CIDI
Centro de Ideoinformática

Universidad de las Ciencias Informáticas

Facultad 1

Trabajo de diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Título:

Sistema de Auditoría de las Políticas de Seguridad Informática del Centro de Ideoinformática

Autor:

Dariem Lázaro García López

Tutores:

Msc. Sahilyn Delgado Pimentel

Ing. Yojahny Chávez Marrero

La Habana, Junio 2019

"Año 61 de la Revolución"

Agradecimientos

Deseo agradecer a esa gran familia conformada por mi madre, que fue padre y madre a la vez, que me complació todos mis caprichos y me inculcó los valores y principios educando con el ejemplo, por todo su amor y consagración. A mi hermano que siempre estuvo pendiente de mis necesidades y me ayudó en el día a día. A mis familiares por compartir su amor y paciencia y por transmitir sus experiencias. A mis amistades por estar presentes en las buenas y en las malas y brindar su ayuda, a Yuniel, Denis y Alex Alain, por enseñarme a programar en Symfony con lo cual pude completar la aplicación, a Fernando por enseñarme a desarrollar videojuegos, lógica de programación, hacer sugerencias y aclararme dudas, a Ignacio por enseñarme a diseñar videos, editar imágenes y desarrollo web. A mis tutores por siempre estar ahí cundo les necesitaba, por sus críticas y consejos que garantizaron la calidad de este trabajo, Sahilyn y Yojahny. A mis profesores por enseñarme con paciencia y dedicación Yaima, Maxora, Aneyty, Dailín, Dargel, Ernesto Soto, Ponce, Yurisbel, Juan Manuel, Dysnayle, Miguel Jaeguer, Mónica, Nery Karen, Nestor, Yariek, Yulio, Hubert, Javier Heredia, Noichel, Anelys, Angela María, Graciela, María Cristina, Jorge Aurelio, Israel Eddy, Julio, Victor, Orlenis, Yusniel, A todos, muchas gracias por ayudarme a recorrer este camino. A mis compañeros de aula, por su apoyo incondicional, por su amistad, por compartir sus momentos, Ignacio Mariana, Daniela, Lissandra, Luis Ernesto, Carel, Miosotis, Ilein, Liana, Mayelin, Mayra, Shadet, Susana, Vladimir, Reimer, Pedro Javier, Adrián, Alfred, Andy, Leonardo, Massiel, Ana Carla, Javier Gómez, Javier Fuentes. A mis compañeros de edificio por su ayuda en los estudios y su compañía, Chirino, Bryan, Daniela Rivera, Darlin, Patricia, Ilianys, Grisel, Yunier, Yudenis,

Yanisbel, Roynel, Rafael, Rachel, Javier Leyva, A mi Team de Linuxeros, Pedro Iván, José Carlos y Guillermo por acompañarme en cada evento científico, cultural o deportivo, por enseñarme CSS, JavaScript, Jquery, Bases de datos, Java, por ayudarme con el documento y la aplicación, por todos los buenos momentos durante estos 5 años. A Nauria Méndez Trujillo por todo el tiempo que pasamos juntos y que nunca olvidaremos, por todo su amor y sencillez. Ella se gradúa junto a todos nosotros.

A mis decanos y vicedecanos por su apoyo en todas las esferas dentro y fuera de la universidad, Juana, Reina Serguey, Javier, Delly, Niurvis, Silvano, Aylin, Mayleidis, Geidis y Miguel Angel. A Miriam Nicado por su sencilles y todo su apoyo en situaciones adversas.

A los trabajadores del CIDI por sus sugerencias, apoyo, aporte de información y ayuda en la programación de la aplicación, Sahilyn, Alexander, Andy, Claudio, Gustavo, Ismary, José Gabriel, Leiny, Martha, Miguel Angel, Yunior, Yordanka, Waldo, Ramón, Victor, Sonia, Pedro, Yojanhy, Karla, Geidy, Erick, Eduardo,

A los trabajadores de CESOL por su ayuda en los temas de seguridad de GNU/LINUX, Adackny, Alejandro, Carlos, Gustavo Quezada, Yosel, Lexys.

A mis profesores de artes plásticas Malcolm, Maray y Yenisley por sus sugerencias y apoyo al desarrollo de mis representaciones artísticas e ideales sociales. A mis profesoras de literatura por los buenos momentos en los talleres Martha y Rosa. Al profesor Armando y todo el equipo de Marabana en la UCI por su convocatoria y esfuerzo en la preparación en cada evento del proyecto Marabana-Maracuba.

A Grisel, Alfred, Andy, Adriana, Amanda Ojeda, Amanda María, Amanda Delgado, Arisneisi, Betsy, Beatriz, Darlin, Ilianis, Enma, Rachel Marichal, Rachel Martínez, Rachel Sánchez, Marylaura, Maidelis, Yudenis, Yanisbel, Stephanie, Alejandro Chirino, Danilo, Daniel, Hugo, Frank, Juan Ramón, Jesús, Javier Villalobos, Juan Pablo, Julio Cesar, Magdiel, Lester Alejandro, Lester Díaz, Leonardo Jimenez, Neiser, Michel Pedrera Maykol, Raynol, Yossan, Yariel, Reynier, Renier, Yandry, Yunier, Susana, Rocío, Alejandro Cano, Alejandro Avelino, Alexis Edgardo, Danny, Claudia, Ariel de Jesús, Ariel Alarcón, Dismaray, Dibániel, Daymel, Daylin, David Heberto, David Gustavo, Gabriel, Flora, Felix Manuel, Jesús Miguel, Javier Espinosa, Iván Javier, Iván Antonio, Ignacio, Luis Enrique, Lissette, Osvaldo, Hector Daniel, Sulema, Yaiselis, Arlenis, Arlene, Cristy, Pedro Iván, José Carlos y Guillermo, por contar con su apoyo y los momentos pasados en los variados eventos científicos, culturales y deportivos de la universidad.

A mis enemigos por enseñarme que siempre se puede ser mejor persona sin importar las circunstancias y que en medio del caos salen nuestras mejores habilidades y virtudes.

Y finalmente un agradecimiento especial a la persona culpable de este sueño cumplido. Alguien que me recibió al llegar por primera vez a la UCI, que me guió durante este largo camino, que me empujó a seguir adelante cuando estuve a punto de abandonar la carrera. Fue mi amiga , consejera, pedagoga, psicóloga, entrenadora, tutora y más que todo una madre para mi, Rosa de la Caridad.

A esta gran familia, muchas gracias por todos los grandes momentos.

Dedicatoria

Dedico este trabajo a mis familiares por su sacrificio y dedicación, a mis profesores por su paciencia y empeño y a todos los que de una forma u otra contribuyeron a este sueño cumplido.

Lo dedico también a la memoria de una estudiante que pasó grandes momentos con nosotros, gran investigadora, deportista, artista y mi eterna compañera. Nos abandonó prematuramente, pero la recordaremos por siempre con su sonrisa constante. Nunca olvidaré su despedida en mi pecho ese 22 de febrero de 2019, a las 9:15 de la mañana.

Gracias por los momentos que pasamos juntos Nauria Méndez Trujillo.

Declaración de autoría

Declaro por este medio que yo Dariem Lázaro García López, con carné de identidad 95071730447 soy el autor principal del trabajo titulado “Sistema de Auditoría de las Políticas de Seguridad Informática del Centro de Ideoinformática” y autorizo a la Universidad de las Ciencias Informáticas a hacer uso de la misma en su beneficio, así como los derechos patrimoniales con carácter exclusivo.

Para que así conste firmamos la presente a los __ días del mes de Junio de 2019.

Autor

_____

Dariem Lázaro García López

Tutor

_____

Msc. Sahilyn Delgado Pimentel

Tutor

_____

Ing. Yojahny Chávez Marrero

Resumen

La Universidad de las Ciencias Informáticas posee una amplia infraestructura tecnológica imprescindible para el desarrollo de productos de software para Cuba y el mundo, así como otras actividades en las ramas de las Tecnologías de la Información (TIC). Estas actividades se encuentran expuestas a vulnerabilidades de seguridad como virus, *malwares,* acceso no autorizado y *hackers*. La presente investigación tiene como objetivo el desarrollo de un sistema de auditoría de la seguridad informática que permita disminuir el tiempo y esfuerzo asociado al proceso de auditoría de las políticas de seguridad informática (PSI) en el Centro de Ideoinformática (CIDI). En este centro se producen aplicaciones web y se realiza análisis de información sensible e importante, por lo cual se deben proteger para garantizar el cumplimiento de sus funciones. Los asesores del centro realizan la auditoría de forma semiautomática, máquina a máquina, resultando en un proceso lento y engorroso para la emisión de reportes, además, en caso de ausencia de los mismos, el centro quedaría expuesto a riesgos. Para el desarrollo de la aplicación se utilizó la metodología AUP-UCI, el *framework* de desarrollo web Symfony, entre otras tecnologías. El sistema permite realizar escaneo de IP y auditar las estaciones de trabajo generando los reportes en formato HTML, los cuales podrán ser exportados a PDF para su impresión. Por tanto disminuye el tiempo y esfuerzo necesario asociado al proceso de auditoría de las PSI y automatiza el proceso al 100%, asegurando la protección de los medios tecnológicos, productos de *software* e información del CIDI.

Palabras clave: auditoría, automática, seguridad, sistema, Symfony.

Índice General

Introducción

A partir de los años 40 la auditoría interna se encuentra como labor incipiente y primitiva que dependía del contador y que daba satisfacción a la necesidad de conciliar partidas, analizar cuentas y corroborar documentos de registro. La auditoría interna fue evolucionando de tal forma que ya no dependió del contador sino del contralor y ya no sólo analizaba los documentos fuente sino que ya recurría a reportes tales como sumarios, además su alcance se amplió y analizaba también los procedimientos de generación o elaboración de dichos reportes y realizaba pruebas parciales. A finales de la década de los 60 se empieza a reconocer la necesidad de auditar los sistemas de información y es cuando nace la auditoría informática. En el año 1968 en Estados Unidos, se reconoce la necesidad de elaborar y suministrar documentación a los auditores con objetivo de que los guie en cuanto a la auditoría y control de los sistemas computarizados. Así, el *Institute of Internal Auditors*, editó el manual titulado *Internal Auditing of EDP Systems*. El *Bank Administratión Institute* editó el *Auditing Bank EDP Systems.* Asimismo, el *American Institute of Certified Public Accountants* editó el manual *Auditing and EDP* (Morales 2000).

La seguridad informática ha tomado gran auge, debido a las cambiantes condiciones y nuevas tecnologías disponibles. La posibilidad de interconectarse a través de redes e INTERNET, ha abierto nuevos horizontes a las empresas para mejorar su productividad, lo cual ha traído consigo, la aparición de amenazas para los sistemas de información. Estos riesgos condicionaron que muchas desarrollen documentos y directrices que orientan en el uso adecuado de estas tecnologías. En este sentido, las políticas de seguridad informática (PSI) surgen como una herramienta organizacional para concientizar sobre la importancia y sensibilidad de la información y servicios críticos de las empresas.

Un asesor de seguridad informática es el encargado de asesorar y supervisar las medidas de seguridad necesarias para proteger de manera efectiva los bienes de una empresa o de un cliente, en tal sentido, hacen uso de su conocimiento y experticia para evaluar las potenciales amenazas de seguridad y violaciones para prevenirlas y desarrollar protocolos y planes de contingencia en caso de cualquier incidencia. Para verificar que las PSI se cumplen, los asesores de seguridad informática, realizan auditorías periódicas a sus sistemas de cómputo, para detectar y eliminar o reducir las vulnerabilidades

manteniendo la integridad de sus empresas y su infraestructura tecnológica y de la información. La auditoría de la seguridad en la informática abarca los conceptos de seguridad física y lógica. El término seguridad física se refiere a la protección del *hardware* y los soportes de datos, así como la seguridad de los edificios e instalaciones que los albergan. Por su parte, la seguridad lógica se refiere a la seguridad en el uso de *softwares*, la protección de los datos, procesos y programas, así como la del acceso ordenado y autorizado de los usuarios a la información (Romero y Coite 2018).

En Cuba los estudios en cuanto auditoría informática se iniciaron en el año 1996 con el Seminario de Auditoría Informática celebrado en la Feria “Informática 96” (MINCOM 2016). Le corresponde al Ministerio del Interior, según se dispone en el decreto ley No. 199 de 1999, en su artículo 45, el realizar la auditoría a la seguridad informática y es el facultado en autorizar a otros órganos, organismos o entidades u otras personas naturales o jurídicas a realizarlas según dispuso en este decreto nuestro comandante en jefe (Castro 1999). Los estudios y técnicas de auditoría se desarrollaron y expandieron a todas las instituciones y empresas tanto para la auditoría informática como para la auditoría asistida por computadoras.

La auditoría informática es uno de los procesos que se controlan en el Centro de Ideoinformática (CIDI) de la Universidad de las Ciencias Informáticas (UCI). Este proceso está organizado en el centro con una estructura jerárquica encabezada por la directora del centro como máxima responsable, los jefes de cada departamento, el asesor de seguridad informática y los especialistas y trabajadores del centro. Las PSI establecen los requerimientos para garantizar la seguridad de la información en el CIDI. Estas se rigen por la resolución 127/07 del Ministerio de la Informática y las Comunicaciones, además de las políticas y “Código de Ética” de la UCI. Estas políticas están presentes en el Plan de Seguridad Informática del centro y son de obligatorio cumplimiento por todos los estudiantes y trabajadores vinculados al CIDI. Para el control de estas políticas el asesor de seguridad informática realiza chequeos diarios a una muestra de las estaciones de trabajo, además de auditorías a estas políticas cada tres meses. Para este control el asesor debe ejecutar un *script* en cada estación de trabajo y analizar los resultados de forma manual para luego generar un informe que se presenta al consejo de dirección. El tiempo y el esfuerzo requerido para este proceso imposibilita realizar la auditoría de las políticas de forma sistemática y con la calidad

requerida, provocando que las políticas puedan ser vulneradas y no sea detectada la infracción. El continuo traslado de una máquina a otra ejecutando el *script* y almacenado los resultados en una memoria USB resulta muy rústico y lento para las necesidades actuales, además del peligro de rotura o pérdida de la USB, que pueden causar pérdida de los datos en el primer caso y exposición ante un agente externo en el segundo. Otra problemática es que cuando el asesor se encuentre ausente las políticas no puedan ser auditadas y alertar a los responsables de cualquier hecho. A pesar de que existe en la UCI el Sistema XILEMA GRHS 2.0, este necesita ser instalado en cada PC y está orientado al inventario de los equipos, presenta un módulo que brinda información sobre el estado técnico de la PC pero no realiza auditorías.

Por lo anteriormente planteado se define como **problema de investigación**: ¿Cómo disminuir el tiempo y esfuerzo asociado al proceso de auditoría de las políticas de seguridad informáticas en el Centro de Ideoinformática?

Se enmarca el **objeto de estudio** en el proceso de auditoría a través de una aplicación web.

Se plantea como **objetivo general**, desarrollar un sistema que permita la auditoría en el Centro de Ideoinformática, para disminuir el tiempo y esfuerzo necesario asociado al proceso de auditoría de las políticas de seguridad informáticas.

Los **objetivos específicos** son los siguientes:

1. Definir los referentes teóricos fundamentales que sustentan la investigación relacionados con el desarrollo de sistemas para la auditoría de PSI.

2. Analizar el estado de la auditoría de PSI a nivel mundial.

3. Identificar las funcionalidades del Sistema de Auditoría de las PSI del CIDI.

4. Implementar las funcionalidades del Sistema de Auditoría de las PSI del CIDI.

5. Validar las funcionalidades del Sistema de Auditoría de las PSI del CIDI.

Delimitando así el **campo de acción:** al proceso de auditoría a través de una aplicación web de las políticas de seguridad informáticas en el Centro de Ideoinformática.

Después de haber tratado los elementos fundamentales del área de la ciencia a incidir y los objetivos primordiales, se formula la siguiente **idea a defender:**

El sistema que permitirá la auditoría, disminuirá el tiempo y esfuerzo asociado al proceso de auditoría de las políticas de seguridad informáticas en el Centro de Ideoinformática, a través de una aplicación web.

Para dar cumplimiento a los objetivos específicos se definieron las siguientes **tareas de investigación:**

1. Realización de un estudio sobre las tendencias en la auditoría automática de las PSI a nivel nacional e internacional.

2. Selección de las tecnologías, herramientas y estándares que se necesitan para implementar la propuesta de solución.

3. Selección de la metodología de desarrollo.

4. Definición de los requisitos funcionales y no funcionales de la propuesta de solución.

5. Implementación de la propuesta de solución.

6. Documentación de las pruebas de funcionalidad, seguridad, carga y estrés, aceptación e integración.

Los métodos que se utilizan en la presente investigación son:

Métodos teóricos:

- **histórico-lógico**, se utiliza para realizar un estudio sobre la existencia o desarrollo de proyectos informáticos de este tipo (auditorías automáticas) en el proceso de verificación de las PSI.

- **inductivo,** se utiliza para definir de las generalidades en cuanto a auditoría informática, las necesidades específicas de la propuesta de solución.

- **modelación,** es utilizado en la representación, mediante el uso de diagramas, de las características del sistema, y relaciones entre objetos que intervienen en los procesos implementados por la propuesta de solución.

- **analítico-sintético**, para seleccionar los elementos más importantes relacionados con la auditoría automática de las PSI.

Métodos empíricos:

- **observación** para entender como se realiza el proceso de auditoría y sus diferentes fases.

- **encuesta** para aclarar las actividades de auditoría que realizan los asesores de seguridad informática y definir los requisitos funcionales de la aplicación.

El documento consta de una introducción, tres capítulos, conclusiones, recomendaciones, referencias bibliográficas y anexos que ampliarán la información que se aporta en la investigación.

Capítulo 1: “Herramientas para la auditoría de las PSI, funcionalidades y tecnologías para su desarrollo”. Contiene los fundamentos teóricos que giran sobre los sistemas para la auditoría de las PSI. En el mismo se hace un estudio de sistemas homólogos analizando el objetivo de su uso, su funcionamiento y tecnologías que utilizan.

Capítulo 2: “Caracterización del sistema de auditoría de las PSI del CIDI”. En este capítulo se explica cómo se desarrolla el flujo actual de los procesos, y se describe la propuesta de solución para resolver el problema planteado. Por otra parte, se especifican los requisitos funcionales y no funcionales, y los elementos fundamentales del diseño y de la arquitectura.

