Research

REVISTA Ingeniería

Knowledge Base Model for Security Audits in Web Services with SQL Injection

Modelo base de conocimiento para auditorías de seguridad en servicios web con inyección SQL

John Edison Moreno Marín [*,1], ***Paulo Cesar Coronado Sánchez*** [**] [1]

[1] Universidad Distrital Francisco José de Caldas (Bogotá-Colombia)

[*] Correspondence e-mail: jemorenom@correo.udistrital.edu.co

[**] paulo_cesar@udistrital.edu.co

Recibido: 23/12/2020. Modificado: 30/06/2020. Aceptado: 14/07/2020.

Abstract

Context: Due to the large number of cyber-attacks at international and national levels (Colombia), preventive mechanisms and procedures have been triggered by organizations in order to counteract vulnerabilities in information security. The issue studied by this project arises from the need to make a proposal to the DIAN information security office to implement and follow up on the MinTIC Online Government Strategy in the Information Security and Privacy component, through the institutional information security policy and through this knowledge base model for audits in web services, applied to a particular prototype.

Method: The general methodology for the knowledge base model the first corresponds to the collection, processing, and purification of the base, and the second corresponds to the systematization process of the proposed model. OpenKM (an open software) was implemented to support the knowledge base. For the development of the audit, it is important to keep in mind that, within the general methodology, a series of guides were included in each of the phases of the model. The project uses standards, good practices, tools, and professional advice such as ISO27000, OSSTMM, OWASP, JUnit, and the Risk Management and Audit guides issued by MinTIC. For the development of the prototype with the presented WS, the OPENUP method was used. The implementation was limited to the construction of two HTTP methods: GET and POST for consultation and information entry actions.

Results: With this project, it was possible to create a knowledge base model implemented on OpenKM, executing a web services security audit with SQL Injection on an organizational prototype.

Conclusions: It must be taken into account that there will never be a 100 % secure infrastructure, since there will always be risks on the platforms due to the changing nature of the attacks. However, there will always be alternatives such as this base model of information security auditing to avoid or mitigate such risks or attacks.

Keywords: Information model, information security, computer security, web services, web application auditing, SQL Injection, knowledge base, ontologies, taxonomies.

Language: Spanish

Open access

Cite this paper as: J. Moreno y P. Coronado: "Modelo base de conocimiento para auditorías de seguridad en servicios web con inyección SQL", Ingeniería, Vol. 25, Num. 3, pp. 264-283 (2020).

© The authors; reproduction right holder Universidad Distrital Francisco José de Caldas.

DOI: https://doi.org/10.14483/23448393.15740

Resumen

Contexto: Debido a la gran cantidad de ciberataques a nivel internacional y nacional (Colombia), se han activado mecanismos y procedimientos preventivos en las organizaciones para contrarrestar estas vulnerabilidades en la seguridad de la información. El tema de este proyecto surge de la necesidad de hacer una propuesta a la oficina de seguridad de la información de la DIAN para implementar y dar seguimiento a la estrategia de Gobierno en línea de MinTIC en el componente de seguridad de la información y privacidad, ello a través de la política de seguridad de la información institucional y de este conocimiento aplicado al modelo base para auditorías en servicios web, aplicado a un prototipo particular.
Método: La metodología general para el modelo base de conocimiento incluye dos partes del trabajo. La primera corresponde a la recolección, procesamiento y purificación de la base. La segunda corresponde al proceso de tematización del sistema del modelo propuesto. OpenKM (un *software* abierto) se implementó para sustentar la base de conocimiento. Para el desarrollo de la auditoría es importante tener en cuenta que dentro de la metodología general se incluyeron una serie de guías en cada una de las fases del modelo. El proyecto utiliza estándares, buenas prácticas, herramientas y asesoramiento profesional como ISO27000, OSSTMM, OWASP, JUnit y las guías de gestión de riesgos y auditoría emitidas por el MinTIC. Para el desarrollo del prototipo con el WS a exponer, se utilizó el método OPENUP. La implementación se limitó a la construcción de dos métodos HTTP, GET y POST para consultas y acciones de entrada de información.
Resultados: Con este proyecto fue posible crear un modelo de base de conocimiento implementado en OpenKM, ejecutando una auditoría de seguridad de servicios web con inyección de SQL en un prototipo organizacional.
Conclusiones: Se debe tener en cuenta que nunca habrá una infraestructura 100 % segura, ya que siempre habrá riesgos en las plataformas debido a la naturaleza cambiante de los ataques. Sin embargo, siempre habrá alternativas como esta, un modelo base de auditoría de seguridad de la información para evitar o mitigar tales riesgos o ataques.
Palabras clave: Auditoría de aplicaciones web, base de conocimiento, inyección SQL, modelo de información, ontologías, seguridad de la información, seguridad informática, servicios web, taxonomías.
Idioma: Español

1. Introducción

Desde el momento en que una aplicación web de tipo organizacional es desplegada en un ambiente de producción bajo una determinada infraestructura es susceptible de recibir ataques malintencionados. Prevenir que esos ataques resulten exitosos es uno de los principales intereses del área de seguridad de cualquier entidad gubernamental o privada.

El proyecto se enfoca en dos áreas complementarias: (a) modelos de base de conocimiento del dominio de seguridad de aplicaciones web [1] y (b) auditorías de vulnerabilidad a vectores de ataque SQL Injection en servicios web [2]. Lo anterior abordado desde los fundamentos, estándares, métodos y conceptos técnicos [3].

Este proyecto en modalidad de profundización busca crear un modelo base de conocimiento para guiar procesos de auditoría de seguridad web en un prototipo organizacional, frente a los riesgos informáticos a los que esté expuesto. En concordancia con los recientes ataques cibernéticos a las

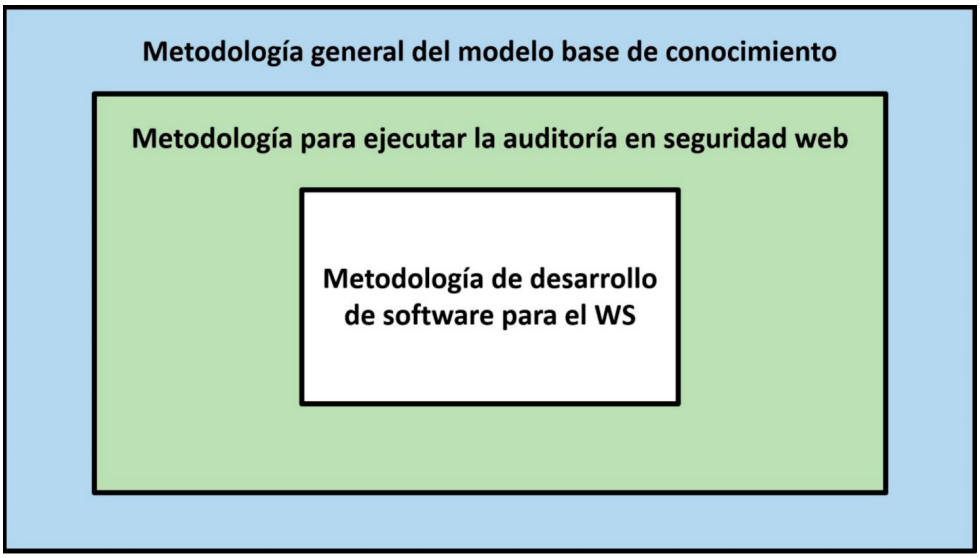

Figura 1. Ejemplificación de metodologías utilizadas en el proyecto.

plataformas de las grandes multinacionales, se ofrecen alternativas de solución con el fin de contribuir en el campo de investigación de la seguridad de la información [4].

A su vez, se busca a futuro evaluar el estado de seguridad en los servicios web expuestos en los aplicativos en producción de una entidad. Enmarcado en un modelo de información específico, y mediante herramientas y técnicas del modelo, se permitirá la visualización y definición de un estado de seguridad de su información digital y de los datos expuestos de dicha organización [5].

Este proyecto bajo la modalidad de profundización ejecuta una prueba del modelo en el marco del área de seguridad de la información, específicamente la seguridad informática, para mitigar ataques de tipo SQL Injection (SQLi) como vector de ataque de este proyecto [5]. La taxonomía del modelo se desarrolló con ideas propias y extrayendo ideas de varias investigaciones previas, las cuales se irán referenciando a medida que se adelanta el documento [6].

2. Metodología

Para la elaboración de este modelo, se utilizaron varias metodologías integradas una dentro de otra, las cuales se ejemplifican en un diagrama (Figura 1).

