20$, bloques de secuencia interceptada de valor $B = 120$ bits y *Cantidad de Suposiciones* = 34, se ha alcanzado un éxito de rotura del criptosistema del 82%. Por tanto, la cantidad de bits necesarios para lograr este porcentaje de éxitos será del orden de $O(2^{0.33*L})$. Dicho orden podría disminuir si se agregara alguna suposición adicional. A partir de los ejemplos programados puede también apreciarse que, con una cantidad de secuencia interceptada entre 100 - 200 bits, se obtiene un éxito de rotura del 93 %. Luego, para $L = 20$, la cantidad promedio de bits necesarios para romper el generador estaría en torno a los 150 bits.

Es importante destacar también que, para bloques de $B = 120$ bits, aunque haya un 18% de casos que no puedan resolverse sólo el 2% de los mismos son consecutivos. Luego, si bien hay ya garantizado un 82% de éxitos, tendremos un 98% de éxitos al analizar el bloque siguiente.

Cabe la misma disquisición para bloques de $B = 100$ bits, donde sólo el 6% de los bloques no resueltos son consecutivos, con lo cual si bien existe ya un 75% de éxitos, tendremos un 94% de éxitos de rotura con el bloque siguiente.

El tiempo de procesamiento hay que computarlo considerando el número de suposiciones que se utilizan en el ataque y la cantidad de bits que componen el bloque de secuencia interceptada. Se tendría así el número total de intentos para romper el criptosistema.

Hay que tener presente que solo son necesarios L bits de secuencia interceptada para plantear el sistema de ecuaciones y romper el generador. El problema es que no sería realista considerar que el ataque comienza precisamente en el bit de secuencia interceptada a partir del cual el sistema es capaz de romper el generador. Por ello, hay que tomar B bits de secuencia interceptada y desplazarse dentro del bloque hasta encontrar la solución correcta o estado inicial del LFSR.

Los tiempos de procesamiento deben ser aún evaluados convenientemente. De todas maneras, la forma en que están desarrollados los programas permite un procesamiento en paralelo, lo cual también disminuirá drásticamente los tiempos de procesamiento.

El ataque en paralelo supondría, entre otras cosas, tener una CPU por cada suposición. De este modo el tiempo de procesamiento disminuiría en un orden que sería directamente proporcional al número de suposiciones. Otro ejemplo de paralelismo estaría dado si se consideraran tantas CPU's como bits hay en el bloque de secuencia interceptada. En este caso, el tiempo de procesamiento estaría dado por el tiempo que

tarda el programa en procesar los L bits de secuencia interceptada que necesita para resolver el sistema de ecuaciones.

La cantidad de memoria en uso es muy baja, y responde a la cantidad de secuencia interceptada necesaria para plantear el sistema de ecuaciones y a la tabla de suposiciones que tenemos que tener pre-computada.

Vemos en las siguientes Tablas algunos resultados concretos. En la Tabla 1, se representa el porcentaje de aciertos, o número de veces en que se rompió el generador, para distintas longitudes L del LFSR, R_1 . También en esta tabla aparece la cantidad de suposiciones consideradas para distintos valores de L .

Así por ejemplo, para un auto-shrinking con $L = 20$ etapas, 16 suposiciones y 300 bits de secuencia interceptada, se obtiene un 84 % de aciertos considerando un número de bloques igual a 200. Es decir se ha corroborado que suponiendo ECU valores de c_i , en este caso 11, se obtendrían los 9 coeficientes c_i restantes y los 20 valores de b_i .

Tabla 1. Diferentes parámetros para generadores auto-shrinking con diferentes longitudes L

L	Cantidad Suposiciones	Bloques: N	Longitud del Bloque: B	Período: T	Porcentaje de aciertos	de ECU: $L/2+1$
20	13-16	200	300	2^{19}	84%	11
16	8	405	80	2^{15}	83%	9
14	8	171	48	2^{13}	87%	8
10	4	332	16	2^9	86%	6

Para seguir con las comparaciones, presentamos la Tabla 2. Se puede distinguir en la misma, como el éxito en la rotura del generador depende de la cantidad de secuencia interceptada o tamaño del bloque.

Así puede observarse que siguiendo con polinomios de grado 20 y aumentando el número de suposiciones, se aumenta la capacidad de éxito del ataque. De todos modos, hay que señalar que el éxito del ataque depende prioritariamente de la cantidad de bits que componen el bloque. Sin embargo, hay que tener presente que en la práctica este número va a ser pequeño. Son conocidas las dificultades para obtener secuencia interceptada para el criptoanálisis, y además que cuanto mayor sea este número mayor será el tiempo de procesamiento, pues el ataque se basa en ir desplazándose dentro del bloque de secuencia interceptada.

Tabla 2. Diferentes parámetros para generadores auto-shrinking con la misma longitud L

L	Cantidad Suposiciones	Bloques: N	Longitud del Bloque: B	Período: T	Porcentaje de aciertos	de ECU: $L/2+1$
20	34	301	100	2^{19}	75%	11
20	34	301	120	2^{19}	82%	11
20	13-16	200	300	2^{19}	84%	11

Como se aprecia en la Tabla 2, a medida que disminuye el número de bits en el bloque, disminuye asimismo el porcentaje de éxito en la rotura. Sin embargo, realizando un

estudio más exhaustivo, sí se observa que si no se logra romper el generador con los B bits definidos, la probabilidad de encontrar la solución para este polinomio de grado 20 en los siguientes B bits aumenta considerablemente. Este porcentaje de éxitos se refleja en la Tabla 3.

Tabla 3. Comparación para Segundos intentos entre polinomios de igual grado L , pero distinta longitud de Bloque

L	Cantidad Suposiciones	Bloques: N	Longitud Bloque: B	del	Período T	Porcentaje de aciertos	ECU: $L/2+1$
20	34	85	100		2^{19}	94%	11
20	34	85	120		2^{19}	98%	11
20	34	85	150		2^{19}	99%	11

3.3. Pre-Procesamiento

No es necesario para nuestro criptoanálisis, realizar un pre-procesamiento con tiempos elevados. Los datos a pre-procesar son las suposiciones que vamos a necesitar tener calculadas antes de correr el programa.

De hecho es relativamente fácil encontrar estos supuestos que luego se utilizan para la determinación de los coeficientes c_i y b_i ($0 \leq i \leq L-1$).

Calcular las suposiciones, en realidad, implica recorrer el universo de posibles soluciones al sistema de ecuaciones. Como se ha dicho anteriormente, existen suposiciones con un número X de 0's en los bits supuestos de c_i , lo que supone ($ECU-X$) ecuaciones válidas para resolver. A su vez, de estas ecuaciones no todas nos permitirán obtener una solución. Por otro lado, el lugar donde se encuentren los 0's dentro de la suposición también será un factor a considerar en la resolución del sistema.

En resumen, hallar las suposiciones correctas implica hallar la mayor cantidad posible de ecuaciones que resuelvan el sistema.

El método práctico utilizado en este criptoanálisis, para hallar las suposiciones, consiste en tomar en principio aquellas que tienen solo 1 cero y corroborar que se resuelve el sistema de ecuaciones. Luego en lugar de seleccionar esas suposiciones, se consideran las que se encuentran a continuación, en general tienen más de un cero, aumentándose la cantidad de ceros, en función del grado L , del polinomio en cuestión.

Debe ser tomada en cuenta la resolución del sistema de ecuaciones. En muchos casos un cero en determinado bit de la suposición, genera que el sistema no pueda ser resuelto. Dichos bits deben ser analizados, para el posterior descarte de la suposición, de ser necesario, en cada generador Auto-Shrinking.

Para el caso de $L=10$, ya considerado, Tabla 1, las 4 suposiciones elegidas y con 2 ceros cada una, son :

$$Sup = [111110;111010;110110;101110]$$

Vemos en la Tabla 4. , la distribución de estas suposiciones a lo largo del período para $L=10$ como ejemplo. Es importante que la selección se distribuya convenientemente a lo largo del bloque de 16 bits, en este caso, de secuencia interceptada. El objetivo es hallar la condición inicial del generador Auto-Shrinking, con solo esos 16 bits, y por ende las suposiciones deberían estar distribuidas para encontrar esa solución. Como ejemplo práctico vemos en la Tabla 4, la distribución de estas soluciones propuestas a lo largo del período.

Tabla 4. Ubicación de las correspondientes suposiciones dentro de la secuencia de Auto-Shrinking. Suposición vs Puntero a la secuencia para un $L=10$ con período $T=512$

<i>Supu estos</i>	<i>Punteros de la secuencia Auto-Shrinking</i>														
111110	14	46	136	152	191	230	239	248	255	317	344	379	430	471	505
111010	110	181	193	211	390	432	482	497							
110110	17	36	120	127	294	300	405	421							
101110	34	60	90	95	101	104	114	175	210	217	243	287	310	366	437

Vemos en la Tabla 4., que con estas suposiciones, las soluciones halladas se distribuyen a lo largo del período, arribando a un éxito de un 86 % tomando solo 16 bits de secuencia para criptoanalizar, de un período $T=2^{L-1}$.

El cálculo de suposiciones, no es en tiempo real de procesamiento, por ende su cálculo no perjudica al sistema para nada. La cantidad de suposiciones que integran el sistema criptoanalítico , es la variable que debe ser maximizada para un criptoanálisis efectivo en tiempo real. En el caso analizado de 2^{Ecu} posibles suposiciones, el sistema solo debe procesar 4, y ahí el tiempo de procesamiento es óptimo.

3.4. Programas

Todos los programas están realizados en Matlab. Consisten en un programa o función principal que requiere de otros programas o funciones secundarias. Dichas funciones son llamadas para realizar tareas específicas tales como:

- Separar en bloques de longitud estipulada los bits que componen la serie cifrante y que se consideran los datos de entrada.
- Introducir como datos todos los supuestos. Éstos son valores del estado inicial de la secuencia $\{c_n\}$. Dicha secuencia se genera a partir del LFSR del generador auto-shrinking.
- Plantear el sistema de ecuaciones necesario a partir de la secuencia cifrante correspondiente. El sistema de ecuaciones se plantea ya que es conocida la expresión de los coeficientes $b_i = F(c_i)$.
- Resolver el sistema de ecuaciones mediante método de Gauss. Es posible que con los datos propuestos no se pueda resolver, en ese caso habrá que considerar otra suposición. Si se logra resolver el sistema, entonces como salida produce la condición inicial de $\{c_n\}$ de L bits.

- Verificar si cada uno de los datos c_i y b_i hallados son correctos. Esta comprobación se lleva a cabo generando a partir de los resultados obtenidos más secuencia de salida y comparándola con la secuencia interceptada.

Vemos en el siguiente diagrama de flujo, la secuencia de tareas que están programadas. Todos los programas corren en Matlab.

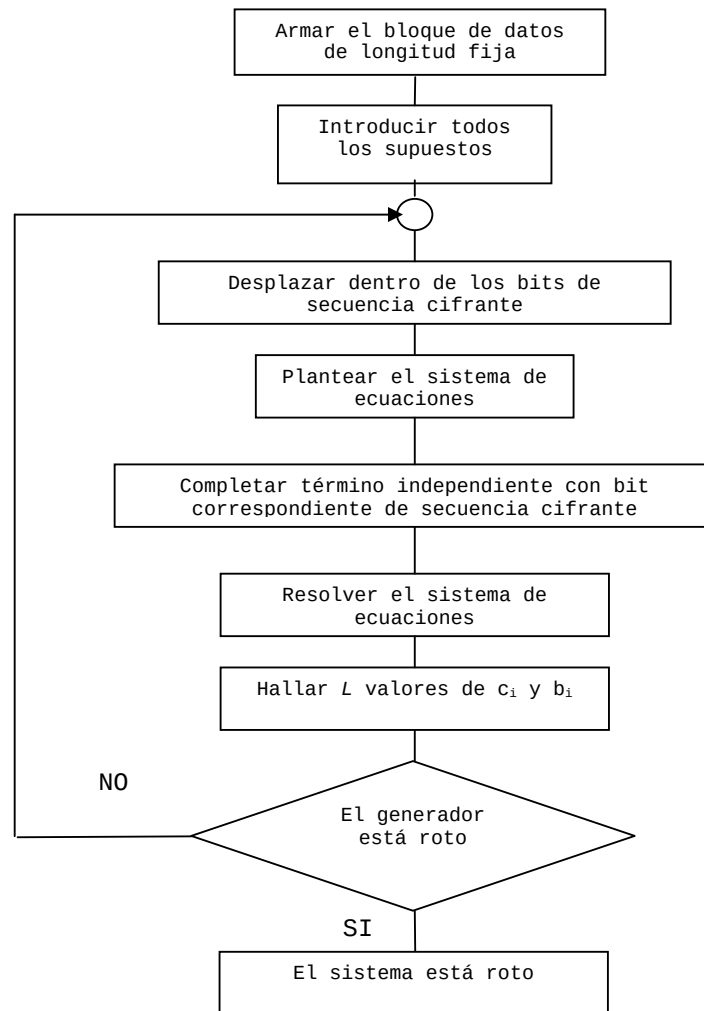


Fig. 2. Diagrama de flujo del proceso para romper el generador auto-shrinking

Existen otras funciones o tareas llevadas a cabo por programas secundarios, que no se corren en tiempo real como ocurre con los anteriores, pero que son necesarios para el criptoanálisis. Las tareas que realizan se pueden agrupar en:

- Determinar F , función que define $b_i = F(c_i)$.
- Armar y guardar la matriz de coeficientes H .
- Generar secuencia cifrante para determinar suposiciones válidas.

Los programas son de índole general, pudiendo ser aplicados en la resolución de un generador auto-shrinking cualquiera sea el polinomio generador del mismo. De hecho, estos programas han evaluado satisfactoriamente polinomios generadores que van desde grado 10 al grado 20. A partir del análisis realizado con registros o polinomios de grados relativamente bajos, [10 - 20], es posible extrapolar los resultados a LFSRs de mayor número de etapas.

4 Otros Criptoanálisis

En esta sección se muestran algunos criptoanálisis efectuados sobre este generador. En el marco del proyecto STORK, Strategic Road Map for Crypto [5], anteriormente referenciado se plantean una serie de problemas abiertos en criptografía. Entre ellos se encuentra el criptoanálisis del generador auto-shrinking logrando órdenes de complejidad menores a los obtenidos aplicando la Técnica de Time-Memory Trade-Off.

A continuación veremos algunas propiedades tanto de la técnica de TMTO, como de otra que ha logrado mejores resultados y que se conoce como Técnica de Guess-and-Determine.

4.1 Time Memory Trade Off

Existen trabajos publicados sobre técnicas criptoanalíticas aplicadas al generador auto-shrinking. En general, los parámetros que se tienen en cuenta son la complejidad de la fase pre-computacional, la complejidad de memoria, la complejidad en tiempo y la cantidad de serie cifrante o secuencia interceptada que se requiere para su rotura.

Aplicando esta técnica de TMTO [5][6] se logran órdenes de $O(2^{0.5*L})$ para la complejidad en tiempo, $O(2^{0.5*L})$ para la complejidad de memoria, $O(2^{0.25*L})$ para la cantidad de secuencia interceptada necesaria con una pre-computación de $O(2^{0.75*L})$. Estos órdenes son elevados y un criptoanálisis en tiempo real para un generador auto-shrinking en un rango de aplicación criptográfica sería sumamente complejo por los recursos que requeriría.

Esta Técnica de TMTO es ampliamente conocida [8] y es aplicada actualmente para el criptoanálisis del algoritmo A5/1 utilizado en comunicaciones cifradas de telefonía celular, en la banda de GSM, [9].

4.2 Guess and Determine

Es conocida en la Literatura una técnica de criptoanálisis, denominada Guess-and-Determine [6], y que ha sido aplicada sobre el generador auto-shrinking por B. Zhang and D. Feng (Científicos de la Academia China de Ciencias). Si bien el criptoanálisis que logran es aparentemente muy eficaz, su puesta en práctica resulta compleja. En efecto, se muestra un ejemplo de criptoanálisis de un generador auto-shrinking de longitud $L = 40$, aunque no resulta tan clara la necesidad que tienen en lo que se refiere a complejidad de memoria ni a la implementación de los supuestos. Más aún, no aparece comentado el porcentaje de aciertos que logran para distintos polinomios ni la cantidad exacta de secuencia interceptada que han utilizado. Logran resolver en forma confiable el generador auto-shrinking con una complejidad de tiempo de $O(2^{0.556*L})$, complejidad de memoria $O(L^2)$, y con sólo $O(2^{0.161*L})$ de bits de secuencia interceptada. En cualquier caso, su trabajo ha sido de relevancia para la elaboración del nuestro.

5 Conclusiones y Trabajo Futuro

El presente trabajo sienta las bases para lograr de cara al futuro una programación del criptoanálisis del generador auto-shrinking que optimice órdenes de complejidad, especialmente en lo que se refiere a tiempos de procesamiento.

Este criptoanálisis se presta a ser programado utilizando programación en paralelo, como ya hemos visto.

El ataque es efectivo y rompe el generador, siempre que se trabaje con las suposiciones correctas. La elección de estas suposiciones se realiza durante el tiempo de pre-procesamiento, por ende el tiempo que consume hallarlas no constituye un tiempo a añadir al ataque en tiempo real.

Romper un generador de aplicación criptográfica no es tarea fácil, de todos modos aquí se presenta una técnica criptoanalítica con parámetros muy claros y bien definidos para su resolución. Si bien todavía es prematuro hablar de tiempos de procesamiento, pues habría que optimizar la programación, si se ha logrado sintetizar una Técnica Criptoanalítica eficaz en la rotura de este generador. Se ha logrado asimismo encontrar un método de aplicación general que es independiente del polinomio generador o de su grado, y que consigue romper el Generador Auto-Shrinking. Para su implementación, sólo hay que variar en un programa el vector o array que representa el polinomio de realimentación de R_I .

Este trabajo se refiere a una propuesta práctica para un criptoanálisis efectivo y pensando en un criptoanalista sin grandes recursos. Concretamente, hemos logrado órdenes de complejidad que están en $O(2^{0.33L})$ en lo que se refiere a secuencia interceptada, $O(L)$ en cuanto a cantidad de memoria consumida y cuya complejidad pre-computacional es sumamente baja, únicamente lo requerido para hallar los supuestos.

Referencias

1. GSM, Global Systems for Mobile Communications, available at <http://cryptome.org/gsm-a512.htm>)
2. Bluetooth, Specifications of the Bluetooth system, Version 1.1, 2001, available at <http://www.bluetooth.com/>
3. Rivest, L.: RSA Data Security, Inc., March 12, 1998. (1999)
4. Golomb, S.W.: Shift Register Sequences, Aegean Park Press, Laguna Hill, California (1982)
5. Stork Project, available at http://www.stork.eu.org/documents/RUB-D6-2_1.pdf
6. Zhang, B., Feng, D.: New Guess-and-Determine Attack on the Self-Shrinking Generator. In: Proc. Asiacrypt'06. LNCS, vol. 4284, pp. 54--68. Springer, Heidelberg (2006)
7. Meier, W., Staffelbach, O.: The Self-Shrinking Generator. In: Proc. Eurocrypt '94. LNCS, vol. 950, pp. 205--214. Springer, Heidelberg (1994)
8. Hellman, M.: A Cryptanalytic Time-Memory Trade-Off. IEEE Trans. Information Theory, 26 (4), 234--247 (1980)
9. Barkan, E., Biham, E., Keller, N.: Instant Cipher Text-Only Cryptanalysis of GSM Encrypted Communication. Technion, Computer Science Department. Technical Report CS2006-07, (2006)

Diseño e Implementación de una Función Hash Basada en Caos

César José Ramírez López, Leobardo Hernández Audelo

Laboratorio de Seguridad Informática, Centro Tecnológico Aragón, Facultad de Estudios Superiores Aragón, Universidad Nacional Autónoma de México, Av. Rancho Seco S/N, Col. Impulsora, C.P 57130, Cd. Nezahualcóyotl, Tel. (55)56231070, Estado de México, México.

cesar_jose_54@yahoo.com.mx, leo@servidor.unam.mx

Resumen. En este trabajo se presenta el diseño y la implementación de una Función Hash basada en Caos llamada *Lúthien-Tiníviel* que procesa bloques de mensajes con longitudes menores a 2^{56} bits y entrega un valor hash de 512 bits. El diseño combina mapas caóticos y aritmética de campos finitos de Galois (GF) para realizar el proceso. Para generar las constantes de ronda se utiliza el mapa bidimensional caótico Baker, el atractor de Lorenz y operaciones en GF para terminar dicho proceso. Estas últimas también son utilizadas para procesar el bloque de datos de entrada correspondiente al i -ésimo bloque de datos del mensaje de entrada. Se utilizan dos tipos diferentes de constantes predeterminadas, las cuales son calculadas a partir de la iteración del atractor de Lorenz y son fijas para cada una de las doce rondas que definen la función hash. El diseño está basado en la función hash Whirlpool propuesta por Paulo S.L.L. Barreto y Vincent Rijmen.

Palabras clave: Función hash, mapa Baker, campos finitos de Galois, sustituciones, permutaciones, irreducibles, atractor de Lorenz, no linealidad, Whirlpool.

1 Introducción

Actualmente las funciones hash son de trascendental importancia en aplicaciones donde se requiere eficiencia para implementar verificación de integridad y autenticación, a través de la generación de códigos MIC/MAC, entre otras aplicaciones. Es de resaltar el hecho de que la función hash MD5, que se había venido utilizando en la mayoría de aplicaciones para estos fines, ha dejado de ser opción debido a las vulnerabilidades encontradas en los últimos años. Aunque en su lugar se está utilizando SHA con 384 bits, es un hecho que se continúa la búsqueda de una nueva función hash que venga a reemplazar a MD5.

El presente trabajo se enmarca en el contexto anterior y pretende que el diseño de la función hash que en él se describe pueda ser una opción viable para dichos fines, ya que actualmente es una propuesta académica de combinación de características de

mapas caóticos con operaciones de campos finitos. De este modo, en el presente trabajo se describe el diseño y la implementación de una función hash, a la que denominamos “*Lúthien – Tinúviel*” (cuyo nombre hace referencia al personaje de ficción creado por J. R. R. Tolkien en el libro *El Silmarillion* hija del rey Thingol y Melian), la cual procesa mensajes cuya longitud máxima debe ser a lo más de 2^{256} bits, obteniendo como resultado un valor hash de 512 bits.

El diseño de *Lúthien – Tinúviel* está basado en la iteración de una misma función de compresión, la cual procesa bloques del mensaje de entrada de 512 bits a manera de cifrado con una llave autogenerada de 512 bits, mediante 12 rondas.

2 Trabajos Previos

La utilización de mapas caóticos en criptografía se ha enfocado principalmente en operaciones de cifrado de imágenes, pero esto no ha limitado su utilización con respecto a las funciones hash.

En la International Journal of Computer Science and Network Security (IJCSNS), Vol. 8 No. 2, publicada en febrero 2008, Mahmoud Maqableh, Azman Bin Samsudim y Mohammad A. Alia, proponen CHA-1 (Chaos Hash Algorithm-1) una función hash basada en teoría del caos y en SHA-1.

De manera general las principales características de CHA-1 [1] son:

- Longitud de bloque 160 bits
- Longitud del valor hash 160 bits
- Longitud máxima del mensaje 2^{80} bits

Puesto que una función hash puede ser vista como un generador de números pseudoaleatorios, dada la aleatoriedad de la salida requerida, un sistema dinámico es muy adecuado para ser utilizado como motor de la función hash. Sin embargo, no todos los mapas caóticos son adecuados para ello; entre los mapas caóticos más prometedores criptográficamente hablando está el mapa logístico.

El mapa logístico tiene entre sus propiedades el ser simple, rápido, sensible a las condiciones iniciales e impredecibilidad, las cuales se pueden transportar a una función hash. El mapa logístico se define como sigue: $X_n = r(I - X_{n-1})X_{n-1}$; donde $r \in [0, 4]$, $X_n \in (0, 1)$ y $n \in N$.

2.1 Chaos Hash Algorithm-1 (CHA-1)

El diseño de la función Chaos Hash Algorithm – 1 tiene cuatro fases principales:

- Primera fase: Utiliza la función R para producir una salida de 160 bits que será la entrada X_0 del mapa logístico en la fase 3.
- Segunda fase: La función X_0 es utilizada para producir el valor r que es el segundo parámetro del mapa logístico de la fase 3.
- Tercera fase: Consiste en encontrar un valor real entre $[3.592, 4.0]$ en la que se define el comportamiento caótico del mapa logístico. Los valores iniciales R y

X_0 son usados como parámetros de entrada del mapa logístico. CHA-1 toma los primeros 64 bits del resultado de la función R para ser agregados al valor de r para calcular el siguiente valor de $X(X_{n+1})$.

- Cuarta fase: CHA-1 usa el resultado de la fase 3 como valor inicial y recursivamente produce X_n usando un valor de 64 bits a la vez del mensaje original.

2.2 Función X_0

El objetivo de la función X_0 es convertir un bloque de 160 bits a un valor real dentro del intervalo $[0.33, 0.59]$ que fungirá como el valor inicial del mapa logístico de la fase 3. Para lograr esto utiliza dos funciones adicionales las cuales producirán los valores X_0 y r del mapa logístico.

Su definición es simple ya que usa 3 operaciones no conmutativas XOR suma mod 2^{32} y corrimientos circulares a la izquierda de 32 bits. El mensaje de entrada es dividido en bloques de 160 bits; dichos bloques a su vez son divididos en 5 de 32 bits, para formar el buffer $ABCDE$ que será procesado por la función X_0 .

La cual produce una salida de 160 bits a la que se le aplica la operación XOR con el bloque siguiente. El resultado se utiliza como entrada a la siguiente iteración de la función X_0 . Este proceso continúa hasta que la función termina de procesar el mensaje. Para el caso en el que el último bloque no tenga la longitud requerida se procede a efectuar el relleno concatenando un 1-bit seguido de 0-bit hasta completar los 160 bits necesarios para formar el bloque.

2.3 Función R

La función R utilizada en CHA-1 es similar a la función X_0 procesa bloques de 160 bits generando una salida de la misma longitud. Esta salida es sumada XOR con el siguiente bloque de entrada; el resultado de esta operación es la siguiente entrada a la función R en la siguiente iteración. Después de procesar todos los bloques del mensaje, el resultado es un bloque de 160 bits. Este valor se utiliza como el valor incremental r en el mapa logístico de la fase 3. Este proceso se repite hasta terminar el mensaje. La función R es una función simple que usa tres operaciones no conmutativas similares a la función X_0 .

Los primeros 64 bits del bloque de salida de la función R son convertidos a un número real de 160 bits entre 0 y 1. Este número se usa en la fase 3 como el valor incremental r del mapa logístico.

2.4 Finalización

Después de aplicar el mapa logístico en la fase 3 se tiene un valor de 160 bits como salida, el cual se usa como valor inicial para el mapa logístico de la fase 4. En la fase 4 se toman 64 bits del mensaje original (cada vez) para ser usados como el valor incremental r . Este proceso se repite con valores de 64 bits hasta que el mensaje se

procese en su totalidad. Al final un valor de 160 bits será un punto en el diagrama de bifurcación y este punto será el valor hash del mensaje.

3 Desarrollo

La función hash *Lúthien – Tinúviel*, es una función Merkle (cf. [2, algoritmo 9.25]) basada en una función de cifrado por bloques a la que se denominará en adelante como *LT-CHA*, la cual opera sobre un bloque de entrada de 512 bits que llamaremos bloque *LT-CHA*. Esta función hash utiliza una llave de estado derivada de los datos de entrada de la ronda en turno. En esta sección se definirán los componentes esenciales que conforman la función hash propuesta, así como los procesos que definen la generación de las llaves.

3.1 Entrada y Salida

El bloque de datos de la función *Lúthien – Tinúviel* consiste de 512 bits, el cual corresponde a un bloque de 64 bytes, los cuales pueden ser agrupados en 8 bloques de 8 bytes. Este bloque se denomina *estado LT-CHA*, y puede ser visto como una matriz $\mu: \text{GF}(2^8)^{64} \rightarrow M_{8 \times 8}[\text{GF}(2^8)]$ y su inversa:

$$\mu(a) = b \Leftrightarrow b_{ij} = a_{8i+j}, 0 \leq i, j \leq 7$$

Gráficamente corresponde a lo siguiente:

$$\text{Bloque LT-CHA} = \begin{pmatrix} P_0 & P_1 & P_2 & P_3 & P_4 & P_5 & P_6 & P_7 \\ P_8 & P_9 & P_{10} & P_{11} & P_{12} & P_{13} & P_{14} & P_{15} \\ P_{16} & P_{17} & P_{18} & P_{19} & P_{20} & P_{21} & P_{22} & P_{23} \\ P_{24} & P_{25} & P_{26} & P_{27} & P_{28} & P_{29} & P_{30} & P_{31} \\ P_{32} & P_{33} & P_{34} & P_{35} & P_{36} & P_{37} & P_{38} & P_{39} \\ P_{40} & P_{41} & P_{42} & P_{43} & P_{44} & P_{45} & P_{46} & P_{47} \\ P_{48} & P_{49} & P_{50} & P_{51} & P_{52} & P_{53} & P_{54} & P_{55} \\ P_{56} & P_{57} & P_{58} & P_{59} & P_{60} & P_{61} & P_{62} & P_{63} \end{pmatrix}$$

Tabl 1. Definición del bloque LT-CHA

3.2 Función de Sustitución

La función *Lúthien – Tinúviel* define una sustitución no lineal $(M_{8 \times 8}[\text{GF}(2^8)] \rightarrow M_{8 \times 8}[\text{GF}(2^8)])$ para cada uno de los elementos que conforman al bloque *LT-CHA*. De manera individual, dicha sustitución puede expresarse de la siguiente forma:

$$\text{SubBytes}(a) = b \Leftrightarrow b_{ij} = S[a_{ij}], 0 \leq i, j \leq 7$$

Esta función de sustitución se define mediante la siguiente tabla:

		Y															
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
X	0	96	FD	53	EF	04	1E	2A	E3	AF	E5	A2	3C	B8	D9	2C	11
	1	0A	EB	5F	79	8C	CA	B3	F3	81	FA	31	18	C8	F0	25	CE
	2	A8	FE	58	CF	02	FF	61	4B	9B	D8	C8	4D	74	EC	54	D0
	3	1D	A3	A0	BD	35	93	D1	06	48	4A	D5	77	3F	B0	BA	5D
	4	F9	8F	D2	E1	F1	B9	3A	D4	DC	68	52	82	6D	36	08	C2
	5	10	EA	B6	7A	C9	70	7B	91	E7	E8	AB	BE	F7	85	C5	6A
	6	23	7D	0C	EE	8D	A1	73	AD	47	DD	64	7F	45	B5	AE	B1
	7	89	F4	88	ED	37	BF	66	1F	32	42	F5	09	8C	20	03	17
	8	21	16	1A	62	C4	A6	2D	A4	55	2F	01	DA	C0	1C	B7	7C
	9	C3	90	99	E8	84	40	9C	67	6B	4E	C6	5A	A9	6C	BC	2E
	A	A5	00	D8	30	86	41	E0	F8	49	DF	95	FB	6C	B2	65	24
	B	5E	8B	DE	D7	78	44	F2	28	26	76	6F	50	4F	6E	98	5B
	C	4C	E4	13	3E	DB	BB	AA	39	1B	57	0D	87	14	AC	0B	D3
	D	0E	05	43	83	9F	97	12	F6	0F	C7	07	63	8A	9A	75	3B
	E	69	3D	A7	59	E9	C1	2B	38	46	33	72	19	9E	71	22	7E
	F	B4	15	FC	E2	27	92	29	CC	9D	34	CD	8E	5C	51	56	94

Tabl2 2. S Box de la función Lúthien – Tinúviel

La tabla de sustitución está calculada con base en el siguiente algoritmo:

- Se obtienen los inversos multiplicativos en $GF(2^8)$ del byte correspondiente, teniendo como polinomio irreducible a $x^8 + x^5 + x^3 + x + 1$.
- Se efectúa la operación matricial consistente en el producto del inverso multiplicativo, obtenido en el paso anterior, representado como un byte, con la matriz 8×8 mostrada a continuación. Al resultado se le suma la constante aditiva $x^7 + x^4 + x^2 + x$ en $GF(2^8)$, la cual corresponde al $0x96$ en hexadecimal, con el objetivo de difundir más los bits, teniendo como salida el valor por el cual se sustituirá el byte de entrada.

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

Fig. 1. Operación $\Delta(P_i)$

- A esta operación se le denominará $\Delta(P_i)$.

3.3 Funciones de Permutación

En la función *Lúthien – Tinúviel* se definen dos funciones de permutación: una basada en el mapa Baker para la generación de las constantes de ronda y una función

adicional basada en corrimientos a nivel de columna y renglón, utilizada en el procesamiento del bloque *LT-CHA*.

El mapa Baker [3] está descrito con las siguientes expresiones:

$$B(x, y) = (2x, y/2) \quad \text{donde } 0 \leq x \leq 1/2$$

$$B(x, y) = (2x - 1, y/2 + 1/2) \quad \text{donde } 1/2 \leq x \leq 1$$

Lo anterior corresponde a la biyección caótica de un cuadrado unitario $[x]$ sobre sí mismo. La columna de la izquierda $[0, 1/2) \times [0, 1)$ es ampliada horizontalmente y reducida verticalmente a $[0, 1) \times [0, 1/2)$. La columna vertical derecha $[1/2, 1) \times [0, 1)$ tiene una correspondencia semejante a la anterior pero sobre $[0, 1) \times [1/2, 1)$.

Basado en la generalización del mapa Baker, se define la permutación utilizada para generar las constantes de ronda de la siguiente manera:

0	1	2	3	4	5	6	7	13	5	22	14	6	23	15	7
8	9	10	11	12	13	14	15	37	29	21	38	30	47	39	31
16	17	18	19	20	21	22	23	61	53	45	62	54	46	63	55
24	25	26	27	28	29	30	31	8	0	9	1	10	2	3	4
32	33	34	35	36	37	38	39	24	16	17	18	19	11	20	12
40	41	42	43	44	45	46	47	33	25	34	26	35	27	36	28
48	49	50	51	52	53	54	55	40	32	41	42	51	43	52	44
56	57	58	59	60	61	62	63	56	48	57	49	58	50	59	60

Fig. 2. Permutación basada en el mapa Baker

La anterior corresponde a una de las permutaciones del mapa Baker para una imagen de 8 pixeles cuando los enteros n no son divisores de 8. En el presente caso se utilizan los valores 3 y 5.

Después de efectuar dicho mapeo, se procede a realizar una permutación de corrimiento a nivel columna, quedando finalmente definida, en términos de los elementos que componen al bloque de constantes *LT-CHA*, como sigue:

$$\begin{bmatrix} P_{0,0} & P_{0,1} & P_{0,2} & P_{0,3} & P_{0,4} & P_{0,5} & P_{0,6} & P_{0,7} \\ P_{1,0} & P_{1,1} & P_{1,2} & P_{1,3} & P_{1,4} & P_{1,5} & P_{1,6} & P_{1,7} \\ P_{2,0} & P_{2,1} & P_{2,2} & P_{2,3} & P_{2,4} & P_{2,5} & P_{2,6} & P_{2,7} \\ P_{3,0} & P_{3,1} & P_{3,2} & P_{3,3} & P_{3,4} & P_{3,5} & P_{3,6} & P_{3,7} \\ P_{4,0} & P_{4,1} & P_{4,2} & P_{4,3} & P_{4,4} & P_{4,5} & P_{4,6} & P_{4,7} \\ P_{5,0} & P_{5,1} & P_{5,2} & P_{5,3} & P_{5,4} & P_{5,5} & P_{5,6} & P_{5,7} \\ P_{6,0} & P_{6,1} & P_{6,2} & P_{6,3} & P_{6,4} & P_{6,5} & P_{6,6} & P_{6,7} \\ P_{7,0} & P_{7,1} & P_{7,2} & P_{7,3} & P_{7,4} & P_{7,5} & P_{7,6} & P_{7,7} \end{bmatrix} \rightarrow \begin{bmatrix} P_{0,7} & P_{1,5} & P_{0,5} & P_{2,6} & P_{1,6} & P_{0,6} & P_{2,7} & P_{1,7} \\ P_{3,7} & P_{4,5} & P_{3,5} & P_{5,6} & P_{4,6} & P_{3,6} & P_{5,7} & P_{4,7} \\ P_{6,7} & P_{7,5} & P_{6,5} & P_{5,5} & P_{7,6} & P_{6,6} & P_{5,6} & P_{7,7} \\ P_{0,4} & P_{1,0} & P_{0,0} & P_{2,1} & P_{0,1} & P_{1,2} & P_{0,2} & P_{0,3} \\ P_{1,4} & P_{3,0} & P_{2,0} & P_{2,1} & P_{2,2} & P_{2,3} & P_{1,3} & P_{2,4} \\ P_{3,4} & P_{4,1} & P_{3,1} & P_{1,2} & P_{3,2} & P_{4,2} & P_{2,2} & P_{4,4} \\ P_{5,4} & P_{3,0} & P_{4,0} & P_{3,1} & P_{5,2} & P_{6,3} & P_{5,3} & P_{6,4} \\ P_{7,4} & P_{7,0} & P_{6,0} & P_{7,1} & P_{6,1} & P_{7,2} & P_{6,2} & P_{7,2} \end{bmatrix}$$

Fig. 3. Permutación utilizada para la generación de constantes de LT-CHA.

A esta permutación se le denominará $\Pi(P_i)$.

Se define una permutación adicional, la cual será aplicada sólo al bloque *LT-CHA* como parte de la obtención del valor hash durante cada una de las rondas. Dicha permutación se define, en base al bloque *LT-CHA*, como sigue:

$$\begin{bmatrix} P_{0,0} & P_{0,1} & P_{0,2} & P_{0,3} & P_{0,4} & P_{0,5} & P_{0,6} & P_{0,7} \\ P_{1,0} & P_{1,1} & P_{1,2} & P_{1,3} & P_{1,4} & P_{1,5} & P_{1,6} & P_{1,7} \\ P_{2,0} & P_{2,1} & P_{2,2} & P_{2,3} & P_{2,4} & P_{2,5} & P_{2,6} & P_{2,7} \\ P_{3,0} & P_{3,1} & P_{3,2} & P_{3,3} & P_{3,4} & P_{3,5} & P_{3,6} & P_{3,7} \\ P_{4,0} & P_{4,1} & P_{4,2} & P_{4,3} & P_{4,4} & P_{4,5} & P_{4,6} & P_{4,7} \\ P_{5,0} & P_{5,1} & P_{5,2} & P_{5,3} & P_{5,4} & P_{5,5} & P_{5,6} & P_{5,7} \\ P_{6,0} & P_{6,1} & P_{6,2} & P_{6,3} & P_{6,4} & P_{6,5} & P_{6,6} & P_{6,7} \\ P_{7,0} & P_{7,1} & P_{7,2} & P_{7,3} & P_{7,4} & P_{7,5} & P_{7,6} & P_{7,7} \end{bmatrix} \rightarrow \begin{bmatrix} P_{2,0} & P_{1,1} & P_{0,2} & P_{7,3} & P_{6,4} & P_{5,5} & P_{4,6} & P_{3,7} \\ P_{3,0} & P_{2,1} & P_{1,2} & P_{0,3} & P_{7,4} & P_{6,5} & P_{5,6} & P_{4,7} \\ P_{4,0} & P_{3,1} & P_{2,2} & P_{1,3} & P_{0,4} & P_{7,5} & P_{6,6} & P_{5,7} \\ P_{5,0} & P_{4,1} & P_{3,2} & P_{2,3} & P_{1,4} & P_{0,5} & P_{7,6} & P_{6,7} \\ P_{6,0} & P_{5,1} & P_{4,2} & P_{3,3} & P_{2,4} & P_{1,5} & P_{0,6} & P_{7,7} \\ P_{7,0} & P_{6,1} & P_{5,2} & P_{4,3} & P_{3,4} & P_{2,5} & P_{1,6} & P_{0,7} \\ P_{0,0} & P_{7,1} & P_{6,2} & P_{5,3} & P_{4,4} & P_{3,5} & P_{2,6} & P_{1,7} \\ P_{1,0} & P_{0,1} & P_{7,2} & P_{6,3} & P_{5,4} & P_{4,5} & P_{3,6} & P_{2,7} \end{bmatrix}$$

Fig. 4. Permutación definida para el bloque LT-CHA.

3.4 Capa de Difusión Lineal

En esta sección se define el procedimiento mediante el cual se combinan los renglones que conforman el bloque *LT-CHA* con una matriz A en $GF(2^8)$, que se define de la siguiente forma:

$$A = \begin{bmatrix} 01_x & 01_x & 03_x & 02_x & 08_x & 04_x & 09_x & 05_x \\ 05_x & 01_x & 01_x & 03_x & 02_x & 08_x & 04_x & 09_x \\ 09_x & 05_x & 01_x & 01_x & 03_x & 02_x & 08_x & 04_x \\ 04_x & 09_x & 05_x & 01_x & 01_x & 03_x & 02_x & 08_x \\ 08_x & 04_x & 09_x & 05_x & 01_x & 01_x & 03_x & 02_x \\ 02_x & 08_x & 04_x & 09_x & 05_x & 01_x & 01_x & 03_x \\ 03_x & 02_x & 08_x & 04_x & 09_x & 05_x & 01_x & 01_x \\ 01_x & 03_x & 02_x & 08_x & 04_x & 09_x & 05_x & 01_x \end{bmatrix}$$

Fig. 5. Matriz de difusión lineal de la función LT-CHA.

Cada byte que conforma el bloque *LT-CHA* se interpreta como un elemento en $GF(2^8)$, y a partir de este se realiza una multiplicación matricial con A , de la siguiente manera:

$$\begin{bmatrix} s'_{i,0} \\ s'_{i,1} \\ s'_{i,2} \\ s'_{i,3} \\ s'_{i,4} \\ s'_{i,5} \\ s'_{i,6} \\ s'_{i,7} \end{bmatrix} = \begin{bmatrix} s_{i,0} \\ s_{i,1} \\ s_{i,2} \\ s_{i,3} \\ s_{i,4} \\ s_{i,5} \\ s_{i,6} \\ s_{i,7} \end{bmatrix} \cdot A = \begin{bmatrix} s_{i,0} \\ s_{i,1} \\ s_{i,2} \\ s_{i,3} \\ s_{i,4} \\ s_{i,5} \\ s_{i,6} \\ s_{i,7} \end{bmatrix} \cdot \begin{bmatrix} 01_x & 01_x & 03_x & 02_x & 08_x & 04_x & 09_x & 05_x \\ 05_x & 01_x & 01_x & 03_x & 02_x & 08_x & 04_x & 09_x \\ 09_x & 05_x & 01_x & 01_x & 03_x & 02_x & 08_x & 04_x \\ 04_x & 09_x & 05_x & 01_x & 01_x & 03_x & 02_x & 08_x \\ 08_x & 04_x & 09_x & 05_x & 01_x & 01_x & 03_x & 02_x \\ 02_x & 08_x & 04_x & 09_x & 05_x & 01_x & 01_x & 03_x \\ 03_x & 02_x & 08_x & 04_x & 09_x & 05_x & 01_x & 01_x \\ 01_x & 03_x & 02_x & 08_x & 04_x & 09_x & 05_x & 01_x \end{bmatrix}$$

Fig. 6. Definición de la capa de difusión lineal en LT-CHA.

Como consecuencia de esta operación matricial el primer byte del renglón, por ejemplo, es remplazado por:

$$s'_{i,0} = s_{i,0} \oplus (s_{i,0} \cdot 05_x) \oplus (s_{i,2} \cdot 09_x) \oplus (s_{i,3} \cdot 04_x) \oplus (s_{i,4} \cdot 08_x) \oplus (s_{i,5} \cdot 02_x) \oplus (s_{i,6} \cdot 03_x) \oplus s_{i,7}$$

El símbolo “ $\oplus$ ” expresa la suma en $\text{GF}(2^8)$, la cual corresponde a la operación XOR a nivel de bits. Las multiplicaciones indicadas se realizan modulo el polinomio irreducible que, para el caso de *LT-CHA*, es $x^8 + x^5 + x^3 + x + 1$.

3.5 Suma de la Llave de Ronda

La suma de la llave $\sigma[k] : M_{8 \times 8}[\text{GF}(2^8)] \rightarrow M_{8 \times 8}[\text{GF}(2^8)]$ generada se realiza a nivel de bits (xor), visualizándola como una matriz $k \in M_{8 \times 8}[\text{GF}(2^8)]$:

$$\sigma[k](a) = b \Leftrightarrow b_{ij} = a_{ij} \oplus k_{ij}, 0 \leq i, j \leq 7$$

Esta operación se utiliza también para la generación de las constantes de ronda.

3.6 Función de Generación de Llaves de Ronda

En esta sección se definirá el algoritmo utilizado para calcular las constantes de ronda, las cuales fungirán como llaves de ronda para la función *Lúthien – Tinúviel*. Para obtenerlas se utilizará el atractor de Lorenz de la siguiente forma:

ALGORITMO_GENERADOR_DE_LLAVES_DE_RONDA:

ObtieneConstantesDelAtractorLorenz()

Calcula las constantes de ronda a partir del atractor de Lorenz utilizando como punto fijo el establecido, así como la organización y concatenación de los valores obtenidos.

Variables:

1. Punto inicial del atractor de Lorenz: $P_0 = \langle x, y, z \rangle = \langle 1, 1, 1 \rangle$
2. Definición de parámetros: $A = 10, B = 28, C = 8/3$.
3. Incremento en el valor de los parámetros $\Delta = 0.01$
4. $i, j = \text{enteros}$.

Pasos:

1. Se inicializan las variables $i = 0, j = 0$.
2. Se obtiene una lista de 104 constantes de 64 bits agrupadas en 13 grupos de 8.
 - While $j < 104$ do
 - For $i = 0$ to 10 do
 - $x' = a(x - y)$
 - $y' = x(b - z) - y$
 - $z' = xy - cz$
 - $x = x + \Delta x'$
 - Se suprime la parte entera de x, y e z . La parte fraccionaria se multiplica por 100000000 ó 1×10^8 , ($\lfloor a_i \rfloor$ obtiene sólo la parte fraccionaria).
 - $x = 100000000 * \lfloor x \rfloor$
 - $y = 100000000 * \lfloor y \rfloor$
 - $z = 100000000 * \lfloor z \rfloor$

- Se convierte a hexadecimal el valor absoluto de los valores anteriores, con lo que se obtienen 3 constantes de 32 bits en cada iteración.
- Los valores obtenidos se agrupan en bloques de 64 bits, siguiendo el siguiente orden: $x_0 | y_0 | z_0 | x_1 | y_1 | z_1 | \dots | x_n | y_n | z_n$

3. Regresa los 13 grupos de 8 constantes de 64 bits calculados.

3.7 La Función de Ronda *LT-CHA*

Con los planteamientos anteriormente descritos, la función hash *Lúthien – Tinúviel* se define de la siguiente manera:

$$LTCHA[k] = SubBytes(k) \circ PermutaBytes(k) \circ DifusionLineal(k) \circ AddKey(k')$$

Dicho de otra manera: para procesar cada bloque $M(i)$ – bloque i^{th} de 512 bits – es necesario realizar los siguientes pasos:

INICIA LT-CHA

FASE DE PREPROCESAMIENTO DEL MENSAJE (M)

SE OBTIENEN T BLOQUES DE 64 BYTES

PARA $i = 0$ HASTA T HAZ

INICIO

// Se forma el bloque LT-CHA(S) con el i^{th} bloque xor
// el hash anterior.

$S = M[i] \oplus H[i-1]$ con $H[0] = 0$

//Se asigna la llave inicial del conjunto de Lorenz

$K[0] = H[i-1] \oplus KR[0][0..7]$

// Se calcula el hash como sigue

PARA $r = 1$ HASTA R HACER

INICIA

// Primero se calculan las llaves de ronda

$L =$ Substitución de bytes del bloque LT-CHA (K^{r-1})

$L =$ Permutación Baker del bloque LT-CHA (K^{r-1})

$L =$ Difusión lineal del bloque LT-CHA (K^{r-1})

$L =$ Suma las constantes de ronda (K^{r-1} , KR^r)

// Con esto se obtienen las constantes de ronda K^r

// Se procesa con éstas el bloque LT-CHA

$S =$ Substitución de Bytes del bloque LT-CHA (S)

$S =$ Permutación del bloque LT-CHA (S)

$S =$ Mezcla de renglones del bloque LCH (S)

$S =$ Suma la constante de ronda (S, K^r)

FIN PARA

// Se aplica la operación Miyaguchi-Preneel.

$H[i+1] = H[i] \oplus S \oplus M[i]$

FIN PARA

FIN LT-CHA

3.8 Relleno y Fortaleza MD

El algoritmo *LT-CHA* sigue lo indicado por R. Merkle y Damgård [3] para la fase de expansión del preprocesamiento. La longitud máxima en bits L de un mensaje M válido, para la función hash Lúthien–Tinúviel (LT-CHA), es $L < 2^{256}$. A partir de este se le concatena un 1-bit, seguido de los 0-bits necesarios para cumplir la condición de formar un bloque cuya longitud sea un múltiplo impar de 256 y, finalmente, una representación binaria de 256 bits, la cual representa la longitud original del mensaje, con lo que se obtiene un mensaje relleno m , el cual será dividido en t bloques $m_1, \dots, m_t$. Estos bloques son vistos como arreglos de bytes por la agrupación secuencial de pedazos de 8 bytes.

4 Pruebas y Resultados

Se realizó un análisis estadístico a la función *Lúthien–Tinúviel* y se compararon los resultados con los mismos resultados conocidos para la función hash *Whirlpool* [3].

4.1 Análisis Estadístico

Para el análisis estadístico se procede, en primera instancia, a generar el espacio muestral definido de la siguiente manera: “El conjunto formado por los valores hash de las cadenas de 0-bits cuya longitud L esta en el intervalo $0 \leq L < 2048$ ”. Se obtuvieron los siguientes resultados:

LT-CHA	32.0434	31.91992	31.97412	31.96387	31.93604	31.88379	31.81104	31.92285
Whirlpool	31.9458	31.97217	32.02637	31.97314	31.93359	31.98047	32.1333	32.14746

Tabla 3. Media del número de bits por cada bloque de 64 bits.

LT-CHA	4.024651	4.094875	3.883536	3.937954	3.923625	3.981914	3.978081	4.010595
Whirlpool	3.995419	3.934486	4.006256	3.959112	3.959695	3.970424	4.120773	4.005701

Tabla 4. Desviación media de cada uno de los bloques de 64 bits

Graficando los resultados obtenidos, se observa lo siguiente:

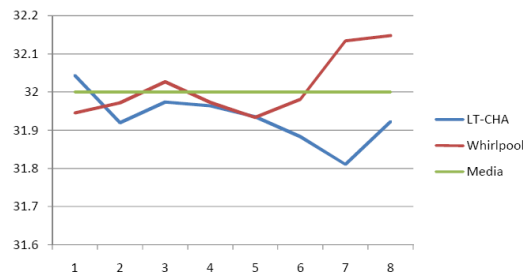


Fig. 7. Media en bits de cada una de las funciones comparadas

Un resultado obtenido de este espacio muestral es que la función *LT-CHA* contiene en promedio 255.4550781 bits con valor 1-bit. En la función Whirlpool la cantidad es de 256.1123 bits; con lo que se podría decir que el hash calculado por Whirlpool tiene al menos la mitad con valor de 1-bit y LT-CHA con valor de 0-bit. Dicha distribución no es uniforme para el *estado* definido, sino que es relativamente diferente entre ambas funciones, como se ve en la gráfica anterior, la cual indica la cantidad de 1-bit para cada bloque de 64 bits. En esta gráfica se observa un comportamiento muy marcado en los bloques 1, 6, 7 y 8, siendo semejante en los restantes.

Graficando la desviación estándar se tiene:

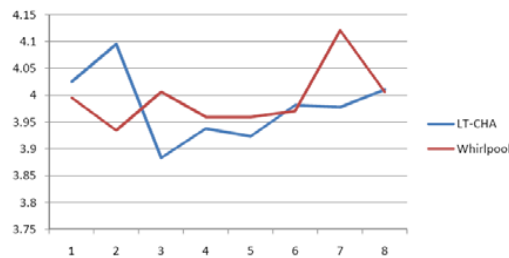


Fig. 8. Desviación estándar para cada una de las funciones hash.

Un comportamiento semejante se obtiene al calcular la desviación estándar del espacio muestral definido.

Resultados semejantes se obtienen al procesar el espacio muestral definido como: “El conjunto *S* formado por todos los hash values de las cadenas de 1024-bits conteniendo un solo 1-bit”. Estos resultados son los siguientes:

LT-CHA	31.95605	32.03906	31.82324	32.03809	32.08105	31.87598	32.17578	31.98438
Whirlpool	31.96387	31.80957	32.22949	32.2207	31.9375	32.03027	31.97949	31.93457
Media	32	32	32	32	32	32	32	32

Table 5. Media del número de bits por cada bloque de 64 bits.

LT-CHA	4.031979	3.997367	3.984223	3.966598	3.926357	4.023764	3.979483	4.017023
Whirlpool	3.983813	4.034261	3.982024	4.001479	3.99487	4.138244	3.919424	4.029992

Table 6. Desviación media de cada uno de los bloques de 64 bits

4.2 Pruebas de Rendimiento

Se realizaron pruebas de rendimiento entre las funciones *LT-CHA* y Whirlpool, con el fin de medir el rendimiento de las mismas cuando éstas son aplicadas a bloques de datos de diferentes longitudes.