Capítulo 3: ‘’Construcción y pruebas del sistema de auditoría de las PSI del CIDI’’. En este capítulo se incluye la programación realizada a partir de los requisitos, así como las pruebas realizadas para su validación. Además, se establecen los estándares de codificación que se tuvieron en cuenta para el desarrollo del sistema.

Capítulo 1: Sistemas para la auditoría de las PSI, funcionalidades y tecnologías para su desarrollo

En el presente capítulo se define el marco conceptual que sustenta la investigación estrechamente vinculada a los sistemas de auditoría de las PSI. Se exponen los principales conceptos relacionados con el estudio del estado del arte, de las herramientas que permiten la auditoría de las PSI. A partir de los resultados se analizan diferentes lenguajes y herramientas que sean factibles para desarrollar una solución a los inconvenientes planteados, así como la metodología para guiar el proceso de desarrollo de software.

1.1 Política de seguridad informática

Las investigaciones de (Romero y Coite 2018) definen que una política de seguridad informática es una forma de comunicarse con los usuarios, ya que las mismas establecen un canal formal de actuación del personal, en relación con los recursos y servicios informáticos de la organización. No se puede considerar que una política de seguridad informática es una descripción técnica de mecanismos, ni una expresión legal que involucre sanciones a conductas de los empleados, es más bien una descripción de los que deseamos proteger y él por qué de ello, pues cada política de seguridad es una invitación a cada uno de sus miembros a reconocer la información como uno de sus principales activos así como, un motor de intercambio y desarrollo en el ámbito de sus negocios. Por tal razón, las políticas de seguridad deben concluir en una posición consciente y vigilante del personal por el uso y limitaciones de los recursos y servicios informáticos.

1.2 Auditoría automática de las PSI

Según (Morales 2000), los primeros auditores en informática comienzan a reunirse para intercambiar experiencias, y aprovechar el conocimiento formando una base común de conocimiento, así en 1969 fundan la *EDP Auditors Association, Inc.*. Las primeras auditorías a los sistemas de información, consideraban a estos como una “caja negra” a la cual se le introducían datos y ésta los procesaba y

producía ciertos resultados. Poco interés se tenía en las actividades internas de los sistemas de información, la auditoría consistía en examinar tanto las entradas, como las salidas, dejando a un lado dicha caja negra. Esto debido básicamente al desconocimiento de informática.

A finales del año 2000, algunas empresas de vanguardia han entendido la necesidad de prevenir errores o fraudes, en lugar de identificarlos y corregirlos una vez que se han presentado y es así como, actualmente, la Auditoría Informática, audita no sólo el sistema de información en operación, sino también su diseño y construcción, con el fin de evaluar su control interno. Además con el aumento de las estaciones de trabajo en las empresas y las dificultades para los auditores de auditar máquina a máquina se empiezan a desarrollar auditorías de forma automática. La Automatización de Procedimientos de Auditoría es una función de servicio que se desarrolla fundamentalmente para agilizar las revisiones de auditoría, sobre todo de aquellas en las que se puede hacer uso de información que se encuentra en medios magnéticos. Es una forma de dar servicio a las propias áreas de auditoría, a través de la automatización de procedimientos de recolección de información, organización, clasificación, comparación de datos y cifras relevantes para el auditor, entre otras, que los auditores deben realizar, añade (Morales 2000).

1.3 Estudio de sistemas homólogos

A continuación, se realiza un análisis de distintas herramientas utilizadas para la auditoría de las PSI. Se tienen en cuenta los sistemas de auditoría automática de las PSI más utilizados en la actualidad, se exponen las características, ventajas y desventajas para realizar una valoración crítica.

1.3.1 Nessus

Según el sitio web oficial (Tenable 2018), Nessus es la solución de evaluación de vulnerabilidades estándar de la industria para profesionales de la seguridad. Ayuda a los profesionales de la seguridad de primera línea a identificar y reparar vulnerabilidades con rapidez y facilidad, incluso fallas de software,

parches faltantes, *malware* y configuraciones erróneas, en una variedad de sistemas operativos, dispositivos y aplicaciones.

Esta aplicación es la más usada a nivel mundial y presenta las siguientes valoraciones[1]:

- Exactitud: Tiene el índice de falsos positivos más bajo de la industria con una precisión de *six sigma*[2].

- Cobertura: Tiene la cobertura de vulnerabilidades más amplia y profunda de la industria de acuerdo con el Informe de seguridad de aplicaciones de Cybersecurity Insiders durante el año 2018.

- Adopción: Cuenta con la confianza de más de 24.000 organizaciones con 2 millones de descargas en todo el mundo.

  Esta aplicación permite:

- Realizar con políticas y plantillas diseñadas previamente, auditorías de configuración hasta la efectividad de la gestión de parches. Incluye más de 450 plantillas de cumplimiento y configuración para auditar el cumplimiento de la configuración respecto de los puntos de referencia de CIS y otras buenas prácticas.

- Elaborar fácilmente informes en función de las vistas personalizadas, incluidos tipos de vulnerabilidad específicos, vulnerabilidades por *host* o por complemento. Crear informes en una

1 Estos datos de aplicaciones internacionales, son obtenidos de acuerdo con encuestas a asesores de seguridad informática y el Informe de seguridad de aplicaciones de Cybersecurity Insiders durante el año 2018.

2 **Six SIGMA** es una metodología de mejora de procesos creada en Motorola por el ingeniero Bill Smith en la década de los 80, esta metodología está centrada en la reducción de la variabilidad, consiguiendo reducir o eliminar los defectos o fallos en la entrega de un producto o servicio al cliente.

variedad de formatos (HTML, csv y .nessus XML) y personalizarlos fácilmente por equipo o cliente, de modo que puedan enviarse por correo electrónico con cada escaneo.

A pesar de que esta aplicación es la más usada a nivel mundial y posee un conjunto amplio de más de 400 características, presenta una licencia privativa con un costo de $2190 dólares por año, lo que la hace inaccesible para un país que busca la soberanía tecnológica como Cuba. Además el pago se realiza mediante cuentas en línea como Paypal, perteneciente a Estados Unidos, país que impone un bloqueo económico comercial y financiero contra Cuba durante más de 60 años y bloquea el acceso a este y otros servicios en línea, siendo imposible la adquisición de la aplicación con todas sus funcionalidades, solo se puede acceder a la versión de prueba por tiempo limitado. Además no se ajusta a todas las necesidades del centro, siendo una inversión infructuosa. No obstante, fue de utilidad para entender las bases de los sistemas y aplicaciones de auditoría automática, obteniéndose como buenas practicas las siguientes:

- Definir políticas y plantillas para realizar la auditoría de forma más flexible y configurable.
- Generar los informes en varios formatos como html o pdf.

1.3.2 Rapid7

Las soluciones de Rapid7 permiten a las empresas implementar un enfoque activo, basado en el análisis de la seguridad cibernética utilizando técnicas usadas por atacantes reales dando la oportunidad de ver un panorama general de las brechas de seguridad existente en la organización. Las soluciones de detección y respuesta a incidentes, complementan su programa de seguridad con tecnología de análisis de intrusos y conocimientos expertos en la materia. Las soluciones de Rapid7 ayudan a detectar ataques con mayor precisión, le ofrecen respuestas más rápidas en sus investigaciones de incidentes y añaden a su equipo personal de respuesta a incidentes con experiencia cuando lo necesita. El equipo de Metasploit se encuentra entre los primeros en poner a prueba nuevos métodos, como robar credenciales, y Rapid7 estudia cómo los atacantes apuntan a las vulnerabilidades más comunes. Sus servicios profesionales penetran en 500 empresas al año para poner a prueba sus defensas.

Esta aplicación ocupa el tercer puesto a nivel mundial y presenta las siguientes características:

1. NEXPOSE: reduzca el riesgo de brechas. Con Nexpose, los mayores riesgos se priorizan empleando inteligencia de amenazas teniendo en cuenta los aspectos importantes para su empresa.

2. METASPLOIT: ponga a prueba sus defensas. Con ella puede desvelar las debilidades de sus defensas, centrarse en los mayores riesgos y mejorar sus resultados de seguridad.

3. USERINSIGHT: encuentre ataques que desconocía. Identifica a los intrusos que utilizan métodos de ataque sigilosos, como credenciales robadas y movimiento lateral (Ohka 2018).

Esta aplicación ocupa la posición 3 a nivel mundial por sus amplias características. A causa del bloqueo estadounidense, esta aplicación se encuentra bloqueada para Cuba. Esto impide la descarga de la misma y su soporte. Además no permite realizar configuraciones personalizadas para adecuarse a cada empresa. De esta aplicación se tomó como elemento importante, la necesidad de revisar que usuarios tienen acceso a los sistemas de cómputo y si este acceso está previamente autorizado.

1.3.3 White Hat Security

La plataforma de seguridad de aplicación *WhiteHat* proporciona todos los servicios necesarios para asegurar todo el ciclo de vida del desarrollo del software. Desde soluciones para el equipo de seguridad hasta productos rápidos y precisos para desarrolladores en entornos DevOps[3], ayuda a las organizaciones a disfrutar de todos los beneficios de la transformación digital sin los problemas de seguridad. Más de 15 años de datos y análisis. Con la mejor tecnología de seguridad de aplicaciones de su clase, las evaluaciones siempre detectan vectores de ataque y escanean el código de su aplicación. La plataforma

3 DevOps es un acrónimo inglés de *development* (desarrollo) y *operations* (operaciones), que se refiere a una metodología de desarrollo de *software* que se centra en la comunicación, colaboración e integración entre desarrolladores de *software* y los profesionales de sistemas en las tecnologías de la información (IT).

de seguridad de aplicaciones WhiteHat cierra la brecha entre la seguridad y los DevOps para asegurar que pueda encontrar y corregir vulnerabilidades antes de que los *hackers* puedan explotarlas.

Brinda seguridad a lo largo de todo el ciclo de vida del desarrollo de software con:

1. Pruebas de seguridad de aplicaciones dinámicas (DAST): WhiteHat Sentinel Dynamic identifica y verifica con precisión las vulnerabilidades en sitios web y aplicaciones web.

2. Pruebas de seguridad de aplicaciones estáticas (SAST): WhiteHat Sentinel Source y WhiteHat Scout escanean código fuente completo, identifican vulnerabilidades y proporcionan descripciones de vulnerabilidad detalladas y consejos de remediación.

3. Pruebas de seguridad de aplicaciones móviles: WhiteHat Sentinel Mobile emplea una combinación de escaneo automático dinámico y estático, así como también evaluaciones manuales realizadas por expertos ingenieros de seguridad en nuestro Centro de Investigación de Amenazas.

4. Elearning: WhiteHat Security eLearning ofrece la forma más rentable de respaldar y escalar una iniciativa de capacitación en su empresa (WhiteHat Security 2018).

Esta aplicación ocupa el lugar 5 a nivel mundial y es de uso gratuito por solo 30 días. El pago de la aplicación también es por servicios de cuentas en línea, los cuales están bloqueados para Cuba. Esto impide el uso continuado y el soporte de la misma. Además se basa solo en el escaneo de vulnerabilidades de seguridad, no realiza auditorías físicas ni lógicas de los sistemas de cómputo sino que solo analiza las aplicaciones y softwares. De esta solución se define como requisito importante, el escaneo completo de código fuente para identificar vulnerabilidades.

1.3.4 Checkmarx

La Plataforma de Exposición de Software Checkmarx alinea la Seguridad del Software con la cultura de DevOps, detectando, priorizando de manera inteligente y remediando la exposición a lo largo del ciclo de

vida del desarrollo del software (SDLC) desde la etapa de codificación hasta la etapa de prueba de la aplicación en tiempo de ejecución. Checkmarx proporciona una combinación de integraciones para la automatización requerida en un entorno de desarrollo acelerado junto con una variedad de tecnologías para capacitar a los equipos de desarrollo y seguridad para mejorar la postura de seguridad general de una organización.

Presenta los siguientes servicios:

1. CxSAST: Un producto de análisis de código fuente altamente preciso y flexible que permite a las organizaciones escanear automáticamente código no compilado/sin construir e identificar cientos de vulnerabilidades de seguridad en los lenguajes de codificación más prevalentes.

2. CxOSA: Aplica el análisis de código abierto como parte del SDLC y administra los componentes de código abierto al tiempo que garantiza que los componentes vulnerables se eliminen o reemplacen antes de que se conviertan en un problema.

3. CxIAST: Una solución que detecta vulnerabilidades en la ejecución de aplicaciones bajo prueba. Creado para DevOps, se integra a la perfección en su canalización de CI / CD. CxIAST proporciona detección avanzada de vulnerabilidad con cero impacto en los tiempos de ciclo de prueba.

4. CxCodebashing: Una plataforma de capacitación interactiva de seguridad del software que agudiza las habilidades que los desarrolladores necesitan para evitar problemas de seguridad, corregir vulnerabilidades y escribir códigos seguros en primer lugar.

5. AppSec Accelerator: Un servicio administrado de Software Security que ayuda a las organizaciones de desarrollo a hacer la transición a un SDLC seguro. Con AppSec Accelerator, un equipo de expertos calificados lo ayuda a aumentar, optimizar y automatizar sus pruebas de AppSec (Checkmarx 2018).

Esta aplicación presenta la posición 8 a nivel mundial y es de la mejores escaneando código fuente. Sin embargo es privativa y no presenta todas las funcionalidades requeridas para las auditorías en las empresas. Requiere de pago mediante cuentas en línea, las cuales son inaccesibles desde Cuba. Como buena práctica se obtiene la automatización del proceso de auditoría de forma sistémica.

1.3.5 SPS-Auditor

Secure Police Script (SPS) es un conjunto de instrucciones programadas en Bash para configurar y comprobar las PSI de manera semi-automática. Para utilizar SPS se debe copiar la carpeta “SPS-Auditor” que contiene el *script* y un archivo de licencia para el SavUnix en el escritorio y seguir los pasos orientados en el manual adjunto, según afirman (Chávez y Quezada 2018).

Esta solución presenta las siguientes características:

- Es necesario tener permisos de administración para ejecutar los archivos .sh los cuales se compilarán en la consola del sistema, mostrando varias opciones de configuración.

- Se debe actualizar con cada nueva versión de los sistemas operativos GNU/Linux, para añadir los correspondientes repositorios.

- El asesor para chequear las PSI, debe transitar por cada máquina, ejecutar el *script* y copiar los informes en una memoria USB para luego imprimirlos.

- Contiene todas las políticas de seguridad del Centro de Ideoinformática.

A pesar de que esta herramienta cumple con las expectativas de seguridad del Centro de Ideoinformática, resulta muy engorroso para los asesores de seguridad el auditar máquina a máquina, lo que causa pérdida de tiempo en traslados y espera de la ejecución del *script*. Esta solución será usada como guía para la nueva solución, manteniendo los principios básicos identificados y reutilizando su código fuente.

Valoración crítica de los sistemas estudiados

Existen distintos módulos para la auditoría automática de las PSI, pero no responden a las necesidades de desarrollo del Centro de Ideoinformática. La mayoría de los sistemas de auditoría automática existentes son privativos lo que no asegura a Cuba la soberanía tecnológica y no existen homólogos en este campo de acción. Por otro lado, la solución en Bash de la UCI es semi-automática, auditando máquina a máquina y necesita ser actualizada de forma regular. Con todas estas herramientas se comprendieron las bases y los principios básicos del sistema a desarrollar para dar solución a la problemática planteada.

1.4 Estudio de tecnologías para el desarrollo de la solución

Después de analizar las particularidades requeridas para la propuesta de solución y los componentes necesarios, se comienza un estudio sobre los marcos de trabajo (*framework*), de ellos los orientados al desarrollo web, como mejor alternativa para desarrollar la aplicación debido a la flexibilidad de configuración y control que estos permiten.

1.4.1 Framework

En general, un marco de trabajo, o *framework*, es una estructura real o conceptual destinada a servir de soporte o guía para la construcción de algo que expande la estructura en algo útil (Rouse 2016a).

1.4.2 Framework de desarrollo

En los sistemas informáticos, un *framework* es a menudo una estructura en capas que indica qué tipo de programas pueden o deben ser construidos y cómo se interrelacionan. Algunos marcos de trabajo de sistemas informáticos también incluyen programas reales, especifican interfaces de programación u ofrecen herramientas de programación para usar los marcos (Rouse 2016a).

En otras palabras, un *framework* se puede considerar como una aplicación genérica incompleta y configurable a la cual se puede añadir las últimas piezas, para construir una aplicación concreta. Los objetivos principales que persigue un *framework* son: acelerar el proceso de desarrollo, reutilizar código ya existente y promover buenas prácticas de desarrollo como el uso de patrones. Un *framework* web, por tanto, se puede definir como un conjunto de componentes que conforman un diseño reutilizable que facilita y agiliza el desarrollo de sistemas web (Gutiérrez 2014).

Entre ellos se pueden encontrar los siguientes:

1.4.3 Django

Django es un marco web de Python de alto nivel que fomenta el desarrollo rápido y el diseño limpio y pragmático, asegura su sitio oficial (Django 2018). Creado por desarrolladores experimentados, se encarga de gran parte de las complicaciones del desarrollo web, por lo que puede concentrarse en escribir su aplicación sin necesidad de innovar. Es gratis y de código abierto.

Presenta las siguientes características:

1. Rápido. Django fue diseñado para ayudar a los desarrolladores a llevar las aplicaciones desde el concepto hasta su finalización lo más rápido posible.

2. Completamente cargado. Django incluye docenas de extras que puede usar para manejar tareas comunes de desarrollo web. Django se encarga de la autenticación del usuario, la administración de contenido, los mapas del sitio, las fuentes RSS y muchas más tareas, de manera inmediata.

3. Seguro. Django toma en serio la seguridad y ayuda a los desarrolladores a evitar muchos errores comunes de seguridad, como la inyección de SQL, los *scripts* entre sitios, la falsificación de solicitudes entre sitios y el *clickjack*. Su sistema de autenticación de usuarios proporciona una forma segura de administrar cuentas de usuario y contraseñas.

4. Muy escalable. Algunos de los sitios más concurridos del planeta utilizan la capacidad de Django para escalar de manera rápida y flexible para satisfacer las demandas de tráfico más pesadas.

5. Versátil. Compañías, organizaciones y gobiernos han usado Django para construir todo tipo de cosas, desde sistemas de administración de contenido hasta redes sociales y plataformas de computación científica.

1.4.4 Symfony 3.4.21

Symfony es un conjunto de componentes PHP reutilizables. La base estándar sobre la cual se construyen las mejores aplicaciones PHP. Presenta 50 componentes independientes disponibles para sus propias aplicaciones y un marco PHP para proyectos web. Acelera la creación y el mantenimiento de aplicaciones web PHP, plantea su creador (Potencier 2017).

