Universidad de las Ciencias Informáticas

Facultad 4

Trabajo de diploma para optar por el título de Ingeniero en Ciencias Informáticas

“PROCEDIMIENTO PARA LA REALIZACIÓN DE PRUEBAS DE INTRUSIÓN, PARA EL SERVICIO DE PRUEBAS DE SOFTWARE, QUE OFRECE EL GRUPO DE CALIDAD DE FORTES”

Autor: Moraima Nuñez Gato.

Tutor: Ing. Yinimary Ortega Montoya

Co-Tutor: Msc. Roberto López Dosaguez

La Habana, 2011.

“Año 53 de la Revolución”

"Creo en aquel que es capaz de alzar su puño y su voz por defender las ideas que ama".

DECLARACIÓN DE AUTORÍA

Declaro ser autor de la presente tesis y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

_____	_____
Autor	Tuto
Moraima Nuñez GatoIng.	Ing. Yinimary Ortega Montoya

AGRADECIMIENTOS

A Fidel Castro Rúz, por haberme dado la posibilidad de realizar mis estudios en la Universidad, siendo el fundador de la misma, en la que he realizado mis sueños.

A mis padres, gracias a los dos por confiar en mí y por alumbrarme el camino, con sus sabios consejos. A mi hermanita, gracias por ser tan especial y alegrar cada minuto de mi vida.

A Roberto López Dosagües, por ser más que un guía, un amigo entrañable, te agradezco el esfuerzo y dedicación que me has brindado, a mi tutora Yinimary y a Javier, por haber sido siempre comprensivos conmigo, y por muchas veces dejar a un lado su trabajo para dedicarme un tiempo cuando lo necesitaba.

A Greisy, Damir y a Maritza por ayudarme tanto en la investigación, haciendo posible que esta tesis se realizara, por ser ustedes mi consultantes, gracias a todos por su apoyo incondicional.

A mi oponente, por haber elaborado con profesionalidad y fundamentos las preguntas necesarias para complementar la naturaleza de mi trabajo.

A Suli y a Yuliet, gracias por su amistad tan bonita, por haberme ayudado en todo momento, por comprenderme y aceptarme, por ser más que unas amigas, las quiero mucho.

A mi apartamento 142205, gracias por ser buenísimas compañeras de apartamentos.

Lisy y Maibe que son de mis viejas amistades desde primer año las quiero mucho.

Evelin, Annia, Diana,lili ,Anita y Ari por ofrecerme su amistad.

Ahora viene la parte buena de verdad agradecerle a todas aquellas personas que de una forma u otra me apoyaron, me mimaron, me complacieron, con los cuales pasé momentos muy lindos que siempre llevaré en mi corazón. Personitas que supieron ser amigos en todo momento, pero sobre todo soportaron mi carácter y son parte de la gran familia que hice aquí en la UCI.

DEDICATORIA

Le dedico este trabajo principalmente a mi padre Daniel y a mi madre Moraima por ser tan importantes en mi vida, por hacer de mi lo que hoy soy, por tener siempre una palabra cariñosa, tantas noches de desvelos y su apoyo incondicional.

A mi abuelita Vítalia que y tanto me ha querido siempre.

A mi hermanita por cuidarme siempre y estar a mi lado en todos los momentos.

A toda mi familia y amigos que me apoyaron, me cuidaron, me aconsejaron, sin ellos no hubiese podido llegar hasta aquí, por ello me esforcé todo cuanto pude y les dedico íntegramente todo mi trabajo.

RESUMEN

En la actualidad existen disímiles técnicas de pruebas de seguridad, que son empleadas para explotar vulnerabilidades, cuando un software está completamente elaborado, entre estas destacan las pruebas de intrusión, utilizadas como la principal técnica de comprobación de la seguridad en aplicaciones web. En tal sentido, se han desarrollado diferentes herramientas para llevar a cabo este tipo de prueba.

El grupo de calidad del Centro de Tecnologías para la Formación (FORTES), perteneciente a la Universidad de la Ciencias Informáticas, en la actualidad realiza solamente pruebas de Funcionalidad a nivel de sistema, por lo que aun no se efectúan pruebas de Intrusión. El presente trabajo de diploma describe el desarrollo de un procedimiento para pruebas de Intrusión, destinado al servicio que ofrece el grupo de calidad de FORTES. El documento describe el análisis realizado acerca de las principales herramientas que se utilizan para efectuar estas pruebas, y su selección acorde a sus características y necesidades. Además, se abordan temas relacionados con las pruebas de seguridad a las aplicaciones web y los tipos de técnicas utilizadas.

El método de investigación científica seleccionado, permitió el cumplimiento del objetivo general de la investigación y sus objetivos específicos. Se demuestra además su validación, con la aplicación del procedimiento a uno de los proyectos de desarrollo del centro FORTES.

Se incluyen las conclusiones y un grupo de recomendaciones en interés de perfeccionar la solución propuesta.

Palabras claves:

Calidad, Pruebas de Seguridad, Prueba de Intrusión, Pruebas de Software, Pruebas de Caja Negra.

ÍNDICE

INTRODUCCIÓN

Actualmente, el desarrollo de software desempeña un papel importante en la vida económica de cada país. Día a día las grandes empresas productoras de software, se empeñan en informatizar todo lo que les sea posible con su ingenio. En un principio solo les preocupaba la entrega en tiempo del software al cliente, pero a medida que ha pasado el tiempo, los procesos de calidad han llegado a formar parte indisoluble de esta tarea, ahora es imprescindible que el producto satisfaga las necesidades del cliente. Para ello se han creado instituciones encargadas de documentar cuanto sea posible, acerca del proceso de control de la calidad, publicando normas y estándares que rigen el proceso de desarrollo de software, para que el producto final alcance la calidad requerida sin que su producción acarree costos y tiempo innecesarios.

La calidad del software es una preocupación a la que se dedican muchos esfuerzos. Todo proyecto tiene como objetivo producir software de la mejor calidad posible, que cumpla, y si puede supere las expectativas de los usuarios

La obtención de un software con calidad, implica la utilización de metodologías o procedimientos estándares para el análisis, diseño, programación y prueba, que permitan uniformar la filosofía de trabajo y que a la vez eleven la productividad, tanto para la labor de desarrollo como para el control de la calidad del software.

Hasta el momento, el aumento creciente de la informática y tecnologías de la información en los procesos de negocios de la mayoría de las empresas del mundo, ha provocado un incremento en las exigencias de los clientes por adquirir productos de software con mayor calidad. La evolución de la informática y las telecomunicaciones ha traído numerosos beneficios, pero también aumento de los riesgos, por lo que es necesario llevar a cabo métodos cada vez más eficaces para dar seguridad a las aplicaciones.

La seguridad, como parte necesaria del desarrollo de software, es un tema que hasta hace poco tiempo, no era de sumo interés para los desarrolladores de sistemas para el mercado. Frecuentemente se ven noticias relacionadas con sistemas informáticos, donde aparecen graves problemas de seguridad detectados, ya que estos están constantemente enviando y recibiendo información de gran valor, que de ser manipulada podría significar como mínimo pérdida en términos económicos.

La Universidad de las Ciencias Informáticas (UCI), se encuentra en estos momentos inmersa en el desarrollo de aplicaciones informáticas de suma importancia, a partir de convenios firmados con diferentes empresas, tanto del ámbito nacional como internacional. Muchos de estos productos que se implementan, manejan información sensible sobre los clientes o sus empresas.

El objetivo de estos sistemas de desarrollo de software, es incrementar su calidad a partir de mayor transparencia y control. Producir un software de calidad a un costo razonable, trae beneficios tanto para los clientes, como para los desarrolladores. Un cliente satisfecho seguirá requiriendo más funciones y solicitará más productos.

Los clientes exigen en sus contratos, el desarrollo y liberación de un producto de software que garantice la confidencialidad, integridad, autenticidad, confiabilidad, corrección, fiabilidad, mantenibilidad y seguridad de la información.

En la Universidad, existen varios centros productivos que suplen las necesidades de los clientes que mantienen contratos con esta, distribuidos entre las distintas facultades. Estos centros se especializan en diferentes ramas de la informática actual como son: bioinformática, la teleinformática, realidad virtual, etc. FORTES, ubicado en la Facultad 4, es el centro encargado de cumplir con los contratos contenidos dentro de la rama informática, concebida para la producción de software para el apoyo al proceso de enseñanza – aprendizaje, o software educativo, el cual cuenta con el Grupo de Calidad de FORTES (GCF).

El GC F está conformado por estudiantes de tercero a quinto año, y tiene la misión de:

- Ofrecer servicios de pruebas de software a todos los proyectos productivos de FORTES y a otros interesados, realizando actividades de verificación en correspondencia con los roles definidos.
- Formar profesionales competentes especializados en pruebas de software desde la práctica laboral, en el área productiva de FORTES y con la dirección del Grupo de Calidad.

EL GCF, posee limitaciones en su trabajo, ya que no tienen las herramientas que le permitan facilitar el proceso de desarrollo del software manteniendo su integridad durante todo el ciclo de vida, y contribuir de esta manera, a su elaboración con la calidad requerida por el cliente.

Los estudios realizados sobre la seguridad de la información, demuestran que alrededor del 80 al 90% de las violaciones de la seguridad se originan desde dentro. Existen las llamadas Pruebas de Intrusión, que permite realizar el control de la calidad desde la red interna. En ella se verifica explícitamente, hasta qué punto los usuarios internos pueden burlar o violar las medidas de seguridad. Además, da la posibilidad de examinar la implementación técnica de sus medidas de seguridad, para detectar cualquier vulnerabilidad que no haya sido identificada y explotada por las pruebas realizadas.

La no utilización de este tipo de prueba en el GCF, trae consigo que en el producto final no se pueda identificar vulnerabilidades en aplicaciones y bases de datos, también que no permita desarrollar

correcciones rápidas de seguridad, oportunidades de mejora y además no admita desarrollar prácticas recomendables para la corrección e implementación de medidas de prevención.

En la actualidad existen diversas dificultades que impiden que se realice el proceso de Pruebas de Intrusión en el GCF de manera satisfactoria, algunos de estos problemas son:

- En el GCF no se hacen pruebas de seguridad al software.
- Falta de una estrategia de prueba en la que se definan las categorías de Pruebas de Intrusión y que herramientas serán utilizadas.
- El GCF no cuenta con un plan para las Pruebas de Intrusión.
- Poco uso de las buenas prácticas de seguridad
- No cuenta con la documentación necesaria donde adquirir información referente a este tema.
- Desconoce cómo emplear las herramientas necesarias y efectivas para este tipo de control.

Luego de analizar la problemática anterior, se plantea el siguiente **problema de investigación:** El desconocimiento de cómo aplicar Pruebas de Intrusión, para el Servicio de Pruebas de Software que ofrece el GCF, con el propósito de que se exploten al máximo todas las posibilidades que brindan las técnicas de pruebas existentes, para ampliar las probabilidades de detectar fallas en los sistemas, antes de que estos se desplieguen y sean entregados a los clientes.

Objetivo general: Desarrollar un procedimiento para la aplicación de Pruebas de Intrusión, para el Servicio de Pruebas de Software que ofrece el Grupo de Calidad FORTES, con la utilización de herramientas automatizadas y documentadas.

Objetivos Específicos:

- Realizar un estudio sobre el estado del arte, referente al objeto de estudio de la investigación.
- Desarrollar el procedimiento para realizar las Pruebas de Intrusión.
- Validar el procedimiento.

El **objeto de estudio**, son los procesos de Pruebas de Seguridad

Enmarcado en el **campo de acción,** son los procesos de Pruebas de Intrusión en el GCF.

De acuerdo con lo expresado anteriormente se plantea como **Idea a defender:** Con el desarrollo de un procedimiento para la aplicación de las Pruebas de Intrusión para el Servicio de Pruebas del GCF, se podría elevar el nivel del proceso de control de la calidad en los proyectos de FORTES.

Tareas de la Investigación:

- Estudio y valoración de los tipos de pruebas que se llevan a cabo en la industria del software.
- Estudio y fundamentación de las pruebas de seguridad que se le realizan al software.
- Estudio y valoración de las categorías de las Pruebas de Intrusión que se le realizan al software.
- Estudio y valoración de las herramientas a aplicar en los procesos de Pruebas de Intrusión.
- Búsqueda de herramientas para la automatización de las Pruebas de Intrusión en el GCF.
- Desarrollo de un procedimiento para las Pruebas Intrusión, para el Servicio de Pruebas de Software que ofrece el Grupo de Calidad FORTES.
- Validación del procedimiento.

Para lograr los objetivos de este trabajo, se emplearon la combinación dialéctica de los **métodos teóricos y empíricos**. En cuanto a los primeros, se emplean: el *análisis sintético*, en el análisis de la revisión bibliográfica de documentos, sitios web, tesis anteriores, etc., y la posterior síntesis de lo analizado. El *método histórico-lógico,* en el estudio de los antecedentes y evolución de las técnicas y herramientas para la realización de Pruebas de Intrusión.

En cuanto a los **métodos empíricos,** se desarrollaron *entrevistas* de tipo semidirigidas y abiertas a especialistas en el tema, así como la *observación* para la obtención de un conocimiento más detallado.

La estructura del documento que se presenta es la siguiente

El primer capítulo: "FUNDAMENTACIÓN TEÓRICA"

Contiene los conceptos más importantes sobre las Pruebas de Software y su relación con la Calidad de Software, se hace énfasis en las Pruebas de Intrusión, así como los principios, técnicas de pruebas de seguridad, los tipos de pruebas y un resumen de las herramientas para la automatización de las Pruebas de Intrusión.

El segundo capítulo: "PROCEDIMIENTO PARA PRUEBAS DE INTRUSIÓN"

Se explica detalladamente el procedimiento para realizar Pruebas de Intrusión para el Servicio de Pruebas de Software que ofrece el Grupo de Calidad FORTES.

El tercer capítulo: "VALIDACIÓN DEL PROCEDIMIENTO"

Validación del procedimiento luego de su aplicación a un producto de uno de los proyectos que se desarrollan en la Facultad.

CAPÍTULO I: “FUNDAMENTACIÓN TEÓRICA”

Introducción

En este capítulo se realiza un análisis de conceptos y elementos que son de gran importancia para elaborar un procedimiento para el desarrollo y aplicación de las Pruebas de Intrusión. Se aborda de manera general el proceso de pruebas de software, enfatizando las pruebas de seguridad, específicamente el tema de las Pruebas de Intrusión como tema fundamental para este trabajo. Se muestra un estudio de las herramientas que existen para la ejecución de pruebas de seguridad.

1.1 Estado del Arte

Calidad

En la actualidad, los clientes demandan productos de calidad, y dado que existe una gran oferta, podrán elegir aquellos productos que más les satisfagan. Los fabricantes, ante la escasez de su demanda particular, buscan diferenciar sus productos de los de la competencia.

En un principio, los fabricantes no necesitaban diferenciarse, ya que los clientes compraban lo que les proporcionasen, pero esta situación acabó pronto. Posteriormente los fabricantes buscaron diferenciarse mediante el precio, ya que pensaban que el cliente compraría siempre el producto más barato. Aunque esta situación se da aún en ciertos mercados, lo cierto es que el cliente ahora puede elegir y que lo hace en función de la calidad del producto.

El término calidad tiene múltiples significados y, como concepto, cambia y evoluciona con el tiempo. Según la Norma ISO 8402:1994 calidad es el *“Conjunto de características de un producto o servicio que le confieren su aptitud para satisfacer las necesidades expresadas e implícitas”.*

La Norma ISO 9000:2000 define la calidad como el “*grado en el que un conjunto de características inherentes cumple con los requisitos”* (1).

Cuando se refiere al término de calidad no quiere decir mejor, sino lo mejor para el cliente en servicio, es ser preciso, reducir al mínimo la variabilidad y hacer las cosas bien desde el primer momento.

Calidad de Software

“La calidad del software es el grado con el que un sistema, componente o proceso cumple los requerimientos especificados y las necesidades o expectativas del cliente o usuario” (1) .

"Concordancia del software producido con los requerimientos explícitamente establecidos, con los estándares de desarrollo prefijados y con los requerimientos implícitos no establecidos formalmente, que desea el usuario" (2).

Un software debe tener relacionados enfoques de proceso, producto y efecto, a través de la aplicación de varios modelos y estándares como se menciona anteriormente, además garantizar que sea aceptado por clientes satisfechos: entonces, será un producto software con calidad.

Proceso

"Un proceso no es más que la sucesión de pasos y decisiones que se siguen para realizar una determinada actividad o tarea".

"Los procesos deben desarrollarse de forma que quede suficientemente claro qué pasos deben darse para realizarlo. Es decir, se hace necesaria una explicación, fase por fase, de las actividades que componen el proceso" (3).

El término proceso no es más que la asociación de actividades, métodos, prácticas y transformaciones que las personas llevan a cabo para desarrollar y mantener un software.

Seguridad Informática

"Los bienes informáticos de una institución lo constituyen los activos informáticos de tipo Hardware, Software y Datos que son los componentes fundamentales de un sistema de computación. Los datos son generalmente el activo informático más preciado para cualquier institución. El hardware y el software pueden ser caros pero fáciles de reponer, en cambio los datos o la información contienen la vida de una institución y su valor a veces es incalculable (4)*"*.

El término de seguridad es el conjunto de métodos y herramientas destinados a proteger los bienes informáticos.