A continuación se describen las diferentes pruebas de rendimiento que fueron efectuadas para realizar la comparación:

1. “**Procesar un mismo bloque de datos el número de veces necesario para simular un mensaje de longitud L** ”: En esta prueba no se realizan accesos directos al disco duro, por lo que el tiempo obtenido sólo corresponderá al tiempo que tarda cada función en procesar la cantidad de bloques necesarios para un mensaje de longitud específica.
2. “**Iterar 100000000 veces la función hash (*LT-CHA* y *Whirlpool*) con un mismo bloque de datos**”: El tiempo obtenido corresponderá al necesario para realizar esta operación.

Estas pruebas se realizaron en un equipo con las siguientes características: “AMD Turion 64 Mobile Processor, 1.8 GHz, 512 KB L2 Cache, 1800 MHz de bus de sistema, 1.5 Gb en RAM, 80 Gb de disco duro”. Se obtuvieron los siguientes resultados:

DATOS	LT-CHA(s)	Whirlpool(s)
1150 KB	0.0099999998	0.0099999998
2 MB	0.0299999993	0.0199999996
10 MB	0.1400000006	0.1199999973
50 MB	0.7200000286	0.5799999833
500 MB	7.2399997711	5.8499999046
1 GB	14.8299999237	11.9700002670
8 GB	118.2099990845	95.9499969482
40 GB	590.9799804688	479.4299926758

Table 7. Tabla comparativa (*LT-CHA* y *Whirlpool*) para procesar un bloque de longitud L .

Graficando estos resultados obtenemos:

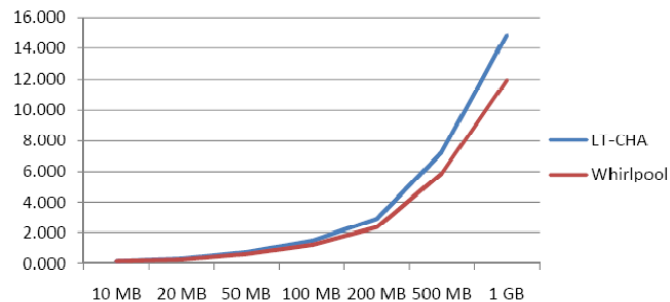


Fig. 9. Rendimiento de Whirlpool y LT-CHA.

En la gráfica se observa que para un conjunto de valores, *LT-CHA* tiene el mismo rendimiento que Whirlpool pero, para bloques de datos mayores a 4MB, el

rendimiento de Whirlpool tiende a ser mejor que *LT-CHA* y esta diferencia se incrementa conforme aumenta la cantidad de bloques a procesar.

Los resultados correspondientes a la segunda prueba son:

No. de Veces	LT-CHA	Whirlpool
100000000	1618.8900146484	1350.4899902344
	1625.2900390625	1348.9399414062
	1620.0000000000	1348.9200439453
	1619.7900390625	1348.6899414062
	1618.3399658203	1350.2800292969

Table 8. Rendimiento en la iteración de las funciones Whirlpool y LT-CHA.

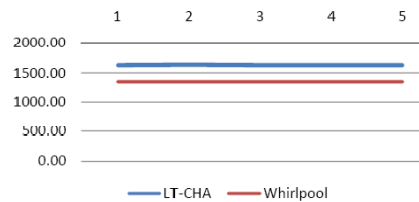


Fig. 10. Comparación en eficiencia al iterar a la función Whirlpool y LT-CHA

Se observa que *LT-CHA* toma más tiempo que Whirlpool en procesar 1×10^8 veces la función hash aunque este resultado no sea notorio al procesar una pequeña cantidad de bloques de datos.

4.3 Seguridad de la Función de Compresión

El esquema Miyaguchi-Preneel es uno de los pocos métodos sin romper utilizados para la construcción de funciones hash a partir de un cifrado por bloques [4]. Sus propiedades de seguridad son discutidas en [2, sección 9.4.1]. En particular, esta es “probablemente segura” si varias propiedades ideales se mantienen para el cifrado por bloque. Los resultados de desarrollos recientes realizados por Black, Rogaway y Shrimpton [5] acerca de las propiedades de seguridad del esquema Miyaguchi-Preneel y otros esquemas, desde la perspectiva de una caja negra, corroboran de manera cuantitativa la elección realizada para Lúthien-Tinúviel.

5 Mejoras por Realizar

Una característica particular del diseño de la función hash presentada en este trabajo es la utilización de mapas caóticos para la generación de las llaves de ronda. Para realizar esto se utilizan valores precalculados (constantes obtenidas a partir de la

iteración del atractor de Lorenz), además de la definición de un algoritmo para su generación basado en operaciones en $GF(2^8)$. No obstante, la utilización de mapas caóticos es sólo para las constantes de ronda. Una posible mejora al diseño es que también se utilicen estos mapas en el procesamiento del bloque *LT-CHA*.

La utilización de mapas caóticos continuos en la definición de primitivas criptográficas es relativamente reciente debido, en gran parte, a lo costoso en ciclos de reloj del cálculo de valores de punto flotante, ya que la exactitud de los mismos está limitada a la capacidad de poder expresarlos y a la arquitectura utilizada. Por esto, generalmente se utiliza en combinación con funciones no lineales que tengan operaciones de corrimiento, sustitución a nivel de bits a fin de que el cálculo sea lo más “determinístico” posible. Una solución a este problema es la definición de ventanas de rangos de valores que permitan “establecer” el valor calculado previamente y que será utilizado. Esta solución resolvería en parte el problema de la exactitud del cálculo, pero la selección y el orden de elección de los valores debe ser resistente al criptoanálisis diferencial, no debe tener una secuencia lineal y que los valores definidos estén lo más distribuidos posibles en el rango de valores idóneos. Ya que de este rango depende la definición del objeto fractal, se deben considerar los valores que estén fuera de la órbita generada por la definición del objeto, tales como los definidos por el conjunto de Julia.

Cabe destacar que este planteamiento aplica sólo para mapas caóticos continuos, para el caso de mapas discretos no aplica por obvias razones, el planteamiento original de esta propuesta es la posibilidad de utilizar mapas caóticos lineales para el desarrollo de funciones hash.

Una prueba que está en realización es el criptoanálisis diferencial (del algoritmo generación de sub-llaves, junto con el procesamiento del bloque *LT-CHA*) con un conjunto de valores mayor, a fin de determinar si existen o no características diferenciales, así como también determinar su probabilidad, para asegurarse que la definición de la función es resistente a este tipo de ataques.

Se realizó un criptoanálisis diferencial para establecer si se efectuaba o no el mapeo Baker para la generación de constantes y para el procesamiento del bloque *LT-CHA*, llegando a la conclusión de que sólo se utilizaría para la generación de constantes, debido a que al utilizarlo en ambos se tiene un alto grado de linealidad.

Otra de las pruebas que se encuentran en realización son las referentes a los ataques del cumpleaños, de primera y de segunda preimagen, a fin de determinar el grado de seguridad que se obtiene. Se realizó un análisis estadístico para determinar el grado de distribución de los valores en los que se observó que la función tiene una distribución más uniforme en los conjuntos de mensajes que se definieron.

6 Conclusiones

En el presente trabajo se presentó el diseño y análisis de la función *Lúthien-Tinúviel*, cuyo diseño se basa en la utilización del mapa Baker, el atractor de Lorenz y en operaciones de $GF(2^8)$, cuyo carácter es académico. Los dos primeros son utilizados como el factor no lineal más importante para la generación de constantes,

tomando ventaja de sus características caóticas no lineales. Las operaciones en $GF(2^8)$ son utilizadas para las operaciones aritméticas no lineales en el procesamiento del bloque LT-CHA. El diseño aquí planteado combina operaciones caóticas con operaciones de campos finitos. Dicha combinación no tiene antecedentes conocidos en la literatura, por lo que puede considerarse innovador en este sentido.

En este mismo sentido el mapa Baker ha sido utilizado en algoritmos de cifrado caótico de imágenes pero no se tiene constancia de su utilización en funciones hash (al menos no encontramos un documento que así lo establezca).

La utilización del mapa Baker tanto en la generación de constantes como en el procesamiento del bloque LT-CHA, genera lineabilidad debido a que si el bloque de entrada presente pocos cambios sólo varía en un bit el mapa Baker solo permuta a los bytes como si fueran los píxeles de una imagen (a nivel de renglón y columna), para las regiones significativas no refleja dicho cambio, al utilizar una permutación a nivel de renglón en su mayor parte los cambios se reflejan mejor, con lo que se pierde dicha lineabilidad.

La combinación de elementos caóticos con operaciones no lineales a nivel de bytes, utilizando operaciones en $GF(2^8)$, permiten mantener una distribución más uniforme de los bits, según lo mostrado en los resultados del análisis estadístico que evalúan la variación con respecto a la media y a la desviación estándar. Esto muestra que el diseño presentado tiene mejores resultados que los obtenidos por Whirlpool.

No obstante, es necesario realizar las pruebas de seguridad faltantes a fin de verificar el nivel de resistencia a colisiones, primera y segunda preimagen. Dependiendo de los resultados que se obtengan en las pruebas que están realizándose, se podrán derivar algunas otras conclusiones sobre el presente diseño.

7 Referencias

1. Mahmoud Maqableh, Azman Bin Samsudin y Mohammad A., "New Hash Function Based on Chaos Theory (CHA-1)", en International Journal of Computer Science and Network Security (IJCSNS), vol. 8 No. 2, publicada en febrero 2008
2. A. J. Menezes, P.C. van Oorschot, and S.A. Vanstone, "Handbook of applied cryptography", CRC Press, 1997.
3. Van Rompay Bart, "Analysis and Design of Cryptographic Hash Functions, MAC Algorithms and Block Ciphers", Tesis para alcanzar el grado de doctor por la Katholieke Universiteit Leuven – Faculteit Toegepaste Wetenschappen Departement Elektrotechniek-Esat, Junio 2004, pp. 240.
4. B. Preneel, Analysis and design of cryptographic hash functions, Ph. D. thesis, Katholieke Universiteit Leuven, January 1993.
5. J. Black, P. Rogaway, and T. Shrimpton, "Black-box analysis of the block-cipher-based hash-function constructions from PGV", Advances in Cryptology – Crypto'2002. Lectures Notes in Computer Science, vol. 2442, Springer-Verlag, 2002, pp. 320-335.

Gestión Automatizada de Requisitos de Seguridad para Proyectos de Desarrollo de Líneas de Producto Software

Jesús Rodríguez¹, Daniel Mellado², Eduardo Fernández-Medina¹ y Mario Piattini¹

¹ Universidad de Castilla – La Mancha, Departamento de Tecnologías y Sistemas de Información, Grupo de Investigación Alarcos, Universidad de Castilla – La Mancha, Paseo de la Universidad 4, 13071 Ciudad Real, España. (+34 925268800)
{Jesus.RValencia, Eduardo.FdezMedina, Mario.Piattini}@uclm.es

² Agencia Española de Administración Tributaria, Equipo de Auditoría Informática, Paseo de la Castellana, 108, 28046 Madrid, España.
Daniel.Mellado@uclm.es

Resumen. Tanto la seguridad como la ingeniería de requisitos son factores clave para el éxito en el desarrollo de una línea de productos software, debido a que la compleja estructura de dependencias entre las características de la línea de productos y de cada producto en particular puede verse amenazada si se encuentra una debilidad de seguridad en las características de la línea, pudiéndose propagar este problema a todos los productos de la misma. La aplicación de procesos de ingeniería de requisitos de seguridad o metodologías para definir requisitos de seguridad que mitiguen las posibles amenazas sobre una línea se convierte en una tarea muy difícil debido a que sería necesario realizarla manualmente. Por lo tanto, en este artículo presentamos una herramienta llamada SREPPLineTool que proporciona soporte automatizado, guiado e intuitivo para facilitar la aplicación del proceso de ingeniería de requisitos de seguridad para líneas de producto software que hemos desarrollado, denominado SREPPLine.

Palabras clave: requisitos de seguridad, línea de producto, ingeniería de requisitos de seguridad, Criterios Comunes, seguridad.

1 Introducción

En los últimos años se han observado una gran cantidad de organizaciones que se han visto comprometidas debido a la aparición de brechas de seguridad. De hecho, el número de vulnerabilidades que han aparecido en aplicaciones ha aumentado de 171 en 1995 a 7.236 en 2007 [1], esto está motivado en parte por la tendencia hacia sistemas de mayor envergadura que se distribuyen a través de Internet introduciendo de esta manera un gran conjunto de amenazas de seguridad que podrían manifestarse [2], lo que implica que hoy en día los sistemas de información son susceptibles de verse amenazados por cyber-ataques, usuarios maliciosos, ciber-terroristas, etc. [3].

Las Líneas de Productos Software (LPS) se han convertido en el enfoque más exitoso en el campo de la reutilización de componentes, reduciendo de un modo significativo tanto el coste de desarrollo como el tiempo de comercialización. En los sistemas de software complejos como las LPS, la seguridad es una preocupación transversal que debe ser objeto de un cuidadoso análisis y de aplicación de técnicas de ingeniería de requisitos adaptadas a la problemática de las líneas de producto [4].

Debido a esto, la seguridad en el software se está convirtiendo en un factor de gran interés para los ingenieros software [5]. Esto tiene como consecuencia que la disciplina de Ingeniería de Requisitos de Seguridad se considere una parte importante de la Ingeniería de Seguridad aplicada al proceso de desarrollo de sistemas de información, sin embargo hasta hace poco tiempo no se le había prestado mucha atención [6]. Esta disciplina aporta técnicas, métodos y estándares claves para conseguir productos y LPS seguras, además de asegurar que se cumplen los requisitos de seguridad definidos y las propiedades de seguridad establecidas en el modelo de variabilidad de las LPS a lo largo de todo el ciclo de vida.

A pesar de lo importante que resultan las disciplinas anteriores para la gestión de la seguridad en las LPS, las metodologías de ingeniería del software y las propuestas de estándares de ingeniería de LPS han ignorado tradicionalmente tanto los requisitos de seguridad como las cuestiones más específicas como la gestión de la variabilidad de seguridad, y aunque existen varios trabajos relacionados con procesos y herramientas orientadas a la gestión de requisitos de seguridad, ninguno de ellos da cobertura al paradigma de desarrollo de LPS, principalmente porque no tienen en cuenta la gestión de la variabilidad de los requisitos de seguridad.

En este artículo, describimos el prototipo de una herramienta de gestión de requisitos de seguridad para LPS que hemos desarrollado para proporcionar soporte automatizado al proceso SREPPLine (Proceso de Ingeniería de Requisitos de Seguridad para Líneas de Producto Software) [7], llamada SREPPLineTool. SREPPLineTool permite ejecutar el proceso SREPPLine de una forma sistemática, guiada e intuitiva. También facilita la integración con los Criterios Comunes (CC) [8] y con el estándar ISO/IEC 27001 [9] dentro del proceso de desarrollo software de los productos de una LPS, así como contribuye al cumplimiento del estándar IEEE 830:1998 [10]. Para conseguir esto, la herramienta se ayuda de las funcionalidades en cuanto a gestión de requisitos ofrecidas por 'IBM Rational RequisitePro' (herramienta CARE que extiende SREPPLineTool).

Además, la herramienta ayuda a desarrollar LPS y productos conforme a los estándares de seguridad más relevantes relacionados con la gestión de requisitos de seguridad de una forma asistida y haciendo de esta forma que no sea imprescindible tener un conocimiento absoluto de los estándares y reduciendo así la participación de expertos en seguridad, es decir, la herramienta mejora la eficiencia de SREPPLine. Asimismo, y gracias al modelo de referencia de seguridad implementado en SREPPLineTool, la gestión y visualización de los artefactos de variabilidad y su

trazabilidad se convierte en una tarea mucho más sencilla e intuitiva, mejorando así la calidad sucesivamente.

El resto del artículo está organizado de la siguiente forma: En la sección 2, se resumen algunas de las características básicas del proceso SREPPLine con el objetivo de que se entienda la posterior explicación de la herramienta. En la sección 3, introducimos las características de la herramienta, se explica detenidamente la funcionalidad de la herramienta aplicada sobre un caso de estudio que nos sirve para una validación inicial de la misma y expondremos las lecciones aprendidas. En la sección 4 presentamos el trabajo relacionado. Y finalmente, en la última sección comentaremos el trabajo futuro y las conclusiones.

2 Visión general de SREPPLine: Ingeniería de Requisitos de Seguridad para Líneas de Productos Software

Una línea de productos software es un sistema intensivo de software que comparte un conjunto común gestionado de características que satisfacen unas necesidades específicas de un segmento particular del mercado y que son desarrollados de una forma pre-establecida a partir de un conjunto común de componentes [11].

El paradigma de la ingeniería del software en líneas de producto diferencia dos procesos: la ingeniería del dominio, que se encarga de definir las partes comunes y variables de una línea; y la ingeniería de la aplicación, que es el proceso de la ingeniería de las LPS en el que se desarrollan los productos de la línea reutilizando artefactos del dominio y explotando la potencia de la variabilidad de la línea de productos [12].

El proceso SREPPLine [7] es un add-in de actividades (las cuales se muestran en la Fig. 1 usando la notación de SPEM 2.0 [13]), que se integran sobre el proceso de desarrollo de LPS existente en una organización, proporcionándole un enfoque en ingeniería de requisitos de seguridad específico para LPS. En este proceso hemos definido las actividades clave que deben ser parte de cada desarrollo de LPS. El orden en el que deben ser realizadas estas actividades depende del proceso particular que este establecido en cada organización. En concreto, proceso se integra dentro del marco de trabajo propuesto por Pohl et al. en [12], y está compuesto de dos subprocesos con sus respectivas actividades: el subproceso de Ingeniería de Requisitos de Seguridad del Dominio de las LPS (PLSecDomReq) y el subproceso de Ingeniería de Requisitos de Seguridad de la Aplicación de las LPS (PLSecAppReq). Aunque podría integrarse en otros procesos de desarrollo con las correspondientes adaptaciones puntuales en ciertas tareas concretas de las actividades que lo componen.

Es un proceso basado en características y metas de seguridad, dirigido por el riesgo y los estándares de seguridad (más concretamente por la norma ISO/IEC 27001[9] y los Criterios Comunes [8]), que se ocupa de la elicitación de requisitos de seguridad y de la gestión de los artefactos relacionados con dichos requisitos, desde las primeras

etapas del desarrollo de las LPS de una forma intuitiva y sistemática especialmente adaptada para el desarrollo basado en el paradigma de las LPS. SREPPLine utiliza las últimas técnicas ampliamente extendidas para elicitar y modelar requisitos de seguridad, como casos de uso de seguridad [14] o casos de mal uso [2], además de la integración en el ciclo de vida de las LPS de los componentes pertenecientes a los Criterios Comunes (CC) (ISO/IEC 15408) y los controles descritos en el estándar ISO/IEC 27001, facilitando así la certificación de seguridad de los productos desarrollados en una LPS determinada.

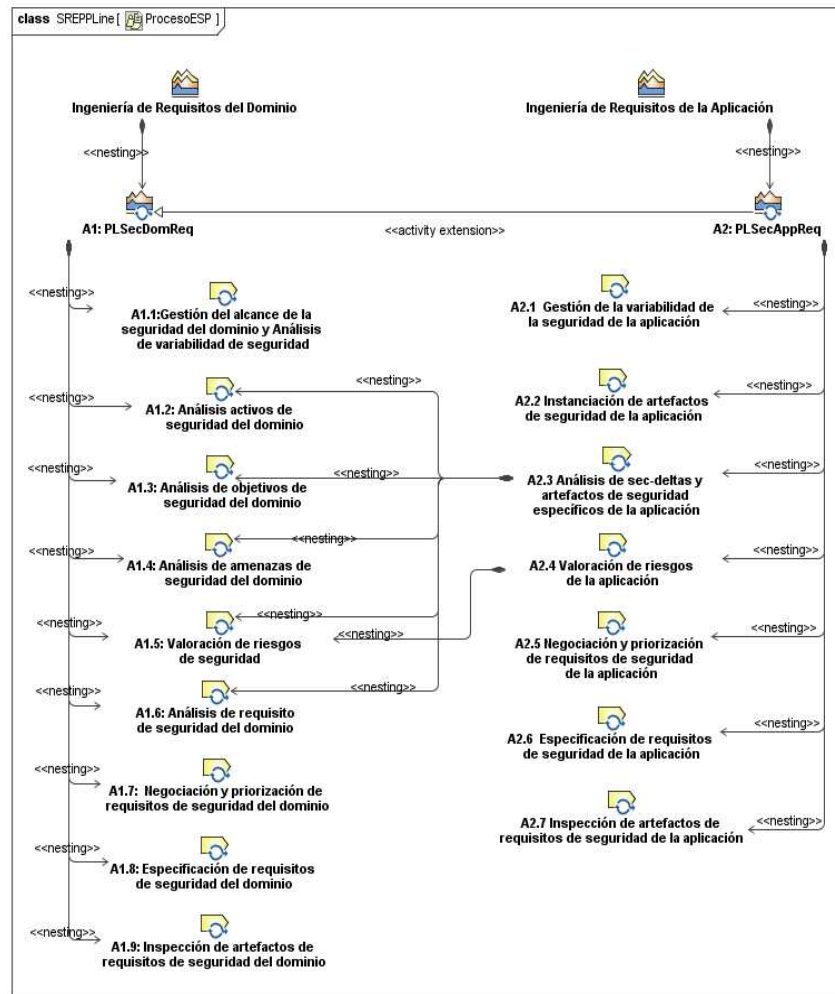


Fig. 1 Especificación de las actividades de SREPPLine usando SPEM 2.0

Además, el proceso SREPPLine sugiere el uso de un método para llevar a cabo la evaluación del riesgo que se ajuste al estándar ISO/IEC 13335 [15], concretamente para la evaluación del riesgo tanto de la línea como de los productos de una línea actualmente SREPPLine y SREPPLineTool utilizan la metodología Magerit [16] (que es conforme a dicho estándar y es una metodología oficialmente reconocida y utilizada en el ámbito de sistemas de información de la OTAN [17] y por la OCDE [18]).

También, como se menciona anteriormente, SREPPLine tiene el objetivo de minimizar el conocimiento necesario de estándares de seguridad así como la participación de expertos de seguridad durante el desarrollo. Para este fin, se propone el uso de un Repositorio de Activos de Seguridad facilitando así la reutilización de artefactos y la implementación del Meta Modelo de Referencia de Seguridad, el cual está compuesto por el Sub-Meta Modelo de Variabilidad de Seguridad y el Sub-Meta Modelo de Decisión de Requisitos de Seguridad, que dan soporte a la gestión de la variabilidad y trazabilidad de los artefactos de seguridad relacionados con las LPS y sus productos, como se detalla en SREPPLine [19].

Este meta modelo, el cual se explica en detalle en [19], es la base a través de la cual SREPPLine define, representa, y almacena el conocimiento sobre todos los artefactos de seguridad de una LPS, ayudando así a una posible certificación contra los estándares de seguridad más extendidos relacionados con la Ingeniería del Software. En esencia, estos modelos representan un repositorio de conocimiento estructurado para dar soporte a los requisitos de seguridad en la ingeniería de requisitos de la LPS.

3 SREPPLineTool

Para dar soporte automatizado a las actividades del proceso SREPPLine, hemos desarrollado una herramienta prototipo CARE (Computer Aided Requirements Engineering), llamada SREPPLineTool que nos permite aplicar el proceso en escenarios de aplicación reales donde el número de artefactos es elevado y la complejidad de su gestión manual se complica. La aplicación de SREPPLine nos permitió obtener experiencia para refinar el proceso y la herramienta, y así mejorar la calidad sucesivamente. La herramienta implementa el Meta Modelo de Referencia de Seguridad (explicado en [19]) por medio de los repositorios dinámicos de artefactos de seguridad, y guiando la ejecución del proceso de forma secuencial. Pudiendo así proponer artefactos de seguridad relacionados con cada actividad del proceso SREPPLine dependiendo de las categorías de artefactos del dominio de un proyecto de desarrollo de LPS.

Además, la forma de interacción con SREPPLineTool es guiada de una forma intuitiva, llevando la ejecución de una etapa a la siguiente, manteniendo en todo momento una representación visual de los artefactos que se manejan en dicha etapa y permitiendo de esta manera una gestión sencilla tanto de la variabilidad como de la trazabilidad entre los distintos artefactos de seguridad que se tratan en cada etapa. Así mismo, es posible generar documentos de seguridad sobre un proyecto de LPS y

exportar información sobre los modelos variabilidad en XML, más adelante se explicará en detalle los aspectos relativos a la funcionalidad aplicada a un caso de estudio representativo.

3.1 Desarrollo de la herramienta

Este prototipo se ha desarrollado utilizando la tecnología .NET de Microsoft y esta implementado en el lenguaje C#, además trabaja contra una base de datos Microsoft SQL Server 2005 y está vinculado con la herramienta de gestión de requisitos IBM Rational RequisitePro a través de una interfaz implementada en Visual Basic.NET como se describe en la Fig. 2 , pudiéndonos así servir de la popularidad de esta herramienta expandida en el mundo empresarial en las tareas de elicitación y gestión de requisitos en sistemas informáticos, permitiéndonos importar y exportar requisitos desde y hacia ella respectivamente.



Fig. 2. Arquitectura de SREPPLineTool.

3.2 Aplicando SREPPLineTool

En esta sección, describiremos como se puede utilizar SREPPLineTool en la práctica para automatizar el proceso SREPPLine, para ello aplicaremos SREPPLineTool sobre un caso de aplicación representativo en el que desarrollaremos los requisitos de seguridad para una LPS de una Administración Pública española encargada del servicio online de pago de tasas de examen. Esta LPS posee un conjunto de configuraciones diferentes para cada institución pública distinta, así como un conjunto común de funcionalidad que conforma el núcleo común de todos los sistemas, y además cuenta con un conjunto variable de parámetros de configuración y de requisitos no funcionales. Por lo tanto, la LPS de servicios online de pago de tasas de examen es una LPS cuyos productos varían en función de la configuración del sistema y de los servicios de negocio online, pero manteniendo todas las funcionalidades proporcionadas por el núcleo común. Se había realizado un caso de estudio del proceso SREPPLine en [7], pero sin utilizar ningún tipo de herramienta que diese soporte al proceso.

Este ejemplo se concentra en la aplicación de SREPPLineTool sobre la ingeniería del dominio para obtener todos los artefactos de seguridad de la LPS propuestos en el

subproceso PLSecDomReq de SREPPLine. Con el objetivo de facilitar la comprensión del ejemplo se ha resumido y simplificado, destacando los puntos más importantes en cuanto al soporte automatizado que ofrece la herramienta.

Antes de comenzar la ejecución de SREPPLineTool, las características más importantes para nuestra LPS a la que llamaremos para abreviar “SOPE” deben ser identificadas y registradas en la herramienta IBM Rational RequisitePro. A continuación describiremos cada actividad (ordenadas en pestañas) de SREPPLineTool que coincide secuencialmente con cada etapa del proceso SREPPLine.

Actividad 1: Alcance de la Gestión de la Seguridad y Análisis de Variabilidad

Esta primera actividad se divide en varias sub-actividades que se describirán a continuación:

En la primera sub-actividad se realiza la actualización del repositorio, se introducen manualmente los artefactos de seguridad ya definidos en la LPS (que se desarrollaron sin usar SREPPLineTool) y además se recibió una petición de uno de los responsables de la LPS para importar un conjunto de características desde IBM Rational RequisitePro. Entonces, el ingeniero de Requisitos de Seguridad buscó en el Repositorio de Recursos de Seguridad de la herramienta para elicitación y proponer características de seguridad para la LPS, el sugirió relacionar las características de seguridad: Envíos Seguros y Autenticidad de Usuarios como características de seguridad variantes de la LPS SOPE.

La siguiente sub-actividad es la Selección del Personal Responsable, donde el director de desarrollo de la LPS selecciona los usuarios y roles con los que dichos usuarios participaran en el proyecto dentro de la herramienta.

La tercera sub-actividad es la Identificación de la Variabilidad, como se observa en la Fig. 3, SREPPLineTool nos permite representar en un árbol de variabilidad de características el modelo de variabilidad de la LPS previamente diseñado. Además, se observa la estructura de características de seguridad que se definieron en la primera sub-etapa, de esta manera en esta sub-actividad se diseña el modelo de variabilidad de seguridad, el cual se puede exportar a un fichero XML para que sea usado por otras herramientas.

La última sub-actividad es la de Acuerdos de Definiciones donde SREPPLineTool nos da soporte para conseguir acuerdos sobre un conjunto común de definiciones de seguridad como: Seguridad de la Información, Amenazas, Confidencialidad, etc., proporcionando las definiciones de estos conceptos de acuerdo con los estándares ISO/IEC 27002 [20] y ISO/IEC 27001. Además, SREPPLineTool nos permite definir nuevos estándares así como conceptos, pudiéndose así almacenar en el repositorio y también nos posibilita definir el estado de evaluación del nivel de aseguramiento (EAL) de los Criterios Comunes (CC), para nuestra línea SOPE se eligió el nivel EAL-2 de los CC.

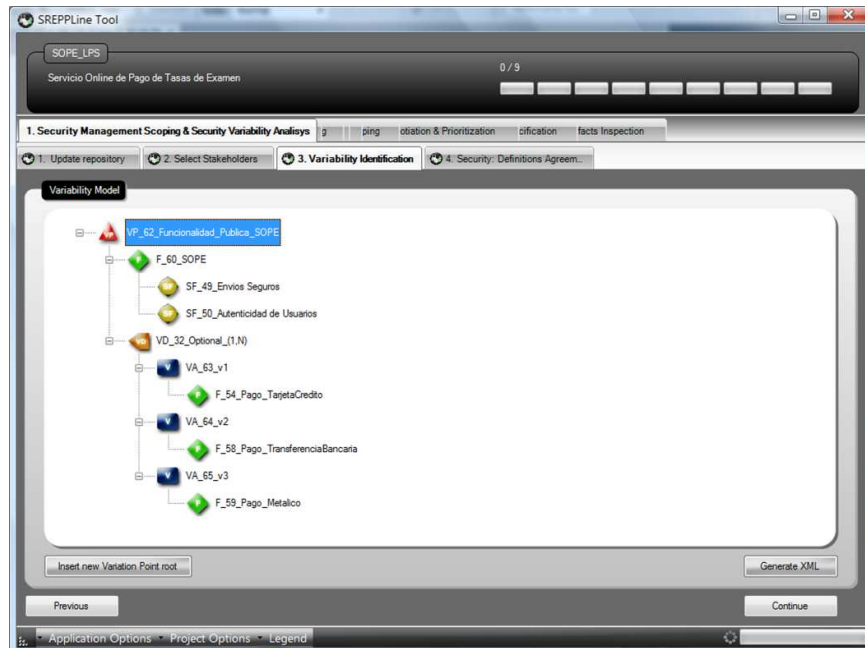


Fig. 3. Identificación de la Variabilidad.

Actividad 2: Alcanse de los Activos de Seguridad

En esta actividad, identificamos los activos de seguridad para cada característica de seguridad y las dependencias entre dichos activos. Por ejemplo, para la característica de seguridad “Autenticidad de Usuarios” identificamos una serie de activos de seguridad, los cuales fueron propuestos por la herramienta al estar previamente clasificados en el repositorio como pertenecientes al dominio de componentes online de la administración, los activos fueron los siguientes: Contraseña de Usuario, Certificado Electrónico, Numero Nacional de identidad/Numero de Cedula. Además, la herramienta nos permite dar un valor numérico de 0 a 10, el cual describe como de importante o crítico es el activo, activos con valores altos representan una mayor importancia y degradación del producto si se llegase a manifestar una amenaza que afectase ha dicho activo.

En nuestro ejemplo de aplicación representativo, la característica de seguridad “Autenticidad de Usuarios” contiene información critica, y por tanto los activos de seguridad que representan la información fueron valorados de la siguiente forma (información de la Contraseña de Usuario, valor = 9; Información del Certificado, valor = 9; Información del Numero Nacional de Identidad, valor = 6). Además, SREPLLineTool nos permite definir jerarquías de dependencias entre activos, de esta forma si un activo depende de otro que tiene una mayor criticidad el valor de dicho

activo se propagará en el otro activo, de esta forma se crea una estructura en árbol en el que el valor de los activos se propaga del activo raíz a las hojas.

Actividad 3: Alcance de los Objetivos de Seguridad.

Seleccionamos los objetivos de seguridad para cada activo; la herramienta nos muestra los objetivos de seguridad y las relaciones actuales entre activos y objetivos de seguridad junto con el valor de cada par (activo, objetivo de seguridad), el cual representa cómo de importante es el cumplimiento de dicho objetivo para proteger ese activo, este valor definido en Magerit [16] varía de 0 (mínimo) a 10 (máximo).

En nuestro caso de estudio, se identificaron los siguientes objetivos de seguridad: integridad, confidencialidad, disponibilidad, autenticidad del origen de los datos, trazabilidad del uso del servicio y trazabilidad del acceso a los datos. Estos objetivos de seguridad se añadieron para los activos: Contraseña de Usuario, Certificado Electrónico y Número Nacional de Identidad.

Actividad 4: Alcance de las Amenazas de Seguridad.

Esta actividad es la responsable de especificar y relacionar los pares de activo-objetivo de seguridad con las posibles amenazas que son susceptibles de crear brechas de seguridad en nuestra LPS. El repositorio de recursos de seguridad de SREPPLineTool nos permite seleccionar las amenazas desde las siguientes fuentes: amenazas previamente introducidas en el repositorio en este proyecto de LPS o en otros proyectos, Objetivos de Control de la ISO/IEC 27001 y Familias de los Criterios Comunes (ISO/IEC 15408). Cuando una nueva amenaza es creada, podemos especificar un conjunto de casos de mal uso y/o árboles de ataque [2] para definir el comportamiento de una amenaza cuando se manifiesta sobre un par activo-objetivo de seguridad.

Sobre el caso de estudio en el que nos centramos, SREPPLineTool nos permitió definir y relacionar las amenazas con los pares activo-objetivo de seguridad de la LPS, de modo que identificamos las siguientes amenazas que podrían manifestarse poniendo en peligro la LPS SOPE:

- Amenaza 1: Manipulación de la configuración.
- Amenaza 2: Enmascaramiento de la identidad del usuario.
- Amenaza 3: Modificación de los datos.
- Amenaza 4: Escuchas.
- Amenaza 5: Acceso no autorizado.

Actividad 5: Evaluación de riesgos de seguridad.

Una vez las amenazas han sido identificadas, se llevó a cabo la evaluación de requisitos (Fig. 4). Para realizar esta tarea, SREPPLineTool usa una técnica propuesta en Magerit [16] (técnicas reconocidas oficialmente por la OTAN [18]) que se basa en

el uso de un análisis cuantitativo. Primeramente, y con la ayuda de los responsables de la LPS, por cada par activo-objetivo de seguridad, se estimó la frecuencia con las que las amenazas podrían manifestarse (siendo los valores posibles: 0,1; 1; 10; 100). Así como se estimó la degradación (expresada como un porcentaje) del valor de un activo causada por una amenaza que se ha manifestado. Finalmente, con estos datos la herramienta calcula automáticamente el impacto y el riesgo para cada tupla activo-objetivo de seguridad-amenaza, en donde los valores altos indican un mayor impacto y riesgo.

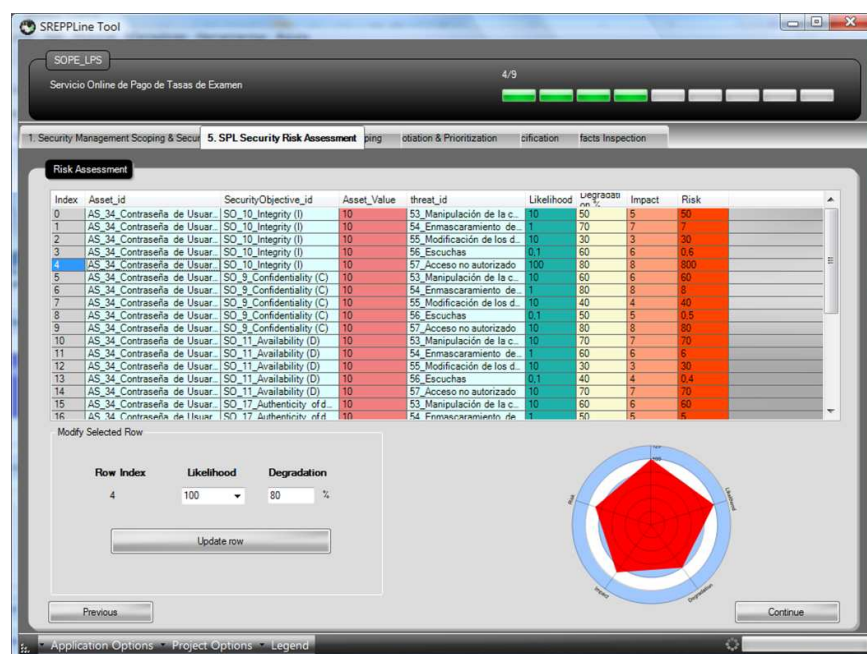


Fig. 4. Actividad 5 SREPLLineTool.

Actividad 6: Alcance de los Requisitos de Seguridad.

En esta actividad se definieron los requisitos o paquetes de requisitos de seguridad para cada tupla de [activo-objetivo]-amenaza en función del riesgo obtenido en la etapa anterior, para así mitigar las posibles amenazas en el caso de que se manifesten. Por lo tanto, una vez fueron seleccionadas las amenazas, se elicitaron aquellos requisitos de seguridad que los responsables de la línea decidieron que eran necesarios. Para facilitar esta tarea SREPLLineTool nos permite:

- Seleccionar los requisitos de seguridad introducidos previamente en el repositorio de la herramienta ya sea en el proyecto de LPS actual o en otros proyectos o creando un requisito nuevo.

- Seleccionar los requisitos de seguridad desde los controles de la ISO/IEC 27001 o desde los componentes de los Criterios Comunes.
- Seleccionar o crear nuevos paquetes o test de seguridad.

Sobre nuestro caso de aplicación de SREPPLineTool se añadieron a nuestra LPS aquellos requisitos de seguridad que nos comunicó el experto en ingeniería de requisitos de seguridad que serian importantes para mitigar las amenazas definidas previamente.

A continuación describimos un requisito de seguridad que seleccionamos del repositorio de recursos de seguridad y que está relacionado con la amenaza 2 “Enmascaramiento de la identidad del usuario”:

SR1: Aseguramiento de la Autenticidad del Usuario.

- Las funciones seguridad de [PV_LPS_dom] deberán identificar y autenticar a los [PV_tipo_usuario] usando [Variante] antes que un [PV_tipo_usuario] pueda conectarse al sistema [PV_LPS_dom]. (Variante = [Contraseña de Usuario, Certificado Electrónico]).

Además, SREPPLineTool nos permite relacionar los requisitos de seguridad con los requisitos funcionales y no funcionales de la LPS. También, facilita la especificación de requisitos de seguridad teniendo la posibilidad de especificar casos de uso de seguridad y árboles de ataque.

Actividad 7: Negociación y Priorización de los Requisitos de Seguridad.

En esta actividad se da prioridad a los requisitos de seguridad conforme al riesgo que poseen las amenazas que pretende mitigar y a las dependencias con otros requisitos funcionales y no funcionales. Para cada uno de los requisitos de seguridad establecidos en un nuestra LPS, seleccionamos el nivel de prioridad (de 0 a 10), y con esa información SREPPLineTool nos ordenará la lista de requisitos de seguridad desde la prioridad más alta a la más baja.

Actividad 8: Especificación de Requisitos de Seguridad.

En este punto se abordó la tarea de realizar la especificación de los requisitos de seguridad. Para llevarla a cabo, SREPPLineTool nos proporciona plantillas XML parametrizadas para definir la variabilidad en los requisitos de seguridad y sus relaciones con el modelo de variabilidad.

Actividad 9: Especificación de Requisitos de Seguridad.

En esta actividad, SREPPLineTool da soporte facilitando la tarea de verificación de que los requisitos de seguridad sean conformes al IEEE 830 [10] y al estándar ISO/IEC 15408 (Criterios Comunes), siendo más sencillo para los responsables de la

LPS el poder verificar y validar los requisitos de seguridad comprobando aquellas amenazas para las que no se han especificados requisitos que las mitiguen en el proyecto actual de LPS.

Por último, en esta actividad, la herramienta genera el documento de perfil de protección para la LPS conforme a los Criterios Comunes (ISO/IEC 15048), que integra toda la información relacionada con el resto de artefactos generados por SREPPLineTool en las actividades anteriores. Finalmente, SREPPLineTool nos permite seleccionar aquellos artefactos de seguridad modificados o generados en esta iteración del proyecto, para que sean añadidos al repositorio general de la herramienta, pudiendo ser así reutilizados en otras iteraciones de la LPS o en otros proyectos de LPS.

3.3 Lecciones Aprendidas

De entre las lecciones más importantes que hemos aprendido utilizando en la práctica SREPPLineTool sobre este caso de estudio, cabe destacar las siguientes:

- El soporte que ofrece la herramienta es fundamental para aplicar el proceso SREPPLine sobre sistemas software de gran envergadura y concretamente sobre el caso de estudio en el que nos hemos centrado, debido al gran número de artefactos que pueden llegar a manejarse y a la complejidad de las relaciones entre dichos artefactos, de tal forma que sin SREPPLineTool hubiera sido complicada la gestión de la trazabilidad y variabilidad de la LPS y hubiera el desarrollo hubiera necesitado de mayor esfuerzo y tiempo.
- Se echó en falta una mayor versatilidad en cuanto a la integración con otras herramientas relacionadas con el paradigma de desarrollo de LPS para obtener un trazabilidad óptima de los artefactos relacionados con los requisitos de seguridad, y para conseguir una implementación apropiada de la ingeniería de requisitos de seguridad dentro de una organización.
- Para la organización en la que se utilizó la herramienta, se obtuvo como mejora el hecho de tener un proceso (SREPPLine) sistemático y específico para gestionar requisitos de seguridad en su LPS que cuenta con un soporte automatizado (SREPPLineTool) que facilita su aplicación, y además siendo conforme con los estándares ISO/IEC 15408 e ISO/IEC 27001, así como se implementó un repositorio de activos comunes que podían ser reutilizados en el desarrollo de las aplicaciones de la línea o en desarrollos futuros de nuevas LPS en la organización.

4 Trabajos Relacionados

En los últimos años se ha llevado a cabo una amplia labor de estudio sobre los requisitos de seguridad, tal y como se presentó en [21, 22], además de algunos trabajos que se ocupan de las herramientas de gestión de requisitos de seguridad, aunque ninguna de las herramientas existentes de ingeniería de requisitos hasta el momento da un soporte suficientemente específico o adaptado al paradigma de

desarrollo basado en LPS, principalmente porque no se ocupan de la variabilidad de los requisitos de seguridad, el cual es un aspecto esencial de este paradigma, y que necesita de una automatización que facilite la gestión de las relaciones de trazabilidad y variabilidad de los numerosos artefactos que se generan.

Actualmente, en cuanto a los requisitos de seguridad se refiere, destacan una serie de herramientas que a continuación vamos a describir brevemente, y que además están relacionadas con SREPPLineTool en lo relativo a la gestión de requisitos de seguridad.

La herramienta SirenTool es un complemento de IBM RequisitePro que da soporte al método SIREN [23], que es un método para elicitar y especificar los requisitos software y de seguridad de un sistema, además incluye un repositorio de requisitos de seguridad basado en los artefactos de Magerit y que puede ser estructurado de acuerdo a dominios y perfiles de una forma parecida a las categorías que implementa SREPPLineTool. No obstante, SirenTool solo reutiliza requisitos basados en los activos de Magerit almacenados en el propio repositorio de la herramienta. Sin embargo, nuestra herramienta trabaja sobre líneas de producto software, que junto con el modelo de referencia de seguridad que implementa nos permite reutilizar requisitos de seguridad, amenazas, características de seguridad, objetivos de seguridad, activos, contramedidas y pruebas. Además, gracias a este modelo, la variabilidad de seguridad puede ser gestionada en el nivel de requisitos en lugar de en el nivel de diseño.

ST-Tool [24] es una herramienta CASE desarrollada para la modelización y análisis funcional, y para el diseño y modelización de requisitos de seguridad. STTool ha sido diseñada para dar soporte a la metodología Secure Tropos [24], y consiste en una herramienta de desarrollo de software orientado a agentes, que maneja el concepto de actor, servicio y relación. En contraste con SREPPLineTool, ST-Tool no trata la reutilización de artefactos de seguridad, tampoco incorpora una integración con los estándares de seguridad (como ISO/IEC 15408 o ISO/IEC 27001) y no facilita la generación de informes.

La herramienta UMLsec-Tool soporta UMLsec [25] y proporciona una extensión al proceso convencional de desarrollo dirigido por casos de uso para sistemas de seguridad críticos. Considera los aspectos de seguridad tanto en el modelo de dominio estático como en la especificación funcional. Para la elaboración de los aspectos funcionales introduce un catálogo de cuestiones y en el modelo del dominio introduce una extensión de UML, UMLsec. Sin embargo, la herramienta no facilita la definición de la variabilidad de la seguridad, que es una diferencia fundamental entre el desarrollo de sistemas tradicionales y el de LPS.

5 Conclusiones y Trabajo Futuro

Actualmente, la seguridad en el software está generando un creciente interés y aún más en el desarrollo orientado a LPS, debido a que la gestión de requisitos de

seguridad es extremadamente importante en las LPS porque una debilidad en la seguridad de una línea puede producir problemas a lo largo de todo el ciclo de vida de los productos de dicha línea.

Aunque se han realizado algunos intentos de cubrir el agujero que separa la ingeniería de requisitos de la ingeniería de requisitos para LPS, actualmente no existe una aproximación sistemática o herramienta que soporte la definición de requisitos de seguridad, que gestione la variabilidad de los mismos, de las relaciones con otros artefactos de seguridad de los modelos de LPS. Además, las herramientas tradicionales de gestión de requisitos no son capaces de dar un soporte directo a los requisitos de seguridad que se manejan en la ingeniería de las LPS.

En este artículo, se ha mostrado como una integración completa de los conceptos de ingeniería de requisitos de seguridad y de ingeniería de LPS, junto con las últimas técnicas de especificación de requisitos de seguridad (como los casos de uso de seguridad [14], casos de mal uso y árboles de ataque [2], UMLsec [25]) y junto con los estándares de seguridad más relevantes que tratan la gestión de requisitos de seguridad (tales como ISO/IEC 15408, ISO/IEC 27001, o ISO/IEC 27002), en estas herramientas es posible.

Teniendo lo anterior en cuenta, podemos decir que las herramientas como SREPPLineTool facilitan que la industria acepte e integre técnicas de ingeniería de requisitos de seguridad dentro del desarrollo de sus LPS, este hecho fue ratificado en el caso de estudio real que realizamos en la Seguridad Social de España [7].

Por último, hay una serie de aspectos previstos para el futuro de la herramienta que nos permitirán aumentar el nivel de automatización del proceso SREPPLine y por lo tanto, una mayor eficiencia del proceso de ingeniería de requisitos dentro de la ingeniería de las LPS de una organización. Entre ellos, podemos destacar los siguientes: ampliar los tipos de especificaciones de requisitos soportadas para así dar soporte a UMLSec [25]; desarrollar nuevos módulos de extensibilidad que permitan que SREPPLineTool sea soportada por otras herramientas CARE; y automatizar la creación de casos de uso de seguridad a través de la creación de casos de mal uso.

6 Referencias

- [1] CERT/CC, "CERT/CC Statistics 1995-2007," Pittsburgh, 2008.
- [2] A. L. Opdahl and G. Sindre, "Experimental comparison of attack trees and misuse cases for security threat identification," *Information and Software Technology. In Press, Corrected Proof*, 2008.
- [3] K.-K. R. Choo, R. G. Smith, and R. McCusker, "Future directions in technology-enabled crime: 2007–09," in *Research and Public Policy Series*. vol. 78, Australian Government, Ed.: Australian Institute of Criminology, 2007.
- [4] A. Birk and G. Heller, "Challenges for requirements engineering and management in software product line development," *International Conference on Requirements Engineering (REFSQ 2007)*, pp. 300-305, 2007.

- [5] J. Viega and G. McGraw, *Building Secure Software: How to Avoid Security Problems the Right Way*. Boston: Addison-Wesley, 2002.
- [6] A. v. Lamsweerde, "Elaborating Security Requirements by Construction of Intentional Anti-Models," *26th International Conference on Software Engineering (ICSE 2004)*, pp. 148-157, 2004.
- [7] D. Mellado, E. Fernández-Medina, and M. Piattini, "Towards security requirements management for software product lines: a security domain requirements engineering process," in *Computer Standards & Interfaces*. vol. 30, 2008, pp. 361-371.
- [8] ISO/IEC, "ISO/IEC 15408:2005 Information technology - Security techniques - Evaluation criteria for IT security, (Common Criteria v3.0)," 2005.
- [9] ISO/IEC, "ISO/IEC 27001 Information technology -- Security techniques -- Information security management systems -- Requirements.," 2005.
- [10] IEEE, "IEEE 830: 1998 Recommended Practice for Software Requirements Specifications," 1998.
- [11] P. Clements and L. Northrop, *Software Product Lines: Practices and Patterns*. Addison-Wesley, 2002.
- [12] K. Pohl, G. Böckle, and F. v. d. Linden, *Software Product Line Engineering. Foundations, Principles and Techniques*. Berlin Heidelberg: Springer, 2005.
- [13] OMG, "Software & Systems Process Engineering Meta-Model Specification v.2.0," 2008.
- [14] D. G. Firesmith, "Engineering Security Requirements," *Journal of Object Technology*, vol. 2, pp. 53-68, 2003.
- [15] ISO/IEC, "ISO/IEC 13335 Information technology - Security techniques - Management of information and communications technology security," 2004.
- [16] Ministry_for_Public_Administration_of_Spain, *Methodology for Information Systems Risk Analysis and Management*: Ministry for Public Administration, 2005.
- [17] CCN-CERT, "Últimos avances en ciberseguridad (9th NATO cyberdefense workshop)," in *Revista auditoria y Seguridad* (www.revista-ays.com). vol. nº23-junio, 2008, pp. 70-71.
- [18] OECD, "The promotion of a culture of security for information systems and networks in OECD countries," Organisation for Economic Co-operation and Development 2005.
- [19] D. Mellado, E. Fernández-Medina, and M. Piattini, "Security Requirements Variability for Software Product Lines," *Symposium on Requirements Engineering for Information Security (SREIS 2008) co-located with ARES 2008*, pp. 1413-1420, 2008.
- [20] ISO/IEC, "ISO/IEC 27002:2005 Information technology - Security techniques - Code of practice for information security management," 2007.
- [21] D. Mellado, E. Fernández-Medina, and M. Piattini, "A Comparison of the Common Criteria with Proposals of Information Systems Security Requirements," *"First International Conference on Availability, Reliability and Security" (ARES'06)*, pp. 654-661, 2006.
- [22] D. Mellado, E. Fernández-Medina, and M. Piattini, "A Systematic Review of Security Requirements Engineering," in *Computers and Security (Being processed - Under review)*, 2008.
- [23] A. Toval, J. Nicolás, B. Moros, and F. García, "Requirements Reuse for Improving Information Systems Security: A Practitioner's Approach," in *Requirements Engineering*. vol. 6, 2002, pp. 205-219.
- [24] P. Giorgini, F. Massacci, J. Mylopoulos, and N. Zannone, "ST-Tool: A CASE Tool for Security Requirements Engineering," in *IEEE International Conference on Requirements Engineering (RE'05)*, 2005.
- [25] J. Jürjens, "UMLsec: extending UML for secure systems development.," *UML 2002 - The Unified Modeling Language. Model Engineering, Languages, Concepts, and Tools. 5th International Conference.*, vol. LNCS 2460, pp. 412-425, 2002.

SLSB: Improving the Steganographic Algorithm LSB

Juan José Roque, Jesús María Minguet
Universidad Nacional de Educación a Distancia (Spain)
juanjose.roque@extremadura.es; jminguet@issi.uned.es

Abstract. This paper presents a novel steganographic algorithm based on the spatial domain: Selected Least Significant Bits (SLSB). It works with the least significant bits of one of the pixel color components in the image and changes them according to the message's bits to hide. The rest of bits in the pixel color component selected are also changed in order get the nearest color to the original one in the scale of colors. This new method has been compared with others that work in the spatial domain and the great difference is the fact that the LSBs bits of every pixel color component are not used to embed the message, just those from pixel color component selected.

Keywords: Security, Steganography, Least Significant Bit.

1 Introduction

The steganography can be considered as a branch of cryptography that tries to hide messages within others, avoiding the perception that there is some kind of message. To apply steganographic techniques cover files of any kind can be used, although archives of image, sound or video files are the most used today. Similarly, information to hide can be anything: text, image, video, sound, etc.

There are two trends at the time to implement steganographic algorithms: the methods that work in the spatial domain (altering the desired characteristics on the file itself) and the methods that work in the transform domain (performing a series of changes to the cover image before hiding information. To select the best areas the Discrete Cosine Transform DCT, Wavelet Transform, etc. are used).

