Actas de la IX Reunión Española sobre Criptología y Seguridad de la Información

Casa de Convalescència, Hospital de la Santa Creu i Sant Pau

7, 8 y 9 de septiembre del 2006, Barcelona

Departament d'Enginyeria de la Informació i les Comunicacions,
Universitat Autònoma de Barcelona
Estudis d'Informàtica, Multimèdia i Telecomunicacions,
Universitat Oberta de Catalunya

Editores
Joan Borrell Viader
Jordi Herrera Joancomartí

Editores: Joan Borrell Viader y Jordi Herrera Joancomartí.

Primera edición: julio 2006.
ISBN: 84-9788-502-3

Prólogo

Esta publicación recoge las actas de la Reunión Española sobre Criptología y Seguridad de la Información (RECSI), celebrada los días 7, 8 y 9 de septiembre del 2006 en Barcelona.

La RECSI llega en el año 2006 a su novena edición, organizada de forma conjunta por el Departamento de Ingeniería de la Información y de las Comunicaciones de la Universidad Autònoma de Barcelona y el Departamento de Informática y Multimedia de la Universidad Oberta de Catalunya. Esta IX RECSI quiere seguir siendo el lugar de encuentro y el foro en el que los criptólogos y, en general, todos aquellos que trabajan en el campo de la Seguridad de la Información expongan sus hallazgos y debatan sus ideas. Se trata de un congreso bienal que se celebra en universidades y centros de investigación de España. Las ediciones anteriores se llevaron a cabo en Palma de Mallorca (U. Illes Balears), Madrid (CSIC), Barcelona (U. Politècnica de Catalunya), Valladolid (U. de Valladolid), Torremolinos (U. de Málaga), Santa Cruz de Tenerife (U. de La Laguna), Oviedo (U. de Oviedo) y Leganés (U. Carlos III).

La expansión de Internet, el incremento exponencial del volumen de datos automatizados que se maneja, la creciente inquietud por la protección de la intimidad y, en general, la entrada en la era de la información hace que la seguridad de ésta se configure como un campo de singular importancia, y por ello concentre un especial interés por parte de las empresas, las administraciones, los profesionales y más ampliamente, la sociedad entera. Por otro lado, la Criptología, en su doble vertiente de diseño de algoritmos criptográficos y de análisis de sus posibles debilidades, se ha convertido en la disciplina vertebral de la seguridad, habiendo abandonado los círculos impenetrables en los que se desplegaba históricamente, para ser tratada en universidades, centros de investigación, empresas y organismos de todo tipo interesados en proteger las informaciones que manejan.

Conscientes de lo anterior, en la IX RECSI se tratan y profundizan los aspectos de estas materias que más despiertan la atención en estos días, así como otros, aún en investigación, pero que están llamados a ser de capital importancia en los sistemas y mecanismos de seguridad en un inmediato futuro. A lo largo de las tres jornadas que conforman la Reunión se presentan 63 comunicaciones en 18 sesiones paralelas. Queremos agradecer desde estas líneas el trabajo realizado por el Comité Científico y los revisores en el proceso de revisión.

La IX RECSI, buscando mantener un elevado nivel académico y también un adecuado nivel de contacto de la comunidad investigadora con las empresas y la sociedad, incluye también:

- Tres conferencias magistrales a cargo de investigadores de reconocido prestigio en el ámbito de la Criptología y la Seguridad de la Información, el Dr. Moni Naor, del Weizmann Institute of Science (Israel), el Dr. Frédéric Cuppens de la Escuela Normal Superior de Telecomunicaciones de Bretaña

(Francia) y el Dr. Gene Tsudik de la Universidad de California en Irvine (USA).

- Dos presentaciones de empresas, Safelayer Secure Communications, compañía líder en el mercado de seguridad y confianza para las TIC, desarrollando tecnología de identificación electrónica, firma electrónica y protección de datos basada en Infraestructura de Clave Pública (PKI), y Scytl Secure Electronic Voting, compañía líder en el desarrollo de plataformas de votación electrónica seguras y confiables, aplicables desde procesos electorales clásicos a juntas generales de accionistas.
- La presentación de la Unidad Central de Informática Forense de la Policía de la Generalitat de Catalunya - Mossos d'Esquadra.

Manifestar también nuestro agradecimiento por la ayuda financiera y de difusión recibida de los distintos patrocinadores, cuya relación aparece en la página de agradecimientos de estas actas.

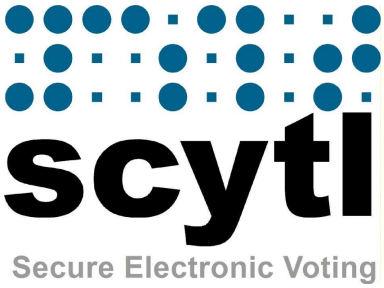
No quisieramos finalizar este prólogo sin recordar a nuestro amigo Andreu Riera Jorba, participante en varias Reuniones, tristemente fallecido en accidente de coche el 11 de marzo de 2006. Andreu, doctor por la UAB, era conocido tanto por su valiosa aportación en el campo de la criptografía aplicada al voto electrónico, como por su espíritu emprendedor que le llevó a fundar Scytl Secure Electronic Voting, empresa de la cual era Consejero Delegado.

Septiembre 2006

Joan Borrell Viader
Jordi Herrera Joancomartí

Agradecimientos

Los organizadores de la RECSI quieren agradecer a los patrocinadores de la Reunión su apoyo logístico y económico.



Organización

La IX RECSI ha sido organizada conjuntamente por el Departament d'Enginyeria de la Informació i les Comunicacions de la Universitat Autònoma de Barcelona y los Estudis d'Informàtica, Multimèdia i Telecomunicacions de la Universitat Oberta de Catalunya.