Características principales:

1. Rápido y consumo menor de memoria: Symfony fue concebido desde el comienzo para favorecer el desempeño. Symfony es más o menos tres veces más veloz que la versión catorce de Zend Framework, utilizando la mitad de consumo de memoria.

2. Flexibilidad ilimitada: Symfony es acomodable:

- Administración de la complejidad: Permite desarrollar aplicaciones complejas con múltiples funcionalidades.

- Pieza a pieza: Se puede edificar conforme con las funciones requeridas.

- *Microframework*: Symfony se puede usar para desarrollar una funcionalidad concreta. Sin reconstruir todo y sin precisar instalar todo el *framework*, solo la pieza específica requerida.

- Soporte: Proporcionado por Sensio, de manera adicional la comunidad y las compañías de servicios que han invertido en este *framework*.

- Por último, asimismo es con miras a un desarrollo sustentable que Symfony se distribuye bajo licencia Open Source MIT, que no impone limitaciones y deja el desarrollo de código abierto, como aplicaciones dueñas.

3. Ampliable: Desde la más pequeña pieza a la base completa en sí, todo se presenta como un *plug-in* en Symfony. Cada *plug-in* está destinado para agregar funcionalidad al *framework*, lógicamente, y cada *plug-in* asimismo puede ser vuelto a utilizar en otro proyecto o bien compartido con el resto de la comunidad. En todo caso, el sistema deja todo cambio en Symfony, incluyendo el propio núcleo.

4. Estable y sostenible: Desarrollado por los Laboratorios Sensio, las primordiales versiones de Symfony son soportados por tres años por la compañía. E inclusive para toda la vida en lo que se refiere a las cuestiones relacionadas con la seguridad se refiere. Para mayor estabilidad, las versiones menores del contrato Symfony y la interfaz asimismo están garantizadas y la compatibilidad entre todas y cada una de las versiones secundarias se realizará en el API definido por las interfaces públicas.

5. La alegría de desarrollo: Al cuidar a una serie de labores desapacibles (desarrollo de funcionalidades de menor relevancia, por poner un ejemplo), Symfony deja a los desarrolladores centrarse en los aspectos más señalados reales de una aplicación y para los dos totalmente validar su papel y prosperar su productividad. Entre las herramientas de Symfony diseñadas a fin de que la vida de un desarrollador sea considerablemente más simple, está la barra de herramientas de depuración web legendario, como soporte nativo para ambientes de desarrollo, páginas de fallo detallados o bien de seguridad nativa.

6. Facilidad de uso: Absolutamente flexible para satisfacer las necesidades de los profesionales y usuarios avanzados por igual, Symfony es asimismo muy alcanzable. Rebosante documentación, la comunidad y el apoyo profesional.

Ventajas:

1. Symfony es un software de código abierto con una comunidad de más de 600,000 desarrolladores y más de 120 países. Presenta más de 3,000 contribuidores y 48,000,000 descargas mensuales.

2. Presenta un conjunto de componentes desacoplados y reutilizables en los que se crean las mejores aplicaciones de PHP, como Drupal, phpBB y eZ Publish.

3. Es flexible: se adapta a casi cualquier necesidad, permitiendo instalar únicamente las piezas requeridas para el proyecto en vez de todo el *framework*.

4. Es ampliable: es el proyecto PHP más activo, lo que garantiza encontrar paquetes para prácticamente cualquier funcionalidad.

5. Es un sistema estable: Laboratorios Sensio garantiza que cada versión de Symfony recibirá soporte (actualizaciones y solución de problemas) durante tres años, además de compatibilidad con las versiones secundarias.

6. Un sistema rápido y que consume poca memoria: Symfony ha sido desarrollado con la idea de ofrecer aplicaciones de alto rendimiento, es más rápido que otros *frameworks* usando la mitad de la memoria.

7. Facilidad de uso: Gracias a la gran cantidad de documentación y tutoriales que se pueden encontrar en la web, cualquier profesional o usuario avanzado puede aprender rápidamente los conceptos más básicos de Symfony. Para ello ha tomado las mejores ideas de sus competidores, como Ruby on Rails o Django.

Selección del *Framework* de desarrollo web

Se define como *framework* de desarrollo web a Symfony, por su flexibilidad, compatibilidad y seguridad. Este *framework* permitirá mediante sus elementos desacoplables, desarrollar la solución necesaria, adaptándose a las necesidades específicas de la problemática planteada. Presenta varios parches de seguridad que se actualizan periódicamente asegurando la aplicación. Su amplia linea de comandos permite generar diversos componentes del desarrollo de forma semiautomática. Además el amplio conjunto de *bundles* permitirán acelerar el desarrollo, reutilizando soluciones comunes y personalizándolas a las necesidades especificas. Todos los elementos anteriores facilitan el trabajo a los programadores.

1.5 Tecnologías, metodologías y herramientas

En el desarrollo de aplicaciones web en Symfony se utilizan varias tecnologías como lenguajes de programación, herramientas de modelado, gestores de bases de datos, y servidores web. A continuación, se realiza una descripción de las tecnologías y herramientas utilizadas para la implementación de la propuesta de solución y se selecciona la metodología de desarrollo que se utiliza para guiar el proceso de desarrollo del software.

1.5.1 Metodología de desarrollo de software

Las metodologías de desarrollo de software son el proceso que se suele seguir a la hora de diseñar una solución o un programa específico. Tiene que ver, por tanto, con la comunicación, la manipulación de modelos y el intercambio de información y datos entre las partes involucradas. Son enfoques de carácter estructurado y estratégico que permiten el desarrollo de programas con base a modelos de sistemas, reglas, sugerencias de diseño y guías, según la definición de (OBS Business School 2018).

1.5.1.1 AUP-UCI

AUP-UCI es una variante realizada por la Universidad de las Ciencias Informáticas a la metodología ágil AUP (Proceso Ágil Unificado) y está definida por la universidad como el documento rector de la actividad productiva. Fue diseñada por (Sánchez 2015) y es aplicada en la mayoría de los procesos productivos como un modelo híbrido.

Por las razones expuestas anteriormente se selecciona esta metodología para el desarrollo de la propuesta de solución, seleccionando el escenario 4 que utiliza en su diseño las historias de usuario. A continuación se definen las características de esta metodología según dispuso su autora.

Fases de Variación de AUP para la UCI:

La metodología Variación de AUP para la UCI está forma por tres fases, (Inicio, Ejecución y Cierre) para el ciclo de vida de los proyectos de la universidad, las cuales contienen las características de las cuatro fases (Inicio, Elaboración, Construcción y Transición) propuestas en AUP. Las características de las fases de la metodología de la universidad son:

- Inicio: Durante el inicio del proyecto se llevan a cabo las actividades relacionadas con la planeación del proyecto. En esta fase se realiza un estudio inicial de la organización cliente que permite obtener información fundamental acerca del alcance del proyecto, realizar estimaciones de tiempo, esfuerzo y costo y decidir si se ejecuta o no el proyecto.

- Ejecución: En esta fase se ejecutan las actividades requeridas para desarrollar el software, incluyendo el ajuste de los planes del proyecto considerando los requisitos y la arquitectura. Durante el desarrollo se modela el negocio, se obtienen los requisitos, se elaboran la arquitectura y el diseño, se implementa y se libera el producto. Durante esta fase el software es transferido al ambiente de los usuarios finales o entregado al cliente junto con la documentación. Además, en esta transición se capacita a los usuarios finales sobre la utilización de la aplicación.

- Cierre: En el cierre se analizan tanto los resultados del proyecto como su ejecución y se realizan las actividades formales de cierre del proyecto.

Las disciplinas definidas en la variación AUP para la UCI son:

- Modelado de negocio: destinada a comprender los procesos de negocio de una organización. Para modelar el negocio se proponen las siguientes variantes: Casos de Uso del Negocio (CUN), Descripción de Proceso de Negocio (DPN) y Modelo Conceptual (MC).

- Requisitos: comprende la administración y gestión de los requisitos funcionales y no funcionales del producto, agrupados en cuatro escenarios condicionados por el Modelado de negocio. Existen tres formas de encapsular los requisitos [Casos de Uso del Sistema (CUS), Historias de usuario (HU) y Descripción de requisitos por proceso (DRP)].

- Análisis y diseño: se modela el sistema y su forma (incluida su arquitectura) para que soporte los requisitos, incluyendo los requisitos no funcionales.

- Implementación: a partir de los resultados del Análisis y Diseño se construye el sistema.

- Pruebas internas: se verifica el resultado de la implementación probando cada construcción, incluyendo tanto las construcciones internas como intermedias, así como las versiones finales a ser liberadas.

- Pruebas de liberación: pruebas diseñadas y ejecutadas por una entidad certificadora externa, dela calidad a todos los entregables de los proyectos antes de ser entregados al cliente para su aceptación.

- Pruebas de Aceptación: es la prueba final antes del despliegue del sistema. Su objetivo es verificar que el software está listo y que puede ser usado por usuarios finales para ejecutar aquellas funciones y tareas para las cuales el software fue construido.

Todas las disciplinas antes definidas se desarrollan en la Fase de Ejecución, de ahí que en la misma se realicen Iteraciones y se obtengan resultados incrementales.

1.5.2 Tecnologías

Para el desarrollo de aplicaciones web en Symfony y la modelación de los artefactos de la metodología seleccionada se utilizan diversas tecnologías. A continuación, se describen y caracterizan cada una de las utilizadas en el desarrollo de la propuesta de solución. En algunos casos se establece una comparación entre distintas tecnologías para seleccionar la que más se adapta al sistema.

1.5.2.1 Preprocesador de Hipertexto (PHP 7)

Afirma (Figueroa 2018) que PHP es un lenguaje de *script* interpretado en el lado del servidor utilizado para la generación de páginas Web dinámicas, similar al ASP de Microsoft o el JSP de Sun, embebido en páginas HTML y ejecutado en el servidor. La mayor parte de su sintaxis ha sido tomada de C, Java y Perl con algunas características específicas de sí mismo. La meta del lenguaje es permitir rápidamente a los desarrolladores la generación dinámica de páginas. No es un lenguaje de marcas como podría ser HTML, XML o WML. Está más cercano a JavaScript o a C. A diferencia de Java o JavaScript que se ejecutan en el navegador, PHP se ejecuta en el servidor, por eso nos permite acceder a los recursos que tenga el servidor como por ejemplo podría ser una base de datos. El programa PHP es ejecutado en el servidor y el resultado es enviado al navegador. El resultado es normalmente una página HTML pero también podría ser una página WML (Wap). Al ser PHP un lenguaje que se ejecuta en el servidor no es necesario que su navegador lo soporte, es independiente del navegador, sin embargo, para que sus páginas PHP funcionen el servidor donde están alojadas debe soportar PHP.

Al ser un lenguaje libre dispone de una gran cantidad de características que lo convierten en la herramienta ideal para la creación de páginas web dinámicas:

- Soporte para una gran cantidad de bases de datos: MySQL, PostgreSQL, Oracle,MS SQL Server, SybasemSQL, Informix, entre otras.

- Integración con varias bibliotecas externas, permite generar documentos en PDF (documentos de Acrobat Reader) hasta analizar código XML.

- Ofrece una solución simple y universal para las paginaciones dinámicas del Web de fácil programación.

Para visualizar la interfaz de Symfony en su versión 3.x.x es necesaria la versión 7.0.1 o superior de PHP por cuestiones de compatibilidad, ya que este *framework, a* medida que se desarrolla el PHP, optimiza sus módulos adecuándose a las nuevas versiones, ganando en rendimiento y seguridad, pero causando conflictos con versiones anteriores.

1.5.2.2 Lenguaje de marcado de hipertexto (HTML 5)

HTML5 es la última versión de HTML definida en (Mozilla MDN 2017). El término representa dos conceptos diferentes:

1. Se trata de una nueva versión de HTML, con nuevos elementos, atributos y comportamientos.

2. Contiene un conjunto más amplio de tecnologías que permite a los sitios web y a las aplicaciones ser más diversas y de gran alcance. A este conjunto se le llama HTML5 y amigos, a menudo reducido a HTML5.

Diseñado para ser utilizable por todos los desarrolladores de Open Web, esta página referencia numerosos recursos sobre las tecnologías de HTML5, clasificados en varios grupos según su función.

- Semántica: Permite describir con mayor precisión cuál es su contenido.

- Conectividad: Permite comunicarse con el servidor de formas nuevas e innovadoras.

- Sin conexión y almacenamiento: Permite a las páginas web almacenar datos localmente en el lado del cliente y operar sin conexión de manera más eficiente.

- Multimedia: Nos otorga un excelente soporte para utilizar contenido multimedia como lo son audio y video nativamente.

- Gráficos y efectos 2D/3D: Proporciona una amplia gama de nuevas características que se ocupan de los gráficos en la web como lo son canvas 2D, WebGL, SVG, etc.

- Rendimiento e Integración: Proporciona una mayor optimización de la velocidad y un mejor uso del *hardware*.

- Acceso al dispositivo: Proporciona APIs para el uso de varios componentes internos de entrada y salida de nuestro dispositivo.

- CSS3: Nos ofrece una nueva gran variedad de opciones para hacer diseños más sofisticados.

1.5.2.3 Hojas de Estilo en Cascada (CSS 3)

Según (Mozilla MDN 2018) CSS3 es la última evolución del lenguaje de las *Hojas de Estilo en Cascada (Cascading Style Sheets*), y pretende ampliar la versión CSS2.1. Trae consigo muchas novedades altamente esperadas, como las esquinas redondeadas, sombras, gradientes, transiciones o animaciones, y nuevos *layouts* como multi-columnas, cajas flexibles o maquetas de diseño en cuadrícula (*grid layouts*). Las partes experimentales son particulares para cada navegador y deberían ser evitadas en entornos de producción, o usadas con extrema precaución, ya que tanto la sintaxis como la semántica pueden cambiar en el futuro.

1.5.2.4 JavaScript

JavaScript (JS) es un lenguaje ligero e interpretado, orientado a objetos con funciones de primera clase, más conocido como el lenguaje de script para páginas web, pero también usado en muchos entornos sin navegador, tales como node.js, Apache CouchDB y Adobe Acrobat. Es un lenguaje script multi-paradigma, basado en prototipos, dinámico, soporta estilos de programación funcional, orientada a objetos e imperativa. El estándar de JavaScript es ECMAScript. Desde el 2012, todos los navegadores modernos soportan completamente ECMAScript 5.1. Los navegadores más antiguos soportan por lo menos ECMAScript 3. El 17 de Julio de 2015, ECMA International publicó la sexta versión de ECMAScript, la cual es oficialmente llamada ECMAScript 2015, y fue inicialmente nombrada como ECMAScript 6 o ES6. Desde entonces, los estándares ECMAScript están en ciclos de lanzamiento anuales (Mozilla MDN 2019).

1.4.2.5 jQuery

jQuery es una biblioteca de JavaScript rápida, pequeña y con muchas funciones. Hace que cosas como la manipulación y manipulación de documentos HTML, el manejo de eventos, la animación y Ajax sean mucho más simples con una API fácil de usar que funciona en una gran cantidad de navegadores. Con una combinación de versatilidad y extensibilidad, jQuery ha cambiado la forma en que millones de personas escriben JavaScript (The jQuery Foundation 2019).

1.5.2.6 Bootstrap

Bootstrap es utilizado para crear proyectos adaptables y móviles en la web con la biblioteca de componentes de *front-end* según su sitio oficial (Bootstrap 2018). Bootstrap es un conjunto de herramientas de código abierto para desarrollar con HTML, CSS y JS. Realiza rápidamente prototipos de ideas o construye una aplicación completa con variables y *mixins* de Sass, (sistema de cuadrícula sensible), extensos componentes precompilados y potentes complementos integrados en jQuery.

1.5.2.7 Bash

Bash es el *Bourne Again Shell* afirma (GNU Project 2018). Bash es un *shell* compatible con sh que incorpora características útiles del *shell* Korn (ksh) y del *shell* C (csh). Está diseñado para cumplir con el estándar IEEE POSIX P1003.2 / ISO 9945.2 Shell y herramientas. Ofrece mejoras funcionales sobre sh tanto para programación como para uso interactivo. Además, Bash puede ejecutar la mayoría de los *scripts* sh sin modificaciones. Las mejoras ofrecidas por Bash incluyen edición de línea de comandos, tamaño ilimitado del historial de comandos, control de trabajo, funciones de shell y alias, arreglos indexados de tamaño ilimitado, aritmética entera en cualquier base de dos a sesenta y cuatro.

1.5.2.8 SSH 2

SSH, (*Secure Shell),* es un enfoque popular, potente y basado en software para la seguridad de la red. Cada vez que una computadora envía datos a la red, SSH los cifra automáticamente. Cuando los datos llegan a su destinatario deseado, SSH lo descifra automáticamente. El resultado es un cifrado transparente: los usuarios pueden trabajar normalmente, sin saber que sus comunicaciones están cifradas de forma segura en la red. Además, SSH utiliza algoritmos de encriptación modernos y seguros, y es lo suficientemente efectivo como para encontrarse dentro de las aplicaciones de misión crítica en las principales corporaciones (Silverman et al. 2002).

1.5.2.9 Cron

Cron es el nombre del programa que permite a usuarios Linux/Unix ejecutar automáticamente comandos o scripts (grupos de comandos) a una hora o fecha específica. Es usado normalmente para comandos de tareas administrativas, como respaldos, pero puede ser usado para ejecutar cualquier cosa. Como se define en las páginas del manual de cron (#> man cron) es un demonio que ejecuta programas agendados. Cron es un demonio (servicio), lo que significa que solo requiere ser iniciado una vez, generalmente con el mismo arranque del sistema (González 2019).

1.5.2.10 Llaves públicas y privadas

Las llaves son un método de autenticación de red que permite a los usuarios autenticarse en las PC sin necesidad de introducir una contraseña. El protocolo que sigue este mecanismo es una combinación de hardware y software, políticas y procedimientos de seguridad que permiten la ejecución con garantías de operaciones criptográficas como el cifrado, la firma digital o el no repudio de transacciones electrónicas. A través de un juego entre distintos caracteres informáticos, la llave pública solamente puede cifrar. La llave privada puede descifrar o hacer las dos cosas, aunque esto último no es tan importante (Noriega 2015).