While the algorithms that work in the transform domain are more robust, that is, more resistant to attacks, the algorithms that work in the spatial domain are simpler and faster.

The best known steganographic method that works in the spatial domain is the LSB [1] (Least Significant Bit), which replaces the least significant bits of pixels selected to hide the information. This method has several implementation versions that improve the algorithm in certain aspects [2][3][4][5][6][7].

This paper proposes a new method, SLSB (Selected Least Significant Bit), that improves the performance of the method LSB hiding information in only one of the three colors at each pixel of the cover image. To select the color it uses a Sample Pairs analysis, given that this analysis is more effective to detect hidden information. Finally, applies a LSB Match [8] method so that the final color is as close as possible to the original one in the scale of colors.

The paper is organized as follows. Section 2 gives a brief classification of the steganographic methods that works in spatial domain. Section 3 describes the

proposed method. Section 4 is on the experimental results, followed by conclusions at Section 5.

2 Methods in Spatial Domain

A basic classification of steganographic algorithms operating in the spatial domain as the method for selecting the pixels distinguishes three main types: non-filtering algorithms, randomized algorithms and filtering algorithms.

2.1 Non-filtering Algorithm

This is the simplest steganographic method based in the use of LSB, and therefore the most vulnerable. The embedding process consists of the sequential substitution of each least significant bit of the image pixel for each bit of the message. For its simplicity, this method can camouflage a great volume of information [9].

This technique is quite simple. It is necessary only a sequential LSB reading, starting from the first image pixel, to extract the secret message. This method also generates an unbalanced distribution of the changed pixels, because the message is embedded at the first pixels of the image, leaving unchanged the remaining pixels.

2.2 Randomized Algorithm

This technique was born as a solution for the problems of the previous method. Each one, the sender and the receiver of the image has a password denominated stego-key that is employed as the seed for a pseudo-random number generator. This creates a sequence which is used as the index to have access to the image pixel. The message bit is embedded in the pixel of the cover image as the index given by the pseudo-random number generator [9].

The two main features of the pseudo-random permutation methods are the use of password to have access to the message, and the well-spread message bits over the image.

2.3 Filtering Algorithm

This algorithm filters the cover image by using a default filter and hides information in those areas that get a better rate. The filter is applied to the most significant bits of every pixel, leaving the less significant to hide information. The filter ensures the choice of areas of the image in the least impact with the inclusion of information, which affects a greater difficulty of detecting the presence of hidden messages [10]. The retrieval of information is ensured because the bits used for filtering are not changed, implying that the reapply the filter will select the same bits in the process of concealment. It is the most efficient method to hide information.

The algorithm SLSB belongs to this type.

3 Description of the Algorithm SLSB

Figure 1 shows the structure of the algorithm SLSB:

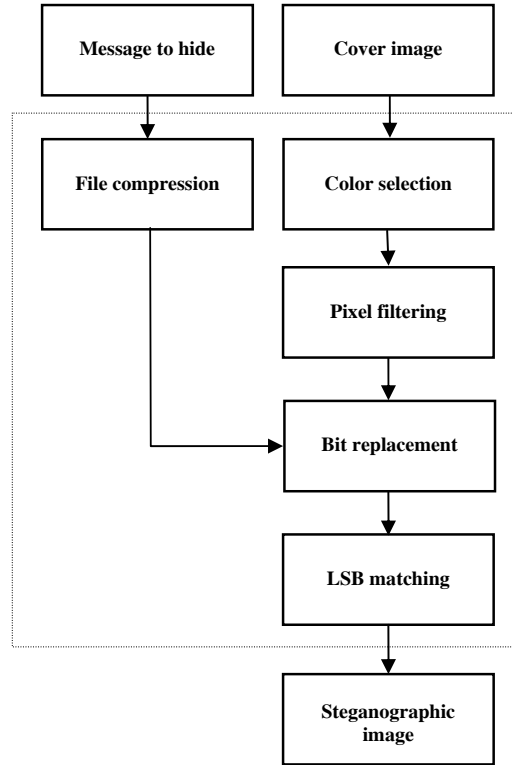


Fig. 1. Structure of the algorithm SLSB.

3.1 Hiding Information in Only One Color

Most of the algorithms that work in the spatial domain using a LSB method (or any of its derivatives) as the algorithm for information hiding, that is, hide one bit of information in the least significant bit of each color of a pixel.

But these methods can't stand a type of statistical analysis (such as RS [11] or Sample Pairs [12]), even if partly camouflaged in the amount of information hidden. The problem stems from the fact that modifying the three colors of a pixel produces a major distortion in the resulting color. This distortion is not visible to the human eye, but detectable by statistical analysis.

For example, if a pixel of the cover image with the RGB (Red-Green-Blue code) color A8A8A8 # is used, binary 10101000-10101000-10101000, and 1 bit with value 1 is set on each LSB bit of each color component, to hide the message 111, the result would be 10101001-10101001-10101001:

Table 1. Results obtained hiding the message 111 in the pixel 10101000-10101000-10101000 with the LSB method.

	Hexadecimal	Decimal	Red	Green	Blue
Original pixel	A8A8A8	11053224	168	168	168
Modified pixel	A9A9A9	11119017	169	169	169

In theory the three least significant bits of the pixel have changed, introducing a small distortion, but the difference between the old and new color represents a leap of 65793 colors in the scale of colors.

One method that would introduce more efficiency and less distortion would store the 3 bits of information to hide in the same color. Using the same example, the 3 bits of information will be introduced in the 3 LSB bits of green color (10101000-10101111-10101000):

Table 2. Results obtained hiding the message 111 in the pixel 10101000-10101000-10101000 with the SLSB method.

	Hexadecimal	Decimal	Red	Green	Blue
Original pixel	A8A8A8	11053224	168	168	168
Modified pixel	A8AFA8	11055016	168	175	168

In this case the leap in the scale of colors is 1792 colors (in the case of changing the color green, if modify the blue color difference would be only 7 colors), that being the extreme case because it has been replaced last 3 bits with 0 value for 3 bits with a 1 value, that is, in most cases the distortion will be much lower.

In order to choose the color for the concealment, the SLSB algorithm performs a preliminary Sample Pairs analysis and select the color with higher ratio because it represents more diversity, leading to less noticeable changes. The choice of Sample Pairs analysis over other stegoanalytics methods is due to the results provided by the work of Ker [13], where this analysis shows that it is offering better results in terms of detecting hidden information. Thus, the chosen color will be the one that provides greater distortion and, therefore, the result of the withholding of information will be less detectable.

3.2 LSB Match Adaptation

Following the work of Van Dijk [14] and Goljan [15], the method LSB Match (designed to work with a single LSB bit) has been adapted to allow an LSB Match with any number of LSB bits.

This method calculates the distance between the original color and the steganographic color. Should the distance is greater than a certain threshold (determined by the number of bits to hide) the color is decremented to get a final color closest to the original, implying a further reduction in the distortion caused by the hidden information.

For example, using a cover byte 11001000 to hide 3 bit of information (111), with a simple LSB results in 11001**111**, which has a difference of 7 values with respect to the original.

Applying the method proposed here to the above example (in this case, decreasing the 4th least significant bit, which have been used 3 bits LSB to hide information) results in 1100**0**111, with a distance of 1 from the original byte but with the same hidden information.

4 Results

To be able to compare the performance of this improvement on the LSB method, the image on Fig. 1 will be used as cover with BMP (Bit Mapped Picture) format and 512x512 pixels in size (24 bits/pixel).



Fig. 2. Cover image.

4.1 Histogram Analysis

The purpose of the histogram analysis is to detect significant changes in frequency of appearance of the colors by comparing the cover image with the steganographic image.

To better align this analysis it has been carried out a detailed examination of the 4 components of any image: brightness and red, green and blue colors.

Histograms in Fig. 3 shows a frequency histogram of the image on Fig. 2 for the four components mentioned above, and his comparison with the results of the image on Fig. 2 with a hidden message of 141.744 bits, using a hiding method of 1 bit/pixel, producing a hiding rate of 54%.

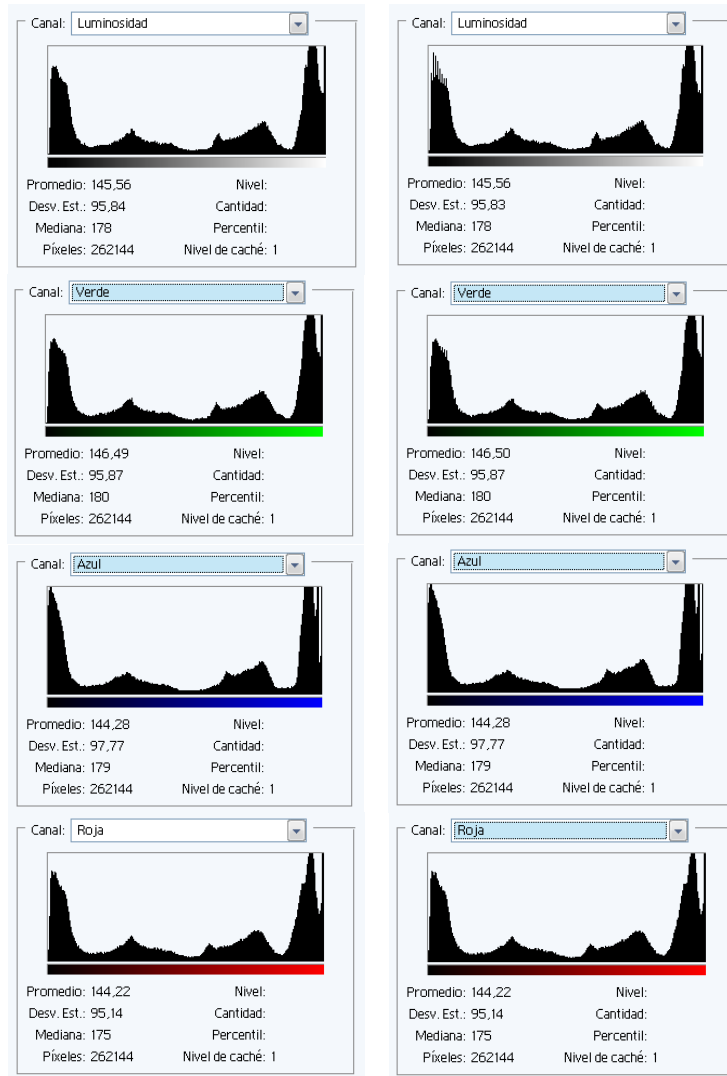


Fig. 3. Histograms of brightness, green, blue and red colors in the image on Fig. 2 (left) and the comparison with the results of the image on Fig. 2 with a hidden message of 141.744 bits, using a hiding method of 1 bit/pixel, producing a hiding rate of 54% (right).

There are only changes in the histograms of brightness and green color (the one chosen by the algorithm as the optimal color of concealment).

Despite having a hiding rate of 54%, the changes are negligible (0.01 in the standard deviation of brightness and an average of 0.01 in the green color).

According to the results, it can be said that the new proposed algorithm is immune

to attacks based on a comparison of histograms of the original image and the steganographic image.

4.2 Another Steganographic Tools Comparison

To conclude the analysis of the results of the new proposed algorithm its performance is compared with that from the best known and more used today steganographic tools.

This comparison focuses on two aspects: the results of the RS and Sample Pairs analysis of steganographic images and the analysis of the results of the metrics of distortion.

Table 3 shows a comparison of the results for the steganographic images obtained with the various tools in front of the RS analysis and the Sample Pairs analysis. The results of these analyses are an estimate of the percentage of hidden information. A lower ratio means a higher quality of the hiding method.

Table 3. Results obtained using a cover image of 786.486 bytes (Fig. 2) and a hidden message of 31.071 bytes (TXT file).

Tool	RS analysis	Sample Pairs analysis
Hermetic Stego [20]	75,46911	73,39835
Invisible Secrets [23]	70,06539	69,32617
Hide4PGP [21]	30,60135	30,19531
Contraband [16]	14,78796	11,83324
wbStego [27]	14,33760	13,42652
White Noise Storm [28]	11,91106	10,12546
Digital Invisible Ink Toolkit [18]	9,61806	7,84342
JPHS [24]	2,62679	2,68511
S-tools [26]	2,30629	2,10435
EikonaMark [19]	1,86631	1,31909
Data Privacy Tools [17]	1,43103	0,96443
Hide In Picture [22]	1,03530	1,06373
SLSB algorithm (1 bit/pixel)	0,89172	0,61744
SLSB algorithm (2 bits/pixel)	0,80084	0,60556
Original image	0,67766	0,51907
SLSB algorithm (3 bits/pixel)	0,64431	0,47867
Steghide [25]	0,63543	0,37671

The results show that the new algorithm, in its 3 versions, offers among the best ever results (even reach a ratio below the original image, thereby preventing distinction between the original and steganographic image).

Table 4 shows a comparison of the results of the metrics of distortion [9] (Average Absolute Difference, Mean Squared Error, Lp-Norm, Laplacian Mean Squared Error, Signal to Noise Ratio, Peak Signal to Noise Ratio, Normalised Cross-Correlation and Correlation Quality) applied to steganographic images obtained by different tools. Obviously, a lower distortion represents a better steganographic method because it is closer to the values of the original image.

Table 4. Results obtained using a cover image of 786.486 bytes (Fig. 2) and a hidden message of 31.071 bytes (TXT file).

Tool	AAD	MSE	LP	LMSE	SNR	PSNR	NC C	CQ
Original image	0,0	0,000	0,0	0,000	0,0	0,0	1,000	1,26643
SLSB algorithm (1 bit/pixel)	5,3	0,020	36,9	1,113	6,7	2,0	0,999	1,26643
SLSB algorithm (2 bits/pixel)	6,1	0,043	53,2	2,279	3,2	9,9	1,000	1,26643
SLSB algorithm (3 bits/pixel)	9,1	0,137	94,9	7,288	1,0	3,1	1,000	1,26643
Contraband	10368,4	0,872	26038,6	0,002	136124,5	415146,6	0,999	1,26597
Data Privacy Tools	25155,1	13,760	110709,9	0,045	7530,1	22965,0	1,001	1,26798
Digital Invisible Ink Toolkit	10377,1	0,947	26053,1	0,002	135973,3	414685,4	1,000	1,26662
EikonaMark	150528,2	90,996	232052,0	0,279	1713,9	5227,1	0,997	1,26307
Hermetic Stego	42354,0	4,500	52662,9	0,009	33278,5	101491,3	1,000	1,26643
Hide4PGP	10523,7	0,477	26211,1	0,002	134339,1	409701,5	1,000	1,26643
Hide In Picture	10493,4	0,665	26194,6	0,002	134508,4	410218,1	1,000	1,26747
Invisible Secrets	32934,0	3,007	46420,3	0,007	42830,9	130623,8	1,000	1,26649
JPHS	54609,9	9,141	89871,2	0,011	11427,0	34849,6	1,000	1,26643
Steghide	1465,3	0,320	16270,8	0,001	348621,5	1063210,6	0,999	1,26643
S-tools	552,4	0,025	6005,0	1,202	2559393,7	7805527,2	1,000	1,26643
wbStego	10408,3	0,947	26092,5	0,002	135563,4	413435,6	1,000	1,26668
White Noise Storm	13828,4	1,101	30071,6	0,003	102060,9	311260,8	0,999	1,26640

This table can verify that the new algorithm (in any of its three versions) offers the best results in the metrics AAD, MSE, LP, SNR, PSNR, NCC and CQ, and provide the same result as the original image in the last two columns.

5. Conclusions

This paper proposes a new method, SLSB (Selected Least Significant Bit), that improves the performance of the LSB method hiding information in only one of the three colors at each pixel of the cover image. For the selection of color it uses a Sample Pairs analysis, given that this analysis is more effective to detect hidden information. Finally, applies a LSB Match [8] method so that the final color is as close as possible to the original one.

A summary of its features could be:

- It is based on the LSB method, but can hide the same information much more effectively using bits of just one color.
- Perform a Sample Pairs analysis prior to steganography, which allows you to select the best color of the three possible to hide information.
- Use a pixel selection filter to obtain the best areas to hide information.
- Implement the LSB Match method to reduce the difference between the original pixel and the steganographic pixel.
- It is immune to visual attacks. Changes are undetectable with the naked eye, and a filter of LSB bits doesn't present areas of random information that could indicate the presence of hidden information.
- It is immune to attacks by comparing histograms, as the frequency of appearance of colors in the steganographic image is very similar to that of the cover image.
- It is immune to statistical attacks, as two colors for each pixel are equal to those of the original image, and the final ratio of analysis is very close to the original image, which doesn't raise suspicion it contains hidden information. Even in some cases get better rates than those of the original image, creating confusion over which of two images would be the original.
- It yields well above that of most steganographic tools used today, both in RS and Sample Pairs analysis and in metric of distortion.

Future works will aim to achieve better performance and be undetectable by the most famous steganographic analysis, for example, changing bits undisturbed by the concealment of the message.

References

- 1 Kurak, C. and McHugh, J.: A Cautionary Note on Image Downgrading. Proc. IEEE 8th Annual Computer Security Applications Conference. San Antonio, USA, Nov./Dec. 1992, pp. 153-155.
- 2 Moskowitz, I., Longdon G. and Chang, L.: A New Paradigm Hidden in Steganography. Proc. 2000 Workshop on new security paradigms, Ballycotton, Country Cork, Ireland, 2000. ACM Press, New York, pp. 41-50.
- 3 Sharp, T.: An implementation of key-based digital signal steganography. Proc. 4th International Workshop on Information Hiding, Pittsburgh, USA, April 25, 2001. Springer LNCS, vol. 2137, pp. 13-26.

- 4 Kawaguchi, E. and Eason, R.: Principle and applications of BPCS-Steganography. Proc. Multimedia Systems and Applications Conference, Boston, MA, USA, November 2, 1998. SPIE series, vol. 3528, pp. 464-473.
- 5 Bender, W., Gruhl, D., Morimoto, N. and Lu, A.: Techniques for data hiding. IBM Systems Journal, vol. 35, nos. 3&4, 1996.
- 6 Moskowitz, I., Johnson, N. and Jacobs, M.: A detection study of an NRL steganographic method. NRL Memorandum Report NRL/MR/5540{02-8635, Naval Research Laboratory, Code 5540, Washington, 2002.
- 7 Noto, M.: MP3Stego: Hiding text in MP3 files. Sans Institute, 2003.
- 8 Sharp, T.: An implementation of key-based digital signal steganography. Proc. 4th International Workshop on Information Hiding. Springer LNCS, vol. 2137, pp.13-26, 2001.
- 9 Katzenbeisser, S. and Petitcolas, F.: Information hiding techniques for steganography and digital watermarking. Artech House Books, 1999.
- 10 Hempstalk, K.: Hiding behind corners: using edges in images for better steganography. Computing Womens Congress Conference, Hamilton, New Zealand, 2006.
- 11 Fridrich, J., Goljan, M. and Du, R.: Reliable detection of LSB steganography in color and grayscale images. Proc. ACM Workshop on Multimedia and Security, Ottawa, ON, Canada, Oct. 5, 2001, pp. 27-30.
- 12 Dumitrescu, S., Wu, X. and Wang, Z.: Detection of LSB steganography via sample pairs analysis. 5th International Workshop on Information Hiding. Noordwijkerhout, Pays-Bas, 7/10/2002. Springer LNCS, vol. 2578, pp. 355-372, 2003.
- 13 Ker, A.: Improved detection of LSB steganography in grayscale images. Proc. 6th Information Hiding Workshop. Springer LNCS, vol. 3200, pp. 97-115, 2004.
- 14 Van Dijk, M. and Willems, F.: Embedding information in grayscale images. Proc. 22nd Symposium on Information and Communication Theory in the Benelux, pp. 147-154, Enschede, The Netherlands, May 15-16, 2001.
- 15 Goljan, M. and Holtyak, T.: New blind steganalysis and its implications. Proc. SPIE Security, Steganography, and Watermarking of Multimedia Contents VIII, vol. 6072, pp. 1-13, 2006.
- 16 Thyssen, J. & Zimmerman, H., Contraband 9g, 1999-01-01 (consultation date 2009-07-07). Available in <http://www.jthz.com/puter/>.
- 17 Bernard Electronics, Data Privacy Tools 3.5, last update 2008-10-03 (consultation date 2009-07-07). Available in http://www.xs4all.nl/~bernard/home_e.html
- 18 Hempstalk, K., Digital Invisible Ink Toolkit 1.5, last update 2006-06-09 (consultation date 2009-07-07). Available in <http://diit.sourceforge.net>
- 19 Alpha Tec Ltd, EikonaMark 4.7, last update 2005-01-01 (consultation date 2009-07-07). Available in <http://www.alphatecltd.com/watermarking/eikonamark/eikonamark.html>

- 20 Hermetic Systems, Hermetic Stego 7.41t, last update 2008-08-09 (consultation date 2009-07-07). Available in <http://www.hermetic.ch/hst/hst.htm>
- 21 Repp, H., Hide4PGP 2.0, last update 200-02-01 (consultation date 2009-07-07). Available in <http://www.heinz-repp.onlinehome.de/Hide4PGP.htm>
- 22 Figueiredo, D., Hide In Picture 2.1, last update 2002-10-01 (consultation date 2009-07-07). Available in <http://sourceforge.net/projects/hide-in-picture>
- 23 Neobyte Solutions, Invisible Secrets 4.6.3, last update 2007-04-01 (consultation date 2009-07-07). Available in <http://www.invisiblesecrets.com>
- 24 Latham, A., JPHS 0.5, last update 1999-08-01 (consultation date 2009-07-07). Available in <http://linux01.gwdg.de/~alatham/stego.html>
- 25 Hetzl, S., StegHide 0.5.1, last update 2003-10-15 (consultation date 2009-07-07). Available in <http://steghide.sourceforge.net>
- 26 Brown, A., S-tools 4, last update 200-11-18 (consultation date 2009-07-07). Available in <http://www.spychecker.com/program/stools.html>
- 27 Bailer, W., wbStego 4.3, last update 2004-03-01 (consultation date 2009-07-07). Available in <http://www.8ung.at/wbailer/wbstego/>
- 28 Arachelian, R., White Noise Storm 2.10, last update 1995-02-01 (consultation date 2009-07-07). Available in <http://www.nic.funet.fi/pub/crypt/steganography/>

Hacia una Arquitectura de Servicios de Seguridad para entornos Grid Móviles

David G. Rosado¹, Eduardo Fernández-Medina¹ y Javier López²

¹ Universidad de Castilla-La Mancha. Grupo Alarcos – Instituto de Tecnologías y Sistemas de Información. Dep. de Tecnologías y Sistemas de Información – Escuela Superior de Informática, Paseo de la Universidad 4, 13071 Ciudad Real
{David.GRosado, Eduardo.FdezMedina}@uclm.es

² Departamento de Lenguajes y Ciencias de la Computación. Universidad de Málaga, jlm@lcc.uma.es

Resumen. Grid móvil incluye las características de los sistemas Grid junto con las peculiaridades de la computación móvil, añadiendo la propiedad de soportar usuarios y recursos móviles de forma homogénea, transparente, segura y eficiente. La seguridad de estos sistemas, debido a su naturaleza abierta y distribuida, es un tema de gran interés. Una arquitectura de seguridad basada en SOA proporciona una arquitectura distribuida diseñada para interoperabilidad de servicios, fácil integración, y acceso seguro, simple y extensible. Por tanto, una arquitectura orientada a servicios de seguridad es construida para entornos Grid móviles, ofreciendo servicios de seguridad a usuarios móviles quienes usan servicios Grid y recursos para ejecutar sus trabajos y tareas. Esta arquitectura es integrada con otras arquitecturas existentes proporcionando mayor seguridad y permitiendo que los usuarios móviles puedan acceder a servicios Grid existentes ofreciendo nuevos y necesarios servicios de seguridad para Grid móviles. Hemos definido un conjunto de servicios de seguridad, que junto a protocolos, políticas y estándares de seguridad forman una arquitectura de seguridad orientada a servicios para entornos Grid móviles. Esta arquitectura es abierta, escalable, dinámica, interoperable y flexible.

Palabras claves: Arquitectura Seguridad, Servicios de Seguridad, computación Grid, computación Móvil, Seguridad.

1 Introducción

La idea del Grid está enfocada fundamentalmente en el acceso remoto a recursos computacionales, solventando el problema de coordinar los recursos compartidos entre las organizaciones virtuales multi-institucionales y dinámicas [1]. Grid móvil, unión de Grid y computación móvil, es un completo heredero del Grid con la característica de dar soporte a los usuarios y recursos móviles de forma transparente, segura y eficiente [2-4]. Grids y Grids móvil pueden ser la solución ideal para muchas aplicaciones a gran escala que son de naturaleza dinámica y que requieren transparencia para los usuarios.

Hay dos posibles roles a la hora de incorporar los dispositivos móviles al grid. Primero, los dispositivos móviles pueden ser usados como interfaces para el grid. Así, un dispositivo móvil puede iniciar el uso de recursos grid, monitorear los trabajos que son ejecutados remotamente, y tomar cualquier resultado desde el grid. Segundo y más interesante, los dispositivos móviles pueden asumir participar en un grid como proveedores de recursos de computación, no sólo como receptores de servicios. Nosotros creemos que el reciente avance de tecnología en dispositivos móviles y comunicaciones wireless hacen este escenario más factible.

La seguridad es un aspecto central en la computación Grid desde el principio, y ha sido considerado como el cambio más significativo de la computación Grid [5, 6]. La seguridad en entornos móviles es más crítica debido a la naturaleza abierta de las redes wireless. Además, la seguridad es más difícil de implementar dentro de una plataforma móvil debido a las limitaciones de recursos de los dispositivos móviles [7].

Debido a la dificultad de incorporar dispositivos móviles dentro de un entorno grid [3, 4, 8, 9], y sumado a la aparición de una nueva tecnología donde la seguridad es fundamental y los avances que la computación móvil ha experimentado en los últimos años, aparece la necesidad de definir, considerar y desarrollar una metodología o proceso de desarrollo en el cual, dentro de todo el ciclo de vida software [10, 11], se analizan e integran todos los requisitos y aspectos de seguridad relacionados con los sistemas Grid móviles, obteniendo como resultado un sistemas Grid móvil seguro, robusto y escalable.

En este artículo, se presenta una visión general sobre la arquitectura de servicios de seguridad que sirva de referencia para cualquier sistema Grid móvil. Esta arquitectura debe ofrecer servicios que soporten y cumplan con todos los requisitos de seguridad identificados y analizados en la actividad de análisis de nuestra metodología, y debe dar soporte a todos los algoritmos, mecanismos, y tecnologías de la actual arquitectura grid.

El resto del artículo está organizado como sigue: en la sección 2 describimos el trabajo relacionado sobre las diferentes arquitecturas de seguridad; en la sección 3 explicamos brevemente nuestra metodología de desarrollo para sistemas Grid móviles seguros; en la sección 4 presentamos la actividad de diseño de nuestra metodología donde definimos la arquitectura de servicios de seguridad para los sistemas Grid considerando la incorporación de dispositivos móviles, e identificamos los servicios de seguridad que forman parte de la arquitectura; finalmente proponemos las conclusiones y el trabajo futuro.

2 Trabajo Relacionado

La actual arquitectura Grid no tienen en cuenta los entornos de computación móvil debido a que los dispositivos móviles no han sido considerados como recursos de computación válidos o interfaces en la comunidad Grid. Es actualmente cuando se está prestando más atención de integrar estas dos tecnologías emergentes, computación Grid y computación móvil, como muestran algunos trabajos en [12, 13], aunque ninguno elabora la forma de incorporar los dispositivos móviles en la actual arquitectura Grid.

El proyecto Legion, desarrollado en la Universidad de Virginia, es un intento de proporcionar servicios Grid creando la ilusión de una máquina virtual. Esta máquina virtual dirige cuestiones Grid claves tales como la escalabilidad, la facilidad de programación, tolerancia a fallos, la seguridad y la autonomía del sitio. En general, el objetivo fundamental para la arquitectura de seguridad Legion [14-16] es permitir que los participantes en un sistema Grid expongan sus recursos de forma complaciente con sus políticas locales. Sin embargo Legion no se ocupa de la replicación de objetos dinámicos, e introduce un mayor nivel en el diseño de la seguridad, acentuando la flexibilidad y extensibilidad, pero menos la arquitectura y protocolos.

Globe (Global Object Based Environment) es un sistema distribuido de área amplia, el cual fue desarrollado para constituir un nivel intermedio entre el sistema operativo y el nivel de aplicación, tal y como hace Legion. Las principales características del sistema Globe son [17]: es un modelo uniforme para soportar sistemas distribuidos, soporta un marco de implementación flexible, y es altamente escalable. La arquitectura de Seguridad de Globe se basa en criptografía de clave pública y certificados digitales con el fin de abordar las cuestiones antes mencionadas.

CRISIS [18] es la arquitectura de seguridad para WebOS. WebOS proporciona un simple y tradicional sistema de ficheros y una interfaz para la creación de procesos remotos autenticados. CRISIS define cuidadosa y eficazmente las políticas de seguridad para estos servicios básicos. Sin embargo, la solución CRISIS no proporciona un medio fácil para el desarrollo de las políticas de seguridad para los nuevos mecanismos que se añaden a WebOS, ni provee un medio para modificar las políticas de seguridad soportadas para los servicios existentes.

Ninguna de las arquitecturas descritas anteriormente tiene en cuenta dispositivos móviles como recursos del Grid, y no dan soporte de seguridad para que los dispositivos móviles puedan ser integrados y manejados por el propio Grid como un recurso más. Nuestra arquitectura de seguridad es construida para un entorno móvil y orientada a servicios, y muchas de las desventajas o características no contempladas en las anteriores arquitecturas serán corregidas en nuestra arquitectura para dar soporte a los dispositivos móviles y asegurar dichos entornos Grid Móviles.

3 Resumen de la metodología

Nuestro objetivo es proporcionar a los desarrolladores, primero, una metodología o proceso sistemático de desarrollo que incluirá el desarrollo completo de sistemas Grid móviles de cualquier complejidad y magnitud, y segundo, una arquitectura de seguridad que ayude a desarrollar un sistema Grid móvil seguro de forma sistemática y ordenada. El proceso sistemático de desarrollo es un proceso iterativo, incremental y reutilizable. Un enfoque iterativo propone una comprensión incremental del problema a través de refinamientos sucesivos, un crecimiento incremental de una solución efectiva a través de varias versiones, y técnica reutilizable para construir componentes desde elementos existentes y probados.

La estructura de la metodología sigue el ciclo clásico, donde tenemos una etapa de planificación, una de desarrollo que incluye análisis, diseño y construcción, y finalmente, de una etapa de mantenimiento, sin embargo, está especialmente diseñada

para este tipo de sistemas, con características tan particulares. Detalle sobre las actividades y tareas de la metodología pueden encontrarse en [19-21]. En la Fig. 1 podemos ver la estructura de la metodología usando SPEM (Software & Systems Process Engineering Metamodel) versión 2.0 [22].

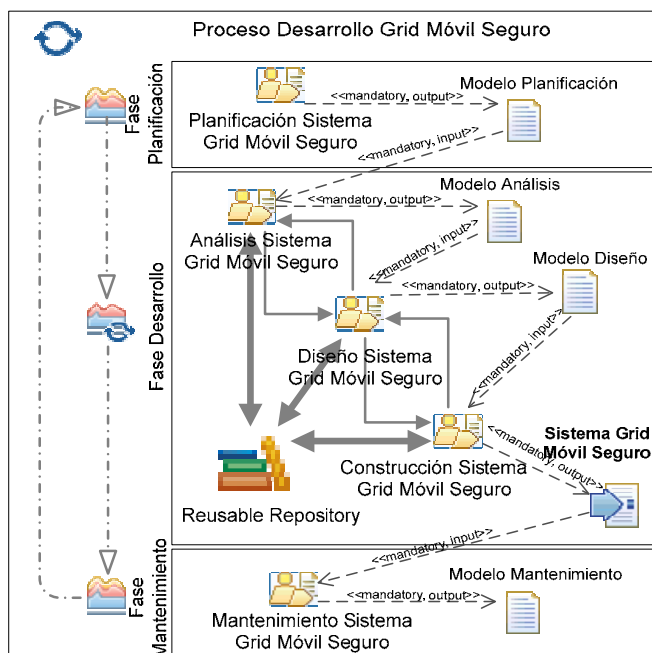


Fig. 1. Metodología de desarrollo para sistemas Grid móviles seguros

En este artículo nos centraremos en la actividad de Diseño y más concretamente, en la definición de una arquitectura de seguridad que sirva de referencia para el desarrollo de cualquier sistema Grid móvil seguro.

La actividad de análisis ha sido definida en algunos trabajos previos [20, 21, 23-25] donde las actividades y tareas son explicadas en profundidad. El análisis está centrado en construir diagramas de casos de uso y casos de uso de seguridad identificando y analizando todos los requisitos y requisitos de seguridad a partir de los casos de uso del diagrama construido para la aplicación, ayudado por los casos de uso reutilizables almacenados en el repositorio. Se construye el modelo de análisis que es la entrada a la actividad de diseño.

4 Actividad de Diseño

El objetivo de esta actividad es el diseño de una solución arquitectural donde se define la arquitectura software junto con la arquitectura de seguridad del sistema y del entorno tecnológico que le dará soporte, especificando de forma detallada todos los componentes del sistema Grid, siempre considerando el uso de dispositivos móviles. Esta arquitectura que se pretende diseñar debe ser una arquitectura de seguridad genérica que sirva de referencia para construir cualquier sistema Grid seguro, por tanto debe contemplar y dar soporte a todas las arquitecturas y tecnologías grid existentes.

Por tanto, se debe diseñar una arquitectura común para la construcción de sistemas Grid con dispositivos móviles siguiendo los requisitos y especificaciones analizadas y obtenidas en las actividades previas, y además, es necesario enriquecer la metodología con aspectos de seguridad, de tal forma que tengamos un proceso sistemático para la construcción de sistemas Grid con dispositivos móviles bajo entornos seguros.

Los dispositivos móviles son expuestos a frecuentes desconexiones desde la red, además de ser poco fiables. No es buena idea que los dispositivos móviles interactúen directamente con un sitio grid estático. Imaginemos que los dispositivos móviles se desconectan cuando estamos intercambiando datos con el sitio grid. A veces, los trabajos ejecutándose en sitios grid necesitan interactuar con el usuario móvil, pero el usuario móvil no tiene conexión. Así, un sistema fiable debe desempeñar el envío de trabajos, la monitorización, la cancelación, y la terminación del proceso por parte del usuario móvil [26].

Nuestra arquitectura se construye bajo un entorno seguro, flexible, escalable, interoperable, dinámico y abierto, por tanto, siguiendo estas metas podremos cubrir la mayoría de requisitos [27, 28], utilizando estándares que puedan ser entendidos por la gran mayoría de aplicaciones, ofreciendo protocolos de seguridad para las comunicaciones y permitiendo múltiples implementaciones dentro de la misma arquitectura y que sean capaces de cooperar entre ellas.

4.1 Definiendo Capas de Servicios de Seguridad

Considerando que la mayoría de los modelos y arquitecturas se definen mediante capas o niveles, nosotros seguiremos el mismo esquema de capas para definir nuestra arquitectura de servicios de seguridad para entornos Grid móviles. La idea de esta arquitectura de servicios es definir una arquitectura de referencia, donde se ordenan y dividen por capas los servicios de seguridad que son necesarios para la arquitectura a construir.

Cada una de las capas puede interoperar con el resto de capas adyacentes para ofrecer o beneficiarse de los servicios de las otras capas para construir nuevos servicios o mecanismos de seguridad. Podemos ver en la Fig. 2 una propuesta de 8 capas, donde se engloban todos los servicios y mecanismos de seguridad necesarios que cubren todos los requisitos de seguridad [27, 28] para los entornos Grid con dispositivos móviles.

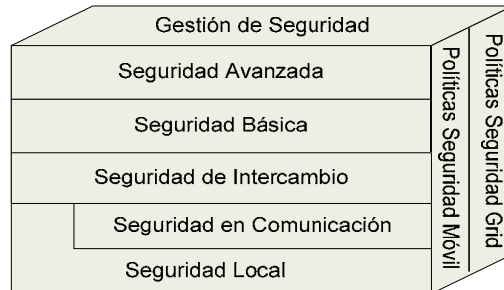


Fig. 2. Capas de Servicios de Seguridad para entornos Grid móviles.

Debemos identificar todos los servicios de seguridad que nuestra arquitectura debe soportar, y asignarlos a las capas correspondientes por su funcionalidad y la relación con otros elementos de la arquitectura.

La Tabla 1 muestra los objetivos de seguridad que debemos cumplir para cada una de las capas, y estudiar los servicios necesarios que deben estar presentes en nuestra arquitectura y que faciliten la consecución de dichos objetivos.

Tabla 1. Relación entre Capas, Objetivos y Servicios de Seguridad

Capas de Seguridad	Objetivos de seguridad	Servicios de seguridad
Seguridad Local	Seguridad del dispositivo; Seguridad hardware; Seguridad Software; Seguridad Recurso; Seguridad del Contenido; Seguridad de Claves; Protección de Datos; Protección de Password	Autorización, Protección Autenticación, Antivirus
Seguridad de Comunicación	Seguridad de transporte; Seguridad de mensajes; Seguridad inalámbrica.	Confidencialidad, Integridad, Privacidad, Conversación Segura
Seguridad de Intercambio	Intercambio políticas; Intercambio Credenciales y Privilegios; Intercambio Identidad y Claves	Intercambio; Traducción Credenciales & Identidad
Seguridad Básica	Autenticación; Autorización; Confidencialidad; Integridad; Disponibilidad; No repudio	Autenticación, Autorización, Disponibilidad, No repudio
Seguridad Avanzada	Confianza; Delegación; SSO; Logging; Audit; Contexto de Seguridad; Antivirus	Confianza; Delegación & SSO; Logging & Audit
Políticas Seguridad Grid	Definir y Manejar políticas de Integridad; Privacidad; Autorización; Autenticación; Delegación; Federación; Confianza.	Políticas Grid
Políticas seguridad dispositivos móviles	Definir y manejar políticas de Acceso a dispositivos móviles; Localización; Protección de Password; Protección tarjeta almacenamiento; Cifrado ficheros; Protección física; Protección datos; Backups; Restricciones Software.	Políticas Dispositivos móviles
Gestión de seguridad	Gestionar Claves, Usuarios, Políticas Grid, Políticas Dispositivos Móviles, Recursos, Credenciales e Identidad.	Gestión Seguridad

Ahora explicaremos cada una de las capas de seguridad consideradas y los servicios de seguridad que debemos considerar en cada capa, indicando los aspectos de seguridad más relevantes que creemos deben ser tenidos en cuenta en cada capa, considerando los aspectos y características de los entornos Grid con dispositivos móviles. Los servicios de seguridad que se desprenden de esta tabla son: Autorización, Protección, Autenticación, Antivirus, Confidencialidad, Integridad, Privacidad, Conversación Segura, Intercambio, Traducción Identidad & Credenciales, Disponibilidad, No repudio, Confianza, Delegación & SSO, Logging & Audit, Políticas Grid, Políticas Dispositivos móviles, y Gestión Seguridad.

4.1.1 Servicios de Seguridad Locales

En esta capa se agrupan los servicios de seguridad que son exclusivos de los recursos y dispositivos móviles que forman parte del Grid. Son los servicios que ofrecen cierto nivel de seguridad a los recursos (incluyendo dispositivos móviles) tanto de forma aislada como conectado al sistema Grid. Entre los servicios nos encontramos con los servicios de Protección de Datos y Passwords para los dispositivos móviles, servicio de Autenticación de usuario, servicio de Backup y Antivirus residentes en los dispositivos móviles, y el servicio de Autorización para el acceso a dicho recurso.

Los dispositivos móviles que almacenan información o sirven para conectarse a la red, deberían proteger la clave con fuertes frases de paso (passphrases/PINs). Esto previene que los ladrones entren en el dispositivo y accedan a la información almacenada en él. Para proteger contra pérdida de datos, los ficheros deben ser volcados a una localización segura fuera del dispositivo (volcar regularmente los datos de un PDA a un PC para evitar daños por virus o gusanos). Es recomendable usar antivirus en los dispositivos como PDAs, ya que los escaneos a nivel de red son más efectivos y es una forma centralizada de prevenir virus y otras interrupciones asociadas con los dispositivos móviles. Las soluciones firewall son más fáciles y efectivas de controlar y manejar que soluciones similares alojadas en el operador móvil.

Siempre que el dispositivo cliente se conecte a una red corporativa, el usuario y el dispositivo deben ser autorizados para hacer cualquier cambio en la red. En general, las soluciones de autenticación efectivas son aquellas que permiten acceder a la red sólo a usuarios autorizados. Las soluciones de autenticación incluyen el uso de nombres de usuario y claves, smart cards, biométricas, o PKI; o una combinación de soluciones (por ejemplo, smart cards con PKI). Cuando confiamos en nombres de usuario y claves, es importante tener políticas especificando una longitud mínima de la clave, caracteres de clave requeridos, y caducidad de claves.

4.1.2 Servicios de Seguridad de Comunicación

La seguridad de comunicación es a menudo descrita en términos de confidencialidad, integridad, autenticación y no repudio de los datos transmitidos. Estos servicios de seguridad son a su vez implementados por varios mecanismos que suelen ser de naturaleza criptográficos [29].

La confidencialidad de los datos transmitidos puede ser proporcionada cifrando la información entre las partes que se comunican, bien end-to-end o en alguna de las partes implicadas. La autenticación de los datos transmitidos es un servicio

asimétrico, queriendo decir que cuando A y B están comunicándose, la autenticación de los datos de A en B es independiente de la autenticación de los datos de B en A. Los tipos de autenticación disponibles dependen del protocolo de seguridad utilizado. En Internet, por ejemplo, SSL permite cifrar con cuatro opciones de autenticación diferentes: 1) autenticación del servidor, 2) autenticación del cliente, o 3) autenticación tanto del servidor como el cliente, o 4) ninguna autenticación, por ejemplo, proporcionando sólo confidencialidad.

Debemos proporcionar seguridad a nivel de mensaje y de transporte. La seguridad a nivel de mensaje es alcanzada mediante tecnologías y estándar tales como WS-Security, XML-Encryption y XML-Signature. La seguridad a nivel de mensaje se basa en WS-Security y soporta la privacidad e integridad, y es un óptimo esquema si se envían pocos mensajes. También podemos establecer seguridad a nivel de mensaje mediante WS-SecureConversation que soporta delegación de credenciales, privacidad, integridad y autenticación anónima. Si se envían muchos mensajes, su rendimiento es bueno.

4.1.3 Servicios de Intercambio de Seguridad

Es posible que este servicio sea necesario cuando trabajamos con identidades de distinta naturaleza y bajo distintos dominios de aplicación. Una identidad que es válida para el Grid puede no ser válida para los dominios porque trabajan con otro tipo de identidad. Es por ello que se debe mapear la identidad para el dominio local cuando se quiera acceder a un recurso, y viceversa si queremos acceder al Grid. Las identidades globales que maneja el Grid no son aceptadas en muchos dominios locales, que obligan a hacer una traducción de identidad en su dominio local y asignarle los derechos locales correspondientes con la identidad local. Por ello, cuando una petición accede al recurso, se debe hacer un mapeo de identidad global a local para que el servicio de autorización local trabaje con dicha identidad local.

Cada dominio local tiene sus propios mecanismos y políticas de seguridad para proteger al recurso. Es tarea de la arquitectura ofrecer los servicios necesarios para poder comunicarse con las políticas de seguridad locales de forma que puedan comunicarse y enviar peticiones de acceso a recursos e intercambio de información. Esto significa que es posible que se requiera un servicio de conversión de formatos, de credenciales o de protocolos para comunicarse con la infraestructura de seguridad local de cada dominio.

4.1.4 Servicios de Seguridad Básicos

Los servicios básicos que consideramos son los servicios básicos de seguridad para entornos abiertos genéricos, y que son igualmente importantes para entornos Grid. Estos servicios son: Autenticación, Autorización, Non-repudio, privacidad, integridad y confidencialidad. A partir de estos servicios básicos, ampliándolos, modificándolos o combinándolos podemos crear nuevos servicios (servicios de seguridad avanzados) que son necesarios para construir una arquitectura de seguridad para sistemas Grid con dispositivos móviles.

Algunos de estos servicios de seguridad que consideramos básicos, ya han sido asignados a la capa de comunicación (Integridad, Privacidad, Confidencialidad), por lo que en esta capa se definen el resto de servicios básicos, Autenticación,

Autorización y No-repudio. Nosotros añadimos el servicio de Disponibilidad que se refiere a la capacidad de las partes autorizadas a obtener acceso a la información cuando sea necesario.

Una función importante de una arquitectura de descubrimiento/entrega de servicio es reaccionar rápidamente a fallos. Un servicio, por ejemplo, podría no estar disponible más debido a un fallo del servidor o a la movilidad, por ejemplo. La disponibilidad puede ser definida como la propiedad de un sistema que siempre cumple con peticiones legítimas por entidades autorizadas. Es violada cuando un atacante consigue denegar el servicio a los usuarios legítimos (por ejemplo, utilizando todos los recursos disponibles).

La protección contra ataques y amenazas a la seguridad es importante, ya que está fuertemente vinculada a la disponibilidad de la red y, por tanto, a todos los servicios y funcionalidades que una red móvil ofrece a los usuarios y diversas organizaciones. Los ataques de denegación del servicio (DoS, Denial-of-service) han llegado a ser una de las más obvias amenazas que pueden tener un efecto severo en la disponibilidad, rendimiento y calidad de los servicios en una red móvil. Las soluciones de seguridad de redes móviles deben proporcionar protección contra una variedad de amenazas existentes y trabajar para asegurar la disponibilidad de los recursos de red [30].

El no repudio es similar a la autenticación en que es un servicio de seguridad asimétrico. Un simple ejemplo para describir la diferencia entre autenticación y no repudio es que con la autenticación el destinatario está seguro del origen del mensaje pero no sería capaz de convencer a nadie más sobre esto, sin embargo, con el no repudio, el destinatario sería capaz de convencer a terceras partes del origen del mensaje. La firma digital es el mecanismo usado para el no repudio. Hay que controlar las acciones que tienen lugar, los solicitantes y receptores de las acciones, para proteger de las falsas negaciones. Algunas técnicas comunes para proporcionar no repudio son la autenticación combinadas con el logging y firmando el contenido de la comunicación con claves privadas.

Para los servicios básicos de seguridad, debemos emplear todos los estándares existentes para crear una arquitectura abierta y que pueda inter-operar con múltiples servicios de otras arquitecturas existentes. Así por ejemplo, para la autenticación, el protocolo basado en TLS es usado para llevar a cabo la autenticación y proporcionar protección del mensaje (cifrado y prueba de integridad). La infraestructura de clave pública (PKI) junto con certificados X.509 son frecuentemente utilizados.

4.1.5 Servicios de Seguridad Avanzados

Uno de los requisitos importantes de los entornos Grid es que un usuario pueda tener acceso a un gran número de recursos distribuidos autenticándose una sola vez. Esto es referido como single sign-on, que lo consideramos como un servicio avanzado donde interviene el servicio de autenticación de la capa de seguridad básica. Para permitir compartir grandes cantidades de datos y recursos computacionales a través de comunidades de usuarios altamente distribuidas de forma eficiente y efectiva, es esencial permitir la delegación. Uno de los factores característicos de una infraestructura Grid es la necesidad de delegar derechos de entidad a otras entidades dentro del Grid.

Confianza o Trust puede generalmente ser definida como tener la confianza que una parte se comporta de una manera esperada a pesar de la falta de capacidad para

supervisar o controlar la otra parte. En un entorno donde se describe la política de acceso en términos de identidad de usuarios o de atributos requeridos, la gestión de la confianza consiste en definir las fuentes de autoridades para la identificación del usuario, la asignación de atributos y, posiblemente, la creación de política. En un sistema donde los usuarios pueden delegar algunas o la totalidad de sus derechos a otros usuarios, el control de dicha delegación es parte de la gestión de confianza [5]. El Grid debe establecer confianza entre los distintos dominios para asegurar que las peticiones, los mensajes y los datos que son manejados en el Grid, proceden de dominios de confianza, que han sido protegidos de ataques, y han seguido las políticas de seguridad que garanticen su seguridad.

Hay que registrar todo lo que vaya sucediendo en el sistema, los trabajos enviados, la memoria, la utilización de recursos, estado de los trabajos, resultados obtenidos, etc. Hay que considerar que los ficheros logs pueden residir en diferentes dominios administrativos, por lo que el acceso seguro a los ficheros logs es una tarea compleja. Los logs deben ser asegurados y a prueba de manipulación, y capacitado para asegurar la integridad del mensaje.

El servicio de Auditoría es conducida por política y responsable de registrar los eventos relevantes de seguridad. Este servicio es típicamente usado por los administradores de seguridad dentro de una VO para comprobar la adherencia a las políticas de control de acceso y autenticación. La auditoría requiere que los eventos sean registrados de forma segura.

4.1.6 Servicios de Política Grid

En esta capa se definen todas las políticas de seguridad que definirán el comportamiento de los servicios de seguridad, de las comunicaciones y de las transacciones que se realicen en el sistema. Tendremos que definir las políticas de integridad, de autorización, de intercambio de políticas, etc. Debemos definir una política de seguridad, un conjunto de roles que definen los sujetos de seguridad (usuarios), los objetivos de seguridad (recursos) y las relaciones entre ellos. Esta capa debe estar en contacto directo con cada una de las capas anteriores para asegurar que cada servicio en cada capa tenga acceso a sus respectivas políticas de seguridad de forma directa y segura.

Las políticas de seguridad son un conjunto de reglas que gobiernan el sistema y que lo protegen de ataques y amenazas. Todas las acciones, peticiones y transacciones deben llevarse siguiendo alguna política de seguridad. La política de seguridad se refiere al conjunto de reglas, leyes y prácticas que regulan cómo una organización gestiona, protege y distribuye la información delicada. Una política de seguridad debe especificar los objetivos de seguridad que el sistema debe cumplir y las amenazas que deben resistir. Esto implica que una política de seguridad debe determinar el tipo de comunicación requerida para las diversas transacciones, el tipo de autenticación, los procedimientos de auditoría, técnicas de recuperación y control de acceso, que es una parte importante de la política de seguridad. En general, la política de control de acceso especifica quién puede acceder a determinados recursos y qué operaciones se permiten en esos recursos.

Se pueden emplear las especificaciones SAML y XACML para expresar las políticas de control de acceso, sentencias de autorización, y protocolos de autorización. La especificación XACML establece el mecanismo de política de

recurso a cada recurso o servicio. SAML también tiene un mecanismo de política pero está muy limitado para el Grid, mientras que XACML proporciona un mecanismo mucho más flexible que puede ser aplicado a cualquier tipo de recurso.

4.1.7 Servicios de Política Móvil

Es conveniente crear políticas de seguridad específicas para el uso de dispositivos móviles, así por ejemplo, si queremos minimizar el impacto por la pérdida de un dispositivo, debemos proteger con contraseñas todos los dispositivos, cifrar los documentos sensibles en el dispositivo, y no usar scripts automáticos para el acceso a VPN. Las políticas de seguridad del dispositivo móvil deben incluir también un acceso mínimo a fuentes limitadas utilizando cortafuegos.

Las empresas deben tratar la seguridad móvil como una tarea independiente, y como tal, las políticas de seguridad específicas para el uso móvil deben ser creadas e implementadas. Un análisis de riesgo global de los posibles riesgos de seguridad asociados con el uso de dispositivos móviles debe ser el primer paso hacia la creación de la política de seguridad de dispositivos móviles.

El primer paso, entonces, es desarrollar las políticas de seguridad razonables para gobernar el uso de dispositivos móviles en el Grid. La organización debe disponer de políticas que son específicas para los dispositivos móviles, no sólo tratar de aplicar las políticas generales de seguridad. También es importante educar a los usuarios del dispositivo móvil sobre las cuestiones de seguridad, incluida la seguridad física. Algunas políticas de seguridad pueden ser aplicadas tecnológicamente, pero otras dependen del cumplimiento del usuario.

4.1.8 Servicios de Gestión de Seguridad

El modelo de seguridad Grid agrupa todas las funciones de gestión de seguridad aplicables a los diversos aspectos de vinculación, política y federación. Éstos incluyen gestión de claves para funciones criptográficas, gestión de registro de usuarios, gestión de dispositivos, autorización, privacidad y gestión de política de confianza, y gestión de reglas de mapeos que permite la federación. También puede incluir la gestión de detección de intrusos, servicios de antivirus y garantía de servicio de información que permita a los solicitantes de servicios descubrir lo que pueden ofrecer los mecanismos de seguridad de un entorno de alojamiento o host. Abordar la gestión de diversos aspectos de la infraestructura de seguridad satisface los requisitos de manejabilidad en el entorno Grid [31].

Cada servicio, ya sea en la capa de comunicación, básica o avanzada, necesitará controlar y gestionar la información que necesita para llevar a cabo su tarea. Por eso, esta capa debe colaborar de forma directa con cada capa de la arquitectura, ofreciendo los servicios de gestión necesarios para cada servicio.