2.1. Metodología general del modelo base de conocimiento

Esta sección constituye la definición metodológica general para el desarrollo del modelo en sus diferentes elementos; por lo tanto, se contextualizan varias investigaciones, artículos, trabajos y definiciones de los componentes esenciales al momento de realizar un análisis a los modelos de base de conocimiento. Este proyecto incluye dos tipos de metodologías de trabajo. La primera corresponde al diseño del prototipo del sistema recolección, procesamiento y depuración de la base de conocimiento para preparar el modelo prototipo. La segunda corresponde al proceso de sistematización del modelo de diagnóstico. Ambas metodologías tienen como soporte la investigación y,

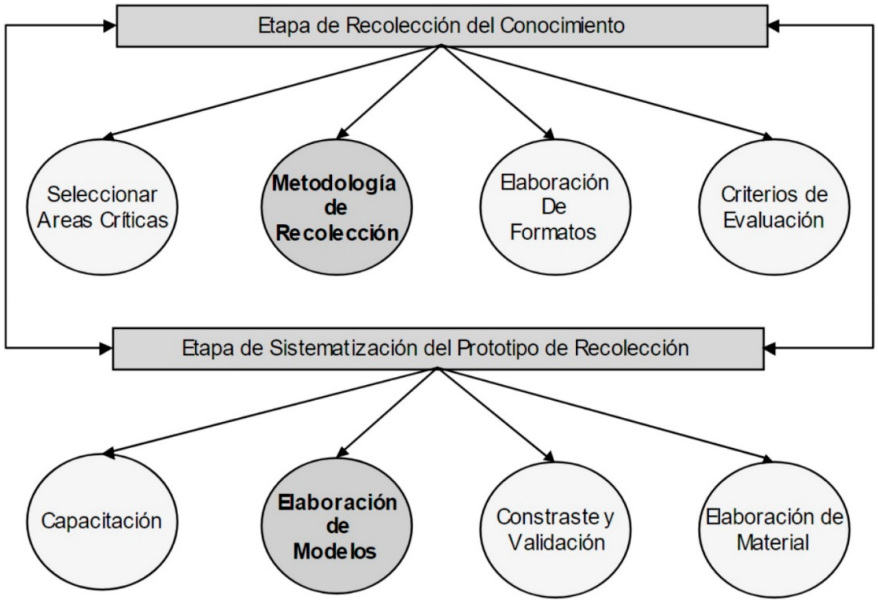

Figura 2. Metodología de desarrollo para la Base de Conocimiento Fuente: [7].

aunque se dan de manera simultánea, para efectos de presentación se analizan por separado (Figura 2).

Una base de conocimiento tiene como objetivo principal modelar y almacenar bajo forma digital un conjunto de conocimiento, ideas, conceptos o datos que permitan ser consultados o utilizados en la base de conocimientos. Existen varios métodos y programas para crear bases de conocimientos [8].

2.2. Metodología para ejecutar la auditoría en seguridad web

Para la preparación de la auditoría se presentará la metodología, los tiempos y recursos que se utilizarán. Se recolecta y analiza la información evidenciando los hallazgos, las oportunidades de mejora y las fortalezas encontradas durante la auditoría. Una vez se culmine, se presentarán las conclusiones. Con base en el informe de la auditoría, se establecerán las acciones de mejora pertinentes [9].

La metodología inicia con un proceso de planeación. En esta se fijan los objetivos y las herramientas a usar, esto implica qué hacer, cómo hacerlo y cuándo hacerlo. Esta etapa incluye una investigación previa con el fin de conocer la operación de lo que se va a evaluar [9].

Para el desarrollo de la auditoría es importante tener presente que dentro de la metodología general se incluyeron una serie de guías en cada una de las fases del modelo. El proyecto hará uso de estándares, buenas prácticas, herramientas y consejos profesionales tales como: NTC-ISO-IEC 27000, 27001, 27002, 27005, 31000, OSSTMM, OWASP, JUnit y las guías de gestión de riesgos y auditoría emitidas por el MinTIC. Todo esto aplicado al vector de ataque SQL Injection [10], [11].

En la Figura 3, extraída del reporte de seguridad de IBM con la herramienta X-Force IRIS (por sus

siglas en inglés de *Incident Response and Intelligence Services*), entre 2011 y 2019 se muestra cómo los ataques de inyección de SQL han afectado diferentes tipos de organizaciones o industrias de diferentes países en 2011. También se determina que en el 2012 hubo la mayor cantidad de ataques de este tipo, 79 en total. A partir de ahí fue disminuyendo; sin embargo, no han desaparecido totalmente, incluso en enero de este año ocurrió un ataque en Dinamarca [12].

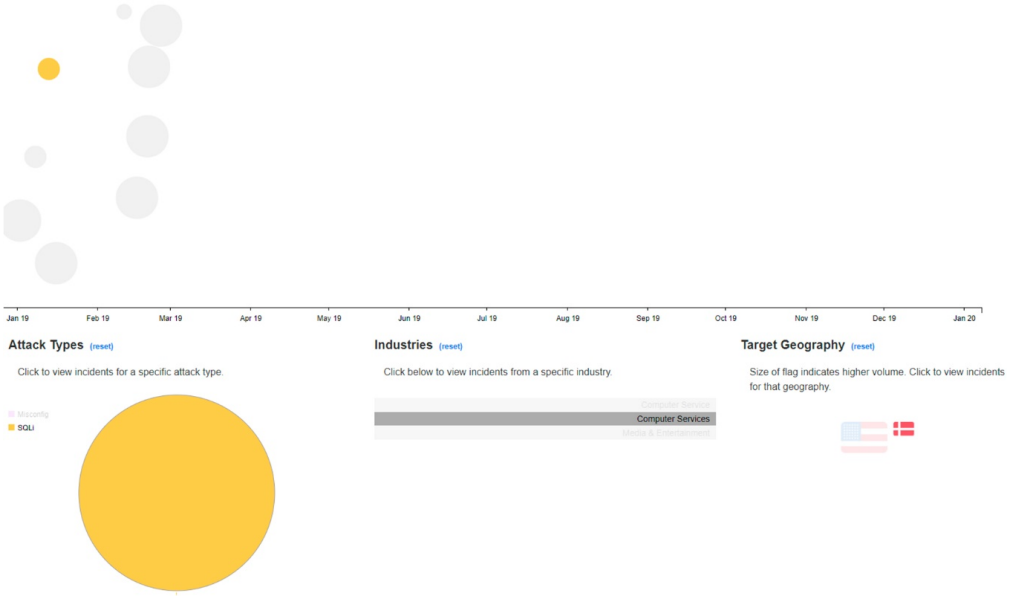

Figura 3. Evolución de ataques de inyección SQL durante 2019 Fuente: [13].

2.3. Metodología de desarrollo de *software*

Para crear la metodología de desarrollo de este proyecto se usó el método OpenUP. Este se define como un proceso unificado ligero que aplica enfoques iterativos e incrementales dentro de un ciclo de vida estructurado. OpenUP adopta una filosofía pragmática y ágil que se centra en la naturaleza colaborativa del desarrollo de *software*. Es un proceso agnóstico de herramientas y de baja ceremonia que se puede extender para abordar una amplia variedad de tipos de proyectos.

Para las pruebas a la aplicación desarrollada se utilizó JUnit, un marco de código abierto diseñado con el propósito de escribir y ejecutar pruebas en el lenguaje de programación Java. JUnit, originalmente escrito por Erich Gamma y Kent Beck, ha sido importante en la evolución del desarrollo basado en pruebas, lo cual forma parte de un paradigma de diseño de *software* más amplio conocido como *extreme programming* (XP).

2.4. Modelo de arquitectura del modelo base de conocimiento

La arquitectura de la solución se desarrollará bajo la arquitectura cliente-servidor del lado de la base de conocimiento y de componentes independientes del lado de la auditoría del web service.

Arquitectura base de conocimiento en OpenKM (cliente-servidor):

Servidor	Apache Tomcat: https://ritaportal.udistrital.edu.co:10229
Cliente	Cualquier cliente web.
Cliente	Glassfish: https://ipservidor.:8080/FacturaElectronica/test-resbeans.html
Servidor	Cualquier cliente dentro del segmento de red privado.

Arquitectura del modelo base de conocimiento para auditoría de web service (Figura 4).