Comité ejecutivo

Joan Borrell Viader
Jordi Herrera Joancomartí
Josep Rifà Coma

Comité científico

Abascal Fuentes, Policarpo (U. de Oviedo)
Arranz Chacón, Maria Luisa (Alcatel)
Areitio Bertolín, Javier (U. de Deusto)
Borrell Viader, Joan (U.Autònoma de Barcelona)
Caballero Gil, Pino (U. de La Laguna)
Dávila Muro, Jorge (U. Politécnica de Madrid)
Domingo-Ferrer, Josep (U. Rovira i Virgili)
Fernández-Medina Patón, Eduardo (U. de Castilla La Mancha)
Ferrer Gomila, Josep Lluis (U. de les Illes Balears)
Fúster Sabater, Amparo (CSIC)
Gómez Skarmeta, Antonio (U. de Múrcia)
González Jiménez, Santos (U. de Oviedo)
Guía Martínez, Dolores de la (CSIC)
Gutiérrez Gutiérrez, Jaime (U. de Cantabria)
Herrera Joancomartí, Jordi (U. Oberta de Catalunya)
Huguet Rotger, Llorenç (U. de les Illes Balears)
López Muñoz, Javier (U. de Málaga)
Martín del Rey, Angel (U. de Salamanca)
Mañas Argemí, José Antonio (U. Politécnica de Madrid)
Miret Biosca, Josep Maria (U. de Lleida)
Padró Laimon, Carles (U. Politècnica de Catalunya)
Peinado Domínguez, Alberto (U. de Málaga)
Ramió Aguirre, Jorge (U. Politécnica de Madrid)
Ramos Álvarez, Benjamín (U. Carlos III de Madrid)
Ribagorda Garnacho, Arturo (U. Carlos III de Madrid)
Rifà Coma, Josep (U. Autónoma de Barcelona)
Robles Martínez, Sergi (U. Autónoma de Barcelona)

Salazar Riaño, Jose Luís (U. de Zaragoza)
Sempere Luna, José Maria (U. Politécnica de Valencia)
Soriano Ibáñez, Miquel (U. Politècnica de Catalunya)
Rifà Coma, Josep (U. Autònoma de Barcelona)
Tena Ayuso, Juan (U. de Valladolid)
Villar Santos, Jorge (U. Politècnica de Catalunya)

Comité Organizador

Joan Arnedo (Universitat Oberta de Catalunya)
Carles Garrigues (Universitat Autònoma de Barcelona)
David Megías (Universitat Oberta de Catalunya)
Alvaro Moratalla (Universitat Autònoma de Barcelona)
Guillermo Navarro (Universitat Autònoma de Barcelona)
Josep Prieto (Universitat Oberta de Catalunya)
Segi Robles (Universitat Autònoma de Barcelona)
Jordi Serra (Universitat Oberta de Catalunya)
Pere Urbón (Universitat Autònoma de Barcelona)

Revisores

Guillermo Azuara Guillén
Óscar Cánovas Reverte
Jordi Castellà Roca
Sergio Castillo Pérez
Vanesa Daza Fernández
Oscar Esparza Martín
Juan M. Estévez Tapiador
Joaquín García Alfaro
Félix J. García Clemente
Maria Isabel González Vasco
Julio César Hernández Castro
Gabriel López Millán
Consuelo Martínez López
Antoni Martínez Ballesté
Gregorio Martinez Perez
José Luis Muñoz-Tapia
Josep Pegueroles
Joan Josep Piles Contreras
Helena Rifà Pous
Francesc Sebé Feixas
Agusti Solanas Gómez

Índice general

Sesión C1

Sesión S1

Sesión C2

Sesión S2

Sesión C3

Sesión S3

Sesión C4

Sesión S4

Sesión C5

Sesión S5

Sesión S6

Sesión S7

Sesión C6

Sesión S8

Sesión S12

Patrones de Seguridad conforme a los Requisitos de Seguridad para Servicios Web

David G. Rosado[1], Carlos Gutiérrez[2], Eduardo Fernández-Medina[1], and Mario Piattini[1]

[1] Grupo ALARCOS
Departamento de Tecnologías y Sistemas de Información
Centro Mixto de Investigación y Desarrollo de Software UCLM-Soluziona
Universidad de Castilla-La Mancha
Paseo de la Universidad 4 - 13071, Ciudad Real, España.
{David.GRosado, Eduardo.Fdez-Medina, Mario.Piattini}@uclm.es
[2] STL. Calle Manuel Tovar 9, 28034 Madrid, España.
carlos.gutierrez@stl.es.

Resumen La seguridad, en sistemas basados en servicios, es un aspecto crítico desde que su infraestructura operativa está basada en Internet, el cual es un medio público e inherentemente inseguro. Actualmente, hay un destacado movimiento en la industria hacia una estandarización de los mecanismos de seguridad a ser usados en los sistemas basados en servicios Web. De hecho, en los últimos años, los más importantes consorcios de Internet, como IETF, W3C u OASIS, han producido un gran número de estándares de seguridad basados en servicios Web. En los entornos tecnológicos, los patrones dan a los arquitectos de sistemas de información un método para definir soluciones reusables a problemas de diseño. El propósito de usar patrones es crear un elemento de diseño reusable. Podemos obtener de una forma sistemática una arquitectura de seguridad que contiene un conjunto de patrones de diseño de seguridad a partir de los requisitos de seguridad encontrados. Así bien, usando mapeos adecuados, podemos expresar los requisitos de seguridad , los patrones arquitectónicos de seguridad y los patrones de diseño de seguridad en términos de estándares de seguridad basados en servicios Web, y sus mejores prácticas y tecnologías relacionadas.

Palabras clave: Requisitos de Seguridad, Patrones de Seguridad, Servicios Web, Arquitectura de Seguridad.

1. Introducción

Los servicios Web son una consecuencia natural de la evolución de la Web. Desde su comienzo como una forma de compartir y distribuir información a escala global, convirtiéndose en una gigante biblioteca de contenido distribuido, la Web ha estado extendiendo progresivamente sus límites haciendo posible más formas sofisticadas de interacción entre programas clientes y servidores: interacciones basadas en forma única, aplicaciones de comercio electrónico e interacciones *business-to-business* más complejas

[9]. IDC estima que 2.3 billones de dólares fueron gastados en 2004 en todo el mundo en software de servicios Web, más del doble que en años anteriores. IDC anticipa que el gasto continuará incrementándose sobre los próximos 5 años, alcanzando aproximadamente los 14.9 billones de dólares en 2009 [27].

Los beneficios de tener una unión débil, lenguaje neutral, y plataforma independiente para enlazar aplicaciones dentro de las organizaciones, a través de las empresas, y a través de Internet están llegando a ser más evidentes, así los Servicios Web son usados en programas pilotos y en producciones de amplia escala. Avanzando, nuestros clientes, analistas de industrias, y la prensa identifican un área clave que necesita ser dirigida con los servicios Web: la seguridad [26].

La mayoría de departamentos IT (*Information Technology*) están implementando esta tecnología con alta prioridad, haciéndolos operables, dejando de lado, al menos hasta etapas avanzadas, los problemas relacionados con la seguridad. En general, las compañías son todavía reticentes de incorporar esta tecnología, debido al poco entendimiento que ellos tienen de los riesgos involucrados con la seguridad, y la falsa creencia de que tendrán que hacer una reinversión costosa en sus infraestructuras de seguridad [24].