Los servicios de gestión de la seguridad se encargan de gestionar, por ejemplo los usuarios móviles que pertenecen al grid o quieren pertenecer, los derechos y privilegios para cada usuario o role, las claves generadas para cada usuario o grupo de usuarios, los recursos cedidos o eliminados de los que dispone el sistema, las identidades de cada usuario o servicio dentro del Grid, y por supuesto la gestión de las políticas de seguridad (integridad, privacidad, autorización, etc.).

5 Conclusiones

Este artículo presenta la arquitectura de seguridad para manejar diversos problemas de seguridad en el Grid. Cómo construir un sistema Grid seguro es también una tarea importante debido a la alta complejidad del sistema de computación Grid. Es difícil incorporar de forma segura dispositivos móviles existentes dentro del Grid, de manera que el impacto sea mínimo y transparente para el usuario. Esta es la razón de la necesidad de elaborar y definir una metodología de desarrollo de un sistema basado en Grid y tecnología móvil, considerando, desde las primeras etapas del desarrollo, las peculiaridades y necesidades de seguridad para este tipo de sistemas. El Grid necesita construir un sistema de seguridad unificado que soporte todas las funciones de seguridad, las cuales residen en los sistemas Grid, sea independiente de otros sistemas y tenga una buena relación de cooperación con el resto de sistemas.

Una fase importante de la metodología es el diseño de la arquitectura de seguridad que hemos propuesto con un conjunto de servicios de seguridad que cubren todos los requisitos de seguridad y usa tecnologías estándar y especificaciones de los servicios web para obtener una arquitectura de seguridad abierta, escalable e interoperable. Esta arquitectura es una arquitectura de referencia de nuestra metodología y sirve como base para construir una arquitectura específica con necesidades y requisitos especiales para los diferentes tipos de entornos Grid móvil.

Como trabajo futuro, completaremos la metodología describiendo formalmente las fases, actividades y tareas con las especificaciones SPEM. Estudiaremos en profundidad los servicios de seguridad propuestos, identificando las tecnologías de seguridad, los protocolos y mecanismos que podemos utilizar en cada servicio y relacionaremos los requisitos de seguridad con los servicios de seguridad utilizados para resolver estos requisitos. Construiremos una arquitectura orientada a servicios de seguridad desde varios puntos de vista (lógica, comunicación, implementación, servicios, red, etc.) y aplicaremos esta metodología a un caso real. Una adaptación de esta propuesta para Cloud computing, que es una tecnología que permite ofrecer servicios de computación a través de Internet, está siendo estudiada.

Agradecimientos. Esta investigación es parte de los siguientes proyectos: QUASIMODO (PAC08-0157-0668) financiado por la “Viceconsejería de Ciencia y Tecnología de la Junta de Comunidades de Castilla-La Mancha” (España), SISTEMA (PII2I09-0150-3135) financiado por FEDER y la “Consejería de Educación y Ciencia de la Junta de Comunidades de Castilla-La Mancha” (España), y ESFINGE (TIN2006-15175-C05-05) concedida por la “Dirección General de Investigación del Ministerio de Educación y Ciencia” (España).

Referencias

1. Foster, I., Kesselman, C., Tuecke, S.: The Anatomy of the Grid: Enabling Scalable Virtual Organizations. 7th International Euro-Par Conference Manchester on Parallel Processing. 15(3), 1 - 4 (2001)

2. Litke, A., Skoutas, D., Varvarigou, T.: Mobile Grid Computing: Changes and Challenges of Resource Management in a Mobile Grid Environment. In: 5th International Conference on Practical Aspects of Knowledge Management (PAKM 2004), (2004)
3. Guan, T., Zaluska, E., Roure, D.D.: A Grid Service Infrastructure for Mobile Devices. In: First International Conference on Semantics, Knowledge, and Grid (SKG 2005), Beijing, China (2005)
4. Jameel, H., Kalim, U., Sajjad, A., Lee, S., Jeon, T.: Mobile-To-Grid Middleware: Bridging the gap between mobile and Grid environments. In: European Grid Conference EGC 2005, pp. 932-941. Springer, Amsterdam, The Netherlands (2005)
5. Humphrey, M., Thompson, M.R., Jackson, K.R.: Security for Grids. Lawrence Berkeley National Laboratory. Paper LBNL-54853. (2005)
6. Chakrabarti, A., Damodaran, A., Sengupta, S.: Grid Computing Security: A Taxonomy. IEEE Security & Privacy. 6, 44-51 (2008)
7. Bradford, P.G., Grizzell, B.M., Jay, G.T., Jenkins, J.T.: Cap. 4. Pragmatic Security for Constrained Wireless Networks. In: Publications, A. (ed.): Security in Distributed, Grid, Mobile, and Pervasive Computing, The University of Alabama, Tuscaloosa, USA (2007) 440
8. Kwok-Yan, L., Xi-Bin, Z., Siu-Leung, C., Gu, M., Jia-Guang, S.: Enhancing Grid Security Infrastructure to Support Mobile Computing Nodes. Lecture Notes in Computer Science. 2908/2003, 42-54 (2004)
9. Sajjad, A., Jameel, H., Kalim, U., Han, S.M., Lee, Y.-K., Lee, S.: AutoMAGI - an Autonomic middleware for enabling Mobile Access to Grid Infrastructure. In: Joint International Conference on Autonomic and Autonomous Systems and International Conference on Networking and Services - (icas-icns'05), pp. 73-73. (2005)
10. Baskerville, R.: Information systems security design methods: implications for information systems development. ACM Computing Surveys. 25, 375 - 414 (1993)
11. Anderson, R.: Security Engineering - A Guide to Building Dependable Distributed Systems. John Wiley & Sons (2001)
12. Phan, T., Huang, L., Dulani, C.: Challenge: Integrating Mobile Wireless Devices Into the Computational Grid. In: 8th annual international conference on Mobile computing and networking (MobiCom'02), pp. 271 - 278. ACM Press, Atlanta, Georgia, USA (2002)
13. Clarke, B.A.M.H.: Beyond the 'Device as Portal': Meeting the Requirements of Wireless and Mobile Devices in the Legion Grid Computing System,". In: Parallel and Distributed Computing Issues in Wireless Networks and Mobile Computing at the International Parallel and Distributed Processing Symposium., IEEE Press, (2002)
14. Chapin, S., Wang, C., Wulf, W., Knabe, F., Grimshaw, A.: A New Model of Security for Metasystems. Future Generation Computer Systems. 15, 713-722 (1999)
15. Foster, I., Kesselman, C.: The Grid: Blueprint for a Future Computing Infrastructure. Morgan Kaufmann Publishers; 1ST edition, San Francisco, CA (1999)
16. Ferrari, A., Knabe, F., Humphrey, M., Chapin, S., Grimshaw, A.: A Flexible Security System for Metacomputing Environments. In: CS-98-36, T.R. (ed.). Department of Computer Science. University of Virginia (1998)
17. van Steen, M., Homburg, P., Tanenbaum, A.S.: Globe: A Wide-Area Distributed System. IEEE Concurrency. 70-78 (1999)
18. Belani, E., A.Vahdat, Anderson, T., Dahlin, M.: CRISIS: A wide area security architecture. In: Seventh USENIX Security Symposium, (1998)
19. Rosado, D.G., Fernández-Medina, E., López, J., Piattini, M.: PSecGCM: Process for the development of Secure Grid Computing based Systems with Mobile devices. In: International Conference on Availability, Reliability and Security (ARES'08), pp. 136-142. IEEE, Barcelona, Spain (2008)

20. Rosado, D.G., Fernández-Medina, E., López, J., Piattini, M.: Engineering Process Based On Grid Use Cases For Mobile Grid Systems. In: The Third International Conference on Software and Data Technologies- ICSOFT 2008, pp. 146-151. Porto, Portugal (2008)
21. Rosado, D.G., Fernández-Medina, E., López, J.: Obtaining Security Requirements for a Mobile Grid System. International Journal of Grid and High Performance Computing. (to be published in April 1, 2009) (2009)
22. OMG: Software & Systems Process Engineering Meta-Model Specification (SPEM) 2.0. (2008) <http://www.omg.org/spec/SPEM/2.0/PDF>
23. Rosado, D.G., Fernández-Medina, E., López, J.: Reutilización de Casos de Uso en el Desarrollo de Sistemas Grid seguros. In: XII Conferencia Iberoamericana de Ingeniería de Requisitos y Ambientes de Software – IDEAS 2009 (accepted), Medellín, Colombia (2009)
24. Rosado, D.G., Fernández-Medina, E., López, J.: Reusable Security Use Cases for Mobile Grid environments. In: Workshop on Software Engineering for Secure Systems (SESS'09), in conjunction with the 31st International Conference on Software Engineering (ICSE 2009) Vancouver (Canada) (2009)
25. Rosado, D.G., Fernández-Medina, E., López, J., Piattini, M.: PSecGCM: Process for the development of Secure Grid Computing based Systems with Mobile devices. In: International Conference on Availability, Reliability and Security (ARES 2008), pp. 136-142. IEEE, Barcelona, Spain (2008)
26. Park, S.-M., Ko, Y.-B., Kim, J.-H.: Disconnected Operation Service in Mobile Grid Computing. In: International Conference on Service Oriented Computing (ICSOC'2003), Trento, Italy (2003)
27. Vivas, J.L., López, J., Montenegro, J.A.: Cap. 12. Grid Security Architecture: Requirements, fundamentals, standards, and models. In: Publications, A. (ed.): Security in Distributed, Grid, Mobile, and Pervasive Computing, Tuscaloosa, USA (2007) 440
28. Trusted Computing Group Administration: Securing Mobile Devices on Converged Networks. (2006) https://www.trustedcomputinggroup.org/groups/mobile/Final_iGR_mobile_security_white_paper_sept_2006.pdf
29. Jøsang, A., Sanderud, G.: Security in Mobile Communications: Challenges and Opportunities. In: the Australasian information security workshop conference, Adelaide, Australia (2003)
30. Nokia: Mobile Device Management and Security. (2007)
31. Nagaratnam, N., Janson, P., J. Dayka, Nadalin, A., Siebenlist, F., Welch, V., Tuecke, S., Foster, I.: The Security Architecture for Open Grid Services. (2002) <http://forge.gridforum.org/projects/ogsa-sec-wg/>.

Gestión de Identidad en las Administraciones Públicas: Interoperabilidad pan-Europea

Sergio Sánchez García, Ana Gómez Oliva,

DIATEL – EUITT – Universidad Politécnica de Madrid, Ctra. Valencia Km.7, 28031
Madrid, Spain
sergio@diatel.upm.es, agomez@diatel.upm.es

Resumen. En un entorno digital y cada vez más global, la provisión de servicios basados en la identidad electrónica de una persona o entidad y la gestión de dicha identidad por parte de las Administraciones Públicas constituyen un importante reto, muy especialmente cuando los ciudadanos, y en general las entidades a autenticar, presentan acreditaciones procedentes de distintos países y, por tanto, ajustadas a la legislación particular de cada uno de ellos. En este artículo se recogen y analizan las soluciones propuestas hasta la fecha en el marco de la Unión Europea y se presenta la propuesta de una infraestructura totalmente interoperable para gestión de identidad que está desarrollando este grupo de investigación. La novedad de esta propuesta reside en que puede emplearse tanto para la interoperabilidad con otros países europeos como entre los distintos niveles de la Administración española (local, autonómico y central).

Palabras clave: eID, eIDM, identidad electrónica, sistemas de gestión de identidad electrónica, pan-Europeo.

1 Introducción

En algunos países europeos existe tradicionalmente un sistema de identificación de ciudadanos basado en un documento que éstos poseen y que les identifica de forma unívoca. A pesar de los distintos nombres que recibe en función del país (Documento Nacional de Identidad en España, Tarjeta de Ciudadano en Bélgica, etc.) su contenido y funcionalidad son prácticamente similares en todos ellos. Normalmente este documento ha ido evolucionando a lo largo del tiempo, pasando por distintos formatos, desde una simple hoja de papel que contenía un conjunto de datos personales “certificados” por una autoridad oficial a los más recientes documentos de identidad con fuertes medidas antifalsificación y datos adicionales, como por ejemplo fotografía y huella dactilar, que permiten una identificación biométrica y, presumiblemente, más fiable de su poseedor. A pesar de la evolución en el formato, la funcionalidad y ámbito de aplicación de los documentos de identificación se puede considerar estancada. Este tipo de tarjetas ha servido tradicionalmente como medio de identificación del ciudadano ante la Administración Pública de su país y,

adicionalmente, es utilizado por las empresas privadas como medio de identificación de los usuarios de sus servicios, puesto que su identidad está reflejada en el mismo y los empleados pueden verificarla visualmente de forma directa. Sin embargo, la aparición de Internet y el paulatino acercamiento de los ciudadanos a la sociedad digital han provocado un cambio en este aspecto. Cada vez más trámites, tanto con la Administración como con los proveedores de servicios, se realizan a través de la Red y, por lo tanto, existe una creciente demanda de sistemas de identificación que permitan realizar este tipo de transacciones sin pérdida de garantías en cuanto a la seguridad. Por tanto, es preciso dotar a los ciudadanos de una *identidad electrónica o digital* que les permita identificarse en la Red, al menos, con las mismas garantías con las que lo hace con su tarjeta de ciudadano en las interacciones interpersonales.

Con la intención de subsanar el problema anterior, en la mayoría de los países pertenecientes a la Unión Europea se está llevando a cabo la implantación de tarjetas de identificación electrónicas, también denominadas *eID cards*, cuyo aspecto exterior es similar al de los documentos de identificación actuales pero con la salvedad de que se incluye un chip que permite el almacenamiento electrónico de información sobre la identidad, así como la interacción con ciertas aplicaciones, de forma que el usuario pueda demostrar digitalmente su identidad.

Además de un entorno cada vez más digital, los ciudadanos europeos, al igual que los del resto del mundo, se encuentran en un entorno cada vez más global. A día de hoy un ciudadano español puede trabajar para una empresa alemana y desarrollar su labor profesional en Bélgica sin, teóricamente, ningún tipo de traba y, adicionalmente, debe poder llevar a cabo interacciones a través de la Red tanto con su empresa como con las Administraciones Públicas de los distintos países. Este entorno global da lugar a una serie de problemas que surgen cuando nos planteamos, por ejemplo, cuestiones del siguiente tipo: ¿cómo el ciudadano, con su tarjeta de identificación electrónica de España accede a los servicios ofrecidos en la Red por la Administración Pública alemana?, ¿y a sus datos como trabajador en Bélgica?, es más, ¿cómo gestiona la Administración Pública alemana los datos de identidad del ciudadano? La respuesta a estas preguntas no es sencilla, pero en todo caso pasa por la especificación y el desarrollo de un conjunto de infraestructuras técnicas y organizativas que permitan la definición, administración y gestión de los atributos relativos a la identidad de los ciudadanos. Estas infraestructuras reciben el nombre de Sistemas de Gestión de Identidad (*Identity Management Systems* o *IDMs*).

Este artículo recoge, en primer lugar, una visión general del estado actual de los sistemas de gestión de identidad a nivel pan-Europeo, analizando las principales propuestas realizadas hasta la fecha en el marco de la Unión Europea e identificando aquellos puntos sobre los que ya existen unos principios de acuerdo y aquellos otros pendientes de solución que impiden hasta la fecha la adopción de un modelo a gran escala. A continuación se presenta la propuesta de arquitectura de interoperabilidad sobre la que está trabajando este grupo, que da solución a los problemas detectados y que presenta como particularidad que no sólo puede emplearse para la interoperabilidad de los sistemas de gestión entre distintos países europeos, sino que la solución es extrapolable a las distintas esferas de la Administración Pública española (local, autonómica y central). Por último, se comenta la experiencia piloto que se está realizando para poner en marcha las soluciones adoptadas.

Esta propuesta se enmarca dentro de los trabajos que este grupo de investigación lleva a cabo en el proyecto TSI2006-4864 *Plataforma telemática de Administración Electrónica basada en coreografía de servicios* subvencionado por el Ministerio de Educación y Ciencia, en el Plan nacional de I+D+I.

2 La Identidad y la Identidad Digital

Antes de comenzar a hablar de sistemas de gestión de identidad conviene dejar claro lo que se entiende por identidad y por identidad digital.

De acuerdo a [1] el concepto de identidad se puede definir desde tres puntos de vista distintos, el sociológico, el legal y el tecnológico. El punto de vista sociológico, que evidentemente debe ser estudiado y considerado por expertos en la materia, queda fuera del ámbito del estudio que se pretende realizar en este artículo, por lo que nos centraremos en la parte legal y tecnológica. Partiendo de esta consideración nos encontramos con que la identidad del individuo está compuesta, por un lado, por elementos encargados de garantizar la unicidad de una persona física y, por otro, por elementos que son la expresión de la identidad humana en todos sus posibles aspectos. Desde el punto de vista legal podemos decir que la **identidad personal** está formada, tradicionalmente, por el conjunto de datos resultantes de la unión de la información relativa a una persona presente en los registros públicos que va a permitir identificarla de forma unívoca. Podemos afirmar que a mayor cantidad de información sobre una persona, más “única” es esa persona. Por otra parte, ese tipo de información como puede ser el estado civil o la profesión, son consideradas por la legislación actual como superfluas a la hora de la identificación personal por estar demasiado ligadas a la esfera privada del individuo. La Directiva Europea 95/46/CE sobre protección de datos [2] tiene como finalidad el otorgar al sujeto el mayor control posible sobre su identidad y datos personales planteando una serie de requisitos a cumplir por los receptores, controladores, procesadores y terceras partes a la hora de manejarlos. El artículo 2, letra a) define *datos personales* de la siguiente forma:

“«datos personales»: toda información sobre una persona física identificada o identificable (el «interesado»); se considerará identificable toda persona cuya identidad pueda determinarse, directa o indirectamente, en particular mediante un número de identificación o uno o varios elementos específicos, característicos de su identidad física, fisiológica, psíquica, económica, cultural o social”

El concepto de **Identidad Digital** o Identidad de Red como la denomina la Liberty Alliance [3] surge como consecuencia de la interacción de los usuarios con los servicios que les son ofrecidos a través de la Red. Cuando los usuarios interactúan con los servicios a menudo los personalizan de acuerdo a sus preferencias o necesidades, de manera que además de establecer unos datos de acceso como nombre de usuario y contraseña establecen otros parámetros como, por ejemplo, la información que desean que se les muestre, la disposición de los elementos en la página que ofrece el servicio

o la forma de notificar los cambios en el mismo. Normalmente el establecimiento de una cuenta y la personalización de la misma se realiza por parte de los usuarios para cada uno de los proveedores de servicio a los que accede, de forma que el usuario dispone de múltiples cuentas con múltiples parámetros. De acuerdo a la Liberty Alliance la Identidad de Red de un usuario está constituida por el total de los conjuntos de atributos que constituyen las distintas cuentas de un usuario.

Concretamente en el draft *Liberty ID-FF Architecture Overview* [4] se define la Identidad de Red con la siguiente frase:

“A network identity is the global set of attributes composed from a user’s account(s).”

Para una entidad dada, típicamente existirán múltiples identidades digitales que pueden ser únicas o no. Una identidad digital es, por definición, un subconjunto de la identidad y puede ser considerada como la manifestación de la entidad en la Red.

3 Sistemas de Gestión de Identidad y Globalización

De acuerdo a la definición dada en [5], se considera un Sistema de Gestión de Identidad como la infraestructura técnica y organizativa para la definición, gestión y administración de atributos de identidad.

A lo largo de los últimos años en todos los estados miembros de la Unión Europea se están desarrollando iniciativas para la introducción de identidades electrónicas (eID) en los servicios públicos y los correspondientes sistemas de gestión de dichas identidades. La utilización de este tipo de sistemas de gestión y de las identidades electrónicas resulta muy útil a los ciudadanos en cuanto que les permite un acceso mucho más rápido y eficaz a los servicios ofrecidos por las Administraciones Públicas pero, tal como se comentó en la introducción, surgen problemas debido a la globalización. La identidad electrónica de un ciudadano le permite operar correctamente dentro del entorno correspondiente a su propio país pero ¿qué ocurre cuando ese entorno pasa a ser el formado por el conjunto de países de la Unión Europea? El documento de identidad de un ciudadano español sigue siendo válido en Bélgica desde el punto de vista de una identificación o constatación de datos visual, por parte de una persona. Sin embargo, en el caso de la identidad electrónica no ocurre lo mismo puesto que los sistemas de identidad electrónica y de gestión de dicha identidad en cada país no son compatibles, es decir, no están preparados para interoperar. Surge por lo tanto uno de los principales problemas que nos ocupan, la interoperabilidad entre los sistemas de gestión de identidad a nivel pan-Europeo. De forma genérica podemos decir que dada la diversidad de sistemas de gestión de identidad, cuando un usuario de un sistema dado (ya sea un ciudadano, una empresa o la propia Administración) trata de comunicarse con Administraciones que se encuentran fuera del ámbito de su propio Sistema de Gestión de Identidad local surge la necesidad de comunicar los Sistemas de Gestión entre si y conseguir un entendimiento que permita que la identidad de un usuario de un sistema sea entendida y aceptada por el otro. Se hace necesario el establecimiento de un marco de

interoperabilidad a nivel de la Unión Europea para los Sistemas de Gestión de Identidad.

Con la intención de conseguir esto la Comunidad Europea adoptó el 25 de abril de 2006 su *eGovernment action plan* [6], en el que se realiza la siguiente afirmación:

The Commission, in cooperation with the Member States, will pursue policies to grant safe access to services EU wide. [...] EU governments have agreed to facilitate this process by establishing secure systems for mutual recognition of national electronic identities for public administration web-sites and services. The Action Plan foresees a full implementation by 2010.

Para conseguir establecer el marco de interoperabilidad la Unión Europea fijó un mapa de ruta [7]. En dicho mapa de ruta se establecen una serie de principios de diseño que se enuncian a continuación, todos en torno al principio fundamental de la subsidiaridad, es decir, cada estado miembro debe mantener su autonomía y responsabilidad para continuar con sus iniciativas de Sistemas de Gestión de Identidad:

- La usabilidad debe ser un aspecto dominante en el diseño de los sistemas.
- Cada estado miembro debe ser capaz de identificar a los usuarios dentro de sus fronteras.
- Cada estado miembro debe entregar a cada usuario los medios necesarios para identificarse y autenticarse a sí mismo electrónicamente.
- Con respecto a las autorizaciones de órdenes o representaciones, cada estado miembro debe proporcionar los medios para gestionar las competencias de los usuarios identificados dentro de sus fronteras.
- Cada estado miembro debe soportar mecanismos de validación online de identidades, competencias y órdenes.
- Entre los estados miembros se debe establecer un consenso a alto nivel respecto a la terminología en los Sistemas de Gestión de Identidad para garantizar la interoperabilidad semántica y conceptual.

A partir de los principios de diseño anteriores se derivan una serie de criterios sobre cómo debe ser un Sistema de Gestión de Identidad a nivel pan-Europeo:

- Federado. Debe existir una confianza mutua entre las distintas Administraciones en lo que se refiere a los métodos de identificación y autenticación.
- Multinivel. En el sentido de que se debe permitir a los Estados Miembros proporcionar múltiples niveles de seguridad para los servicios de gestión de identidad. Los requisitos de autenticación para cada servicio deben ser adaptados a las necesidades de seguridad de dicho servicio, lo que implica la definición a nivel Europeo de un conjunto de criterios para cada nivel de autenticación.
- Dependiente de fuentes fiables. Para garantizar la calidad de los datos debe existir, en cada Estado Miembro, una única fuente fiable para cada pieza de información correspondiente a una entidad registrada, de manera que se elimine la duplicidad de datos y se asegure una única fuente correcta y oficial.

- Permitir la incorporación del sector privado en aquellos Estados Miembros en los que se elija confiar en empresas privadas, por ejemplo instituciones financieras, para proporcionar servicios de gestión de identidad electrónica.

A pesar de la existencia del mapa de ruta, de los principios de diseño y los criterios comentados, se puede afirmar que, en la práctica, la interoperabilidad entre los sistemas de gestión de identidad de distintos países en Europa sigue siendo más una ambición que una realidad, aunque como veremos en el siguiente apartado se están proponiendo soluciones.

4 Análisis de las propuestas existentes hasta la fecha

Debido a los planes de acción lanzados por la Unión Europea, a lo largo de los últimos años han surgido iniciativas enfocadas a lograr la interoperabilidad entre los sistemas de gestión de identidad. La mayoría no pasan de ser propuestas teóricas que resuelven alguno de los problemas sin dar una solución completa, aunque algunas van más allá y proponen arquitecturas que se encuentran en la actualidad en fase de piloto. A continuación se comentan las más destacadas.

Quizá uno de los primeros estudios/proyectos dedicados a la interoperabilidad de los Sistemas de Gestión de Identidad fue el **Modinis eIDM Study** [8]. Dentro del modelo propuesto se llevó a cabo un estudio completo a alto nivel sobre el uso de los sistemas de gestión de identidad en los países miembros de la Unión Europea junto con un análisis de las principales consecuencias del uso de este tipo de sistemas de eIDM y un conjunto general de recomendaciones. Tras el estudio de los distintos perfiles nacionales, el estudio Modinis intentó definir un conjunto de aspectos clave para el despliegue de sistemas eIDM a escala europea. Su principal aportación a la interoperabilidad es la definición de una infraestructura de alto nivel, un modelo o marco conceptual, llamado Modinis Conceptual Framework, que es, fundamentalmente, un portal basado en federación que recoge las principales propuestas realizadas en el proyecto en cuanto a organización general y principios básicos que deben regir una infraestructura eIDM a nivel pan-Europeo. De forma resumida se puede decir que la infraestructura se basa en un modelo federado que confía en una serie de portales de identidad en cada Estado Miembro (al menos uno por estado, posiblemente más) que serían los responsables de la autenticación de una entidad a nivel nacional y de decidir el nivel de confianza que se otorga a los distintos procedimientos de autenticación realizados en cada Estado Miembro. En este modelo, los requisitos de autenticación para un servicio concreto en un determinado Estado Miembro aceptarían como equivalentes los niveles de autenticación y los mecanismos empleados en otro Estado en base a un conjunto de criterios, de manera que no sería necesario establecer ninguna infraestructura específica a nivel Europeo.

Como podemos ver, el modelo presentado constituye uno de los imaginables para un sistema pan-Europeo de Gestión de Identidad basado en un conjunto específico de conceptos centrales específicamente pensados para aplicaciones de *eGovernment*.

Otro sistema interesante de cara a dar solución a los sistemas de gestión de identidad es el denominado **TLS-Federation** [9].

Este proyecto tuvo como objetivo proporcionar un marco de trabajo regulatorio e interoperable para la gestión de identidad a nivel europeo. Se centró en el uso de tecnologías y estándares suficientemente conocidos y en la protección del lado del usuario frente a posibles escenarios de robo de identidad. El modelo TLS-Federation está basado en el uso de certificados durante el proceso de autenticación, lo que asegura la máxima seguridad para el proceso, y se construye sobre una aproximación centrada en el usuario, de manera que la identidad y los atributos de privacidad son gestionados directamente por él. El sistema está diseñado para ser usado en comunicaciones tanto entre sistemas dentro de un mismo país como entre distintos países. Cuando se compara TLS-Federation con otras posibles formas de llevar a cabo la gestión de identidad aparecen puntos prometedores, sobre todo a la hora de crear una infraestructura eficiente en costes. Es la única solución que no requiere, o requiere muy poca, instalación adicional y no hay necesidad de convertir las credenciales de sesión cuando se accede desde el dominio de cada estado miembro al dominio pan-Europeo.

La autenticación se lleva a cabo utilizando una implementación estándar de TLS y, en algunos casos, se puede utilizar como token de autenticación una I-Card (*Information Card*) [10]. Por otro lado, la implementación de TLS-Federation gira en torno a certificados basados en PKI y al uso de tarjetas inteligentes, por lo que se necesitan estándares adicionales como PKCS#11 y CSPs (*Cryptographic Service Providers*).

El aspecto central de TLS-Federation es la autenticación, estando menos desarrolladas la identificación y la autorización. En la actualidad, las investigaciones en curso han identificado dos vías diferentes para transportar atributos de usuario específicos de servicio, una de ellas se basa en el uso de certificados X.509 dinámicos, que pueden ser creados bajo demanda y de forma instantánea por un Proveedor de Identidad y que tienen un corto periodo de validez, y la otra se basa en la utilización de la tecnología I-Card.

Una I-Card es una pieza software que corresponde a una identidad digital de un usuario, es decir, es una colección de información de identidad de un usuario almacenada en un fichero. Al no ser información almacenada en una tarjeta física, no está limitada en cuanto a longitud, tamaño y capacidad y puede copiarse fácilmente si se desea. Adicionalmente, una I-Card puede, en vez de almacenar directamente los datos de identidad de un usuario, contener punteros a una localización central desde donde pueden ser recuperados (es el caso por ejemplo de una tercera parte de confianza que gestione la información presente en las I-Card de los usuarios).

Si se pone en marcha un sistema de autenticación fuerte y el uso de proveedores de identidad gubernamentales basados en PKI comienza a ser un estándar de-facto en los Estados Miembros Europeos, entonces TLS-Federation se podría llegar a utilizar para la autenticación a nivel pan-Europeo. Independientemente de que esto último ocurra, los aspectos relativos a la autenticación en TLS-Federation deben ser tenidos en cuenta porque no necesitan, o necesitan una cantidad limitada de trabajo de integración. Cada parte de la tecnología ya existe y es soportada por la mayoría de los sistemas operativos, buscadores y servidores web existentes en el mercado actual. Es simplemente una cuestión de activarlo.

El proyecto **GUIDE** (*Creating a European Identity Management Architecture for eGovernment*) [11] tuvo como propósito fundamental el desarrollo de un modelo para

la interoperabilidad en materia de identidad en la Unión Europea tal que permita a los Estados Miembros de la Unión Europea (UE) fiarse de la identidad de una entidad (tanto ciudadanos como empresas) de otro Estado. El concepto básico subyacente en GUIDE es el de una Red Federada de Gestión de Identidad en la que los miembros, usuarios, Administraciones y empresas, alrededor de la Unión Europea puedan participar en los intercambios de información de identidad sin comprometer la privacidad y la seguridad de dicha información. Esto requiere de la afiliación de los miembros en círculos de confianza basados en acuerdos operacionales que definen las relaciones de confianza entre ellos. En otras palabras, un círculo de confianza es una federación de proveedores de servicios y proveedores de identidad que han establecido relaciones formales y acuerdos operacionales y con cuyos consumidores de servicios pueden llevar a cabo transacciones. Dentro del paisaje de la Unión Europea ya se han desarrollado varias de estas federaciones y círculos de confianza para distintos grupos de interés, tanto administrativos como comerciales. En particular, muchos Estados Miembros están involucrados en el desarrollo de dichas federaciones a nivel nacional. Sin embargo, en muchos casos estas federaciones se han constituido o están siendo constituidas de forma aislada unas de las otras. El objetivo de GUIDE ha sido el de definir una arquitectura que permita la unión de dichas federaciones en un gran círculo de confianza con la intención de facilitar un entorno de identidad aparentemente único a lo largo de toda la Unión Europea. Esto es un prerrequisito anterior a que cualquier identidad pueda ser intercambiada de forma segura entre Estados Miembros. A este respecto GUIDE está concebido como una federación pan-Europea de federaciones de identidad que puede ser descrita también como una red de identidad. La esencia de esto es la provisión de un canal de confianza entre Estados Miembros para la realización de transacciones relacionadas con la identidad.

Otra propuesta de sistema de Gestión de Identidad a nivel pan-Europeo es **STORK** (*Secure idenTity acrOss boRders linKed*) [12]. Este proyecto, recientemente iniciado, trata de desarrollar y probar especificaciones comunes para un reconocimiento mutuo y seguro de las identidades electrónicas (eID) nacionales de los países participantes. Como objetivos concretos podemos encontrar el definir modelos y especificaciones comunes para el mutuo reconocimiento de eIDs entre países, probar en un entorno real soluciones de eID seguras y fáciles de utilizar tanto para ciudadanos como empresas e interactuar con otras iniciativas de la Unión Europea para maximizar la utilidad de los servicios de eID. Para conseguir lo anterior el modelo que STORK propone se basa en estudios previos realizados por IDABC [13] que describen, a alto nivel, un modelo federado para interoperabilidad, tecnológicamente neutral y que soporta múltiples niveles de autenticación. Concretamente, se presenta un modelo que confía en un proxy y requiere de la creación de proveedores de identidad a nivel nacional (al menos uno por país). Este sistema de proveedores de identidad se une en STORK a una red de proxys proveedores de servicios denominados PEPS (*Pan European Proxy Services*). Estos proxys serán creados a nivel nacional, aunque el modelo también contempla la posibilidad de que exista un proxy europeo centralizado o incluso un modelo mixto, de tal modo que ciertos países confían en un PEPS nacional mientras que otros lo hacen en un PEPS Europeo. Los PEPS sirven básicamente para superar el problema técnico que se presenta cuando aparece un conjunto amplio de soluciones de identificación/autorización, como es el caso del

escenario europeo. Por ejemplo, en algunos casos un ciudadano puede querer utilizar una combinación nombre de usuario-contraseña para acceder a un servicio mientras que otro pretende utilizar su tarjeta de identificación electrónica. Suponiendo que el dueño de la aplicación dé por buenos ambos métodos de identificación, la infraestructura técnica debe ser capaz de soportar ambas soluciones. Es aquí donde entra en escena el PEPS, cuya principal función es la de conectar los proveedores de servicios con los proveedores de identidad adecuados en cada país y validar la confianza y la seguridad de la información de identidad enviada por los proveedores de identidad. Por lo tanto, el conjunto de PEPS formarían un círculo de confianza en el sentido de, por ejemplo, las soluciones indicadas por Liberty Alliance [3]. Para el transporte de atributos de identidad desde los proveedores de identidad a los proveedores de servicio a través de los PEPS se sugiere el uso de asertos SAML. En lo referente al marco tecnológico elegido para interconectar las soluciones o el modelo conceptual detrás del marco de trabajo las decisiones no se han hecho públicas aún. Sin embargo, dentro de las intenciones del proyecto STORK se encuentra el ser lo más tecnológicamente transparente posible y el asegurar soluciones interoperables con los sistemas nacionales de eID existentes. Así mismo, STORK intenta confiar tanto como sea posible en estándares abiertos.

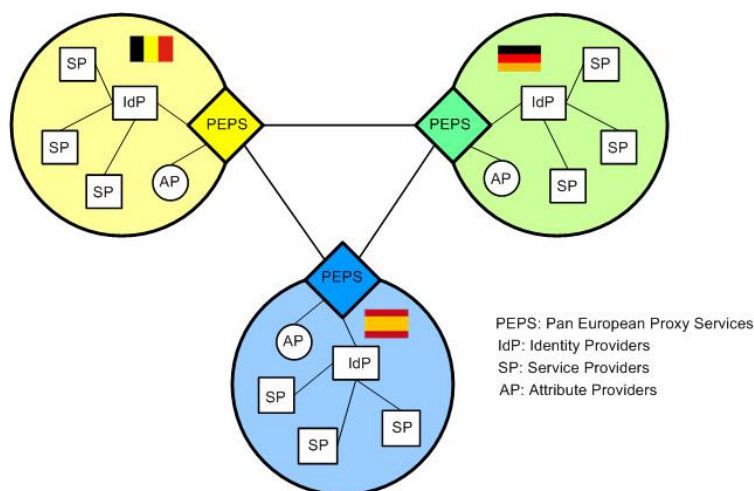


Fig. 1. Esquema de la arquitectura propuesta en el proyecto STORK

Tras un estudio pormenorizado de estos proyectos europeos, se ha realizado un análisis e identificación de las tendencias actuales en cuanto a los sistemas de gestión de identidad así como de las carencias y problemas que presentan.

Parece claro que, a día de hoy, los sistemas de gestión de identidad están orientados hacia la federación y el multinivel. Esta orientación no es ni mucho menos extraña si tenemos en cuenta que estos dos aspectos se derivaban directamente de los principios de diseño establecidos en el mapa de ruta europeo [7] tal y como vimos en el apartado 3.

Si se analiza el estado actual tanto desde el punto de vista de legislación como de implantación de sistemas de gestión de identidad en los distintos países miembros de la Unión Europea aparecen claras diferencias, existiendo países en los que la eID y el eIDM están claramente implantados y otros en los que todavía se está comenzando. No es válida por lo tanto una aproximación a la implantación de un sistema de gestión de identidad único y global, puesto que no todos los países parten del mismo punto y cada país presenta sus peculiaridades en cuanto al tratamiento de los datos y a la relación con los ciudadanos.

Un sistema federado que permita constituir un círculo de confianza entre las infraestructuras de todos los países miembros y que permita una fácil incorporación de futuros países se presenta como una buena solución, ya que permite la creación de la infraestructura pan-Europea sin apenas modificar lo ya implantado. Soluciones como la presentada en Modinis, basada en la existencia de portales de identidad en cada país encargados de la autenticación interna y la gestión de las confianzas depositadas en los sistemas de gestión de otros países, o la de STORK, centrada en el uso de proxys de acceso a servicios en cada país, se adaptan perfectamente a la federación y a la idea de mantener lo más intacto posible el sistema ya implantado en cada estado miembro.

Por otro lado, la heterogeneidad de los sistemas existentes y de los mecanismos de autenticación/autorización presentes en cada país, hace que sea fundamental dotar a un sistema de gestión de identidad pan-Europeo de la capacidad de mapear tokens de identidad entregados por el sistema de gestión de identidad de un país a sus equivalentes en otro si se pretende un acceso a los servicios lo más transparente posible por parte de los ciudadanos y usuarios en general. Esto se traduce en que el sistema debe ser necesariamente multinivel. Varios de los proyectos presentados cumplen con este criterio.

Otra de las tendencias detectadas es la utilización de estándares, concretamente la utilización de SAML, para el intercambio de tokens de autenticación entre sistemas de gestión de identidad. Muchos de los proyectos analizados, como por ejemplo STORK y GUIDE, se basan en el uso de SAML, llegando GUIDE a proponer modificaciones en el estándar para adaptarlo a su modelo. Evidentemente la utilización de soluciones estándar siempre es beneficiosa, pero antes de tomar una decisión clara se debe analizar la problemática y tratar de buscar la solución más adecuada. Una vez elegida una solución, se debe justificar su elección y dar solución a los problemas que se puedan derivar de la misma. Ninguno de los proyectos estudiados presenta un análisis claro y una justificación de la elección de SAML, adoleciendo por lo tanto de los problemas derivados del uso del mismo. Por ejemplo, STORK se basa en el uso de SAML para el intercambio de información entre sus PEPS y sufre, como consecuencia de ello, de la posibilidad de ataques del tipo *man-in-the-middle*. Aunque la tendencia actual es hacia la utilización de SAML, ésta presenta ciertos problemas y soluciones alternativas como la utilización de certificados X.509, generados de forma dinámica, propuesta por proyectos como TLS-Federation, deben ser tenidas en cuenta y evaluadas antes de tomar una decisión definitiva.

Además de las tendencias, se han identificado en los proyectos analizados diferentes problemas que se podrían considerar comunes a todos ellos. El primero aparece en la gestión de la información, concretamente en la fiabilidad y calidad de los datos manejados. Normalmente, en los sistemas de gestión de identidad actuales

no existe una única fuente de datos y esto puede dar lugar a problemas de duplicidad e incoherencias. De acuerdo al mapa de ruta, se debe disponer en cada país de una única fuente fiable para cada pieza de información correspondiente a una entidad registrada, de manera que se elimine la duplicidad de datos y se asegure que éstos son correctos y oficiales. Sólo el proyecto STORK considera la existencia de estas fuentes a través de los proveedores de atributos de cada país.

Por otra parte, todos los proyectos analizados presentan soluciones a la gestión de identidad a nivel pan-Europeo muy centradas en el entorno de la Administración Electrónica, del eGovernment. No se presenta en ninguno de los proyectos un estudio sobre la posibilidad de interoperabilidad con el entorno privado, ya sea como proveedores de infraestructura para la gestión de identidad o como proveedores de servicios, por ejemplo, un banco que permita el uso de la tarjeta de identificación electrónica para el acceso seguro a la aplicación de banca electrónica por parte de los usuarios. Esto se produce a pesar de que el mapa de ruta establece que el desarrollo de aplicaciones del sector privado que se apoyen en la infraestructura de gestión de identidad pública serán aspectos muy a tener en cuenta para facilitar e incrementar la aceptación y el atractivo de los sistemas de identificación electrónica por parte de los usuarios.

Finalmente, un problema común a todos los sistemas propuestos para la gestión de identidad, tanto a nivel nacional como a nivel pan-Europeo, es el de la delegación de identidad, la delegación de autorización y el manejo de roles. Muchas transacciones son realizadas a día de hoy por representantes legales autorizados a operar en nuestro nombre. Por ejemplo, en España un ciudadano puede autorizar a otra persona, típicamente a través de una Oficina Gestora, a realizar todos los trámites relativos a la presentación de impuestos anuales con la Administración. Así mismo, un ciudadano puede tener distintos roles de forma simultánea dentro del sistema de gestión de identidad, siendo a la vez ciudadano y representante legal de una empresa u organización. Los sistemas de gestión de identidad propuestos hasta la actualidad han hecho muy pocos progresos en estos aspectos y ninguno da un soporte completo al manejo de roles y delegaciones.

5 Propuesta de infraestructura de interoperabilidad

Dentro del mencionado proyecto de investigación en el que trabaja este grupo se ha analizado la problemática de la gestión de identidad en todos los niveles de la Administración Pública, desde el entorno local al pan-Europeo. Tras un estudio pormenorizado de dichos niveles se llegó a la conclusión de que los problemas de interoperabilidad que aparecen entre distintos países miembros de la Unión Europea en cuanto a gestión de identidad, aparecen en la interacción entre los distintos niveles de la Administración, incluso dentro del mismo país como por ejemplo España. La intención del grupo de investigación, y concretamente de los autores de este artículo, es proponer un modelo de interoperabilidad que pueda ser aplicado no solo a nivel pan-Europeo, sino también a nivel nacional, autonómico y local, de tal forma que la interoperabilidad en la gestión de la identidad esté garantizada en todos los niveles. El modelo toma como puntos de partida las ventajas identificadas en los sistemas de

gestión de identidad presentadas en el apartado anterior y pretende, adicionalmente, solucionar los problemas identificados.

A día de hoy podemos decir que el sistema estará basado en la federación, conseguida mediante el establecimiento de círculos de confianza en cada uno de los niveles y entre niveles, tal como se presenta en la figura 2. En dicha figura se puede apreciar cómo en cada uno de los niveles de la Administración (local, autonómico y nacional) se establece un círculo de confianza que agrupa a uno o varios Proveedores de Identidad (IdP) y Proveedores de Servicios (SP) en ese nivel. En lo que se refiere a la autenticación se hará uso del mapeo de tokens, es decir, se permitirá el mantenimiento de los sistemas ya establecidos de manera que los elementos de autenticación en uso serán mapeados a elementos comunes que garanticen la interoperabilidad. Para ello se adopta el concepto de proxy como interfaz entre los sistemas de gestión de identidad, de manera que garantizarán la interoperabilidad tanto dentro de un mismo nivel (por ejemplo la local entre dos ayuntamientos) como entre niveles (por ejemplo entre un ayuntamiento y su comunidad autónoma). Para el intercambio de tokens de autenticación se utilizarán los certificados X.509. Concretamente se utilizarán certificados generados de forma dinámica y se explotarán las capacidades de las extensiones, lo que nos ofrecerá la posibilidad, por ejemplo, de llevar a cabo la implantación de servicios con delegación de identidad y manejo de roles, hasta ahora apenas considerados en las arquitecturas propuestas.

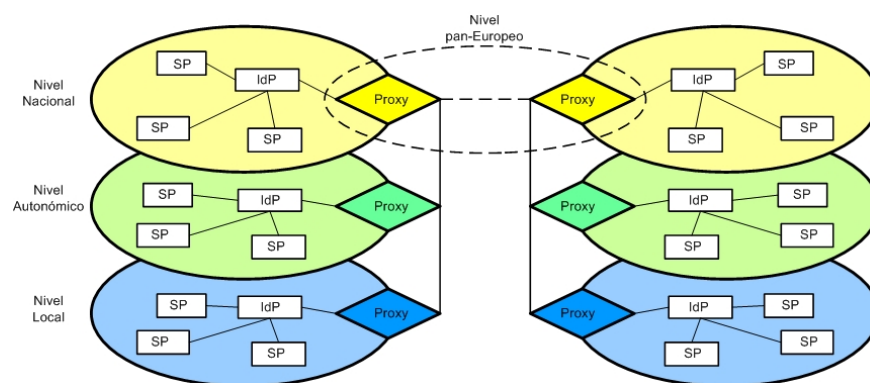


Fig. 2. Infraestructura propuesta

Por último, un problema aún en estudio en la propuesta es el de garantizar la unicidad y evitar incoherencias y duplicidad de datos. La intención es establecer fuentes de datos únicas en cada nivel y sincronizadas entre niveles, pero esto último presenta una serie de dificultades derivadas del hecho de que una misma entidad puede estar presente en distintos niveles y con datos que, a pesar de ser verdaderos, no guardan una relación directa, por lo que aparecen problemas a la hora de garantizar la sincronización.

En la actualidad se ha iniciado la implementación de un piloto de la infraestructura propuesta en el Ayuntamiento de Mejorada (Madrid), con la intención de demostrar la viabilidad de la solución y las posibilidades de la misma. En esta experiencia piloto se

implementa en un escenario real y, a escala reducida, un círculo de confianza con varios Proveedores de Servicios (SP) y un Proveedor de Identidad (IdP), empleando las herramientas ofrecidas por Shibboleth [14]. Para ello, se han seleccionado varios servicios telemáticos que próximamente proporcionará el ayuntamiento a través de Internet: solicitud de bonificaciones y exenciones tributarias, solicitud de autorizaciones (instalación de quioscos, terrazas, bares en la vía pública, grúas, etc.) y solicitud de licencias de actividad, de manera que cada uno de estos servicios se ofrece a través de un Proveedor de Servicios distinto, pero todos ellos dentro del mismo círculo de confianza. Para el acceso a estos servicios se contempla que la identificación de los ciudadanos se realice a través de un certificado digital X.509 integrado en el Documento Nacional de Identidad (eDNI) o emitido por una Autoridad de Certificación aceptada para operar dentro de la Administración española. La verificación de la identidad de los ciudadanos se lleva a cabo a través del Proveedor de Identidad de ese círculo de confianza, de manera que un ciudadano una vez autenticado para acceder a un servicio puede acceder a otro servicio del mismo círculo. Una vez creado el círculo de confianza se trabaja en la definición del proxy que se encargará del mapeo de tokens y permitirá que el Proveedor de Identidad pueda admitir como válidas identidades procedentes de otros Proveedores, de manera que se pueda probar la viabilidad de las soluciones propuestas.

7 Conclusiones

La búsqueda de soluciones a los problemas de interoperabilidad en la gestión de identidad es un tema muy importante que, sin duda, contribuirá a facilitar la vida a los ciudadanos en un mundo globalizado. Sin embargo, parece claro que estas soluciones no deberían obligar a modificar los esquemas nacionales de identificación digital desarrollados por cada país conforme a sus necesidades y legislación específicas, pero sí deberían ser lo suficientemente fiables y robustos para ser aceptados mayoritariamente por todos los países del entorno, en nuestro caso de Europa.

Aunque la UE ha puesto en marcha distintas iniciativas para buscar una solución a este problema creciente, a día de hoy existen importantes aspectos por resolver como la falta de integración de las soluciones con el sector privado, la ausencia de fuentes de datos únicas que garanticen la unicidad y coherencia de la información relativa a las entidades o los problemas derivados de la utilización de determinados estándares.

En esta línea, el proyecto de investigación que en la actualidad realiza este grupo está orientado a buscar solución a estos problemas bajo dos vertientes distintas. Por una parte, se pretende que las soluciones propuestas tengan eco a nivel europeo, para ello se mantiene contacto regular con los principales grupos que trabajan en este tema. Por otra, se busca extrapolar las soluciones planteadas al entorno de la Administración española, donde los problemas de interoperabilidad entre los sistemas de gestión de identidad en los distintos niveles de la Administración son semejantes a los existentes a nivel europeo. Con todo ello, se pretende realizar aportaciones que contribuyan a avanzar hacia una interoperabilidad total en la gestión de identidad, tanto a nivel nacional como pan-europeo, que permita una provisión de servicios sencilla y totalmente transparente para el usuario.

Bibliografía

1. Independent Centre for Privacy Protection; Unabhängiges Landeszentrum für Datenschutz; Schleswig-Holstein; Studio Notarile Genghini; Identity Management Systems (IMS): Identification and Comparison Study, http://www.datenschutzzentrum.de/idmanage/study/ICPP_SNG_IMS-Study.pdf (2003).
2. Parlamento Europeo y Consejo de la Unión Europea: Directiva 95/46/CE del Parlamento Europeo y del Consejo, de 24 de octubre de 1995, relativa a la protección de las personas físicas en lo que respecta al tratamiento de datos personales y a la libre circulación de estos datos. Diario Oficial de la Comunidad Europea, vol. L, 281, pp. 31-50. Luxemburgo: Unión Europea (23 de Noviembre de 1995).
3. Liberty Alliance Project, <http://www.projectliberty.org>.
4. Wason, T.: Liberty ID-FF Architecture Overview. Version: 1.2-errata-v1.0. Liberty Alliance Project (2005), <http://www.projectliberty.org/liberty/content/download/318/2366/file/draft-liberty-idff-arch-overview-1.2-errata-v1.0.pdf>.
5. The Modinis IDM Study Team: Modinis Study on Identity Management in eGovernment: Common Terminological Framework for Interoperable Electronic Identity Management. Version 2.01. eGovernment Unit, DG Information Society and Media, European Commission (23 de Noviembre de 2005), <https://www.cosic.esat.kuleuven.be/modinis-idm/twiki/bin/view.cgi/Main/GlossaryDoc>.
6. Varios: eGovernment: Commission calls for ambitious objectives in the EU for 2010. Bruselas (25 de Abril de 2006), <http://europa.eu/rapid/pressReleasesAction.do?reference=IP/06/523&format=PDF&aged=1&language=EN&guiLanguage=en>.
7. Varios: A Roadmap for a pan-European eIDM Framework by 2010. Version 1.0. European Commission, Information Society and Media Directorate-General, eGovernment Unit, http://ec.europa.eu/information_society/activities/egovernment/docs/pdf/eidm_roadmap_paper.pdf.
8. ModinisIDM, <https://www.cosic.esat.kuleuven.be/modinis-idm/twiki/bin/view.cgi/Main/WebHome>.
9. Bruegger, B. P., Hühnlein, D., Schwenk, J.: TLS-Federation - a Secure and Relying-Party-Friendly Approach for Federated Identity Management. http://porvoo14.dvla.gov.uk/documents/tls_federation_final.pdf.
10. Microsoft Corporation. A technical reference for InfoCard v1.0 in Windows (Agosto 2005), <http://download.microsoft.com/download/5/4/0/54091e0b-464c-4961-a934-d47f91b66228/infocard-techref-beta2-published.pdf>.
11. GUIDE, Creating a European Identity Management Architecture for eGovernment, <http://istrig.som.surrey.ac.uk/projects/guide/overview.html>.
12. STORK, Secure idenTity acrOss boRders linked, <http://www.eid-stork.eu/>.
13. Majava, J. and Graux, H.: Common Specifications for eID interoperability in the eGovernment context. IDABC European eGovernment Services (Diciembre 2007), <http://ec.europa.eu/idabc/servlets/Doc?id=30989>.
14. Shibboleth, <http://shibboleth.internet2.edu/>.

MGSM-PYME: Metodología para la gestión de la seguridad y su madurez en las PYMES

Luís Enrique Sánchez¹, Daniel Villafranca¹, Eduardo Fernández-Medina² y Mario Piattini²

¹SICAMAN Nuevas Tecnologías. Departamento de I+D,
Juan José Rodrigo, 4. Tomelloso, Ciudad Real, España.

{dvillafranca, lesanchez} @sicaman-nt.com

²Grupo de Investigación ALARCOS. Universidad de Castilla-La Mancha
Paseo de la Universidad, 4 – 13071 Ciudad Real, Spain

{Eduardo.FdezMedina, Mario.Piattini}@uclm.es

Resumen. Con la creciente dependencia que la sociedad de la información tiene de las Tecnologías de la Información y las Comunicaciones (TIC), la necesidad de proteger la información tiene cada vez mayor importancia para las empresas. En este contexto, surgen los Sistemas de Gestión de la Seguridad de la Información (SGSI), que tienen una gran importancia para la estabilidad de los sistemas de información de las compañías. El hecho de poder disponer de estos sistemas ha llegado a ser cada vez más vital para la evolución de las PYMES. En este artículo mostramos la metodología desarrollada para el desarrollo, implantación y mantenimiento de un sistema de gestión de seguridad, adaptada a las necesidades y recursos de los que disponen las PYMES. Este enfoque está siendo aplicado directamente a casos reales, consiguiendo así una constante mejora en su aplicación.