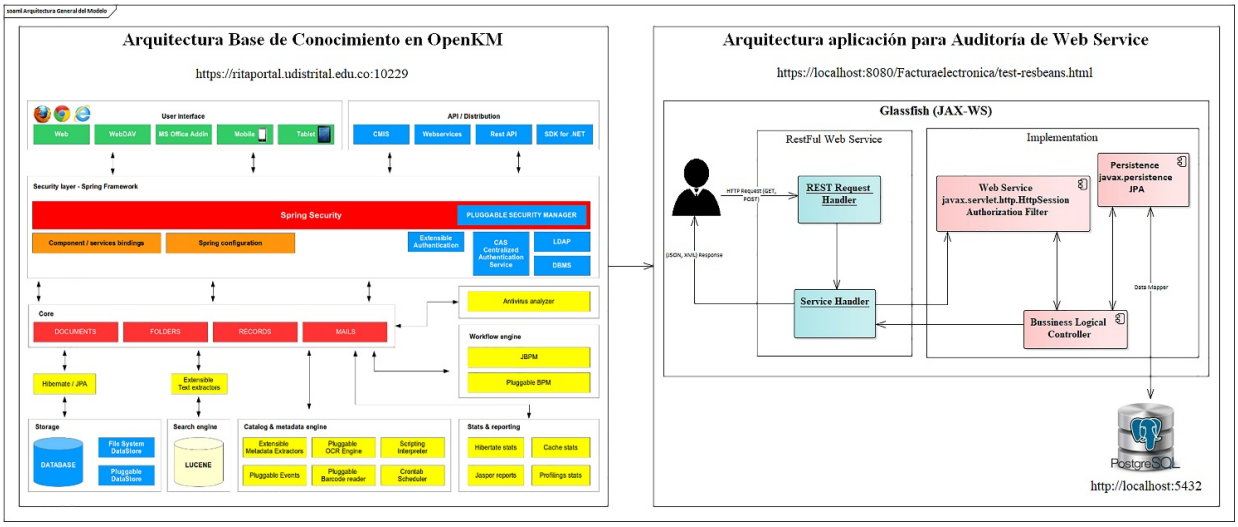

Figura 4. Arquitectura del modelo general Fuente: [14].

El objetivo consiste en aplicar la revisión de SQL Injection a un dominio de información (Factura Electrónica Web) para evitar ataques en servicios web mediante una auditoría de seguridad (Figuras 5, Figura 6 y Figura 7).

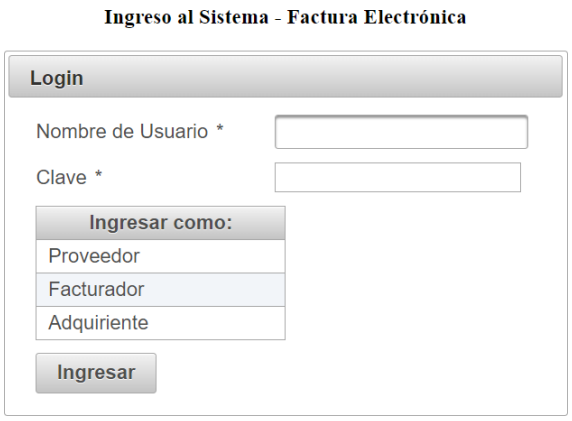

Figura 5. Inicio de sesión general en el prototipo.

Considerando los requisitos funcionales, se establecen los comportamientos deseados del aplicativo Factura Electrónica Web, los cuales fueron traducidos en especificaciones técnicas utilizando el lenguaje de descripción de aplicaciones web (WADL) utilizado por la herramienta Netbeans, que permitió desarrollar la construcción del servicio web RESTFul. Este permite escoger el método (GET, utilizado únicamente para consultar información al servidor, o POST, utilizado para solicitar

la creación de un nuevo registro) y el formato (XML, un lenguaje de marcado extensible, o JSON, un formato ligero de intercambio de datos) para la consulta o ingreso de datos de la aplicación (Figura 8).

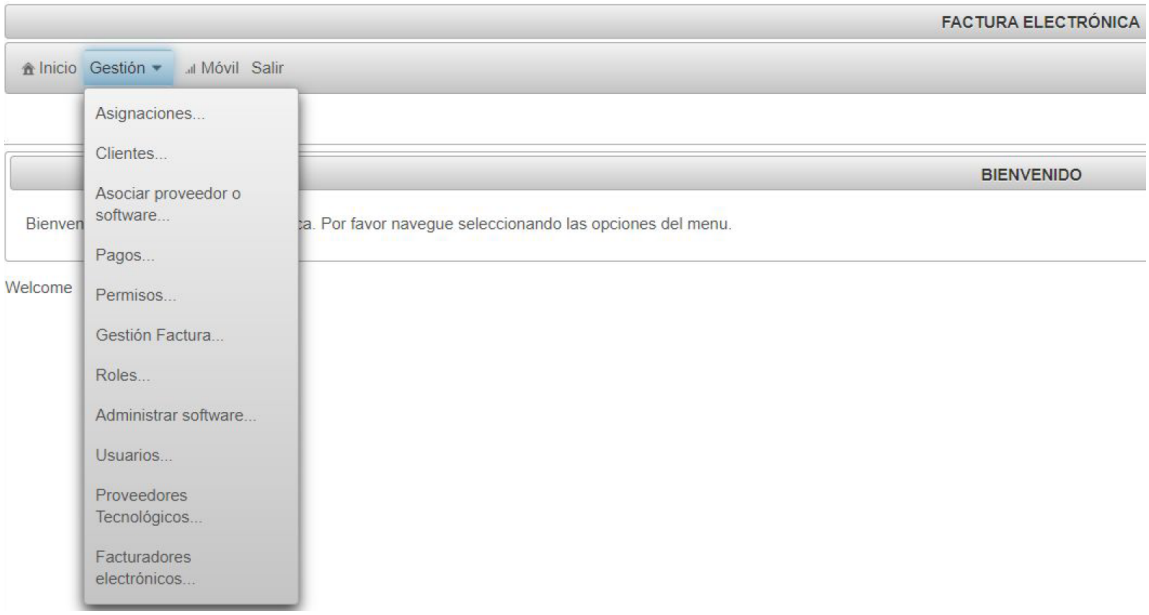

Figura 6. Menú general con los módulos de gestión del prototipo.

Figura 7. Ingreso a módulo de gestión de facturas electrónicas vía web.

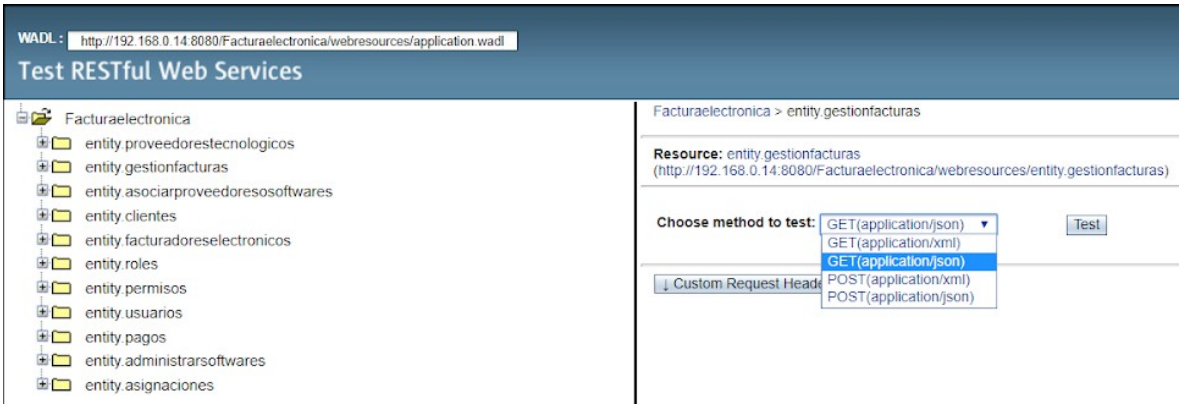

Figura 8. Interfaz gráfica del web *service* RESTFul.

Se virtualizó Kali Linux para ejecutar la propuesta de auditoría sobre el Web Service de la aplicación Factura Electrónica Web (Figura 9).