Aplicación Web

En la ingeniería de software se denomina aplicación web a aquellas aplicaciones que los usuarios pueden utilizar accediendo a un servidor web a través de Internet o de una Intranet mediante un navegador. En otras palabras, es una aplicación de software que se codifica en un lenguaje soportado por los navegadores web (HTML, Java Script, Java, etc.) en la que se confía la ejecución al navegador (5).

Las aplicaciones web son populares, debido a lo práctico del navegador web como cliente ligero, así como a la facilidad para actualizar y mantener aplicaciones web sin distribuir e instalar software a

miles de usuarios potenciales. Existen aplicaciones como los web mails, wikis, weblogs, tiendas en línea y la propia Wikipedia que son ejemplos bien conocidos de aplicaciones web.

Pruebas.

La IEEE, "*define las pruebas como una actividad en la cual un sistema o componente es ejecutado bajo condiciones específicas, se observan o almacenan los resultados y se realiza una evaluación de algún aspecto del sistema o componente"* (6).

"Una buena prueba tiene una alta probabilidad de encontrar un error, para alcanzar esto el responsable de prueba debe entender la aplicación e intentar desarrollar una imagen mental de cómo podría fallar el producto, una buena prueba no debe ser redundante y no debe ser ni demasiado sencilla ni demasiado compleja" (7).

Dado estos conceptos manejados en el mundo del software sobre pruebas. Se debe destacar que la prueba de software es un elemento crítico para la garantía de la calidad del software y representa una revisión final de las especificaciones, del diseño y de la codificación.

1.2 Pruebas del Software

El único instrumento adecuado para determinar el status de la calidad de un producto software es el proceso de pruebas. En este proceso se ejecutan pruebas dirigidas a componentes del software o al sistema de software en su totalidad, con el objetivo de medir el grado en que el software cumple con los requerimientos. En las pruebas se usan casos de prueba, especificados de forma estructurada mediante Técnicas de prueba. El proceso de pruebas, sus objetivos y los métodos y técnicas usados se describen en el plan de prueba. (8)

1.2.1 Objetivos de las Pruebas

Se debe tener bien claro con qué objetivos se prueba un software, para enfocar el trabajo en función de dar cumplimiento a estos para garantizar la calidad del producto desarrollado.

Según Glen Myers en el libro de Pressman, establece normas que pueden servir para los objetivos de las pruebas:

- La prueba es el proceso de ejecución de un programa con la intención de descubrir un error.
- Un buen caso de pruebas es aquel que tiene una alta posibilidad de mostrar error no descubierto hasta entonces.
- Una prueba tiene éxito si descubre un error no detectado hasta entonces. (9)

1.2.2 Principios de las pruebas

Para realizar un buen proceso de pruebas es fundamental guiarse por los principios básicos que facilitará a los ingenieros de pruebas mediante métodos y técnicas crear un buen diseño de casos de pruebas.

- Todas las pruebas se les debería poder hacer un seguimiento hasta los requisitos del cliente.
- Las pruebas deberían planificarse antes de que empiecen.
- El principio de Pareto es aplicable a la Prueba del Software.
- Las pruebas deben empezar por lo pequeño y progresar hacia lo grande.
- No es posible hacer pruebas exhaustivas.
- Las pruebas deberían ser conducidas por un equipo independiente. (10)

1.2.3 Plan de Prueba

El propósito del plan de pruebas es dejar de forma explícita el alcance, el enfoque, los recursos requeridos, el calendario, los responsables y el manejo de riesgos de un proceso de pruebas.

Cada prueba debe:

- Dejar claro qué tipo de propiedades se quieren probar (corrección, robustez, fiabilidad, amigabilidad).
- Dejar claro cómo se mide el resultado.
- Especificar en qué consiste la prueba (hasta el último detalle de cómo se ejecuta).
- Definir cuál es el resultado que se espera (identificación, tolerancia,...). ¿Cómo se decide que el resultado es acorde con lo esperado?
- Las pruebas carecen de utilidad, tanto, sí no se sabe exactamente lo que se quiere probar, sí no se está claro cómo se prueba, o si el análisis del resultado se hace a simple vista. (11)

1.2.4 Estrategias de prueba del software

Una estrategia de prueba del software integra las técnicas de diseño de casos de pruebas del software en una serie de pasos bien planificados que dan como resultados una correcta construcción del software.

La estrategia proporciona un mapa que describe los pasos que hay que llevar a cabo como parte de la prueba, cuando se deben planificar y realizar esos pasos, y cuanto esfuerzo, tiempo y recursos que van a requerir (10).

1.2.5 Niveles de pruebas

Las pruebas están estructuradas por diferentes niveles (9) estos son:

- Nivel de Unidad.
- Nivel de Integración.
- Nivel de Sistema.
- Nivel de Aceptación.

Nivel de Unidad: La prueba de unidad se centra en el módulo. Usando la descripción del diseño detallado como guía, se prueban los caminos de control importantes con el fin de descubrir errores dentro del ámbito del módulo. La prueba de unidad hace uso intensivo de las técnicas de prueba de caja blanca[1].

Nivel de Integración: La prueba de integración es una técnica sistemática para construir la estructura del programa mientras que, al mismo tiempo, se lleva a cabo pruebas para detector de errores asociados con la interacción. El objetivo es coger los módulos probados mediante la prueba de unidad y construir una estructura de programa que esté de acuerdo con lo que dicta el diseño. (15)

Nivel de Sistema: Pruebas al software funcionando como un todo. La prueba del sistema constituye el proceso de un sistema integrado de hardware y software que permite comprobar que los requisitos funcionales se cumplen, así como el buen funcionamiento y rendimiento del software conjuntamente con la parte hardware que implica dicho software. Permite una adecuación de la documentación de usuario así como la ejecución y rendimiento en condiciones de límites y de sobre carga. Las pruebas de sistema persiguen la integración de cada módulo, además de buscar las discrepancias entre el sistema, especificaciones y su documentación.

Nivel de Aceptación: La prueba de aceptación es una prueba determinante para la justificación de las ventajas que ofrece el empleo del producto desarrollado. En esta prueba se evalúa el grado de calidad del software, y es realizada por un grupo de usuarios finales o los clientes, para asegurarse de que el sistema cumple con los requisitos planteados. (9)

[1] Ir al epígrafe caja blanca, 1.3.1, de la presente investigación.

1.2.6 Clasificación de las pruebas según el grado de automatización

Las pruebas se pueden categorizar según el grado de automatización, éstas pueden ser:

- **Pruebas manuales:** Son las que se hacen normalmente al programar o las que ejecuta una persona con la documentación generada durante la codificación. Por ejemplo comprobar cómo se visualiza el contenido de una página web en dos navegadores diferentes.
- **Pruebas automatizadas:** Se usa un determinado software para sistematizar las pruebas y obtener los resultados de las mismas. Por ejemplo verificar un método de ordenación. (12)

1.3 Técnicas de pruebas

A un software determinado es necesario realizarles distintos de técnicas de pruebas, para lograr una buena calidad en el mismo y así lograr la menor cantidad posible de errores.

Estas técnicas son: Pruebas de caja blanca y las pruebas de caja negra.

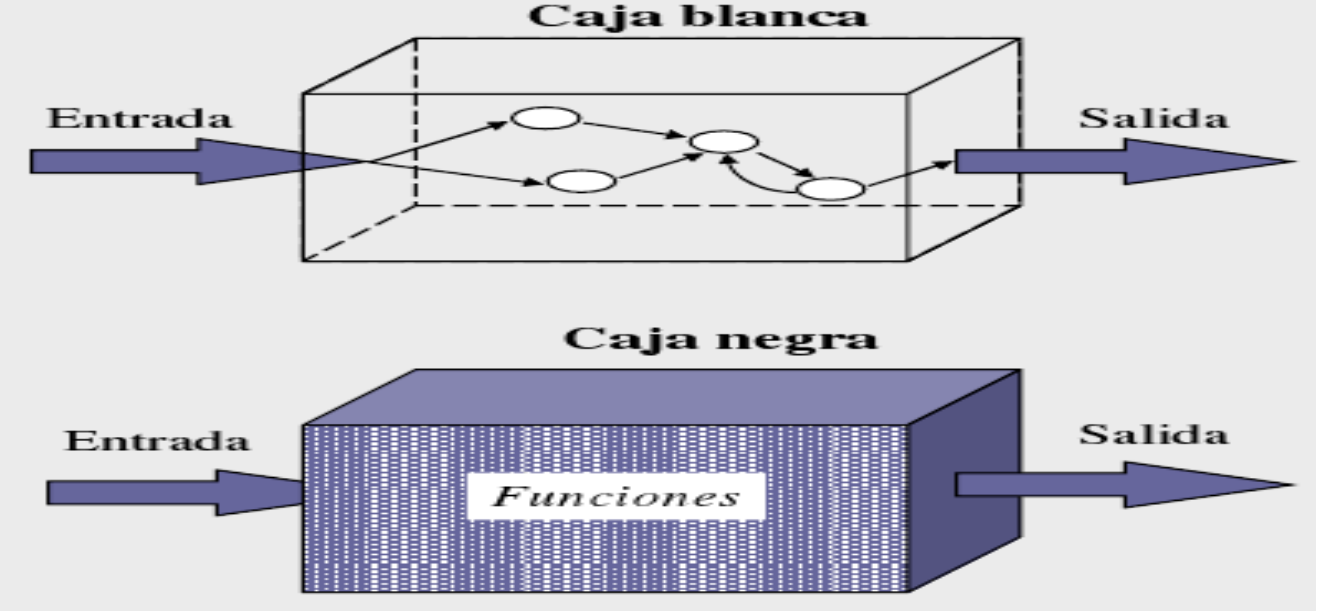

Figura 2: Muestra gráfica del funcionamiento de las pruebas de caja negra y caja blanca.

1.3.1 Las Pruebas de Caja Blanca

Es conocida también como caja de cristal. Este es una técnica de diseño de casos de prueba que usa la estructura de control del diseño procedimental para obtener los mismos. Mediante esta técnica, el ingeniero del software puede obtener casos que:

- Se ejerciten por lo menos una vez todos los caminos independientes para cada módulo.

- Se ejerciten todas las decisiones lógicas en sus vertientes verdaderas y falsa.
- Ejecuten todos los bucles en sus límites y con sus límites operacionales.
- Se ejerciten las estructuras internas de datos para asegurar su validez. (9)

En la actualidad este tipo ha tomado un gran auge tanto que sin esta prueba un sistema desarrollado carecería de garantías y credibilidad en sus resultados.

Existen varios métodos **de Prueba de Caja Blanca** las cuales se mencionan a continuación:

- Prueba del Camino Básico.
- Prueba de Condición.
- Prueba de Bucles.
- Prueba de Flujo de Datos. (9)

1.3.2 Las Pruebas de Caja Negra

Las pruebas de caja negra, también denominadas como comportamiento, se centran en los requisitos funcionales del software. O sea, esta técnica permite al ingeniero del software obtener conjuntos de condiciones de entrada que ejerciten completamente todos los requisitos funcionales de un programa. Más bien se trata de un enfoque complementario que intenta descubrir diferentes tipos de errores:

- Funciones incorrectas o ausente.
- Errores de interfaz.
- Errores en estructuras de datos o en accesos a las bases de datos externas.
- Errores de rendimiento.
- Errores de inicialización y terminación.

A diferencia de la prueba de caja blanca, que se lleva a cabo previamente en el proceso, la de caja negra tiende a aplicarse durante fases posteriores de la prueba. (9)

Mediante los **métodos de Prueba de Caja Negra** se obtiene un conjunto de casos de prueba que satisfacen los siguientes criterios: (1) casos de prueba que reducen, en un coeficiente que es mayor que uno, el número de casos de prueba adicionales que se deben diseñar para alcanzar una prueba razonable y (2) casos de prueba que nos dicen algo sobre la presencia o ausencia de clases de errores en lugar de errores asociados solamente con la prueba que estamos realizando.

- Clases de equivalencia.
- Análisis de valores límite.
- Prueba de comparación.
- Tabla Ortogonal.
- Grafo Causa-efecto. (9)

Partición de equivalencia: Una partición equivalente es una técnica de prueba de Caja Negra que divide el dominio de entrada de un programa en clases de datos de los que se pueden derivar casos de prueba. El diseño de casos de prueba para la partición equivalente se basa en una evaluación de las clases de equivalencia para una condición de entrada. Una clase de equivalencia representa un conjunto de estados válidos o inválidos para condiciones de entrada. Regularmente, una condición de entrada es un valor numérico específico, un rango de valores, un conjunto de valores relacionados o una condición lógica. (9)

Análisis de valores límite (AVL): El análisis de valores límite nos lleva a una elección de casos de prueba que ejerciten los valores límite**.** El análisis de valores límite es una técnica de diseño de caso de prueba que complementan a la partición equivalente. En lugar de seleccionar cualquier elemento de una clase de una clase equivalente, AVL lleva a la elección de casos de prueba en los extremos de la clase. En lugar de centrarse solamente en las condiciones de entrada, el AVL obtiene casos de prueba también para el campo de salida. (9)

Grafos de Causa – Efecto: Es un enfoque sistemático para seleccionar conjuntos de casos de prueba de alto rendimiento que exploran combinaciones en las condiciones de entrada. Se registran restricciones describiendo combinaciones de causas y efectos posibles, también se crean grafos de causa-efecto y se convierten en tablas de decisiones. (9)

Prueba de Comparación: Cuando se desarrolla un software en el cual la fiabilidad es absolutamente crítica en ese tipo de aplicaciones se utiliza hardware y software redundante para minimizar la posibilidad de error. Cuando se desarrolla software redundante varios equipos de ingeniería del software desarrollan versiones independientes de una aplicación usando las mismas especificaciones. Cuando ocurre esto se deben probar todas las versiones con los mismos juegos de datos para comprobar si las salidas son las mismas. Luego se ejecutan las versiones en paralelo para hacer una comparación en tiempo real de los resultados para garantizar la consistencia del software. (9)

Tabla Ortogonal: Esta prueba puede aplicarse a problemas en que el dominio de entrada es relativamente pequeño pero demasiado grande para posibilitar pruebas exhaustivas. El método de la tabla ortogonal es particularmente útil al encontrar errores asociados con fallos localizados una categoría de error asociada con defectos de la lógica dentro de un componente del software. (9)

De las técnicas anteriormente mencionadas resulta la más efectiva una combinación entre Partición equivalente y Valores Límite, la primera permite simplificar el número de casos de prueba, obteniendo valores válidos e inválidos de las entradas. Teniendo como ventaja que cada vez que un usuario usa el programa está a la vez realizando una prueba al mismo, mientras que la segunda complementa a la de Partición equivalente porque en lugar de realizar la prueba con cualquier elemento de la partición equivalente, se escogen los valores en los bordes de la clase. Se derivan tanto casos de prueba a partir de las condiciones de entrada como con las de salida.

1.4 Tipos de prueba

A medida que se avanza en los niveles antes mencionados se hace necesaria la aplicación de los diferentes tipos de pruebas, los cuales se muestran a continuación. (13)

Pruebas de Función:

Prueba centrada en validar las funciones que son objeto de prueba como lo que deben ser, ofreciendo los servicios, métodos o casos de usos requeridos. Esta prueba es implementada y ejecutada contra diferentes objetos de pruebas, incluyendo Unidades, Unidades Integradas, Aplicaciones y Sistemas. (13)

Pruebas de Seguridad:

Prueba centrada en asegurar que los datos o sistemas que son objetos de prueba, son accedidos sólo por los actores que tienen permiso para hacerlo. Esta prueba es implementada y ejecutada contra varios objetos de prueba. Este tipo de prueba contiene varias técnicas de pruebas y una de la más importante es la técnica de la prueba de Intrusión. Posteriormente se profundizará todo lo relacionado con este tipo de prueba[2]. (13)

Pruebas de Volumen:

Prueba centrada en verificar las habilidades de los objetos de prueba para manejar grandes cantidades de datos, tanto en entrada como en salida, o residente en la base de datos. Puede incluir un procedimiento que indique el uso de consultas que devuelvan todo el contenido de la base de datos, o cuando la cantidad de datos de entrada excede a la cantidad establecida de cada campo. (13)

[2] Ir al epígrafe 1.5 Pruebas de Seguridad de la presente tesis

Pruebas de Usabilidad:

Prueba encaminada a factores humanos, estéticos, consistencia en la interfaz de usuario, ayuda sensitiva al contexto y en línea, asistente documentación de usuarios y materiales de entrenamiento. (13)

Pruebas de Fiabilidad:

Se enfocan en probar la robustez (resistencia a fallas) y el uso adecuado del lenguaje, sintaxis y uso de recursos. Este tipo de prueba puede aplicarse tanto a unidades como a integración de unidades. (13)

Pruebas de Estructura:

Enfocada a la valoración a la adherencia a su diseño y formación. Este tipo de prueba es hecho a las aplicaciones Web asegurando que todos los enlaces están conectados, el contenido deseado es mostrado y no hay contenido huérfano. (13)

Pruebas de Estrés:

Enfocada a evaluar cómo el sistema responde bajo condiciones anormales. (Extrema sobrecarga, insuficiente memoria, servicios y hardware no disponible, recursos compartidos no disponible). (13)

Benchmark:

Compara el rendimiento de un elemento nuevo o desconocido a uno de carga de trabajo de referencia conocido. (13)

Pruebas de Contención:

Enfocada a la validación de las habilidades del elemento a probar para manejar aceptablemente la demanda de múltiples actores sobre un mismo recurso (registro de recursos, memoria, etc.). (18)

Pruebas de Carga:

Usada para validar y valorar la aceptabilidad de los límites operacionales de un sistema bajo carga de trabajo variable, mientras el sistema bajo prueba permanece constante. La variación en carga es simular la carga de trabajo promedio y con picos que ocurre dentro de tolerancias operacionales normales. (13)

Pruebas de Perfil de Desempeño:

Enfocadas a monitorear el tiempo en flujo de ejecución, acceso a datos, en llamada a funciones y sistema para identificar y direccional los cuellos de botellas y los procesos ineficientes. (13)

Pruebas de Configuración:

Enfocada a asegurar que funciona en diferentes configuraciones de hardware y software. Esta prueba es implementada también como prueba de rendimiento del sistema. (13)

Pruebas de Instalación:

Enfocada a asegurar la instalación en diferentes configuraciones de hardware y software bajo diferentes condiciones (insuficiente espacio en disco, etc.). (13)

1.4.1 Tipo de Pruebas por niveles de prueba

- ✓ **Nivel de Unidad:**

Pruebas de caja negra

- Partición de equivalencia.
- Análisis de valores límite.
- Grafos de Causa – efecto.