Palabras clave: SGSI, PYMES, Gestión de la Seguridad, Niveles de Madurez

1 Introducción

En un entorno empresarial globalizado y competitivo como el existente en la actualidad, las compañías dependen cada vez más de sus sistemas de información, pues se ha demostrado que tienen una enorme influencia para aumentar su nivel de competitividad. Pero sin una adecuada gestión de la seguridad estos sistemas de información carecen de valor real, ya que no pueden aportar las suficientes garantías de continuidad a las empresas. Por ello, las compañías empiezan a tener conciencia de la enorme importancia que tiene el poseer unos sistemas de seguridad de la información adecuados, así como una correcta gestión de los mismos. De esta forma, pese a que muchas empresas todavía asumen el riesgo de prescindir de las medidas de protección adecuadas, otras muchas han comprendido que los sistemas de información no son útiles sin los sistemas de gestión de seguridad y las medidas de protección asociados a ellos.

Gran parte de este cambio de mentalidad en las empresas tiene su origen en el cambio social producido por Internet y la rapidez en el intercambio de información, que ha dado lugar a que las empresas empiecen a tomar conciencia del valor que tiene la información para sus organizaciones y se preocupen de proteger sus datos. De esta forma, las empresas ya han tomado conciencia de que la información y los procesos que apoyan los sistemas y las redes son sus activos más importantes [1]. Estos activos están sometidos a riesgos de una gran variedad, que pueden afectar de forma crítica a la empresa. Así, la importancia de la seguridad en los sistemas de información viene avalada por numerosos trabajos [2, 3], por citar sólo algunos.

Anteriormente, las compañías que habían decidido proteger sus sistemas de información habían afrontado proyectos desde la perspectiva de que la seguridad era algo individual que afectaba a un objeto pero no al conjunto al que pertenecía el objeto, es decir, se basaban en la implantación de medidas de seguridad, pero sin llevar a cabo una adecuada gestión de dichas medidas [4]. Con el tiempo, al no disponer de una gestión adecuada, los controles implantados dejaban de mantenerse y se convertían en controles pasivos, que en lugar de ayudar a mejorar la seguridad contribuían a desinformar, ofreciendo información errónea en muchos casos. Así, en [5] se destaca que para la construcción de un sistema de seguridad no bastan los aspectos tecnológicos, sino que también son necesarios los aspectos de gestión, así como los aspectos legales y éticos.

Actualmente, los expertos consideran que la seguridad en los sistemas de información tiene un carácter bidimensional [6]. La seguridad en los sistemas de información ya no se trata como un aspecto exclusivamente técnico, en el que el uso correcto de ciertos mecanismos de seguridad (ej.: protocolos de seguridad, esquemas de cifrado, etc.) garantiza la seguridad de un sistema en términos absolutos. Además, y dada la integración social de los sistemas software, existe una nueva dimensión que cobra gran relevancia y que debe ser analizada detenidamente. Esta nueva dimensión tiene un carácter eminentemente social y organizativo y está ligada a la cada vez mayor interacción de los humanos con los sistemas de información seguros. Resultados de investigaciones han demostrado que el factor humano posee un impacto significativo en la seguridad [7].

El problema de la seguridad de la información se caracteriza por la complejidad y la interdependencia. La gestión de la seguridad contiene un número importante de factores y elementos que se interrelacionan entre sí. Las PYMES en los países desarrollados suelen tener una débil comprensión de la seguridad de la información, tecnologías de seguridad y medidas de control, y suelen dejar el análisis de riesgos o el desarrollo de las políticas de seguridad olvidados [8]. Esto puede deberse a que las PYMES carecen de los recursos, tiempo y conocimientos especializados para coordinar la seguridad de la información u ofrecer información adecuada sobre la seguridad, la formación y la educación [8]. Sin embargo, la literatura sugiere una explicación muy diferente. [9] dicen que las PYMES no quieren pagar por la seguridad, y que prefieren mantener una seguridad física con la que están familiarizados. [8] señalan que, al carecer de un conocimiento especializado de tecnologías de seguridad, las PYMES suelen mantener la seguridad con las tecnologías con las que ya están familiarizados. Así mismo, las PYMES no ven la seguridad vinculada a la estrategia empresarial, lo que impacta directamente en el cumplimiento de la misma [10]. De hecho, una investigación reciente pone de

manifiesto la necesidad de vincular la seguridad de la información con los sistemas de información de planificación estratégica y, por tanto, con los objetivos de la empresa [11].

A pesar de que existen numerosos estándares de seguridad en las TIC, como el código de buenas prácticas [12], metodologías para la gestión de la seguridad como COBIT [13], o para el análisis y la gestión de riesgos como Magerit [14], e incluso modelos de madurez para la gestión de la seguridad de los sistemas de información como SSE-CMM [15], éstos suelen estar diseñados para grandes corporaciones, son muy rígidos y su aplicación práctica en pequeñas y medianas empresas requiere de mucho tiempo y suele ser muy costosa. Estos motivos hacen que muchas compañías ofrezcan resistencia a la implantación de técnicas de gestión de seguridad adecuadas, asumiendo de este modo unos riesgos de seguridad, y por tanto de pérdida de competitividad, que resultan inaceptables en la empresa moderna.

En numerosas fuentes bibliográficas se detecta y resalta la dificultad que supone para las PYMES la utilización de las metodologías y modelos de madurez para la gestión de la seguridad tradicionales, que han sido concebidos para grandes empresas [16-19]. Se justifica en repetidas ocasiones que la aplicación de este tipo de metodologías y modelos de madurez para las PYMES es difícil y costosa.

En el presente artículo describimos la metodología que hemos desarrollado para la gestión de la seguridad en las PYMES, que pretende solucionar los problemas detectados en las metodologías clásicas, las cuales no se están mostrando eficientes a la hora de su implantación en las PYMES debido a su complejidad y otra serie de factores que serán analizados con detalle en las siguientes secciones del artículo.

El artículo continúa en la Sección 2, describiendo brevemente las metodologías y modelos para la gestión de la seguridad existentes y su tendencia actual. En la Sección 3 se introduce nuestra propuesta de metodología para la gestión de la seguridad orientado hacia las PYMES. En la Sección 4 se muestra la herramienta desarrollada para dar soporte a la metodología. Finalmente, en la Sección 5 concluimos indicando cuál será el trabajo que desarrollaremos en el futuro.

2 Trabajo relacionado

En los últimos años han aparecido un gran número de procesos, marcos de trabajo y métodos para la gestión de la seguridad de la información, cuya necesidad de implantación está siendo cada vez más reconocida y considerada por las organizaciones, pero que como se ha mostrado, son ineficientes para el caso de las PYMES.

Entre ellas destaca el modelo presentado en la ISO/IEC27001 [20], el de COBIT [13] y el modelo de madurez de la gestión de la seguridad de la información [21]. [22] y [23] investigan la coexistencia y uso complementario de COBIT [13] y la ISO/IEC17799 [24] desarrollando un mapa para la sincronización de ambos marcos de referencia. Algunos de los detractores de la ISO/IEC17799 [24] presentan como desventaja que es una guía de soporte, pero que no alcanza todo el marco necesario para el gobierno de las tecnologías de la información. Su principal ventaja frente a COBIT [13] es que es más detallada y tiene más guías orientadas a cómo deben

hacerse las cosas. Un reciente informe del Instituto para el gobierno de la tecnología de la información (ITGI) soluciona el problema de sincronización desarrollando el mapa entre COBIT [13] y la ISO/IEC17799 [24].

Siguiendo esta “filosofía” también se han propuesto otros muchos modelos de madurez más específicos: para gestión de proyectos [25], ingeniería de requisitos [26], desarrollo distribuido [27], mantenimiento [28], outsourcing [29], arquitecturas [30-33], seguridad [15], servicios e-Government [34], etc.

En numerosas fuentes bibliográficas se detecta y resalta la dificultad que supone para las PYMES la utilización de las metodologías y modelos de madurez para la gestión de la seguridad tradicionales, que han sido concebidos para grandes empresas [18]. Se justifica en repetidas ocasiones que la aplicación de este tipo de metodologías y modelos de madurez para las PYMES es difícil y costosa. Además, las organizaciones, incluso las grandes, tienden más a adoptar grupos de procesos relacionados como un conjunto que a tratar los procesos de forma independiente [35].

El problema principal de todos los modelos de gestión de la seguridad y su madurez presentados es que no están teniendo éxito a la hora de implantarse en PYMES, debido principalmente a que:

- Unos fueron desarrollados pensando en organizaciones grandes (ISO/IEC27001, COBIT) y en las estructuras organizativas asociadas a éstas.
- Otros (ISM3, etc) han intentado centrarse en los problemas de las PYMES, pero son modelos incompletos que sólo afrontan parte del problema, o intentan aportar unas guías básicas de los pasos a realizar, pero sin entrar en cómo gestionar realmente el SGSI. Además, la mayoría son modelos teóricos y están todavía en desarrollo.

3 MGSM-PYME: Metodología para los SGSI en las PYMES

La metodología para la gestión de la seguridad y su madurez en las PYMES que se ha desarrollado, permite a cualquier organización gestionar, evaluar y medir la seguridad de sus sistemas de información, pero está orientado principalmente a las PYMES, ya que son las que tiene mayor tasa de fracaso en la implantación de las metodologías de gestión de la seguridad existentes.

Uno de los objetivos perseguidos en la metodología MGSM-PYME es que sea sencilla de aplicar, y que el modelo desarrollado sobre ella permita obtener el mayor nivel de automatización posible con una información mínima, recogida en un tiempo muy reducido. En la metodología se ha priorizado la rapidez y el ahorro de costes, sacrificando para ello la precisión que ofrecían otras metodologías. Es decir, la metodología desarrollada pretende generar una de las mejores configuraciones de seguridad pero no la óptima, priorizando los tiempos y el ahorro de costes frente a la precisión, aunque garantizando que los resultados obtenidos tengan la calidad suficiente.

Otra de las principales aportaciones que presenta la metodología que se ha desarrollado es un conjunto de matrices que permiten relacionar los diferentes componentes del SGSI (controles, activos, amenazas, vulnerabilidades, criterios de riesgo, procedimientos, registros, plantillas, instrucciones técnicas, reglamentos y

métricas) y que el modelo utilizará, para generar de forma automática gran parte de la información necesaria, reduciendo de forma muy notable los tiempos necesarios para el desarrollo e implantación del SGSI. Este conjunto de interrelaciones entre todos los componentes del SGSI, permite que el cambio de cualquiera de esos objetos altere el valor de medición del resto de objetos de los que se compone el modelo, de forma que se pueda tener en todo momento una valoración actualizada de cómo evoluciona el sistema de seguridad de la compañía.

De esta forma y a partir de la información obtenida mediante la implantación en diferentes empresas, se ha desarrollado una metodología de gestión y madurez de la seguridad de los sistemas de información y un modelo asociado a la misma (ver Fig. 1).

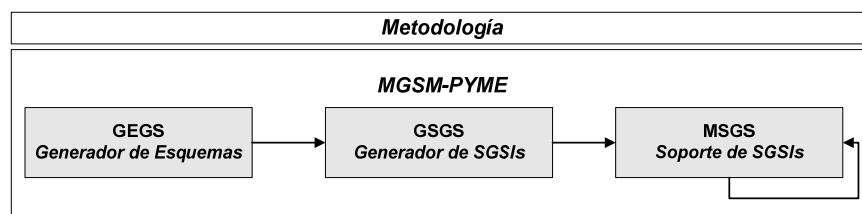


Fig. 1. Subprocesos de la metodología.

Esta metodología consta de tres subprocesos principales:

- GEGS – Generación de Esquemas de Gestión de Seguridad.
- GSGS – Generación de Sistemas de Gestión de Seguridad.
- MSGS – Mantenimiento del Sistema de Gestión de Seguridad.

La generación de esquemas es una labor que será realizada por los expertos en seguridad y aunque su elaboración es un proceso costoso, se ve compensado por las enormes reducciones de costes que produce en los otros subprocesos al poder ser reutilizado por compañías con características parecidas (mismo sector y mismo tamaño).

3.1 GEGS – Generación de Esquemas de Gestión de Seguridad.

La Generación de Esquemas para la Gestión de la Seguridad (GEGS), es el primer subproceso de la metodología MGSM-PYME y su objetivo principal es producir un esquema que contenga todas las estructuras necesarias para generar un SGSI y aquellas relaciones que puedan establecerse entre ellas para un determinado tipo de compañías (del mismo sector y tamaño), con el objetivo de ahorrar tiempo y recursos a la hora de generar un SGSI para una compañía que comparta la misma características que aquellas para las que fue creado el esquema. En la Fig. 2 se pueden ver con detalle los diferentes objetos que componen el esquema.

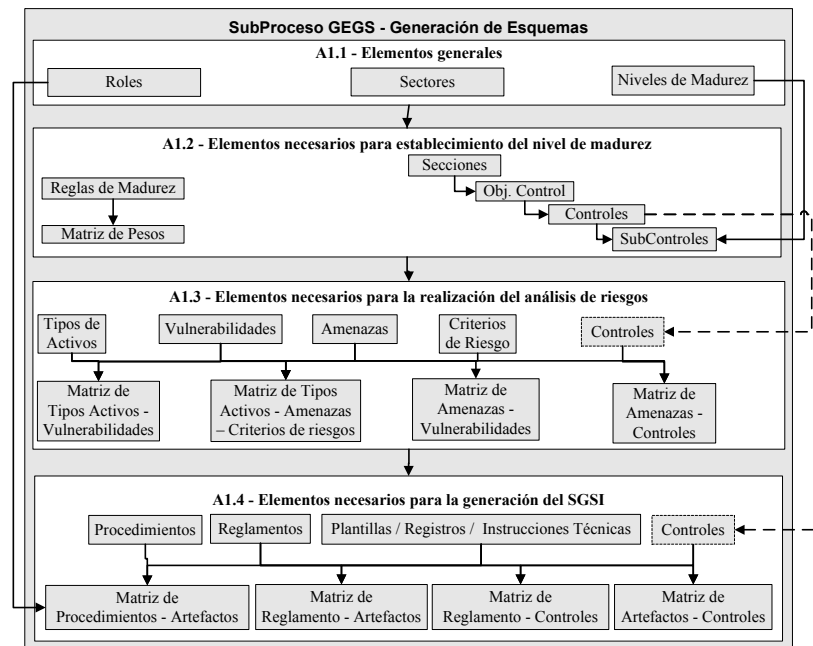


Fig. 2. Elementos del subproceso GEGS.

El subproceso GEGS para la generación de esquemas se compone básicamente de las siguientes actividades:

- A1.1 – Generación de tablas maestras: Se establecen las tablas de configuración iniciales, que contendrán: i) los roles de usuarios del sistema de información que podrán intervenir en el sistema; ii) los diferentes sectores empresariales a los que puede pertenecer la compañía; y iii) los niveles de madurez sobre los que podrá evolucionar el SGSI a lo largo de su ciclo de vida.
- A1.2 – Generación de tablas del nivel de madurez: Se seleccionarán las reglas de madurez que permitirán determinar el nivel de madurez actual del SGSI de la compañía y la lista de controles que se podrán establecer. Los controles se dividirán en subcontroles para poder aproximar el nivel en que estos se cumplen actualmente con mayor precisión. Estos subcontroles tendrán asociados los niveles de madurez definidos en la actividad anterior.
- A1.3 – Generación de tablas del análisis de riesgos: Se selecciona el listado de elementos de los artefactos asociados al análisis de riesgos, así como las relaciones existentes entre ellos.
- A1.4 – Generación de tablas de la biblioteca de artefactos: Se selecciona el listado de elementos de los artefactos asociados a la generación del SGSI, así como las relaciones existentes entre ellos.

Existe una dependencia en la actividad A1.2 ya que requiere una entrada de la actividad A1.1. De igual forma las actividades A1.3 y A1.4 requieren de un elemento de entrada generado durante la actividad A1.2, pero no tienen dependencia entre ellas.

Este subproceso recibirá como entradas:

- Conocimiento de los expertos adquirido durante otras implantaciones de SGSIs. (ej.: relaciones entre elementos, procedimientos, etc).
- Listados de elementos procedentes de otras normas, guías de buenas prácticas (ej.: ISO/IEC27002) o metodologías (ej.: Magerit v2).

Y generará como salida un esquema, que será utilizado por los siguientes subprocesos y que se componen básicamente de los siguientes elementos:

- Un subconjunto de elementos seleccionados de las listas de entrada.
- Una serie de matrices que relacionan entre sí los principales elementos (controles, tipos de activos, vulnerabilidades, amenazas y criterios de riesgo) necesarios para la elaboración de un análisis de riesgos.
- Una serie de matrices que relacionan entre sí los principales elementos (controles, procedimientos, reglamentos, plantillas, registros, instrucciones técnicas) necesarios para la generación del SGSI.

Todo este conjunto de artefactos, necesario para poder generar el sistema de gestión del sistema de información de la compañía, son incluidos en el repositorio de esquemas para el SGSI que se actualiza constantemente con el nuevo conocimiento obtenido en cada nueva implantación.

Debido a la complejidad del desarrollo de un esquema y como parte de la investigación se ha desarrollado un esquema inicial, denominado esquema base (EB), obtenido a partir del conocimiento adquirido durante el proceso de investigación, con el objetivo de posibilitar la creación de nuevos esquemas mediante un proceso de clonación (generar un nuevo esquema a partir de un esquema existente) del esquema base y realizando posteriormente los ajustes necesarios en el nuevo esquema para adecuarlo al tipo de compañías deseada.

3.2 GSGS – Generación del Sistema de Gestión de Seguridad

La Generación del Sistema de Gestión de Seguridad (GSGS), es el segundo subproceso y tiene como principal objetivo la generación del SGSI mediante la selección del esquema más adecuado para el tipo de compañía y la solicitud de información empresarial y técnica a la empresa por medio de un interlocutor (Int), designado por la misma. En la Fig. 3 se pueden ver los diferentes objetos que componen este subproceso.

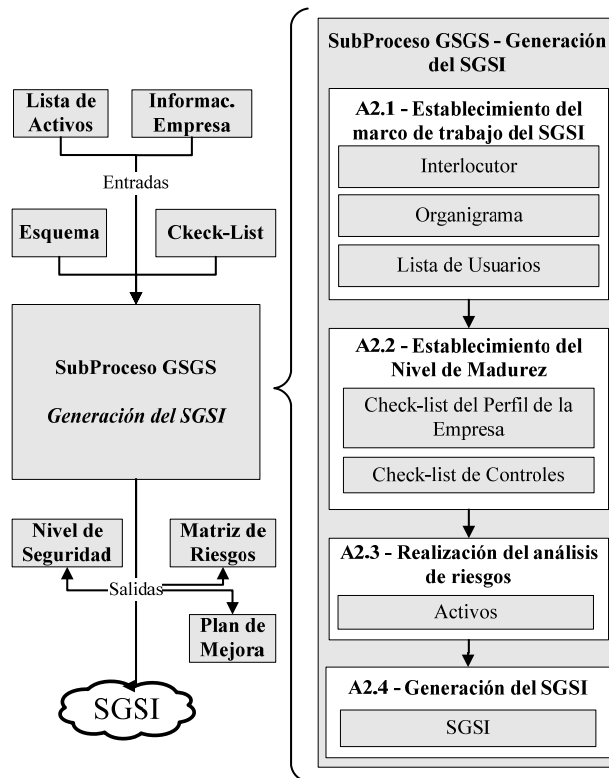


Fig. 3. Elementos del subproceso GSGS.

El subproceso GSGS para la generación del SGSI se compone básicamente de las siguientes actividades:

- A2.1 – Establecimiento del marco de trabajo del SGSI: Se establecerán las relaciones con la compañía, definiendo el interlocutor válido y solicitando información de la misma: i) organigrama de la compañía; ii) usuarios con acceso al sistema de información y roles que desempeñan.
- A2.2 – Establecimiento del nivel de madurez:
 - Se solicita mediante una entrevista de carácter empresarial, información relacionada con la compañía (número de empleados, facturación, etc) con el objetivo de determinar el esquema que más se adecua a ese tipo de compañía de los existentes en el repositorio de esquemas.
 - Se realizará una segunda entrevista de carácter técnico, para determinar con detalle la situación actual de la compañía con respecto a la gestión de la seguridad de su sistema de información.

- A2.3 – Realización del análisis de riesgos: Se identificarán un conjunto de activos básicos de grano grueso, determinando el coste (cualitativo y cuantitativo) que tendría para la organización su pérdida. A partir del conjunto de activos se determinarán los riesgos de seguridad a que están sometidos y se generará un plan para mitigar los riesgos de seguridad de una forma eficiente.
- A2.4 – Generación del SGSI: A partir de toda la información obtenida y del esquema seleccionado, se generan los elementos que conformarán el SGSI para la compañía y se procederá a la implantación del mismo en la compañía.

Este subproceso recibirá las siguientes entradas:

- Información de la compañía sobre la que se desea realizar el SGSI: i) información empresarial; ii) interlocutor válido para el desarrollo del SGSI; iii) organigrama de la compañía; iv) el listado de usuarios y los roles que desempeñan dentro del sistema de información de la compañía.
- El esquema más adecuado para generar el SGSI a partir del perfil empresarial de la compañía y del repositorio de esquemas.
- Dos listas de verificación: i) una lista de verificación con información del negocio; ii) una lista de verificación con información sobre el nivel de la gestión de la seguridad.
- Un listado de los activos asociados al sistema de información de la compañía, intentando agruparlos en el menor número de activos posibles (grano grueso), para reducir el coste de generación y gestión del sistema de información.

Y generará las siguientes salidas, que contienen una descripción completa del SGSI de la compañía:

- El nivel de madurez actual de la compañía con respecto a su sistema de gestión de seguridad de la información y hasta qué nivel de madurez debería progresar.
- Una matriz con los riesgos a los que están sometidos los activos de la compañía.
- Un plan de mejora ordenado, que indica qué controles deben ser reforzados para que el nivel de seguridad de la compañía evolucione lo más rápido posible.
- Un conjunto de elementos que componen el SGSI de la compañía, que incluyen: i) un cuadro de mandos que indicara el nivel de seguridad para cada control relacionado con la gestión de la seguridad; ii) un conjunto de reglamentos, plantillas e instrucciones técnicas válidos para esa compañía en el momento actual; iii) un conjunto de métricas; iv) un conjunto de usuarios, asociados a roles que permitirán ejecutar una serie de procedimientos para interactuar con el sistema de información; v) y un conjunto de reglamentos que se deben cumplir para el funcionamiento del SGSI.

Todo este conjunto de objetos, que componen el SGSI, son incluidos en el repositorio de SGSIs y serán utilizados por la compañía para poder gestionar correctamente la seguridad del sistema de información.

3.3 MSGS – Mantenimiento del Sistema de Gestión de Seguridad.

El Mantenimiento del Sistema de Gestión de Seguridad (MSGs) es el tercer subproceso definido en MGSM–PYME y su principal propósito es permitir realizar el conjunto de tareas necesarias para poder trabajar con el SGSI, poder medir la evolución del mismo y facilitar la obtención de conocimiento para la mejora continua de los esquemas y los SGSI generados. En la Fig. 4 se pueden ver los diferentes objetos que componen este subproceso.

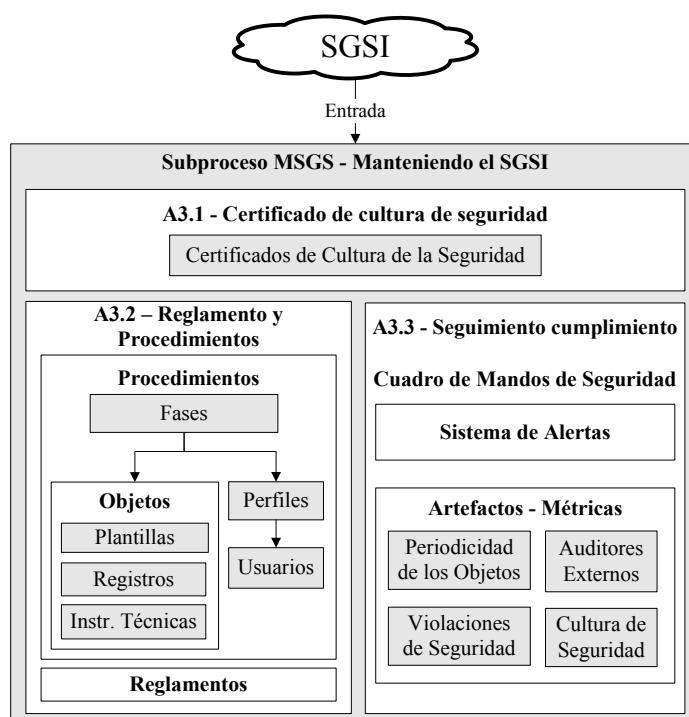


Fig. 4. Elementos del subproceso MSGS.

El subproceso MSGS para el mantenimiento del SGSI se compone básicamente de las siguientes actividades:

- A3.1 – Obtener o renovar el certificado de cultura de seguridad: Se establecerá un sistema que permita crear de forma progresiva una conciencia de seguridad entre los usuarios del sistema de información, que garantice la calidad del mismo.
- A3.2 – Ejecutar procedimientos del SGSI: Se ejecutarán procedimientos generales y específicos (ej.: procedimiento de denuncia) que permitirán mantener actualizado el SGSI de la compañía.

- A3.3 – Seguimiento del cumplimiento del SGSI: Se dispondrá de un conjunto de métricas para mantener actualizado de forma dinámica y en tiempo real el cuadro de mandos de seguridad de la compañía, de forma que el responsable de seguridad podrá tomar decisiones sin necesidad de esperar a la realización de una auditoría externa.

Este subproceso recibirá las siguientes entradas, procedentes del subproceso anterior:

- Un conjunto de usuarios y los roles que desempeñarán dentro del sistema de información, que determinarán a qué procedimientos tienen acceso.
- Un conjunto de reglamentos que se deben cumplir para el buen funcionamiento del SGSI.
- Un conjunto de procedimientos de seguridad, y los elementos (plantillas, registros, instrucciones técnicas) asociados a los mismos.
- Un cuadro de mandos que indicara el nivel de seguridad para cada control relacionado con la gestión de seguridad de la compañía.

Y generará las siguientes salidas:

- Una serie de instancias de los procedimientos existentes, que se irán ejecutando a lo largo del tiempo y que permitirán gestionar y mantener el SGSI de la compañía.
- Un conjunto de métricas que permitirán mantener actualizado el cuadro de mandos asociado al nivel de seguridad del sistema de gestión de seguridad de la información: i) un conjunto de métricas generales; ii) métricas específicas: periodicidad de los objetos, violaciones de seguridad, conciencia de seguridad y auditoras externas.
- Estadísticas extraídas del uso diario del SGSI por los usuarios del sistema de información, que se convertirá en conocimiento para que los expertos en seguridad puedan elaborar nuevos esquemas y refinar los existentes.

Toda la información de salida generada durante la vida útil del SGSI, será incluida en el repositorio de información del SGSI y será utilizada por la compañía para poder gestionar correctamente la seguridad del sistema de información y por el grupo de expertos de seguridad para mejorar los esquemas del subproceso GEGS.

4 MGSM-TOOL: Automatización de la metodología.

Para validar la metodología MGSM-PYME, se ha desarrollado una herramienta denominada MGSM-TOOL que permite desarrollar modelos de gestión de seguridad sencillos, económicos, rápidos, automatizados, progresivos y sostenibles, que son los principales requerimientos que tienen este tipo de compañías a la hora de implantar estos modelos.

Desde el punto de vista de usuario la herramienta presenta dos ventajas claras:

- Simplicidad: Todas las actividades del SGSI se han orientado a reducir la complejidad del proceso de construcción y mantenimiento de un SGSI, pensando en organizaciones (PYMES) cuyas estructuras organizativas son muy sencillas.
- Automatización: Todo el sistema utiliza un concepto denominado esquemas para poder automatizar los pasos necesarios para construir y mantener el SGSI de la compañía.

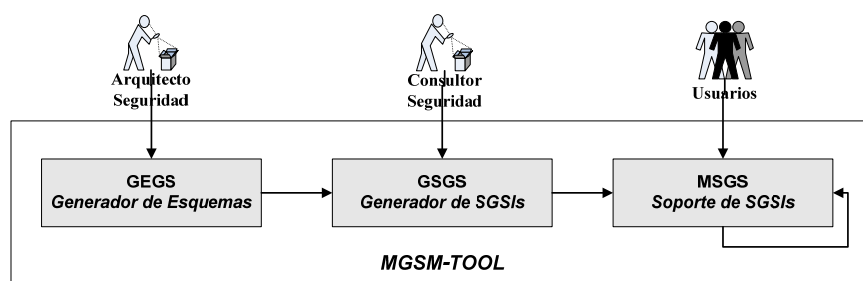


Fig. 5. Zonas de la aplicación MGSM-TOOL.

La herramienta se compone de tres partes claramente diferenciadas que se pueden ver en la Fig. 5, y que se corresponden con los subprocesos de la metodología:

- Generador de Esquemas (GEGS): Esta zona de la herramienta es sólo accesible por el arquitecto de gestión de la seguridad (AGS) y el grupo de expertos del dominio (GED), y desde ella pueden realizar tres operaciones básicas: i) Crear nuevos esquemas; ii) Clonar esquemas a partir de un esquema existente; y iii) Modificar esquemas para mejorar la generación del SGSI.
- Generador del SGSI (GSGS): Esta zona de la herramienta es sólo accesible por el consultor de seguridad (CoS) y su objetivo es generar el SGSI para la compañía.
- Soporte del SGSI (MSGS): Esta zona de la herramienta es accesible por los usuarios del sistema de información. El perfil que más relevancia tiene dentro de esta zona es el responsable de seguridad (RS). Desde ella se pueden realizar tres operaciones básicas: i) Gestión de los certificados de cultura de la seguridad; ii) Gestión de procedimientos; y iii) Gestión del cuadro de mandos.

Los esquemas son el núcleo sobre el que se desarrolla la herramienta, ya que permiten la automatización de los SGSIs. Estos esquemas están formados por un conjunto de elementos y asociaciones entre ellos, definidos a partir del conocimiento adquirido por los clientes.

La herramienta ha permitido reducir los costes de implantación de los sistemas y supone un mayor porcentaje de éxito de las implantaciones en las PYMES. Por estas razones, se considera que los resultados de esta investigación pueden ser muy positivos para las PYMES, ya que les permite acceder al uso de sistemas de gestión de la seguridad con un coste de recursos razonable para su tamaño. Además, mediante

el uso de la metodología y la herramienta que la soporte, se pueden obtener resultados a corto plazo y reducir los costes que supone el uso de otros modelos y herramientas, consiguiendo un mayor grado de satisfacción y eficiencia en la empresa.

Adicionalmente, la herramienta permite mantener repositorios con información acerca de las especificaciones de los esquemas necesarios para construir los SGSI y con información de los resultados obtenidos en los diferentes casos de uso, lo que permite mejorar constantemente la metodología y los modelos.

5 Conclusiones y futuros trabajos.

En este artículo se ha presentado la propuesta de una nueva metodología para la gestión de la seguridad y su madurez en las PYMES. Esta metodología permite a las PYMES desarrollar y mantener un SGSI con un coste en recursos aceptable para este tipo de compañías.

Para demostrar la validez de la metodología se ha definido un modelo (esquema base) que permite soportar los resultados generados durante la investigación y que cumple con los objetivos perseguidos.

Se ha definido cómo se debe utilizar esta metodología y las mejoras que ofrece con respecto a otras metodologías que afrontan el problema de forma parcial, o de manera demasiado costosa para las PYMES.

Las características ofrecidas por la nueva metodología y su orientación a las PYMES ha sido muy bien recibida, y su aplicación está resultando muy positiva ya que permite acceder a este tipo de empresas al uso de sistemas de gestión de seguridad de la información, algo que hasta ahora había estado reservado a grandes compañías. Además, con esta metodología se obtienen resultados a corto plazo y se reducen los costes que supone el uso de otras metodologías, consiguiendo un mayor grado de satisfacción de la empresa.

Finalmente, se considera que el trabajo realizado debe ser ampliado con nuevas especificaciones, nuevos esquemas, ampliando el conjunto de artefactos de la biblioteca y profundizando en el modelo con nuevos casos de ejemplo.

Entre las mejoras del modelo sobre las que se está trabajando de cara al futuro destacan:

- Mejoras asociadas al subproceso GEGS: Adaptación de los esquemas predefinidos para PYMES con las nuevas normas y estándares que surjan asociados a la gestión de la seguridad.
- Mejoras asociadas al subproceso GSGS: Revisar aspectos relacionados con la generación del SGSI.
- Mejoras asociadas al subproceso MSGS: Mejorar y aumentar los mecanismos de medición y auto-evaluación de la seguridad introduciendo nuevas métricas en el modelo que permitan conocer en todo momento el nivel de la seguridad, minimizando el número de auditorías de auto-ajuste necesarias para mantener actualizado dicho nivel de seguridad.

Todas las mejoras futuras de la metodología y el modelo se están orientando a mejorar la precisión del mismo, pero siempre respetando el principio de coste de

recursos, es decir, se busca mejorar el modelo sin incurrir en costes de generación y mantenimiento del SGSI.

Mediante el método de investigación “investigación en acción”, con la ayuda de la retroalimentación obtenida directamente de nuestros clientes, esperamos conseguir una mejora continua de estas implantaciones.

Agradecimientos

Esta investigación es parte de los proyectos ESFINGE (TIN2006-15175-C05-05), concedido por el Ministerio de Educación y Ciencia y QUASIMODO (PAC08-0157-0668) financiado por la Consejería de Educación y Ciencia de la Junta de Comunidades de Castilla-La Mancha.

Referencias

1. Dhillon, G. and J. Backhouse, *Information System Security Management in the New Millennium*. Communications of the ACM, 2000. 43(7): p. 125-128.
2. Masacci, F., M. Prest, and N. Zannone, *Using a security requirements engineering methodology in practice: The compilanse with the Italian data protection legislation*. Computer Standards & Interfaces, 2005. 27: p. 445-455.
3. Walker, E., *Software Development Security: A Risk Management Perspective*. The DoD Software Tech. Secure Software Engineering, 2005. 8(2): p. 15-18.
4. Humphrey, E. *Information security management standards: Compliance, governance and risk management*. in *Information Security Tech. Report*. 2008.
5. Tsujii, S. *Paradigm of Information Security as Interdisciplinary Comprehensive Science*. in *International Conference on Cyberworlds (CW'04)*. 2004: IEEE Computer Society.
6. Siponen, M.T., *Information Security Standards Focus on the Existence of Process, Not Its Content?*, C.o.t. ACM, Editor. 2006. p. 97-100.
7. Schumacher, M., *Security Engineering with Patterns*. Lecture Notes in Computer Science. Vol. 2754. 2003: Springer-Verlag. 208.
8. Gupta, A. and R. Hammond, *Information systems security issues and decisions for small businesses*. Information Management & Computer Security, 2005. 13(4): p. 297-310.
9. Johnson, D.W. and H. Koch. *Computer Security Risks in the Internet Era: Are Small Business Owners Aware and Proactive?* in *39th Annual Hawaii International Conference on System Sciences (HICSS'06)*. 2006.
10. O'Halloran, J., *ICT business management for SMEs*. Computer Weekly, 2003. December 11.
11. Doherty, N.F. and H. Fulford, *Aligning the Information Security Policy with the Strategic Information Systems Plan*. Computers & Security, 2006. 25(2): p. 55-63.
12. ISO/IEC17799, *ISO/IEC 17799. Information Technology - Security techniques - Code of practice for information security management*. 2005.
13. COBITv4.0, *Cobit Guidelines*, Information Security Audit and Control Association. 2006.
14. MageritV2, *Metodología de Análisis y Gestión de Riesgos para las Tecnologías de la Información, V2*. 2005, Ministerio de Administraciones Públicas.
15. SSE-CMM, *Systems Security Engineering Capability Maturity Model (SSE-CMM), Version 3.0*. Department of Defense. Arlington VA. 326. 2003.

16. Batista, J. and A. Figueiredo, *SPI in very small team: a case with CMM*. Software Process Improvement and Practice, 2000. 5(4): p. 243-250.
17. Hareton, L. and Y. Terence, *A Process Framework for Small Projects*. Software Process Improvement and Practice, 2001. 6: p. 67-83.
18. Tuffley, A., B. Grove, and M. G, *SPICE For Small Organisations*. Software Process Improvement and Practice, 2004. 9: p. 23-31.
19. Calvo-Manzano, J.A., *Método de Mejora del Proceso de desarrollo de sistemas de información en la pequeña y mediana empresa (Tesis Doctoral)*. Universidad de Vigo. 2000.
20. ISO/IEC27001, *ISO/IEC 27001, Information Technology - Security Techniques Information security management systemys - Requirements*. 2005.
21. ISM3, *Information security management matuty model (ISM3 v.2.0)*. 2007, ISM3 Consortium.
22. Von Solms, B., *Information Security governance: COBIT or ISO 17799 or both?* Computers & Security . 2005. 24: p. 99-104.
23. Pertier, T.R., *Preparing for ISO 17799*. Security Management Practices, 2003. jan/feb: p. 21-28.
24. ISO/IEC17799, *ISO/IEC 17799, Information Technology - Security Techniques - Code of practice for information security management*. 2005.
25. McBride, T., B. Henderson-Sellers, and D. Zowghi. *Project Management Capability Levels: An Empirical Study*. in *11th Asia-Pacific Software Engineering Conference (APSEC'04)*. 2004: IEEE Computer Society.
26. Sommerville, I. and J. Ransom, *An Empirical Study of Industrial Requirements Engineering Process Assessment and Improvement*. ACM Transactions on Software Engineering and Methodology, 2005. 14(1): p. 85-117.
27. Ramasubbu, N., M.S. Krihsnan, and P. Kompalli, *Leveraging Global Resources: A Process Maturity Framework for Managing Distributed Development*. IEEE Software, 2005: p. 80-86.
28. April, A., et al., *Software Maintenance Maturity Model: the software maintenance process model*. Journal of Software Maintenance and Evolution. Research and Practice, 2005. 17: p. 197-223.
29. KcKinney, C., *Capability Maturity Model and Outsourcing: A Case for Sourcing Risk Management*. Information Systems Control, 2005. 5.
30. OMB, *OMB Enterprise Architecture Assessment v 1.0. The Office of Management and Budget, The Executive Office of the President*. 2004.
31. Van der Raadt, B., J.F. Hoorn, and H. Van Vliet. *Alignment and Maturity are siblings in architecture assesment*. in *Caise 2005*. 2005.
32. NASCIO, *National Association of State Chief Financial Officers. Enterprise Architecture Maturity Model, Version 1.3. National Association of State Chief Financial Officers*. 2003: Lexington KY.
33. Schekkerman, J., *Extended Enterprise Architecture Maturity Model. Institute for Enterprise Architecture Developments (IFEAD)*. 2003: Amersfoort, The Netherlands.
34. Widdows, C. and F. Duijnhouwer, *Open Source Maturity Model. Cap Gemini Ernst & Young*. 2003: New York NY.
35. Mekelburg, D., *Sustaining Best Practices: How Real-World Software Organizations Improve Quality Processes*. Software Quality Professional, 2005. 7(3): p. 4-13.

Metodología para la selección de métricas en la construcción de un Cuadro de Mando Integral

Daniel Villafranca¹, Luis Enrique Sánchez¹, Eduardo Fernández-Medina², Mario Piattini²

¹SICAMAN Nuevas Tecnologías. Departamento I+D,
Juan José Rodrigo, 4. Tomelloso, Ciudad Real, España
{[dvillafranca](mailto:dvillafranca@sicaman-nt.com), [lesanchez](mailto:lesanchez@sicaman-nt.com)} @sicaman-nt.com

²Grupo de Investigación ALARCOS, Universidad Castilla-La Mancha
Paseo de la Universidad, 4 – 13071 Ciudad Real, Spain
{[Eduardo.FdezMedina](mailto:Eduardo.FdezMedina@uclm.es), [mario.piattini](mailto:mario.piattini@uclm.es)} @uclm.es

Resumen. La implantación práctica de Sistemas de Gestión de la Seguridad de la Información presenta una problemática añadida para el caso de las PYMES debido a la falta de herramientas y guías adaptadas a su estructura organizativa y procesos en el área de las Tecnologías de la Información (TI). La selección de indicadores adecuados y la definición de métricas acordes para la construcción de un Cuadro de Mando Integral (CMI) de la Seguridad de la Información es un problema que las guías y métodos estándar no resuelven completamente. Es por ello que hemos desarrollado un nuevo método, orientado a PYMES y desde un punto de vista empírico, para la selección de indicadores y construcción de las métricas que van a proveer la información para nuestro CMI. En este artículo exponemos en detalle dicho método.

Palabras clave: Métricas, Selección de indicadores, Cuadros de Mando Integral (CMI), SGSI.

1. Introducción

En la última década, el mundo empresarial ha experimentado una transformación radical de sus procesos de trabajo con una dependencia cada vez mayor de las TI. Una de las consecuencias de este hecho es que las organizaciones necesitan aplicaciones cada vez más seguras. Las cifras manejadas por EITO (Observatorio Europeo de las Tecnologías de la Información) son una prueba de que la seguridad es una de las principales preocupaciones de las empresas [18], aunque en un reciente informe del Small Business Technology Institute [31] se ha descubierto que el 20% de ellas no poseen protección antivirus adecuada. Para mejorar la seguridad de las tecnologías de información para las empresas es necesario definir controles de seguridad, y que su cumplimiento sea monitorizado continuamente [10]. Esto permitiría mejorar la eficiencia de esos controles y conocer las vulnerabilidades al principio.

Asociados a estos controles de seguridad, se deberían usar métricas o indicadores de seguridad que nos permitan tener datos objetivos del cumplimiento de estos

controles. Para ello es preciso contar con un marco adecuado para la selección e implementación de métricas [6]. Por lo tanto, las métricas de seguridad son necesarias para saber el estado de un sistema de información [9] y tienen por finalidad conocer, evaluar y gestionar la seguridad de los sistemas de información.

Las métricas de seguridad son por lo tanto especialmente importantes para gestionar la seguridad de las empresas. Esta gestión se hace mediante los sistemas conocidos como Sistemas de Gestión de Seguridad (SGSI) [20], que requieren herramientas y metodologías adecuadas para su implantación. Actualmente, la mayoría de las grandes empresas han abordado la implementación de SGSI, con base en modelos de madurez, para la gestión de su seguridad [1], pero desafortunadamente la implantación de este tipo de sistemas en pequeñas y medianas empresas es muy complejo debido fundamentalmente que no disponen de herramientas y metodologías adecuadas para este tipo de empresas [14].

Una de las herramientas más importantes para los SGSI son los CMI de la seguridad. Desde su aparición en 1992 [8] los CMI han sido usados por centenares de organizaciones como un medio para describir su estrategia y medir su rendimiento. Los CMI son muy útiles para controlar procesos regulares con un flujo de información continuo (como es el caso de la gestión de la seguridad en las empresas), ya que obtener y agrupar la información más relevante y útil para la toma de decisiones, y que resulte crucial para tener un conocimiento permanente de la situación que se gestiona y de su evolución en el tiempo [5]. Junto a los modelos de madurez, los CMI nos ayudarán a conocer la situación de la seguridad conseguida mediante la implantación del SGSI, así como su evolución.

Las métricas de seguridad han sido siempre difíciles de evaluar [1]. Cada vez se utilizan en los procesos de auditoría de TI herramientas automatizadas para el desarrollo de sus revisiones, lo que ha contribuido enormemente a que se pueda suplir la falta de conocimiento en determinados campos específicos durante el desarrollo de una auditoría de TI. Por otro lado, para solucionar estos problemas, algunas propuestas abogan por incorporar nuevas tecnologías que permitan captar la experiencia humana en la auditoría de TI, mediante los Sistemas Expertos (SE) [10]. En este sentido, los SE permiten fundamentalmente obtener respuestas inmediatas y fiables, extender el conocimiento de un experto en una materia concreta.

Como hemos expuesto en trabajos anteriores [15] [16], en las pequeñas y medianas empresas, la aplicación de normativas de seguridad cuenta con el problema adicional de no tener recursos humanos y económicos suficientes para realizar una adecuada gestión. Los modelos de madurez generales no han sabido dar respuesta práctica en este caso, lo que nos ha llevado a desarrollar un modelo propio, tomando como marco de referencia la norma ISO/IEC 27002 [14]. También se ha visto la necesidad de complementarlo con un proceso particular de selección de métricas para la construcción de CMIs adecuados al contexto de este tipo de empresas [20].

Por lo tanto, el objetivo de esta investigación es la definición de un método sistemático basado en ingeniería que nos permita seleccionar y segmentar las métricas de seguridad para la construcción de cuadros de mando de seguridad optimizados para cada caso. Para la construcción de este método nos hemos basado en los planteamientos de construcción de cuadros de mando clásicos top-down y bottom-up [13]. La idea es ofrecer soluciones consolidadas y que pueden ser reutilizadas en el contexto del diseño del CMI de la seguridad.

Asimismo, este proceso de construcción del CMI de Seguridad encaja con nuestro modelo de madurez desarrollado para PYMES [14] [17], ya que se realiza de forma incremental partiendo del nivel que la organización tiene según nuestro modelo de madurez en espiral [15] y conjuga otros elementos adicionales (*know-how* previo, sistemas expertos probabilísticos, herramientas para la selección automática de indicadores,...) conformando de esta forma un procedimiento recurrente para la selección de métricas.

El resto del artículo se organizará así: en el apartado siguiente se revisarán los indicadores y métricas de seguridad, los estándares que los definen y su planteamiento para su utilización en la construcción del CMI de la seguridad. En el apartado 3 presentaremos nuestra metodología para la construcción del CMI de seguridad, expondremos los detalles y mecanismos de selección en el apartado 4 y finalmente presentaremos las conclusiones y marcaremos los hitos para futuros trabajos.

2. Antecedentes

El Cuadro de Mando de Seguridad de la Información es una herramienta de control de gestión que traduce la estrategia de seguridad en un conjunto de objetivos relacionados, medidos a través de indicadores, con unas metas fijadas y ligados a unos objetivos que facilite la toma de decisiones.

A continuación describiremos cómo se fijan las métricas de seguridad (los objetivos), para la construcción de nuestro CMI de seguridad que será la herramienta para la gestión de la seguridad, para finalmente describir la metodología que hemos desarrollado.

2.1 Indicadores y Métricas de seguridad en la construcción del CMI de la Seguridad

Las métricas de seguridad facilitan el cumplimiento de los objetivos, cuantificando la implantación de los controles de seguridad y la eficacia y eficiencia de los mismos, analizando la adecuación de los procesos de seguridad e identificando posibles acciones de mejora [3]. Las métricas deben proporcionar información cuantitativa (porcentajes, medias, rangos).

En cambio, los indicadores proporcionan un solo punto en el tiempo, son puntos de vista específicos, factores discretos, mientras que las métricas son derivadas de la comparación de varios indicadores sobre una referencia [12]. Los indicadores son generados a partir de una medición, mientras que las métricas son generadas a partir de un análisis [6]. En otras palabras, los indicadores son datos objetivos en bruto, mientras que las métricas son interpretación de esos datos. Sin embargo en COBIT 4.0 [25] existen algunos matices sobre lo anteriormente expuesto, ya que algunos de los términos que se refieren como indicadores serían lo que se denominan métricas en el NIST 800-55 [8].

En un intento de especificar las métricas y las técnicas de medida aplicables para determinar la eficacia de un SGSI y de los controles relacionados, es interesante destacar la normativa ISO 27004 [26], que actualmente está en fase de desarrollo (se

ha retrasado su publicación), y que nos va a aportar una visión en el área de las métricas de seguridad.

Una vez revisados los conceptos sobre las métricas, la dificultad siguiente se encuentra en definir cómo deben generarse para construir un programa de métricas de seguridad. En [12] se presentan una serie de pasos principales, que puede utilizarse como referencia en procesos parecidos, y que es una guía para establecer este programa de seguridad:

- Definir el programa de métricas objetivo(s) y los objetivos del mismo.
- Decidir los indicadores que van a conformar las métricas a generar.
- Desarrollar estrategias para la generación de las métricas.
- Establecer puntos de referencia y *benchmarks*.
- Determinar la forma la que las métricas serán reportadas.
- Crear un plan de acción y llevarlo a cabo, y
- Establecer un programa formal de revisión y refinamiento del ciclo.

Una vez definido el proceso, el paso siguiente es desarrollar un formato estándar que garantice el análisis y la evaluación para la selección de estos indicadores de forma repetida. Con esta idea, en NIST [11] podemos encontrar un formato o plantilla que nos provee detalles de sobre una serie de parámetros que van a definir la métrica, tales como el tipo de control, propósito de medida, valores, etc., y que nos han servido de referencia en la construcción de nuestro modelo.

Aún estando claro su propósito, no es sencilla la construcción de una buena métrica. En la definición de métricas es habitual encontrarse con numerosos problemas, siendo los más relevantes los siguientes [21]:

- Las métricas no están siempre definidas en un contexto en donde el objetivo o interés industrial que se pretende alcanzar mediante su utilización es explícito.
- En ocasiones, aunque el objetivo sea explícito, las hipótesis experimentales a menudo no están hechas de forma explícita.
- Las definiciones de métricas no siempre tienen en cuenta el entorno o el contexto en el cual serán aplicadas.
- A menudo, no es posible realizar una adecuada validación teórica de las métricas porque el atributo que una métrica pretende cuantificar no está bien definido.
- Un gran número de métricas no han sido nunca objeto de validación empírica

De acuerdo a nuestra experiencia en SICAMAN, con el uso de métricas de seguridad en nuestros clientes, hemos observado que en la implantación de un SGSI es fundamental tener pocos indicadores al principio pero bien definidos, teniendo claro lo que se está midiendo, porqué y para qué. Se debe de poder hacer comparativas (históricas o benchmarking), hay que medir las cosas de la misma manera. Para ello se deben de buscar procedimientos sencillos y de fácil implementación.

Por otro lado, encontramos el concepto del gobierno de las TI que surge como respuesta a la brecha existente entre las expectativas y los resultados obtenidos en el uso de las TI en las organizaciones. En [4] se propone que entre las medidas a realizar para lograrlo deben realizarse una mejora en las operaciones con un enfoque integrado de seguridad, disponibilidad e integridad de proceso.

También encontramos en COBIT [25] una respuesta para dar soporte al gobierno de TI al brindar un marco de trabajo que garantiza que (figura 1):

- TI está alineada con el negocio
- TI capacita el negocio y maximiza los beneficios
- Los recursos de TI se usen de manera responsable
- Los riesgos de TI se administren apropiadamente



Fig. 1. Áreas principales del Gobierno de TI (COBIT [25])

Esta figura representa un enfoque para el gobierno de TI describiendo los tópicos en los que la gerencia requiere poner atención en sus empresas. La dirección operacional usa procesos para organizar y administrar las actividades cotidianas de TI. Se establecen equivalencias entre los modelos de procesos COBIT y las áreas principales del gobierno de TI, ofreciendo así un puente entre lo que los responsables de seguridad y operaciones deben realizar y lo que la gerencia desea controlar.

Aunque no existe un consenso en cómo realizar el alineamiento estratégico de una organización, en las próximas secciones veremos como el cuadro de mando de la seguridad es una respuesta a estos problemas [32] y va a contribuir a establecer el Gobierno de las TSI. Como más adelante expondremos, el enfoque anterior nos ha servido de base para definir las principales áreas que vamos a contemplar en la construcción de nuestro CMI.

Tal y como hemos expuesto, las conclusiones sobre el marco de Gobierno de TI, las diferentes normativas, indicadores y métricas de seguridad representan una escenario complejo. La implantación de los SGSI requiere la realización de un análisis inaccesible para pequeñas organizaciones en las que es difícil alinear objetivos de gobierno (demasiado abstractos) con las necesidades de seguridad que se tienen en la operativa diaria [16]. Esto nos ha llevado a la elaboración de un modelo de madurez de la seguridad que está especialmente diseñado para ser implantado en las PYMES [17] [20], en las que debido a sus características particulares, resulta difícil adecuar los estándares y modelos sobre métricas y seguridad de la información.

2.2 Proceso de construcción del CMI de Seguridad

Por nuestra experiencia hemos comprobado que cada compañía tiene intereses distintos en materia de seguridad, y las métricas se establecen de acuerdo a lo que se esté tratando de proteger y medir, así como de la situación de la empresa [6]. Las compañías se imponen como objetivo gestionar la seguridad en base a información cuantitativa que facilite la toma de decisiones y el análisis de inversiones y dé confianza a accionistas, dirección y usuarios.

El Cuadro de Mando de Seguridad de la Información es una herramienta de control de gestión que traduce la estrategia de seguridad en un conjunto de objetivos relacionados, medidos a través de indicadores, con unas metas fijadas y ligados a unas iniciativas. Esta es una herramienta que nos va a permitir en nuestro caso sintetizar los procesos de control de seguridad para ofrecer una información sencilla, resumida y eficaz para observar la evolución de los indicadores y métricas de seguridad.

El modelo de funcionamiento básico sobre el que se sostiene el cuadro de mando es la fijación de unos objetivos en la organización, que son realizados mediante unas actuaciones que tienen reflejo en unas variables clave y que se controlan a través de indicadores [8]. De esta forma, el CMI como herramienta, debe monitorizar los procesos de seguridad en TI y facilitar la toma de decisiones a las organizaciones [5], que en el caso de las PYMES, que suelen carecer de estructura y departamentos especializados en TI, requieren presentar la información de forma muy precisa y simplificada a la dirección.

El aspecto fundamental que determina la elección de la metodología o aproximación que escojamos para el cuadro de mando es la finalidad del mismo. De esta forma, en el proceso general para la construcción de nuestro CMI orientado a PYMES, hay que revisar cómo se han definido los indicadores y seleccionado las métricas. [20].