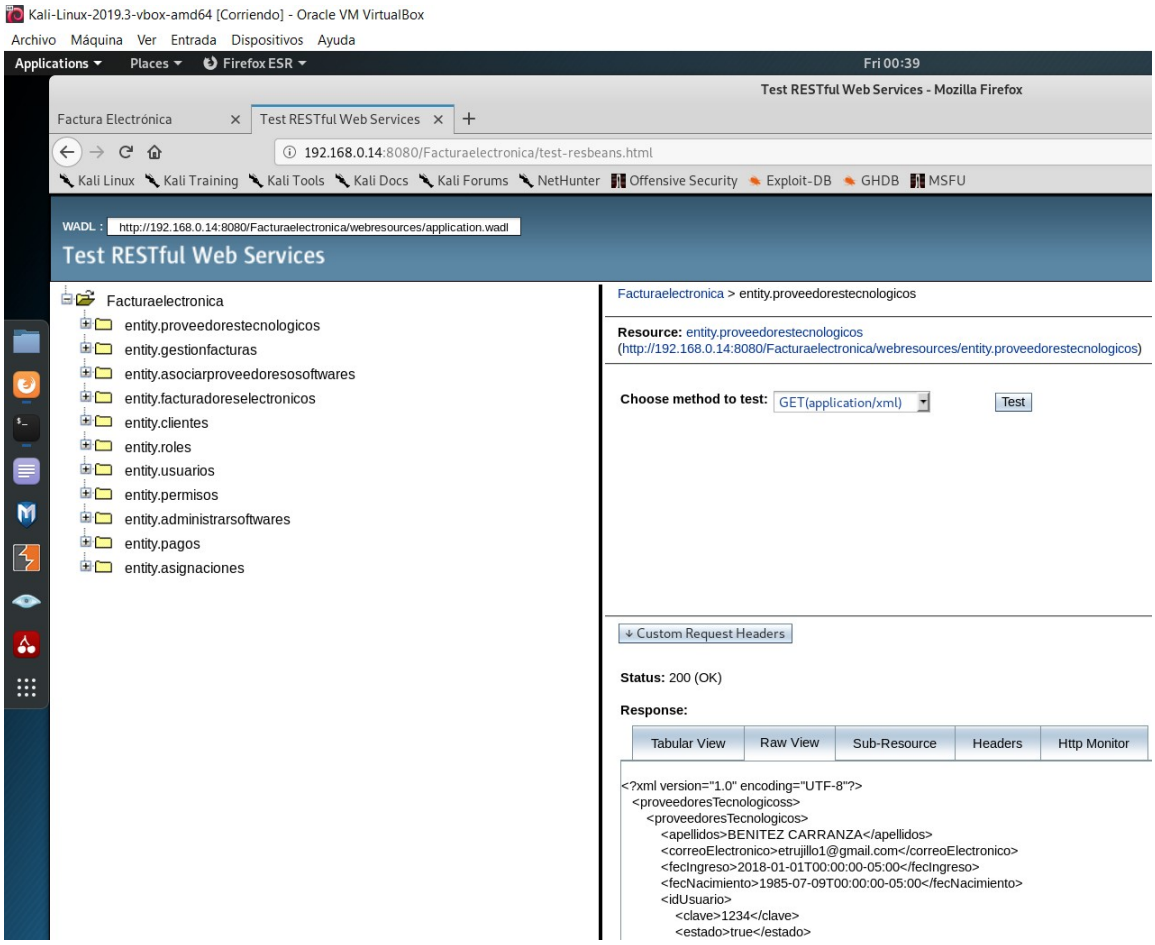

Figura 9. Acceso desde Kali al servicio web registrado para pruebas de auditoría con SQLi.

Se realizaron pruebas unitarias con la herramienta JUnit a la clase encargada de gestionar la seguridad para el aplicativo y el web service (Figura 10).

3. Resultados

• Implementación del estilo arquitectónico en el prototipo.

Para implementar el prototipo se utilizó el estilo arquitectónico por capas ejemplificado en un modelo vista-controlador (MVC). Estos sistemas constituyen uno de los estilos que aparecen con mayor frecuencia mencionados como categorías mayores del catálogo, o, por el contrario, como una de las posibles encarnaciones de algún estilo más envolvente. En [15] definen el estilo en capas como una organización jerárquica, tal que cada capa proporciona servicios a la capa inmediatamente superior y se sirve de las prestaciones que le brinda la inmediatamente inferior. Instrumentan así una vieja idea de organización estratigráfica que se remonta a las concepciones formuladas por el patriarca Edsger Dijkstra en la década de 1960, largamente explotada en los años subsiguientes.

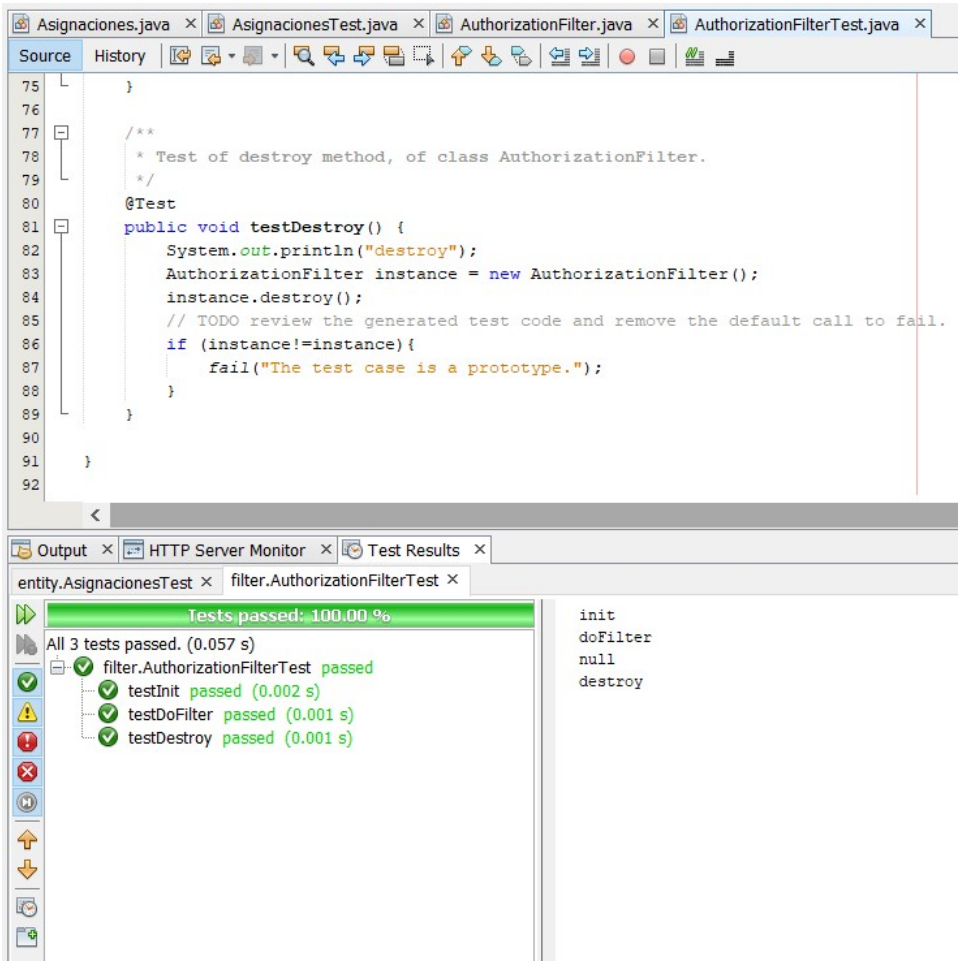

Figura 10. Pruebas con JUnit para Clase AuthorizationFilter.

Se definió el modelo de datos e identificadores que representan los recursos a exponer. La construcción del WS se limitó a la implementación de la tecnología de servicio web RESTful (*Representational State Transfer, based on Rest architecture*), con dos métodos HTTP, GET y POST, para especificar las acciones de consulta y registro de información respectivamente. Cabe aclarar que se estableció así, de acuerdo con el masivo uso que se tiene en las entidades y organizaciones del estado colombiano (probablemente por su potencial escalable, así como el acceso con escaso consumo de recursos). Sin embargo, si se prefiere implementar SOAP por su robustez, la seguridad, el control y la validación que ofrecen los esquemas debido a su alto acoplamiento, se podría usar en integraciones de cores bancarios o pasarelas de pago. Para las fases de inicio, elaboración, construcción y transición de la aplicación y del servicio web de la aplicación objeto de auditoría se implementó el método OpenUP en cuatro iteraciones, una por cada fase.

Fase inicial: En primera instancia, se realizaron reuniones de trabajo entre el director del proyecto y el estudiante, en las que se explicó la visión de la aplicación, la metodología y el entorno en que se desenvolverá el proyecto. Finalmente se seleccionó la tecnología y técnica a utilizar en el desarrollo del prototipo de aplicativo del proyecto.

- Caracterización previa de la aplicación objeto de auditoría.