Para asegurar servicios Web, organismos tales como el *World Wide Web Consortium* (W3C), la *Organization for the Advancement of Structured Information Standards* (OASIS), la *Liberty Alliance* y otros están desarrollando estándares de seguridad basado en XML para solventar los problemas de seguridad [28]. Esta diversidad, también encontrada en el contexto de la seguridad de los Servicios Web [24], nos ha hecho considerar su aplicación, desde una perspectiva global, como un proceso duro y complejo de entender con una muy difícil curva de aprendizaje. Actualmente, todavía hay una falta de propuesta general que ofrezca un desarrollo metodológico para construir arquitecturas de seguridad para sistemas basados en servicios Web. En trabajos previos, el proceso PWSSec (*Process for Web Services Security*) [21] [22] ha sido desarrollado. Este proceso puede ser usado una vez que la arquitectura funcional del sistema ha sido construida o durante las etapas utilizadas para elaborar esta arquitectura. En ambos casos, el resultado será una arquitectura de seguridad formada por un conjunto de mecanismos de seguridad coordinados que usan los estándares para servicios Web que cumplen con los requisitos de seguridad del sistema.

Los patrones de diseño dicen a sus lectores cómo diseñar un sistema, dan una definición del problema y un conjunto de soluciones que actúan sobre el sistema. En los entornos de tecnologías de la información, los patrones dan información a los arquitectos de sistemas de un método para definir soluciones reusables para diseñar problemas sin tener que hablar sobre o escribir código de programa; son verdaderamente lenguajes de programación independientes.

El propósito de usar patrones es crear un elemento de diseño reusable. La combinación de patrones ayuda a los responsables de implementar la seguridad, a producir diseños válidos y consistentes que incluyen todas las operaciones requeridas, y así asegurar que el resultado de las implementaciones puede ser completado eficientemente y ejecutado eficazmente.

En este artículo, estudiaremos los más importantes tipos de requisitos de seguridad para servicios Web, obtenidos a través de un proceso PWSSec [21] [22] estudiado en trabajos previos; luego estudiaremos un conjunto de patrones de seguridad que cubren

todos los requisitos de seguridad especificados, y estos patrones nos ayudarán a crear una arquitectura de seguridad de referencia donde todos los requisitos deberían ser cubiertos.

El resto del artículo está organizado como sigue: en la sección 2, mostraremos varios tipos de requisitos de seguridad para servicios Web e indicaremos brevemente la fase de PWSSec para la obtención de requisitos (WSSecReq). En la siguiente sección hablaremos sobre patrones de seguridad (diseño y arquitectura), también hablaremos sobre una arquitectura de seguridad para servicios Web (WSSecArch), daremos un catálogo de patrones agrupados juntos por tipos de requisitos que cumplen, y terminaremos esta sección mostrando un ejemplo de patrón. Finalmente, pondremos nuestras conclusiones.

2 Requisitos de Seguridad de Servicios Web

Veremos los tipos de requisitos de seguridad que creemos más importantes cuando trabajamos con Servicios Web y daremos una breve descripción de la etapa del proceso para la obtención de requisitos en servicios Web (WSSecReq).

2.1 Tipos de requisitos basados en Servicios Web

En esta sección, el conjunto de subfactores [18] [31] [41] [43] componiendo el catálogo de requisitos de seguridad de servicios Web tenidos en cuenta son los siguientes: autenticación, autorización, confidencialidad, integridad, privacidad, disponibilidad, no repudio, auditoría de seguridad, seguridad de perímetro, manejo de la confianza, delegación, federación y fiabilidad de mensajería.

a) ***Autenticación***. La autenticación en sistemas distribuidos puede ser dividida en dos procesos: i) autenticación de entidad: consistiendo en verificar la identidad demandada por un agente consumidor o usar al agente consumidor (o ambos). ii) autenticación del mensaje: consiste en saber con certeza que un cierto mensaje es auténtico, en otras palabras, que viene de la fuente que lo creó y sin ninguna modificación. b) ***Autorización***. En el contexto de WS, la autorización básicamente consiste en garantizar que sólo los correctos agentes consumidores de WS, y bajo circunstancias adecuadas, pueden acceder a los servicios ofrecidos por un cierto agente proveedor de WS. El referente estándar en este sentido es XACML (*eXtensible Access Control Mark-up Language*). c) ***Confidencialidad***: la confidencialidad en servicios Web es principalmente considerada en relación al contenido de los mensajes intercambiados. La forma más común usada para garantizar la confidencialidad del mensaje es usar criptografía. Guardar la información intercambiada entre nodos secretos de servicios Web es otro de las principales propiedades que debería garantizarse para considerar un canal seguro. d) ***Integridad***: la integridad en el contexto de los WS está principalmente relacionada con la integridad de los mensajes que son intercambiados entre servicios Web. Esta propiedad garantiza que la información recibida por un servicio Web sea la misma que la enviada por el cliente. Una simple detección podría ofrecer integridad cuando se realicen cambios accidentales en los datos. e) ***Privacidad***: el principal propósito de la privacidad en WS