Prueba de Caja Blanca

- Del camino básico.
- Bucles.
- Flujo de Datos.
- Condición.

- ✓ **Nivel de Integración:**
  - Integración descendente (Top Down).
  - Integración ascendente (Bottom-up).

- ✓ **Nivel de Sistema**

Funcionales: Enfocadas a los requisitos funcionales del software, a su interacción con el cliente de la forma que ha sido pactada. (13)

No funcionales: Enfocadas a los requisitos no funcionales del proyecto. Desglosadas en un número de pruebas definidas como: (13)

- Prueba de Seguridad.
- Prueba de Disponibilidad y Red.
- Prueba de Rendimiento o Carga.
- Prueba de Compatibilidad.
- Prueba de Resistencia o Stress.
- Prueba de Usabilidad.
- Prueba de Fiabilidad. (13)

1.5 Pruebas de Seguridad

Cualquier sistema basado en computadora que maneje información sensible o lleve a cabo acciones que pueda perjudicar o beneficiar impropiamente a las personas es un posible objetivo para entradas impropias o ilegales al sistema. Este acceso al sistema incluye un amplio rango de actividades: piratas informáticos, que intentan entrar en los sistemas por deporte, empleados disgustados que intentan penetrar por venganza e individuos deshonestos que intentan penetrar para obtener ganancias personales ilícitas. (10)

Durante la prueba de seguridad, el responsable de la prueba desempeña el papel de un individuo que desea entrar en el sistema. Debe intentar conseguir las claves de acceso por cualquier medio, puede atacar al sistema con software a medida, diseñando para romper cualquier defensa que se haya construido, debe bloquear el sistema, negando así el servicio a otras personas, debe producir a propósito errores del sistema, intentando acceder durante la recuperación o debe curiosear en los datos sin protección, intentando encontrar la clave de acceso al sistema. (10)

Las Pruebas de Seguridad pretenden medir la Confidencialidad, Integridad y Disponibilidad de los datos, desde la perspectiva del aplicativo, es decir, partiendo de identificar amenazas y riesgos desde el uso o interface de usuario final. Las pruebas de seguridad pretenden medir y cuantificar los riesgos a los cuales se ven expuestos los aplicativos, tanto en la infraestructura interna como externa, valiéndose de la filosofía del Hacking ético. Las pruebas de seguridad simulan un ataque informático desde cualquier perspectiva (Internet, red interna, redes asociadas, acceso remoto, etc.) para establecer qué tan posible sería para un atacante, comprometer la seguridad de la información y validar la posible ocurrencia de un fraude. (14)

La seguridad y garantía de la información que reside en las aplicaciones web no es un tema que se deje a la ligera. Tomar las medidas preventivas es garantía de que ni las organizaciones ni los usuarios sufrirán algún tipo de pérdida económica, robo de identidad o desprestigio frente a clientes e inversionistas.

La UCI, se encuentra inmersa en el desarrollo de un proceso de mejora para alcanzar el nivel 2 de certificación de CMMI. Para ello, algunas áreas de la universidad realizan un conjunto de actividades que acrediten al mejoramiento de las mismas, entre ellas se encuentra el área de calidad. El área de calidad está representada por el Centro de Calidad para Soluciones Tecnológicas (Calisoft).

Calisoft es un centro de referencia que garantiza el crecimiento continuo de producción de software con calidad y un órgano de certificación de software. Cuando se va a realizar la liberación de un producto en el centro calisoft , se trabaja conjuntamente con la facultad 2 este tiene un contrato con la facultad 2 para que realice las pruebas de seguridad, ya que en la universidad la única que efectúa este tipo de pruebas es la facultad 2.

1.5.1 Principios de Pruebas de Seguridad

Existen algunas concepciones equivocadas a la hora de desarrollar una metodología de pruebas para corregir errores de seguridad en el software. Este epígrafe cubre algunos de los principios básicos que deberían ser tenidos en cuenta por los profesionales a la hora de realizar pruebas en busca de errores de seguridad en el software. Para realizar el estudio de estos principios se consultó el documento OWASP. (12)

- **No existe la bala de plata:** Aunque es tentador pensar que un scanner de seguridad o un cortafuegos de aplicación nos proporcionará o una multitud de defensas o identificará una multitud de problemas, en realidad no existen balas de plata para el problema del software inseguro. El software de evaluación de seguridad de aplicaciones, aunque útil como un primer paso para encontrar la hora de proporcionar una cobertura de pruebas adecuada. Hay que recordar que la seguridad es un proceso, no un producto. (12)
- **Piensa estratégicamente, no tácticamente:** En los últimos años, los profesionales de la seguridad han llegado a darse cuenta de la mentira que representa el modelo de parchear y penetrar, generalizado en seguridad de la información durante los 90. El modelo de parchear y penetrar comprende corregir un error reportado, pero sin la investigación adecuada de la causa origen. El modelo de parchear y penetrar se asocia a menudo con la ventana de vulnerabilidad mostrada en la siguiente figura1. La evolución de vulnerabilidades en el software común usado por todo el mundo ha demostrado la ineficacia de este modelo. Estudios de las vulnerabilidades han mostrado que con el tiempo de reacción de los atacantes por todo el mundo, la ventana de vulnerabilidad típica no provee del tiempo suficiente para la instalación de parches, ya que el tiempo entre que la vulnerabilidad es descubierta y un ataque automatizado es desarrollado y divulgado decrece cada año. Existen también varias asunciones erróneas en este modelo de parchear y penetrar: los parches interfieren con la operativa normal y pueden quebrantar aplicaciones existentes, y no todos los usuarios podrían

ser conscientes de la disponibilidad de un parche. Por lo tanto, no todos los usuarios del producto aplicarán los parches, debido a la incidencia o por falta de conocimiento de la existencia del parche. (12)

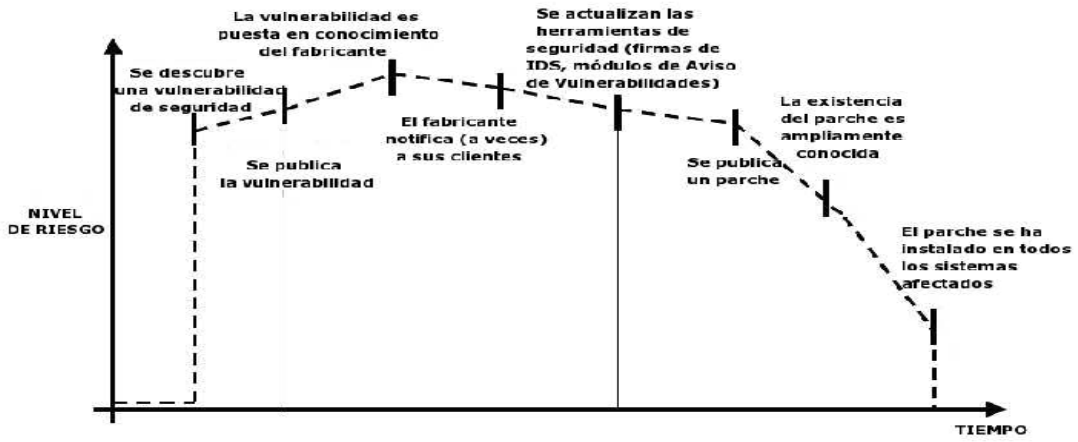

Figura 2: Ventana de exposición.

- **Ciclo de Vida de Desarrollo del Software, en inglés Software DevelopmentsLifeCycle (SDLC) es el rey:** El SDLC es un proceso bien conocido por los desarrolladores. Mediante la integración de la seguridad en cada fase del SDLC, permite una aproximación integral a la seguridad de aplicaciones que se apoya en los procedimientos ya existentes en la organización. Ten en cuenta que mientras los nombres de las varias fases pueden cambiar dependiendo del modelo SDLC usado por una organización, cada fase conceptual del arquetipo SDLC será usada para desarrollar la aplicación (es decir, definir, diseñar, desarrollar, implementar, mantener). Cada fase tiene implicaciones de seguridad que deberán formar parte del proceso existente, para asegurar un programa de seguridad rentable y exhaustiva.
- **Prueba pronto y prueba a menudo:** El detectar un error en una etapa temprana dentro del SDLC permite que pueda ser abordado con mayor rapidez y a un coste menor. Un error de seguridad no es diferente de uno funcional o de un error en el rendimiento, en este contexto. Un factor clave en hacer esto posible es educar a las organizaciones de desarrollo y Calidad sobre problemas de seguridad comunes y los métodos para detectar y prevenirlos. A pesar de que las nuevas bibliotecas, herramientas o lenguajes pueden ayudar a diseñar mejores programas (con menos errores de seguridad), nuevas amenazas están apareciendo constantemente y los desarrolladores deben ser conscientes de aquellas que afectan al software que están desarrollando. La educación en cuanto a pruebas de seguridad ayuda también a los desarrolladores a adquirir el enfoque apropiado para probar una aplicación desde la perspectiva de un atacante. Esto permite a cada organización considerar los problemas de seguridad como una parte de sus responsabilidades. (12)

- **Comprende el alcance de la seguridad:** Es importante saber cuánta seguridad requerirá un proyecto determinado. A la información y activos que van a ser protegidos les debería ser asignada una clasificación que indique como van a ser manejados (por ejemplo confidencial, secreto, alto secreto). Debería discutirse con un consejo legal para asegurarse de que se cumplirá cualquier necesidad de seguridad específica.
- **Enfoque:** Probar con éxito una aplicación en busca de vulnerabilidades de seguridad requiere pensar "fuera de lo convencional". Los casos de uso habituales realizarán una comprobación del comportamiento normal de la aplicación cuando un usuario está empleándola del modo que se esperas. Una buena comprobación de seguridad requiere ir más allá de lo que se espera y pensar como un atacante que está intentando quebrantar la aplicación. Pensar creativa mente puede ayudar a. Esta es una de las razones por las que las herramientas automatizadas son realmente malas en la prueba automática de vulnerabilidades, esta mentalidad creativa debe ser usada caso por caso, y la mayoría de las aplicaciones web son desarrolladas de forma única y singular (incluso aunque usen marcos de desarrollo comunes).

- **Comprende el objeto de estudio:** Una de las primeras iniciativas principales en cualquier buen programa de seguridad debería ser pedir documentación precisa de la aplicación. La arquitectura, diagramas de flujo de datos, casos de uso, y demás deberían ser escritos en documentos formales y puestos a disposición para revisión. Los documentos de aplicación y especificación técnica deberían incluir información que liste no solo los casos de uso deseados, sino también cualquier caso de uso específicamente no admitido. Finalmente, es bueno tener como mínimo una infraestructura de seguridad básica que permita la monitorización y el análisis de la evolución de los ataques contra sus aplicaciones y red (por ejemplo, Sistemas de Detección de Intrusos (IDS). (12)

- **Utiliza las herramientas adecuadas:** Aunque ya se enuncian que no existe una herramienta bala de plata, las herramientas desempeñan un papel crítico en el programa de seguridad global. Hay toda una gama de herramientas de código abierto y comercial que pueden ayudar en la automatización de muchas tareas de seguridad rutinarias. Estas herramientas pueden simplificar y acelerar el proceso de seguridad ayudando al personal de seguridad en sus tareas. Sin embargo, es importante comprender exactamente lo que estas herramientas pueden hacer y lo que no, de forma que no se exagere su uso o sean usadas incorrectamente.

- **Lo importante está en los detalles:** Es un factor crítico no realizar una revisión de seguridad superficial de una aplicación y considerarla completa. Esto infundirá una falsa sensación de confianza que puede ser tan peligrosa como no haber hecho una revisión de seguridad. Es vital revisar cuidadosamente los resultados encontrados y eliminar cualquier falso positivo que

pueda quedar en el informe. Informar de un resultado de seguridad hallado que sea incorrecto puede desacreditar el resto de los mensajes válidos de un informe de seguridad. Debe tomarse con cuidado en verificar que la posible sección de la lógica de aplicación ha sido probada, y que cada escenario de caso de uso ha sido explorado en busca de posibles vulnerabilidades.

- **Usa el código fuente cuando esté disponible:** A pesar de los resultados de las Pruebas de Intrusión de caja negra pueden ser impresionantes y útiles para demostrar como son expuestas las vulnerabilidades en producción, no son el método más efectivo para escarizar una aplicación. Si el código fuente de la aplicación está disponible, debería ser entregado al personal de seguridad para ayudarles durante la realización de la revisión. Es posible descubrir vulnerabilidades dentro del código fuente de la aplicación que se perderían durante el trabajo de pruebas de caja negra.

- **Desarrolla métricas:** Una parte importante de un buen programa de seguridad es la habilidad para determinar si las cosas están mejorando. Es importante seguir la pista de los resultados de los trabajos de prueba, y desarrollar métricas que podrán revelar la evolución de la seguridad de la aplicación en la organización. Estas métricas pueden mostrar si son necesarias más educación y formación, si hay algún mecanismo de seguridad en particular que no es comprendido con claridad por desarrollo, y si el número total de problemas relacionados con seguridad que se han encontrado cada mes se está reduciendo. Unas métricas consistentes que puedan ser generadas automáticamente a partir de código fuente disponible ayudarán también a la organización en la evaluación de la eficacia de los mecanismos introducidos para reducir el número de errores de seguridad en el desarrollo de software. Las métricas no son fáciles de desarrollar, por lo que usar métricas estándar, como las provistas por el Proyecto de Seguridad de Aplicaciones Web Abiertas (OWASP) las métricas y otras organizaciones podría ser una buena base de la que partir para empezar. (12)

1.5.2 Clasificación de las Técnicas de Pruebas de Seguridad

Existen diversas técnicas utilizadas para la construcción de un programa de pruebas, las cuales son de gran importancia para mantener la seguridad en las aplicaciones web, estas técnicas son una forma para controlar y mantener la aplicación segura y poco vulnerable ante ataques de cualquier tipo.

Inspecciones y Revisiones Manuales: Son revisiones realizadas por personas. Generalmente comprueban las implicaciones de seguridad de personas, políticas y procesos, aunque pueden incluir la inspección de decisiones tecnológicas, como puede ser los diseños de la arquitectura escogidos. Casi siempre se llevan a cabo analizando documentación o mediante entrevistas con los diseñadores

o propietarios de los sistemas. Aunque el concepto de inspección manual y revisión personal es simple, se considera entre las técnicas más efectivas y eficaces. Las inspecciones y revisiones manuales son uno de los pocos métodos para probar el ciclo de vida de desarrollo del software y asegurar que en el mismo existe la política de seguridad o competencia adecuada. Una de sus ventajas es que no requiere tecnología de apoyo, puede ser aplicada a una variedad de situaciones, es flexible, además fomenta el trabajo en grupo. Mientras que dentro de sus desventajas podemos encontrar que consume mucho tiempo, material de apoyo no siempre disponible y precisa de bastantes conocimientos. (12)

Modelado de Amenazas: En el contexto del ámbito técnico, el modelado de amenazas se ha convertido en una técnica popular para ayudar a los diseñadores de sistemas acerca de las amenazas de seguridad a las que se enfrentan sus sistemas. Les permite desarrollar estrategias de mitigación para vulnerabilidades potenciales. El modelado de amenazas ayuda a las personas a concentrar sus limitados recursos y atención en aquellas partes del sistema que más lo necesitan. Los modelos de amenaza deberían ser creados tan pronto como sea posible en el ciclo de vida de desarrollo del software, y deberían ser revisados a medida que la aplicación evoluciona y el desarrollo va progresando. El modelado de amenazas es esencialmente la evaluación del riesgo en aplicaciones. Se recomienda que todas las aplicaciones tengan un modelo de amenaza desarrollado y documentado. Dentro de sus ventajas podemos encontrar que tiene una visión práctica del sistema desde el punto de vista de un atacante, es flexible y se aplica en una fase temprana del ciclo de vida del desarrollo del software. Una de sus desventajas es que es una técnica relativamente nueva y unos buenos modelos de amenaza no significan un buen software. (12)

Revisión de código Fuente: Es el proceso de comprobar manualmente el código fuente de una aplicación web en busca de incidencias de seguridad. Muchas vulnerabilidades de seguridad serias no pueden ser detectadas con ninguna otra forma de análisis o prueba. Como dice la conocida frase “si quieres saber qué es lo que está pasando realmente, vete directo a la fuente”. Casi todos los expertos en seguridad están de acuerdo en que no hay nada mejor que ver realmente el código. Toda la información necesaria para identificar problemas de seguridad está en el código en algún lugar. De modo diferente de comprobar software cerrado de terceras partes, como sistemas operativos. (12)

Cuando se realizan pruebas en aplicaciones web (especialmente cuando han sido desarrolladas internamente), el código fuente debería ser puesto a disposición para comprobarlo. Muchos problemas de seguridad no intencionados pero significativos son también extremadamente difíciles de descubrir con otras formas de comprobación o análisis, como la Prueba de Intrusión. Con el código fuente, la persona comprobándolo puede determinar con exactitud qué está pasando (o qué se supone que está pasando), y eliminar el trabajo de adivinar de la comprobación de caja negra.