Fase elaboración: Tomando en cuenta los requisitos funcionales y no funcionales se implementó la aplicación, la cual soporta la solución de Factura Electrónica Web y se crea el inicio de sesión general de dicho aplicativo con un tipo de usuario.

- Iteración 2. Implementación del aplicativo Factura Electrónica Web como prototipo.

Fase implementación: Considerando los requisitos funcionales, se establecen los comportamientos deseados del aplicativo Factura Electrónica Web, los cuales fueron traducidos en especificaciones técnicas a través del lenguaje de descripción de aplicaciones web (WADL) utilizado por la herramienta Netbeans. Este permitió desarrollar la construcción del servicio web RESTFul, el cual posibilitó escoger el método (GET o POST) y el formato (XML) para la consulta o ingreso de datos de la aplicación.

- Iteración 3. Implementación del web service a través del IDE Netbeans.

Fase transición: Finalmente, para realizar el despliegue en el ambiente de desarrollo y la integración a del aplicativo con el web *service*, se usó un módulo de seguridad de la solución, el cual consiste en la implementación de una librería llamada AuthorizationFilter, debido al impacto que tiene en el modelo de negocio del proyecto.

- Iteración 4. Integración del aplicativo y el WS a través del módulo de seguridad.

• Se definió el modelo y se implementó en el portal de RITA UD (Red de Investigaciones de Tecnología Avanzada de la Universidad Distrital) la herramienta en la cual se soporta la base de conocimiento, ello de acuerdo con las investigaciones descritas anteriormente y seleccionando la más idónea para el proyecto. Esta solución es OpenKM en su versión libre (Figura 11).

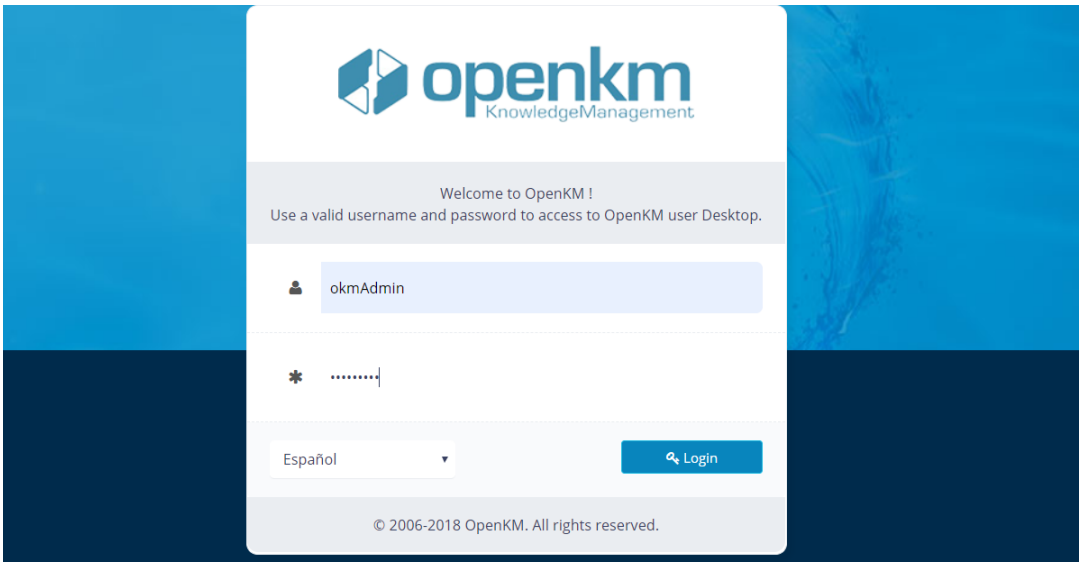

Figura 11. Ventana de inicio de sesión de la herramienta OpenKM que soporta el modelo.

• Se realizó una auditoría de seguridad en servicios web de una entidad o un prototipo organizacional enfatizando en el tema de SQL Injection.

Como resultado final, se ejemplifica el modelo con cada uno de los segmentos de la jerarquía de base de conocimiento (la cual se expondrá a continuación para futuras confrontaciones) para la auditoría en servicios web con el fin evitar los ataques de tipo SQL Injection como vector de ataque del proyecto. Esta servirá para que el usuario tenga una orientación de la taxonomía manejada en este modelo. La taxonomía del modelo se desarrolló extrayendo ideas propias y de varias investigaciones previas, las cuales se irán referenciando a medida que va avanzando el documento. A continuación, se presenta cada uno de los segmentos [6] y [2].

• Con el documento maestro resultado del punto anterior, se alimentó la base de conocimiento orientado a la seguridad informática.

En cada subsección se mostrará al lado un ítem de obligatoriedad u opcionalidad de acuerdo con el modelo que se está proponiendo, por ejemplo: Escaneo de puertos y versiones (obligatorio), análisis SSL (opcional). En caso tal de que el usuario no desee desarrollar los ítems propuestos en el modelo, este modelo no se hará responsable de los resultados finales en cuestión de la base de conocimiento y de la auditoría de los servicios web. Las secciones principales son obligatorias en este modelo, ya que se manejan en las auditorías de servicios de aplicaciones web [16] (Figura 12).

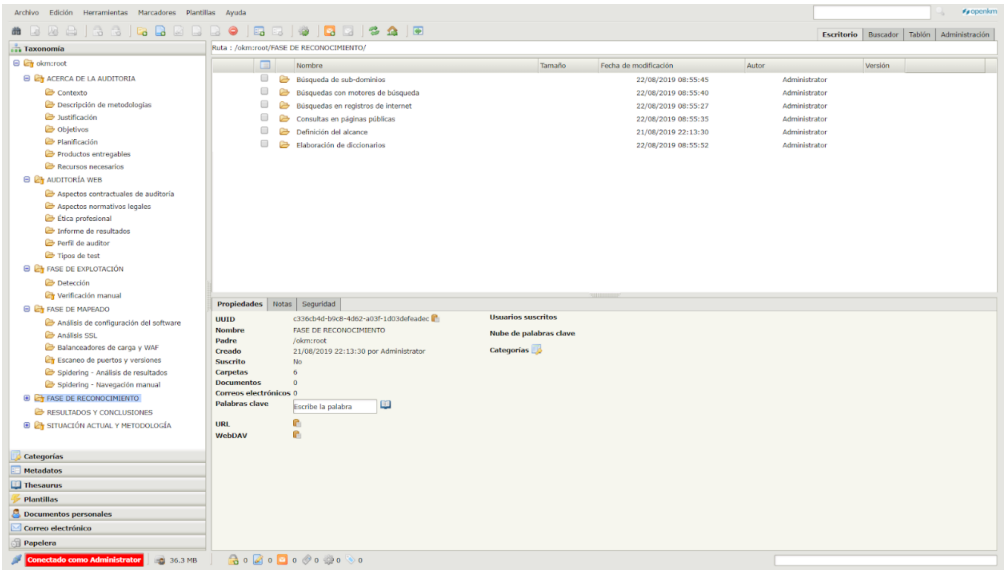

Figura 12. Pantalla inicial de la herramienta OpenKM del modelo propuesto.

La herramienta permite visualizar los archivos en .pdf almacenados en la base, con formato .doc para documentos, .xls para hojas de cálculo y .ppt para presentaciones (Figura 13).

3.1. Jerarquía del modelo base de conocimiento para auditorías de servicios web

Acerca de la auditoría

En esta sección introductoria de vital importancia se realiza un primer acercamiento al negocio de la auditoría de aplicaciones web, ya que se deben conocer los inicios del proyecto que abordará el

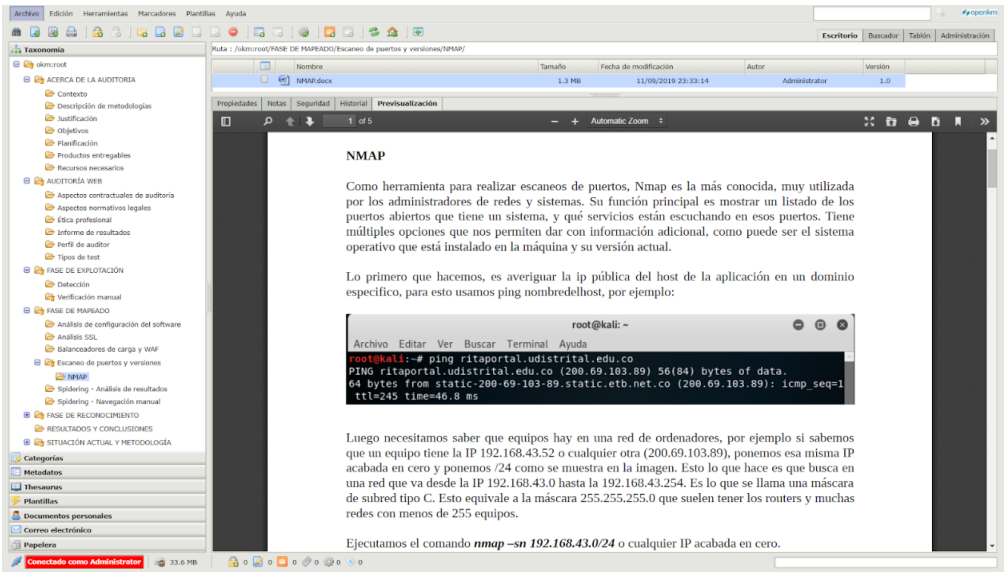

Figura 13. Visualización de archivos del modelo propuesto en la herramienta OpenKM.

usuario. Esta sección se desarrollará de acuerdo con el modelo del negocio definido por el usuario. Se hace con el fin de orientar al usuario y ubicarlo en una primera instancia del modelo.