consiste en garantizar la custodia correcta (es decir la divulgación correcta) de la información delicada almacenada por una cierta organización sobre sus clientes. La infraestructura completa sobre privacidad ha sido establecida por el consorcio W3C publicando las recomendaciones P3P (*Policy for Privacy Preferences*) [8], APPEL (*P3P Preference Exchange Language*) [7] y EP3P o EPAL [2]. f) ***No repudio***: en el mundo de los servicios web, es necesario ser capaz de confirmar que un cliente utilizó un servicio (solicitante de no repudio) y que el servicio procesó la petición del cliente (proveedor de no repudio). Esta cuestión de seguridad es cubierta por medio de firmas. g) ***Disponibilidad***: la necesidad de encargarse de los aspectos de disponibilidad para prevenir ataques de Denegación de Servicio, o disponer de sistemas redundantes es un punto crucial en la tecnología de los servicios Web. Sobre todo, en esos escenarios en los cuales los servicios proporcionan servicios críticos: servicios de tiempo real, servicios de listas de revocación de certificación, etc. h) ***Auditoría de Seguridad***: un servicio que registra eventos relacionados con la seguridad de forma fiable y segura produciendo un seguimiento, permitiendo la reconstrucción y examinación de una secuencia de eventos. Los eventos de seguridad podrían incluir eventos de autenticación, decisiones de ejecución de políticas, y otras. El seguimiento resultante podría ser usado para detectar ataques, conformidad con políticas, detección de abusos, u otros propósitos. i) **Seguridad del Perímetro**: tú puedes asegurar el perímetro de tu red y los recursos de información con nuestras soluciones de seguridad del perímetro dirigido al mercado. Ellos aseguran que la gente correcta puede acceder a tus recursos de red. Y ellos detectan y combaten ataques para asegurarse que la gente extraña no tenga acceso. j) ***Manejo de la Confianza:*** el manejo de la confianza es una propuesta para controlar el acceso por lo cual el acceso es concedido en base a la confianza establecida en una negociación entre el servicio solicitante y el servicio proveedor. En esta negociación, las credenciales, declaraciones firmadas que describen los atributos del propietario, son intercambiadas frecuentemente para construir confianza entre los participantes de la negociación. Las credenciales están basadas típicamente en estándares tales como X.509v3 [25], infraestructura simple de clave pública (SPKI) [12], OASIS *Security Assertion Markup Language* (SAML) [30], W3C *XML Key Management System* [20], o IBM & Microsoft WS-Trust [1]. k) ***Delegación***: La delegación es una propuesta que una entidad proporciona, todos o algunos de sus privilegios o derechos, a otras entidades. Esto es considerado una utilidad y método efectivo que acentúa la escalabilidad de un sistema distribuido y descentraliza tareas de control de acceso. El estándar de referencia *WS-Trust* [1], y SAML v2.0 proporciona un mecanismo para delegación. l) ***Federación***: Es virtualmente imposible confiar en un punto de control universal para la información de identidad. En otras palabras, ningún único administrador de seguridad tiene la responsabilidad de autenticar a todos los usuarios y manejar sus cuentas. En algunos casos, las compañías tienen muchos repositorios de identidad para sus aplicaciones, así, crean una infraestructura corporativa fragmentada dentro de un depósito de actividades. Además, cuando las compañías hacen negocio unos con otros, ellos necesitan intercambiar información sobre sus respectivos usuarios en un medio de confianza. Las compañías implicadas en la federación de identidad establecen relaciones de confianza, permitiendo a sus respectivos usuarios que accedan a los recursos hospedados en un socio. En este caso, las compañías emiten tickets de seguridad a sus usuarios para que puedan ser procesados por las

partes de confianza. m) ***Fiabilidad de mensajes***: En el mundo de los servicios web, el componente emisor de mensajes puede tolerar fallos, al enviar un mensaje repetidamente hasta que sea reconocido por el componente receptor; esta interacción puede ocurrir incluso después de que el proceso de envío haya terminado (o de lo contrario no está disponible). El componente receptor puede tolerar la indisponibilidad del proceso receptor al mantener los mensajes hasta que el proceso receptor esté listo. El estándar de referencia de OASIS *WS-ReliableMessaging* [4], y las especificaciones *WS-Reliability* [13] y HTTPR [39] proporcionan un marco de trabajo de mensajería fiable que pueden ser usadas sobre protocolos y redes abiertas.

2.2 Obtención de requisitos en PWSSec

La etapa del proceso PWSSec [21] preocupada por la obtención y especificación de requisitos de seguridad en sistemas basados en WS es la etapa anteriormente mencionada WSSecReq. El principal propósito de la etapa WSSecReq es producir, por medio de una propuesta sistemática, una especificación (o parte de ella) de los requisitos de seguridad de los sistemas basados en WS. Esta etapa está basada en el método de ingeniería de requisitos basado en reutilizar requisitos llamada SIREN (*SImple REuse of software RequiremNts*) [40] para producir los documentos iniciales de especificación de requisitos de un proyecto conteniendo los requisitos de seguridad para WS obtenidos desde un catálogo de requisitos reutilizables.

3 Patrones de Seguridad y Requisitos de Seguridad en Servicios Web

En esta sección, definiremos qué son los patrones de seguridad, tanto los patrones arquitectónicos como de diseño, también describiremos la etapa del proceso para definir una arquitectura de seguridad (WSSecArch), y estudiaremos la relación entre requisitos y patrones de seguridad, intentando dar un conjunto de patrones disponibles que cumplen y aseguran un tipo de requisito seleccionado; finalmente veremos un ejemplo de un patrón de arquitectura.

3.1 Definiendo Patrones de Seguridad

Para Ramachandran [34] un patrón es un idioma repetitivo y común de una solución de diseño y de arquitectura. Un patrón está definido como una solución a un problema en el contexto de una aplicación. Los patrones traen equilibrio en la definición de arquitectura de seguridad porque sitúan igual énfasis en buena arquitectura y fuerte seguridad. Nuestras elecciones de propiedades de seguridad, mecanismos de autenticación y modelos de control de acceso pueden, o conducir nuestra arquitectura hacia algún patrón bien definido de diseño, o volvernos hacia alguna solución ad hoc con tensiones arquitectónicas considerables.

Los patrones de seguridad proporcionan técnicas para identificar y solucionar cuestiones de seguridad, trabajan juntos para formar una colección de mejores prácticas (o soportar una estrategia de seguridad) y se dirigen al servidor, a la red y a la seguridad de la aplicación. Los beneficios de usar patrones son: pueden ser revisados e implementados en cualquier momento para mejorar el diseño de un sistema; los menos expertos se pueden beneficiar de la experiencia de los más influyentes en patrones de seguridad; proporciona un lenguaje común para argumentar, probar y desarrollar; puede ser fácilmente registrado, clasificado y refactorizado; proporciona reutilidad, se puede repetir y documentar prácticas seguras; no definen estilos codificados, lenguajes de programación o proveedores [3].

Las estrategias de diseño determinan qué tácticas de aplicación o patrones de diseño deben ser usados para escenarios y restricciones particulares de seguridad de la aplicación. Los patrones de diseño de seguridad son una abstracción de problemas de negocio que dirigen una variedad de requisitos de seguridad y proporcionan una solución a los conocidos problemas de seguridad relacionados. Pueden ser patrones de arquitectura que describen cómo un problema de seguridad puede ser resuelto arquitectónicamente, o pueden ser estrategias de diseño defensivas sobre cuyo código seguro puede ser construido más adelante [38].

En el contexto de los Servicios Web, nosotros concebimos que una metodología basada en patrones de seguridad fuera seguida, tanto a nivel de arquitectura como de diseño, a fin de facilitar la construcción de tales sistemas también como garantizar su reutilización a través de diferentes proyectos. Hoy, los arquitectos software pueden aplicar muchos de los patrones de diseño existentes para servicios Web. El uso de estos patrones pueden, en gran medida, ayudar a los arquitectos a construir arquitecturas para servicios Web escalables, fiables y robustas.