1.5.2.11 PAM

Módulos de autenticación conectables, PAM por sus siglas en inglés (*Pluggable Authentication Modules*), es un mecanismo flexible para la autenticación de usuarios centralizado. Permite modelar políticas de seguridad personalizada dependiendo el servicio para distintos usuarios. Red hat lo implementa por defecto en su sistema. Permite el desarrollo de programas independientes del mecanismo de autenticación que se utilice, es un refuerzo de estos como por ejemplo del clásico /etc/passwd que cifra la contraseña en el /etc/shadow (Nexolinux 2013).

1.5.2.12 Eventos enviados por el servidor (SSE)

Un evento enviado por el servidor, SSE por sus siglas en inglés (*Server Sent Events)* es cuando una página web recibe actualizaciones de un servidor automáticamente, afirma (w3schools 2019). Esto también era posible antes, pero la página web tendría que preguntar si había actualizaciones disponibles. Con los eventos enviados por el servidor, las actualizaciones vienen automáticamente. Ejemplos: actualizaciones de Facebook/Twitter, actualizaciones de precios de acciones, noticias, resultados deportivos u otros. Para que el SSE funcione, necesita un servidor capaz de enviar actualizaciones de datos como PHP o ASP. La sintaxis del flujo de eventos del lado del servidor es simple. Se establece el encabezado "Content-Type" en "text/event-stream". Ahora se puede comenzar a enviar transmisiones de eventos. El *server-sent event* API está contenido en la interfaz EventSource; para abrir una conexión al servidor para recibir eventos de él. Se crea un nuevo objeto EventSource, especificando el URI de un script que genera los eventos. Por ejemplo:

```
var evtSource = new EventSource("ssedemo.php");
```

Cuando se producen problemas (como un tiempo de espera o problemas relacionados con el control de acceso), se genera un evento de error. Se puede tomar acción sobre esto al implementar una devolución de llamada al objeto EventSource.

1.5.2.13 Lenguaje Unificado de Modelado (UML)

El Lenguaje de Modelado Unificado (UML) es “un lenguaje estándar para escribir diseños de software. El UML puede usarse para visualizar, especificar, construir y documentar los artefactos de un sistema de software intensivo”. En otras palabras, tal como los arquitectos de edificios crean planos para que los use una compañía constructora, los arquitectos de software crean diagramas de UML para ayudar a los desarrolladores de software a construir el software. Si usted entiende el vocabulario del UML (los elementos pictóricos de los diagramas y su significado) puede comprender y especificar con mucha más facilidad un sistema, y explicar su diseño a otros (Pressman 2010).

1.5.3 Sistemas Gestores de Bases de Datos

Plantea (PowerData 2018) que los Sistemas Gestores de Bases de Datos (SGBD, por sus siglas en inglés), también conocidos como sistemas manejadores de bases de datos o DBMS (*DataBase Management System*), son un conjunto de programas que manejan todo acceso a la base de datos, con el objetivo de servir de interfaz entre ésta, el usuario y las aplicaciones utilizadas. Gracias a este sistema de software específico el usuario puede gestionar la base de datos (almacenar, modificar y acceder a la información contenida en ésta) mediante el uso de distintas herramientas para su análisis, con las que puede realizar consultas y generar informes. Además de gestionar los datos y mantener su consistencia, su utilización supone numerosas ventajas a la hora de construir y definir la base de datos a diferentes niveles de abstracción para distintas aplicaciones, pues facilita los procesos y también su mantenimiento.

La ejecución de las operaciones sobre la base de datos para luego proporcionarlos al usuario en función de su requerimiento se realiza de un modo eficiente y seguro. Sus características de un SGDB posibilitan el cumplimiento de una serie de funciones, que pueden agruparse de la siguiente manera:

1. Definición de los datos: El SGBD ha de poder definir todos los objetos de la base de datos partiendo de definiciones en versión fuente para convertirlas en la versión objeto.

2. Manipulación de los datos: El SGBD responde a las solicitudes del usuario para realizar operaciones de supresión, actualización, extracción, entre otras gestiones. El manejo de los datos ha de realizarse de forma rápida, según las peticiones realizadas por los usuarios, y permitir la modificación del esquema de la base de datos gracias a su independencia.

3. Seguridad e integridad de los datos: Además de registrar el uso de las bases de datos, ante cualquier petición, también aplicará las medidas de seguridad e integridad de los datos (adopta medidas garantizar su validez) previamente definidas. Un SGBD debe garantizar su seguridad frente a ataques o simplemente impedir su acceso a usuarios no autorizados por cualquier razón.

4. Recuperación y restauración de los datos: La recuperación y restauración de los datos ante un posible fallo es otra de las principales funciones de un SGBD. Su aplicación se realizará a través de un Plan de recuperación y restauración de los datos que sirva de respaldo.

Symfony soporta diferentes gestores de bases de datos, esto se consigue mediante una capa de abstracción de la base de datos que convierte las instrucciones genéricas proporcionadas por Symfony en instrucciones particulares de cada una de ellas.

1.5.3.1 MySQL

MySQL es la base de datos de código abierto más popular del mundo. Con su rendimiento, confiabilidad y facilidad de uso comprobados, MySQL se ha convertido en la opción de base de datos líder para aplicaciones basadas en web, utilizadas por propiedades web de alto perfil como Facebook, Twitter,

YouTube, Yahoo! y muchos más. Oracle impulsa la innovación de MySQL y ofrece nuevas capacidades para potenciar la próxima generación de aplicaciones web, en la nube, móviles e integradas (MySQL 2018).

1.5.3.2 PostgreSQL

PostgreSQL es un potente sistema de base de datos relacional de objetos de código abierto que utiliza y amplía el lenguaje SQL combinado con muchas características que almacenan y escalan de forma segura las cargas de trabajo de datos más complicadas. Los orígenes de PostgreSQL se remontan a 1986 como parte del proyecto POSTGRES en la Universidad de California en Berkeley y tiene más de 30 años de desarrollo activo en la plataforma central. PostgreSQL se ha ganado una sólida reputación por su arquitectura probada, confiabilidad, integridad de datos, conjunto de características sólidas, extensibilidad y la dedicación de la comunidad de código abierto detrás del software para ofrecer constantemente soluciones innovadoras y de alto rendimiento. PostgreSQL se ejecuta en todos los sistemas operativos principales, ha sido compatible con ACID desde 2001, y tiene complementos poderosos como el popular extensor de base de datos geoespacial PostGIS. No es sorprendente que PostgreSQL se haya convertido en la base de datos relacional de código abierto elegida por muchas personas y organizaciones (PostgreSQL 2018).

Selección del sistema gestor de base de datos

Se selecciona como sistema gestor de base de datos a MySQL por su amplia compatibilidad con sistemas operativos tanto Windows, Mac OS X, UNIX y GNU/Linux, sus múltiples herramientas y contar con las interfaces de programación de aplicaciones con los lenguajes C, C ++, Java, Perl, PHP y Ruby. Se ha convertido en la opción de base de datos líder para aplicaciones basadas en web, siendo usada por propiedades web de alto perfil. Es un sistema más simple que permite trabajar de forma sencilla, rápida y con comodidad con *queries* simples y bases de datos pequeñas o medianas. Se requiere de consultas pequeñas y simples para lo cual presenta el mayor rendimiento en comparación con PostgreSQL Que es más óptimo con consultas largas y complejas.

1.5.4 Servidor de aplicación web

Un servidor de aplicaciones según (Rouse 2016b) es un programa de servidor en un equipo en una red distribuida que proporciona la lógica de negocio para un programa de aplicación. El servidor de aplicaciones se ve frecuentemente como parte de una aplicación de tres niveles, que consta de un servidor gráfico de interfaz de usuario (GUI), un servidor de aplicaciones (lógica empresarial) y un servidor de bases de datos y transacciones. De manera más descriptiva, se puede visualizar como la división de una aplicación en:

1. Una interfaz gráfica de usuario de primer nivel, de *front-end*, basada en el navegador web, normalmente en un equipo de cómputo personal o una estación de trabajo.

2. Una aplicación de lógica de negocio de nivel medio o conjunto de aplicaciones, posiblemente en una red de área local o un servidor de *intranet*.

3. Un servidor de *back-end*, base de datos y transacciones de tercer nivel, a veces en un *mainframe* o servidor grande.

Las bases de datos de aplicaciones más antiguas y las aplicaciones de administración de transacciones forman parte del *backend* o tercer nivel. El servidor de aplicaciones es el intermediario entre bases de datos basadas en navegador y bases de datos de *back-end* y sistemas heredados. En muchos usos, el servidor de aplicaciones combina o funciona con un servidor Web (*Hypertext Transfer Protocol*) y se denomina servidor de aplicaciones web. El navegador web admite un *front-end* fácil de crear basado en HTML para el usuario. El servidor Web proporciona varias formas diferentes de reenviar una solicitud a un servidor de aplicaciones y de reenviar una página Web nueva o modificada al usuario. Estos enfoques incluyen la interfaz de *gateway* común (CGI), FastCGI, la página Active Server de Microsoft y la página de servidor de Java. En algunos casos, los servidores de aplicaciones Web también admiten interfaces de "intermediación" de solicitud, como el Protocolo de Internet Inter-ORB (IIOP) de CORBA.

1.5.4.1 Nginx

La plataforma de aplicaciones NGINX según define su sitio oficial (NGINX 2018) es un conjunto de productos que juntos forman el núcleo de lo que las organizaciones necesitan para crear aplicaciones con rendimiento, confiabilidad, seguridad y escala. La plataforma de aplicaciones NGINX incluye NGINX Plus para el equilibrio de carga y la entrega de aplicaciones, NGINX WAF para seguridad y la unidad NGINX para ejecutar el código de la aplicación, todo monitoreado y administrado por el controlador NGINX. Es el servidor web de código abierto que impulsa 400 millones de sitios web, único equilibrador de carga todo en uno, servidor web y caché de contenido.

1.5.4.2 Apache

El Proyecto de Servidor HTTP Apache es un esfuerzo por desarrollar y mantener un servidor HTTP de código abierto para sistemas operativos modernos, incluyendo UNIX y Windows. El objetivo de este proyecto es proporcionar un servidor seguro, eficiente y extensible que brinde servicios HTTP en sincronización con los estándares HTTP actuales. El servidor HTTP Apache ("httpd") se lanzó en 1995 y ha sido el servidor web más popular en Internet desde abril de 1996. Ha celebrado su vigésimo cumpleaños como proyecto en febrero de 2015. El servidor HTTP Apache es un proyecto de The Apache Software Foundation. Además, cientos de usuarios han aportado ideas, códigos y documentación al proyecto (The Apache Software Foundation 2018a).

Selección del servidor web

Se selecciona el servidor web Apache en su versión 2 (versión Ubuntu) por su potencia y su capacidad de configuración, además el sitio web oficial de Symfony [https://symfony.com/] recomienda el uso de este servidor en el desarrollo de aplicaciones y módulos. No obstante en caso de que se requiera puede utilizarse cualquier otro servidor de aplicaciones debido a la compatibilidad de Symfony, afirma (Potencier 2017).

1.5.5 Herramientas CASE

CASE es el acrónimo de *Computer Aided Software Engineering*, o ingeniería de software asistida por computadora, y se refiere al uso de programas de cómputo para organizar y administrar el desarrollo de software, sobre todo en aquellos proyectos donde se involucre una cantidad considerable de recursos y personal, tarea que para el gestor de proyectos representa una inversión considerable de tiempo, por tal motivo estas herramientas tienen como propósito el fungir como apoyo durante el ciclo de vida de todo el proyecto incluyendo todos sus componentes. El uso de estas herramientas también debe ser categorizado dentro del contexto del tipo de tecnología en la que se encuentre disponible la herramienta, así como por el propósito de su uso, para este trabajo considera la clasificación en función de la fase del ciclo de vida: *Upper* (administradores de proyectos), *middle* (herramientas de diseño) y *Lower* (generadores de código, repositorios, reporteadores) (Cerca et al. 2014).

1.5.5.1 Visual Paradigm

Visual Paradigm según define su sitio oficial (Visual Paradigm 2018) es una suite de gestión empresarial y desarrollo de software galardonada, líder a nivel mundial, que ofrece todas las funciones que se necesitan para la arquitectura empresarial, la gestión de proyectos, el desarrollo de software y la colaboración en equipo en una solución integral. Presenta software de diseño con UML, SysML, ERD, DFD y SoaML. Permite crear planos visuales de forma rápida y sin esfuerzo. Arquitectura empresarial con TOGAF ADM. Modelo con herramienta certificada ArchiMate 3.

1.5.6 NetBeans 8.2

NetBeans es un proyecto de código abierto dedicado a proporcionar productos de desarrollo de software sólidos (el IDE de NetBeans y la Plataforma de NetBeans) que atienden las necesidades de los desarrolladores, usuarios y las empresas. El IDE de NetBeans proporciona soporte para varios idiomas (PHP, JavaFX, C / C ++, JavaScript, etc.) y marcos. Proporciona analizadores y editores de código listos para usar para trabajar con las últimas tecnologías Java 8: Java SE 8, Java SE Embedded 8 y Java ME Embedded 8. El IDE también tiene una gama de nuevas herramientas para HTML5 / JavaScript, en

particular para Node.js, KnockoutJS y AngularJS; mejoras que mejoran aún más su soporte para Maven y Java EE con PrimeFaces; y mejoras al soporte de PHP y C / C ++ (Netbeans 2018).

1.5.7 Apache Jmeter

La aplicación Apache JMeter es un software de código abierto, una aplicación Java 100% pura diseñada para cargar el comportamiento funcional de la prueba y medir el rendimiento. Originalmente fue diseñado para probar aplicaciones web, pero desde entonces se ha expandido a otras funciones de prueba. Apache JMeter puede usarse para probar el rendimiento tanto en recursos estáticos como dinámicos, aplicaciones web dinámicas. Se puede usar para simular una carga pesada en un servidor, grupo de servidores, red u objeto para probar su resistencia o para analizar el rendimiento general bajo diferentes tipos de carga (The Apache Software Foundation 2018b).

1.5.8 Acunetix

Acunetix es el escáner de vulnerabilidad web líder utilizado por compañías serias de Fortune 500 y ampliamente reconocido por incluir la inyección de SQL más avanzada y la tecnología de escaneo de caja negra XSS. Rastrea automáticamente sus sitios web y realiza técnicas de *hacking* de caja negra Y caja gris que encuentran vulnerabilidades peligrosas que pueden comprometer su sitio web y sus datos. Acunetix realiza pruebas de Inyección SQL, XSS, XXE, SSRF, Inyección de encabezado de host y más de 4500 vulnerabilidades web. Cuenta con las más avanzadas técnicas de escaneo generando los menos falsos positivos posibles. Simplifica el proceso de seguridad de la aplicación web a través de sus características de administración de vulnerabilidades incorporadas que lo ayudan a priorizar y administrar la resolución de vulnerabilidades (Acunetix 2019).

1.6 Conclusiones Parciales

Como parte del desarrollo del presente capítulo se determinan las siguientes conclusiones parciales:

- El análisis de las diferentes herramientas y tendencias para la auditoría automática de las PSI permitió determinar las características que constituyen la base para el diseño de las funcionalidades que se definen en la propuesta de solución.

- El estudio sobre las metodologías, herramientas y lenguajes permitió definir los componentes base para el desarrollo de la solución, donde se define a AUP-UCI como metodología de desarrollo y como herramienta CASE Visual Paradigm 8.0 para el modelado de los artefactos del análisis y diseño de la solución. Se decide que el framework que se utilizará para la solución del problema sea Symfony en su versión 3.4.21. Se selecciona además el servidor web Apache 2.4 (versión Ubuntu), PHP 7.2, HTML 5, CSS 3, Bootstrap 4.3.1, JavaScript 5.0, jQuery 3.3.1, . Como sistemas gestores de base de datos MySQL 5.0.12. y como IDE de desarrollo NetBeans 8.2. Como herramientas de pruebas se utilizarán Apache Jmeter 5.0 y Acunetix 12.

Capítulo 2: Caracterización del sistema de auditoría automática de las PSI del CIDI

2.1 Introducción

En el presente capítulo se caracteriza la propuesta de solución a través de la definición de los requisitos funcionales y no funcionales, las tareas a realizar durante la implementación y la arquitectura y patrones utilizados en el desarrollo del sistema de auditoría automática de las PSI del CIDI. Además, se presentan los principales artefactos generados en las primeras fases del desarrollo.

2.2 Modelo del dominio

Un modelo de dominio, también conocido como modelo conceptual según (Larman 2002), es una representación visual en forma de diagrama de las clases conceptuales u objetos del mundo real que son significativos en un dominio de interés. No se trata de un conjunto de diagramas que describen clases u objetos de software con responsabilidades .

Descripción de clases del modelo del dominio:

Asesor de seguridad informática: Persona que realiza auditoría a las estaciones de trabajo.

Auditoría: Mecanismo que verifica el cumplimiento de las PSI.

Script en Bash: Lenguaje de linea de comandos.

PC: Computadora que pertenece al conjunto de activos del centro.

Laboratorio: Local donde se encuentran las estaciones de trabajo.

Terminal: Constituye la vista mediante la cual se ejecutan los Scripts en Bash y se obtienen los resultados.

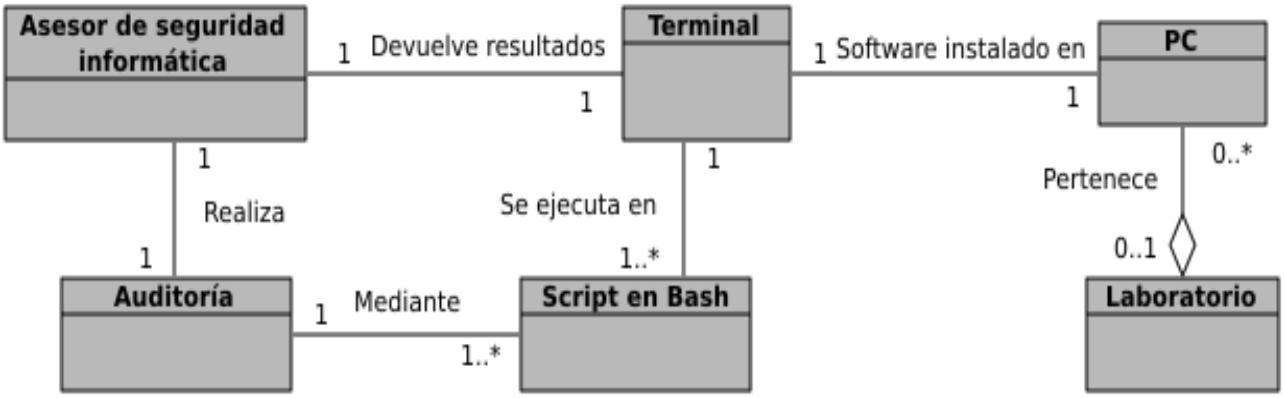


Figura 1: Modelo del dominio

2.3 Especificación de requisitos de software

En la ingeniería del software,define (Sommerville 2011), los requisitos se utilizan como datos de entrada en la etapa de diseño del producto y establecen qué debe hacer el sistema, pero no cómo hacerlo. Son una condición o capacidad que un usuario necesita para poder resolver un problema o lograr un objetivo. De manera general estos requisitos son lo que el sistema debe hacer o una cualidad que el sistema debe poseer.