Contexto (obligatorio)
En esta subsección se realiza una descripción y contextualización del negocio al cual el usuario requiere hacer la auditoría. La subsección se desarrollará con los modelos y diagramas que se obtengan del negocio definido por el usuario, tales como el modelo de arquitectura, diagrama de componentes o diagrama de clases. Esta subsección se hace con el fin de que el usuario halle disponibles sus modelos y diagramas iniciales del negocio objeto de auditoría en la base de conocimiento.

Justificación (obligatorio)
En esta subsección se redactan los motivos de la importancia de realizar la auditoría en un modelo de negocio predefinido por el usuario. Se desarrollará con las necesidades propias del negocio por el cual se quiere hacer la auditoría. Tiene el fin de que el usuario halle siempre disponibles las necesidades y motivos del negocio objeto de auditoría en la base de conocimiento.

Objetivos (obligatorio)
En esta subsección se redactan los objetivos generales y específicos a alcanzar por el usuario en la auditoría. Se desarrollará con los objetivos del usuario en el contexto propio del modelo de negocio. Tiene el fin de que el usuario halle siempre disponible los objetivos generales y específicos del negocio objeto de auditoría en la base de conocimiento.

Productos entregables (obligatorio)
En esta subsección se describen los productos a entregar previstos por el usuario una vez realizada la auditoría. Se desarrollará con los productos previstos propios del modelo de negocio que el usuario tiene que entregar. Tiene el fin de que el usuario halle siempre disponible la lista de productos a entregar de acuerdo con la auditoría realizada.

Recursos necesarios (obligatorio)
En esta subsección se describen los recursos mínimos preliminares y necesarios que debe tener el proyecto y el modelo de negocio. Se desarrollará con investigaciones previas que se han realizado a nivel de infraestructura (*hardware* y *software*) y de las herramientas necesarias en el proyecto. Tiene el fin de que el usuario halle siempre disponible el listado de recursos mínimos necesarios para lograr los objetivos del proyecto en cuestión para realizar la auditoría

Descripción de metodologías (obligatorio)
En esta subsección se describen las metodologías y buenas prácticas utilizadas de acuerdo con el proyecto y al modelo de negocio. Se desarrollará con investigaciones previas que se han realizado sobre las metodologías acordes al proyecto y definidas en el anteproyecto, con aplicación en contexto nacional e internacional. Tiene el fin de que el usuario halle siempre disponible las metodologías definidas para la elaboración del proyecto en cuestión y para realizar la auditoría.

Planificación (obligatorio)
En esta subsección se propone como opcional debido a que, dependiendo de las metodologías implementadas por el proyecto y también la dimensión del modelo de negocio, el concepto de tiempo en la planificación entraría como temporal. Esta subsección se desarrollará con la herramienta Project de Microsoft con el proyecto en cuestión. Tiene el fin de que el usuario halle siempre disponible un cronograma planificado con las iteraciones, fases, actividades y tiempos del proyecto actual para realizar la auditoría.

Actual situación y metodología
En esta sección de referencia del modelo se realiza un estado del arte de la seguridad de las aplicaciones web y de las metodologías utilizadas en la misma, ya que es necesario conocer el punto de partida del proyecto. Se desarrollará de acuerdo con el estado del arte de seguridad de las aplicaciones web, las metodologías y las buenas prácticas actuales en este campo. Tiene el fin de hacer que el usuario esté inmerso en este tema de seguridad de aplicaciones web y para ver cuáles metodologías y buenas prácticas puede aplicar de acuerdo con su modelo de negocio.

Osstmm (obligatorio)
En esta subsección se expone brevemente, el manual de metodologías de la OSSTMM y los casos de pruebas catalogados por el manual en cuestión, ya que es un elemento importante que se requiere en la auditoría y que ayudará al usuario en la práctica. Se desarrollará con los tipos y casos de acuerdo con las últimas versiones propuestas por el OSSTMM. Tiene el fin de que el usuario halle disponibles los conceptos, tipos y casos de prueba de seguridad en el modelo propuesto.

Owasp (opcional)
En esta subsección se exponen los tipos de ataques y las vulnerabilidades más populares actualmente, concernientes a la guía de seguridad de pruebas de OWASP, ya que es un referente mundial importante en el tema de la seguridad de aplicaciones y servicios web. Esta subsección se desarrollará de acuerdo con el último top diez propuesto por el OWASP. Esta subsección se hace con el fin de que el usuario halle disponibles los tipos de ataques y vulnerabilidades de seguridad de acuerdo con el modelo propuesto [16].

MinTIC (opcional)
En esta subsección se exponen los lineamientos, modelos de seguridad y guías metodológicas y de auditorías propuestas actualmente por MinTIC, ya que es el referente nacional en el tema de la seguridad de aplicaciones y servicios web. Se desarrollará de acuerdo con las últimas versiones de los documentos mencionados previamente. Tiene el fin de que el usuario halle disponibles los lineamientos, modelos de seguridad y guías metodológicas y de auditorías de seguridad de acuerdo con el modelo propuesto.

Auditoría WEB
En esta sección se explican los temas más destacados de auditoría web en un ambiente organizacional.

Perfil de auditor (opcional)
Perfil de auditor (opcional) Esta subsección describe el perfil del auditor informático, encargado de aplicar sus conocimientos para lograr unos resultados de calidad y generar así recomendaciones de valor para la auditoría. Se marca como opcional en tanto no es un ítem bloqueante para poder proceder con la auditoría. Esta subsección se desarrollará con la información que se tenga de perfiles de auditores en revisiones en auditorías de información. Tiene el fin de que el usuario halle disponibles las características del perfil de un auditor de seguridad informático de acuerdo con el modelo propuesto.

Tipos de *test* (obligatorio)
En esta subsección se exponen los tipos (interna o externa) y enfoques de la auditoría, ya que puede ser aplicada a diferentes contextos. Se desarrollará con la información de revisiones que se tenga de los diferentes enfoques. Tiene el fin de que el usuario halle disponibles los enfoques de auditoría de acuerdo con el modelo propuesto.

Aspectos contractuales de auditoría (obligatorio)
En esta subsección se exponen los aspectos contractuales de la auditoría, ya que son los que el auditor firma con la organización en caso de ser aplicada. Se desarrollará con la información de revisiones que se tenga de los tipos de permisos, acuerdos, reglas y del alcance al aplicar una auditoría. Tiene el fin de que el usuario halle disponibles los aspectos más comunes de una auditoría.

Aspectos normativos legales (obligatorio)
En esta subsección se exponen las normas y leyes que apliquen en la auditoría, ya que si la organización encuentra vulnerabilidades que afecten información personal (*habeas data*), esta debe blindarse según las leyes expuestas en este apartado. Esta subsección se desarrollará con la información de revisiones que se tenga de las leyes y normas que apliquen en la auditoría. Tiene el fin de que el usuario halle disponibles las leyes y normas, ya sean nacionales o internacionales, al realizar la auditoría.

Ética profesional (obligatorio)
En esta subsección se exponen algunos aspectos éticos del auditor de vital importancia, ya que la reputación y la confianza son temas que se gana el auditor en el desarrollo y resultados de su trabajo. Esta subsección se desarrollará con la información de revisiones que se tenga de asuntos

destacables en el tema de la ética profesional en la auditoría. Tiene el fin de que el usuario halle disponibles los acuerdos de confidencialidad y de imparcialidad que se tenga al aplicar la auditoría.