3.1.1 Patrones de Arquitectura.

Un patrón de arquitectura expresa un esquema de organización estructural fundamental para sistemas software. Proporciona un conjunto de subsistemas predefinidos, especifica sus responsabilidades, e incluye reglas y guías para organizar las relaciones entre ellos [6]. Un patrón arquitectónico es una abstracción de alto nivel. La elección del patrón arquitectónico a ser usado es una decisión de diseño fundamental en el desarrollo de un sistema software. Determina la estructura del sistema y obliga las elecciones disponibles de diseño para los diversos subsistemas. Es, en general, independiente del lenguaje de implementación a ser usado. Ejemplos de patrones arquitectónicos son "*broker*", "multi capas", "*pipe and filter*", "transacción-procesamiento", etc.

3.1.2. Patrones de Diseño.

Un patrón de diseño proporciona un esquema para refinar los subsistemas o componentes de un sistema software, o las relaciones entre ellos. Describe una estructura comúnmente recurrente de componentes de comunicación que solucionan un problema de diseño general dentro de un contexto particular. Un patrón de diseño es una abstracción de medio nivel. La elección de un patrón de diseño no afecta a la estructura fundamental del sistema software, pero afecta a la estructura de un subsistema. Como el patrón arquitectónico, el patrón de diseño tiende a ser

independiente del lenguaje de implementación a ser usado. Ejemplos de patrones de diseño son "*adapter*", "*composite*", "*delegation*", "*facade*", "*observer*", etc.

3.2 Arquitectura de Seguridad para Servicios Web en PWSSec

Una vez que los requisitos de seguridad han sido obtenidos, podemos conseguir, de una forma sistemática, una arquitectura software de seguridad que contiene un conjunto de patrones de diseño. Así, usando el mapeo adecuado, podemos expresar los requisitos de seguridad, los patrones arquitectónicos de seguridad, y los patrones de diseño de seguridad en términos de estándares de seguridad basados en servicios Web, y sus mejores prácticas y tecnologías.

PWSSec ofrece una metodología para llevar a cabo todas estas actividades en el contexto de los servicios Web, mas es razonablemente factible abstraer sus actividades a fin de que puedan ser mapeadas a otra clase de tecnologías distribuidas. Esta idea nos conduce a considerar MDA (*Model-Driven Approach Architecture*) [33] y UML como la base para describir tales mecanismos de arquitectura de seguridad y su correspondiente mapeo a las tecnologías específicas.

Un subsistema de seguridad ofrecerá uno o más servicios relativos a la seguridad. Estos servicios pueden ser clasificados en dos tipos: a) Servicio de Seguridad de la Aplicación de Seguridad: dirige un conjunto de tipos de requisitos de seguridad (por ejemplo, privacidad, autenticación del origen de datos, etc.); b) Servicio de Seguridad de Apoyo: proporciona ciertos servicios necesitados por los Servicios de Seguridad de la Aplicación de Seguridad (por ejemplo, el servicio *SecurityTokenManager*, Servicio de Delegación [42] o Servicio Cambiador de Credenciales).

La etapa WSSecArch del proceso PWSSec [23] facilita el desarrollo de los sistemas de seguridad basados en servicios Web, aplicando un conjunto de patrones arquitectónicos para servicios Web que son integrados dentro de una arquitectura de seguridad de referencia. En la fig. 1, es mostrada la entrada, salida y actividades para esta etapa.

Actualmente estamos elaborando un repositorio donde se están recogiendo patrones arquitectónicos de seguridad bien conocidos. Está siendo definido un perfil basado en servicios Web para este repositorio en el contexto de PWSSec, así que podemos saber cómo esos patrones de seguridad pueden ser aplicados en sistemas basados en servicios Web.

La salida es una completa especificación, conforme al estándar IEEE 1471-2000, de la arquitectura de seguridad desarrollada, llamada *Software Security Architecture Specification* (SASSec), indicando: i) cómo los requisitos funcionales usados como entrada en la etapa son integrados dentro de las especificaciones citadas anteriormente; ii) qué requisitos de seguridad son alcanzados y cómo son localizados en la arquitectura [4]; y iii) cuáles son los servicios Web de seguridad que necesitan ser introducidos como mecanismos de seguridad.

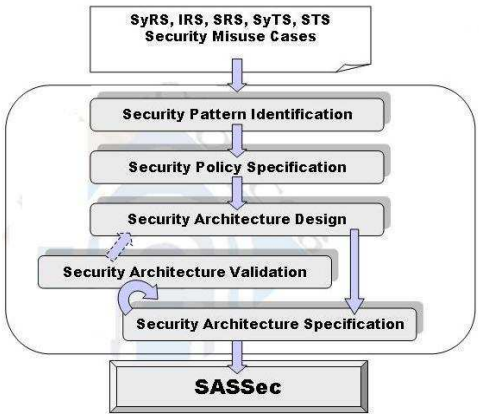


Fig. 1. Actividades de WSSecArch.

3.3 Relación entre Patrones Arquitectónicos, Patrones de Diseño y Requisitos de Seguridad

Proponemos un conjunto de patrones de seguridad que garantice, de alguna forma, uno o varios tipos de requisitos de seguridad, esto es, para un tipo de requisito de seguridad, conocemos uno o varios patrones que aseguran los requisitos mencionados. El uso de patrones nos ayuda a desarrollar un sistema seguro.

En la tabla 1, podemos ver la relación entre requisitos de seguridad , estándares de seguridad, patrones arquitectónicos de seguridad y patrones de diseño de seguridad que podemos usar para estar seguros que nuestro diseño basado en estos patrones cumplen y garantizan estos requisitos de seguridad para el sistema diseñado. Hemos seleccionados un subconjunto de los requisitos anteriormente mencionados; hemos estudiado varios patrones de seguridad que conducen y nos guía hacia un desarrollo seguro tan bien como hacia una arquitectura software de seguridad basada en patrones de seguridad. A continuación, debido a restricciones de espacio, definimos brevemente un conjunto de patrones de seguridad de entre los que aparecen en la tabla 1, para dar a conocer la funcionalidad principal de algunos de los patrones, dando referencias de cada uno de ellos para que el lector que lo desee pueda obtener mayor información sobre los patrones. Algunas descripciones de patrones son las siguientes:

- *Secure Logger:* Este patrón [38] define cómo capturar los eventos específicos de la aplicación y las excepciones de una manera segura y fiable para soportar la auditoría de seguridad.
- *Audit Interceptor:* Este patrón [38] trabaja en conjunción con el patrón *Secure Logger*, proporciona instrumentación de los aspectos de registro por delante, y la administración y manejo del registro y auditoría por detrás.
- *Message Interceptor* [38] comprueba y verifica la calidad de los mecanismos de seguridad a nivel de mensaje XML, tal como *XML Signature* y *XML Encryption* en conjunción con un token de seguridad. Este patrón también ayuda a verificar y validar mecanismos de seguridad adaptados en un mensaje SOAP cuando se

procesa por múltiples intermediarios (actores). Soporta una variedad de formatos de firmas y tecnologías de encriptación usadas por esos intermediarios.