Una de las ventajas es que es su rapidez para revisores competentes, su eficacia e integridad y también su precisión. Una de sus desventajas es que requiere de desarrolladores de seguridad altamente competentes, no puede detectar errores en tiempo de ejecución con facilidad y que el código fuente realmente en uso puede ser diferente del que está siendo analizado. (12)

Pruebas de Intrusión: Este tipo de prueba se ha convertido desde hace muchos años en una técnica común empleada para comprobar la seguridad de una red (12) .También son conocidos comúnmente como pruebas de caja negra o hacking ético. Las pruebas de intrusión son esencialmente el "arte" de comprobar una aplicación en ejecución remota, sin saber el funcionamiento interno de la aplicación. En muchos casos al encargado de las pruebas se le da una cuenta válida en el sistema. Cuando se realizan pruebas de intrusión en redes y sistemas operativos, la mayoría del trabajo se centra en encontrar y explotar vulnerabilidades conocidas en tecnologías específicas. Dado que las aplicaciones Web son casi todas hechas a medida exclusivamente, la prueba de intrusión en el campo de aplicaciones Web es más similar a investigación pura. (12)

Una Prueba de Intrusión es un método de evaluación de la seguridad de un sistema de ordenadores o una red mediante la simulación de un ataque. Una Prueba de Intrusión de aplicación Web está enfocada solamente a evaluar la seguridad de una aplicación Web. (12)

Las ventajas de este tipo de prueba, puede ser rápido (y por tanto, barato), requiere un conocimiento relativamente menor que una revisión de código fuente y comprueba el código que está siendo expuesto realmente. Una de las desventajas que presenta es que es demasiado tardío en el Ciclo de vida de desarrollo de sistemas (SDLC) y prueba solo de impactos frontales. (12)

1.5.3 Categoría de Pruebas de Intrusión

El proceso conlleva un análisis activo de la aplicación en busca de cualquier debilidad, fallos técnicos vulnerabilidades. Cualquier incidencia de seguridad que sea encontrada será presentada al propietario del sistema, junto con una evaluación de su impacto, y a menudo con una propuesta para su mitigación o una solución técnica.

Prueba de Firma.

La obtención de la firma digital de un servidor Web es una tarea esencial para la persona que realiza una prueba de Intrusión. Saber el tipo y versión del servidor en ejecución le permite determinar vulnerabilidades conocidas, y los exploits apropiados a usar durante la prueba. (12)

Pruebas de Gestión de Configuración de la Infraestructura.

A menudo, los análisis sobre la infraestructura o la topología de la arquitectura pueden revelar datos importantes sobre una aplicación Web. Se pueden obtener datos como por ejemplo el código fuente,

los métodos HTTP permitidos, funcionalidades administrativas, métodos de autenticación y configuraciones de la infraestructura. (12)

Pruebas de SSL/TLS Secure Socket Layer (SSL) y Seguridad para Capa de Transporte (TLS).

Son dos protocolos que proporcionan, con soporte criptográfico, canales seguros para la protección, confidencialidad y autenticación sobre la información que se transmite. Por considerarse críticas estas implementaciones de seguridad, es importante verificar la utilización de un algoritmo de cifrado fuerte, y su correcta implementación (12).

Pruebas del receptor de escucha de la Base de Dato (BD).

Durante la configuración del servidor de base de datos, muchos administradores de bases de datos no tienen en consideración la seguridad del receptor de escucha de la base de datos. Este componente podría revelar información sensible como por ejemplo las configuraciones o las instancias de base de datos en ejecución en caso de encontrarse incorrectamente configurado o analizado mediante técnicas manuales o automáticas. La información mostrada podría ser de gran utilidad a la hora de realizar seguidamente otras pruebas que supongan un mayor impacto. (12)

Pruebas de gestión de configuración de la aplicación

Las aplicaciones Web esconden información que no son generalmente considerados durante el desarrollo o la configuración de la aplicación en sí. Estos datos pueden ser descubiertos en el código fuente, en archivos de registro o en los códigos de error predeterminados de los servidores Web. El planteamiento correcto de este tema es fundamental durante un análisis de seguridad. (12)

Gestión de extensiones de archivo

Las extensiones de los ficheros que se encuentran en el servidor o en una aplicación hacen posible el identificar las tecnologías en las que se basa la aplicación que se quiere analizar, por ejemplo, extensiones jsp y asp. Las extensiones de los ficheros también pueden mostrar sistemas adicionales que se conectan a la aplicación. (12)

Pruebas de Gestión del Caché de navegación y de Salida de sesión

En esta fase, comprobamos que la función de cierre de sesión está correctamente implementada, y que no es posible “reutilizar” una sesión después del cierre. También se comprueba que la aplicación automáticamente cierra la sesión de un usuario cuando ha estado inactivo durante un cierto lapso de tiempo, y que ningún dato sensible permanece en el caché del navegador. (12)

Pruebas de Gestión de Sesiones.

En el núcleo de toda aplicación web se encuentra el sistema en que la aplicación mantiene los estados, y por lo tanto controla la interacción del usuario con el sitio.

La gestión de sesiones cubre ampliamente todos los controles que se realizan sobre el usuario, desde la autenticación hasta la salida de la aplicación. HTTP es un protocolo sin estados, lo que significa que los servidores web responden a las peticiones de clientes sin enlazarlas entre sí.

Incluso la lógica de una simple aplicación requiere que las múltiples peticiones de un usuario sean asociadas entre sí a través de una "sesión". Para ello se necesitan soluciones de terceros a través, o bien de soluciones externas disponibles en el mercado y soluciones de servidor Web, o bien de implementaciones a medida. (12)

Pruebas para CSRF.

Cross-Site Request Forgery (CSRF) trata de forzar a un usuario final a ejecutar acciones no deseadas en una aplicación web en la cual él/ella ya está autenticado. Con un poco de ayuda de ingeniería social (por ejemplo enviando un enlace vía email/chat), un atacante puede forzar a los usuarios de una aplicación web a ejecutar acciones a su antojo. Un exploit CSRF que tenga éxito puede comprometer los datos de un usuario final y sus operaciones en el caso de un usuario normal. Si el usuario objetivo del ataque es la cuenta de administrador, se puede comprometer la aplicación web por completo. (12)

Pruebas de Validación de Datos.

La debilidad más común en la seguridad de aplicaciones web, es la falta de una validación adecuada de las entradas procedentes del cliente o del entorno de la aplicación. Esta debilidad conduce a casi todas las principales vulnerabilidades en aplicaciones, como inyecciones sobre el intérprete, sobre el sistema de archivos y desbordamientos de búffer. (12)

Inyección SQL.

Un ataque de Inyección SQL consiste en la inserción o “inyección” de datos en una consulta SQL desde un cliente de la aplicación. El éxito en una inyección SQL puede leer datos sensibles de la base de datos, modificar los datos (insertar/actualizar/borrar), realizar operaciones de administración sobre la base de datos (como reiniciar la Base de datos del Sistema de Gestión (DBMS)), recuperar el contenido de un archivo del sistema de archivos del DBMS y, en algunos casos, ejecutar órdenes en el sistema operativo. Los ataques de inyección SQL son un tipo de ataques de inyección en los que órdenes SQL son inyectados en texto para afectar la correcta realización de una consulta SQL predefinida. (12)

Inyección Lenguaje de Marcas Extensible (XML).

Se habla acerca de comprobación de inyección XML cuando intentamos inyectar un documento XML en la aplicación: si el intérprete XML (parser) falla al intentar hacer una validación apropiada de los datos, entonces el resultado de la comprobación será positivo. (12)

Pruebas de vulnerabilidad Incubada

Llamadas también a menudo ataques persistentes, este tipo de prueba es compleja, que precisan de más de una vulnerabilidad de validación de datos para funcionar. (12)

Pruebas de Denegación de servicio DOS.

El tipo más común de ataque de Denegación de Servicio (en inglés, Denial of Service, DOS) es del tipo empleado en una red para hacer inalcanzable a la comunicación a un servidor por parte de otros usuarios válidos.

El concepto fundamental de un ataque DOS de red es un usuario malicioso inundando con suficiente tráfico una máquina objetivo para conseguir hacerla incapaz de sostener el volumen de peticiones que recibe. Cuando el usuario malicioso emplea un gran número de máquinas para inundar de tráfico una sola máquina objetivo, se conoce generalmente como ataque denegación de servicio distribuido. Este tipo de ataques generalmente van más allá del alcance de lo que un desarrollador de aplicaciones puede prevenir en su propio código. (12)

1.6 Tendencia actual de la Seguridad de Software en las Aplicaciones Web

Los primeros elementos de Seguridad Informática surgieron a inicios de los años 60 donde por motivos de la guerra con la crisis de los misiles, el incipiente desarrollo de la tecnología y el potencial investigativo que generaban las Universidades fue necesario formular estrategias de seguridad sobre los dispositivos tecnológicos que garantizaban una fortaleza en la defensa y ataque; con vistas de mantener una posición ventajosa en el escenario mundial. (15)

Diseñar sistemas seguros es una tarea compleja, pues las amenazas y los ataques son, en muchos casos, poco cuantificables y muy variados. La aplicación de medidas de seguridad para proteger un sistema supone un análisis y cuantificación previa de los riesgos o vulnerabilidades del sistema.

La seguridad de software se refiere a la seguridad de información como la protección de sistemas de información contra el acceso desautorizado o la modificación de información, si está en una fase de almacenamiento, procesamiento o tránsito. La seguridad de software protege el sistema contra la negación de servicios y provisión a usuarios desautorizados, incluyendo las medidas necesarias para

detectar, documentar y contrariar tales amenazas. Una vez que se identifiquen los riesgos, identificar medidas de seguridad apropiadas llega a ser manejable.

La Seguridad de Aplicaciones cobra cada vez mayor importancia debido a la creciente necesidad de las organizaciones de realizar transacciones en línea. Por ello, proteger las aplicaciones web contra intentos de hacking se ha convertido en necesidad primordial para operar un negocio exitoso en línea. Fallar en el intento implicaría: cuantiosas pérdidas financieras, complicaciones legales e inclusive, podría impactar negativamente la reputación de su organización. (16)

La mayor parte de los presupuestos de seguridad en las compañías están dedicados a la infraestructura de la red y de acuerdo con estudios realizados, más del 60% de los ataques en seguridad de la información no están dirigidos ahí, sino a la capa de aplicaciones web; otro estudio demuestra que más del 90% de estas aplicaciones, son vulnerables a ataques informáticos y para el 2010 el 80% de las organizaciones experimentará un incidente de seguridad. (17)

En la actualidad hay cada vez mayor tendencia a la implementación de aplicaciones Web, debidas entre otros aspectos al desarrollo de internet. Estas son además las aplicaciones más usadas para efectuar el comercio electrónico, por esta y otras razones deben tenerse muy presente, las cuestiones de seguridad en la misma. Varias son las vulnerabilidades que se conocen que afectan a las aplicaciones Web.

1.6.1 Las 10 vulnerabilidades más comunes de una aplicación Web

Existen diez vulnerabilidades fundamentales en las aplicaciones web. Estas se dividen en varios grupos los cuales se mostraran a continuación.

1. **Entrada no validada.** La información de entradas web no es validada antes de ser usadas por la aplicación web. Los agresores pueden usar estas fallas para atacar los componentes internos a través de la aplicación web. Tipo de datos (strings, integer, real, etc…). (18)

   Tipo de datos (strings, integer, real, etc…).

   - Conjunto de caracteres permitidos.
   - Longitud mínima y máxima.
   - Si nulo es permitido.
   - Si el parámetro es requerido o no.
   - Si los duplicados son permitidos.

- El rango numérico.
- Valores específicos permitidos (enumeración).
- Patrones específicos (expresiones regulares).

2. **Control de Acceso Interrumpido.** Las restricciones de aquello que tienen permitido hacer los usuarios autentificados no se cumplen correctamente. Los agresores pueden explotar estas fallas para acceder a otras cuentas de usuarios, ver archivos sensitivos o usar funciones no autorizadas.
3. **Administración de Autenticación y Sesión Interrumpida.** Las credenciales de la cuenta y los tokens de sesiones no están propiamente protegidos. Los agresores que pueden comprometer las contraseñas, claves, cookies de sesiones u otro tokens, pueden vencer las restricciones de autenticación y asumir la identidad de otros usuarios.
4. **Fallas de Cross Site Scripting (XSS).** La aplicación web puede ser usada como un mecanismo para transportar un ataque al navegador del usuario final. Un ataque exitoso puede comprometer el tokens de sesión del usuario final, atacar la máquina local o enmascarar contenido para engañar al usuario.
    - Validación de los scripts de salida.
    - Uso de correo electrónico
5. **Desbordamiento del Búfer.** Los componentes de aplicaciones web en ciertos lenguajes que no validan adecuadamente las entradas de datos pueden ser derribados y en algunos casos, usados para tomar control de un proceso. Estos componentes pueden incluir CGI, bibliotecas, rutinas y componentes del servidor de aplicación web. (18)
6. **Fallas de Inyección.** La aplicación web puede pasar parámetros cuando accede a sistemas externos o al sistema operativo local. Si un agresor puede incrustar comandos maliciosos en estos parámetros, el sistema externo puede ejecutar estos comandos por parte de la aplicación web. (24)
7. **Manejo Inadecuado de Errores.** Condiciones de error que ocurren durante la operación normal que no son manejadas adecuadamente. Si un agresor puede causar que ocurran errores que la aplicación web no maneja, éste puede obtener información detallada del sistema, denegar servicios, causar que mecanismos de seguridad fallen.
8. **Almacenamiento Inseguro.** Las aplicaciones web frecuentemente utilizan funciones de criptografía para proteger información y credenciales. Estas funciones y el código que integran a ellas han sido difíciles de codificar adecuadamente, lo cual frecuentemente redunda en una protección débil.

9. **Negación de Servicio.** Los agresores pueden consumir los recursos de la aplicación web al punto de que otros usuarios legítimos no puedan ya acceder o usar la aplicación. Los agresores también pueden dejar a los usuarios fuera de sus cuentas y hasta causar que falle una aplicación entera.
10. **Administración de Configuración Insegura.** Tener una configuración de servidor estándar es crítico para asegurar una aplicación web. Estos servidores tienen muchas opciones de configuración que afectan la seguridad y no son seguros desde la instalación original del software. (18)

Para mantener la seguridad de la información contenida en las aplicaciones es necesario garantizar la confidencialidad, integridad y disponibilidad, que todo esto se resume en la protección de las aplicaciones

Confidencialidad: La confidencialidad es la propiedad de prevenir la divulgación de información a personas o sistemas no autorizados.
Es un requisito importante, que los sistemas de software, garanticen el acceso a los recursos, por parte de los usuarios que hacen uso de ellos, de manera controlada. Esto se debe a que muchos procesos de negocios manejan información que puede ser de uso confidencial tales como números de cuentas bancarias, resultados médicos y científicos y donde un ataque contra la confidencialidad puede desencadenar en ganancias para unos y pérdidas para otros.

Integridad: Para la seguridad de la información, la integridad es la propiedad que busca mantener a los datos libres de modificaciones no autorizadas.
La violación de integridad se presenta cuando un empleado, programa o proceso modifica o borra los datos importantes que son parte de la información. Es un riesgo común que el atacante al no poder descifrar un paquete de información y conociendo su relevancia, simplemente lo intercepte y lo borre. Existen mecanismos que intentan mitigar4 ataques de este tipo, algunos están relacionados con la inclusión de firma electrónica en los documentos digitales.

Disponibilidad: La disponibilidad es la característica, cualidad o condición de la información de encontrarse a disposición de quienes deben acceder a ella, ya sean personas, procesos o aplicaciones.
En el caso de los sistemas informáticos utilizados para almacenar y procesar la información, los controles de seguridad utilizada para protegerlo, y los canales de comunicación protegidos que se utilizan para acceder a ella deben estar funcionando correctamente. La alta disponibilidad de sistemas debe estar presente en todo momento, evitando interrupciones del servicio debido a cortes de energía, fallos de hardware, y actualizaciones del sistema. Garantizar la disponibilidad implica también la prevención de ataques y denegación de servicio.

La Industria Cubana del Software está llamada a convertirse en una significativa fuente de ingresos para Cuba. Se encuentra desempeñando una ardua labor en la calidad para los productos informáticos, con el objetivo de cumplir las expectativas del usuario, como proceso importante durante el ciclo de vida del software, donde el objetivo principal es el cliente.