Informe de resultados (obligatorio)
En esta subsección se expone la estructura y la forma como se elabora el informe de resultados final, ya que este documento reúne todo el trabajo realizado en la auditoría. La subsección se desarrollará con la información de revisiones que se tenga de estructuras y elaboraciones de informes que se tenga en auditorías. Tiene el fin de que el usuario halle disponible una estructura general del informe de resultados de la auditoría.

Fase de reconocimiento
En esta sección se hace la recolección de toda la información que sea posible acerca del objetivo en el marco organizacional establecido o modelo de información definido; información como direcciones IP, nombres de máquinas, infraestructura de la red, perfiles y configuración de servidores, *software*, entre otros [6].

Definición del alcance (obligatorio)
En esta subsección se define con el cliente el alcance de la auditoría, ya que se debe saber con exactitud lo que se va a hacer, qué tanto se va a hacer y hasta dónde se va a hacer. Esta subsección se desarrollará con la información de los requerimientos funcionales y no funcionales del modelo de negocio. Se hace con el fin de que el usuario tenga disponible el alcance de la auditoría y pueda tomar decisiones concernientes al tipo de prueba que se aplicará (caja negra, blanca o gris).

Búsquedas en registros de internet (obligatorio)
En esta subsección se definen y se aplican herramientas para búsqueda específica en la web, ya que se deben identificar datos particulares de la infraestructura básica del modelo de negocio objetivo de testeo. La subsección se desarrollará con datos clave como el tipo de servidor de la máquina, bajo qué sistema operativo funciona, qué servicios tiene, qué puertos están abiertos, cuál es su configuración, con qué otras máquinas se conectan, entre otros. Se tiene el fin de que el usuario tenga los datos generados con las herramientas disponibles.

Consultas en páginas públicas (opcional)
Esta subsección muestra información asociada a la organización a la cual le se le está realizando la auditoría; sin embargo, se marca opcional, ya que si bien pueden ser útiles los datos que se encuentran en la web pública, no es un ítem bloqueante para poder proceder con la auditoría. La subsección se desarrollará con páginas, grupos, listas y motores de búsqueda generales públicos en la web. Se tiene el fin de que el usuario halle disponibles las fuentes públicas de las cuales se extrajo información extra.

Búsquedas con motores de búsqueda (opcional)
Esta subsección muestra información relacionada con motores de búsqueda concretos; sin embargo, se marca opcional, ya que si bien pueden ser útiles los datos que se encuentren con el motor, no es un ítem bloqueante para poder proceder con la auditoría. La subsección se desarrollará con información del fichero "robots.txt" y datos de motores de búsqueda concretos. Se tiene el fin de que el usuario halle disponibles las fuentes concretas de las cuales se extrajo información extra.

Búsqueda de subdominios (obligatorio)
Esta subsección muestra información relacionada con la forma de encontrar nuevas máquinas objetivo, ya que se puede comprobar si el dominio que se audita tiene subdominios asociados. La subsección se desarrollará con información que se extraiga de la herramienta Fierce de Kali Linux. Se tiene el fin de que el usuario halle disponibles los datos extraídos de la herramienta de penetración Fierce.

Elaboración de diccionarios (obligatorio)
Esta subsección muestra información relacionada con la elaboración de diccionarios, ya que es otra técnica de *hacking* que puede servir en la auditoría. Esta subsección se desarrollará con información que se extraiga de la herramienta *CeWL* de Kali Linux. Se hace con el fin de que el usuario tenga disponibles los datos extraídos de la herramienta generadora de listas de palabras *CeWL.*

Fase de mapeado
En esta sección se hace la recolección de toda la información que sea posible acerca aplicación web objetivo, el servicio web expuesto y el servidor web que los aloja. Para ello, se emplearán distintas herramientas que generarán datos específicos como puertos, uso de protocolos y seguridad explícita de la aplicación [17].

Escaneo de puertos y versiones (obligatorio)
Esta subsección muestra información relacionada con el escaneado de puertos, ya que en el tema de mapeado es una tarea inevitable para la auditoría. La subsección se desarrollará con información que se extraiga de la herramienta de escaneado de puertos TCP y UDP *Nmap*. Se tiene el fin de que el usuario tenga disponible la información de los puertos extraídos con herramienta.

Análisis SSL (obligatorio)
Esta subsección muestra información relacionada con el análisis del protocolo seguro SSL, ya que este garantiza la confidencialidad de las comunicaciones vía web para la aplicación objeto de auditoría. Esta subsección se desarrollará con información que se extraiga de las herramientas TLSSled y SSLDigger, las cuales analizan la seguridad de las implementaciones en los protocolos SSL/TLS de un servidor objetivo. Se tiene el fin de que el usuario tenga disponible la información del análisis SSL realizado con dichas herramientas.

Balanceadores de carga y WAF (obligatorio)
Esta subsección muestra información relacionada con los balanceadores de carga y con cortafuegos de aplicaciones web (WAF), ya que con estos datos se sabrá si la aplicación objeto de auditoría se encuentra alojada en varios servidores o solo en uno, también si está protegida detrás de un cortafuegos. La subsección se desarrollará con información que se extraiga de las herramientas Wafw00f y Halberd, las cuales analizan el balanceo de carga y protección de la aplicación web. Se tiene el fin de que el usuario halle disponible la información sobre balanceadores de carga y *firewall* de aplicaciones web.

Análisis de configuración del *software* (obligatorio)
Esta subsección muestra información relacionada con la configuración de la aplicación objeto de auditoría, ya que con estos datos se entenderá cómo está construido el sistema a nivel de software.

La subsección se desarrollará con información que se extraiga de la herramienta Nikto, la cual sirve para buscar estas vulnerabilidades en aplicaciones web. Se hace con el fin de que el usuario tenga disponible la información sobre dichas vulnerabilidades, si las hay.

***Spidering*. Navegación manual (obligatorio)**
Esta subsección muestra información relacionada con la navegación manual en profundidad de la aplicación (también llamado rastreo de la aplicación), ya que con estos datos es posible conocer a fondo el sitio web e identificar todas las funciones que implemente. La subsección se desarrollará con información que se extraiga de la herramienta ZAP, la cual sirve para identificar objetivos, recursos y parámetros que luego se escanearon con el *fuzzer* contenido en la herramienta. Se hace con el fin de que el usuario tenga disponible la información con base en los datos extraídos con ZAP.

***Spidering*. Análisis de resultados (obligatorio)**
Esta subsección muestra información relacionada con el análisis de los resultados una vez realizado el recorrido manual de todas las páginas, ya que el objetivo es encontrar archivos en el servidor web. La subsección se desarrollará con información que se extraiga con la función "Spider" de la herramienta ZAP. Se hace con el fin de que el usuario tenga disponible la información con base en los datos encontrados con la función "Spider" de ZAP.

Fase de explotación
En esta sección se desarrollarán dos tareas principales, las cuales son detección y verificación. Cada una de estas tareas estará compuesta por hallazgos concretos que se logren encontrar en esta auditoría. Cada una de las demás auditorías será distinta, sobre todo en esta fase, ya que los hallazgos serán distintos cada vez, así que las pruebas que habrá que realizar cambiarán. También cambiarán en función de otros factores como la aplicación a auditar, o las tecnologías que use el prototipo. Para este caso específicamente son JSF, JPA, EJB, Primefaces, Glassfish, entre otras [7].

Detección (obligatorio)
Esta subsección muestra información relacionada con la búsqueda y detección de vulnerabilidades a partir de los resultados obtenidos durante la fase de mapeado, ya que se debe probar cómo se comporta la aplicación ante errores introducidos a propósito. La subsección se desarrollará con información que se arroje del ejercicio del comportamiento de la aplicación ante errores, luego se escanean los recursos o métodos que envían parámetros al servidor objetivo y, por último, se emplearán ataques de *fuzzing* (técnica de pruebas en *software*) sobre los parámetros encontrados en los puntos anteriores. Se hace con el fin de que el usuario tenga disponible la información obtenida de toda la tarea de detección.

Verificación SQL Injection (obligatorio)
Esta subsección muestra información relacionada con la verificación manual de los descubrimientos previos, ya que se debe analizar cada hallazgo sospechoso que aún no haya sido confirmado. La subsección se desarrollará con información específica del segmento de SQL Injection en el modelo propuesto con base en la revisión que se hizo en dicho apartado. Se hace con el fin de que el usuario tenga disponible la información obtenida de todo el ejercicio de verificación de vulnerabilidades usando el vector de ataque SQL Injection.