- *Data Filter*: Este patrón [19] filtra los contenidos de las peticiones del cliente en un sistema distribuido, conforme a las políticas predefinidas. El filtrado puede ocurrir local o remotamente. En muchos sistemas distribuidos, por ejemplo, Internet, las peticiones para servicios o datos necesitan ser filtradas según las políticas de la institución, restricciones legislativas, necesidades de privacidad, etc.
- *Secure Message Router* [38] facilita la comunicación XML segura con múltiples puntos finales asociados que adoptan seguridad a nivel de mensajes y mecanismos de federación de identidad. Actúa como un componente intermediario de seguridad que aplica los mecanismos de seguridad a nivel de mensajes para entregar los mensajes a múltiples receptores donde el receptor propuesto sería capaz de acceder sólo a la porción del mensaje requerida y fragmentos del mensaje restantes serían confidenciales.
- *Authoritative Source of Data*: Este patrón [35] es usado para verificar la validez de datos y su origen. Impide al sistema del uso de información caducada e incorrecta y reduce el riesgo potencial de procesar y propagar datos fraudulentos.
- *Credential Tokenizer* [38] describe cómo un token principal de seguridad puede ser encapsulado, empotrado en un mensaje SOAP, encaminado y procesado.
- *Single Sign-on (SSO) Delegator* [38] describe cómo construir un agente delegador para manipular un sistema legado por single sign-on (SSO).
- *Secure Pipe:* Este patrón [38] muestra como asegurar la conexión entre el cliente y el servidor, o entre servidores cuando conectamos entre socios de mercado. En un entorno de aplicación distribuida compleja, habrá una mezcla de requisitos y restricciones de seguridad entre clientes, servidores y algunos intermediarios. Añade valor al requerir autenticación mutua y establecer confidencialidad o no repudio entre socios de mercado. Esto es particularmente crítico para la integración B2B usando servicios Web.

3.4 Ejemplo: Patrón de Arquitectura 'QoP'

Hemos definido el patrón de seguridad 'QoP', abstraído de los mecanismos definidos en la especificación *WS-Security* [32], la cuál dirige los requisitos de seguridad 'QoP', es decir, confidencialidad del mensaje, integridad del mensaje, y autenticación del mensaje. Este servicio Web 'QoP' define una interfaz de protección y verificación de mensajes SOAP. El estándar *WS-Security* es, en la práctica, el estándar de seguridad de servicios Web que implementa este servicio de seguridad. En nuestro ejemplo, hemos aplicado el patrón arquitectónico de servicio Web 'QoP' (*Quality of Protection*) en regla para dirigir estos requisitos de seguridad: autenticación del origen de datos, integridad y confidencialidad.

El patrón arquitectónico de servicio Web 'QoP' define un servicio Web de seguridad capaz de proteger, y verificar la protección de mensajes SOAP salientes y entrantes, respectivamente. Define una plantilla de política de seguridad que debe expresar el tipo de requisito de seguridad que su instancia cubre (autenticación del mensaje e integridad, confidencialidad), el mecanismo específico de seguridad a ser usado y la especificación del servicio Web que emplea. Nosotros podemos ver la

Tabla 1. Relación entre Requisitos, Patrones y Estándares de Seguridad

Requisitos de Seguridad	Patrones de Arquitectura	Patrones de Diseño	Estándares de Seguridad
Autenticación	QoP [32], Role-Based Security Assertion Coordinator [15], Data Filter [19], Check Point [44], SSO [29], Cryptographic [37]	Assertion Builder [38], SSO Delegator [38], Sender Authentication [5], Authenticator [14], Credential Tokenizer [38]	LibAlliance SASL based Authentication Service; SAML 2.0 WS-Security +SAML 2.0 + Kerberos Token Profile XML Key Management System WS-Security + XML Digital Signature SAML 2.0 + WS-Security + SAML Token Profile + XML Digital Signature SAML 2.0, Liberty Alliance Project ID-FF 1.1, WS-Federation
Autorización	PEP + PDP + PRP + PIP + PAP, Data Filter [19], Bodyguard [10], Check Point [44], Firewall [17]	XML Firewall Filter [11], Assertion Builder [38], Authorization [16], RBAC [16], Session [44]	WS-Policy + WS-SecurityPolicy; XACML Profile; XrML; ODRL; WS-Authorization
Confidencialidad	QoP [32], Encryption [37], Cryptographic [37], Layered Security [36]	Message Inspector [38], Information Secrecy [5], Secure Pipe [38], Session [44]	WS-Security + XML Encryption
Integridad	QoP [32], Firewall [11] [17], Data Filter [19], Layered Security [36]	Message Inspector [38], Secure Pipe [38], Message Integrity [5], Secure Message Router [38], Authoritative Source of Data [35], Multilevel Security [16]	WS-Security + XML Digital Signature
Auditoría	Check Point y Single Access Point [44]	Audit Interceptor [38], Secure Logger [38]	

definición de este patrón siguiendo una plantilla que consta de los siguientes elementos, y que aplicamos para este patrón: i) *Dirigiendo un conjunto notorio de tipos de requisitos de seguridad*: Autenticación del origen de datos, Confidencialidad e Integridad de las Comunicaciones; ii) *Motivación e impulsos*. a) Motivación: Seguridad del canal del mensaje básico; b) Impulsos: infraestructuras y tecnología subyacentes heterogéneas de seguridad de empresa, 'QoP' puede ser obtenida por medio de una combinación de mecanismos de seguridad a nivel de mensaje y en capas subyacentes (transporte, red, …); iii) *Solución*: se propone un esquema de solución basada en servicios de seguridad (elementos e interacciones); define una plantilla de políticas de seguridad; contiene guías para su integración dentro de la arquitectura funcional basada en servicios Web; iv) *Estándares de seguridad basados en servicios Web o soluciones de investigación notorias*: OASIS WS-Security 1.0; v) *Implementaciones y usos conocidos*: Apache WSS4J, VordelDirector, XTradyne's WS-DBC, …; vi) *Patrones relacionados*: Security Token Manager.