Entre las empresas cubanas que se desarrollan en esta área se encuentran la empresas Segurmática y Softel, la primera dedicada a la creación de antivirus para proteger los sistemas informáticos cubanos, y la segunda dedicada a la producción y comercialización de software, desarrollando sistemas informáticos en una amplia gama de aplicaciones como el turismo y la medicina.

Desoft es otra de las empresas cubanas dedicada a prestar servicios de desarrollo, producción y comercialización de software. Esta empresa tiene una oficina central y una sede en cada una de las provincias de Cuba.
Dentro de esta esfera de la producción de software se enmarca la UCI, la cual no sólo ha tenido como prioridad la informatización de Cuba sino también la exportación de productos.

La UCI pretende ser la vanguardia del desarrollo de las empresas de software en Cuba y de llevar la informatización a todos los sectores de la sociedad: Salud, Educación, Cultura, Deporte, Turismo, Prensa, etc. Regir y propiciar un avance tecnológico y de la industria del software en Cuba y convertir la industria del software en un renglón fundamental de la economía e insertarnos en el mercado internacional, por lo que el reto de la universidad es producir software de alta calidad, logrando así un software con una alta seguridad. El aspecto de la seguridad en el software es todavía un aspecto muy débil en la UCI. Esta cuenta con un grupo de calidad a nivel central, como también uno en cada facultad. Los procesos de calidad que se llevan a cabo en la UCI no son suficientes para lograr que los productos obtengan la seguridad requerida.

En la Universidad la seguridad en los proyectos productivos es un aspecto que tiene una singular importancia, debido a los riesgos y daños que se pueden ocasionar en el software. Una de las vías para verificar la seguridad en los proyectos es teniendo solo acceso a los servidores físicos aquellas personas autorizadas y se deben cambiar las contraseñas cada semana aproximadamente, para con esto evitar la filtración de la misma. Además cada persona que interviene en el equipo de desarrollo del software es responsable de velar por la seguridad en la parte que atiende dentro del proyecto.

1.7 Tendencias actuales sobre desarrollo de pruebas: Uso de las herramientas para pruebas automatizadas de seguridad

Actualmente el desarrollo de las tecnologías de la Información y las Comunicaciones crece vertiginosamente y se hace necesario el uso de herramientas potentes que verifiquen la calidad de los productos que salen al mercado mundial, para garantizar su total funcionamiento.

Las herramientas para pruebas de software ayudan a los equipos de desarrollo de software a investigar los errores de software, verificar la funcionalidad de los sistemas y asegurarse de que el software que desarrollan es seguro y confiable.

Existen herramientas especiales para cada una de las etapas de un proyecto de desarrollo de software. Algunos proveedores de este tipo de herramientas ofrecen una serie integrada que da soporte tanto a las pruebas como al desarrollo de software durante todo un proyecto, desde que se reúnen los requisitos hasta que se inicia el funcionamiento en vivo del sistema. Sin embargo, hay otros proveedores que se concentran en una parte del ciclo de desarrollo de aplicaciones.

Riesgos de las herramientas para pruebas de software

- Las pruebas automatizadas pueden generar cantidades enormes de datos sin procesar. La herramienta que seleccione debe ser capaz de transformar estos datos para facilitar su manipulación y procesamiento.
- Hay que tener extremo cuidado de no pasar por alto la funcionalidad de inyección de fallos, que puede servir para detectar las debilidades que no son aparentes con las rutas de código típicas -particularmente, las rutas que son propensas a sufrir ataques a la seguridad
- Seleccionar la herramienta de software incorrecta presenta dos desventajas importantes; por un lado, se gastan tiempo y recursos, y por el otro, se genera un obstáculo que sus equipos no podrán resolver fácilmente. (12)

1.7.1 Herramientas para el entorno de pruebas de Intrusión

En el mundo existen varias herramientas para la automatización de pruebas de seguridad, como son: **Nessus**, **Brutus**, **Shadow Security Scanneres (SSS)**, **Nikto**, **Paros**, **WebScarab**.

Nessus: Fue desarrollado inicialmente por SwRI para la NASA para realizar el análisis probabilístico de los componentes del transbordador espacial motor principal. SwRI continúa para desarrollar y aplicar nessus para una amplia gama de problemas, incluyendo estructuras aeroespaciales, las estructuras del automóvil, la biomecánica, los motores de turbina de gas, geomecánica, el envasado de los residuos nucleares, estructuras costa afuera, ductos y rotordynamics. Para lograr esto, los

códigos se han interconectado con muchos conocidos programas de análisis de terceros y comercial determinista. (19)

Nessus es una gran herramienta diseñada para automatizar las pruebas y el descubrimiento de problemas de seguridad conocidos. Normalmente, una persona, un grupo de hackers, una empresa de seguridad, o un investigador descubren un modo específico de violar la seguridad de un producto de software. El descubrimiento puede ser accidental o por medio de la investigación dirigida, la vulnerabilidad, en los distintos niveles de detalle, se libera a la comunidad de seguridad. Nessus ha sido diseñado para ayudar a identificar y resolver estos problemas conocidos, antes de que un hacker se aproveche de ellos. Nessus es una gran herramienta con muchas posibilidades. (20)

Nessus es un programa gratuito liberado bajo la Licencia Pública General (GPL). Históricamente, muchos en el mundo empresarial han ridiculizado software de dominio público, tales como una pérdida de tiempo, en lugar de elegir "el apoyo" productos desarrollados por las empresas establecidas. (20)

Una de las características muy poderosas de Nessus es su tecnología cliente / servidor. Los servidores pueden ser colocados en varios puntos estratégicos en una red que permite las pruebas que se realizaron desde distintos puntos de vista. Un cliente central o varios clientes distribuidos pueden controlar todos los servidores. La parte servidor se ejecutará en la mayoría de cualquier sabor de Unix. Incluso funciona en Mac OS X e IBM AIX / Linux, pero tiende a hacer la instalación más simple. Estas características proporcionan una gran flexibilidad para el probador de penetración. Los clientes están disponibles tanto para Windows y Unix. El servidor de Nessus realiza la prueba real, mientras que el cliente proporciona la configuración y funcionalidad de informes. (20)

Nessus utiliza plugins que son pequeños programas (también llamados exploits) que se aprovechan de un fallo en el diseño para conseguir entrar al sistema. Un exploits clásico se basa en un desbordamiento del buffer. Nessus soporta IMAP, SMTP y POP3, cuenta con una librería COM de fácil uso. Algunas de las pruebas de vulnerabilidades de Nessus pueden causar que los servicios o sistemas operativos se corrompan y caigan. Permite generar reportes en HTML, XML, LaTeX, y texto ASCII; también sugiere soluciones para los problemas de seguridad. Nessus es un escáner de seguridad remoto para Linux, BSD, Solaris y Otros Unix. (12)

Nikto: Es un escaneador de scripts CGI excelente. Nikto tiene la capacidad de no sólo probar vulnerabilidades de CGI sino también que lo hace de forma evasiva, evitando los sistemas de detección de intrusos. Viene con una documentación muy completa, la cual es recomendable revisar antes de ejecutar el programa. Si sus servidores web están sirviendo scripts CGI, Nikto puede ser un recurso excelente para chequear la seguridad de estos servidores. (21)

Nikto es un excelente punto de partida para comprobar la seguridad del servidor web, que funciona tanto en Linux como en Windows y tiene una gran base de datos de ataques (CGI y otros) en 230 tipos de servidores distintos. Nikto puede ser usado no solamente como herramienta escaneadora de puertos, si no como una herramienta de seguridad. Se usa para evaluar la seguridad de sistemas informáticos, así como para descubrir servicios o servidores en una red informática. (21)

Nikto no está diseñado como una herramienta excesivamente cuidadosa. Se pondrá a prueba un servidor web en el menor tiempo posible, y es bastante obvio en los archivos de registro. Sin embargo, hay un apoyo para la lucha contra LibWhisker los métodos de IDS en caso de que quiera darle una oportunidad (o prueba de su Sistema de detector de intruso (IDS)). (21)

Brutus: Es una herramienta de hacking. Estos programas, que son herramientas legítimas y útiles cuando son utilizadas adecuadamente, pueden ser empleadas de forma malintencionada por los hackers. Brutus es un programa que permite averiguar contraseñas con el método de la fuerza bruta, es decir, probando secuencialmente cada una de las diferentes posibilidades existentes, hasta dar con la correcta. (22)

Brutus soporta múltiples protocolos de autenticación de usuarios, como por ejemplo POP3, HTTP, FTP**,** SMB, etc. Brutus no implica ningún riesgo por sí mismo, pero puede ser empleado para obtener contraseñas de forma ilícita. (22)

Paros: Paros es una herramienta especializada en la seguridad web que hace poco presentó su versión 3.2.3**.** Es un proxy y scanner de vulnerabilidades. Permite a los usuarios interceptar, modificar y analizar los datos HTTP y HTTPS que se intercambia entre el servidor web y el navegador cliente. Diferentes módulos permiten grabar la navegación de un usuario por un sitio y más tarde un scanner intenta realizar diferentes ataques (XSS, etc.) para detectar problemas de seguridad típicos de aplicaciones web. (23)

Está programado en Java pero puede usarse para analizar aplicaciones web realizadas con cualquier tecnología, como la Licencia Clarified Artistic License. Proxy Paros es una aplicación para evaluación de vulnerabilidades sobre aplicaciones web. Consiste en un proxy realizado en Java que permite visualizar en tiempo real los paquetes HTTP/HTTPS y ver los elementos que se están editando o modificando, como las cookies y campos de formularios. Además, incluye un registro de tráfico, calculadora de hash y un escáner para probar ataques comunes a aplicaciones web como XSS (cross-site-scripting) e inyección de SQL. (23)

Shadow Security Scanneres (SSS): Es una herramienta que se encarga de escanear y analizar las vulnerabilidades de un sistema operativo; funciona en Windows, Unix, Linux, FreeBSD, OpenBSD y Net BSD. Lo análisis son muy rápidos y se ofrecen informes de cada una de las vulnerabilidades y

sus posibles soluciones, además de actualizarse automáticamente a través de Internet. Además, escanea los siguientes componentes: NetBIOS, HTTP, CGI y WinCGI, FTP, DNS, DOS vulnerabilidades, POP3, SMTP, LDAP, TCP/IP, UDP, registro, servicios, usuarios y cuentas, password vulnerabilidades, MSSQL, IBM BD2, Oracle, MySQL, Postgress SQL, Interbase, MiniSQL.30). (24)

WebScarab: Webscarab es un marco de trabajo para analizar aplicaciones web que se comunica usando los protocolos HTTP y HTTPS. Está escrito en Java, por lo que es portable a muchas plataformas. WebScarac tiene muchos modos de operación, implementados por varios plugins. Su uso más común es operar WebScarab como un proxy de intercepción, que permite al operador revisar y modificar las peticiones creadas por el navegador antes de que sean enviados al servidor, y para revisar y modificar respuestas enviadas por el servidor antes de que sean recibidas por el navegador. Webscarab es capaz de interceptar comunicación en HTTP y HTTPS. El operador puede también revisar las conversaciones (peticiones y respuestas) que hayan pasado por WebScarab. Es una herramienta diseñada principalmente para ser usada por personas que pueden escribir código por ellos mismos, por lo menos tienen un muy buen conocimiento el protocolo HTTP. (25)

W3af: Entre las herramientas automatizadas se encuentra w3af, desarrollado por Andrés Riancho, y que testea un Localizador Uniforme de Recurso (URL) en busca de las vulnerabilidades más conocidas; SQL Injection, XSS, CSRF, osCommanding y muchos más, es herramienta libre para la auditoría de seguridad de páginas web. (26)

W3af: Tiene como Licencia GPLv2 y fue desarrollado en Desarrollado en Python. Sus Funcionalidades Principales son:

- Scannings Automáticos
- Envío de Peticiones HTTP “artesanales”
- Codificador / Decodificador
- Comparador
- Proxy Local (27)

1.7.2 El papel de las herramientas automatizadas

En el mundo existen varias empresas que comercializan herramientas de análisis y comprobación de seguridad automatizadas. Es necesario recordar las limitaciones de estas herramientas, de modo que puedan emplearlas para aquello en lo que son más útiles.

Es muy importante tener en cuenta que estas herramientas no están diseñadas para un código específico, sino para aplicaciones en general. Lo que significa que aunque pueden encontrar algunos

problemas genéricos, no tienen el conocimiento suficiente sobre la aplicación como para permitirles detectar la mayoría de los fallos. Las incidencias de seguridad más serias son aquellas que no son genéricas, sino profundamente intrincadas en la lógica de negocio y diseño a medida de la aplicación. Estas herramientas son ciertamente parte de un programa de seguridad de aplicaciones bien equilibrado. Utilizadas sabiamente, pueden ofrecer soporte a procesos globales para producir código más seguro.

1.8 Selección de las herramientas para las Pruebas Intrusión.

Tras realizar la investigación de las herramientas existentes en el mundo que apoyan el proceso de la realización de las Pruebas de Intrusión, se llegó a la conclusión de utilizar como herramientas documentada la lista de chequeo y como automatizadas Nikto y W3af, por las características que estas presentan.

Las herramientas Nikto y W3af las características más importantes que están poseen es que son libres, son fáciles de manejar. Entre las herramientas que existen para la automatización de las pruebas de Intrusión w3af es una de las más completas de todas. Nikto y W3af aparecen en el repositorio de Linux**.**

1.9 Propuesta del procedimiento de solución de las Pruebas de Intrusión

Después de todo lo estudiado, se comprobó que no existe un procedimiento para aplicar Pruebas de Intrusión en el GCF por lo que se propone que se confeccione el procedimiento, con el apoyo del Modelo de Proceso del Servicio de Pruebas de Software a nivel de sistema en FORTES. Para el mismo, es importante hacer uso de las herramientas automatizadas de Nikto y W3af por las características que estas presentan para automatizar esta prueba y como herramientas documentadas, la lista de chequeo.

Conclusiones del capitulo

En el presente capítulo se han descrito los elementos teóricos sobre los cuales se sustentará el procedimiento para Pruebas de Intrusión en aplicaciones. Se definieron los conceptos generales de pruebas, sus tipos y su automatización. Se realizó un estudio profundo sobre las diferentes herramientas automatizadas para aplicar Pruebas de Intrusión. A partir de las características de cada una de las herramientas descritas, que son compatibles para el desarrollo del objetivo de la presente investigación, se deciden el empleo de las herramientas Nikto y W3af, adaptándolo a las características del centro FORTES. En el siguiente Capítulo se detalla en detalles el procedimiento para las Pruebas de Intrusión en el GCF.

CAPÍTULO II: "PROCEDIMIENTO PARA PRUEBAS DE INTRUSIÓN"

Introducción

Las pruebas de seguridad que se realizan al software tienen un impacto importante en la calidad del producto final, por esta razón en el presente capítulo se ofrece un procedimiento para garantizar la realización de Pruebas de Intrusión en aplicaciones web. Esta idea surge por la necesidad que hay en estos momentos en el Grupo de Calidad FORTES, los cuales no cuentan con la realización de pruebas de seguridad, sino que realizan solo pruebas funcionales. Dicha propuesta será implementada por primera vez en el en el GCF, donde el software llega completamente elaborado.2.1 Pruebas y herramientas que se van a incluir en el procedimiento.

2.1 Pruebas y herramientas que se van a incluir en el procedimiento

A continuación la siguiente tabla se muestra los tipos de pruebas de Intrusión y la herramienta.

Tabla #1: Tipos de pruebas que se emplean en el procedimiento de Prueba de Intrusión.

Categoría de Pruebas	Tipo de prueba	Breve Descripción del tipo de prueba	Herramienta
Recopilación de Información	Pruebas de firma digital.	La determinación de la firma digital de aplicaciones es el primer paso del proceso de recopilación de información.	**W3af**
	Descubrimiento de Aplicaciones.	Es el proceso destinado a identificar aplicaciones web instaladas en una infraestructura dada.	**W3af**
	Análisis de Códigos de Errores.	El factor más importante para esta actividad es enfocar la atención en los errores encontrados en las aplicaciones.	**W3af**
Pruebas de Gestión de Configuración	Pruebas SSL/TLS (SSL Versión, Algoritmos, longitud de Claves, Validez de Certificado Digital).	SSL y TLS son dos protocolos que proveen, con el apoyo de criptografía, para la protección, confidencialidad etc, de la información transmitida.	**Nikto**
Pruebas de	Fuerza bruta.	Consiste en averiguar el usuario y contraseña válidos de un individuo	**W3af**

Autenticación		registrado en el sistema, mediante el intento reiterado de diferentes combinaciones de usuarios y contraseñas.	
	Recordatorio de contraseñas y reset.	Esta prueba usa solo características funcionales de la aplicación y código HTML que siempre está disponible al cliente.	**W3af**
Pruebas de Validación de Datos	Inyección Structured Query Language (SQL).	El objetivo es manipular los datos en la base de datos, un recurso vital para todas las empresas.	**W3af**
	Inyección ObjectRelationalMappingORM(herramienta para mapear objetos relacionales).	La inyección ORM es un ataque donde se utiliza inyección SQL contra un modelo de objeto de acceso a datos generado por ORM.	**W3af**
	Inyección Protocolo de acceso a Directorios Ligeros (LDAP).	Este ataque permite alterar los contenidos de las sentencias LDAP que se generan a través de entradas de la aplicación web.	**W3af**
	Inyección Extensible MarkupLanguageExtensibleMarkupLanguage XML	Hablamos de pruebas de inyección XML cuando tratamos de inyectar un determinado documento XML en la aplicación.	**W3af**
	Inyección *Seguridad* de Ingreso Suplementario SSI (directivas evaluadas por el servidor web antes de servir la página al usuario).	Son unas extensiones muy simples que pueden permitir a un atacante inyectar código dentro de páginas html, o incluso realizar ejecución remota de código.	**W3af**
	Inyección de Código	Estas pruebas pueden tener como	**W3af**

		objetivos diversos motores de scripting del lado del servidor, como pueden ser ASP o PHP. Para protegerse frente a estos ataques será preciso emplear unas medidas adecuadas de validación y programación segura	

Para más información sobre cómo se trabaja con estas dos herramientas ver los manuales de usuario que se elaboraron. En el Anexo 1 para w3af y el Anexo 2 para Nikto.2.2 Procedimiento de Pruebas de Intrusión en el Grupo de Calidad de FORTES

Descripción: Este procedimiento definirá de forma detallada cinco pasos para llevar a cabo las Pruebas de Intrusión.