Existen principalmente dos metodologías para la construcción de un Cuadro de Mando: top-down y bottom-up [13]. El primero es más formal y completo, permitiendo a los distintos grupos de interés definir sus necesidades y objetivos [5]. Por el contrario el efecto abajo-arriba, es menos formal y permite a las organizaciones inmaduras en cuanto a la seguridad, acelerar el desarrollo de sus CMIs. Alternativamente, también se han propuesto otras técnicas en cascada [20]. Estos dos enfoques se exponen de forma práctica en las siguientes tablas [12]:

Tabla 1. Enfoques de construcción del CMI: *top-down*

ENFOQUE <i>TOP-DOWN</i>	
a. Definir/lista de objetivos del programa general de la seguridad.	Ejemplo de objetivo: <i>Reducir el número de infecciones por virus dentro de la compañía en un 30% sobre el año anterior.</i>
b. Identificar métricas que puedan indicar el progreso hacia cada objetivo.	Ejemplo de métrica: <i>Ratio de alertas de virus por infecciones en comparación a la referencia del año anterior.</i>
c. Determinar indicadores necesarios para cada métrica	Ejemplo de indicadores: <i>Número de alertas de virus en la organización por meses</i> <i>Número de infecciones de virus reportadas</i>

Tabla 2. Enfoques de construcción del CMI: *bottom-up*

ENFOQUE <i>BOTTOM-UP</i>	
a. Identificar indicadores que están o pueden ser recogidos en este proceso	Ejemplo de indicador: <i>Número medio de vulnerabilidades de Nivel 1 detectadas por servidor en cada departamento utilizando la herramienta de escaneo xyz</i>
b. Determinar las métricas que pueden ser generadas a partir de los indicadores	Ejemplo de métrica: <i>Cambio en el número de vulnerabilidades críticas detectadas en los servidores por departamento desde el último informe</i>
c. Determinar las asociaciones entre las métricas derivadas y los objetivos establecidas en el programa general de la seguridad	Ejemplo de objetivo: <i>Reducir el nivel de vulnerabilidades detectables en servidores por cada departamento dentro de la compañía</i>

Por considerar algo rígidos estos métodos [19] debido a que están más orientados a modelos de negocio específicos en lugar de la integración de las TI, la idea de nuestro enfoque para la construcción de un CMI de seguridad es un planteamiento mixto entre un enfoque en cascada y que además se realimenta con experiencias de implantaciones anteriores, recogiendo los objetivos que se definen desde la gerencia y el estado previo de los sistemas con los que cuenta la organización [20]. Es importante señalar que en el enfoque bottom-up deben asociarse los datos derivados de las métricas con los objetivos establecidos en el programa del SGSI.

Finalmente, de cara a resolver nuestro problema con un enfoque más orientado a las PYMES, para la construcción de un buen CMI se requerirá que las métricas estén equilibradas [5] [23] de acuerdo a:

- El tiempo: Pasadas (resultados) vs. Futuras (mejora y crecimiento).
- El alcance: Externas (accionistas y clientes) vs. Internas (empleados y procesos).
- Las perspectivas: Que en los modelos generales no orientados a la seguridad serían Indicadores de resultados (financiera y clientes) vs. Inductores de resultados (procesos internos y empleados).

En el caso del CMI que hemos diseñado, una vez obtenidos los valores de las métricas totales para cada uno de los dominios, la presentación de la información en el CMI se agrupará en las áreas, según refleja la Figura 2:

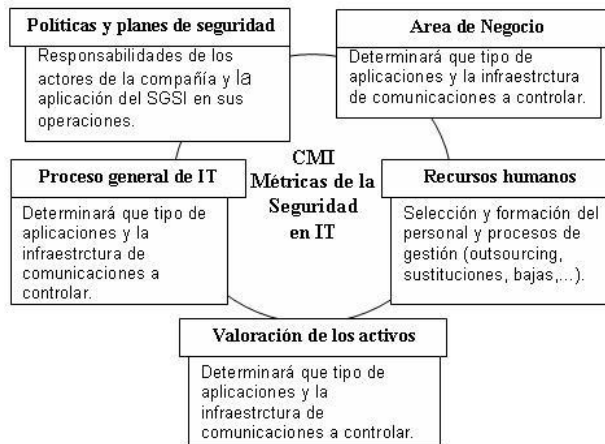


Fig. 2. Perspectivas de nuestro CMI de Seguridad

En relación a nuestro modelo, con base a los dos puntos anteriores y partiendo de una clasificación práctica basada en nuestra experiencia de las mediciones realizadas en las empresas, hemos realizado una clasificación global en cinco categorías, en contraste con el enfoque clásico del CMI de las TI: orientación al cliente, a la organización, a la operativa actual y futura. El objeto de construir un CMI a medida de las necesidades de cada organización nos ha hecho definir las siguientes categorías que están relacionadas con el modelo tradicional [19] y los dominios de COBIT [25]:

- **Los recursos humanos:** relacionada con la selección y formación del personal, así como los procesos de gestión del mismo.
- **El proceso:** en función de la actividad de la empresa y la tecnología utilizada en el mismo, determinará qué tipo de aplicaciones, así como la infraestructura de comunicaciones que será fundamental controlar.
- **Los clientes y el negocio:** será vital determinar qué activos son los más importantes que se deben proteger con el fin de preservar la imagen de la compañía.
- **Valoración de los activos,** la relación coste/resultado que se obtiene de la implantación de un control para mitigar un riesgo va a constituir un factor clave, ya que muchos riesgos se asumen porque el esfuerzo es mayor que el beneficio que se obtiene.
- **Política operativa y planes de seguridad:** Nos permitirá determinar las responsabilidades de los actores de la compañía y la aplicación práctica de nuestro sistema de gestión de la seguridad en función de sus operaciones, definiendo las métricas dentro de los dominios de nuestro modelo en espiral.

El principal problema que se encuentra en la construcción de los CMI de seguridad, particularmente en el caso de proyectos orientados a PYMES, es seleccionar las métricas adecuadas que a partir de la utilización de indicadores que

tengan un menor coste, preferentemente obtenidas de forma automática, y que produzcan un menor impacto en el sistema para cumplir los objetivos del SGSI.

Este proceso de decisión no está definido en ninguna guía o estándar de forma óptima para su aplicación en las organizaciones que nos ocupan, ni se tampoco se ha planeado la incorporación en herramientas de procesamiento automático. Así mismo, al ser uno de los procesos claves y más costosos en tiempo para la evaluación de nuestro SGSI debe ser optimizado y este ha sido la principal motivación en nuestro trabajo.

3. Metodología para la construcción de un CMI de la Seguridad

El objetivo de nuestra investigación ha sido automatizar y optimizar el proceso de selección de indicadores y definición de las métricas de seguridad para la construcción de un Cuadro de Mandos Integral para la Seguridad, partiendo de unos indicadores de seguridad predefinidos que nos da el Esquema del SGSI, otros que disponemos previamente y otros que serán necesarios obtener, para ir construyendo las métricas que conformarán el CMI para los objetivos que son definidos por la gerencia. Es por ello, que al ser un proceso clave y de evolución continuo en la medición de nuestro SGSI, que hemos buscado la definición de un proceso sencillo, repetible y fácilmente automatizable.

En el problema que nos ocupa, existe una disparidad entre los requisitos de seguridad de alto nivel (lo que quieren los gerentes de las organizaciones) y los indicadores que se recogen a bajo nivel (lo que sucede realmente en los sistemas de TI) [6].

Como comentábamos en la sección 2.1, las métricas e indicadores de cara a su identificación e implementación práctica son definidas en base a unas características (tipo de control, forma de obtención,...), [10], [26]. Estos datos son importantes a la hora de construir una buena métrica, pero también se expusieron ciertos problemas que hacen difícil cubrir estas características.

Asimismo, anteriormente hemos relatado la importancia del proceso de selección de los indicadores y de las métricas para la correcta definición de un CMI de seguridad, que garantice el éxito del SGSI y así mantener la confianza de los interesados (stakeholders) en su inversión [10]. Por ello es fundamental que el proceso defina también los objetivos de la gerencia para la evaluación de la seguridad con nuestro CMI.

Sin embargo, a pesar de la próxima aparición del estándar ISO 27004 tratado previamente, tal y como ya expusimos en anteriores trabajos [14], [20], debido a las características particulares, nos sigue resultando complicado adecuar los estándares y modelos sobre métricas y seguridad de la información a las PYMES. Por ello, para adaptar a las necesidades básicas de este tipo de organizaciones, nos dio pie a desarrollar una metodología propia para la evaluación de nuestro SGSI, con base en el modelo de madurez desarrollado [15], y que trata específicamente la problemática en este tipo de empresas.

Este esquema se ha ido refinando y conformando según se presenta en la siguiente figura:

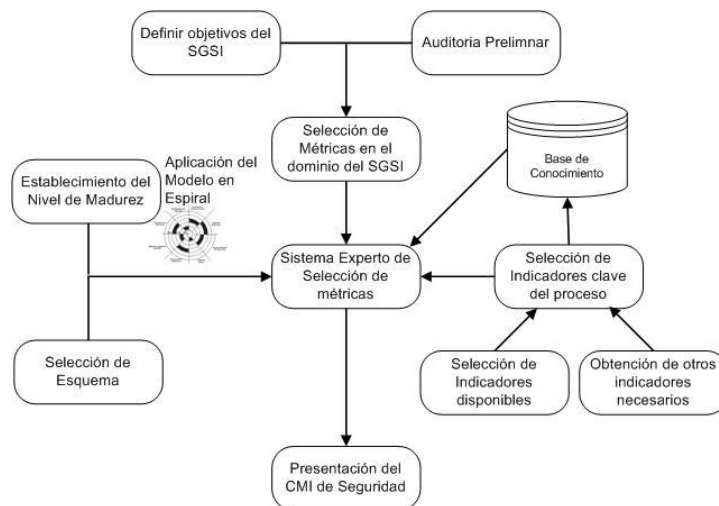


Fig. 3. Esquema de la metodología para la selección de métricas de seguridad

En este esquema se describe el proceso de obtención del CMI de seguridad, a partir de los objetivos definidos para el SGSI, la auditoría preliminar realizada y los indicadores clave recogidos. Otra dato importante que se tiene en cuenta es el nivel de madurez que tiene la empresa en ese momento y el esquema seleccionado para la empresa [14]. Finalmente se hace uso de la experiencia anterior y de los valores de referencia (benchmarks) que se obtienen a partir de las empresas u organizaciones que se dedican a la misma actividad.

Con todas estas variables, el proceso central del proceso es el sistema experto que nos va a permitir seleccionar los mejores indicadores justificado en el trabajo que realizaría un auditor experto (figura 3).

Nuestro método forma parte de un proceso general con diferentes procesos que fueron descritos de forma parcial en anteriores trabajos [14] y [20]. En ellos se exponían algunos aspectos clave a la hora de seleccionar los indicadores y métricas que iban a conformar el CMI de seguridad y que a continuación presentamos.

4. Descripción detallada del proceso de Selección de métricas de seguridad

Debido a la diversidad de criterios encontrados, la problemática suscitada y la importancia que el proceso de definición de las métricas de seguridad tiene para la construcción del CMI, hemos considerado plantear un proceso para la selección de indicadores y construcción de las métricas que conformarán el CMI.

Como propuesta para resolver el problema, hemos diseñado un algoritmo para la selección de indicadores a medida de cada empresa y que está basado en un sistema

experto probabilístico [20]. En su conjunto el proceso global hace uso de nuestro modelo en espiral aplicado a la implantación de SGSI, procesos de selección de Esquemas, sistemas de selección de métricas basados en redes bayesianas y una realimentación de las métricas seleccionadas para nuestro el proceso de selección.

El motor de inferencia implementa un modelo probabilístico basado en una red bayesiana. El proceso de construcción de este modelo tiene varias fases: en primer lugar es preciso definir las variables, a continuación obtener la estructura de la red y finalmente obtener las distribuciones de probabilidad locales.

Las variables que se han utilizado para construir la red de razonamiento bayesiano son las siguientes:

- **TE:** Tipo de Empresa (pequeña, mediana, grande)
- **TA:** Tipo de Activo, (son 23 en el esquema que define nuestro modelo)
- **A:** Obtención Automática (S/N)
- **NM:** Nivel de madurez (1,2,3)
- **R:** Ratio (*benchmark*) de incidencias (en rango %).
- **C:** Coste de obtención (bajo/medio/alto)
- **F:** Frecuencia de medición (bajo/medio/alto).
- **P:** Peso del indicador en el CMI según el tipo de dominio (1-5)

La red bayesiana (RB) se compone de dos partes. Por un lado, la estructura, el modelo o parte cualitativa: un grafo dirigido acíclico (GDA) donde cada nodo representa la variable aleatoria y los arcos representan dependencias probabilísticas entre las variables. Por otra parte, de una distribución condicional de probabilidades de la forma $P(x|\Pi_x)$ para cada nodo x dado su conjunto de padres Π_x .

Para la primera parte, hay distintos enfoques para obtener la estructura de la red, aunque sin entrar en el detalle del razonamiento apuntamos que hemos ajustado las dependencias de las variables según nuestra experiencia [15][16], lo que ha conformado el siguiente GDA propuesto [20]:

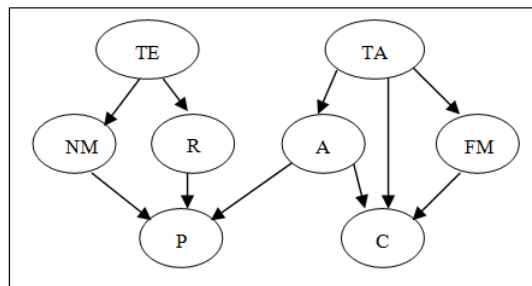


Fig. 4. Red causal para la selección de métricas de seguridad

Para la parte de la distribución condicional, hemos realizado un algoritmo de agrupamiento para obtener los grupos maximales y obtener el cálculo de probabilidades de cada nodo a partir de la red anterior. Con este valor se realiza el cálculo de la propagación de la probabilidad para así analizar cada uno de los

indicadores seleccionados, definiendo su contribución en la métrica si supera un umbral definido [20].

La idea esencial consiste en aprovechar las relaciones de dependencia (y por tanto también las de independencia) existentes entre las características del indicador y del problema para ayudar a definir la contribución del mismo en la formulación de la métrica. De esta forma se logra una implantación progresiva de la seguridad dependiendo de dos parámetros principales:

- **La dimensión de la empresa**, medido con parámetros tales como su actividad, nº de trabajadores y facturación.
- **El nivel de madurez de la seguridad** en la misma, relativo a los objetivos y metas establecidos previamente en la organización.

De esta forma y a partir del aprovechamiento del know-how con la realimentación de métricas de seguridad anteriormente utilizadas con éxito y mediante esquemas de seguridad [14]-[16], se conforma un proceso repetible en las organizaciones que tienen varios elementos en común y que se reflejan en las características de las métricas utilizadas y en algunas de las variables del sistema experto.

Este proceso de **formulación de las métricas** que conforman nuestro cuadro de mando, lo que es el núcleo repetible del proceso, se muestra en el siguiente algoritmo:

Algoritmo de selección de indicadores de nuestro mediante uso de red bayesiana

```

/* Create list of indicators */
FOR Each Control object in the schema
  IF Exist Indicator
    (Automatic or Knowledgebase) THEN
       $I_i.Value = V_i$ 
    ELSE
      Calculate  $V_i.Value$ 
    ENDIF
  Insert into IndicatorList ( $I_i, V_i$ )
END FOR
/* Calculate metric value */
FOR Each element  $M_i$  of MeasurementsList
  Calculate  $P(M_i)$  (Bayesian Net)
  IF  $P(M_i) > \text{Umbral}$  THEN
     $M_i = P(I_i) * V_i$ 
  ENDIF
  Metric.Value =  $\sum (M_i) / \prod (P(I_i))$ 
  Insert into MetricList ( $M_i$ )
END FOR
Return MetricList

```

Se parte de una selección inicial de indicadores, que agrupados en su dominio, utilizan el sistema experto para definir su aplicación. Dentro del sistema experto se calcula la probabilidad de la contribución del indicador según sus características.

La implementación de este algoritmo es una parte del proyecto global que conforma nuestra herramienta de gestión de seguridad: SCMM-PYME. El proceso de evaluación de dicho SGSI implementa el proceso descrito en punto 3 e incorpora el proceso de selección como el núcleo nuestro método de construcción del CMI de la seguridad (Figura 5):

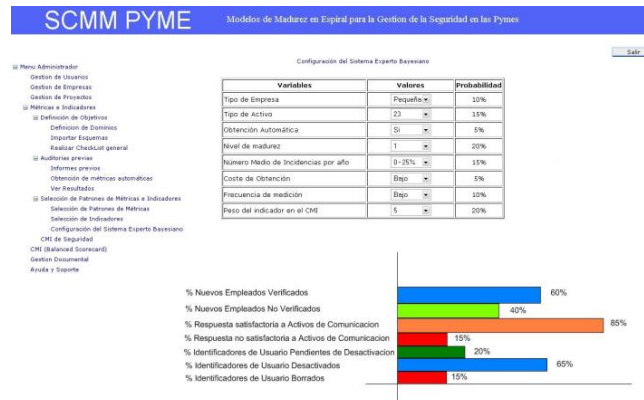


Fig. 5. Imagen de nuestra herramienta con la configuración del SE y para conformar el CMI de la Seguridad

La utilización de esta herramienta en nuestros clientes, como ya fue comentado, arroja datos satisfactorios sobre su aplicación y el cumplimiento de los objetivos propuestos. En cualquier caso, será preciso una evolución de este proceso para la obtención de nuevos indicadores, ya que son múltiples las herramientas que cada día aparecen y que sirven para obtener los indicadores automáticos y a partir de los datos recogidos en su utilización.

5. Conclusiones y próximos trabajos

Para lograr un gobierno efectivo, la seguridad de la información es fundamental, siendo necesario poder evaluar y medir la capacidad de los procesos. Para ello se deben alinear las necesidades de seguridad con las de negocio, con lo que es preciso disponer de indicadores adecuados que nos ofrezcan unas métricas eficaces para medir si nuestro SGSI responde a la inversión realizada en él.

Las métricas de seguridad son la clave que van a permitir cuantificar la implantación de los controles y evaluar la eficacia de los mismos, identificando posibles acciones de mejora. Por ello, en estos procesos de definición de métricas es vital tener en cuenta la naturaleza del negocio y organización, para poder adecuarse a cada tipo de actividad.

A partir de estas métricas se construye el CMI de seguridad como una herramienta que nos proveerá una información muy útil para la gestión y poder revisar los objetivos del SGSI. En base a las métricas e indicadores seleccionados y organizados, el CMI nos aportará:

- Control de la gestión de la seguridad relacionando los objetivos de la organización con el SGSI.
- Perspectiva histórica sobre las mejoras y evolución del SGSI

- Una referencia o comparación (benchmarking) interno y externo de nuestras métricas con las de otras organizaciones.
- Una herramienta de información a la Dirección para soporte a la toma de decisiones.
- Relacionar la seguridad con los objetivos de la empresa o del departamento

Asimismo, los equipos de auditoría en TI deben buscar y reevaluar las herramientas automatizadas que emplean para considerar nuevas y mejores formas en la manera de realizar sus trabajos.

La propuesta presentada ha tenido en cuenta los parámetros anteriores y propone un nuevo método de construcción de CMI de seguridad, mediante la selección y transformación de los indicadores en métricas, que hace uso de Sistemas Expertos (SE) basados en redes bayesianas. Aunque está claro que este tipo de sistemas no son la solución a todas nuestras necesidades, sino a una parte importante de ellas, también está claro que debemos tenerlas en cuenta cuando se logra obtener un mejor rendimiento.

En el presente artículo hemos conjugado todos los puntos anteriores para dar lugar al procedimiento completo de selección, aprovechamiento del *know-how*, automatización e implementación con un sistema experto que permite construir el SGSI de forma óptima y buscando rápidamente el ROI en las organizaciones.

Nuestro proceso forma parte de una herramienta de reciente creación que incorpora nuestro proceso de construcción del SGSI en base al modelo de madurez definido. Aunque los resultados que hemos obtenido en las primeras pruebas, hacen presagiar un buen futuro, se debe seguir trabajando en un refinamiento general y del proceso de selección. Para ello se estudia incorporar nuevos indicadores automáticos, mejorar los algoritmos actuales e incorporar nuevos algoritmos de aprendizaje.

Agradecimientos

Esta investigación es parte de los proyectos ESFINGE (TIN2006-15175-C05-05), concedido por el Ministerio de Educación y Ciencia y QUASIMODO (PAC08-0157-0668) financiado por la Consejería de Educación y Ciencia de la Junta de Comunidades de Castilla-La Mancha.

Referencias

1. Chapin, D., Akridge, S. "How can security be measured?". Information Systems Control Journal, Volume 2, 2005.
2. Corletti, A. ISO-27001 e ISO-27004. <http://www.kriptopolis.org/iso-27001-e-iso-27004>
3. Erro, G. Seguridad TICs, ¿qué hay que medir?. Aplicabilidad de las Métricas en Seguridad. Jornada Técnica Seguridad Informática.
4. Hardy, G. (2006). Using IT governance and COBIT to deliver value with IT and respond to legal, regulatory and compliance challenges. Information Security Technical Report 11 , 55-61.

5. Hervada F. y Piattini, M. Gobierno de las Tecnologías y Seguridad de la Información. RAMA (2007).
6. Heyman, T., Scandariato, R., Huygens, C. From security objectives to security metrics: a round trip.. SecSE Barcelona 2008
7. ITGI (2005). IT Alignment: Who Is in Charge?. IT Governance Domain Practices and Competencies, IT Governance Institute.
8. Kaplan, R.S. y Norton, D.S. (1992): "The Balanced Scorecard-Measures That Drive Performance", Harvard Business Review, septiembre-octubre.
9. Mañas, José A. Security Metrics and Measurements for IT. UPGRADE. Vol. VI, issue no. 4, August 2005.
10. Martín Soria, D. Nuevas Tecnologías Desplegadas con Inteligencia, ¿Serán las Aliadas de los Profesionales de Auditoría de TI en las Grandes Corporaciones? Information Systems Control Journal, Volume 1, 2008.
11. NIST Special Publication 800-80. Initial Public Draft. Guide to Performance Metrics for Information Security. April 2006.
12. Payne, S.C. A Guide to Security Metrics. (SANS Security Essentials GSEC Practical Assignment) SANS Institute. June 2006.
13. Opacki, D. Security Metrics: Building Business Unit Scorecards. Dic 2005. 4-8
14. Sánchez, L.E., Villafranca, D., Fernández-Medina, E. y Piattini, M. Developing a model and a tool to manage the information security in Small and Medium Enterprises. SECRIPT (2007).
15. Sánchez, L.E. , Villafranca, D., Fernández-Medina, E. y Piattini, M. Practical Approach of a Secure Management System based on ISO/IEC 17799. Ares (2005)
16. Sánchez, L.E. , Villafranca, D., Fernández-Medina, E. y Piattini, M. Gestión de la seguridad de los sistemas de información en las empresas desde la perspectiva de su tamaño y nivel de madurez, tomado como base la ISO/IEC 17799. WOSIS 2006.
17. Sánchez, L.E., Villafranca, D., Fernández-Medina, E. y Piattini, M. Developing a model and a tool to manage the information security in Small and Medium Enterprises. SECRIPT (2007).
18. The European Information Technology Observatory : www.eito.com
19. Van der Zee, J. "Alignment is not enough: integrating business and IT management with the balanced scorecard", Proceedings of the 1st Conference on the IT Balanced Scorecard, Antwerp, March 1999
20. Villafranca, D., Sánchez, L.E., Fernández-Medina, E. y Piattini, M. Construcción de un CMI de la Seguridad: Selección de indicadores mediante un sistema experto probabilístico. CIBSI'07. Mar de Plata, Nov.2007
21. Villarrubia, C., Fernández-Medina, E. y Piattini, M. Towards a Classification of Security Metrics. Workshop on Security in Information Systems. WOSIS 2004, Oporto, Portugal., pp. 342-350.
22. Van Grembergen, W., De Haes, S. COBIT's Management Guidelines Revisited: The KGIs/KPIs Cascade. Information Systems Control Journal, Volume 6, 2005.
23. Van Grembergen, W., De Haes, S. Using COBIT and the Balanced Scorecard as Instruments for Service Level Management. Information Systems Control Journal, Volume 4, 2003.
24. Washizaki, H., Kubo, A., Fukazawa, Y.. "Measuring Abstraction Levels of Security Patterns" Proceedings of the 1st International Workshop on Software Patterns and Quality (SPAQu'07)", pp.59-60 (2007)
25. COBIT® 4.0. 2005. IT Governance Institute (ITGI) (www.itgi.org)
26. ISO/IEC WD 27004. Information technology — Security techniques — Information security management — Measurements

Towards Secure Distributed Computations

Felipe Zipitúa

Grupo de Seguridad Informática
Instituto de Computación, Facultad de Ingeniería
Universidad de la República
J. Herrera y Reissig 565, Montevideo, Uruguay
`fzipi@fing.edu.uy`

Abstract In this paper, we present an infrastructure for securing distributed computations between hosts, using a novel technique called Proof Carrying Results. This technique is based on Necula's proof carrying code. Basically, the result of some computation comes equipped with a certificate, or witness, showing that the computation was made correctly. This witness can be used to verify that the value was generated in a good way. We will show how to add the PCR technique and its supporting infrastructure to a distributed programming language. This will make the language more robust against active adversaries, when the returned values of a computation are of abstract types. Finally, to check the values and associated witnesses produced by some host, we use the COQ proof checker for a precise and reliable verification.

Key words: Distributed Computations, Proof Checking, Proof Carrying Results, Security, Type-safety

1 Introduction

There are plenty of networks that work in a cooperative way and form what we know as *grids* of computers. These grids serve a lot of purposes, and they are used with good results for intensive calculation, because the joined computing power aids in solving any kind of complex functions. To cope with these new requirements and facilities, programming languages had to evolve into new paradigms, including facilities to do distributed computing in a straightforward way. These are known as *distributed programming languages*.

Type-safety is a property concerned with *how much* a programming language protects its own abstractions. It is usually viewed as two properties of the semantics of the programming language: *progress* and (type-) *preservation*. Not every programming language has this property, and in the ones that are distributed it is more difficult to conceive. If the programming language semantics and the type system guarantee that the encapsulation provided by type abstraction can never be breached, then the language is called abstraction safe. On a single computer it is usually guaranteed by the compiler.

However, this changes radically when we begin to transmit data over the network. Data can be modified, lost, or attacked. After the creation and utilisation of such languages, an aspect remaining to be introduced is the *security* properties of these computations. The security properties of languages that execute on a single host are hard to maintain. We must take increased precautions when dealing

with lots of hosts and complex networks. When dealing with remote computations, receiving the correct values for our computations is a must: if not, we are wasting time and effort. The correctness of the received values is independent of its type, such if it is concrete or abstract. For values of concrete types, we could have its representation available for checking that the value received is of that type. This is not the case for abstract types: its representation even could be not available at all. Therefore, how can values of abstract types be secured, in the context of a distributed programming language?

We propose the use of a novel technique, called Proof Carrying Results [2] (PCR). Basically, the result of some computation comes equipped with a certificate, or witness, that can be used to check the correctness of these values of abstract types. There are many ways for a host to check the correctness of this value, by using the associated witness. On one hand, it can be verified using a custom made checker for the particular abstract type. This way is prone to errors, in the creation of a correct checker for these values. On the other hand, a proof checker can be used as a reliable tool for checking correctness.

Throughout this work, we will use a language called *Acute* [15]. Its main features are a large part of what is needed to produce a typeful distributed programming, and involve type-safe marshalling of arbitrary values by using two primitives, *marshal* and *unmarshal*. In this work we show how to add the PCR technique to the *Acute* distributed programming language. The supporting infrastructure for the technique is introduced along with it. For checking the values and associated witnesses produced by some host, we use a proof checker for a precise and reliable verification.

Therefore, our contribution is threefold:

- an infrastructure has been defined and implemented for supporting the technique of proof carrying results,
- the *Acute* distributed programming language has been extended, with a mechanism that permits the exchange of abstract values in a certified way, and
- for doing the verification of the results, this infrastructure has been connected with the COQ proof checker [4].

This paper summarises part of the work that has been developed in our Master thesis [18]. The document is organised in this way: section 2 has a description of the general problem and the particular case we focused on. We continue with the introduction of the Proof Carrying Results technique in section 3, and how this technique can be used to help us solve the problem presented. In section 4 we show the infrastructure that was developed to support PCR in *Acute*. Finally, we detail related work, and some concluding remarks and further work.

2 (Un)Trusted remote computations

In this section we present the problem of remote computations between untrusted hosts. Figure 1 shows a basic interaction between hosts. There, some host sends a value a to an untrusted part. It asks for the computation of a function f with the value sent. Afterwards, the untrusted part obtains b as the result of applying the function f to a . Then b is returned to the initial host. The question now is: how can this host be sure that b is effectively the result of applying function f to a ?

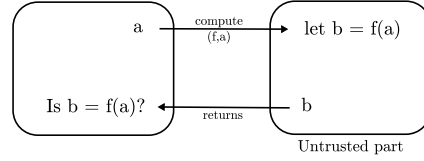


Figure 1: Remote computation problem

For this to work we must require that the result b comes along with an additional information that shows that the computation was made correctly. A solution is proposed in section 3.

In particular, the language Acute has a problem related to this one. Acute has powerful properties: type and abstraction safety are guaranteed along the distributed system. The language has documented security problems when unmarshaling values of abstract types. If the value marshalled is of an abstract type, the representation could not be available at unmarshal time. Therefore, in this case: how can it be assured that this value was generated by a well-behaved Acute run-time?

What happens when there are entities that can tamper with data transmitted between hosts? The presence of active adversaries in the networks changes the scenario. If this situation occurs, safety can be no longer guaranteed. In this case, an additional check is that the marshalled value is a well-formed representation of something of that type. Nevertheless, the implementation of that type could be used for checking types at unmarshal time. But this implementation is not available, and especially in the case of abstract types. This limits the language to a simple decision to handle this case: to work only in a trusted scenario, or to marshal only values of concrete types.

We will extend the language to support marshalling of abstract types in a non-trusted scenario. For this purpose, we will introduce the PCR technique into the language, and a supporting infrastructure for it that will end increasing the properties of safety in a distributed context. The next section summarises the contributions of this work.

3 Proof Carrying Results

This technique was introduced in [2]. The authors propose that some of the concepts introduced by [12] in proof carrying code (PCC) be reused. In that work, Necula put formal mechanisms to solve the problem of remote code execution in a behaviour-controlled way. As in PCC, there are two parts exchanging information, but here the exchange is applied in a different scenario. Instead of a code consumer, we have a results consumer: some host consumes a remote function, and the host that makes the computation sends its results with additional information that allows the consumer to validate this result. The approach is that a consumer host sends an untrusted part some computation to be done remotely. The untrusted part then returns the result of this computation, with a *certificate* that the computation has been done in a correct way. Then this certificate can be used to verify if the result was computed the correct way.

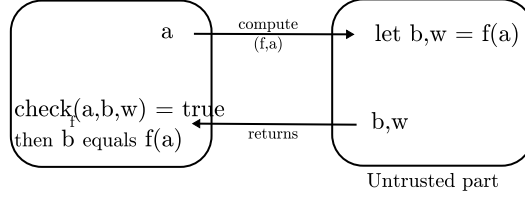


Figure 2: Basic PCR approach

Certificates provide additional data to check the correctness of computations. They can contain *witnesses*, which are generated in the process of computing, and *proofs*, that state properties about the result. These witnesses are like a trace of the computation made, and are closely related to the final result returned. Proofs, which can be part of the additional data carried, can be there to establish properties about the result itself, or about the witnesses.

Formalisation

We have a function f and we want to delegate its computation, using a as argument:

- $f(a)$ is delegated to an untrusted party, $f \in A \rightarrow B, a \in A$
- $b \in B$ is the expected value

To verify that this value is the value we sent to compute, we must have some function *check*:

$$check_f \in A \times B \rightarrow bool \mid \forall (a, b) \in A \times B, check_f(a, b) = true \Rightarrow b = f(a)$$

In general, PCR allows the untrusted part to provide additional data H intended to ease the checking process. Thus, one may have a checker function $check_R \in A \times B \times H \rightarrow bool$ such that $check_R(a, b, h) \Rightarrow R(a, b)$. In this way, the functional specification f is generalised to an input-output specification R .

This technique challenges the traditional algorithms, because more information could be needed from them, not only their result. In the next section we introduce a new kind of algorithm that is the foundation of this approach. This kind of algorithms are called Certifying Algorithms. The term was introduced in [10], and a more general approach was given in [11]. An algorithm or program is certifying when, along with the result it was supposed to give, it returns a certificate or proof that this result is indeed the correct for the given input. This is a pragmatic approach to program correctness.

4 Infrastructure for PCR in Acute

In this section we present a generic infrastructure for doing Proof Carrying Results. To this aim, we begin introducing primitives that describe flows in distributed computations. We show how the infrastructure is implemented in the Acute language, for certified result communication. Furthermore, it will be used for verifying the witnesses of the computations. Finally, we show how the certificate checking process is made, connecting the Acute language with the COQ proof checker.

4.1 Primitives for distributed computation

The primitives introduced here are operations, or a sequence of operations, that describe different ways of obtaining a certified result. Each primitive will then be shown as an interaction flow between entities.

In this case, these entities will be 3 hosts interacting with each other, which are called *Alice*, *Bob* and *Trent*.

The first flow is from *Bob* to *Alice* in Figure 3a. *Bob* sends *Alice* a value v of type T . In this case, *Alice* did not directly ask for this value. There is no guarantee that *Alice* may ever receive or process the value sent by *Bob*. Thereafter, we could say that this value has been **pushed** from *Bob* to *Alice*.

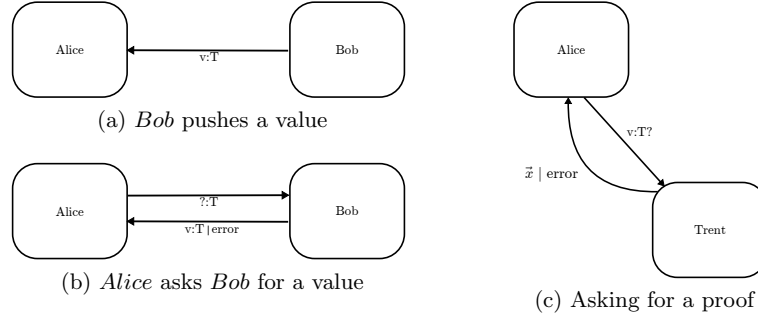


Figure 3: Flows

The second flow is from *Alice* to *Bob* in Figure 3b. Now *Alice* asks *Bob* for some value of type T . Then *Bob* returns that value of type T .

The last flow happens between *Alice* and *Trent* in Figure 3c. *Alice* has a value v and asks *Trent* for a proof that v is of type T . If *Trent* can construct such proof $\bar{x}$, it returns $\bar{x}$ to *Alice*. When some exception happens, for example, if no proof can be found at all, an error is returned.

These primitives can be combined to increase the types and complexity of flows in the system. We will be interested in the particular flows (or sub-flows) that include proofs of the value sent along the wire. In the next subsection we show how these primitives, combined, are used as a PCR infrastructure model.

4.2 Infrastructure model

Using the primitives and flows previously described, we will model an infrastructure for performing result certification. The infrastructure is modelled by the interaction of the 3 hosts shown in Figure 4.

A simplification of this infrastructure is that $Alice \equiv Trent$. The full infrastructure considers all the flows and primitives presented. In our implementation, we will focus on the primitive $v : T, w$.

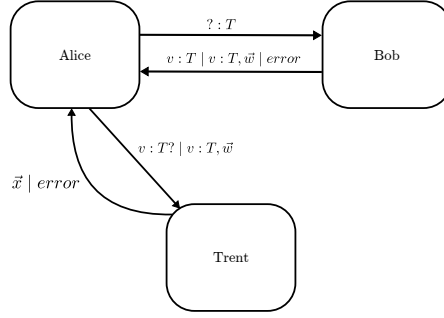


Figure 4: Infrastructure model

4.3 Implementing the infrastructure in Acute

Now that we have introduced a general infrastructure for using in distributed computations, we show how it is applied to the Acute language for obtaining certified result communication.

We extend the way that we use the marshal primitive by adding a witness, obtaining a new syntax for it

marshal e1 e2:T witness

which may now include the information that can be used by the receiver to check an invariant. The use of witnesses for certification is optional: if there are no witnesses the primitive will behave as the old one. In the case that certificates are required by the receiver, the sender cannot choose: the value must be sent with a witness or it will not be used by the receiver, and an *exception* will be raised.

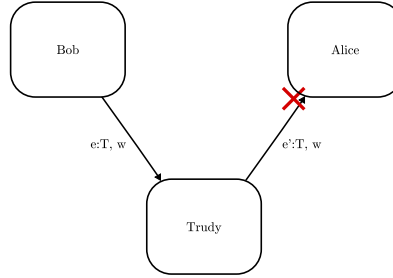


Figure 5: Adversary can not modify values only

The semantic of the primitive is changed accordingly. With this extension, our system is more robust against active adversaries. It is important to note that the adversary could modify the value *and* recalculate the witnesses for this new value. This case does not directly affect the problem we are trying to solve. This happens because, even when the original value has been modified, it complies with the invariants the new value must have. The witnesses will be used to check this

compliance, and all we should care about is that the witnesses are well generated to certify the value. A trivial example is that we ask a remote host for a prime number, and the remote host sends us some number. In the middle of this transaction, an adversary changes this number. As long as the new number is also prime and it is sent with a proof, things will keep working and abstraction safety is kept.

Verifying witnesses

To complete the process of result certification, the receiver must perform some actions when receiving the value. We suggest the use of the received witness to check that the invariant holds for the value received. Adding up, the primitive for `unmarshal`

`unmarshal e:T'`

will be changed to reflect the new extension. The new primitive

`unmarshal e:T' witness`

will support the use of witnesses. In the `unmarshal` process we will check that the invariant holds for the module by using that witness. Therefore, our main contribution is to add functionality to Acute that allow us to send values of an abstract type along with a certificate that proves that the value complies with the type invariant. Enforcing the check of witnesses must be supported, because for some types we want to receive certified values for continuing our computation. The `unmarshal`, as in the `marshal` case, alters its semantics in a similar fashion to include the witness.

4.4 Certificate checking

After receiving the values, with extended information about its computation, the infrastructure uses the certificate to check the value just received. For the process of checking the certificate we decided to use the COQ proof assistant. Even though COQ has many impressive functionalities, in this case it is used only as a proof checker for the received certificate.

In general, to interact with COQ, the user has a command-line interface. This is because it is commonly used interactively. We have that a simple proof contains definitions, declarations, and the actual proof. This proof is surrounded by the “Proof” and “Qed” words. At first sight, we did not have the prerequisites of interaction, we just needed to have certain proof approved or not. To use it in this way, the value and its certificate, along with other pre-defined constants and COQ commands, were written into a file. After that, we call the COQ compiler passing this file to be compiled. This was easy to implement, and we could obtain basic results with it: the compiler returned a value 0 if the proof was successful or $\neq 0$ if it wasn't. This solution requires forking a new process, the COQ compiler.

When writing the file to be sent to COQ we had to take several things into account. First of all, a valid proof had to be constructed with the assertions and proofs that were present in our certificate, including the received value. In the end, this procedure resulted in a concatenation of strings in the file, ordered by some criteria. Following the Figure 6 we provide a complete description of this process. In this Figure there are three different parts represented:

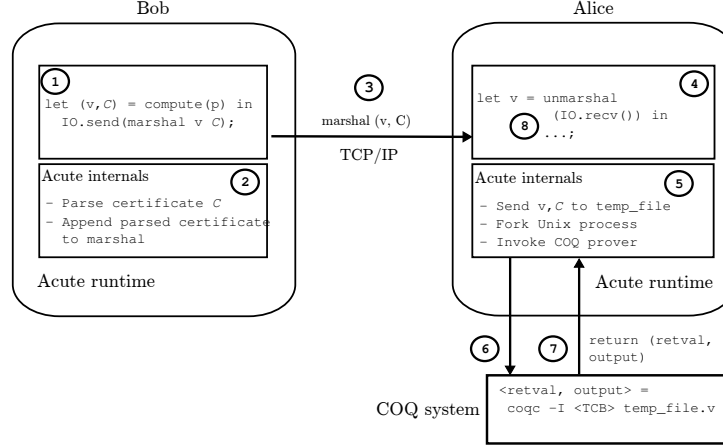


Figure 6: Full proof sequence

- the Acute run-time, responsible of executing Acute source code,
- the Acute internals, and
- COQ.

We describe the process using eight steps. Bob generates some value v with a witness C of that computation, and sends this value to Alice using the marshal primitive. In step (1), there is sample code that *Bob* uses to create the corresponding value v , and its certificate. Next (2), the Acute internal run-time adds the certificate to the value to be marshalled. After that (3), the marshal is transmitted over some network medium (e.g. TCP/IP). Alice, at (4), expects this value after invoking the unmarshal primitive. After the low-level network transport has delivered the message, the Acute internal run-time uses the values received to construct a valid proof sequence to be sent to COQ. This is represented in step (5). Here, a new Unix process forks and the COQ compiler is executed with the specified parameters, the value and its certificate (step (6)). After this fork, the system waits until a response is generated by COQ. When we have a successful compilation of the proof sequence, COQ ends returning a value of 0 (no error), as usual in Unix systems. If any kind of error occurs, the COQ process will returns a value different from 0, and some string output is shown at the standard output of the process (step (7)). We catch this string output, and return it to the Acute run-time, to be used by the language Acute if needed, for example, to raise an exception. This is shown in step (8).

Now we continue with an example, where we present an scenario for applying this technique and a simple test case.

5 Case Study: Using PCR for certified prime computation

This section introduces an example which shows the consolidation of the new functionality in the Acute language. We begin by setting a scenario for this example, and continue with a simple test case. In addition to this, we explore an extension

of the protocol defined before for doing distributed calculation of witnesses and also certificate checking.

5.1 Scenario

Alice wants to start a secure communication with some host, which acts as a server. At some point in this process, she will need one large prime number to use as a seed for some cryptographic algorithm. *Alice* does not have computational resources to do this computation by herself. Thus, she will outsource this computation to *Bob*. *Bob* is an untrusted host for *Alice*, but with lots of resources to compute

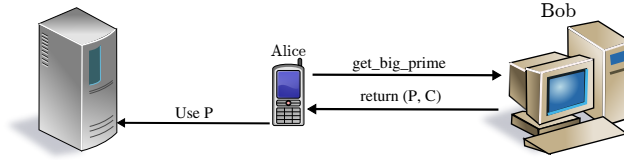


Figure 7: Alice and Bob

this number. Knowing this, *Alice* requests *Bob* a prime number. *Bob* generates this number, along with a proof that it is prime. When this number is returned to *Alice*, she will use the proof to check that the number is prime. If this checking succeeds, she can start the secure communication with the server. All hosts will communicate using the **marshal** / **unmarshal** primitives for the transmission of values with each other. This **marshal** primitive can be used with a witness. At unmarshal time that witness will be checked. This check happens automatically, and is made in a transparent way. In this process, the Acute language will try to verify all properties and values that come in the witness. This can be done in two different ways:

- using only the information received. In this case the proof must be complete, with all needed sub-proofs. The COQ example for this case is when we have no *prelude* at all.
- relying on some pre-shared information. For example, there could be a *trusted common base* of algorithms and proofs. This base will be referred to in some way by the sender. An example TCB table could be 1. When a host needs to prove some property, it will look up its table to find out how to check that property. In the case that the property to check is not on the table, the request cannot be fulfilled, and an exception will be returned.

Table 1: TCB Properties and proofs

Property name	COQ Theorem	COQ Code
Prime	Pocklington	...
OrderedLists	SortedProp	...

In any case, the needed properties must have a proof to be used by the COQ proof assistant. In the next subsection we instantiate this simple scenario into a simple test case, in which we make the proof of concept of our proposal.

5.2 Pocklington certificates

There are different ways to *certify* that one particular number is prime [3]. Pocklington's theorem[13], shown in 1 provides a sufficient condition for primality of natural numbers. We show a resume of it here to help the reader.

Theorem 1. *Given a natural numbers $n > 1$ and a witness a and some pairs $(p_1, \alpha_1), \dots, (p_k, \alpha_k)$, it is sufficient for n to be prime that the following conditions hold:*

$$p_1 \dots p_k \text{ are prime numbers} \quad (1)$$

$$(p_1^{\alpha_1} \dots p_k^{\alpha_k}) \mid (n - 1) \quad (2)$$

$$a^{n-1} = 1 \pmod{n} \quad (3)$$

$$\forall i \in \{1, \dots, k\} \gcd(a^{\frac{n-1}{p_i}} - 1, n) = 1 \quad (4)$$

$$p_1^{\alpha_1} \dots p_k^{\alpha_k} > \sqrt{n} \quad (5)$$

We highlight these two points from the previous definition:

- Given n , it is required much more computational power to determine numbers for $a, p_1, \alpha_1, \dots, p_k, \alpha_k$ than to check that these numbers verify the conditions 2-5. Therefore, we can say that $a, p_1, \alpha_1, \dots, p_k, \alpha_k$ form a *Pocklington Certificate*.
- For a natural number n , provided we are given $p_1, \alpha_1, \dots, p_k, \alpha_k$ and a , checking primality of n is made by numerical computations for conditions 2-5, and condition 1 can be done recursively.

In [8], there is a proposal for using pocklington certificates for certifying prime numbers. We used some of the results obtained in that work (e.g. the *pocklington* executable) in the course of constructing this test case.

5.3 Simple test case

After setting the scenario, we will present a test case in detail. We will be using a number of Acute modules, the COQ proof assistant and a marshalled value to be tested. The following Acute modules will be used for this example:

- *Generator*: used by module *prime* for generating a new prime number using an known algorithm.
- *Prime*: this module will have the code to work with prime numbers.

Alice will be using the prime Module for receiving a prime number.

Listing 1.1: Alice's code

```

includesource "prime.ac" (* Prime module *)
(* Alice receives a prime number *)
let p = (unmarshal (IO.receive ()):Prime.t)
in
  Pervasives.print_endline "Prime is: ";
  Prime.print_prime p;
  Pervasives.print_newline ();;

```

In this listing, *Alice* receives a marshalled prime number, and prints it at the standard output. In the **unmarshal** operation is where the underlying work of certificate checking is done.

Listing 1.2: Bob's code

```

includesource "prime.ac" (* Prime module *)
(* Generate a prime number to be sent to Alice *)
let p = Prime.generate(Prime.init) in
  IO.send(marshal "StdLib" p
    ("prime", "apply (Pocklington_refl
      (Pock_certif 1234567891 2
        ((3607, 1)::(2,1)::nil) 12426)
        ((Proof_certif 3607 prime3607) ::
          (Proof_certif 2 prime2) :: nil)).
      exact_no_check (refl_equal true))."));;

```

Bob generates a new prime, using some seed, and then sends this prime with a certificate for it.

In the process of generating certificates we will be using a tool called *pocklington* from the CoqPrime project [9]. For example, for generating the certificate for number 1234567891, the command *pocklington* 1234567891 generates the COQ code in Listing 1.3.

Listing 1.3: Coq Program

```

Require Import PocklingtonRefl.

Set Virtual Machine.
Open Local Scope positive_scope.

Lemma prime1234567891 : prime 1234567891.
Proof.
  apply (Pocklington_refl
    (Pock_certif 1234567891 2 ((3607, 1)::(2,1)::nil) 12426)
    ((Proof_certif 3607 prime3607) ::
      (Proof_certif 2 prime2) :: nil)).
  exact_no_check (refl_equal true).
Qed.

```

With this helper program, we can easily generate a certificate for any prime number generated by *Bob*. This test case simplifies the scenario, and just sends a prime number from *Bob* to *Alice*.

When we execute both programs, the prime number is marshalled and sent by *Bob*, and received and unmarshalled by *Alice*. The Acute language, at the moment of receiving the certificate, uses the checker to certify this prime number using the algorithm defined in its table. The implementation adds two important files, which must be part of the TCB. This files are to be used by the proof assistant:

- A TCB *prelude* is for defining which prelude to add to a proof (if any), and

- A TCB list is used for defining the list of local directories to be included (-I) for compiling the proofs.

In this case, we rely on code provided by the COQ prime project to prove the number is prime using the certificate received. The result of the execution of *Alice* code is in listing 1.4.

Listing 1.4: Result of execution

```

:~::~: Run ::
:~::~: Executing program ::
Marshallled certificate found
New item on certificate list
Finished writing file with proofs.
/usr/bin/coqc -I /home/fzipi/sources/coqprime/Tactic
-I /home/fzipi/sources/coqprime/N
-I /home/fzipi/sources/coqprime/Z
-I /home/fzipi/sources/coqprime/PrimalityTest
-I /home/fzipi/sources/coqprime/List
-I /home/fzipi/sources/coqprime/elliptic
-I /home/fzipi/sources/coqprime/num
-I /home/fzipi/sources/coqprime/num/W/W8
-I /home/fzipi/sources/coqprime/examples
-I /home/fzipi/sources/coqprime /tmp/cert_d8a3a3.v
COQ said: (no error)
Prime number is:
1234567891
Thread exited cleanly:
n0: [RUNNABLE] ;; ()
:~::~: Done ::
Success:
, Es:{}, s:{}, threads:{}, (RUNNABLE: , SLOWCALL: ), MX:{}, CV:{}

```

With this example, we obtained a certified prime number.

6 Related Works

There are two basic types of work related to ours. First, there are distributed languages with properties similar to the ones in the language we worked with, Acute, that try to solve similar kinds of problems. We will not be exhaustive with programming languages that support typed distributed communications.

In the context of object oriented languages, Java [1] boosted the distributed programming paradigm, permitting the execution on lots of hosts by using its virtual machine as a basis. Primitives for doing distributed computations included the notion of marshallng (serialisation). This facility included version identifiers with the class definitions of serialised objects, as we can see in the Remote Method Invocation facility [16]. The main problem for Java programs is that classes are only identified by their (syntactic) name and class loader. This introduces problems when trying to use two classes with the same name but different methods.

For the Microsoft .NET framework, we have languages that rely on the virtual machine that are functional or object oriented. An example of a functional language for .NET is F# [14]. This language is like a porting of OCaml for the .NET framework, with some additions. F# was developed as a pragmatically-oriented variant of ML that shares a core language with OCaml. Unlike many type-inferred, statically-typed languages it also supports many dynamic language techniques, such as property discovery and reflection where needed. F# includes extensions for working across languages and for object-oriented programming, and it works seamlessly with other .NET programming languages and tools.

Second, there is work on other techniques, which are probabilistic instead of being a verification technique like the one presented in this work. In this document we used one technique for verification of distributed computations to make them more secure. There are other techniques that introduce the utilisation of probabilistic methodology for detecting possible alterations (or cheating) in the context of grid computations. The work by Wenliang Du et. al [5] on grid computing and on distributed computations by Phillippe Golle and Ilya Mironov [7] are examples of these other techniques. The main approaches are different, and even when they have very low probabilistic numbers of not being caught cheating with the results, they are not one hundred percent reliable. This is, of course, dependent on what kind of adversaries you have, and the threat analysis you made for your system. Following the strategy of checking the results of computations, Grid result checking [6] focuses on statistical checking. Because we are using a new technique, based on certification rather than probabilities, direct comparison with other techniques is difficult. Nevertheless, to have a simple measure in execution resources we made a comparison on the execution of the test case presented, with and without this PCR technique, and the results are in table 2.

Table 2: Average execution time

	Original Acute	Acute extended with PCR
send-prime.ac	0.72u 0.02sys 0:00.81elap 91%CPU	0.98u 0.02sys 0:01.09elap 92%CPU
recv-prime.ac	0.65u 0.01sys 0:00.70elap 95%CPU	11.97u 0.83sys 0:13.47elap 95%CPU

Althought it seems that times when receiving certified prime numbers are huge compared with the original execution, most of the time is used by the COQ subprocess. We have also measured this time, by calling independently to the COQ compiler on the same data set, obtaining the following execution time: 11.14user 0.76system 0:12.22elapsed 97%CPU. All measures were averaged in five executions.

7 Conclusions and further work

In this section we present our conclusions, together with lines of further work.

We have defined and implemented an infrastructure for result certification. This infrastructure can be used in other places where there is a need for using the proof carrying results technique and not only in this particular case. It is independent from the language, and also, with little or no effort, from the proof assistant used.

There is a successful implementation, for the Acute language, of a solution for the detected problem. This solution uses the infrastructure defined here, and its implementation is available for further testing [17]. In order to achieve this, we used the technique of proof carrying results, where values transmitted to other hosts carry a witness that proves the computation has been done in a correct way. For the process of verifying that the witness certifies these transmitted values, we used the COQ proof assistant. The proofs are made over these values of abstract types and their properties, carried in a certificate.

The framework seems to have the same order of execution time that the original one, with the exception of the COQ subprocess that its forked to certify the received values. The execution time of the COQ subprocess must be reduced, and optimized, to have a more usable framework in execution times.