Este patrón protege el mensaje (integridad, confidencialidad y autenticación), por ello si tenemos un servicio que envíe y reciba mensajes SOAP, debemos estar seguros que los mensajes son válidos y no producen ninguna amenaza para la aplicación, es decir, podemos usar este patrón para establecer una arquitectura básica de protección y verificación de la protección en mensajes SOAP.

4 Conclusiones

Los arquitectos toman decisiones de diseño muy temprano en el ciclo de vida del proyecto. Muchos de ellas son difíciles, si no imposibles, validar y probar partes del sistema que están actualmente construidas. Debido a la dificultad de validar decisiones de diseño muy temprano, los arquitectos confían sensiblemente en los métodos probados y ensayados para solucionar ciertas clases de problemas. Esto es uno de los grandes valores de los patrones arquitectónicos. Permiten a los arquitectos reducir el riesgo con diseños apropiados con atributos de ingeniería conocidos.

Los patrones de seguridad ayudan a no perder de vista los requisitos no-funcionales de seguridad al inicio del diseño. En las aplicaciones críticas de seguridad es extremadamente importante evitar errores, ya que se debe garantizar la seguridad de dichas aplicaciones y otorgar un alto nivel de seguridad a todas las operaciones e interacciones que se hagan en la aplicación. Por tanto, el uso de los patrones de seguridad es importante para desarrollar un sistema seguro.

Este artículo ha presentado un catálogo de patrones de seguridad tanto arquitectónicos y de diseño destinados a Servicios Web, basados en algunos tipos de requisitos de seguridad que podemos obtener mediante, por ejemplo, con el proceso de obtención de requisitos presentado en este artículo (WSSecReq), obteniendo los más importantes tipos de requisitos de seguridad que todo servicio Web debería cumplir.

Tomaremos esos requisitos de seguridad especificados para crear un borrador de un candidato a arquitectura de seguridad para Servicios Web. Esta actividad capacita las decisiones arquitectónicas a través de, un análisis de riesgo bien definido y de los cambios de procesos de análisis (PWSSec) en orden, para identificarlos y saber cómo mitigarlos. Esta arquitectura candidata también identificará un conjunto de patrones de seguridad que cubren todos esos requisitos de seguridad dentro del componente de arquitectura y los detallarán en forma de alto nivel, localizando los riesgos conocidos, exposiciones y vulnerabilidades.

Nuestro trabajo futuro se basará en el estudio de distintos patrones de seguridad y de obtener un método con el que clasificar qué patrón es mejor, para el tipo de requisito seleccionado, de entre los patrones posibles a usar, guiándonos para clasificarlos con alguna propiedad de seguridad (rendimiento, fiabilidad, grado de seguridad, flexibilidad, etc).

5 Agradecimientos

Esta artículo ha sido desarrollado en el contexto de los proyectos DIMENSIONS (PBC-05-012-2) financiado por FEDER y por la Consejería de Ciencia y Tecnología de la Junta de Comunidades de Castilla-La Mancha, RETISTIC (TIC2002-12487-E) y CALIPO (TIC2003-07804-CO5-03), de la Dirección General de Investigación del Ministerio de Ciencia y Tecnología (España).

Referencias

1. Anderson, S., Bohren, J., Boubez, T., Chanliau, M., Della-Libera, G., Dixon, B., Garg, P., Gudgin, M., Hallam-Baker, P., Hondo, M., Kaler, C., Lockhart, H., Martherus, R., Maruyama, H., Nadalin, A., Nagaratnam, N., Nash, A., Philpott, R., Platt, D., Prafullchandra, H., Sahu, M., Shewchuk, J., Simon, D., Srinivas, D., Waingold, E., Waite, D., Walter, D., y Zolfonoon, R., Web Services Trust Language (WS-Trust). 2005, IBM, Microsoft and Actional, BEA, Computer Associates, Layer 7, Oblix, OpenNetwork, Ping Identity, Reactivity, and Verisign. ftp://www6.software.ibm.com/software/developer/library/ws-trust.pdf
2. Ashley, P., Hada, S., Karjoth, G., y Schunter, M. E-P3P privacy policies and privacy authorization. in Workshop on Privacy in the Electronic Society, WPES'02. 2002. Washintong, DC, USA.
3. Berry, C.A., Carnell, J., Juric, M.B., Kunnumpurath, M.M., Nashi, N., y Romanosky, S., Chapter 5: Patterns Applied to Manage Security, in J2EE Design Patterns Applied. 2002.
4. Bilorusets, R., Bosworth, A., Box, D., Cabrera, L.F., Collison, D., Ferguson, D., Ferris, C., Freund, T., Hondo, M.A., Ibbotson, J., Kaler, C., Langworthy, D., Lewis, A., Limprecht, R., Lucco, S., Mihic, M., Mullen, D., Nadalin, A., Nottingham, M., Orchard, D., Samdarshi, S., Shewchuk, J., y Storey, T., Web Services Reliable Messaging Protocol (WS-ReliableMessaging). 2004, BEA, IBM, Microsoft. ftp://www6.software.ibm.com/software/developer/library/ws-reliablemessaging200403.pdf
5. Braga, A.M., Rubira, C., y Dahab, R. Tropyc: A Pattern Language for Cryptographic Software. in 5th Pattern Languages of Programming (PLoP'98) Conference. 1998. Allerton Park, Illinouis, USA.
6. Buschmann, F., Meunier, R., Rohnert, H., Sommerlad, P., y Stal, M., Pattern-Oriented Software Architecture: A System of Patterns. 1st ed. 1996: John Wiley & Sons. 476 Pg.
7. Cranor, L., Langheinrich, M., y Marchiori, M., A P3P Preference Exchange Language 1.0 (APPEL1.0). W3C Working Draft, 2002.
8. Cranor, L., Langheinrich, M., Marchiori, M., Presler-Marshall, M., y Reagle, J., The Platform for Privacy Preferences 1.0 (P3P1.0) Specification. W3C Recommendation, 2002.
9. Curbera, F., Nagy, W.A., y Weerawarana, S., Web Services: Why and How. IBM T.J. Watson Research Center, 2001.
10. Das Neves, F. y Garrido, A., BodyGuard, in Pattern Languages of Programs III, Addison-Wesley, Editor. 1998.

11. Delessy-Gassant, N., Fernandez, E.B., Rajput, S., y Larrondo-Petrie, M.M. Patterns for Application Firewalls. in 11th Conference on Pattern Languages of Programs (PLoP'2004). 2004. Allterton Park, Monticello, Illinois.