Alcance: Es aplicable para ser utilizado por los probadores del Grupo de Calidad de FORTES.

2.2.1 Cinco tareas fundamentales para el procedimiento de pruebas de Intrusión.

1. **Atención a las solicitudes de Pruebas**: Revisión de la solicitud y las acepta o la rechaza de la solicitud.
2. **Planificación de las pruebas:** Coordinar el comienzo de las pruebas. Se desarrollará un plan de Pruebas de Intrusión y capacitación para la ejecución de las pruebas.
3. **Diseño de las pruebas:** Diseñar las listas de chequeo y los casos de prueba según las categorías de las pruebas de Intrusión que se vallan a utilizar.
4. **Ejecución de las Pruebas:** Ejecutar las pruebas mediante el uso de las listas de chequeo y los casos de prueba en cada iteración utilizando las herramientas Nikto y W3af. Cierre de iteración.
5. **Cierre de las Pruebas:** Documentar y realizar un informe de las no conformidades encontradas en cada iteración. Solución de los errores encontrados.

Modelo del procedimiento de Pruebas de Intrusión en el (GCF)

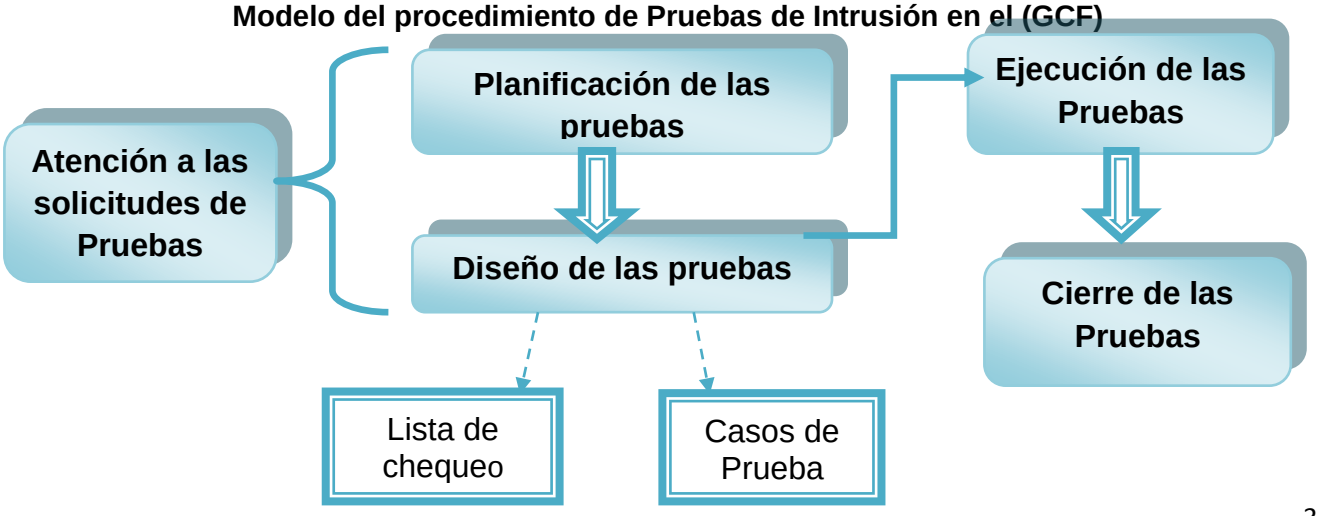

Figura 1: Estructura de las actividades del procedimiento propuesto.

En la figura siguiente se muestra cómo se desarrollan las fases del procedimiento.

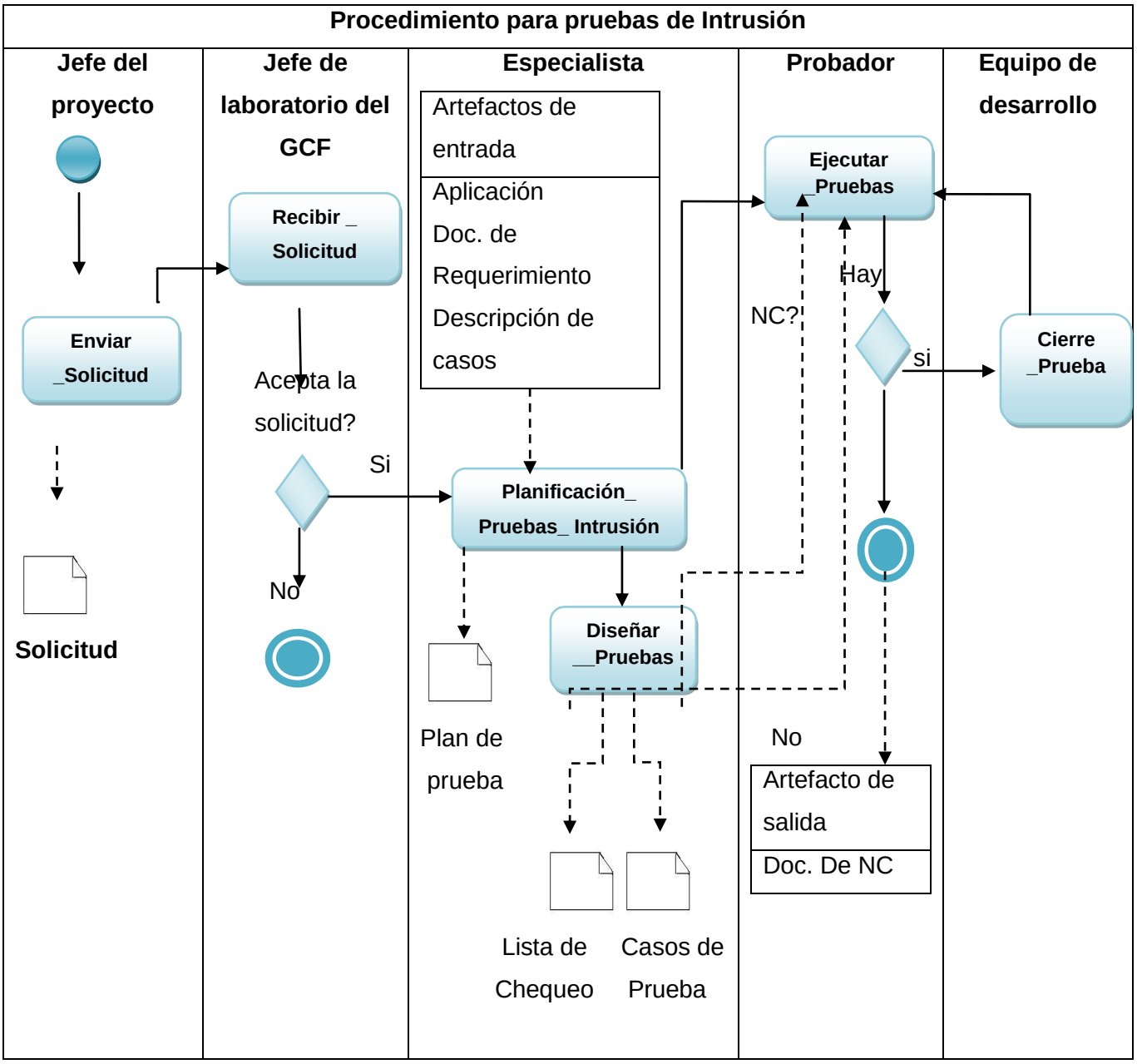

Figura 2: Flujo gráfico del procedimiento

2.2 Descripción de las actividades del procedimiento

Artefactos de entrada: Los siguientes artefactos son los que debe proporcionar el equipo de desarrollo para ser utilizados en la ejecución de las pruebas.

- Aplicación.

- Reglas del Negocio
- Requisitos Funcionales.
- Requisitos no Funcionales de Seguridad.

2.2.1 Paso #1: Atención a las Solicitudes de Pruebas

Cuando un proyecto solicita el servicio de pruebas al GCF, esto tiene que ser a través de la dirección del Grupo de Calidad. Este tiene que llevar una Plantilla de Solicitud de Pruebas y junto ella, todos los documentos y elementos necesarios para la realización de las pruebas. El jefe del Laboratorio de calidad FORTES revisa y este rechaza la solicitud o la acepta. Si este la acepta le hace una reunión de Inicio de Pruebas y en caso que no acepte la solicitud, deben volver a realizar la solicitud.

Para esta actividad se tomó en cuenta la atención de solicitud del Modelo de Proceso del Servicio de Pruebas de Software a nivel de sistema en FORTES.

En la siguiente tabla se muestra los roles que intervienen en este paso y su responsabilidad

Tabla #2: Roles y responsabilidad de la actividad Atención a la Solicitud de Prueba.

Roles que intervienen	Responsabilidad
Jefe de Proyecto. (Representante por el Equipo de Desarrollo).	Hacer la Solicitud.
Jefe del Laboratorio de Pruebas.	Recepcionar las solicitudes.

Documentación que interviene

- Solicitud de Pruebas.
- Listas de chequeo.
- Casos de Pruebas.
- Manuales de usuarios.
- Casos de uso.

2.2.2 Paso #2: Planificación de las Pruebas

En este paso se planificarán que tipos de categorías de las pruebas de Intrusión se van a realizarán a la aplicación, así como los roles y los recursos que intervendrán en la ejecución de las pruebas. Se desarrollará el plan de pruebas con el propósito de explicar el alcance, enfoque, recursos requeridos, responsables y manejo de riesgos de un proceso de pruebas.

Para el diseño del plan de prueba se va a utilizar la plantilla “Plan de pruebas”. El plan de prueba identificará los elementos de prueba, los recursos necesarios para la ejecución de las pruebas, se va a definir y recomendar las estrategias de prueba y el cronograma de las pruebas.

Introducción:

Este documento se confecciona con el objetivo de definir el plan de las Pruebas de Intrusión a la aplicación.

1.1- Objetivos

El objetivo de este Plan de Pruebas de Intrusión consisten en identificar los elementos de pruebas, identificar los recursos y configuraciones necesarias para llevar a cabo las pruebas, describir y recomendar las estrategias de las pruebas a ser empleadas y otro objetivo importante es definir el cronograma de las pruebas.

1.2- Alcance

Este procedimiento va a ser utilizado por los probadores del GCF para realizar Pruebas de Intrusión a las aplicaciones.

1- Recursos

2.1- Roles y Responsabilidades

A continuación, mediante de una tabla se muestra los roles y responsables que serán empleados en la realización de las pruebas.

Tabla #3: Descripción de la Actividad: Roles y Responsabilidades.

ROL	Cantidad	Responsabilidad
Planificadora del Servicio de Pruebas de Software a nivel de Sistema del GCF	Total Planificadora del Servicio de Pruebas de Software a nivel de Sistema del GCF	Elaborar lista de chequeo para realizar las pruebas de Intrusión. Elaborar el plan de pruebas. Supervisar el trabajo de pruebas, recogiendo las no conformidades para elaborar el informe de no conformidades. Controlar, Monitorear y Ejecutar el plan de pruebas. Evaluación del proceso de pruebas y los resultados de las mismas.
Probadores del GCF.	Total de probadores del equipo de pruebas del GCF	Ejecutar las pruebas. Registrar no conformidades

Jefe del proyecto o (Representante por el Equipo de Desarrollo)	Total de miembros de desarrolladores del proyecto a revisar relacionados con las pruebas.	Verificar que el escenario de prueba concuerda con el de la aplicación. Instruir al equipo de prueba sobre el ambiente de la aplicación. Recepcionar el informe de no conformidades para dar respuesta a cada una de las no conformidades encontradas.

2.2- Escenario de pruebas

Se colocan las especificaciones de hardware necesarias para las pruebas y una imagen que represente la configuración, puede ser un diagrama de despliegue, esta responsabilidad es del equipo de desarrollo.

2.3- Recursos de Sistema:

Tabla #4: Recursos del sistema que contiene el GCF

Recursos	Tipo
Servidor de Base de Datos.	[Características del software. Ej: Mysql 5.0.2.]. [Características de hardware. Ej: 4: 4 Procesadores a 1.6 GHz. 24 GB de Memoria RAM. Dos discos duros de 72GB a 7200 rpm en RAID1. Tarjeta de Red de 10/100/1000 Gigabit.].
Servidor de Aplicación.	[Características de software Ej: Apache 6.2.1.] [Características de hardware Ej: 2 Procesadores a 3 GHz 8 GB de Memoria RAM. Dos discos duros de 72GB a 7200 rpm en RAID1. Tarjeta de Red de 10/100/1000 Giga bit.]

PC Cliente para pruebas.	[Características de hardware]
Requerimientos especiales	Algún periférico, necesario para las pruebas ej.: impresora, escáner, cámara digital, etc.
Red o subred.	Tipo de red e: inalámbrica, TCP/IP.

Estrategia de las pruebas: A continuación se describirá el flujo de trabajo que será implementado durante todo el período de ejecución de las pruebas, de igual forma se detalla las diferentes estrategias que en cada una de las etapas y fases serán realizadas.

Descripción del flujo de trabajo

El flujo de trabajo se inicia cuando los especialistas comienzan a diseñar las listas de chequeo y los casos de pruebas en correspondencia del la categoría de Prueba de Intrusión a utilizar. Luego si todas las condiciones están creadas se pasa a la 1ra iteración de pruebas, se realizan las pruebas. En la medida que detecten errores en el trabajo con el producto, estas serán anotadas. Al finalizar el día estas no conformidades son revisadas por el especialista de pruebas. Al concluir la iteración de prueba se realizará el Informe Final de Resultados, atendiendo a los Partes diarios.

Una vez aprobado es entregado al Líder del proyecto para que este comience a ejecutar los arreglos acordados.

En la figura se muestra de forma general el flujo de trabajo.

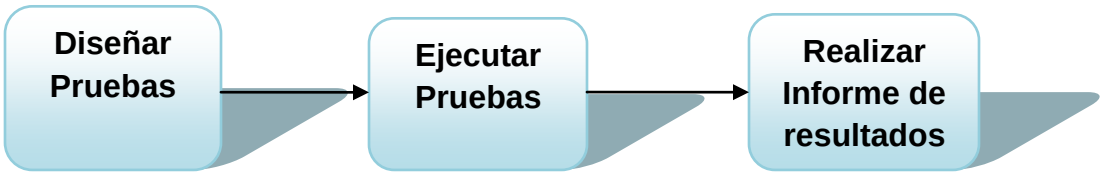

Figura 3: Descripción del Flujo de Trabajo.

Descripción de las estrategias y tipos de pruebas

Todas las pruebas se realizarán de forma manual o utilizando herramientas en dependencia del tipo de pruebas.

- **Primera Fase**

  Organización del escenario de pruebas, capacitación del equipo de pruebas, diseño de los casos de pruebas y elaboración de Listas de Chequeo.

- **Segunda Fase:**

- Realización de la categoría de pruebas **Recopilación de la Información**. Su objetivo es verificar si la aplicación brinda datos sensibles que puedan ser utilizados por cualquier atacante.
- Realización de la categoría de pruebas de **Gestión de Configuración.** Su objetivo es verificar que la información sea confiable y este bien protegido.
- Realización de la categoría de pruebas **Comprobación de la Autenticación**. Su objetivo es poner a prueba el sistema de autenticación de la aplicación.
- Realización de la categoría de pruebas **Validación de Datos**. Su objetivo es verificar que todas las entradas de datos estén validadas.

Cronograma de Trabajo

En la siguiente Tabla se muestra de forma general el Cronograma de trabajo.

Tabla #5: Descripción de la actividad: Cronograma de Trabajo

No.	Tarea	Fecha	Participantes	Iteración	Observaciones
	Actividad a desarrollar.	Fecha de inicio y fecha de fin de la actividad.	Responsable		
1					
2					

2.2.3 Paso #3: Diseño de las Pruebas

En esta actividad se diseñarán los casos de pruebas correspondientes a la categoría de la prueba Validación de Datos. Además, se elaborarán las listas de chequeo que se utilizarán en las categorías de pruebas Recopilación de Información, Gestión de Configuración y Comprobación de la Autenticación. La confección de las listas de chequeo va en correspondencia de las características del proyecto en revisión.

Tabla #6: Roles y responsabilidad de la actividad Diseño de las pruebas.

Roles que intervienen	Responsabilidad

Planificadora del Servicio de Pruebas de Software a nivel de Sistema del GCF	Confecciona las lista de chequeo y los casos de uso en correspondencia del las categoría de las pruebas de Intrusión.