2.3.1 Requisitos Funcionales

Los Requisitos Funcionales (RF) expresan la naturaleza del funcionamiento del sistema (cómo interacciona el sistema con su entorno y cuáles van a ser su estado y funcionamiento) (Sommerville 2011).

Las observaciones y encuestas realizadas arrojaron un total de 57 requisitos funcionales, que a continuación se presentan.

Tabla 1. Requisitos funcionales

Código	Descripción (Requisitos Funcionales)	Prioridad
RF1	Configurar rango de escaneo de IP.	Alta
RF2	Realizar escaneo de IP.	Alta
RF3	Mostrar progreso de escaneo de IP.	Media
RF4	Mostrar resultados de escaneo de IP.	Alta
RF5	Configurar auditoría a realizar.	Alta
RF6	Acceder remotamente a la PC.	Alta
RF7	Autenticarse remotamente en la PC.	Alta
RF8	Obtener privilegios de *root* en la PC.	Alta
RF9	Mostrar progreso de audición de la PC.	Media
RF10	Verificar usuarios con acceso.	Alta
RF11	Verificar instalación y configuración del cortafuegos.	Alta
RF12	Verificar IP.	Alta
RF13	Verificar dominio.	Alta
RF14	Verificar nombre de la PC	Alta
RF15	Verificar Veracrypt.	Alta
RF16	Verificar estado del antivirus.	Alta
RF17	Verificar código fuente.	Alta
RF18	Verificar actualizaciones pendientes.	Alta
RF19	Verificar cliente grhs.	Alta
RF20	Verificar reglas personalizadas.	Alta
RF21	Listar perfiles de auditoría.	Media
RF22	Crear perfiles de auditoría.	Media
RF23	Mostrar perfiles de auditoría.	Media
RF24	Editar perfiles de auditoría.	Media
RF25	Eliminar perfiles de auditoría.	Media
RF26	Listar reglas.	Media
RF27	Crear reglas.	Media
RF28	Mostrar reglas.	Media
RF29	Editar reglas.	Media

Código	Descripción (Requisitos Funcionales)	Prioridad
RF30	Eliminar reglas.	Media
RF31	Listar programas.	Media
RF32	Crear programas.	Media
RF33	Mostrar programas.	Media
RF34	Editar programas.	Media
RF35	Eliminar programas.	Media
RF36	Listar tareas programadas.	Media
RF37	Crear tareas programadas.	Media
RF38	Mostrar tareas programadas.	Media
RF39	Editar tareas programadas.	Media
RF40	Eliminar tareas programadas.	Media
RF41	Activar tareas programadas.	Media
RF42	Desactivar tareas programadas.	Media
RF43	Forzar ejecución de tareas programadas.	Media
RF44	Desbloquear tareas programadas.	Media
RF45	Generar reportes de auditoría.	Alta
RF46	Listar reportes de auditoría.	Alta
RF47	Eliminar reportes de auditoría.	Alta
RF48	Mostrar reportes de auditoría con listado de laboratorios.	Alta
RF49	Mostrar reportes de un laboratorio con listado de PC.	Alta
RF50	Eliminar reportes de un laboratorio.	Alta
RF51	Mostrar reportes de una PC.	Alta
RF52	Eliminar reportes de una PC.	Alta
RF53	Generar pdf del reporte de una PC.	Alta
RF54	Generar pdf de los reportes de un Laboratorio.	Alta
RF55	Generar pdf de un reporte.	Alta
RF56	Automatizar el proceso de auditoría mediante fechas programadas.	Alta
RF57	Notificar a los asesores de seguridad informática mediante correo electrónico.	Alta

2.3.2 Requisitos no Funcionales

Los requisitos no funcionales (RNF) son propiedades que hacen al producto atractivo, usable, rápido, seguro y confiable. Son los que otorgan el acabado al producto para iniciar su uso (Sommerville 2011).

Se identificaron un total de 14 requisitos no funcionales, agrupados en 6 categorías que a continuación se presentan.

Requisitos de entorno:

- RNF 1. Se requiere la instalación de un servidor web como Apache o Nginx en cualquiera de sus versiones y PHP 7.0.1 o superior para poder visualizar la interfaz web de Symfony en su versión 3.4.21.

- RNF 2. Se requiere que las PC tengan instalada, alguna versión de la distribución GNU/LINUX y puedan ejecutar código Bash.

- RNF 3. Se requiere que las PC posean las configuraciones de SSH 2 que permitan la conexión y la ejecución de comandos remotos.

- RNF 4. Se requiere que la PC donde se encuentren ejecutando los servicios del servidor de aplicaciones web, se conecte al resto de las PC mediante llaves públicas y privadas y posea las configuraciones de Cron necesarias para ejecutar las tareas programadas.

- RNF 5. Los asesores de seguridad que utilizarán la herramienta deben contar con navegadores web que soporten HTML5, CSS3 y Javascript.

Requisitos de hardware:

- RNF 6. Para la interfaz web: 2 GB RAM, CPU de 2 núcleos y al menos 10 GB de Disco Duro.

- RNF 7. Para el servidor de base datos MySQL: 2 GB RAM, CPU de 2 núcleos y al menos 10 GB de Disco Duro.

Requisitos de diseño e implementación:

- RNF 8. Para el desarrollo de la aplicación web se deberá utilizar Symfony 3.4.21 como marco de trabajo y MySQL 5.0.12 u otra versión similar, como sistema gestor de base datos.

Requisitos de eficiencia

- RNF 9. El tiempo y esfuerzo asociado al proceso de auditoría debe disminuir.

Requisitos de usabilidad:

- RNF 10. El sistema debe proporcionar mensajes de error que sean informativos y orientados al usuario final.

- RNF 11. El sistema debe contar con manuales de usuario estructurados adecuadamente.

Requisitos de seguridad:

- RNF 12. Solo los asesores de seguridad podrán usar la herramienta con todas las funcionalidades y configuraciones.

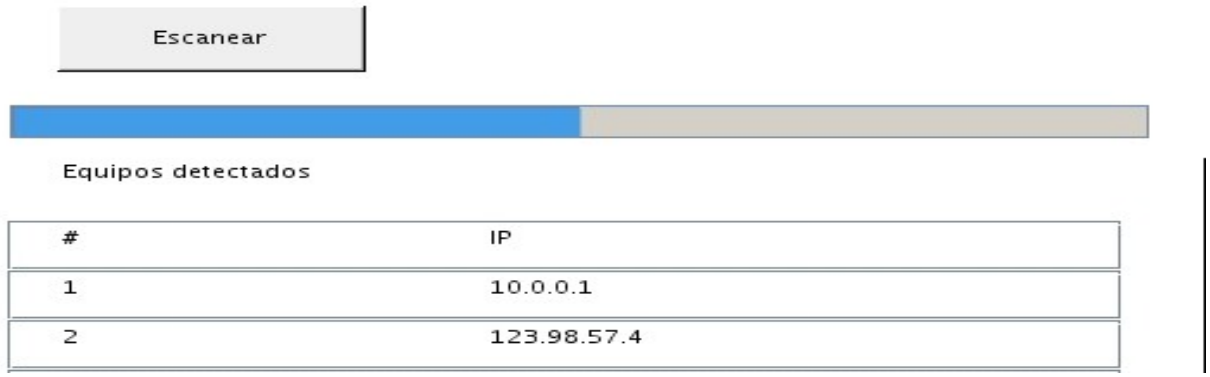
- RNF 13. Se requiere que las PC posean las configuraciones de SUDO y PAM que garanticen los privilegios *root* al asesor de seguridad informática.

- RNF 14. Se requiere que las PC posean las configuraciones de red y cortafuegos que garanticen el acceso desde la PC donde se encuentre montado el servidor de aplicaciones web.

2.4 Historias de usuario

La Historia de Usuarios (HU) se utilizan para registrar los requerimientos de los clientes según el negocio y son utilizadas para poder realizar la estimación de cada una de las iteraciones durante la fase de planificación. Las HU son escritas por el equipo de trabajo en conjunto con los clientes en base a lo que se estima que es necesario para el sistema (Penadillo et al. 2013).

Tabla 2. Historia de usuario #1

Historia de usuario	
Número: HU_1	**Nombre:** Realizar escaneo de IP.
Prioridad en negocio: Alta	
Descripción: El sistema escanea la red y lista los resultados con los IP activos detectados.	
Prototipo: 	

2.5 Estilo arquitectónico

Para el desarrollo del sistema de auditoría de las PSI del CIDI se empleará como marco de trabajo Symfony, que utiliza el patrón arquitectónico Modelo-Vista-Controlador (MVC), usado por la mayoría de los marcos de trabajo web. No obstante, según su creador Fabien Potencier, “Symfony no es un marco de trabajo MVC. Este marco de trabajo sólo proporciona herramientas para la parte del Controlador y de la Vista. La parte del Modelo es responsabilidad del usuario” (Potencier 2017).

El uso de la arquitectura MVC en el desarrollo de la solución propuesta estableció una división lógica en tres elementos principales. El modelo de datos, la presentación o interfaz de usuario (vistas) y la lógica de negocio (controladores). En la siguiente imagen se aprecia que un usuario accede a un documento HTML generado por Symfony, donde se realizan acciones desde la vista llamando al controlador. Este ultimo interactúa con el modelo para insertar u obtener valores de la base de datos, los cuales se procesan y son enviados a la vista.

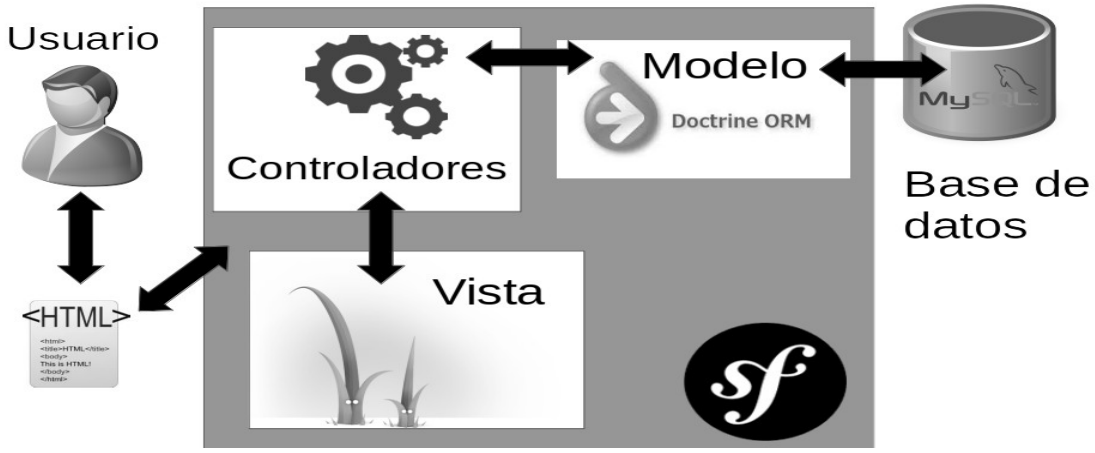


Figura 2: Arquitectura del sistema

2.6 Patrones de diseño

Un patrón de diseño según (Goñi's 2018) es una descripción de clases y objetos comunicándose entre sí, adaptada para resolver un problema de diseño general en un contexto particular. El mismo identifica clases, instancias, roles, colaboraciones y la distribución de responsabilidades.

Los Patrones Generales de Software para Asignar Responsabilidades (GRASP del inglés *General Responsibility Assignment Software Patterns*) tienen gran utilidad en el diseño de aplicaciones, al igual que la Pandilla de los Cuatro (GoF por sus siglas en inglés, *Gang-of-Four*). A continuación, se muestra una selección de estos patrones, los cuales serán utilizados durante el diseño del módulo:

Experto

Este patrón tiene como objetivo principal otorgar una responsabilidad dada a la clase que tenga la mayor cantidad de información para hacer esta tarea. Es la razón anterior la que le da su apellido Experto “en información”. Este patrón se evidencia en la clase PCController, es quien tiene acceso a todas las entidades necesarias y por ello a la información, por lo cual se le asigna la responsabilidad de generar todos los reportes de auditoría que se requieran.

Creador

Se aplica para la asignación de responsabilidades a las clases relacionadas con la creación de objetos, de forma tal que una instancia de un objeto sólo pueda ser creada por el objeto que contiene la información necesaria para ello. En este caso el patrón se refleja en varias clases como por ejemplo en BashRepository encargadas de crear las entidades de Bash, datos que serán usados luego por BashController.

```
<?php
namespace Cidi\SapsiBundle\Repository;
class BashRepository extends \Doctrine\ORM\EntityRepository
{
    public function findOneBynombreCodigo($nombreCodigo)
    {
        $query = $this->getEntityManager()->createQuery(
            'SELECT bash FROM Cidi:SapsiBundle:Bash bash
            WHERE bash.nombreCodigo = :nombreCodigo'
        )->setParameter('title', $nombreCodigo);
    }
}
```

Bajo Acoplamiento

El acoplamiento es una medida de la fuerza en que una clase está conectada a otras, que la conoce y recurre a ellas. El objetivo de este patrón consiste en mantener un bajo nivel de dependencia de otros elementos, por lo que constituye un principio que debe estar presente en todas las decisiones de diseño con lo que se reduce el impacto de los cambios. Esto se evidencia en la clase Bash que se encuentra relacionada con Perfil pero no con otras clases del sistema, como se muestra en BashController:

```
<?php
namespace Cidi\SapsiBundle\Controller;
use Cidi\SapsiBundle\Entity\Bash;
use Symfony\Bundle\FrameworkBundle\Controller\Controller;
use Cidi\SapsiBundle\Entity\Perfil;
use Sensio\Bundle\FrameworkExtraBundle\Configuration\Method;
use Sensio\Bundle\FrameworkExtraBundle\Configuration\Route;
use Symfony\Component\DependencyInjection\ContainerInterface;
use Symfony\Component\HttpFoundation\Request;
use Symfony\Component\HttpFoundation\Response;
use Cidi\SapsiBundle\Form\BashType;
//..
```

Alta Cohesión

La cohesión es una medida de cuán relacionadas y enfocadas están las responsabilidades de una clase. Se puede afirmar que cada una de las clases del sistema tiene alta cohesión, de manera que estas poseen la característica de tener las responsabilidades estrechamente relacionadas. Esta propiedad evita en cada caso, tener que realizar un trabajo enorme al garantizar un mejor diseño en ocasiones para el resultado global. Se tiene por ejemplo la clase PC que está relacionada con Laboratorio como se muestra en PCController.php:

```
<?php
namespace Cidi\SapsiBundle\Controller;
use Cidi\SapsiBundle\Entity\Laboratorio;
use Symfony\Component\HttpFoundation\Response;
use Cidi\SapsiBundle\Entity\PC;
use Symfony\Bundle\FrameworkBundle\Controller\Controller;
use Symfony\Component\HttpFoundation\Request;
//..
```

Controlador

Para este caso es necesario conocer que un evento del sistema es una operación que se realiza en este, por la acción de un usuario externo. Un controlador, es un objeto de interfaz que no está destinada al usuario y se ocupa de manejar un evento del sistema. Determina también el método de su operación. En esta solución se encuentra por ejemplo, la clase BashController que es la encargada de gestionar todas las reglas de auditoría y otras acciones necesarias:

```
<?php
class BashController extends Controller {
   public function indexAction() {...}
   public function newAction(Request $request) {...}
   public function showAction(Bash $bash) {...}
   public function editAction(Request $request, Bash $bash) {...}
   public function deleteAction(Request $request, Bash $bash) {...}
   }
```

Decorador

Forma parte de la familia de patrones conocidos como estructurales. Este tipo de patrones describen como las clases y objetos pueden ser combinados para formar grandes estructuras y proporcionar nuevas funcionalidades. Decorador permite modificar, retirar o agregar responsabilidades a un objeto dinámicamente.

La ventaja es que permite extender objetos aún en situaciones donde la extensión vía herencia no se requiere. Este patrón se evidencia en las entidades PC, Laboratorio y Bash, ya que mediante YAML en los archivos de doctrine “orm.yml” se definen los atributos de las clases y se puede especificar la columna de la base de datos en la cual será guardado como se muestra en el siguiente ejemplo en el archivo Bash.orm.yml:

```
<?php
Cidi\SapsiBundle\Entity\Bash:
    type: entity
    table: null
    repositoryClass: Cidi\SapsiBundle\Repository\BashRepository
    id:
        id:
            type: integer
            id: true
            generator:
                strategy: AUTO
    fields:
        nombreCodigo:
            type: string
            column: Nombre_codigo
        codigo:
            type: text
            length: 255
```

column: Codigo
lifecycleCallbacks: { }

Registro. Controlador Frontal

Es uno de los patrones de diseño más conocido de Symfony. Es una sección de código que atiende todas las solicitudes en la aplicación y devuelve una respuesta al navegador. Este controlador se utiliza además para direccionar todas las peticiones, es el encargado de iniciar el núcleo del sistema lo que permite decorarlo con características adicionales. Uno de los problemas que evita es la duplicación de código, ya que define un método comando en una clase abstracta, que luego se podrá redefinir en las demás clases según las especificaciones que se necesiten, con el uso de decoradores. Se evidencia en las URL donde todas parten del nodo app_dev.php. Tenemos como ejemplo de uno de estos casos:

http://localhost/SAPSI-CIDI/web/app_dev.php/herramienta.

2.7 Modelo de Diseño

El modelo de diseño se utiliza como medio de abstracción del modelo de implementación y el código fuente del software. Su objetivo fundamental es transmitir, a través de la representación mediante diagramas, una comprensión en profundidad de los aspectos relacionados con los requerimientos no funcionales y restricciones concernientes a los lenguajes de programación (Larman 2002).