12. Ellison, C., Frantz, B., Lampson, B., Rivest, R., Thomas, B., Bell, S., y Ylonen, T., SPKI Certificate Theory. 1999, Internet Eng. Task Force RFC 2693. http://www.rfc-editor.org/rfc/rfc2693.txt

13. Evans, C., Chappell, D., Bunting, D., Tharakan, G., Shimamura, H., Durand, J., Mischkinsky, J., Nihei, K., Iwasa, K., Chapman, M., Shimamura, M., Kassem, N., Yamamoto, N., Kunisetty, S., Hashimoto, T., Rutt, T., y Nomura, Y., Web Services Reliability (WS-Reliability) Ver1.0. 2003, Fujitsu Limited, Hitachi, Ltd., NEC Corporation, Oracle Corporation, Sonic Software Corporation, and Sun Microsystems, Inc. otn.oracle.com/tech/webservices/ htdocs/spec/WS-ReliabilityV1.0.pdf

14. F. Lee Brown, J., DiVietri, J., Diaz de Villegas, G., y Fernandez, E.B. The Authenticator Pattern. in 6th Conference on Pattern Languages of Programs, PLoP 1999. 1999. Allterton Park, Monticello, Illinois.

15. Fernandez, E.B. Two patterns for web services security. in The 2004 International Symposium on Web Services and Applications. 2004. Las Vegas, Nevada, USA.

16. Fernandez, E.B. y Pan, R. A pattern language for security models. in 8th Conference on Pattern Languages of Programs, PLoP 2001. 2001. Allerton Park, Illinois, USA.

17. Fernandez, E.B., Petrie, M.L., Seliya, N., y Herzberg, A. A Pattern Language for Firewalls. in 10th Conference on Pattern Languages of Programs (PLoP'2003). 2003. Allterton Park, Monticello, Illinois.

18. Firesmith, D.G., Specifying Reusable Security Requirements. Journal of Object Technology, 2004. 3: p. 61-75.

19. Flanders, R. y Fernandez, E.B. Data Filter Architecture Pattern. in 6th Conference on Pattern Languages of Programs, PLoP 1999. 1999. Allterton Park, Monticello, Illinois.

20. Ford, W., Hallam-Baker, P., Fox, B., Dillaway, B., LaMacchia, B., Epstein, J., y Lapp, J., XML Key Management Specification (XKMS). W3C Note 30. 2001, VeriSign Inc, Microsoft Corporation, webMethods Inc. http://www.w3.org/TR/xkms/

21. Gutiérrez, C., Fernández-Medina, E., y Piattini, M. PWSSec: Process for Web Services Security. in IEEE International Conference on Web Services. 2005. Orlando, Florida, USA.

22. Gutiérrez, C., Fernández-Medina, E., y Piattini, M. Towards a Process for Web Services Security. in WOSIS'05. 2005. Miami, Florida, USA.

23. Gutiérrez, C., Fernández-Medina, E., y Piattini, M. Web Services Enterprise Security Architecture: a Case Study. in Workshop on Security on Web Services. 2005. Fairfax, Virginia, USA: ACM Press.

24. Gutiérrez, C., Fernández-Medina, E., y Piattini, M., Web Services Security: is the problem solved? Information Systems Security, 2004. 13: p. 22-31.

25. Housley, R., Ford, W., Polk, W., y Solo, D., Internet X.509 Public Key Infrastructure Certificate and CRL Profile. 1999, Internet Eng. Task Force RFC 2459. www.rfc-editor.org/rfc/rfc2459.txt

26. IBM, Security in a Web Services World: A Proposed Architecture and Roadmap, in *White Paper from IBM Corporation and Microsoft Corporation*. 2002

27. IDC. 2005. http://www.idc.com/getdoc.jsp?containerId=prUS00190705

28. Imamura, T. y Tatsubori, M. Patterns for Securing Web Services Messaging. in OOPSLA'03 Workshop for Web Services and Service Oriented Architecture Best Practice and Patterns. 2003. Anaheim, California, USA.

29. King, C., Osmanoglu, E., y Dalton, C., Security Architecture. 2001: McGraw-Hill.

30. OASIS, Assertions and Protocol for the OASIS Security Assertion Markup Language (SAML) V1.1. 2003, OASIS Standard. http://www.oasis-open.org/committees/download.php/3406/oasis-sstc-saml-core-1.1.pdf

31. OASIS, Quality Model for Web Services. 2005

32. OASIS, Web Services Security (WS-Security)- Specification 6 April 2004. 2004

33. OMG, MDA (Model-Driven Architecture). 2000. http://www.omg.org/mda/

34. Ramachandran, J., Designing Security Architecture Solutions. 2002: John Wiley & Sons.

35. Romanosky, S. Enterprise Security Patterns. in 7th European Conference on Pattern Languages of Programs (EuroPlop'02). 2002. Irsee, Germany.

36. Romanosky, S., Security Design Patterns. 2001. http://www.cgisecurity.com/lib/securityDesignPatterns.html

37. Schumacher, M. y Roedig, U. Security Engineering with Patterns. in 8th Conference on Patterns Lnaguages of Programs, PLoP 2001. 2001. Monticello, Illinois, USA.

38. Steel, C., Nagappan, R., y Lai, R., Core Security Patterns. 2005: Prentice Hall PTR. 1088 Pg.

39. Todd, S., Parr, F., y Conner, M., A Primer for HTTPR. An overview of the reliable HTTP protocol. 2001. http://www-128.ibm.com/developerworks/webservices/library/ws-phtt/

40. Toval, A., Nicolás, J., Moros, B., y García, F., Requirements Reuse for Improving Information Systems Security: A Practitioner's Approach. Requirements Engineering Journal, 2001. 6(4): p. 205-219.

41. W3C, Web Services Architecture. 2004, W3C Working Group. http://www.w3.org/TR/2004/NOTE-ws-arch-20040211/

42. Wang, J., Vicchio, D.D., y Humphrey, M. Extending the Security Assertion Markup Language to Support Delegation for Web Services and Grid Services. in International Conference on Web Services (ICWS'05). 2005. Orlando, Florida, USA: IEEE Press.

43. WS-I, Security Challenges, Threats and Countermeasures Version 1.0. 2005. http://www.ws-i.org/Profiles/BasicSecurity/SecurityChallenges-1.0-20050507.doc

44. Yoder, J. y Barcalow, J. Architectural Patterns for Enabling Application Security. in 4th Conference on Patterns Language of Programming, PLop 1997. 1997. Monticello, Illinois, USA.