An important aspect of the definition of the infrastructure, is that it can be used in other places where there is a need for using the proof carrying results technique and not only in this particular case. It should not be difficult to extend the framework to any other programming language which suffers from similar problems. This means that if we can run a COQ proof assistant, and either 1) we have access to the source code of the programming, or 2) the programming language has an API for interfacing in a similar way with the system; then it should be a reasonable amount of work to add this mechanism to the language/system.

The proof checker is used as a reliable tool to prove that the certificates can be verified correctly. In this step we used the COQ proof assistant, but without interacting with it; only as a verification tool. Any other trustworthy proof checker can be considered. The proofs are made over these values of abstract types and their properties, carried in a certificate. Working with a proof assistant is a good way of delegating the checking process. It is important to use a well-designed, well proven tool for doing this step.

Proof Carrying Results is a promising new approach. The progress on this subject is directly bound to the progress on Certifying Algorithms. This happens because it is difficult to find an algorithm that works for a general case. Despite this consideration, there are many applications of this technique in a number of places, given that we do not trust other hosts. In particular, in global grids where we could have many distributed computations, this technique can obtain good results. Finally, we have made a proof of concept that implementations can be carried to extend systems in a successful way. The source code of the Acute language with support for PCR can be found in [17]. There are many improvements that can be made that complement the work done up to now, and these are presented in the next section.

There are a number of lines that can be followed starting from this point. Throughout this work we have opened many possible interesting lines of research. We will try to summarise them here:

- Proof Carrying Results can be implemented in other distributed languages, with the same problems that we introduced in Acute. If we have few resources for performing computations (e.g. Java cards, cellular phones, etc.) it is an ideal place to be used. This is because computations can be performed in a server with lots of computing resources, and just checked by the client application, given a clear API for doing result certification.
- For the COQ proof assistant, it could be interesting to have a *Proof* service: this should be a server, that listens in a TCP/IP port and forwards the terms and lemmas received to an executing COQ process.

References

1. K. Arnold and J. Gosling. *The Java Programming Language*. Addison Wesley, 1996.
2. G. Barthe and F. Pastawsky. Notes on proof carrying results. INRIA Sophia-Antipolis Technical Report, 2006.

3. D. J. Bernstein. Distinguishing prime numbers from composite numbers: the state of the art in 2004, 12 2004. Aimed at Math. Comp. Available from: <http://cr.yp.to/primetests/prime2004-20041223.pdf> [cited 2008.04.29].
4. T. C. development team. *The Coq proof assistant reference manual*. LogiCal Project, 2008. Version 8.1pl3. Available from: <http://coq.inria.fr> [cited 2008.10.01].
5. W. Du, J. Jia, M. Mangal, and M. Murugesan. Uncheatable grid computing. In *24th International Conference on Distributed Computing Systems (24th ICDCS'2004)*, pages 4–11, Tokyo, Japan, Mar. 2004. IEEE Computer Society.
6. C. Germain-Renaud and D. Monnier-Ragaine. Grid result checking. In *CF '05: Proceedings of the 2nd conference on Computing frontiers*, pages 87–96, New York, NY, USA, 2005. ACM Press. doi:10.1145/1062261.1062280.
7. P. Golle and I. Mironov. Uncheatable distributed computations. In D. Naccache, editor, *CT-RSA*, volume 2020 of *Lecture Notes in Computer Science*, pages 425–440. Springer, 2001. Available from: <http://link.springer.de/link/service/series/0558/bibs/2020/20200425.htm> [cited 2008.02.20].
8. B. Grégoire, L. Théry, and B. Werner. A computational approach to pocklington certificates in type theory. In *FLOPS*, pages 97–113, 2006. doi:10.1007/11737414_8.
9. INRIA. Coq prime project. Certifying Prime Numbers with the Coq prover. Available from: <http://coqprime.gforge.inria.fr/> [cited 2008.06.04].
10. D. Kratsch, R. M. McConnell, K. Mehlhorn, and J. P. Spinrad. Certifying algorithms for recognizing interval graphs and permutation graphs. In *SODA '03: Proceedings of the fourteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 158–167, Philadelphia, PA, USA, 2003. Society for Industrial and Applied Mathematics.
11. K. Mehlhorn, A. Eigenwillig, K. Kanegossi, D. Kratsch, R. McConnel, U. Meyer, and J. Spinrad. Certifying algorithms (a paper under construction). Max-Planck-Institut für Informatik working paper, 2005. Available from: <http://www.mpi-inf.mpg.de/~mehlhorn/ftp/CertifyingAlgorithms.pdf> [cited 2007.04.24].
12. G. C. Necula and P. Lee. Proof-carrying code, Oct. 28 1996. Available from: <http://citeseer.ist.psu.edu/50371.html>; <http://www.cs.cmu.edu/~necula/tr96-165.ps.gz> [cited 2007.08.30].
13. H. C. Pocklington. The determination of the prime or composite nature of large numbers by Fermat's theorem. *Proceedings of the Cambridge Philosophical Society*, 18:29–30, 1914–1916.
14. M. Research. F#: A succinct, type-inferred, expressive, efficient functional and object-oriented language for the .net platform, 2007. Available from: <http://research.microsoft.com/fsharp/fsharp.aspx> [cited 2008.10.01].
15. P. Sewell, J. J. Leifer, K. Wansbrough, F. Z. Nardelli, M. Allen-Williams, P. Habouzit, and V. Vafeiadis. Acute - high-level programming language design for distributed computation: Design rationale and language definition. Technical report, University of Cambridge and INRIA Rocquencourt, October 2004. Available from: <http://www.cl.cam.ac.uk/users/pes20/acute> [cited 2007.01.04].
16. A. Wollrath, R. Riggs, and J. Waldo. A distributed object model for the Java system. *Computing Systems*, 9(4):265–290, Fall 1996.
17. F. Zipitria. PCR extension to the Acute source code, 2008. Available from: <http://www.fing.edu.uy/inco/grupos/gsi/sources/acute-pcr/index.html> [cited 2008.10.01].
18. F. Zipitria. Towards secure distributed computations. Master's thesis, Universidad de la República, Nov. 2008. PEDECIBA Informática - Technical Report - TR 08-22. Available from: <http://www.fing.edu.uy/~fzipi/tesis/tesis.pdf> [cited 2009.04.01].

Protocolo de creación de evidencias en entornos vehiculares

J.M. de Fuentes, A.I. González-Tablas, A. Ribagorda, B. Ramos

Universidad Carlos III, Avda. Universidad 30, 28911 Leganés, España.

{jfuentes, aigonzal, arturo, benja1}@inf.uc3m.es

Teléfono: +34 91 624 94 22

Resumen Las redes vehiculares son un novedoso escenario de comunicación. Estas redes permiten el diálogo entre vehículos, y de estos con la infraestructura. Gracias a estas redes se puede proporcionar más información y nuevos servicios a conductores y pasajeros. Uno de esos nuevos servicios es la creación de evidencias sobre el comportamiento de un vehículo. Esto será útil, por ejemplo, para la correcta determinación de responsabilidad en un accidente o para justificar un comportamiento adecuado ante una sanción recibida. Utilizando las redes vehiculares se puede obtener la descripción de ese comportamiento a través de los vehículos del entorno. Con ello se impide que el propio vehículo describa su actuación de una forma modificada acorde con sus intereses. Para abordar este nuevo servicio es necesario garantizar la seguridad de la información intercambiada. En este trabajo se presenta un protocolo de creación de evidencias sobre el comportamiento de un vehículo, obteniendo los datos desde los cercanos. Se incluye el protocolo de verificación de las evidencias, así como el análisis de seguridad de la propuesta.

Palabras clave: Evidencia, VANET, testigos.

1. Introducción

El transporte terrestre mediante vehículos es, hoy día, una de las actividades humanas imprescindibles. Por este motivo, las inversiones para la mejora de todos los elementos implicados son cada vez más notables. Los avances en esta dirección están orientados a facilitar la tarea de conducción y proporcionar un mayor bienestar y seguridad a los conductores y pasajeros.

Como muestra del nivel de desarrollo de los transportes, la siniestralidad en las carreteras ha disminuido en un 44 % en los últimos 5 años en España [1]. No obstante, la sociedad de la información en la que actualmente estamos inmersos demanda un nuevo tipo de bienestar. Se debe proporcionar medios audiovisuales (música, vídeo) así como acceso a las fuentes de información (Internet) y a las nuevas formas de comunicación (correo electrónico, mensajería instantánea, etc.).

Para satisfacer estas necesidades, las tecnologías de la información y las comunicaciones están experimentando un fuerte desarrollo en el ámbito vehicular. En los últimos tiempos, las redes ad-hoc vehiculares (Vehicular Ad-hoc NETWORK, VANET) se han sumado a las posibilidades de conectividad actuales (Wi-Fi,

GPRS, UMTS). Este nuevo tipo de redes están especialmente concebidas para interconectar vehículos entre sí y con las infraestructuras de comunicación que se despliegan en las carreteras.

Gracias a estas nuevas posibilidades de comunicación intervehicular, surgen nuevas aplicaciones interesantes para la mejora del desarrollo del tráfico. Por ejemplo, los propios vehículos pueden advertir a sus seguidores acerca de incidencias en la vía o del estado de la circulación. Sin embargo, a pesar del grado de desarrollo alcanzado, existen situaciones del tráfico para las que las soluciones tradicionales no son suficientes. En la actualidad, por ejemplo, la determinación de responsabilidad ante un accidente se enfrenta a la falta de pruebas fehacientes que reflejen adecuadamente los hechos. Igualmente, los conductores no disponen de medios de prueba que acrediten que se actuó correctamente en el momento de una supuesta infracción. Las redes vehiculares pueden contribuir en esta dirección utilizando una característica inherente: los vehículos que presenciaron el hecho pueden ofrecer los datos que conocen para describir la situación en cuestión. De esta manera, se pueden construir evidencias relativas al comportamiento de un vehículo en un pasado reciente. Este tipo de elementos probatorios son de gran interés para todos los implicados en el transporte. Los conductores pueden, por primera vez, disponer de pruebas para defenderse de acusaciones inciertas o aclarar las circunstancias de un accidente. Así mismo, las aseguradoras evitan los fraudes derivados de partes de accidente con datos inciertos. Por su parte, las autoridades pueden simplificar el proceso de esclarecimiento de las circunstancias de un siniestro, así como la burocracia de la tramitación de sanciones.

En este trabajo se propone un protocolo para la creación de evidencias sobre el comportamiento de un vehículo obtenidas a través de aquellos que circulan en su entorno. Para construir esa evidencia, el solicitante podrá obtener el consenso a partir de los testimonios recogidos. Con ello se facilita su interpretación por parte del destinatario (que, generalmente, será un juez o autoridad competente). Además, se persigue el anonimato de los testigos frente al solicitante, con lo que se previenen posibles represalias derivadas de testimonios no favorables para este. Para la consecución de los aspectos de seguridad, el protocolo utiliza los servicios establecidos por el estándar IEEE 1609.2, de seguridad en redes vehiculares [2].

El resto del artículo se estructura como sigue. El apartado 2 describe el entorno vehicular y señala los aspectos relevantes del escenario y del estándar sobre el que se apoya la propuesta. El apartado 3 introduce el concepto de evidencia y detalla cómo se puede obtener en el ámbito vehicular. El apartado 4 desarrolla el protocolo de creación de evidencias que se propone. El apartado 5 se centra en el protocolo de verificación de las evidencias creadas. El apartado 6 expone el análisis de seguridad de la propuesta. El apartado 7 incluye los trabajos relacionados y, finalmente, el apartado 8 recoge las conclusiones y líneas futuras.

2. El entorno vehicular

El entorno vehicular, como escenario de comunicación, afronta numerosos retos que no se habían planteado hasta el momento. En primer lugar, los vehículos

(que serán nodos de comunicación) se desplazan a una alta velocidad (incluso por encima de los 100 km/h) y con distintos itinerarios. Esto provoca que las redes se formen con un conjunto de nodos muy volátil. Por otro lado, pueden concentrarse numerosos vehículos en una misma área (por ejemplo, en un semáforo o en un atasco), lo que da lugar a redes con un elevado número de nodos.

Las comunicaciones vehiculares se establecen generalmente ad-hoc, siendo los nodos los propios vehículos o la infraestructura de comunicaciones (si existe) situada a lo largo de la vía. Los elementos de comunicación de la infraestructura se sitúan longitudinalmente a la vía, generalmente aprovechando los elementos de balizamiento o señalización vertical tradicional. Dada la extensión de la red de carreteras, no se puede asumir la existencia de dicha infraestructura en todo punto de la red viaria. Este hecho limita fuertemente las suposiciones sobre los servicios accesibles desde el vehículo. Desde el punto de vista de la seguridad, es importante destacar que no se dispone de terceros de confianza (autoridades de sellado de tiempo o de certificación) permanentemente accesibles.

2.1. El estándar IEEE 1609.2

Las características especiales que se han descrito han motivado la creación de una variante del estándar IEEE 802.11 de comunicación inalámbrica. Esta se conoce como WAVE (Wireless Access in Vehicular Environments). La familia de estándares IEEE 1609 define la arquitectura y los servicios necesarios para posibilitar las comunicaciones vehiculares.

En particular, la norma IEEE 1609.2 define los servicios de seguridad para aplicaciones de este entorno [2]. Las características de esta norma, en lo relativo a la propuesta que se presenta, son las siguientes:

- *Modelos de comunicación.* Se recogen dos tipos de comunicación. El primero permite la oferta de servicios al resto de vehículos y su aceptación por parte de estos. El segundo se basa en el intercambio de mensajes cortos (WSM, Wave Short Messages). Mientras que la intención de aquéllos es establecer una sesión entre comunicantes, los WSM pueden utilizarse esporádicamente.
- *Algoritmos criptográficos.* Se establece la utilización de ECIES (para cifrado asimétrico), ECDSA (para firma) y AES en modo CCM (para cifrado simétrico). Para compartir la clave simétrica, se debe realizar un ensobrado digital con ECIES.
- *Certificación.* Los nodos disponen de certificados de clave pública. Para comprobar su estado se utilizan listas de certificados revocados (CRL).
- *Tipos de mensajes.* Los mensajes pueden viajar en claro, firmados o cifrados. Así mismo, es posible enviar mensajes cifrados y firmados.
- *Datos accesibles.* Se asume que cualquier unidad de comunicaciones WAVE conoce su posición y el tiempo actual (y una estimación del error cometido). Para firmar, se asume que se dispone de la clave criptográfica, los certificados asociados, una implementación de los algoritmos de firma y un generador de números aleatorios criptográficamente seguro.

- *Aseguramiento adicional de las aplicaciones.* Las aplicaciones pueden prevenir los ataques de repetición utilizando sellos de tiempo (*time stamps*). Además, se permite que las aplicaciones no envíen mensajes firmados cuando el margen de error en el tiempo o posición actual sea demasiado grande.
- *Aspectos de privacidad.* El estándar no aborda ningún mecanismo para conseguir el envío anónimo de mensajes autenticados. Por otra parte, el estándar se centra en proteger los datos contenidos en los mensajes y no contempla el anonimato en los encabezados de los paquetes. Por ello, pueden producirse pérdidas de privacidad derivadas de la revelación de cierta información estática (e.g. direcciones IP fijas, etc.)

3. Evidencias electrónicas en el ámbito vehicular

Antes de abordar cómo crear evidencias en el ámbito vehicular es preciso, en primera instancia, determinar qué es una evidencia electrónica. Tras abordarlo en el primer apartado, el segundo describe para qué pueden servir y el tercero qué elementos mínimos se necesitan para su creación.

3.1. Evidencias electrónicas. Definición y requisitos

El término “evidencia electrónica” dispone de numerosas definiciones en la literatura. En el contexto de este trabajo, dicho término se refiere a *“cualquier registro generado por, o almacenado en, un sistema computacional que puede ser utilizado como prueba en un proceso legal”* [3].

Partiendo de esta definición, se habla de evidencia cuando se cree que las conclusiones derivables de esa información son indiscutibles y que, por tanto, el juez aceptará la evidencia como prueba [4]. Para que dicha aceptación sea posible y a pesar de la heterogeneidad legislativa existente, se establecen habitualmente cuatro principios para la admisibilidad de las pruebas [5]:

- Autenticidad y Confiabilidad, es decir, que la prueba sea auténtica y contenga datos fiables.
- Suficiencia, es decir, que sea completa para acreditar el hecho.
- Conformidad con las leyes, especialmente, que se haya obtenido sin violar otros derechos.

Estos principios constituyen requisitos necesarios para que la evidencia tenga peso como prueba en un proceso judicial. El juez o magistrado es (habitualmente) el encargado de verificar el cumplimiento de dichos principios, así como de valorar la relevancia de lo aportado dentro del proceso en curso. Por este motivo, para la utilización de evidencias electrónicas en procesos policiales o judiciales debe extremarse la corrección técnica en todo el proceso de custodia.

3.2. Usos previstos de las evidencias vehiculares

Para satisfacer, en el ámbito vehicular, las exigencias planteadas en el apartado anterior, se debe conocer qué se pretende acreditar a través de las evidencias. Por ello, algunos de sus usos previstos son los siguientes:

1. Acreditación espacio-temporal, demostrar que un determinado vehículo se encontraba en un determinado lugar en un momento concreto.
2. Demostración de correcto comportamiento, por ejemplo, ante una sanción por exceso de velocidad, siempre que esta se notifique inmediatamente.
3. Determinación de las circunstancias de un accidente, por ejemplo, si se activó o no el intermitente en un cambio de carril que provocó una colisión.

3.3. Elementos necesarios para crear evidencias vehiculares

Para construir estas evidencias se necesita obtener, de acuerdo al principio legal de *suficiencia* antes introducido, cuantos datos sean necesarios para acreditar el suceso. Así, además del momento de la evidencia, será necesario recabar datos adicionales, tales como información de posicionamiento (GPS, etc.) o datos procedentes de los sensores incorporados en el vehículo. Todas estas fuentes de datos son susceptibles de falsificación. Los sistemas de posicionamiento son susceptibles de ataques basados en la alteración de la señal (e.g. *jamming* [6]), entre otros. Por su parte, la seguridad física de las redes sensoriales en los vehículos no es muy elevada. Aunque se han propuesto algunas medidas de protección, estas no han sido globalmente implementadas en los sistemas de automoción [7]. De esta manera, se hace posible que el propietario de un vehículo manipule los datos para así dibujar una falsa realidad más propicia a sus intereses.

Afortunadamente, se pueden paliar estas deficiencias utilizando las redes vehiculares, pues los propios vehículos del entorno pueden servir como fuente de datos. Se necesita que dichos vehículos (que llamaremos *testigos*) sean mayoritariamente *participativos* y *honestos*. Se requiere que los testigos participen (es decir, ofrezcan su testimonio al solicitante) para que se dispongan de los datos externos que permitan crear las evidencias. Por otro lado, se requiere que sean honestos para que los datos que ofrezcan no hayan sido maliciosamente manipulados en origen. Dicha honestidad no asegura que los testimonios describan fielmente la realidad: siguen siendo posibles errores sensoriales derivados de las propias técnicas de medición utilizadas. Sin embargo, habitualmente es posible cuantificar el grado de incertidumbre propio de cada medida, lo que permite dar la credibilidad adecuada a cada testimonio.

Para cerrar el ciclo de custodia de la evidencia, una vez obtenidos los datos es preciso almacenarlos de forma fiable. Los vehículos deben disponer de capacidad computacional para realizar tareas criptográficas (esencialmente firma digital, para asegurar origen e integridad). Además, deben disponer de una fuente fiable de tiempo para añadir información temporal a la evidencia. Todas estas cualidades se integran en los dispositivos Hardware Security Module (HSM) que ya se han propuesto en el ámbito vehicular [8].

En esta sección se han abordado los requisitos esenciales para crear evidencias en el entorno vehicular. Se necesita disponer de fuentes confiables de datos, así como un almacenamiento seguro para dicha información. Es preciso diseñar un protocolo que permita la transferencia de dicha información entre los testigos y el solicitante, aspecto que se abordará en el siguiente apartado.

4. Protocolo de creación de evidencias vehiculares

Teniendo en cuenta las peculiaridades del entorno vehicular (apartado 2) y los aspectos particulares sobre evidencias electrónicas (apartado 3), en este apartado se propone un protocolo para crear evidencias electrónicas en este contexto. Para ello, primeramente se introduce la filosofía subyacente del protocolo propuesto. Posteriormente se enuncian las suposiciones necesarias para implantarlo y finalmente se presenta el protocolo en sí mismo.

4.1. Idea general del protocolo

El protocolo persigue la creación de evidencias sobre el comportamiento reciente de un vehículo. Sin embargo, como ya se ha dicho, el vehículo en cuestión puede manipular los datos para obtener una descripción de la situación más favorable para sus intereses. Por este motivo, se recurre a la obtención distribuida de los datos desde los vehículos del entorno. Debido a la alta movilidad de los vehículos, un protocolo de intercambio de información demasiado extenso podría provocar la pérdida de conectividad del solicitante con los vehículos aptos para ser testigos. Por este motivo, el protocolo requiere únicamente dos transferencias de datos: petición de información (solicitud) y envío de estimaciones (respuesta). Estas fases aparecen representadas en la Figura 1.

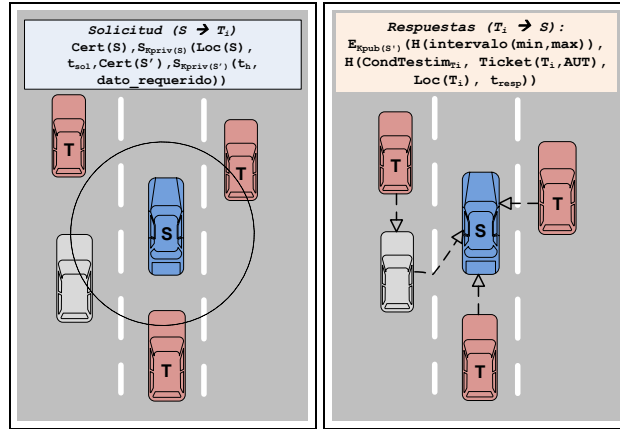


Figura 1. Fases del protocolo de creación de evidencias.

Es importante observar que no todos los vehículos disponen de esa información. En otras palabras, no todos los vehículos del entorno son aptos para ser testigos. En la Figura 1, el vehículo azul (marcado con 'S') es el solicitante de la evidencia, mientras que los rojos (marcados con 'T') son posibles testigos. El vehículo blanco no dispone de información sobre el solicitante y representa

los vehículos del entorno que participan como intermediarios en la comunicación debido a la lejanía de los comunicantes.

Un aspecto interesante del protocolo es que se autoriza al solicitante a crear una evidencia que contenga el dato consensuado tomando en consideración los datos procedentes de los testigos. Gracias a esto, se simplifica la interpretación de la evidencia, siendo más clara la circunstancia que se quiere demostrar y su relevancia para el proceso en el que se utilice. Para hacer posible la construcción de ese consenso, el testigo envía, junto con sus estimaciones acerca del hecho considerado (e.g. velocidad del solicitante), un *token* especial. Este token contiene en su interior las condiciones bajo las cuales el solicitante puede construir el consenso utilizando el dato aportado por el testigo. Dichas condiciones expresan, entre otras cosas, quién era el solicitante y cuáles fueron las estimaciones enviadas. Así, el solicitante no podrá (legítimamente) crear un consenso apoyado por el testigo utilizando un dato que no se ajuste a las estimaciones que este hizo.

Finalmente, debe destacarse que el protocolo crea una única evidencia autocontenida (e.d. contiene toda la información necesaria para verificarla). Esta cualidad incrementará su confiabilidad ante la autoridad a la que se presente.

4.2. Suposiciones sobre el entorno

Para que el protocolo propuesto sea factible en el entorno vehicular es preciso que existan cuatro requisitos mínimos, derivados de las necesidades expuestas para la creación de evidencias vehiculares (apartado 3.3):

- Existencia de una Autoridad de Certificación. El estándar IEEE 1609.2 exige que la autenticación de las entidades se realice mediante certificados. Por este motivo, se exige que exista una Autoridad de Certificación que se ocupe de la gestión de dichos certificados.
- Incorporación de una plataforma confiable en el vehículo. Es preciso contar con un almacenamiento confiable con capacidades criptográficas y una fuente fiable de tiempo. Se necesita, por tanto, un dispositivo HSM en el vehículo.
- Incorporación de sensores en el vehículo. El vehículo debe disponer de sensores (vídeo, radar, etc.) para conocer el estado de su entorno (estado de la vía, obstáculos, etc.). A través de los sensores es posible reconocer el comportamiento de otros vehículos, lo que servirá como fuente principal de datos para este protocolo [9]. Dichos datos son almacenados en el citado HSM.
- Mayoría honesta y atacantes no colaborativos. Los testigos deberán ser mayoritariamente honestos, lo que permitirá aislar los testimonios que difieran significativamente de las estimaciones de la mayoría. Los atacantes, por su parte, deben operar de manera independiente, es decir, no colaboran para construir un falso testimonio común.

Además de lo anterior, es conveniente que los vehículos utilicen las redes vehiculares para comunicarse datos útiles sobre el estado del tráfico. Existen dos paradigmas para llevar a cabo esta comunicación. Por un lado, los vehículos pueden enviar periódicamente mensajes de reporte de estado (*beaconing*).

El cometido de esos mensajes es ofrecer a los vehículos del entorno información sobre las circunstancias propias (posición, velocidad, estado, etc.). Por otro lado, gracias a la conducción cooperativa (e.g. Cooperative Cruise Control), los vehículos se agrupan en función de sus parámetros de conducción (e.g. destino, velocidad deseada). Ambas formas de comunicación son interesantes para este trabajo ya que permiten que el resto de vehículos conozcan más detalles sobre el comportamiento del vehículo solicitante.

Finalmente, a fin de preservar la identidad real de los participantes, se asume que los vehículos emplean seudónimos para su identificación. Estos deben poder ser resolubles por la Autoridad, a fin de obtener la identidad real de los implicados en caso necesario [10]. Además, se asume que los vehículos cambian periódicamente de seudónimo, para evitar su trazabilidad.

4.3. Notación

Para describir el protocolo se utilizará la notación siguiente. En primer lugar, los roles posibles son:

- S, el vehículo que solicita la creación de la evidencia (solicitante).
- T_i , vehículo (testigo) que tiene datos sobre el comportamiento reciente de S.
- AUT, la autoridad legal a la que se presentará la evidencia.

Las funciones criptográficas se denotan de la forma siguiente:

- $H(M)$, función resumen sobre 'M'. Incluye también el mensaje resumido (M).
- $S_K(M)$, firma de 'M' con la clave K y el algoritmo ECDSA. Incluye también el mensaje a firmar (M).
- $E_K(M)/D_K(M)$, cifrado/descifrado asimétrico de 'M' con la clave K y el algoritmo ECIES.
- $SE_K(M)$, cifrado simétrico de 'M' con la clave K y el algoritmo AES-CCM.
- $\text{Verify}(F, \text{Cert}(X))$, verificación del mensaje firmado 'F' utilizando la clave contenida en el certificado $\text{Cert}(X)$.

Finalmente, los datos intercambiados se denotan como sigue:

- $\text{ID}(X)$, el identificador (seudónimo) del vehículo X.
- $\text{Cert}(X)$, certificado de clave pública del vehículo X.
- $\text{intervalo}(a,b)$, expresión de los valores contenidos en el intervalo $[a,b]$, independientemente de la naturaleza de a y b .
- t_i , marca de tiempo del momento i .
- $\text{Loc}(X)$, coordenadas del lugar actual del vehículo X.

4.4. Descripción del protocolo

A continuación se expone el protocolo de creación de evidencias, junto con una breve explicación del cometido de los mensajes.

Fase 1. Solicitud de datos

En este mensaje, el solicitante envía un mensaje firmado (Signed WSM) indicando la información que se requiere (dato_ requerido) y el tiempo de la misma (t_h). Debido al cambio periódico de seudónimos, es posible que el actual del solicitante (S) fuera distinto del que se usaba en el momento al que se refieren los datos requeridos (S'). A través de este mensaje, el solicitante demuestra que esa era su identidad en el momento de los hechos.

$S \rightarrow T_i : \text{Cert}(S), S_{K_{priv}(S)}(\text{Loc}(S), t_{sol}, \text{Cert}(S'), S_{K_{priv}(S')}(t_h, \text{dato\_ requerido}))$

El mensaje se envía a todos los vehículos que se encuentren en un entorno razonable del solicitante (*geocasting*), utilizando para ello los datos de localización proporcionados por este. Con ello se evita la propagación de la solicitud a vehículos que no pueden disponer de dicha información (por estar situados demasiado lejos del solicitante).

Fase 2. Envío de respuestas

La segunda fase tiene como objetivo enviar las respuestas (testimonios) desde los vehículos testigos al solicitante. Esta fase solo tiene lugar si la información ha sido recibida de forma íntegra, lo que se asegura verificando las firmas de la solicitud. Igualmente, se verifica que tanto la localización como el tiempo de la solicitud toman valores razonables. En [11] se describe una forma de realizar estas comprobaciones. Si resultan satisfactorias, se realizan los siguientes pasos:

2.a. Estimación del dato basado en información del exterior.

Cada uno de los testigos estima cuál es el valor del dato requerido por el solicitante. Para ello, utiliza las informaciones que se le han proporcionado desde el exterior sobre el solicitante (beaconing, Cooperative Cruise Control). Usando estas fuentes de información, se crea un intervalo posible.

$T_i : \text{beaconing}(S') \times \text{Coop. Cruise Control}(S') \Rightarrow \text{intervalo\_ posible}$

2.b. Contraste de hipótesis.

Utilizando como entrada el intervalo obtenido en el paso anterior (intervalo_ posible), se utiliza el conocimiento propio (sobre la propia situación y sobre el solicitante) para establecer si el intervalo anterior es razonable. Para ello se emplean las mediciones obtenidas desde los propios sensores (radar, vídeo, ordenador a bordo, etc.). Supongamos que el solicitante desea que los testigos acrediten su velocidad. El intervalo_ posible (del paso anterior) es (100,120). Si el testigo circulaba en t_h a 110 y los sensores detectan que ese vehículo fue adelantado, el subintervalo de velocidades (110,120) queda descartado.

$T_i : \text{intervalo\_ posible} \times \text{informacion\_sensores} \Rightarrow \text{intervalo}(\text{min}, \text{max})$

El diseño de este cálculo es una tarea compleja, pues requiere otorgar credibilidad a múltiples fuentes de datos. En [12] y [13] se analizan diversas técnicas para abordar esta cuestión.

2.c. Envío de respuesta.

Si el intervalo construido en el paso anterior contiene algún valor, se procede a enviar la respuesta. La respuesta contiene tres partes. La primera va dirigida al solicitante y contiene la estimación sobre el dato_ requerido. Dicha estimación está cifrada (por confidencialidad) utilizando la clave del solicitante en el momento de los hechos (e.d. $K_{pub}(S')$). La segunda parte va dirigida a la Autoridad

a la que se presente la evidencia. Esta parte (denotada como Ticket) contiene la clave $K_{Testim(T_i)}$:

$$\text{Ticket}(T_i, \text{AUT}) = E_{K_{pub}(\text{AUT})}(\text{Cert}(T_i), S_{K_{priv}(T_i)}(K_{Testim(T_i)}))$$

$K_{Testim(T_i)}$ se utiliza para cifrar las condiciones bajo las que el solicitante puede crear un consenso utilizando el dato proporcionado por el testigo. Dichas condiciones cifradas conforman el token CondTestim_{T_i} . El solicitante podrá usar dicho token para reflejar que el consenso alcanzado está avalado por las observaciones efectuadas por el testigo.

$$\text{CondTestim}_{T_i} = SE_{K_{Testim(T_i)}}(\text{ID}(T_i), \text{ID}(S'), \text{intervalo}(\text{min}, \text{max}), t_h, t_{resp})$$

Es preciso destacar que gracias a esta distribución de la información, la identidad del testigo queda oculta para el solicitante. Esto resulta especialmente beneficioso en caso de que el testimonio no resulte favorable para el solicitante, lo que podría originar represalias contra aquel. La respuesta definitiva es:

$$T_i \rightarrow S: E_{K_{pub}(S')}(\text{H}(\text{intervalo}(\text{min}, \text{max})), \text{H}(\text{CondTestim}_{T_i}, \text{Ticket}(T_i, \text{AUT})), \text{Loc}(T_i, t_{resp}))$$

Fase 3. Generación de evidencia.

El solicitante verifica que la información espacio-temporal del testigo es razonable, así como la integridad de la estimación recibida. Si es así, calcula el dato de consenso entre todos los testigos, es decir, $\text{datoConsenso} \in \text{intervalo}_{T_i} \forall i$. Si no es posible dicho consenso, se busca el valor que esté contenido en la mayor cantidad posible de observaciones de los testigos.

Usando ese dato, se crea la evidencia definitiva ($\text{EV}(S)$), incluyendo los tokens de autorización de los distintos testigos. Para n testigos, la evidencia queda:

$$\text{EV}(S) = \text{Cert}(S), S_{K_{priv}(S)}(\text{datoConsenso}, t_h, t_{evid}), (\text{Ticket}(T_1, \text{AUT}), \text{CondTestim}_{T_1}, \dots), \text{Ticket}(T_n, \text{AUT}), \text{CondTestim}_{T_n})$$

5. Protocolo de verificación de la evidencia

En esta sección se presenta el protocolo de verificación asociado a la evidencia creada. La notación empleada es la misma que la expuesta en el apartado 4.3. Excepto la Fase 1, todas se realizan sobre cada uno de los testimonios contenidos en la evidencia. La verificación se supera si todas las fases se superan.

Fase 1. Verificación de firma de $\text{EV}(S)$

En primer lugar, se comprueba la firma sobre la evidencia, para asegurar su origen e integridad.

$$\text{AUT} : \text{Verify}(\text{EV}(S), \text{Cert}(S))$$

Fase 2. Descifrado de Ticket y obtención de condiciones del testimonio

El Ticket contiene la clave de protección de las condiciones del testimonio, por lo que es necesario obtenerla.

$$\text{AUT} : D_{K_{priv}(\text{AUT})}(\text{Ticket}(T_i, \text{AUT})), \text{ con lo que se obtiene } (\text{Cert}(T_i), S_{K_{priv}(T_i)}(K_{Testim(T_i)})).$$

Si la verificación de la firma sobre $K_{Testim(T_i)}$ es correcta, se utiliza dicha clave para obtener las condiciones bajo las que se facilitó el testimonio:

$$\text{AUT} : SE_{K_{Testim(T_i)}}(\text{CondTestim}_{T_i}), \text{ obteniendo } (\text{ID}(T_i), \text{ID}(S'), \text{intervalo}(\text{min}, \text{max}), t_h, t_{resp})$$

Fase 3. Verificación del estado de los certificados

La autoridad comprueba que los certificados eran válidos (no caducados ni revocados) en los momentos en que se utilizaron (t_{resp} para el testigo, t_{evid} para el solicitante).

Fase 4. Verificación del cumplimiento de las condiciones del testimonio**4.a** Verificación de identidades

Debe comprobarse que el solicitante coincide con el autorizado por el testigo (a pesar de que puede haber cambiado de seudónimo). También debe comprobarse que el testigo y el solicitante son entidades distintas, así como los distintos testigos entre sí. Igualmente, los usos previstos de las evidencias exigen obtener la identidad real del solicitante, por lo que deberá resolverse el seudónimo. Para realizar estas operaciones, la autoridad que reciba la evidencia deberá contactar con la entidad encargada de la gestión de los seudónimos.

4.b Dato y momento legítimos

Se debe verificar que el dato consensuado está contemplado en las estimaciones del testigo. Además, se debe comprobar que el momento al que se refieren los testimonios es el momento del dato consensuado. También se comprueba que t_{evid} sea posterior (y no muy lejano) a t_h .

6. Análisis de seguridad

Tras la presentación de los protocolos propuestos, es necesario realizar un análisis de seguridad sobre la propuesta. En este trabajo resulta especialmente importante analizar si se satisfacen las propiedades necesarias para su admisibilidad como prueba (Sección 3.1). Si bien el cumplimiento del principio de *suficiencia* es responsabilidad del solicitante (que debe requerir los datos necesarios), el resto de principios dependen de la realización del protocolo descrito.

6.1. Autenticidad de la evidencia

Es preciso garantizar que la evidencia es auténtica. Para ello, se discutirá acerca de la integridad y el no repudio de la información en juego, así como la autenticación de las entidades participantes.

La *integridad* de la solicitud puede comprobarse gracias a las firmas incluidas. La integridad de los testimonios es también verificable gracias al uso de funciones resumen. De esta manera, es posible comprobar alteraciones maliciosas sobre los mensajes producidas por un intermediario. En lo referente al *no repudio*, en emisión está garantizado (tanto en los testimonios como en la evidencia) gracias al uso de firmas digitales. Sin embargo, el no repudio en recepción no está contemplado al no ser necesario en el contexto del protocolo.

Por lo que respecta a la *autenticación* de las entidades, tanto el solicitante como los testigos utilizan los certificados impuestos por el estándar. Sin embargo, la certificación conlleva ciertos problemas que deben ser considerados. En primer lugar, desde el punto de vista del testigo, si el certificado del solicitante

es revocado con anterioridad a la solicitud, es posible que no disponga de esa información. Esto origina que los testigos pueden participar en una ejecución inútil del protocolo, puesto que la evidencia creada carecerá de valor probatorio (al no ser confiable el solicitante). Esta amenaza está motivada por la compleja distribución de la información del estado de los certificados. Para paliarlo se podría utilizar un protocolo específico para la distribución de esta información [14].

6.2. Confiabilidad de la evidencia

Se debe asegurar que la evidencia contiene datos confiables. Gracias al uso de testigos (y bajo el supuesto de mayoría honesta), se consigue una fuente de datos presumiblemente imparcial. El consenso alcanzado por el solicitante se verifica utilizando los elementos de información contenidos en la propia evidencia. Sin embargo, desde el punto de vista del solicitante existen dos problemas que el protocolo actual no soluciona. En primer lugar, un testigo deshonesto podría dar una estimación al solicitante distinta de la incluida en CondTestim_{T_i} . De esta manera, el solicitante puede crear una evidencia que carezca de valor al no ajustarse (involuntariamente) a las supuestas estimaciones del testigo. Esta desventaja para el solicitante también puede manifestarse en que se reciban varios testimonios desde el mismo testigo, empleando distintos seudónimos. La comprobación por parte de la autoridad revelará este hecho y restará credibilidad a la evidencia, aunque en principio no debería ser causa de anulación.

Ambas cuestiones constituyen amenazas presentes, aunque de impacto presumiblemente limitado. Para su realización sería necesario alterar la plataforma confiable, o crear un componente alternativo que llevara a cabo el ataque. Además, no parece existir una motivación clara para este comportamiento, ya que requiere un procesamiento no despreciable para obtener un dudoso beneficio.

Por otra parte, la frescura de los testimonios se comprueba en la verificación de la evidencia. Dado que el HSM dispone de una fuente fiable de tiempo, vehículos no presentes no tienen capacidad para crear testimonios “a posteriori”. Estas marcas de tiempo protegen también al protocolo de ataques de repetición.

6.3. Conformidad con las leyes. Privacidad

La evidencia debe obtenerse sin violación de otros derechos. En este contexto, la privacidad es el derecho más amenazado. De hecho, a nivel conceptual, el solicitante pide a los testigos que le confirmen cuál fue su comportamiento en el pasado. La trazabilidad (del solicitante) es una necesidad en el protocolo. Sin embargo, esta violación de privacidad (inherente al protocolo) se ve compensada por el beneficio obtenido, la evidencia creada. Además, las estimaciones sobre su comportamiento se encuentran cifradas, por lo que se evita que el resto de vehículos conozcan más datos sobre el solicitante. Por su parte, la identidad de los testigos se encuentra protegida. Como ya se dijo, permanecen anónimos frente al solicitante. Además, según la descripción del protocolo de verificación, no es necesario resolver la identidad del testigo, con lo que se evita su trazabilidad por parte de la autoridad.

6.4. Otras amenazas posibles y riesgo de materialización

En primer lugar, la *disponibilidad* de las capacidades de comunicación de los vehículos puede quedar comprometida si el protocolo de creación de evidencias se utiliza frecuentemente. De hecho, todas las fases de dicho protocolo conllevan un cierto procesamiento criptográfico, por lo que es posible realizar un ataque de denegación de servicio mediante inundación de solicitudes. La utilidad de este tipo de ataque vendría determinada por la finalidad de la evidencia. Si, por ejemplo, se pretende obtener una evidencia que acredite una circunstancia tras la ocurrencia de un accidente, la parte culpable intentará evitar la creación de pruebas. Como contramedida, el protocolo requiere tan solo dos mensajes, para evitar en lo posible la sobrecarga del medio.

Los vehículos intermediarios también pueden afectar al desarrollo del protocolo. Si no se propaga la solicitud, o si no se encaminan adecuadamente los testimonios, el protocolo de creación pierde efectividad. Para disminuir el riesgo de esta amenaza se podría utilizar un mecanismo de prevención de comportamientos no colaborativos [15].

Por otro lado, el token de autorización del consenso CondTestim_{T_i} es el resultado de cifrar las condiciones válidas de consenso con una clave simétrica ($K_{\text{Testim}(T_i)}$). Si dicha clave se reutiliza en varias respuestas consecutivas a un mismo solicitante, es posible realizar un *ataque de texto en claro conocido*. No obstante, asumiendo que los vehículos están en movimiento, es improbable que se produzcan los intercambios necesarios para comprometer la clave. En cualquier caso, se recomienda que la clave $K_{\text{Testim}(T_i)}$ cambie periódicamente.

7. Trabajos relacionados

El aseguramiento de las comunicaciones vehiculares se ha abordado de forma extensa en la literatura [16]. No obstante, y hasta donde se ha desarrollado esta investigación, la creación de evidencias en el entorno vehicular ha sido escasamente abordada. Las propuestas más relevantes se centran en la reconstrucción de accidentes. En [17] se aborda este problema utilizando la plataforma confiable (la citada HSM) como elemento de registro de datos, a modo de “caja negra”. Tras la ocurrencia del accidente, los vehículos implicados envían mensajes de advertencia, utilizando sus propias mediciones sensoriales como fuente de datos. Como ya se ha dicho, dichas fuentes podrían ser manipuladas, por lo que podría cuestionarse la validez de las evidencias que se pudieran derivar.

Por otra parte, en [18] se aborda la reconstrucción de accidentes utilizando datos propios y aquellos recibidos en las comunicaciones intervehiculares. Aunque nuestro enfoque es similar, en el presente trabajo se asume que los vehículos pueden obtener datos del estado de su entorno, lo que enriquece el proceso. Además, las aplicaciones objetivo de la presente propuesta se extienden más allá de la reconstrucción de accidentes.

La generación de evidencias ha sido también abordada dentro del marco de seguridad propuesto por Lin *et al.* [19]. Este marco se centra en proporcionar

seguridad y privacidad, junto con la debida trazabilidad en caso necesario. Para ello utilizan criptografía basada en identidad y firmas en grupo. Si bien esta última herramienta podría incorporarse a la propuesta que se presenta, para la autenticación se ha preferido utilizar certificados de clave pública, siguiendo las directrices marcadas por el estándar de seguridad en redes vehiculares.

Finalmente, las propuestas de creación de grupos podrían verse como un medio para la selección de testigos. Este tipo de técnicas han sido extensivamente abordadas, e incluso se han propuesto mejoras destinadas a mejorar la eficiencia de la comunicación dentro del grupo. De forma significativa, Raya *et al.* [20] abordan la agregación de valores de los miembros de un grupo. A pesar de ello, el contexto de uso de la presente propuesta requiere contactar con los vehículos aptos para ser testigos, es decir, aquellos que disponen de información sobre el comportamiento pasado del solicitante. Teniendo en cuenta la volatilidad de los grupos formados, es posible que no todos los testigos formen parte del grupo en el momento en que se solicita la creación de la evidencia. Por este motivo, no parece adecuado utilizar la formación de grupos para la selección de testigos.

8. Conclusiones y trabajo futuro

En este trabajo se ha presentado un protocolo para la creación de evidencias sobre el comportamiento de un vehículo utilizando datos procedentes de algunos vehículos del entorno. Gracias a esta nueva forma de creación de evidencias se obtiene una descripción de la realidad tal y como la observaron los vehículos cercanos, que actúan como testigos. De esta manera, se evita que los datos de la evidencia provengan de los sensores del propio vehículo, que podrían ser maliciosamente modificados según los intereses de su propietario. El protocolo propuesto sólo requiere de dos transferencias de información (solicitud y respuestas), lo cual resulta clave para su idoneidad en este escenario.

Además de este protocolo de creación, se ha descrito el correspondiente protocolo de verificación de la evidencia. El análisis de seguridad sobre la propuesta ha mostrado que se satisfacen las propiedades adecuadas para el contexto de utilización del protocolo. Se han detectado, sin embargo, algunas carencias de seguridad derivadas del comportamiento potencialmente deshonesto de los testigos participantes. El trabajo futuro estará centrado, en primer lugar, en incorporar mecanismos frente a los problemas detectados. Igualmente, está previsto incluir mecanismos basados en incentivos para fomentar la participación, de forma que las solicitudes estén condicionadas a la satisfacción de un cierto pago. Así, se pueden diseñar pagos positivos a aquellos testigos que ofrezcan un testimonio valioso. Igualmente, los pagos negativos pueden servir para evitar testimonios deshonestos. Esta medida puede extenderse también a la labor de los intermediarios. Por otro lado, se pretende incorporar un mecanismo de agregación que permita reducir el número de respuestas (testimonios) que recibe el solicitante, garantizando al mismo tiempo la seguridad de la información en juego.

Agradecimientos

Este trabajo ha sido parcialmente financiado por el Ministerio de Ciencia e Innovación (España), dentro del Plan Nac. de Investigación Científica, Desarrollo e Innovación Tecnológica 2008-2011, contrato TIN2009-13461 (proy. E-SAVE).

Referencias

1. DIRECCIÓN GENERAL DE TRÁFICO (España), “*Accidentes mortales en carretera. Año 2008*”, Observatorio Nacional de Seguridad Vial, 2009.
2. *IEEE Trial-Use Std. for Wireless Access in Vehicular Environments. Security Services for Applications and Management Messages* (1609.2). IEEE CS, 2006.
3. CANO, J.J., “Introducción a la informática forense”. En: *Revista Asociación Colombiana de Ing. de Sistemas (ACIS)*.2006, núm. 96, pp. 64-73.
4. LÓPEZ-SILVES, A., “*Las evidencias electrónicas*”, TyD Consultores, [s.f.].
5. LLANEZA, P.; LÁZARO, F., “Evidencias electrónicas: de la información a la prueba electrónica”. En: *Rev. Seg. en Informática y Com. (SIC)*,2009, núm. 83, pp. 92-97.
6. XU, W. *et al.*, “The feasibility of launching and detecting jamming attacks in wireless networks”. En: *Proc. MobiHoc’05*,ACM, 2005.
7. WOLF, M.; WEIMERSKIRCH, A.; PAAR, C., “Security in automotive bus systems”. En: *Proc. 2nd Workshop on Embedded Security in Cars (ESCAR)*, 2004.
8. KARGL, F. *et al.*, “Secure vehicular communication systems: implementation, performance, and research challenges”. En: *IEEE Communications*, 2008, v. 46, n. 11.
9. GOLLE, P.; GREEN, D.; STADDON, J., “Detecting and correcting malicious data in VANETs”. En: *Proc. 1st ACM Intl. Workshop on VANETs*. 2004, pp.29-37.
10. PAPADIMITRATOS, P. *et al.*, “Architecture for Secure and Private Vehicular Communications”, En: *Proc. 7th Intl. Conf. on ITS*, 2007, pp.1-6.
11. NAI-WEI, L.; HSIAO-CHIEN, T., “Illusion Attack on VANET Applications. A Message Plausibility Problem”.En: *Globecom Workshops*, IEEE CS, 2007, pp. 1-8.
12. WEX, P. *et al.*, “Trust Issues for Vehicular Ad Hoc Networks”. En: *Vehicular Technology Conf.*, 2008. IEEE CS, pp. 2800-2804.
13. RAYA, M. *et al.*, “On Data-Centric Trust Establishment in Ephemeral Ad Hoc Networks”. En: *INFOCOM 2008. 27th Conf. on Computer Comm.*. IEEE CS, 2008.
14. RAYA, M.; PAPADIMITRATOS, P.; HUBAUX, J.-P., “Securing vehicular communications”. En: *Wireless Comm.*, IEEE CS. 2006, vol.13, núm.5, pp.8-15.
15. BUTTYÁN, L.; HUBAUX, J.-P., “*Security and cooperation in wireless networks*”, Cambridge University Press, 2007.
16. RAYA, M.; HUBAUX, J.-P., “Security aspects of inter-vehicle communications”. En: *Proc. 5th. Swiss Transport Research Conf.*, 2005.
17. RAHMAN, S.U.; HENGARTNER, U., “Secure crash reporting in vehicular Ad hoc networks”. En: *Proc. 3rd Intl. Conf. on Security and Privacy in Comm. Networks*. IEEE CS, 2007, pp.443-452.
18. YOUNG, C-P.; CHANG, B.; LIN, J-J.; FANG, R-Y., “Cooperative collision warning based highway vehicle accident reconstruction”. En: *Proc. 8th Intl. Conf. on Intelligent Systems Design and Applications*,2008.
19. LIN, X.; SUN, X.; HO, P-H.; SHEN, X., “GSIS: A secure and privacy-preserving protocol for vehicular communications”. En: *IEEE Transactions on vehicular technology*, vol. 56, núm. 6, Noviembre 2007.
20. RAYA, M.; AZIZ,A.; HUBAUX, J.-P., “Efficient secure aggregation in VANETs”. En: *Proc. 3rd Intl. workshop on VANETs*. 2006. pp.67-75.

Control de Calidad en Imágenes de Iris mediante Razonamiento Ontológico

Alberto de Santos Sierra¹, Javier Guerra Casanova¹, Carmen Sánchez Ávila¹ y
Vicente Jara Vera²

¹ Grupo de Biometría y Tratamiento Numérico de la Información.
Centro de Domótica Integral (CeDInt)
<alberto@cedint.upm.es>
<jguerra@cedint.upm.es>
<csa@mat.upm.es>

² Dpto. de Matemática Aplicada a las Tecnologías de la Información.
ETSI de Telecomunicación. Universidad Politécnica de Madrid.
Ciudad Universitaria s/n, 28040 Madrid
<vjara@mat.upm.es>

Resumen La segmentación es uno de los pasos que consume más tiempo en el proceso de reconocimiento de Iris. Mediante Morfología Matemática y Redes Neuronales, este algoritmo consigue aislar el Iris dentro de la imagen del ojo, no sólo con una alta precisión, sino también con una gran reducción en el tiempo y en la carga computacional. Además, este innovador esquema presenta una ontología capaz de decidir si la extracción de características posterior a la segmentación puede llevarse a cabo. Los algoritmos de reconocimiento de Iris podrán beneficiarse de este nuevo enfoque, y además, la esencia del algoritmo puede ser extendida a otros procesos de segmentación biométrica.

Palabras clave: Biometría, Reconocimiento de Iris, Segmentación, Lógica Difusa, Morfología Matemática, Redes Neuronales, Razonamiento Ontológico.

1. Introducción

El reconocimiento de Iris engloba diferentes aspectos desde que se captura la imagen, hasta que el sistema es capaz de rechazar o aceptar al usuario que pretende acceder al sistema. Dentro de todos estos pasos, la segmentación es el que más tiempo consume en el proceso completo, siendo esta la razón por la cual se debe hacer especial esfuerzo en reducir no sólo el tiempo de computación sino también la complejidad del procedimiento.

Algunas soluciones han sido implementadas para resolver este problema. La mayoría de ellas se aprovechan de la geometría circular del Iris y de la Pupila, y de que ambos son casi concéntricos. La estrategia consiste en detectar el Iris, la Pupila y los párpados, [1], [2], [3], tomando como solución la intersección de

todos ellos. De este enfoque se derivan algunos problemas, ya que algunas veces el Iris en general está escondido en parte debajo del párpado, haciendo que aislar el Iris, sea algo bastante difícil para llevar a cabo.

El enfoque nuevo propuesto en este documento únicamente se centra en la parte del Iris que se puede ver, en otras palabras, qué sección de Iris no se encuentra escondida debajo del párpado, mediante Morfología Matemática, [4], y Redes Neuronales Artificiales, [5].

Una vez que la segmentación se ha llevado a cabo, un proceso de decisión basado en Ontologías ha sido implementado para dilucidar si de la imagen segmentada con anterioridad pueden ser extraídas las características necesarias para elaborar el patrón biométrico que posteriormente servirá para identificar al individuo, [3].

El documento continúa presentando primeramente el proceso de segmentación, y el control de calidad basado en razonamiento ontológico. Los resultados serán presentados tras estas secciones, considerando aspectos temporales y de precisión, terminando con las conclusiones y el trabajo futuro.

2. Segmentación de Iris

2.1. Detección de Pupila

El primer paso en el procedimiento de reconocimiento de Iris precisa de una correcta detección de la Pupila, [2], [3]. Considerando las características geométricas de la Pupila, (i.e., casi circular), es obviamente innecesario utilizar todo el contorno como resultado de la segmentación. Por el contrario, se propone el uso de únicamente cinco puntos: $pCenter$, $pVertical_{min}$, $pVertical_{max}$, $pHorizontal_{min}$ y $pHorizontal_{max}$.