En la siguiente figura se muestra el diagrama de clases de diseño pertenecientes al sistema de auditoría de las PSI del CIDI en el cual se puede apreciar las relaciones de dependencia entre los distintos componentes de Symfony y el Núcleo del Sistema Operativo GNU/LINUX necesarias para el funcionamiento de la solución propuesta.

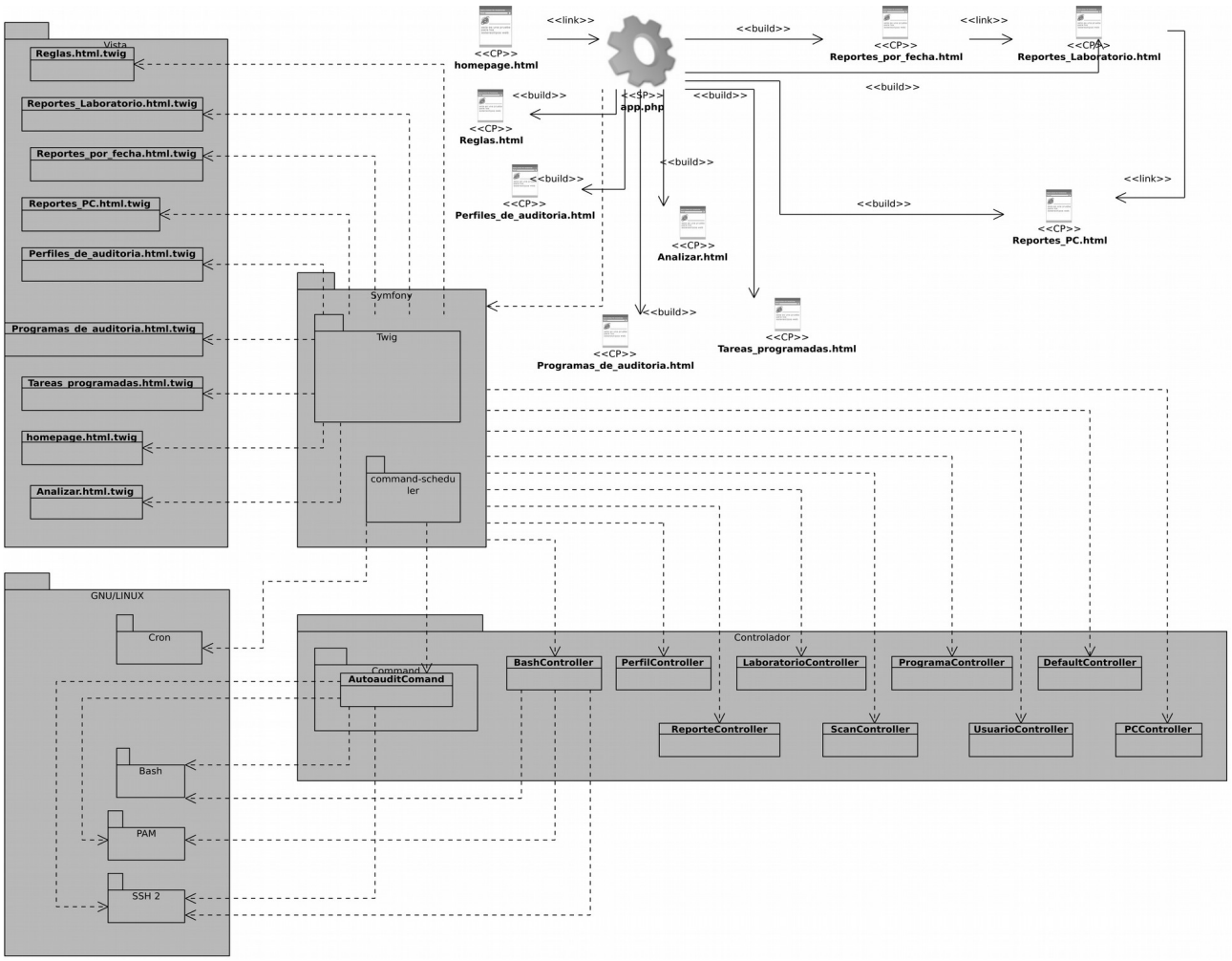


Figura 3: Diagrama de clases del diseño del sistema de auditoría de las PSI del CIDI

2.8 Modelo de datos

Un modelo de datos es un sistema formal y abstracto que permite describir los datos de acuerdo con reglas y convenios predefinidos o se podría decir que es un conjunto de conceptos que permiten describir, a distintos niveles de abstracción, la estructura de una base de datos (Redondo y Jesús 2017).

Principales características de un modelo de datos:

- Independencia lógica y física de los datos.
- Redundancia mínima
- Respaldo y recuperación
- Seguridad de acceso y auditoría
- Integridad de los datos
- Consultas complejas optimizadas.

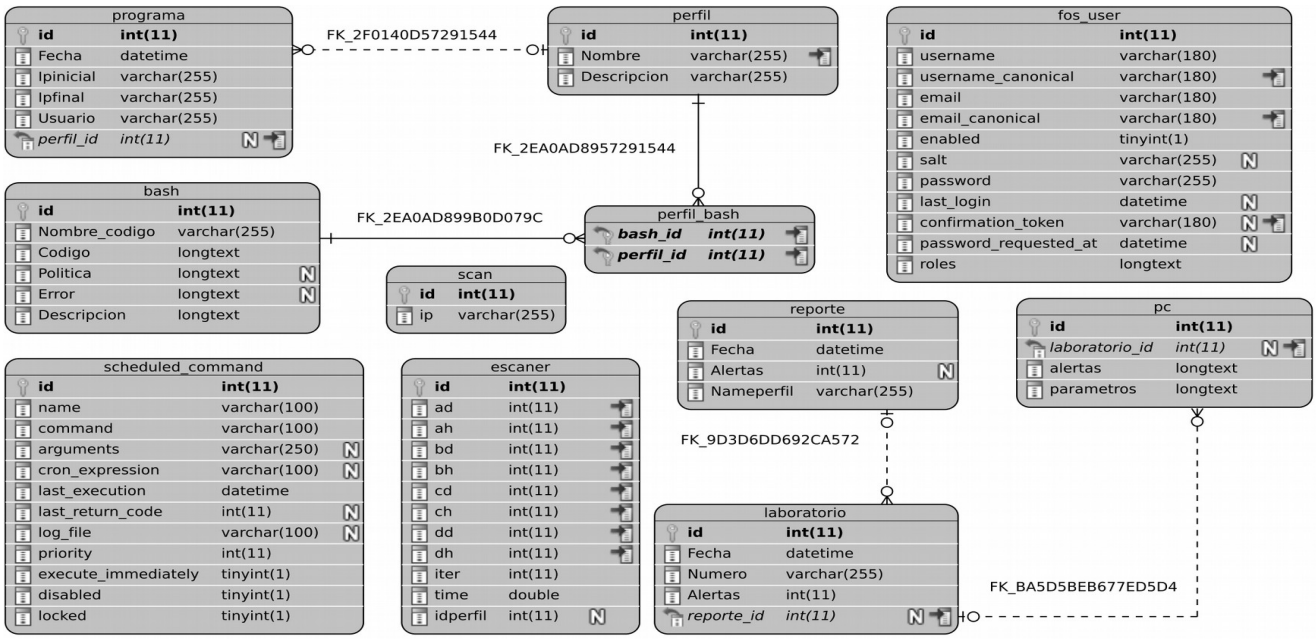


Figura 4: Modelo de datos

En la imagen anterior se aprecia el modelo de datos del sistema de auditoría de las PSI del CIDI el cual presenta 11 tablas y relaciones OneToMany, ManyToOne y ManyToMany para asociar los elementos.

2.9 Diagrama de despliegue

El diagrama de despliegue se utiliza para mostrar la estructura física del sistema, incluyendo las relaciones entre el hardware y el software que se despliega, estas relaciones son representadas por los protocolos de comunicación que se utilizan para acceder a cada uno, publica (Sparx Systems 2019). En la siguiente figura puede visualizarse el diagrama de despliegue definido para la solución propuesta:

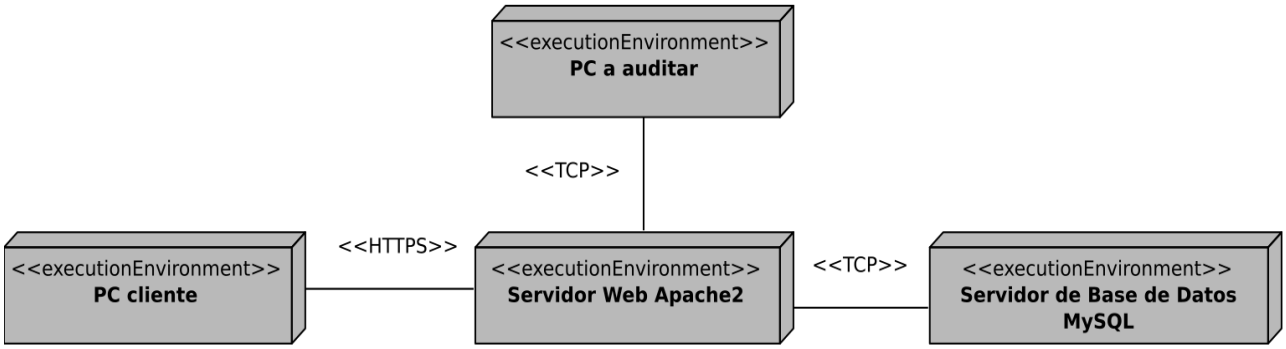


Figura 5: Diagrama de despliegue

2.10 Conclusiones del capítulo

Con la investigación antes expuesta se arribó a las siguientes conclusiones:

- Con la especificación de los requisitos funcionales y no funcionales del sistema, se logró una mejor comprensión, de los resultados previstos a obtener de una manera precisa y sirven de guía para la implementación del sistema.

- La representación y descripción de los artefactos generados garantizaron un mejor entendimiento de los flujos de trabajos presentes en el proceso de generar los reportes de auditorías.

- El uso de los patrones de diseño permitió identificar elementos importantes de la estructura del diseño del sistema web propuesto, lo que garantizó una mayor organización e hizo el código más legible.

- La elaboración del diagrama de despliegue permitió identificar la disposición física de los artefactos del sistema de auditoría de la PSI del CIDI.

Capítulo 3: Construcción y pruebas del sistema de auditoría de las PSI del CIDI

3.1 Introducción

En el presente capítulo se describen los diferentes mecanismos utilizados para llevar a cabo el desarrollo y validación del sistema propuesto. Así como los estándares de codificación que se debe seguir para implementar el software para un mejor entendimiento y organización del mismo. Luego se realiza la fase de prueba, cuyo objetivo es comprobar si el sistema cumple con los requerimientos establecidos.

3.2 Modelo de implementación

El modelo de implementación es comprendido por un conjunto de componentes y subsistemas que constituyen la composición física de la implementación del sistema. Fundamentalmente, se describe la relación que existe desde los paquetes y clases del modelo de diseño a subsistemas y componentes físicos (Hernández 2013).

3.3 Diagrama de componentes

Un componente es una parte física de un sistema (módulo, base de datos, programa ejecutable). Se puede decir que es la materialización de una o más clases (Hernández 2013).

El diagrama de componentes permite concebir el diseño según los bloques principales y ayuda a entender un diseño existente y crear uno nuevo. Al establecer el sistema como una colección de componentes con interfaces proporcionadas y necesarias bien definidas se garantiza la correcta separación entre los componentes. A su vez, se facilita la comprensión de los cambios al modificar los requisitos. A continuación se presenta el diagrama de componentes correspondiente al sistema de auditoría de las PSI del CIDI donde se aprecian los principales paquetes y componentes necesarios para su funcionamiento.

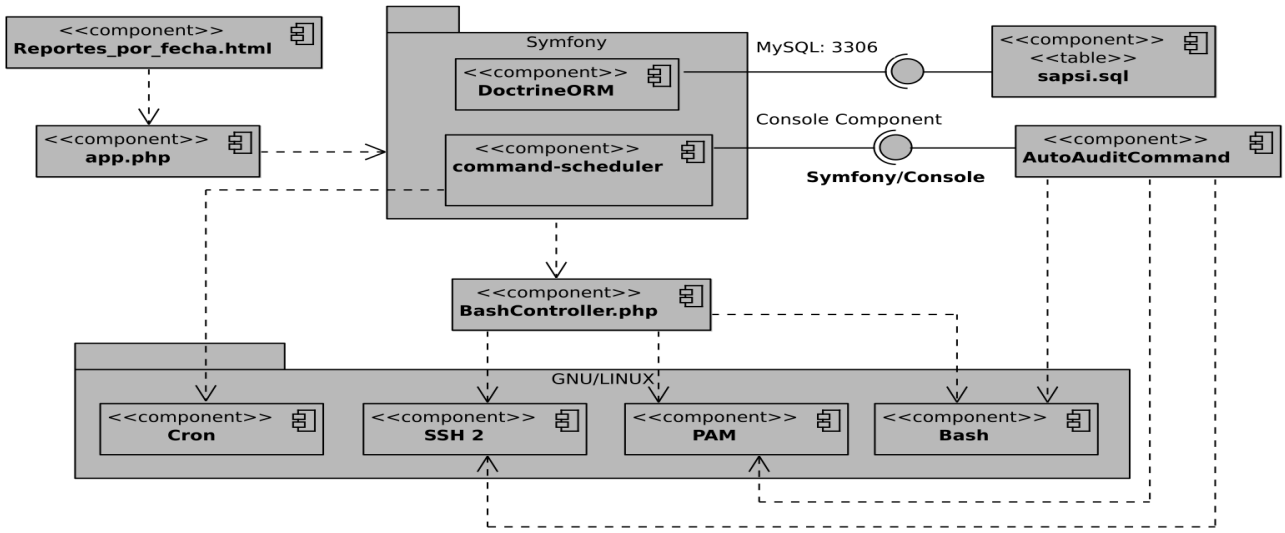


Figura 6: Diagrama de componentes

3.4 Estándares de codificación utilizados

Un estándar de codificación completo comprende todos los aspectos de la generación de código. Un código fuente completo debe reflejar un estilo armonioso, como si un único programador hubiera escrito todo el código de una sola vez. El estándar de codificación debería establecer cómo operar con la base de código existente (Reyes 2019). A continuación se especifican los estándares de codificación a utilizar en la construcción de la solución:

- El tamaño máximo de las líneas de código debe ser de cien o menos caracteres aproximadamente, mientras sea posible, de manera tal que se garantice la completa visibilidad de las líneas de código sin necesidad de realizar desplazamiento horizontal.

```
public function formAction()
  {
```

```
        $bash = new Bash();
        $form = $this->createForm(BashType::class, $bash);
        return $this->render('bash/form.html.twig', array(
            'form' => $form->createView()
        ));
    }
```

- Los nombres de las clases y las funciones adoptarán la notación lowerCamelCase y no se utilizará el guión bajo como delimitador entre palabras.

  ```
  public function scanAction() { … }
  ```

- Los nombres de los atributos, variables y parámetros tendrán todas las letras en minúsculas y no se usarán delimitadores entre palabras.

  ```
  $scan = shell_exec('for i in $(seq 1 1 253) ; do if ping 10.53.8.$i -i 0.2 -c 1 -W 1 > /dev/null; then echo 10.53.8.$i,;fi;done');
  $newscan = substr($scan, 0, -2);
  ```

3.5 Validación del sistema

En la ingeniería de software la validación es el proceso de revisión que verifica que el sistema de software producido cumple con las especificaciones y que logra su cometido, se trata de evaluar el sistema o parte de este durante o al final del desarrollo para determinar si satisface los requisitos iniciales (Junta de Andalucía 2019).

3.5.1 Pruebas de rendimiento

Mediante la herramienta Apache Jmeter, se le realizaron análisis al sistema para evaluar su rendimiento, para un uso simultaneo de 50 usuarios, corriendo en Apache, sobre Linux Mint 19, en una PC con

microprocesador core i3 a 3.60 GHz y 4 GB de RAM DDR3. En la siguiente imagen se muestran los resultados generados por Apache JMeter:

Etiqueta	# Muestras	Media	Mediana	Linea de 90%	Mín	Máx	% Error	Rendimiento	Kb/sec
/SAPSI-CIDI/web/app.php/	100	47	30	127	10	175	0,00%	81,9/sec	228,4
/SAPSI-CIDI/web/app.php/documentacion	50	27	27	41	11	39	0,00%	41,1/sec	87,5
/SAPSI-CIDI/web/app.php/acercade	50	24	23	36	10	47	0,00%	40,9/sec	53,9
/SAPSI-CIDI/web/app.php/herramienta	50	27	27	39	11	53	0,00%	40,5/sec	67,9
/SAPSI-CIDI/web/app.php/perfil/	50	50	49	70	25	85	0,00%	40,1/sec	72,7
/SAPSI-CIDI/web/app.php/bash/	50	48	47	68	19	91	0,00%	40,4/sec	100,4
/SAPSI-CIDI/web/app.php/herramienta/analiz...	50	21	20	33	9	38	0,00%	40,8/sec	68,7
Total	400	36	30	64	9	175	0,00%	294,8/sec	614,9

Figura 7: Resultados de las pruebas de rendimiento (Generado por Apache Jmeter)

Se evidencia que el tiempo de respuesta máximo es de 175 milisegundos y el mínimo de 9 milisegundos, para un tiempo de respuesta óptimo de la aplicación ante carga y estrés. Estos resultados se encuentran acorde a las métricas de calidad de la UCI las cuales establecen un tiempo máximo de 5 segundos.

3.5.2 Pruebas funcionales

Una prueba funcional es una prueba basada en la ejecución, revisión y retroalimentación de las funcionalidades previamente diseñadas para el software. Las pruebas funcionales se hacen mediante el diseño de modelos de prueba que buscan evaluar cada una de las opciones con las que cuenta el paquete informático. Dicho de otro modo son pruebas específicas, concretas y exhaustivas para probar y validar que el software hace lo que debe y sobre todo, lo que se ha especificado. El objetivo final de esta prueba es garantizar que los requerimientos hayan sido cumplidos y que el sistema goce de buena calidad

(Globe Testing 2019c). A continuación, se muestra una parte de la propuesta de diseño de casos de prueba para algunos requisitos del sistema de auditoría de las PSI del CIDI.

Las celdas de las tablas contienen V, I, N/A. V indica válido, I indica inválido, y N/A indica que no es necesario.