A continuación se muestra la estructura de las listas de chequeo y casos de prueba que se proponen para cada una de las pruebas. Estos artefactos de pruebas son elaborados por el o los especialistas que estén al frente de las pruebas.

Lista de chequeo para la categoría de Recopilación de Información y Pruebas de Gestión de Configuración

A continuación, en la siguiente tabla se muestra una lista de chequeo para la categoría de la prueba de Recopilación de Información y Gestión de Configuración que serán empleadas en la realización del procedimiento propuesto.

Tabla #7: Descripción de la actividad: Lista de chequeo para la categoría de Recopilación de Información y Pruebas de Gestión de Configuración

Indicadores a Evaluar	Descripción	Resultado Esperado	Resultado Real	Herramienta
1. Se puede obtener la firma digital (tipo y versión) del servidor web	Este es el primer paso del proceso de recopilación de información, saber la versión y tipo de servidor web en ejecución, permite a las personas realizando la prueba, determinar vulnerabilidades conocidas, y los exploits adecuados a emplear durante las pruebas.	Se pone el tipo y la versión del servidor web que está en el plan de prueba.		W3af
2. Se identifican más aplicaciones web instaladas en el servidor web	Es el proceso destinado a identificar aplicaciones web instaladas en una infraestructura dada.	Ver las aplicaciones que están instaladas en el servidor		W3af

3. Se identifican todos los puertos abiertos en el IP del servidor y los servicios asociados a esos puertos	Es el proceso para encontrar que aplicaciones específicas se encuentran instaladas en un servidor a partir de todos los puertos abiertos en el IP del servidor y los servicios asociados.	Ver los puertos abiertos en el IP del servidor.		W3af
4. El código de error de la aplicación muestra información sobre el sistema operativo o base de datos del servidor web	Durante la configuración de un servidor de base de datos, muchos administradores de BBDD no toman en consideración adecuadamente la seguridad del componente receptor de escucha de la base de datos.	Ver el código error de la aplicación		W3af
5. Existen cifrados (SSL/TSL) débiles	Estos son dos protocolos que proveen, con el apoyo de criptografía, de canales de transmisión de datos seguros, para la protección, confidencialidad y autenticación de la información transmitida.	Verificar la confidencialidad y autenticación de la información.		Nikto

Lista de chequeo para la Comprobación de la autenticación

El objetivo de esta lista de chequeo es probar el sistema de autenticación de la aplicación en ella prueban los siguientes objetivos.

- Obtener un psw e id de un usuario privilegiado de la aplicación.
- Saltarse el sistema de autenticación mediante la petición directa de páginas.
- Probar el sistema de recordatorio de contraseña que brinda la aplicación.

A continuación en la siguiente tabla se muestra una lista chequeo para la categoría Comprobación de la autenticación que serán empleadas en la realización del procedimiento propuesto

Tabla #8: Descripción de la actividad: Lista de chequeo para la Comprobación de la autenticación

Indicadores a Evaluar	Descripción	Resultado Esperado	Resultado Real	Herramienta
1. Puede obtenerse mediante el ataque de fuerza bruta el usuario y la contraseña de un usuario privilegiado en la aplicación.	Consiste en averiguar el usuario y contraseña válida de un individuo registrado en el sistema.	Obtener el usuario y la contraseña de un usuario privilegiado en la aplicación.		W3af
2. Puede saltarse el sistema de autenticación mediante la petición directa de páginas	La clave para explotar con éxito y saltarse un sistema de autenticación de contraseña es encontrar una pregunta o conjunto de preguntas que ofrezcan la posibilidad de encontrar las respuestas fácilmente.	Saltarse el sistema de autenticación		no
3. El sistema envía la contraseña por correo sin hacerle alguna pregunta	Enviar la contraseña (o un enlace al reset de la contraseña) a la dirección de email del usuario sin realizar primera una pregunta	Enviar la contraseña por correo mediante el sistema.		no

	secreta significa confiar al 100% en la seguridad de esa dirección de email, algo que no es adecuado si la aplicación requiere de un nivel de seguridad alto.			
4. Se le realizan al usuario dos o más preguntas	A menudo un sistema de reset ofrece la elección entre varias preguntas; esta es una buena señal para el posible atacante, porque le ofrece opciones.	Realizar más de dos preguntas		no
5. Después de responder las preguntas la aplicación le envía la contraseña al usuario por correo	Se obtiene el mejor nivel de seguridad si la contraseña se envía mediante un e-mail a la dirección con la que el usuario se registró inicialmente, esto obliga al atacante a no solo adivinar a que dirección e-mail se envió la contraseña, sino también comprometer esa cuenta para tomar control del acceso de la víctima sobre la aplicación.	Una vez respondida las preguntas la aplicación envía los datos por correos.		no
6. Se pueden responder las	Busca preguntas que tengan pocas	Responder las preguntas a		no

preguntas a través de una búsqueda en internet o mediante un ataque de ingería social	opciones posibles, estas preguntas presentan al atacante con una lista de corta de respuestas entre estas adivinar la correcta y basándose en estadísticas el atacante podría clasificar las respuestas de la más probable a la menos probable.	través de una búsqueda en internet.		
7. Las preguntas pueden tener varias respuestas	Alternativamente (o adicionalmente), la aplicación podría pedir al usuario responder a una o más "preguntas secretas", que son escogidas por el usuario entre un conjunto de preguntas posibles.	Dar varias respuestas a las preguntas		
8. Luego de responder las preguntas la aplicación muestra la contraseña antigua	Este sistema se basa en la asunción de que el email del usuario no ha sido comprometido y es lo suficientemente seguro para este cometido.	Mostrar la contraseña antigua		
9. Luego de responder las preguntas la aplicación obliga	Una vez se ha encontrado una respuesta correcta a la pregunta.	Cambiar la contraseña una vez que responda las		

al usuario a cambiar inmediatamente la contraseña		preguntas		

Caso de prueba para la Validación de Datos.

Esta plantilla tiene el siguiente formato

- **Descripción General**
  <Resumen del CU.>
- **Condiciones de Ejecución.**
  <Precondiciones del CU>.
- **Secciones a probar en el Caso de Uso.**
  <Para cada sección los escenarios van a ser flujo básico + los alternativos>.

Tabla #9: Descripción de la actividad: Secciones a probar en el Caso de Uso.

Nombre de la sección.	Escenarios de la sección	Descripción de la Funcionalidad.
<SC 1: Nombre de la sección.>	<EC 1.1: Nombre del Escenario.>	<Descripción de la Funcionalidad.>
	EC 1.n: Nombre del Escenario	<Descripción de la Funcionalidad.>

- **Descripción de Variable.**

Tabla #10: Descripción de la actividad: Descripción de Variable

No.	Nombre del campo.	Clasificación.	Valor Nulo.	Descripción.
[1]<Se enumera	<Nombre del	<La clasificación	<Se	<Una breve descripción

todos los campos ó variable, descrito en el Caso de Uso.>	campo de entrada>	es según el componente de diseño utilizado ejemplo: campo de texto, lista desplegable o campo de selección.>	especifica si el campo puede ser nulo o no> Para ello solo se pone Sí o No.	de los datos que deben introducirse (Reglas que tiene que cumplir el campo.).>
[2]				
[3]				

- **Matriz de Datos.**

SC 1<Nombre de la Sección. >

Escenario	**Variable 1 *(Nombre de la variable)***	**Variable 2*(Nombre de la variable)***	**Variable N*(Nombre de la variable)***	**Respuesta del Sistema**	**Resultado de la Prueba**	**Flujo Centra**
<Nombre del escenario.>	V	V	V	<Se escribe el resultado que se espera al realizar la prueba.>	<Se escribe el resultado que se obtiene al realizar la prueba. Ejemplo: Satisfactorio. No Satisfactorio.>	<Pasos a desarrollar para probar la Funcionalidad que se indicó. Ejemplo: Paso 1 Paso 2 >
	I	V	V			
	V	I	V			

<Nombre del escenario.>	NA	NA	NA			
	I	I	I			

Las celdas de la tabla contienen V, I, o N/A. V indica válido, I indica inválido, y N/A que no es necesario proporcionar un valor del dato en este caso, ya que es irrelevante.

- **Registro de defectos y dificultades detectados.**

Elemento	**No**	**No Conformidad**	**Aspecto Correspondiente**	**Etapa de Detección**	**Significativa**	**No Significativa**	**Recomendación**	**Estado NC**	**Respuesta del Equipo de Desarrollo**
Módulo e Portafolio	<1>	<Descripción de la No Conformidad>	<Descripción del Aspecto correspondiente>	<Etapa de detección del error>	<X>	<X>	<X>	<Se coloca el estado de la NC y la fecha, cada vez que se revise se deja el estado anterior y se coloca el nuevo con la fecha en que se revisó.> RA: Resuelta PD: Pendiente NP: No Procede	<Esta columna se comienza a llenar a partir de la 2da iteración, y es responsabilidad del equipo de desarrollo, quien especifica la conformidad con lo encontrado o no y en caso de no proceder

									la no conformidad explica por qué.>

2.2.4 Paso #4: Ejecución de la pruebas

Se ejecutan las según la planificación realizada por las categorías de pruebas definidas haciendo uso de las listas de chequeo y los casos de prueba diseñados en los epígrafes anteriores. La ejecución de las pruebas se va a realizar comenzando por la Recopilación de Información, luego de esta prueba se pueden realizar las de Gestión de Configuración, Comprobación de la Autenticación y Validación de Datos en paralelo o en el orden que el especialista de prueba decida. Esta ejecución de pruebas cae fundamentalmente sobre el equipo de prueba.

Este equipo de prueba a medida que va ejecutando las pruebas va registrando las no conformidades (NC) detectadas en la aplicación del Remane que es la que usa el GCF o en la aplicación del proyecto que se esté revisando.

El jefe del laboratorio del prueba al finalizar el día revisar las no conformidades subidas por el equipo de prueba, este las guarda en el documento Parte Diario, este envía el documento: Parte Diario al jefe del proyecto que se está revisando. Toda esta ejecución se hace en todas las iteraciones que se realizan y se termina cuando el número de NC identificadas haya disminuido a más del 80 % de los defectos encontrados desde el inicio.

Tabla #11: Roles y responsabilidad de la actividad Ejecución de las pruebas.

Roles que intervienen	Responsabilidad
Probadores	Ejecuta las pruebas.
Jefe del Laboratorio de Pruebas.	Guarda las no conformidades encontradas, en el documento Parte Diario.
Jefe de Proyecto. (Representante por el Equipo de Desarrollo).	Orienta

2.2.5 Paso #5: Cierre de las Pruebas.

Como se mencionó anteriormente el equipo de prueba ejecuta las pruebas y en paralelo van a ir registrando las no conformidades encontradas. Además, luego que se haya registrado todas las no

conformidades encontradas en la ejecución de las pruebas el jefe del laboratorio de prueba al frente de las pruebas las revisa y las guarda en el documento Parte Diario.

El jefe del laboratorio de prueba se reúne con el jefe del proyecto que se está revisando y le hace entrega de las no conformidades encontradas. El jefe del proyecto se reúne con el equipo de desarrollo, ven las no conformidades encontradas en la fase anterior y las resuelven. Después de todo esto se finalizan las revisiones.

Conclusiones del capítulo.

Durante el presente capítulo se desarrolló detenidamente el procedimiento para pruebas de seguridad en aplicaciones web, mostrando paso a paso la composición del mismo para llevar a cabo las pruebas, revelando las categorías y fases por las cuales está constituido el procedimiento, además del diseño de las listas de chequeos y casos de pruebas. Este procedimiento puede ser utilizado por los probadores del GCF y por los probadores de los grupos de calidad de las facultades.

CAPÍTULO III: “VALIDACIÓN DEL PROCEDIMIENTO”

Introducción:

En el presente capítulo se evaluará el procedimiento para Pruebas de Intrusión, aplicándolo en el Módulo e- Portafolio perteneciente a la Plataforma Educativa ZERA.
Para lograr una validación satisfactoria para el procedimiento realizado con el objetivo de comprobar que esta constituye una solución al problema planteado en la investigación es sin lugar a dudas una de las grandes satisfacciones para cualquier investigador.

3.1 Módulo e-Portafolio perteneciente a la Plataforma Educativa ZERA.

En el marco de los contratos surgidos a raíz del reconocimiento de la universidad como empresa productora de software, surge como proyecto, el desarrollo de una plataforma para la gestión del aprendizaje, la cual fue nombrada ZERA, siendo el primer cliente el grupo editor mexicano Alfaomega. Se establecía en el contrato, la adecuación de los hiperentornos de aprendizaje de la colección Futuro.

ZERA es una plataforma para la gestión del aprendizaje que como se había mencionado anteriormente, tiene sus orígenes en una concepción pedagógica denominada “Hiperentornos de aprendizaje“, propuesta y desarrollada por pedagogos y especialistas del Ministerio de Educación de Cuba (MINED), los hiperentornos no son más que una mezcla de diferentes tipos de software educativos sustentados en tecnología hipermedia, estos constituyeron un modelo de medio de enseñanza para el apoyo al proceso de enseñanza – aprendizaje concebidos además como fundamento teórico y conceptual del desarrollo de las colecciones cubanas de software educativo.

El módulo e-Portafolio perteneciente a esta plataforma tiene como objetivo principal, almacenar las evidencias que posee cada estudiante. Las evidencias constituyen archivos que se emiten como respuesta a una tarea orientada por el profesor, representan una prueba tangible de que una asignación ha sido resuelta. El portafolio del estudiante permite además recepcionar las evaluaciones, y permite que estas sean consultadas en un momento determinado. Está estructurado en 4 secciones fundamentales, Notificaciones, Mis Calificaciones, Evidencias compartidas y Archivo de evidencias. A través de este módulo cada estudiante puede compartir sus evidencias resueltas, y comentar aquellas que hayan sido compartidas para él. Todo esto permite que se establezca un ambiente colaborativo en el proceso de enseñanza aprendizaje, y que cada estudiante almacene y consulte sus resultados de forma sistemática.

3.2 Aplicación del Procedimiento sobre el Módulo e-Portafolio

Objetivo del procedimiento para Pruebas de Intrusión

El objetivo de este procedimiento tiene encontrar las No Conformidades (NC) que se observen durante la ejecución de los pasos de este procedimiento en la aplicación del módulo e-Portafolio.

Alcance del procedimiento

Se realizan las Pruebas de Intrusión al módulo e-Portafolio del proyecto Alfaomega. Con el fin de detectar vulnerabilidades en la aplicación con el uso de las herramientas automatizadas Nikto y W3af y como herramientas documentadas las listas de chequeo.

3.2.1 Pasos para aplicar el procedimiento

Para la aplicación del procedimiento se han seleccionado las funcionalidades más importantes para el correcto funcionamiento del programa.

Paso 1: Como primer paso se recogieron todos los documentos y elementos necesarios para la realización de las pruebas. Al verificar que en el proyecto Alfaomega contiene los documentos necesarios para ejecutar las Pruebas de Intrusión como son: manuales de usuarios, casos de uso, casos de pruebas, un manual de ayuda para navegar por el sistema, se determina que si se le puede efectuar las pruebas**.**

Paso 2: Luego de tener todos los datos necesarios se procede a identificar que tipos de categorías de las pruebas de Intrusión se van a utilizar para la aplicación del procedimiento al módulo .También se efectúa la realización del plan de pruebas, la estrategia y se elabora el cronograma de trabajo.

Antes de realizar el plan de de pruebas y estrategia es muy importante obtener una visión rápida de lo que hace el módulo e-Portafolio que se va a probar.

Plan de Pruebas de Intrusión para el Módulo e Portafolio perteneciente a la Plataforma Educativa ZERA

- **Roles y Responsabilidades**

Tabla #1 Roles y Responsabilidades que intervienen en estas revisiones.

ROL	Cantidad	Responsabilidad
Planificadora del Servicio de Pruebas de Software a nivel de Sistema del GCF	1	• Elaborar lista de chequeo y casos de prueba para realizar las Pruebas de Intrusión.

		• Elaborar el plan de pruebas. • Supervisar el trabajo de pruebas, recogiendo las no conformidades para elaborar el informe de no conformidades.
Probadores del GCF.	**1**	• Elaborar lista de chequeo y caso de prueba para realizar las Pruebas de Intrusión. • Elaborar el plan de pruebas • Elaborar la estrategia de prueba • Ejecutar las pruebas. • Registrar no conformidades
Representante por el Equipo de Desarrollo.	**2**	• Verificar que el escenario de prueba concuerda con el de la aplicación. • Instruir al equipo de prueba sobre el ambiente de la aplicación.

- **Recursos de Sistema**

Recursos	**Tipo**
Servidor de Base de Datos.	**Característica del software** • PostgreSQL 8.4.
Servidor de Aplicación.	**Característica del software** • Apache versión 2.1 • Sistema Operativo: Windows XP Servi Pack 2 o Linux. • Gestor de Base de Datos: PostgreSQL 8.4.

	• Antivirus: Kaspersky Workstation 6.0 • Máquina Virtual Java: Virtual Machine 1.3 o superior **Características del hardware** • 1 Procesador CoreDuo • 2GB de RAM • 120 GB de HDD mínimo, aunque puede ser configurable de acuerdo a la cantidad de contenidos que se vayan contratando. • Ancho de banda: 10/100 Mbps UTP 5 LAM.
PC Cliente para pruebas.	**Características del hardware** • Microprocesador: Intel(R) Core(TM)2 Duo CPU E4500 a 2.20 GHz. • Memoria RAM: 1 GB (PC Clientes). • Ancho de Banda y tipo de conexión de red: 10/100 Mbps UTP 5 LAM. • Disco Duro: 160 GB (PC Clientes). **Característica del software** • Sistema Operativo Instalado: Linux. • Antivirus: Kaspersky Workstation 6.0. • Máquina Virtual Java: Virtual Machine 1.3 o superior. • Software de prueba: Nikto v2.03/2.04 y W3a.