Estos puntos, Fig. 1, engloban toda la información necesaria para afirmar no sólo dónde se encuentra la pupila sino también hasta qué punto se encuentra dilatada.

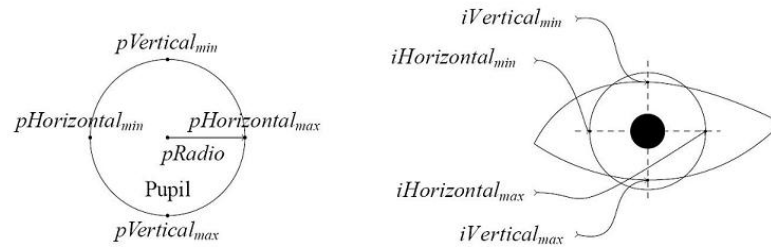


Figura 1. Puntos más importantes del Ojo: Puntos de Pupila (Izquierda), Puntos de Iris (Derecha).

Debe resaltarse que esta detección de Pupila únicamente considera la parte visible de la Pupila, es decir, si el párpado cubre parte de la Pupila, $pVertical_{min}$ sería más cercano a $pCenter$, haciendo más fácil la tarea de un Control de Calidad, tal y como se detallará en la Sección 2.4.

La segmentación podría incluso reducirse más aún y reducir el número de puntos requeridos para localizar la Pupila a únicamente dos: $pCenter$ y $pRadio$, ya que estos dos puntos pueden describir una circunferencia de forma precisa. Sin embargo, existen situaciones donde la Pupila puede estar desplazada en la imagen (debido a la concavidad ocular), perdiendo parte de sus propiedades circulares. Esto no quiere decir que la pupila deje de ser circular, sino que debido a la captura de la imagen, la perspectiva puede ofrecer una Pupila con cierta excentricidad, más similar a una elipse que a una circunferencia. En estas situaciones se requerirían más puntos para extraer toda la información de la Pupila, de ahí que se haya optado por la elección de los puntos anteriores.

Para llevar a cabo la segmentación, se precisa de operadores morfológicos de Apertura y Clausura, [4], y umbralización para detectar la parte visible de la Pupila dentro de una imagen.

Finalmente, en la Sección 3, se proporcionarán los resultados sobre la precisión de este algoritmo. Sin embargo, se puede adelantar que el procedimiento es capaz de detectar cada Pupila para todas las imágenes existentes en la base de datos CASIA (v1 y v3), [6], con ningún error y con un tiempo de computación despreciable.

El siguiente paso, la detección de Iris, precisará únicamente del punto $pCenter$, porque una de las grandes ventajas de este algoritmo se basa en el hecho de que el centro del Iris no es ni considerado ni detectado, decreciendo aún más el tiempo de computación del proceso completo.

2.2. Detección de Iris: Primera Aproximación

En este nuevo enfoque se detectará y aislará únicamente la parte visible del Iris dentro de una imagen. Se dividirá el proceso completo en dos aproximaciones: Primeramente, los puntos de Iris serán definidos y extraídos tal y como se hizo con la pupila anteriormente (Sección 2.1). Aunque esta aproximación pudiera poco precisa, proporciona una muy buena solución para la detección de Iris, debido al hecho de que los puntos de Iris encierra perfectamente el Iris completo.

En segundo lugar, una segunda aproximación aísla el Iris como en enfoques previos propuestos en la literatura, tomando la información extraída en la primera aproximación. Aunque este nuevo enfoque puede obtener el aislamiento del Iris completo (tal y como se verá en la siguiente sección), la primera aproximación proporciona al Control de Calidad suficiente información para llevar a cabo el rechazo o aceptación de la imagen capturada.

Una vez que se han presentado ambas aproximaciones, se explica la primera. En este caso, cuatro puntos para el Iris son definidos: $iVertical_{min}$, $iVertical_{max}$, $iHorizontal_{min}$ y $iHorizontal_{max}$. Estos puntos se corresponden con los límites del Iris visible en una imagen, tal y como puede verse en la Fig. 1.

En dicha Figura, se aprecia como el párpado cubre parte del Iris.

Para continuar con el algoritmo de detección de Iris, el lector debe ser consciente de ciertas propiedades geométricas en el Iris. Dos características han sido consideradas para este procedimiento:

- *pCenter* y el centro del Iris están suficientemente cerca como para ignorar la diferencia entre ambos
- El Iris y la Esclerótica³, son claramente diferente en términos de color.

Considerando la geometría circular del Iris, la idea de un filtro capaz de proporcionar valores altos (en términos de intensidad) cercanos al contorno de la pupila, y valores cada vez más bajos a medida que los píxeles se alejen del centro de la pupila, puede ofrecer un buen resultado a la hora de aislar el Iris. De hecho, dicho filtro toma *pCenter* como parámetro de entrada para obtener una mayor precisión a la hora de ubicar los filtros dentro de la imagen.

Se implementaron cuatro filtros para llevar a cabo tal idea. Tres de ellos dependen enormemente de la distribución elegida para aclarar las componentes de la imagen que se encuentran más alejadas de *pCenter*, dejando casi intactas las componentes más cercanas a la Pupila. Únicamente se diferencian en la distribución usada. En la Cuadro 1, se muestran los nombres de los filtros y sus expresiones matemáticas. Las coordenadas representadas por (x_0, y_0) se corresponden con *pCenter*.

Cuadro 1. Descripción de los Filtros.

Nombre	Expresión Matemática	Parámetros
Gaussian	$A_\gamma e^{-\sigma_x(x-x_0)^2} e^{-\sigma_y(y-y_0)^2}$	$A_\gamma, \sigma_x, \sigma_y, (x_0, y_0)$
Coseno I	$A_I \cos(x - x_0) \cos(y - y_0)$	$A_I, (x_0, y_0)$
Coseno II	$A_{II} \cos^2(x - x_0) \cos^2(y - y_0)$	$A_{II}, (x_0, y_0)$

Todas las amplitudes A_γ , A_I y A_{II} definen de qué forma el filtro difumina la imagen.

En la Fig. 2, se pueden ver las distribuciones de los filtros. El lector podrá notar ahora el efecto que tendría cada filtro en la imagen si el pico más alto de cada filtro es desplazado a *pCenter* de cada imagen. Esas componentes más alejadas de la Pupila serán atenuadas en mayor medida que aquellas más cercanas a la Pupila.

Una vez que los filtros previos han difuminado la imagen, se precisa de un filtro extra para eliminar esas componentes más claras dentro de la imagen. Este filtro está basado en Morfología Matemática y umbralización, y ofrece como resultado una máscara con las componentes más oscuras dentro de la imagen.

Aunque la ejecución del algoritmo no es tan precisa como en el caso de la detección de pupila, esta primera aproximación ofrece al Control de Calidad

³ Parte blanca del ojo

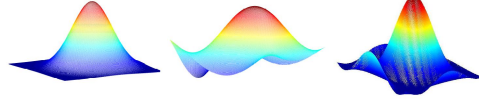


Figura 2. De izquierda a derecha: Gaussian, Coseno I y Coseno II.

posterior una información muy útil sobre el Iris, y de forma muy rápida, en otras palabras, puede indicar cómo de abierto está el párpado, si el Iris está desplazado en la imagen, si el Iris se encuentra dentro de la imagen, entre otros.

Finalmente, y acorde a lo sugerido en la sección anterior, los resultados de este algoritmo, así como de un ejemplo visual se proporcionarán en la Sección 3.

2.3. Detección de Iris: Segunda Aproximación

Como se vio con anterioridad, la primera aproximación proporciona al Control de calidad posterior la información suficiente para decidir la situación del Iris dentro de la imagen. Sin embargo, el enfoque ‘tradicional’ de la detección de Iris requiere un aislamiento completo del Iris, y algunas aplicaciones necesitan esta segmentación para poder funcionar correctamente. Para una mayor generalización de este algoritmo, se necesita la Segunda Aproximación para aislar completamente el Iris.

Este procedimiento es el paso final en el proceso completo de segmentación de Iris, aunque se ha sugerido en este documento que ese paso no es obligatorio para una segmentación precisa.

Para alcanzar este fin, es decir aislar completamente el Iris, se precisa de varios pasos. Primeramente, un Filtro con distribución Gaussiana es elegido para llevar a cabo la segmentación, ya que sus parámetros σ_x and σ_y , ver Cuadro 1, puede ser modificada y adaptada para delimitar el Iris.

Estos parámetros pueden ser fijados a mano. Sin embargo, esto es algo impensable para un sistema automático. El problema consiste, entonces, en relacionar estos parámetros con alguna medida ya realizada en etapas anteriores (por ejemplo, la distancia entre $pVertical_{min}$ y $iVertical_{min}$).

Algunos parámetros fueron propuestos y estudiados en este documento. Sin embargo, únicamente cuatro de estos parámetros fueron significativos en este estudio. Estos parámetros están recogidos en la Cuadro 2, y son definidos en términos de los puntos de Iris extraídos en la primera aproximación previa.

Esta distinción fue hecha en base a un estudio estadístico previo, [3], donde todas las variables fueron consideradas. Dichas variables relacionaban todas las posibles distancias entre los puntos de Iris, $pCenter$ y los puntos de Pupila. Además de esta conclusión, del estudio estadístico se puede obtener una conclusión mucho más importante: Las componentes horizontales de los parámetros en Cuadro 2, i.e. $i\Delta H_{max}$ y $i\Delta H_{min}$, estaban relacionadas fuertemente con σ_x y por otro lado, $i\Delta V_{max}$ y $i\Delta V_{min}$ estaba también relacionado con σ_y .

Para encontrar una función que relacionara los parámetros anteriores, dos redes neuronales, [5], [12], fueron implementadas. Esta red neuronal encontró la

Cuadro 2. Parámetros significativos relacionados con σ_x y σ_y .

Nombre	Definición
$i\Delta H_{max}$	Distancia entre $iHorizontal_{max}$ y $pCenter$
$i\Delta H_{min}$	Distancia entre $iHorizontal_{min}$ y $pCenter$
$i\Delta V_{max}$	Distancia entre $iVertical_{max}$ y $pCenter$
$i\Delta V_{min}$	Distancia entre $iVertical_{min}$ y $pCenter$

solución para ξ y ζ , cuya definición viene a continuación: $\sigma_x = \xi(i\Delta H_{max}, i\Delta H_{min})$ y $\sigma_y = \zeta(i\Delta V_{max}, i\Delta V_{min})$. De hecho, el error cuadrático medio (MSE, en inglés) para ambas redes es similar a $2 \cdot 10^{-3}$, algo insignificante si es comparado al error de precisión en las medidas de σ_x and σ_y , cuyos valores están cercanos a $\epsilon = \pm 0,01$.

Estas redes neuronales proporcionan una rápida computación de los valores de entrada σ_x y σ_y . Por lo tanto, la segunda aproximación únicamente necesita tiempo para computar los Filtros basados en distribuciones y el filtro encargado de eliminar las componentes más claras en la imagen. Un ejemplo visual de la ejecución de este algoritmo se presenta en la Sección 3, junto con un estudio temporal de dicho algoritmo.

2.4. Control de Calidad

El Control de Calidad, en el contexto de la segmentación de Iris, consiste en aceptar o rechazar una imagen de Iris en base a características extraídas de la imagen en sí misma, [3]. En este documento, un Control de Calidad basado en políticas se encargará de relacionar las características del Iris obtenidas en las secciones 2.1 y 2.2. Debido a este control, un programador (o usuario final) es capaz de definir diferentes políticas para aceptar o rechaza una imagen de Iris, o por el contrario, para tratar dicha imagen para poder mejorar su calidad. Estas políticas dependerán de los algoritmos usados para extraer información biométrica del Iris en etapas siguientes.

Consecuentemente, la ontología denominada *OntIrisQualityControl* ha sido desarrollada para almacenar la información sobre las políticas aplicadas a las imágenes de Iris. Cada vez que una imagen de iris es procesada, algunos parámetros importantes son almacenados en las instancias de la ontología. Entonces, el razonamiento sobre la ontología se realiza a través de reglas SWRL (Semantic Web Rule Language), [7], para obtener de forma automática el tipo de Control de Calidad que debe ser aplicado a la imagen de iris, dependiendo de las características del Iris y de la imagen procesada.

2.5. Ontología

La ontología *OntIrisQualityControl* para el Control de Calidad basado en políticas y en las imágenes de Iris, ha sido desarrollado con Protegé 3.4, [8]. Esta ontología esta compuesta por las siguientes clases:

- **ImageFeatures:** En esta clase, hay información sobre la calidad de la imagen. Cada vez que una imagen de iris es procesada, una nueva estancia ImageFeatures es creada, rellendo sus propiedades con valores obtenidos del estudio de algunas características de la imagen con diferentes algoritmos de tratamiento de imágenes, [4]. Estas características son seleccionadas puesto que son los factores más representativos en lo referente a la calidad de imágenes. Está compuesta además por las siguientes propiedades, que se expresan en porcentaje:
 - **Image_blur:** Medida de borrosidad de una imagen. Su rango está entre 0 y 100 %
 - **Image_distorsion:** Medida de distorsión de la imagen. Su rango está entre 0 y 100 %
 - **Image_occupied:** Medida de la porción ocupada por algunos objetos como sombras y párpados. Su rango está entre 0 y 100 %
 - **Image_noise:** Medida del ruido de la imagen.
 - **Image_outfocused:** Medida del desenfocado en la imagen.
- **IrisImage:** Clase con las características del iris bajo procesamiento. Cada vez que una imagen de iris es procesada, una nueva instancia de la clase IrisImage es generada, y sus propiedades son rellenas con los valores calculados con los puntos obtenidos en las Secciones 2.1 y 2.2. Las propiedades de IrisImage son:
 - **Iris_height_min:** Es el menor valor entre $|iVertical_{min} - pVertical_{min}|$ y $|pVertical_{max} - iVertical_{max}|$. Representa la mínima apertura vertical del Iris.
 - **Iris_width_min:** Es el menor valor entre $|iHorizontal_{min} - pHorizontal_{min}|$ y $|pHorizontal_{max} - iHorizontal_{max}|$. Hace referencia a la mínima apertura horizontal del Iris.
 - **Iris_axis_difference:** Es el valor $|pHorizontal_{max} - pHorizontal_{min}| - |pVertical_{max} - pVertical_{min}|$. Representa el grado en el que el Iris está centrado.
 - **Iris_radius_360_max:** Es el radio máximo de la corona circular que se puede extraer en el Iris.
 - **Iris_radius_256_max:** Es el radio máximo de la corona circular de 256° que se puede extraer en el Iris.
- **QualityControl:** Esta clase almacena las diferentes posibilidades del Control de Calidad a la hora de reaccionar frente a los diferentes tipos de imágenes e iris preprocesados. Una imagen puede ser rechazada, aceptada o tratada para mejorar su calidad. Esta clase está compuesta por las siguientes propiedades:
 - **qualityControl_type:** El tipo de Control de Calidad. Puede tomar estos valores:
 - Aceptada: La imagen es aceptada.
 - Rechazada: La imagen es rechazada.
 - Tratamiento: Antes de extraer la información de la imagen, se debe tratar previamente con algoritmos almacenados en la propiedad qualityControl_mode, [4].
 La descripción de estos algoritmos está fuera del alcance de este documento.

- **qualityControl_mode**: Algoritmo para tratar la imagen y mejorar sus características.
- **Policy**: Cualquier instancia de esta clase es una política de reacción para el Control de Calidad. Está compuesto por los mínimo o máximo valores que deben tener los parámetros de las instancias IrisImage e ImageFeature para aplicar un Control de Calidad específico. Los programadores finales definirán unas políticas de Control de Calidad dependiendo de los requisitos de los algoritmos usados para extraer la información biométrica de las imágenes de Iris. Las propiedades de esta clase casan con las propiedades de IrisImage e ImageFeatures:
 - policy_iris_width_min
 - policy_iris_height_min
 - policy_iris_axis_difference_min
 - policy_iris_radius_256_min
 - policy_iris_radius_360_min
 - policy_image_blur_max
 - policy_image_distorsion_max
 - policy_image_noise_max
 - policy_image_occupied_max
 - policy_image_outfocused_max
 - policy_qualityControl
- **Result**: Esta clase almacena las instancias de qualityControl que han sido activadas en el proceso de razonamiento. Está compuesto por las siguientes propiedades:
 - result_qualityControl

2.6. Razonamiento y manejo de conflictos

Una vez que los puntos importantes y las características de la imagen del Iris han sido obtenidas y se han generado nuevas instancias de ImagesFeatures e IrisImage, la fase del razonamiento comienza. En esta fase, el Control de Calidad será activado acorde con las políticas definidas y las características de la imagen y el Iris obtenidas previamente. Como resultado de esto, habrá una instancia de Result, rellena con todas las políticas de control activadas, [9].

Es posible que de acuerdo con las características de la imagen de Iris y las políticas definidas, algunas contradicciones aparezcan en el Control de Calidad, haciendo un conflicto de resultados. Las reglas siguientes han sido definidas para manejar estos conflictos:

- Si hay al menos un Reject, entonces se rechaza.
- Si existe más de un posible Treatment, entonces la imagen es tratada con cada algoritmo propuesto por el razonamiento.
- Si se genera Accept y Treatment, entonces la imagen es tratada.

La fase de razonamiento ha sido desarrollada con el plug-in Jess 7, [10], para razonar sobre ontologías con reglas SWRL en Protegé.

3. Resultados

La intención de esta sección es doble: Por un lado, proporciona un estudio temporal de cada uno de las etapas en la segmentación de Iris aquí presentada, y por otro lado, presenta algunos ejemplos de la ejecución del procedimiento completo.

3.1. Estudio temporal

Primeramente, el entorno donde se ha llevado a cabo el experimento es presentado, pues los tiempos de computación dependen enormemente de ello. Un ordenador Core 2 Duo HP PC 1.86 GHz fue usado, con 2GB de memoria RAM, y con R14 Matlab 2007 para ejecutar los algoritmos programados.

Considerando esto, el tiempo promedio de cada proceso es mostrado para medir la ejecución temporal. La detección de Pupila conlleva 0,28 segundos, la primera aproximación 2,24 segundos y la segunda aproximación 1,21 segundos. Por lo tanto, únicamente casi 2,25 segundos serían necesarios para detectar el Iris en promedio (con la primera aproximación) en términos de requisitos de un buen Control de Calidad, tal y como se vio en la Sección 2.4.

Además, con casi un segundo más, el proceso completo sería llevado a cabo. Además, el tiempo para procesar la ontología es despreciable, con lo cual no se ha considerado en esta sección.

Esto es un resultado importante si es comparado con otros algoritmos y los entornos de experimentación, considerando el hecho de que la implementación fue hecha en Matlab y no ha sido depurada de forma detallada, ni optimizada. Una implementación en C o C++ será un trabajo futuro para así mejorar más aún los tiempos de computación.

3.2. Ejemplo Visual

En esta sección se presenta cómo actúan los Filtros basados en distribuciones anteriores y los resultados de las aproximaciones, primera y segunda.

Fig. 3 muestra los diferentes efectos de los filtros previos. El lector puede fácilmente observar como la imagen es difuminada en aquellas regiones más alejadas de la Pupila. Los efectos dependen no solamente de los parámetros mostrados en la Tabla 1, sino también de la intensidad de la imagen. En otras palabras, la respuesta del filtro para ojos oscuros es diferente que en ojos claros.

Además, Fig. 4 muestra dos ejemplos de la primera aproximación en el proceso de detección de Iris. Tal y como se dijo en las secciones anteriores, la información proporcionada por esta primera aproximación es más que suficiente para localizar el Iris y decidir sobre la calidad de la imagen adquirida.

También puede verse el resultado de la detección de Pupila en la misma imagen. En la segunda imagen, el Iris se encuentra parcialmente cubierto por las pestañas, siendo ésta una situación de gran dificultad para detectar el Iris.

En la imagen superior se aprecian los resultados de la detección de Pupila e Iris, tal y como se ha propuesto en este documento.

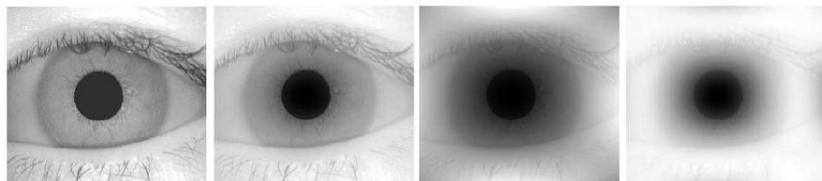


Figura 3. Efectos de los Filtros basados en distribuciones en imágenes de iris: Imagen Original, Gaussian, Coseno II, Coseno I.

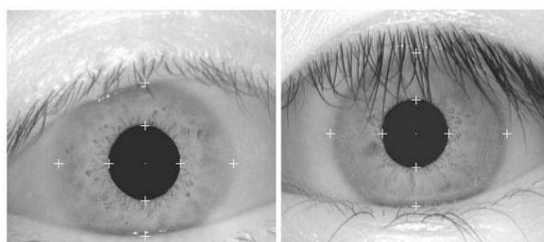


Figura 4. Dos ejemplos de la detección de Pupila y la primera aproximación en la detección de Iris.

Fig. 5 muestra el resultado de la segunda aproximación para dos imágenes diferentes. Los valores de los parámetros han sido obtenidos a partir de una Red Neuronal que fue entrenada en base a los valores extraídos de las imágenes. La frontera interna del iris coincide con el resultado de la frontera externa de la Pupila. A pesar de ser posible aislar completamente la pupila, en este documento se propone extraer únicamente cinco puntos que delimiten la pupila en vez de toda la frontera.

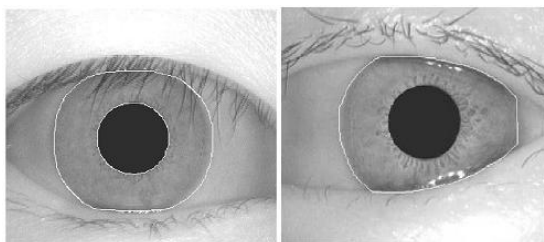


Figura 5. Resultados de la segunda aproximación en detección de Iris.

4. Conclusiones y Trabajo Futuro

4.1. Conclusiones

Esta detección de Iris tiene dos objetivos: primeramente, el algoritmo es capaz de aislar el Iris, mediante la obtención de ciertos puntos importantes (cuatro para el Iris, cinco para la Pupila). Esta propuesta ahorra coste computacional y tiempo en el proceso, a parte de espacio en memoria a la hora de almacenar la segmentación de la imagen.

Además, el proceso de detección de Iris contiene una segunda etapa, donde el Iris puede ser aislado completamente. Esto permite que este algoritmo pueda ser integrado en sistemas actuales basados en detección de Iris. Este segundo paso, no sólo realiza la segmentación de forma rápida sino que además es bastante preciso a la hora de aislar el Iris.

Por otro lado, el Control de Calidad basado en ontologías y razonamiento basado en reglas, es capaz de proporcionar al proceso de segmentación nuevas herramientas para decidir si una posterior extracción de características es factible con la imagen que se ha capturado o si, por el contrario, la imagen precisa de un tratamiento numérico o si el sistema requiere una nueva captura de imagen.

Finalmente, el proceso completo mejora a los algoritmos actuales de segmentación en diferentes aspectos, haciendo de este procedimiento un algoritmo fácilmente extensible a otras técnicas biométricas.

4.2. Trabajo Futuro

Como trabajo futuro, el algoritmo será utilizado con otras bases de datos para probar su eficiencia. Sin embargo, teniendo en cuenta que el algoritmo ha sido implementado de forma independiente de la base de datos, los resultados prometen ser interesantes.

Por otro lado, un algoritmo evolutivo, [11], podría ser utilizado para evitar el problema de ‘overfitting’ en la red neuronal, y además, se estudiará la posibilidad de utilizar Lógica Difusa, [12], junto con la red neuronal para hacer este algoritmo aún más preciso.

5. Agradecimientos

Los autores quieren agradecer al Proyecto CENIT Segur@: Seguridad y Confianza en la Sociedad de la Información, financiado por el Ministerio de Industria, Turismo y Comercio.

Referencias

1. J. Daugman, *High Confidence Visual Recognition of Persons by a Test of Statistical Independence*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 15, No 11, Nov. 1993.

2. J. Daugman, 'How Iris Recognition Works', IEEE Transactions on Circuits and Systems For Video Technology, Vol. 14, n 1, January 2004.
3. A. de Santos Sierra, C. Sánchez Ávila, E. Marchiori, *Iris Recognition: Segmentation enhancement by using Morphological Operators*, Master Final Thesis, June, 2007. Amsterdam.
4. R. C. González, R. E. Woods, S. L. Eddins, *Digital Image Processing*, 2nd Edn, Prentice All, 2004.
5. J. R. Hiler, V. J. Martínez, *Redes Neuronales Artificiales: Fundamentos, modelos y aplicaciones*, Ra-ma, 1995
6. CASIA Iris Image Database. <http://www.sinobiometrics.com>
7. I. Horrocks, P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosz, M. Dean, *SWRL: A Semantic Web Rule Language Combining OWL and RuleML*, W3C Member Submission 21 May 2004. <http://www.w3.org/Submission/SWRL/>
8. Ontology editor and knowledge-base framework Protegé. <http://protege.stanford.edu/>
9. J. Guerra Casanova, *Diseño y desarrollo de un sistema de reacción ante ataques en una red basada en ontologías*, Master Final Thesis, July, 2008. Madrid.
10. Rule Engine plugin Jess. <http://herzberg.ca.sandia.gov/>
11. A. E. Eiben and J. E. Smith, *Introduction to Evolutionary Computing*, Berlin, Germany: Springer, 2003.
12. P. Stavroulakis, *Neuro-Fuzzy and Fuzzy-Neural Applications in Telecommunications*, Springer, 2004, Berlin.

Una propuesta de arquitectura biométrica de control de acceso basada en ontologías

Javier Guerra Casanova¹, Alberto de Santos Sierra¹, Carmen Sánchez Ávila¹ y Vicente Jara Vera²

¹ Grupo de Biometría y Tratamiento Numérico de la Información.
Centro de Domótica Integral (CeDInt)
<jguerra@cedint.upm.es>
<alberto@cedint.upm.es>
<csa@mat.upm.es>

² Dpto. de Matemática Aplicada a las Tecnologías de la Información.
ETSI de Telecomunicación. Universidad Politécnica de Madrid.
Ciudad Universitaria s/n, 28040 Madrid
<vjara@mat.upm.es>

Resumen Hoy en día existen multitud de investigaciones sobre técnicas biométricas para autenticar a personas y cuantificar su estado físico y psíquico (cansancio, ansiedad, estrés...) En este artículo se propone integrar estas técnicas en una arquitectura de control de acceso basada en el modelo OrBAC ("Organization-based Access Control"). Con esta propuesta, el administrador de seguridad de una organización va a ser capaz de definir las políticas de seguridad de alto nivel que van a dar acceso a un usuario al sistema, así como las políticas de autenticación y comprobación de características que un usuario ha de verificar para que el sistema pueda confirmar que es quién dice ser y que está capacitado para realizar la acción para la que solicita acceso. Para la definición de los elementos de una organización a bajo y alto nivel, para la especificación de las técnicas biométricas disponibles en una organización y para la gestión de las políticas de seguridad, autenticación y comprobaciones biométricas se ha desarrollado una ontología, que al razonar sobre ella, traducirá las políticas de alto nivel a órdenes concretas a realizar por el usuario que trata de acceder al sistema. Esta arquitectura propuesta puede ser utilizada por cualquier organización que quiera imponer unas políticas de seguridad de acceso muy alto, por tanto, puede ser de gran utilidad para entornos militares, hospitales, centrales nucleares o aeropuertos.

Palabras clave: Biometría, Arquitectura de Control de Acceso, OrBAC, Razonamiento Ontológico, Autenticación, Comprobación Biométrica.

1. Introducción

Este artículo trata de definir una arquitectura completa de control de acceso según la identidad y las capacidades de las personas para realizar una acción en concreto.

Esta arquitectura puede ser muy útil para ambientes de alta seguridad, donde no solo hay que asegurar que la persona que accede sea realmente ella, sino que también es importante que para realizar una acción la persona esté en plenas facultades físicas y psicológicas.

Hoy en día existen muchas líneas de investigación para autenticar a una persona mediante sus características biométricas, [1], [2], así como para la medición de ciertos parámetros que indican el estado físico y psicológico de una persona antes de realizar una acción, [3], [4]. Esta arquitectura va a tratar de integrar todas estas técnicas, para que puedan utilizarse unas u otras según se requieran.

Gracias a este sistema, pueden aplicarse autenticaciones multimodales o comprobaciones de diversas características físicas o psicológicas de las personas para la realización de acciones de alta seguridad, [5], [6], [7].

Este sistema permite elevar el nivel de seguridad de acceso a un sistema no solo controlando el acceso, sino comprobando biométricamente la identidad de la persona que accede y la capacidad física y psíquica que tiene para realizar ciertas acciones en el sistema.

Esta arquitectura puede ser de gran utilidad en entornos de alta seguridad como aeropuertos, hospitales, bancos y servicios secretos o militares, donde se requiere un control de acceso muy rígido a determinados lugares y estar en plenas facultades a la hora de realizar ciertas operaciones, como manejar un avión, operar en el quirófano, realizar una gran transferencia bancaria o acceder a documentos secretos.

El artículo se compondrá de las siguientes secciones: En primer lugar se explicará la arquitectura general propuesta, introduciendo cada una de las fases que va a tener la misma. A continuación se comentará la tecnología de acceso OrBAC, [8], que se utilizará como base del modelo que se ha definido. En las secciones siguientes se expondrán cada una de las fases propuestas en la arquitectura, detallando la ontología necesaria para controlar el razonamiento en cada una de ellas. Por último el artículo concluirá con unas conclusiones de las ventajas del modelo propuesto.

2. Arquitectura propuesta

La arquitectura que se propone para el control de acceso se basa en tres fases:

1. Identificación : El usuario se identificará en el sistema e indicará qué quiere hacer. El sistema comprobará que el usuario existe en su base de datos y comprobará que tiene el permiso necesario para realizar la acción que solicita.
2. Autenticación: Si el usuario tiene el permiso necesario para ejecutar la acción que solicita, el sistema le exigirá que autentique su identidad mediante

ciertas técnicas biométricas que la organización ha de disponer. Según el perfil del usuario, la acción que quiera realizar y las políticas de seguridad correspondientes definidas por el administrador del sistema el usuario deberá autenticarse en diferentes aparatos de manera multimodal. Para ejecutar acciones que se quieran dotar de mayor de seguridad será necesario autenticarse a través de más técnicas biométricas.

3. Comprobación de capacidades: Una vez autenticado, el sistema podrá exigir realizar las comprobaciones biométricas necesarias que aseguren que el usuario está en condiciones de realizar la acción que pretende. Según las políticas de decisión del sistema, se decidirá si el usuario tiene acceso o es rechazado por el mismo.

Según esta arquitectura, el rechazo de un usuario puede realizarse tras cualquier etapa, en todos estos supuestos:

- Si no hay una identificación almacenada en el sistema del usuario que trata de acceder al mismo.
- Si las políticas de seguridad del sistema no permiten que el usuario realice la acción que solicita.
- Si el sistema reconoce mediante técnicas biométricas que el usuario que trata de acceder no es quien ha dicho que es.
- Si el usuario no se encuentra en condiciones de realizar el acceso al sistema.

Para el control de los accesos y rechazos al sistema se proponen unas políticas de seguridad en dos niveles basadas en la arquitectura OrBAC, donde el administrador de seguridad podrá definir unas políticas de seguridad de alto nivel, que derivarán en políticas de seguridad de nivel concreto, [9].

Para controlar cada una de las fases y las políticas de seguridad del sistema se han definido varias ontologías, que se explicarán en la sección propia de cada fase del modelo.

3. Control de acceso basado en la filosofía OrBAC

El control de acceso a una organización se basa en la arquitectura OrBAC, que define dos niveles de comportamiento.

En el nivel concreto, un sujeto realiza una acción sobre un objeto. Para un nivel superior, se definen las entidades Rol; como un conjunto de sujetos con las mismas características de seguridad, Actividad; como una agrupación de acciones del mismo nivel de seguridad y Vista; como un grupo de objetos protegidos por las mismas condiciones de seguridad.

El administrador de seguridad en OrBAC define unas políticas de seguridad de alto nivel en el que va a prohibir/permitir/obligar a todos los sujetos que pertenecen a un rol realizar las acciones agrupadas en una actividad sobre todos los objetos incluidos en una vista.

Además, OrBAC define el concepto de Contexto, [10], para definir distintas condiciones que pueden alterar las políticas de seguridad de una organización,

por ejemplo, condiciones temporales (horario nocturno, navidad), condiciones de localización (oficinas de Montevideo) u otras. Las políticas de seguridad de alto nivel, incluyen los contextos de aplicación, y solo se activarán si en el momento de intentar acceder, la organización se encuentra en ese contexto.

Según la arquitectura OrBAC, las políticas de alto nivel definidas, se traducen automáticamente en políticas de seguridad de nivel concreto que son las que indican si un usuario en particular tiene permiso para realizar la acción que solicita sobre el objeto correspondiente.

A partir de la arquitectura OrBAC, este trabajo propone incluir las comprobaciones biométricas de autenticidad y de disposición a realizar la actividad dentro del modelo mediante ontologías, tal y como se explicará en las secciones siguientes.

4. Fase de identificación

Cuando un usuario trata de acceder al sistema, lo primero que ha de hacer es identificarse, mediante un código, una tarjeta o similar. En este primer paso el sistema comprobará que el sujeto pertenece al sistema y que además, el usuario tiene permiso para realizar la acción. Para ello se basará en la definición de una organización según el modelo OrBAC junto a las políticas de acceso definidas por el administrador del sistema que van a decidir si el Sujeto que llega al Sistema tiene permiso para realizar la Acción que solicita en el Contexto en el que se encuentra. Puede observarse como funciona el razonamiento en el esquema de la figura 1:

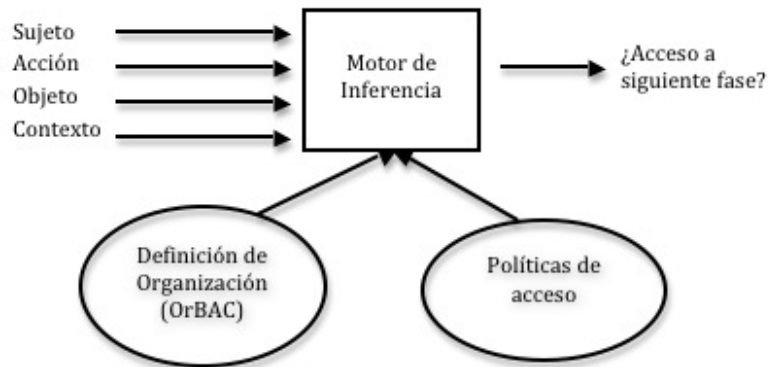


Figura 1. Razonamiento en la fase de identificación

Para controlar esta fase, se ha definido una ontología que implementa todas las clases necesarias. Se agrupan en dos grandes grupos:

- Para representar los elementos de una organización: Se han definido como clases de la ontología todos los elementos posibles de una arquitectura OrBAC.
 - Sujeto
 - Rol
 - Acción
 - Actividad
 - Objeto
 - Vista
 - Contexto
- Para definir las políticas de acceso a la organización:
 - PoliticaAltoNivel
 - Permiso

El administrador del sistema define instancias de la clase PoliticaAltoNivel que indican qué Roles tienen permiso para realizar qué Actividades sobre qué Vistas.

Al llegar un usuario al sistema e identificarse, el motor de inferencia razona sobre la ontología para decidir si el sujeto que ha llegado tiene permiso para realizar la acción solicitada sobre el objeto indicado, en el contexto que se encuentra la aplicación. Para ello, es necesario que el propio razonador haga una conversión de políticas de seguridad de alto nivel a Permisos de bajo nivel, que indiquen qué Sujetos pueden ejecutar qué Acciones sobre qué Objetos.

En el caso de que el usuario tenga el permiso para realizar la acción, se pasa a la siguiente fase de la arquitectura propuesta: la fase de autenticación biométrica multimodal.

5. Autenticación biométrica multimodal

En esta fase, se propone que el sistema realice una autenticación biométrica del usuario. Una vez que se ha determinado que el usuario que dice ser tiene permiso para ejecutar la acción que solicita, puede ser necesario para ciertas operaciones realizar una autenticación biométrica para comprobar que la persona que accede es quien dice ser.

El administrador del sistema va a definir políticas de autenticación que exijan que un usuario pase con éxito una/varias/ninguna autenticaciones biométricas a través de varias técnicas en función del Rol al que pertenezca, el tipo de acción que quiera ejecutar sobre el tipo de objeto determinado y el contexto del sistema.

En esta fase, la inferencia sigue el esquema de la Figura 2:

Para la implementación de esta fase se han añadido en la ontología de la sección anterior varias clases agrupadas de la siguiente manera:

- Clases para definir los distintos tipos de autenticación biométrica. Para ello se propone adaptar a ontologías la propuesta de Wayman de sistematización de métodos biométricos, [11], incluyendo las siguientes clases:

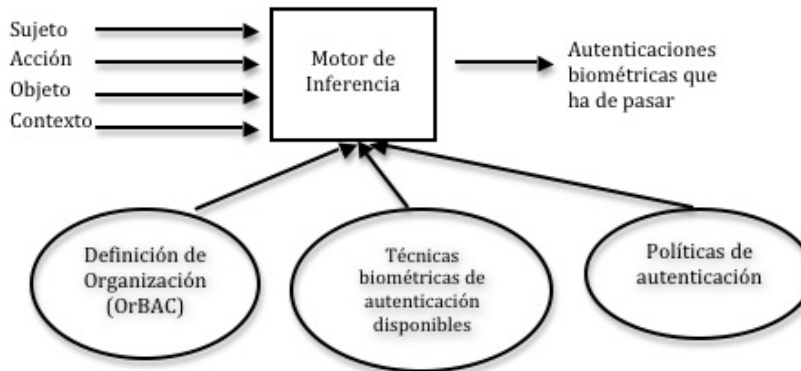


Figura 2. Razonamiento en la fase de autenticación

- Pattern: La característica biométrica que se va a utilizar para autenticar al usuario (Cara, [12], Huella, [13], Iris, [14], Geometría de la mano, [15] ...)
- Structure: Lo que se va a medir de la característica biométrica. Por ejemplo, en firma: velocidad y aceleración, presión..
- Extraction: Método para extraer los patrones biométricos del usuario: Análisis matemáticos morfológicos, análisis de wavelets, Vectores propios..
- QualityControl: Métodos para comprobar que las muestras obtenidas del usuario son de calidad o no: Modelos de Markov, Vectores propios, procesamientos de imágenes.
- Recognition: Métodos de comparación de patrones: Correlaciones, Modelos de markov, Redes neuronales. . .

Además, habría que incluir la clase Autenticacion, que va a incluir las características de cada una de las técnicas biométricas que va a soportar la organización. Estarán definidas las propiedades autenticacion_pattern, autenticacion_structure, autenticacion_extraction, autenticacion_qualityControl y autenticacion_recognition que relacionan las instancias Autenticacion con cada una de las clases correspondientes. También, es imprescindible añadir una propiedad authentication_id que relacione la técnica biométrica con el dispositivo físico que posee la organización para realizar la comprobación biométrica que se ha definido.

- Clases para definir las políticas de autenticación de la organización: El administrador de seguridad de la organización va a poder ofrecer distintos niveles de seguridad, decidiendo los tipos de autenticaciones biométricas que ha de pasar el usuario según el perfil que tenga, el tipo de acción que desee ejecutar, la clase del objeto sobre el que va a realizar la acción y el contexto en que se encuentre el sistema.

- **PolíticaAutenticacion:** Cada instancia de esta clase será un política de autenticación de alto nivel que incluirá la autenticación biométrica que han de pasar todos los sujetos de un Rol para poder ejecutar las acciones de una Actividad sobre los objetos de una Vista en cierto Contexto.
- **SolicitudAutenticacion:** Esta clase es la equivalente a la clase Permiso para esta fase. En ella se incluyen las solicitudes de autenticación concretas que ha de pasar un Sujeto para poder realizar una Acción sobre un objeto. Las instancias de esta clase serán inferidas por el sistema.

El razonamiento en esta fase es similar a la anterior, ya que el razonador ha de traducir las políticas de autenticación representadas en alto nivel (Rol, Actividad, Vista) a solicitudes de autenticación de nivel concreto (Sujeto, Acción, Objeto) según las distintas entradas del sistema, la descripción de la organización y las políticas de alto nivel establecidas.

Tras esta fase, al usuario se le exigirá pasar las autenticaciones biométricas para demostrar su identidad, según los resultados de la inferencia. El sistema ha de tener un módulo que sea capaz de traducir las autenticaciones exigidas a ordenes concretas sobre equipos concretos donde ha de pasar dichas autenticaciones (lectores de huella, cámaras...). En caso de que el usuario no pase alguna de las autenticaciones en las distintas técnicas biométricas, el sistema le rechazará.

6. Comprobación biométrica de capacidades físicas y psíquicas

Llegados a este punto, el sistema ha identificado al usuario, sabe que tiene permiso para realizar la acción que solicita y ha verificado que el usuario es quien dice ser. Esta última fase va a decidir qué pruebas biométricas ha de pasar el usuario autenticado para comprobar que tiene las capacidades físicas y psíquicas necesarias para realizar la acción requerida. (grado de estrés, cansancio, miedo, ansiedad...) En la actualidad existen muchas líneas de investigación trabajando en la detección de estas características, [3], [4], y más que aparecerán en años sucesivos; con esta arquitectura, se pueden integrar todas ellas en un sistema real.

En esta fase, la inferencia tiene un esquema similar que en las fases anteriores, tal y como se observa en el esquema de la Figura 3:

Para este razonamiento, es necesario añadir en la ontología que se está describiendo las clases necesarias para la definición de las comprobaciones biométricas que tiene instaladas la organización y para definir las políticas de seguridad que van a seguir:

- Clases para la representación de comprobaciones biométricas. De igual manera que en la sección anterior, se propone utilizar la propuesta de Wayman para la definición de métodos biométricos, en este caso, aplicados a técnicas de comprobaciones físicas o psíquicas.

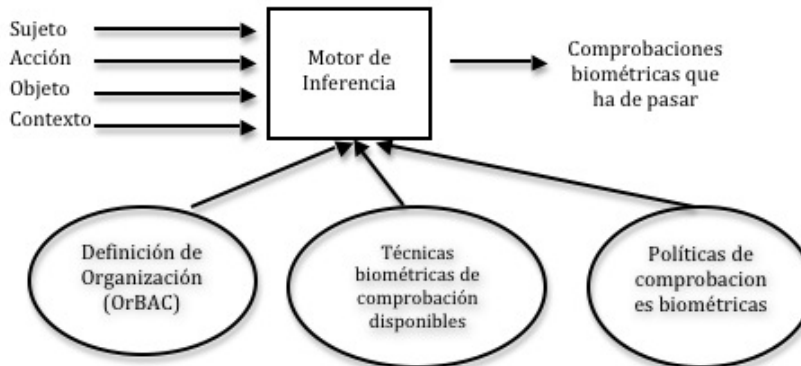


Figura 3. Razonamiento en la fase de comprobación de capacidades

- **ComprobaciónBiométrica:** Esta clase contiene las propiedades `comprobacion_pattern`, `comprobacion_structure`, `comprobacion_extraction`, `comprobacion_qualityControl` y `comprobacion_recognition` para relacionar cada una de las propiedades de la propuesta de Wayman de definición de métodos biométricos con las técnicas biométricas implementadas en la organización para la comprobación de estados físicos y psíquicos. En esta clase, es necesario incluir también la propiedad `comprobación_id` para relacionar la técnica biométrica con el dispositivo físico concreto de la organización que va a llevar a cabo la medida biométrica y la comprobación de la característica biométrica correspondiente.
- **Clases para definir las políticas de la organización de comprobaciones de aptitudes de acceso:** El administrador de seguridad de la organización va a poder definir las pruebas que ha de pasar cada usuario para estar seguros de la capacidad del mismo para poder realizar una acción.
 - **PoliticaComprobacion:** Las instancias de esta clase son políticas de alto nivel definidas por el administrador de seguridad que especifican las comprobaciones biométricas físicas y psíquicas que han de realizar todos los sujetos de un Rol para poder realizar las acciones incluidas en una Actividad sobre todos los objetos que pertenecen a una vista en un Contexto.
 - **SolicitudComprobacion:** Esta clase se corresponde a la clase `Permiso` en esta fase. Se incluyen las solicitudes de comprobaciones de características concretas que ha de pasar el sujeto que accede al sistema. Estas instancias serán inferidas por el motor de razonamiento según las Políticas de comprobación definidas. En esta clase se incluye la propiedad `solicitud-Comprobacion_umbral`, que marca el porcentaje máximo que puede tener una persona de una características física o psíquica para tener acceso al sistema (Por ejemplo, un 75 % de estrés).

En esta fase el razonamiento consiste en traducir las políticas de comprobación de alto nivel (Rol, Actividad, Vista) a las solicitudes de comprobación de bajo nivel del Sujeto, Acción y Objeto entradas del sistema.

Como resultado de esta fase, al usuario se le exigirá realizar unas comprobaciones específicas para demostrar que se encuentra en el estado físico y psíquico imprescindible para realizar la acción que quiere sobre el objeto deseado. Nuevamente, el sistema ha de tener un módulo para traducir las solicitudes de comprobaciones exigidas a órdenes concretas sobre equipos concretos donde se van a llevar a cabo las comprobaciones biométricas. Si el usuario sobrepasa el valor umbral de una de las características biométricas, el sistema no le permitirá el acceso.

7. Conclusiones

En este artículo se ha presentado una propuesta de arquitectura para el control de acceso integrando un esquema típico basado en OrBAC con las posibles tecnologías biométricas de autenticación y comprobación de estados físicos y psíquicos para poder tener acceso a la ejecución de una acción.

Gracias a este modelo, una organización puede aumentar sus niveles de seguridad manteniendo sus políticas de seguridad y ampliándolas a políticas de autenticación y comprobación de estado.

Al ser una arquitectura general, se puede integrar fácilmente cualquier tipo de tecnología biométrica que exista o vaya apareciendo; siendo esta arquitectura fácilmente escalable y actualizable según se vayan instalando nuevos dispositivos para realizar las autenticaciones y comprobaciones biométricas necesarios.

Este modelo permite autenticar o comprobar las características de un mismo usuario mediante distintas técnicas, dando un enfoque de multimodalidad biométrica al sistema.

Para controlar las distintas fases de la arquitectura se ha utilizado una ontología, para que mediante un motor de razonamiento se infieran automáticamente las solicitudes concretas de autenticación y comprobación biométrica que un usuario ha de pasar en base a las políticas de seguridad definidas en la organización.

Esta arquitectura puede tener una gran aplicación en entornos de alta seguridad, donde para realizar una acción o acceder a un lugar es necesario tener permiso, tener la seguridad de que el que accede es quien dice ser y que esté en plenas facultades físicas y psíquicas. Puede aplicarse en entornos militares, de control de acceso en aeropuertos, hospitales, centrales nucleares..

Referencias

1. A. K. Jain, A. Ross and S. Pankanti, "Biometrics: A Tool for Information Security", IEEE Transactions on Information Forensics and Security Vol. 1, No. 2, pp. 125-143, June 2006.

2. S. Prabhakar, S. Pankanti, and A. K. Jain, "Biometric Recognition: Security and Privacy Concerns", *IEEE Security & Privacy Magazine*, Vol. 1, No. 2, pp. 33-42, March-April 2003.
3. Barreto, A., Zhai, J., Adjouadi, M., "Non-intrusive Physiological Monitoring for Automated Stress Detection in Human-Computer Interaction", *LNCS 4796*, pp 29-38, Springer-Verlag Berlin Heidelberg 2007
4. Healey, J. A., Picard, R. W., "Detecting Stress During Real-World Driving Tasks Using Physiological Sensors", *IEEE Transaction on Intelligent Transportation Systems*, Vol. 6, No. 2, Junio 2005.
5. A. Ross and A. K. Jain, "Multimodal Biometrics: An Overview", *Proc. of 12th European Signal Processing Conference (EUSIPCO)*, (Vienna, Austria), pp. 1221-1224, September 2004.
6. A. K. Jain and A. Ross, "Multibiometric Systems", *Communications of the ACM, Special Issue on Multimodal Interfaces*, Vol. 47, No. 1, pp. 34-40, January 2004.
7. A. Ross, A. K. Jain, and Jian Zhong Qian, "Information Fusion in Biometrics", *Proc. 3rd International Conference on Audio- and Video-Based Person Authentication (AVBPA)*, pp. 354-359, Sweden, June 6-8, 2001.
8. A. Abou El Kalam, R. El Baida, P. Balbiani, S. Benferhat, F. Cuppens, Y. Deswarte, A. Miège, C. Saurel and G. Trouessin. "Organization Based Access Control". *IEEE 4th International Workshop on Policies for Distributed Systems and Networks (Policy 2003)*, Lake Como, Italy, June 4-6, 2003.
9. F. Cuppens, N. Cuppens-Boulahia, T. Sans and A. Miège. "A formal approach to specify and deploy a network security policy". *Second Workshop on Formal Aspects in Security and Trust (FAST)*. Toulouse, France. August, 2004.
10. F. Cuppens and A. Miège. "Modelling contexts in the Or-BAC model". *19th Annual Computer Security Applications Conference*, Las Vegas, December 2003.
11. J. Wayman, A. K. Jain, D. Maltoni, and D. Maio, "Biometric Systems: Technology, Design and Performance Evaluation", Springer Verlag, 2005.
12. R.-L. Hsu, "Face Detection and Modeling for Recognition", Ph.D. Thesis, 2002.
13. S. C. Dass and A. K. Jain, "Fingerprint-Based Recognition", *Technometrics*, vol. 49, no. 3, pp. 262-276, 2007.
14. Wildes, R.P., "Iris recognition: an emerging biometric technology," *Proceedings of the IEEE*, vol.85, no.9, pp.1348-1363, Sep 1997
15. Arun Ross. "A Prototype Hand Geometry-based Verification System", M.S. Project Report, 1999.

Índice de autores

Ariza, Andrea	1	Hernández, Emilio	298
Baptista, Patricio	148	Herrera-Joancomartí, Jordi	202
Bastos, Fernando	148	Huguet-Rotger, Llorenç	340
Bavera, Francisco	88	Igualada Menor, Angel	213
Belleboni, Emilia Pérez	16	Isern-Deyà, Andreu Pere	340
Benitez, Carlos	31	Jaimes, Patricia	225
Bertacchini, Maximiliano	46	Jara Vera, Vicente	497, 509
Bertol, Viviane	61	L. Hernandez-Ardieta, Jorge	240
Blanco, Alejandro	73	laRed Martinez, David	163
Bonelli, Eduardo	88	Leite, Thiago	148
Botero, Armando	103	Lopez, Javier	409
Camero, Ivan	103	Luna, Carlos	255, 283
Campo, Juan Diego	73	Martinez-Cagnazzo, Carlos	270
Cano, Jeimy	1, 103	Mazeikis, Gustavo	283
Carracedo Gallardo, Justo	16, 310	Megías, David	202
Castellaro, Marta	118	Mellado, Daniel	383
Climent, Joan-Josep	133	Minguet, Jesús María	398
Corti, María Eugenia	325	Monteiro, Edmundo	148
Cruz, Tiago	148	Morales, Mireya	298
Custodio, Ricardo	61	Muñoz Muñoz, Alfonso	310
de Fuentes, José María	482	Mujica, Sergio	163
de Santos Sierra, Alberto	497, 509	Mut-Puigserver, Macia	340
de Sousa Júnior, Rafael Timóteo	61	Pallas, Gustavo	325
Escanellas, Lucía	73	Payeras-Capellà, M. Magdalena	340
Feck, Carlos	118	Pazo Robles, María Eugenia	355
Fernández, Hugo	173	Piattini, Mario	383, 437, 452
Fernández-Medina, Eduardo	383, 409, 437, 452	Pintado, Carlos	73
Fernandez, Eduardo B.	163	Ramírez López, César José	368
Ferrer-Gomila, Josep L.	340	Ramos Álvarez, Benjamin	240, 482
Fierens, Pablo	31, 46	Ramos, Juan Carlos	118
Fuster Sabater, Amparo	355	Requena, Verónica	133
Gómez, Ana	423	Ribagorda Garnacho, Arturo	240, 482
García, Francisco J.	133	Roberto, Gomez	225
Gaspoz, Ivana	118	Rodríguez, Marcelo	73
González-Tablas, Ana Isabel	240, 482	Rodríguez, Jesus	383
Grosclaude, Eduardo	173	Romaniz, Susana	118
Guerra Casanova, Javier	497, 509	Roque, Juan José	398
Hecht, Juan Pedro	188	Rosa, Cristian	255
Hermosillo, Gabriel	225	Rosado, David G.	409
Hernández Audelo, Leobardo	368	Ruíz, Juan	1
		Sánchez Ávila, Carmen	497, 509
		Sánchez Crespo, Luis Enrique	437, 452

Sánchez, Sergio	423
Simões, Paulo	148
Sznek, Jorge	173
VanHilst, Michael	163
Vilão, Rui	148
Villafranca Alberca, Daniel	437, 452
Zipitría, Felipe	467