Tabla 3. Caso de prueba # 1

Escenario	Descripción	Acceso	Respuesta del sistema	Flujo central
EC 1.1 Autenticarse en el sistema.	El usuario accede a la página de acceso y se autentica.	V dariem ********** ymarrero ***************	Si los datos son correctos y presenta los permisos necesarios, el usuario se autentica, es redirigido a la página de inicio y se le muestra el enlace a la herramienta. En caso contrario permanece en la página de acceso.	El usuario accede a la pagina de acceso e inserta sus credenciales. El sistema comprueba los datos y si son correctos, autentica al usuario, lo redirecciona a la pagina de inicio y muestra el enlace a la herramienta.
EC 1.2 Autenticarse en el sistema.	El usuario no accede a la página de acceso y no se autentica o inserta credenciales erróneas.	I usuario ********* (usuario o contraseña incorrectos)	El sistema no muestra el enlace a la herramienta o el usuario permanece en la página de acceso, no se autentica y no se muestra el enlace a la herramienta.	El usuario no accede a la página de acceso y no se autentica o inserta credenciales erróneas. El sistema no autentica al usuario y no muestra el enlace a la herramienta.

Tabla 4. Variables empleadas en el diseño del caso de prueba # 1

No	Nombre de campo	Clasificación	Valor nulo	Descripción
1	Acceso	Seguridad	No	Permite autenticarse en el sistema y que se muestre el enlace a la herramienta

Resultados de las pruebas funcionales

Para probar el correcto funcionamiento del sistema se realizaron cuatro iteraciones de pruebas. En la tabla que a continuación se presenta, se muestran los resultados obtenidos en cada iteración de pruebas al sistema de auditoría de las PSI del CIDI, así como la corrección de cada uno de los errores.

Tabla 5. Cantidad de no conformidades por cada iteración de las pruebas

No conformidades	Primera iteración	Segunda iteración	Tercera iteración	Cuarta iteración
Detectadas	12	4	1	0
Resueltas	12	4	1	0
Pendientes	0	0	0	0

En la tabla 5 se muestra el comportamiento de las no conformidades encontradas durante el proceso de prueba donde se observa que en la primera iteración se detectaron doce (12) no conformidades, de ellas, cinco (5) errores de interfaz, cuatro (4) validaciones incorrectas y tres (3) errores ortográficos. Tras la segunda iteración se detectaron cuatro (4) no conformidades, de ellas, un (1) error de interfaz, dos (2) validaciones incorrectas y un (1) error de autenticación. Luego en la tercera iteración se detectó solo una (1) no conformidad referida a un error de autenticación Finalmente en la cuarta iteración no se detectaron no conformidades, cumpliendo con los requerimientos del cliente y su satisfacción.

3.5.3 Pruebas de seguridad

Las pruebas de seguridad buscan medir la confidencialidad, integridad y disponibilidad de los datos, desde la perspectiva del aplicativo, es decir partiendo a identificar amenazas y riesgos desde el uso o interfaces de usuario final. Una vez ejecutadas las pruebas de seguridad es posible medir y cuantificar los riesgos a los cuales se ven expuestos los aplicativos tanto en la infraestructura interna como externa. Entre los tipos de prueba se encuentran pruebas de caja negra para atacar utilizando las mismas técnicas y métodos de ataque que un *hacker* utilizaría, pruebas de caja blanca para hacer una revisión a fondo del sistema, contando con información detallada del entorno, incluido código fuente y archivos de configuración y

pruebas de caja gris la cual es una combinación de las pruebas de caja negra y caja blanca, disponiendo de cierta información sobre los sistemas y utilizándola en ocasiones (Globe Testing 2019a).

Resultados de las pruebas de seguridad

Para garantizar la seguridad del sistema de auditoría de las PSI del CIDI fue utilizada la herramienta Acunetix en su versión 12.0 donde se detectaron 12 alertas en total, de ellas 4 de clasificación media, 3 de clasificación baja y 5 de información. En la primera iteración se encontraron cuatro (4) envíos de contraseñas en texto plano, dos (2) formularios con autocompletamiento de contraseñas guardadas, dos (2) formularios con envió de datos con el método GET y cuatro (4) errores de configuración del servidor de aplicaciones web como se muestra en la siguiente imagen.

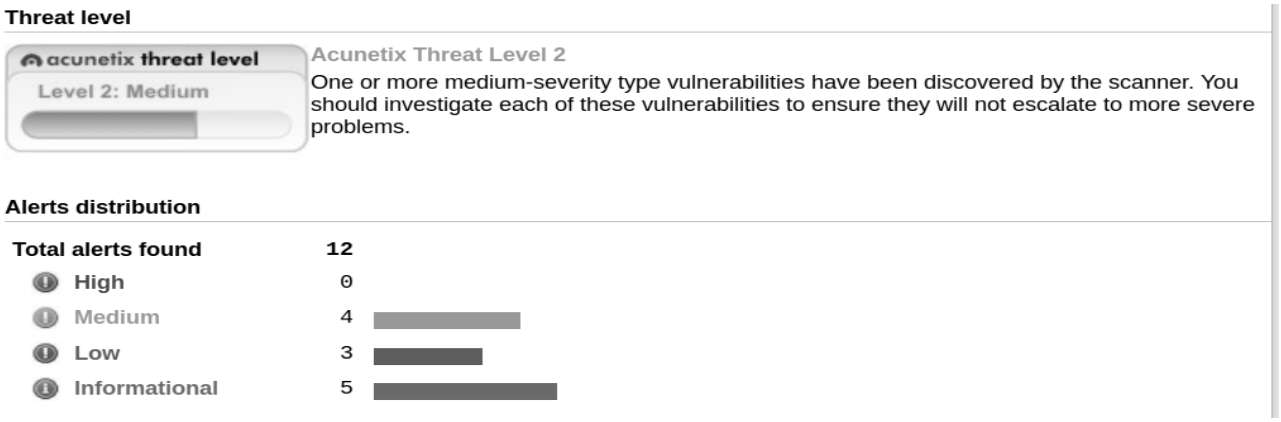


Figura 8: Análisis de seguridad (Generado por Acunetix)

Después de rectificar los errores detectados se realizó el análisis rectificándose por completo las vulnerabilidades detectadas como se muestra en la siguiente imagen.

Figura 9: Análisis de seguridad después de rectificar errores (Generado por Acunetix)

3.5.4 Pruebas de usabilidad

En el contexto del desarrollo de software, la usabilidad está considerada como uno de los factores de calidad de mayor importancia para el éxito de un proyecto. De manera general, el término usabilidad es empleado para referirse a la capacidad que posee un producto de ser utilizado por los usuarios de forma fácil, eficiente y con satisfacción, en un determinado contexto de uso (Globe Testing 2019b). Se le aplicó la lista de chequeo UCI al sistema de auditoría de las PSI del CIDI. Dicha lista está desarrollada por el centro de calidad UCI.

Tabla 6. Resultados de las pruebas de usabilidad

Categoría de los indicadores	Indicadores aplicables	Correctos
Visibilidad del sistema	15	13
Lenguaje común entre sistema y usuario	10	6
Libertad y control por parte del usuario	18	11
Consistencia y estándares	26	22
Estética y diseño minimalista	10	10
Prevención de errores	6	6
Ayuda a los usuarios a reconocer, diagnosticar y recuperarse de los errores	9	7
Ayuda y documentación	6	5
Flexibilidad y eficiencia de uso	4	4
Total	104	84

En la tabla anterior se evidencia que la implementación del sistema de auditoría de las PSI del CIDI cumple 84 de los 104 indicadores de usabilidad aplicables, lo cual representa el 80,77% aproximadamente. De las 20 no conformidades se clasifican en dos (2) de visibilidad del sistema, cuatro (4) de lenguaje común entre sistema y usuario, siete (7) de libertad y control por parte del usuario, cuatro (4) de consistencia y estándares, dos (2) de ayuda a los usuarios a reconocer, diagnosticar y recuperarse de errores y una (1) de ayuda y documentación.

Luego de analizar los resultados obtenidos en las pruebas de usabilidad, se identificaron que los 20 indicadores tenían posibles mejoras de acuerdo al alcance del presente trabajo. Con esto se logró un incremento en el nivel de usabilidad hasta de un 19.23% aproximadamente. En la siguiente gráfica se representa el estado del nivel de usabilidad resultante tras rectificar las no conformidades.

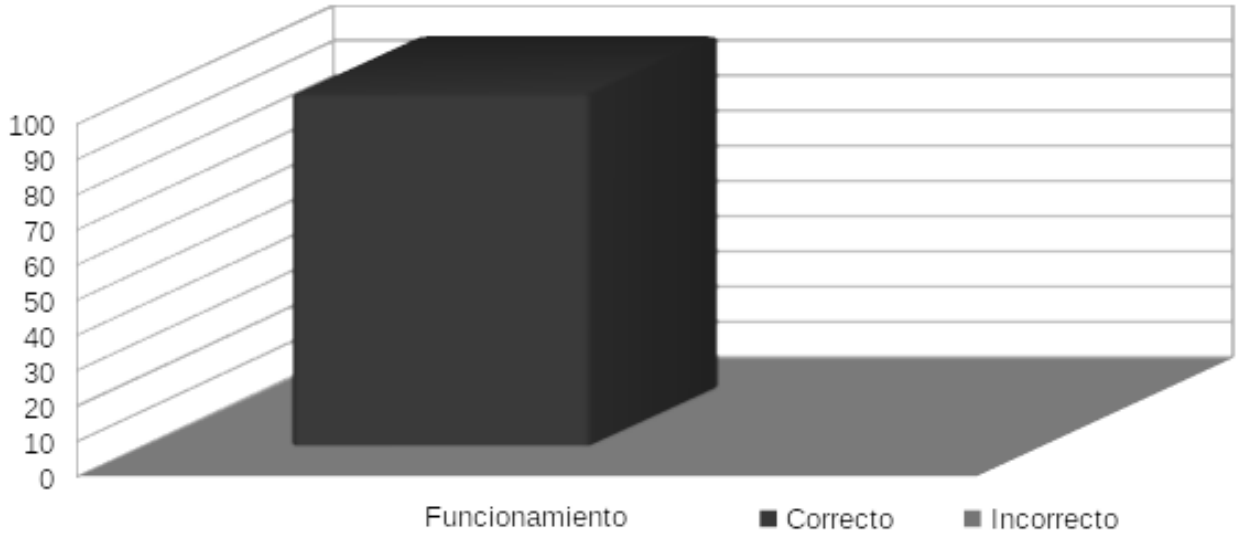


Figura 10: Diagrama de usabilidad

3.6 Conclusiones del capítulo

Con la investigación antes expuesta se concluye que:

- En el presente capítulo se especificó cómo está construido el sistema a partir del diagrama de componentes, lo cual permitió identificar con claridad la estructura y relaciones que existen entre los diferentes componentes empleados en la implementación del sistema.

- La utilización de estándares de código para la implementación de la propuesta de solución posibilitó adoptar una estructura homogénea que facilita la comunicación y una menor cantidad de errores, logrando un código más limpio y fácil de mantener.

- Las pruebas realizadas permitieron detectar los errores presentes, corregirlos en el menor tiempo posible y entregar al cliente una aplicación con mayor calidad, seguridad y usabilidad.

Conclusiones Generales

Una vez completada la presente investigación, se puede concluir que:

- El estudio realizado del sistema de homólogos, evidenció la necesidad de implementar un sistema de auditoría de las PSI del CIDI definiéndose una propuesta de solución de acuerdo a las necesidades existentes.

- La selección de herramientas, lenguajes y tecnologías permitió la implementación del sistema de auditoría de las PSI del CIDI.

- La utilización de la estrategia de pruebas garantizó la identificación temprana de las deficiencias en el sistema que se desarrolló; corrigiéndose y logrando un producto más seguro y funcional.

- El desarrollo del sistema de auditoría de las PSI del CIDI contribuyó a la disminución del tiempo y esfuerzo asociado al proceso de auditoría de las PSI en CIDI.

Recomendaciones

- Implementar las funcionalidades de modificar las PSI de las PC desde la aplicación web, para que cumplan con las mismas y automatizar dicho proceso durante la propia auditoría, mientras sea posible.

- Implementar la detección de sistemas operativos Windows y GNU/Linux para auditar según el sistema instalado en las PC y mejorar la seguridad.

- Generalizar el sistema para que sea desplegado para la Universidad de las Ciencias Informáticas y después de validar su calidad, a otras instituciones del país.

Bibliografía

1. ACUNETIX, 2019. Website security - keep in check with Acunetix. [en línea]. [Consulta: 7 mayo 2019]. Disponible en: https://www.acunetix.com/.

2. GOÑI'S , Alfredo, 2018. *Reutilización del Software. Patrones de Diseño* [en línea]. 2018. S.l.: s.n. Disponible en: http://siul02.si.ehu.es/%7Ealfredo/iso/06Patrones.pdf. 2019-02-14 12:06:49

3. PENADILLO, Balarezo, et al, 2013. *Metodologías Ágiles. Programación extrema XP.* [en línea]. 2013. S.l.: Universidad nacional de Trujillo. [Consulta: 15 febrero 2019]. Disponible en: https://es.slideshare.net/EvelingGiselleCruzVs/metodologia-monografia#_blank.

4. BOOTSTRAP, 2018. Bootstrap. [en línea]. [Consulta: 11 noviembre 2018]. Disponible en: https://getbootstrap.com/.

5. CHECKMARX, 2018. Checkmarx – Application Security Testing and Static Code Analysis. *Checkmarx* [en línea]. [Consulta: 11 noviembre 2018]. Disponible en: https://www.checkmarx.com/.

6. LARMAN, Craig , 2002. *UML y Patrones* [en línea]. 2da edición. España: PRENTICE HALL. [Consulta: 14 febrero 2019]. ISBN 84-205-3438-2. Disponible en: http://www.fmonje.com/UTN/ADES%20-%20208/UML%20y%20Patrones%20%202da %20Edicion.pdf.

7. DJANGO, 2018. The Web framework for perfectionists with deadlines | Django. [en línea]. [Consulta: 11 noviembre 2018]. Disponible en: https://www.djangoproject.com/.

8. POTENCIER, Fabien, 2017. Symfony, High Performance PHP Framework for Web Development. [en línea]. [Consulta: 5 noviembre 2018]. Disponible en: https://symfony.com/.

9. CASTRO R., Fidel, 1999. *CONSEJO DE ESTADO. DECRETO-LEY No.199 SOBRE LA SEGURIDAD Y PROTECCIÓN DE LA INFORMACIÓN OFICIAL* [en línea]. 1999. S.l.: s.n.

[Consulta: 15 noviembre 2018]. Disponible en: https://instituciones.sld.cu/dnspminsap/files/2013/08/Decreto-Ley-199.pdf.

10. GLOBE TESTING, 2019a. Pruebas de seguridad. *Globe Testing* [en línea]. [Consulta: 7 mayo 2019]. Disponible en: https://www.globetesting.com/pruebas-seguridad/.

11. GLOBE TESTING, 2019b. Pruebas de usabilidad. *Globe Testing* [en línea]. [Consulta: 7 mayo 2019]. Disponible en: https://www.globetesting.com/pruebas-de-usabilidad/.

12. GLOBE TESTING, 2019c. Pruebas funcionales. *Globe Testing* [en línea]. [Consulta: 7 mayo 2019]. Disponible en: https://www.globetesting.com/pruebas-funcionales/.

13. GNU PROJECT, 2018. GNU Bash. [en línea]. [Consulta: 15 noviembre 2018]. Disponible en: https://www.gnu.org/software/bash/.

14. GONZÁLEZ R. y JESÚS, Miguel , 2017. Estudio con modelos de datos para la automatización en redes eléctricas inteligentes. [en línea], [Consulta: 15 febrero 2019]. Disponible en: http://helvia.uco.es/xmlui/handle/10396/14506.

15. HERNÁNDEZ L., 2013. MODELO DE IMPLEMENTACIÓN. [en línea]. [Consulta: 7 mayo 2019]. Disponible en: http://ithleovi.blogspot.com/2013/06/unidad-5-modelo-deimplementacion-el.html.

16. ROMERO, Hugo y COITE, Angelica, 2018. Auditoría de Sistema y políticas de Seguridad Informática - Monografias.com. [en línea]. [Consulta: 9 noviembre 2018]. Disponible en: https://www.monografias.com/trabajos12/fichagr/fichagr.shtml.

17. SOMMERVILLE, Ian, 2011. *Software Engineering* [en línea]. 9na edición. S.l.: s.n. [Consulta: 14 febrero 2019]. ISBN ISBN-13: 978-0-13-703515-1. Disponible en: https://edisciplinas.usp.br/pluginfile.php/2150022/mod_resource/content/1/1429431793.203Software%20Engineering%20by%20Somerville.pdf.

18. GUTIÉRREZ, Javier, 2014. *¿Qué es un framework web?* [en línea]. 2014. S.l.: s.n. [Consulta: 24 octubre 2018]. Disponible en: http://www.lsi.us.es/~javierj/investigacion_ficheros/Framework.pdf.

19. RUPERTO M. U., Joaquín, 2000. *AUDITORIA DE TECNOLOGÍAS DE INFORMACIÓN* [en línea]. 2000. S.l.: s.n. [Consulta: 12 noviembre 2018]. Disponible en: http://132.248.9.34/hevila/InvestigacionAdministrativa/2000/vol29/no87/9.pdf.

20. ROSEMBERG R. M., Jose , 2019. Estándares de codificación. *Scribd* [en línea]. [Consulta: 7 mayo 2019]. Disponible en: https://es.scribd.com/presentation/289636418/Estandares-de-codificacion.

21. CERCA V., Juan I., et al., 2014. *Uso de herramientas CASE para la gestión de proyectos de software* [en línea]. 2014. S.l.: s.n. [Consulta: 11 noviembre 2018]. Disponible en: http://www.itcelaya.edu.mx/ojs/index.php/pistas/article/download/1304/1114. ISSN 1405-1249

22. JUNTA DE ANDALUCÍA, 2019. Validar los requisitos del sistema | Marco de Desarrollo de la Junta de Andalucía. [en línea]. [Consulta: 7 mayo 2019]. Disponible en: http://www.juntadeandalucia.es/servicios/madeja/contenido/libro-pautas/185.

23. ROUSE, Margaret, 2016a. ¿Qué es Framework? - Definición en WhatIs.com. *SearchDataCenter en Español* [en línea]. [Consulta: 31 octubre 2018]. Disponible en: https://searchdatacenter.techtarget.com/es/definicion/Framework.

24. ROUSE, Margaret, 2016b. ¿Qué es Servidor de aplicaciones? - Definición en WhatIs.com. [en línea]. [Consulta: 11 noviembre 2018]. Disponible en: https://searchdatacenter.techtarget.com/es/definicion/Servidor-de-aplicaciones.