Resultados y conclusiones
En esta sección se muestran los resultados finales de la auditoría y unas conclusiones del trabajo realizado. Además, se compara con los objetivos propuestos al inicio del proceso para comprobar si efectivamente se cumplieron.

4. Conclusiones

De acuerdo con el objetivo general y los objetivos específicos, se relacionarán a continuación cada uno de estos con la conclusión de cada punto, demostrando si las actividades realizadas cumplen con el planteamiento inicial de este proyecto.

• "Implementar un modelo base de conocimiento para la auditoría de seguridad en servicios web". Con este proyecto se logró crear un modelo de base de conocimiento implementado sobre una herramienta libre, ejecutando una auditoría en seguridad de servicios web con SQL Injection sobre un prototipo organizacional.

• "Implementar metodologías y buenas prácticas en el tema de seguridad de la información". Se logró establecer una metodología propia para el desarrollo del modelo propuesto, incluyendo metodologías ya definidas, métodos, buenas prácticas, técnicas, recomendaciones y tendencias actuales relacionadas con conceptos de base de conocimiento, auditoría de aplicaciones web, seguridad de la información y desarrollo de *software*.

• "Establecer una metodología propia para el modelo, con metodologías definidas, métodos y buenas prácticas". En contexto con el anterior objetivo, se hicieron revisiones literarias sobre investigaciones en el tema y se estudiaron guías, técnicas y herramientas para llevar a cabo la auditoría en servicios web, determinando que para el vector de ataque SQL Injection, en el contexto establecido, la prevención no es responsabilidad del JSF. La forma de evitar el ataque depende del API de persistencia que se esté utilizando (JPA, para el caso), pero todo se reduce a que nunca se debe concatenar la entrada controlada por el usuario en cadenas SQL, siempre se debe usar consultas parametrizadas cuando corresponda.

• "Alimentar un *software* de base de conocimiento con datos resultantes de pruebas de Inyección SQL". Se realizó la instalación, configuración y alimentación del software OpenKM, como herramienta que soporta el modelo propuesto, con los datos y la información resultante del proceso de auditoría de los servicios web en un prototipo definido.

Finalmente, con la socialización de este modelo, se aclara que queda expuesto de manera libre para que sea usado e implementado por cualquier organización, entidad o personas y se pueda trabajar en proyectos futuros si así se quiere.

Referencias

[1] A. Au y W. Fung, "Knowledge Audit Model for Information Security", en *8th International Conference on Innovation and Knowledge Management in Asia Pacific*, Kobe, octubre 2016. ↑265

[2] J. L. Contreras, "Propuesta de auditoría a las aplicaciones web de la empresa C&M consultores aplicando herramientas de software libre", trabajo de grado, Universidad Nacional Abierta y a Distancia, 2017. ↑265, 274
[3] S. Coronado, "Desarrollo de una guía metodológica basada en análisis SQL injection y formas de protección a las bases de datos", trabajo de grado, Pontificia Universidad Católica del Ecuador, Quito, 2017. ↑265
[4] A. Sadeghian, M. Zamani y A. A. Manaf. "A Taxonomy of SQL Injection Detection and Prevention Techniques", en *Informatics and Creative Multimedia (ICICM)*, Kuala Lumpur, septiembre 2013. https://doi.org/10.1109/ICICM.2013.18 ↑266
[5] A. Sadeghian, M. Zamani, S. M. Abdullah, " A taxonomy of SQL Injection Attacks", en *International Conference on Informatics and Creative Multimedia*, Kuala Lumpur, septiembre 2013. ↑266
[6] M. Rodríguez, "Auditoría de aplicaciones web: metodología y práctica profesional". [En línea]. Disponible en: http://openaccess.uoc.edu/webapps/o2/bitstream/10609/40153/7/mrodriguezsanchez1TFC0115memoria.pdf ↑266, 274, 278
[7] G. Méndez y L. Álvarez, "Metodología para la construcción de la base de conocimiento de un sistema experto", *Revista Ingeniería*, vol. 8, n.○ 2, pp. 12-18, 2003. https://doi.org/10.14483/23448393.2686 ↑267, 280
[8] E. A. Varela, D. Estrada y L. Acosta. "Wiki, herramienta informática para la base de conocimiento para el proyecto PROMEINFO de la Universidad de Guayaquil", *Dominio de las Ciencias Sociales*, vol. 3, n.○ 3, pp. 702-727, 2017. http://dx.doi.org/10.23857/dc.v3i3.502 ↑267
[9] Ministerio de las TIC [MinTIC]. "Modelo de seguridad y privacidad de la Información y Guía de auditoría de Seguridad y privacidad de la información", [En línea]. Disponible en: https://www.mintic.gov.co/gestionti/615/articles-5482_Modelo_de_Seguridad_Privacidad.pdf ↑267
[10] D. Guaman, F. Guaman, D. Jaramillo y M. Sucunuta, "Implementation of techniques and OWASP security recommendations to avoid SQL and XSS attacks using J2EE and WS-security", en *12th Iberian Conference on Information Systems and Technologies (CISTI)*, Lisbon, junio 2017. ↑267
[11] ISO/IEC 27000, "Information technology - Security techniques - Information security management systems - Overview and vocabulary". [En línea]. Disponible en: https://www.iso.org/obp/ui/#iso:std:iso-iec:27000:ed-5:v1:en ↑267
[12] J. Singh, "Analysis of SQL Injection Detection Techniques", *Theoretical and Applied Informatics*, vol. 28, n.○ 1-2, 2016. ↑268
[13] IBM Corporation. "X-Force IRIS Data Breach Report". architecture". [En línea]. Disponible en: https://www.ibm.com/security/resources/xforce/xfisi/,2019. ↑268
[14] OpenKM, "Diagram of the system architecture". [En línea]. Disponible en: https://www.openkm.com/en/architecture.html ↑269
[15] M. Shaw y D. Garlan, "Software Architecture: Perspectives on an emerging discipline". Upper Saddle River: Prentice Hall, 1996. ↑271
[16] T. Gunawan, M. Lim, M. Kartiwi, N. Malik y N. Ismail, "Penetration testing using Kali linux: SQL injection, XSS, wordpress, and WPA2 attacks", *Gunawan*, vol. 12, n.○ 2. http://doi.org/10.11591/ijeecs.v12.i2.pp729-737 ↑274, 276
[17] X. Liu, Q. Yu, X. Zhou y Q. Zhou, "OwlEye: An Advanced Detection System of Web Attacks Based on HMM", *IEEE 16th Intl Conf on Dependable, Autonomic and Secure Computing*, Atenas, agosto 2018. ↑279

John Edison Moreno

Recibió su grado en Ingeniería Telemática de la Universidad Distrital, Bogotá, Colombia, en 2012, su grado de especialista en Proyectos de Información de la Universidad Distrital, Bogotá, Colombia, en 2014 y el M.Sc. en Ciencias de la Información y las Comunicaciones de la Universidad Distrital, Bogotá, Colombia, en 2020. Actualmente trabaja como gestor de proyectos TI en la coordinación de desarrollo de sistemas de información en la DIAN – Dirección de impuestos y aduanas nacionales, donde se desempeña como ingeniero DevOps y Full-Stack. Sus principales intereses de investigación incluyen disciplinas de Ingeniería TI como Cloud, Integración Continua y Entrega Continua y Automatización de procesos y flujos de trabajo en el desarrollo de código fuente y las operaciones TI involucradas en éste. e-mail: jemorenom@correo.udistrital.edu.co

Paulo Cesar Coronado

Recibió su grado en Ingeniería Electrónica de la Universidad Distrital, Bogotá, Colombia, en 2004, su grado M.Sc. en Ciencias de la Información y las Comunicaciones de la Universidad Distrital, Bogotá, Colombia, en 2007 y actualmente cursando el Doctorado Interinstitucional en Educación de la Universidad Distrital, Bogotá, Colombia. Actualmente trabaja como profesor en asignaturas de Ingeniería y de Maestría en la Universidad Distrital. Sus principales líneas de interés incluyen: Sistemas de Información Geográfica, Aprendizaje de Máquina, Ambientes inmersivos, Telesalud, Tecnología educativa, Software Libre y de código abierto. Actualmente desarrollando: Proyecto de doctorado para explorar el uso de la realidad aumentada y mixta en procesos de enseñanza de programación de computadores. OpenSITEM Plus: Software de telesalud. También participando en: Modelo de Seguridad Informática con herramientas de código abierto, Aprendizaje de máquina para la identificación de vulnerabilidades en procesos de enseñanza y Metamodelo de arquitectura empresarial para entidades públicas.
e-mail: paulo_cesar@udistrital.edu.co