Luego de realizar el plan de prueba se realiza la estrategia de prueba.

Estrategia de prueba.

A continuación se describirá el flujo de trabajo que será implementado durante todo el período de ejecución de las pruebas.

Descripción del flujo de trabajo.

El flujo de trabajo se inicia cuando la especialista y la probadora comienzan a diseñar las listas de chequeo y los casos de pruebas para su posterior utilización. Luego si todas las condiciones están creadas se pasa a la 1ra iteración de pruebas, se realizan las pruebas. En la medida que detecten errores, molestias o incomodidades en el trabajo con el producto, etc., estas serán anotadas. Al concluir la iteración de prueba se realizará un documento parte diario. Después se le entrega ese documento al equipo de desarrollo para que este comience a ejecutar los arreglos acordados.

Descripción de la estrategia y categoría de las pruebas de intrusión

Se determina las categoría de pruebas de Intrusión que se van a utilizar para hacer las revisiones a la aplicación e-Portafolio, para confeccionar las listas de chequeo y los casos de prueba.

Las categorías son:

- **Recopilación de la Información** y entre esta se va a utilizar: Firma digital, Descubrimiento de aplicaciones y Análisis de Códigos de Errores.
- **Gestión de Configuración** y entre esta se va a utilizar: Prueba de SSL y TSL**.**
- **Comprobación de la Autenticación** y entre esta se va a utilizar: Fuerza Bruta.
- **Prueba de Validación de Datos** y entre esta se va a utilizar: Inyección SQL, Inyección XML, Inyección XSS, Inyección LDAP, Inyección ORM.

Luego se confecciona el cronograma de trabajo

No.	Tarea	Fecha	Participantes	Iteración	Observaciones
	Actividad a desarrollar.	Fecha de inicio y fecha de fin de la actividad.	Responsable		
1	Se escogen las categoría de Pruebas Intrusión	15 de mayo del 2011	Planificadora y probador	1	

	a utilizar en correspondencia a las características de la aplicación				
2	Instalar las herramientas W3af y Nikto a la computadora donde se va a realizar las revisiones.	16 de mayo del 2011	Probador	1	
3	Capacitación acerca el uso de las herramientas.	16 de mayo del 2011	Probador	1	
4	Se confecciona las listas de chequeo y los casos de prueba a utilizar.	17 de mayo del 2011	Planificadora del Servicio y Probador		
5	Se realiza Primero la categoría recopilación de información la subcategoría de firma digital.	18 de mayo del 2011	Probador	1	
6	La categoría recopilación de información la subcategoría Descubrimiento de aplicaciones	19 de mayo del 2011	Probador	1	
7	La categoría Prueba de Validación de Datos la	20 de mayo del 2011	Probador	1	

	subcategoría Inyección SQL, Inyección XSS				
8	La categoría Prueba de Validación de Datos la subcategoría Inyección XML, Inyección LDAP y Inyección ORM	21 de mayo del 2011	Probador	1	
9	La categoría Comprobación de la Autenticación la subcategoría Fuerza Bruta	22 de mayo del 2011	Probador	1	
10	La categoría Gestión de Configuración la subcategoría de Prueba de SSL y TSL	23 de mayo del 2011	Probador	1	

Paso 3: Después de haber realizado el plan de pruebas y estrategia se procede a la confección de las listas de chequeo que se van a usar en el procedimiento.

De acuerdo como se escogió en la selección de los tipos de categorías de las Pruebas de Intrusión se elaboran las listas de chequeo y los casos de prueba.

Lista de chequeo para la categoría de Recopilación de Información y Pruebas de Gestión de Configuración

Indicadores a Evaluar	Descripción	Resultado Esperado	Resultado Real	Herramienta

1. Se puede obtener la firma digital (tipo y versión) del servidor web.	Este es el primer paso del proceso de recopilación de información, saber la versión y tipo de servidor web en ejecución, permite a las personas realizando la prueba, determinar vulnerabilidades conocidas, y los exploits adecuados a emplear durante las pruebas.	Se pone el tipo y la versión del servidor web que está en el plan de prueba.		W3af
2. Se identifican más aplicaciones web instaladas en el servidor web.	Es el proceso destinado a identificar aplicaciones web instaladas en una infraestructura dada.	Ver las aplicaciones que están instaladas en el servidor.		W3af
3. Se identifican todos los puertos abiertos en el IP del servidor y los servicios asociados a esos puertos.	Es el proceso para encontrar que aplicaciones específicas se encuentran instaladas en un servidor a partir de todos los puertos abiertos en el IP del servidor y los servicios asociados.	Ver los puertos abiertos en el IP del servidor.		W3af
4. Existen cifrados (SSL/TSL) débiles	Estos son dos protocolos que proveen, con el apoyo de criptografía, de canales de transmisión de datos seguros, para la protección, confidencialidad y autenticación de la información transmitida.	Verificar la confidencialidad y autenticación de la información.		Nikto

Lista de chequeo para la Comprobación de la autenticación

Indicadores a Evaluar	Descripción	Resultado Esperado	Resultado Real	Herramienta
1. Puede obtenerse mediante el ataque de fuerza bruta el usuario y la contraseña de un usuario privilegiado en la aplicación.	Consiste en averiguar el usuario y contraseña válida de un individuo registrado en el sistema.	Obtener el usuario y la contraseña de un usuario privilegiado en la aplicación.		W3af
2. Puede saltarse el sistema de autenticación mediante la petición directa de páginas	La clave para explotar con éxito y saltarse un sistema de autenticación de contraseña es encontrar una pregunta o conjunto de preguntas que ofrezcan la posibilidad de encontrar las respuestas fácilmente.	Saltarse el sistema de autenticación		no

Para la categoría de validación de datos se va a confeccionar varios casos de pruebas en dependencia de la cantidad de casos de usos que contenga el módulo e-Portafolio perteneciente a la Plataforma Educativa ZERA.

Paso 4: Luego de obtenidos las listas de chequeo se procede a la validación a través de la herramienta automatizadas Nikto y W3af. El resultado obtenido de las pruebas de plasmaron en el Registro de No Conformidades.

Tabla de No Conformidades Detectadas

Elemento	No	No Conformidad	Aspecto Correspondiente	Etapa de Detección	Significativa	No Significativa	Recomendación	Estado NC	Respuesta del Equipo de Desarrollo
Módulo e Portafolio	< 1>	Se Detectó, que el servidor web es vulnerable a Cross Site Tracing.	Descripción del Aspecto correspondiente	Iteración 1				Pendiente	
	<2>	Se detectó que el servidor fue hackeado. Por una puerta trasera.		Iteración 1				Pendiente	

En el **Anexo 4** se muestra las imágenes de las no conformidades encontradas con el uso de las herramientas.

Paso 5: Después de obtener las no conformidades encontradas, se le entrega al representante por el Equipo de Desarrollo. Para la validación de esta propuesta al módulo e-Portafolio solo se realizó una sola iteración. Y así se termina la validación.

3.3 Resultados de la Evaluación del Procedimiento.

El procedimiento se evalúo de forma satisfactoria. Se detectaron un total de 2 no conformidades, una vinculada con la vulnerabilidad de Cross Site Trancig y la otra vinculada a fuerza bruta.

La vulnerabilidad de Cross Site Trancig es llamada también como Cross Site Scripting, conocido como XSS por sus siglas en ingles, es una vulnerabilidad que muchos desarrolladores dejan pasar por alto, quizás por falta de un planeamiento de análisis de riesgos en el proceso de diseño, desarrollo e implementación de sus aplicativos, o simplemente no lo vean como una falla que les va a presentar problemas en su aplicación, incluso por desconocimiento de esta vulnerabilidad.

El Cross-Site-Scripting es una vulnerabilidad que puede causar un daño tanto a una aplicación web como a usuarios que de manera inconsciente activen dicha secuencia de comandos. Dicho código malicioso se compone de cadenas de datos cuyo contenido son scripts completos formados por enlaces o ejecutados desde formularios.

La vulnerabilidad Fuerza Bruta identifica que el servidor fue hackeado por una puerta trasera. Una puerta trasera (o en inglés backdoor), en un sistema informático es una secuencia especial dentro del código de programación mediante la cual se puede evitar los sistemas de seguridad del algoritmo (autentificación) para acceder al sistema. Aunque estas puertas pueden ser utilizadas para fines maliciosos y espionaje no siempre son un error, pueden haber sido diseñadas con la intención de tener una entrada secreta.

Conclusiones del capítulo.

En el presente capítulo se realizó la validación del procedimiento. Se tomó como objeto de pruebas el módulo e-Portafolio perteneciente a la Plataforma Educativa ZERA. Se le aplicaron las categorías de Prueba de Intrusión. Se realizó el plan de pruebas y estrategia para las pruebas de Intrusión. En base a los resultados obtenidos este procedimiento se califica de satisfactorio ya que se encontraron no conformidades.

CONCLUSIONES GENERALES

Con el cumplimiento de esta investigación, se arribó a las siguientes conclusiones:

1. El estudio de las soluciones existentes, permitió identificar los principales problemas que existe en la Universidad de las Ciencias Informáticas, garante para la toma de decisiones en cuanto a planificación y desarrollo de los procedimientos a realizar en el GCF.
2. El estudio del arte sobre las herramientas y metodologías existentes en el objeto de estudio que se analiza, permitió definir las dos herramientas automatizadas para la realización de estas pruebas de Intrusión.
3. La validación realizada al procedimiento, permitió llegar a conclusiones acertadas sobre el cumplimiento del objetivo de la presente investigación, demostrándose en su factibilidad, en uno de los módulos que forman parte de uno de los proyectos investigativos que realiza el centro FORTES.

RECOMENDACIONES

Al finalizar el presente trabajo se recomienda:

- Aplicar la propuesta del procedimiento para realizar Pruebas de Intrusión a Aplicaciones Web en varios proyectos productivos de la facultad.
- Probar y evaluar nuevas herramientas para pruebas de seguridad, estableciendo comparaciones entre las mismas.
- Estudiar sobre los otros tipos de pruebas de seguridad para poder crear algún procedimiento que se pueda utilizar en un futuro.
- Se recomienda para el estudio de las pruebas de Intrusión hacer uso de las métricas para aplicarlas en los proyectos.

BIBLIOGRÁFIA

Referencia Bibliográfica

1. **Hugo Vazquez, Roberto.** Introduccion a la Calidad dek Software. [En línea] 10 de Enero de 2010. http://.frmgridtics.utn.edu.ar/docs/Introduccion%20a%20la%20Calidad%20de%20Software%20Vazquez.pdf.

2. **Pressman, Roger S.** Ingenería del Software. Un enfoque Práctico. [En línea] 20 de Noviembre de 2010. http://eva.uci.cu/mod/resource/view.php?id=21621&subdir=/Ingenieria_del_Software_un_ enfoque_practico..

3. **Hernandez, Miguel.** *Manuel de Diseño de Proceo .* 2010.

4. Introduccin a la Seguridad Informática. [En línea] 15 de Enero de 2011. http://eva.uci.cu/mod/resource/view.php?inpopup=true&id=38414].

5. **Raccitti, H M.** Seguridad en Aplicaciones Web. [En línea] 16 de Enero de 2011. http://www.slideshare.net/ernesto.jimenez/seguridad-en-aplicaciones-web]..

6. *IEEE 1990.*

7. **Pressman, Roger S.** *Ingenería del Software. Un enfoque Práctico, Vplumen 2.*

8. Gestión de Calidad y Pruebas de Software. *2005, PRUEBAS DE SOFTWARE.* [En línea] www.pruebasdesoftware.com/laspruebasdesoftware.htm.

9. **Pressman, Roger S.** *Ingenería de Software, Un enfoque Práctico, Parte 1, Capítiulo 17.*

10. —. *Ingenería de Software, Un enfoque práctico, Parte 1, Capítulo 18.*

11. **Teruel, Alejandro.** *El Plan de Pruebas .* 2001.

12. *GUIA DE PRUEBAS OWASP.* 2008 V3.0.

13. **Software, Departamento de Ingenería de.** *Perfil de Calidad, Conferencia Proceso de Pruebas .*

14. Etrategia Pruebas de Seguridad. [En línea] 12 de Febrero de 2011. www.estratecgias.com/page/pruebas_de_seguridad.

15. Intriducción a la Seguridad Informática, Actividad #! [En línea] 22 de Marzo de 2011. http://eva.uci.cu/course/view.php?id=690]. .

16. **Domínguez, G.** Aplicasiones web seguras. [En línea] 9 de diciembre de 2008. http://www.sg.com.mx/content/view/793.

17. **C, S C.** Riesgos y seguridad en los Sistemas de Informacion . [En línea] 2007. http://ciberconta.unizar.es/leccion/seguro/099.HTM.

18. **Blog, David.** Top 10 Vulnerabilidades de la aplicaciones web. [En línea] 13 de octubre de 2007. www.davidvalverde.com › Blog.

19. Nessus, Southwest Research Intitute. [En línea] http://www.nessus.swri.org/.

20. **Symantec, Harry Anderson.** Introduction to Nesus. [En línea] 27 de octubre de 2003. http://www.symantec.com/connect/articles/introduction-nessus.

21. CIRT.net, Suspincion Breeds Confidence, Nikto2. [En línea] 2 de Enero de 2010. http://cirt.net/nikto2.

22. Brutus. [En línea] 11 de diciembre de 2010. http://www.bujarra.com/Brutus.html.

23. **Navarro, Juanjo.** Verción Cero, Paros 3.2.3. [En línea] http://www.versioncero.com/noticia/276/paros-323.

24. Security Scannerres. [En línea] 19 de septiembre de 2008. http://www.taringa.net/posts/downloads/1559234/Shadow-Security-Scanner-Eamp_-Crack.html.

25. OWASP, The open web Application Security Project. [En línea] 9 de marzo de 2010. http://www.owasp.org/index.php/Proyecto_WebScarab_OWASP.

26. Web Security Dojo-Auditoria de aolicaciones web, Hacking Etico . [En línea] 9 de Dicciembre de 2010. http://hackingetico.wordpress.com/2010/12/09/web-security-dojo-auditoria-de-aplicaciones-web/ .

27. Hacking Etico y Frameworks Open Souerce. [En línea] www.cybsec.com/.../CYBSEC-HackingEticoyFrameworksOpensource.pdf – Argentina.

BIBLIOGRAFÍA CONSULTADA

1. Herramientas para realizar pruebas de seguridad. 28 de enero de 2000. [Disponible en: www.hoobie.net/brutus/brutus-download.html.

2. Las 100 herramientas de seguridad más populares., 21 de junio del 2006. [Disponible en: http://www.kriptopolis.org/las-100-herramientas-de-seguridad-mas-populares

3. Pruebas Controladas de Intrusión [Disponible en: www.kpmg.com.ar/pdf/servicios/.../Pruebas_controladas_de_intrusion.pdf]

4. Pruebas de Seguridad., 2001. [Disponible en: http://www.espaciolinux.com/foros-tema-t36483.html.

5. FERNÁNDEZ-SANGUINO, J. *Seguridad del software*. Disponible en: http://es.tldp.org/Informes/informe-seguridad-SL/informe-seguridad-SL.pdf

6. QUINTANA, D. J. N. *Calidad de software*, 1997. [Disponible en: http://www.monografias.com/trabajos59/calidad-software/calidad-software.shtml

7. RODRÍGUEZ, C. L. *Fases del Desarrollo de Software.*, 28 de Julio de 2003. [Disponible en: http://users.dsic.upv.es/asignaturas/facultad/lsi/ejemplorup/.\

8. SORIANO, A. *Tipos de pruebas*. Disponible en: http://carolina.terna.net/ingsw3/datos/Tipos_Prueba.pdf.

9. Version Cero, Paros 3.2.3, Por Juanjo Navarro, 25 julio 05, http://www.versioncero.com/noticia/276/paros-323

10. Taringal, Security Scanneres, 19 Sep 2008, http://www.taringa.net/posts/downloads/1559234/Shadow-Security-Scanner-Eamp_-Crack.html.

11. OWASP, The open web Application Security Project, 9 Mar 2010http://www.owasp.org/index.php/Proyecto_WebScarab_OWASP.

GLOSARIO DE TÉRMINOS

CGI: Una tecnología que se usa en los servidores web.

HTTP: Protocolo para la conexión segura.

IDS: Sistema de Detección de Intrusos.

LDAP: Protocolo ligero de acceso a directorio.

ORM: Herramienta para mapear objetos relacionales.

Open Source: Código abierto.

SQL: Es un lenguaje declarativo de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones sobre las mismas.

SSL: Protocolo de Capa de Conexión Segura, protocolos criptográficos.

SSI: Directivas evaluadas por el servidor web antes de servir la página al usuario.

Site: Sitio.

TLS: Seguridad de la Capa de Transporte, protocolos criptográficos.

XSS: Es un tipo de inseguridad informática o agujero de seguridad basado en la explotación de vulnerabilidades del sistema de validación de HTML incrustado.

XML: Lenguaje de Etiquetado Extensible muy simple.