25. MINCOM, 2016. Inicia la XVI edición de la Convención y Feria Internacional Informática 2016. *MINCOM | Ministerio de las Comunicaciones en Cuba* [en línea]. [Consulta: 11 noviembre 2018]. Disponible en: http://www.mincom.gob.cu/es/node/1560.

26. MOZILLA MDN, 2017. HTML5. *Documentación web de MDN* [en línea]. [Consulta: 11 noviembre 2018]. Disponible en: https://developer.mozilla.org/es/docs/HTML/HTML5.

27. MOZILLA MDN, 2018. CSS3. *Documentación web de MDN* [en línea]. [Consulta: 11 noviembre 2018]. Disponible en: https://developer.mozilla.org/es/docs/Web/CSS/CSS3.

28. MOZILLA MDN, 2019. JavaScript. *Documentación web de MDN* [en línea]. [Consulta: 7 mayo 2019]. Disponible en: https://developer.mozilla.org/es/docs/Web/JavaScript.

29. MYSQL, 2018. MySQL :: About MySQL. [en línea]. [Consulta: 11 noviembre 2018]. Disponible en: https://www.mysql.com/about/.

30. NETBEANS, 2018. Welcome to NetBeans. [en línea]. [Consulta: 9 noviembre 2018]. Disponible en: https://netbeans.org/.

31. NEXOLINUX, 2013. Autenticación PAM en Linux, Elevando la seguridad. *Nexolinux* [en línea]. [Consulta: 7 mayo 2019]. Disponible en: http://www.nexolinux.com/autenticacion-pam-en-linux-elevando-la-seguridad/.

32. NGINX, 2018. NGINX | High Performance Load Balancer, Web Server, & Reverse Proxy. *NGINX* [en línea]. [Consulta: 9 noviembre 2018]. Disponible en: https://www.nginx.com/.

33. OBS BUSINESS SCHOOL, 2018. ¿Qué son las metodologías de desarrollo de software? | OBS Business School. [en línea]. [Consulta: 11 noviembre 2018]. Disponible en: https://www.obs-edu.com/int/blog-project-management/metodologia-agile/que-son-las-metodologias-de-desarrollo-de-software.

34. OHKA, 2018. Ohka. [en línea]. [Consulta: 11 noviembre 2018]. Disponible en: http://www.ohkasystems.com/rapid7.

35. POSTGRESQL, 2018. PostgreSQL: About. [en línea]. [Consulta: 11 noviembre 2018]. Disponible en: https://www.postgresql.org/about/.

36. POWERDATA, 2018. Tipos y función de los gestores de bases de datos. [en línea]. [Consulta: 11 noviembre 2018]. Disponible en:

https://blog.powerdata.es/el-valor-de-la-gestion-de-datos/bid/406547/tipos-y-funci-n-de-los-gestores-de-bases-de-datos.

37. RODRÍGUEZ S., T., 2015. *Metodología de desarrollo para la Actividad productiva de la UCI.* 2015. S.l.: s.n.

38. PRESSMAN, Roger S., 2010. *Ingeniería del software un enfoque práctico*. séptima edición. S.l.: s.n. ISBN ISBN: 978-607-15-0314-5.

39. NORIEGA, Samuel, 2015. Llave Pública y Llave Privada ¿Qué son y cómo funcionan? *Blog de Certificados SSL y Seguridad en Internet* [en línea]. [Consulta: 7 mayo 2019]. Disponible en: https://www.certsuperior.com/Blog/llave-publica-y-llave-privada-que-son-y-como-funcionan.

40. GONZÁLEZ D., Sergio, 2019. Manual básico de como usar Cron. [en línea]. [Consulta: 7 mayo 2019]. Disponible en: https://www.linuxtotal.com.mx/?cont=info_admon_006.

41. SILVERMAN, et al., 2002. SSH, The Secure Shell: The Definitive Guide. *0596000111L* [en línea]. [Consulta: 7 mayo 2019]. Disponible en: https://docstore.mik.ua/orelly/networking_2ndEd/ssh/index.htm.

42. SPARX SYSTEMS, 2019. Sparx Systems - Tutorial UML 2 - Diagrama de Despliegue. [en línea]. [Consulta: 15 febrero 2019]. Disponible en: http://www.sparxsystems.com.ar/resources/tutorial/uml2_deploymentdiagram.html.

43. TENABLE, 2018. Escáner de vulnerabilidades Nessus Professional™. [en línea]. [Consulta: 24 octubre 2018]. Disponible en: https://es-la.tenable.com/products/nessus/nessus-professional.

44. THE APACHE SOFTWARE FOUNDATION, 2018a. About the Apache HTTP Server Project - The Apache HTTP Server Project. [en línea]. [Consulta: 11 noviembre 2018]. Disponible en: https://httpd.apache.org/ABOUT_APACHE.html.

45. THE APACHE SOFTWARE FOUNDATION, 2018b. Apache JMeter - Apache JMeter™. [en línea]. [Consulta: 9 noviembre 2018]. Disponible en: https://jmeter.apache.org/.

46. THE JQUERY FOUNDATION, J.F.-, 2019. jQuery. [en línea]. [Consulta: 7 mayo 2019]. Disponible en: https://jquery.com/.

47. FIGUEROA, Viridiana, 2018. Características de PHP. [en línea]. [Consulta: 11 noviembre 2018]. Disponible en: https://es.scribd.com/doc/50288837/Caracteristicas-de-PHP.

48. VISUAL PARADIGM, 2018. Ideal Modeling & Diagramming Tool for Agile Team Collaboration. [en línea]. [Consulta: 11 noviembre 2018]. Disponible en: https://www.visual-paradigm.com/.

49. W3SCHOOLS, 2019. HTML5 Server-Sent Events. [en línea]. [Consulta: 7 mayo 2019]. Disponible en: https://www.w3schools.com/html/html5_serversentevents.asp.

50. WHITEHAT SECURITY, 2018. Securing Your Digital Business. *WhiteHat Security* [en línea]. [Consulta: 11 noviembre 2018]. Disponible en: https://www.whitehatsec.com/.

51. CHÁVEZ M., Yojahny y QUEZADA, Gustavo, 2018. *SPS-Auditor*. 2018. S.l.: s.n.

Anexos

Tabla 7. Historia de usuario #2

Historia de usuario	
Número: HU_2	**Nombre:** Mostrar reportes de auditoría
Prioridad en negocio: Alta	
Descripción: Muestra los reportes de auditoría agrupados por fecha.	
Prototipo:	

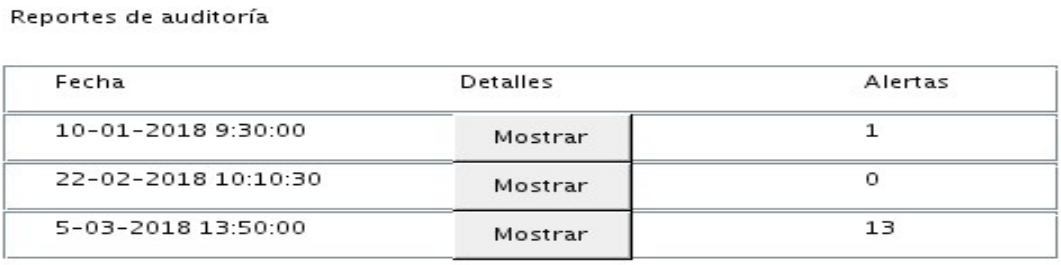


Tabla 8. Historia de usuario #3

Historia de usuario	
Número: HU_3	**Nombre:** Mostrar reportes de auditoría por laboratorio.
Prioridad en negocio: Alta	
Descripción: Muestra los reportes de auditoría de una fecha, agrupados por laboratorios.	
Prototipo:	

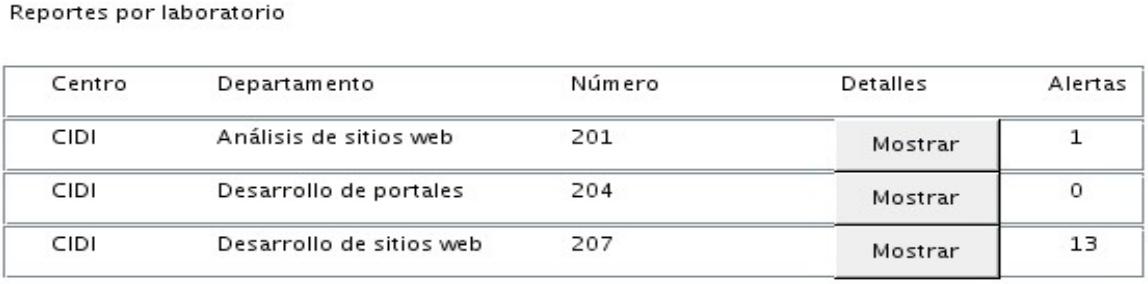


Tabla 9. Historia de usuario #4

Historia de usuario	
Número: HU_4	**Nombre:** Generar informe de las PC.
Prioridad en negocio: Alta	
Descripción: Permite mostrar todos los datos de las estaciones de trabajo, así como sus configuraciones y programas de seguridad e inventarios.	
Prototipo:	

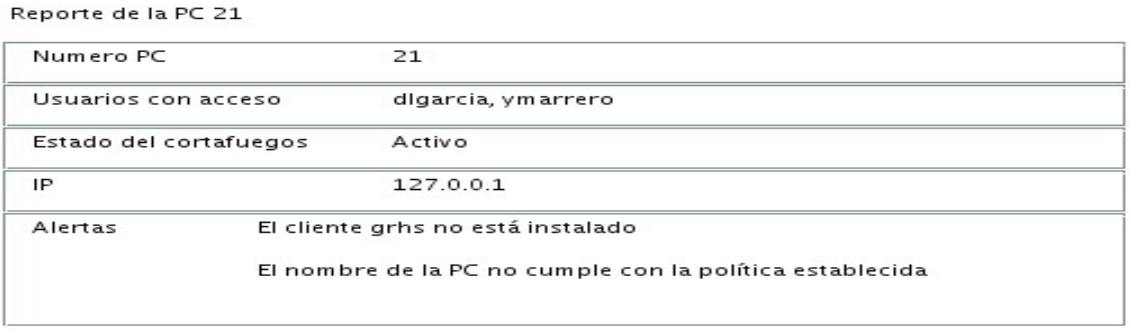


Tabla 10. Historia de usuario #5

Historia de usuario	
Número: HU_5	**Nombre:** Gestionar Perfiles de auditoría
Prioridad en negocio: Media	
Descripción: Permite seleccionar o eliminar los códigos en Bash utilizados para realizar las auditorías de acuerdo a las necesidades del asesor de seguridad.	
Prototipo:	

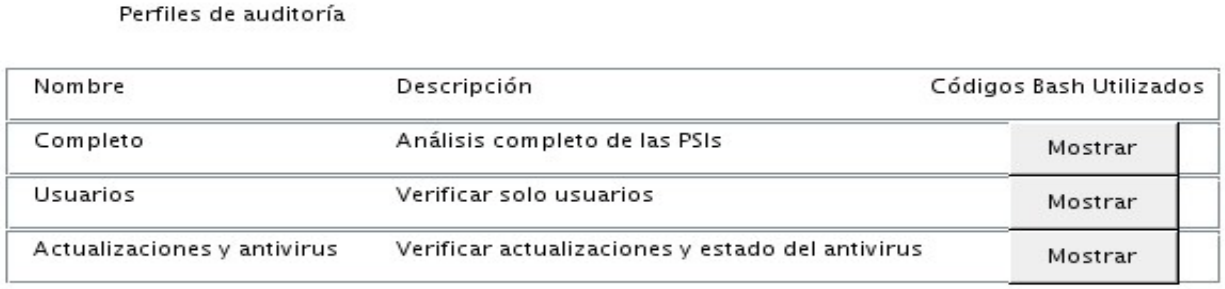


Tabla 11. Historia de usuario #6

Historia de usuario	
Número: HU_6	**Nombre:** Gestionar reglas
Prioridad en negocio: Media	
Descripción: Permite gestionar los códigos en Bash utilizados para realizar las auditorías.	
Prototipo:	

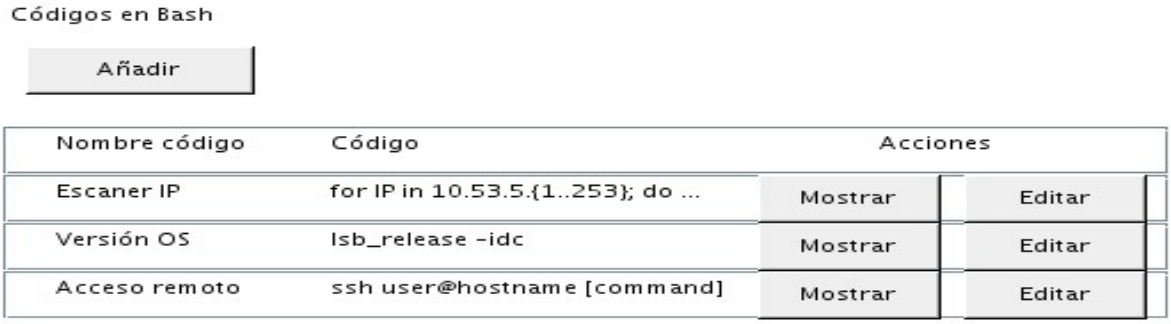


Tabla 12. Historia de usuario #7

Historia de usuario	
Número: HU_7	**Nombre:** Automatizar el proceso de auditoría por fecha.
Prioridad en negocio: Alta	
Descripción: Al seleccionar una fecha el sistema auditará automáticamente.	
Prototipo:	

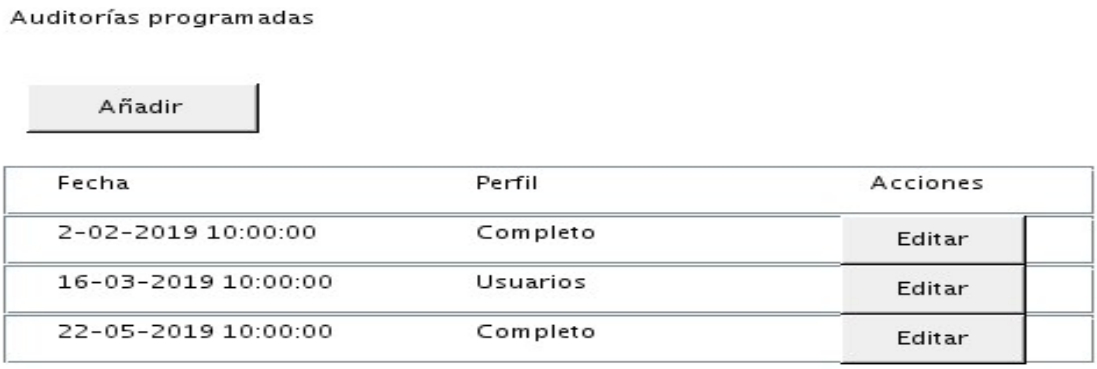


Tabla 13. Caso de prueba # 2

Escenario	Descripción	Escanear	Respuesta del sistema	Flujo central
EC 2.1 Escaneo de IP	El usuario accede a la herramienta, luego al enlace “analizar” y después presiona el botón escanear.	V (Se presiona el botón Escanear)	El sistema escanea los IP activos y muestra un listado con todos ellos y la opción de auditar.	El usuario accede a la herramienta, luego al enlace “analizar” y después presiona el botón escanear. El sistema escanea los IP activos. El sistema muestra un listado con todos los IP detectados y la opción auditar.
EC 2.2 Escaneo de IP	El usuario accede a la herramienta, luego al enlace “analizar” y después presiona el botón escanear.	I (No se presiona el botón Escanear) El sistema escanea los IP activos y muestra un listado con todos ellos y la opción de auditar.	El sistema no escanea los IP activos y no muestra un listado con todos ellos y la opción de auditar.	El usuario accede a la herramienta, luego al enlace “analizar” y no presiona el botón escanear. El sistema no escanea los IP activos. El sistema no muestra un listado con todos los IP detectados y la opción auditar.

Tabla 14. Variables empleadas en el diseño del caso de prueba # 2

No	Nombre de campo	Clasificación	Valor nulo	Descripción
1	Escanear	Escaneo de IP	No	Escanea los IP activos y mostrar un listado, con la opción de auditar.

Tabla 15. Caso de prueba # 3

Escenario	Descripción	Auditar	Respuesta del sistema	Flujo central
EC 3.1 Auditar PC	El usuario después de escanear y obtener el listado, presiona el botón auditar de una de las PC detectadas.	V El usuario presiona el botón auditar de una de las PC del listado. El usuario presenta una	El sistema muestra el resultado de la auditoría a la PC seleccionada en un listado.	El usuario realiza un escaneo y obtiene un listado. El usuario presiona el botón auditar de una de las PC del listado

Escenario	Descripción	Auditar	Respuesta del sistema	Flujo central
		cuenta en la PC y se autentica correctamente mediante acceso remoto.		El sistema se autentica remotamente en la PC con la cuenta de usuario de la aplicación, que debe coincidir con la cuenta de la PC. El sistema obtiene los valores y muestra un listado con los mismos.
EC 1.2 Auditar PC	El usuario después de escanear y obtener el listado, presiona el botón auditar de una de las PC detectadas.	I El usuario presiona el botón auditar de una de las PC del listado. El usuario no presenta una cuenta en la PC y no se autentica correctamente mediante acceso remoto.	El sistema muestra un mensaje de error de autenticación.	El usuario realiza un escaneo y obtiene un listado. El usuario presiona el botón auditar de una de las PC del listado El sistema no se autentica remotamente en la PC con la cuenta de usuario de la aplicación, que debe coincidir con la cuenta de la PC. El sistema no obtiene los valores y muestra un mensaje de error de autenticación.
EC 1.3 Auditar PC	El usuario después de escanear y obtener el listado, no presiona el botón auditar de una de las PC detectadas.	I El usuario no presiona el botón auditar de una de las PC del listado.		El usuario realiza un escaneo y obtiene un listado. El usuario no presiona el botón auditar de una de las PC del listado El sistema no se autentica remotamente en la PC con la cuenta de usuario de la aplicación, que debe coincidir con la cuenta de la PC. El sistema no obtiene los valores y no muestra un listado con los mismos.

Tabla 16. Variables empleadas en el diseño del caso de prueba # 3

No	Nombre de campo	Clasificación	Valor nulo	Descripción
1	Auditar	Auditoría	No	Permite auditar remotamente las PC y que se muestre el listado con los valores de la auditoría.