Universidad de las Ciencias Informáticas

Facultad 4

Título: "Procedimiento para Pruebas de Penetración en Aplicaciones Web. "

Trabajo de Diploma para optar por el título de
Ingeniero Informático.

Autores: María Félix Lorenzo Álvarez
Lisney Gil Loro

Tutores: Ing. Delmys Pozo Zulueta
Ing. Mairelis Quintero Rio
Ing. Violena Hernández Aguilar.

Ciudad de La Habana, Junio del 2009.
"Año del 50 Aniversario del Triunfo de la Revolución"

"Seamos realistas y hagamos lo imposible."

Ernesto "Che" Guevara.

DECLARACIÓN DE AUTORÍA

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmamos la presente a los 15 días del mes de junio del año 2009.

_____ _____

María Felix Lorenzo Alvarez **Lisney Gil Loro**

_____ _____

Ing. Delmys Pozo Zulueta **Ing. Mairelis Quintero Ríos**

_____

Ing. Violena Hernández Aguilar

AGRADECIMIENTOS

Bueno esta parte es la más difícil de mi tesis, es la de recordar a tantas personitas especiales que han formado parte de estos cinco lindos e inolvidables años de carrera.

Primero que nada quiero agradecerle mucho a toda mi familia pero en especial a **mi mamá**, **mi papá y mi hermanita adorada** (mi tatica).

A mi abuelita Edith que sabe lo especial e importante que es para mí.

A **mi abuelo Chichín** y no puedo dejar de agradecerle a **mi prima Adys** que si no hubiese sido por la insistencia de ella no estuviera hoy aquí.

Ahora te toca a ti **mi Chini** que aunque no tienes mi sangre ya formas parte de mi familia. Realmente no tengo palabras para agradecerte lo especial y maravilloso que has sabido ser conmigo en tan poco tiempo, por brindarme tanto amor y comprensión. Gracias por ser paciente conmigo, ayudarme tanto en todo, para ti nunca existen obstáculos si se trata de mí .Por saber ser amigo y pareja y sobre todo por darme la oportunidad de conocer a una familia tan linda como la que tienes. Sabes que Te Amo y estaré ahí siempre para ti.

Bueno no puedo dejar de mencionar nueve nombres que significan mucho para mí. Hablar de ustedes realmente me cuesta un poco de trabajo, pues me vienen a la mente tantos recuerdos de mi primaria, secundaria, ese pre que no vuelve más, esas fiestas de Falcón y tantos recuerdos lindos que nunca olvidaré. Primero quiero mencionar a dos de mis grandes amigas de siempre, **Daine** y **Laury** que a pesar de las tres haber cogido caminos diferentes siempre hemos sabido mantener nuestra amistad a lo largo de estos cinco años que llevamos separadas, saben que las adoro de corazón. También agradecerle **a Eilen, Lismary la del 20 de mayo, Luvy, Katy, Masle, Yady** y **Elizeth.**

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.

A ti **Yu** por ser tan especial conmigo, siempre te voy a recordar y a querer mucho.

A ti **Yele** por ser tan buena e incondicional, por todas las gozaderas, las alegrías y las tristezas compartidas, siempre te voy a querer mucho.

AGRADECIMIENTOS

A **Mile, Magda, Eli** (mi mamá) y **Lisy** que han estado conmigo en las buenas y en las malas, saben las cuatro que las quiero muchísimo y nunca las olvidaré.

A ti **Nimia** que aunque ya no estás nunca olvidaré los momentos lindos que pasamos juntas.

A ti **Alia** por los buenos momentos compartidos en el 53 y por ser tan especial.

A ti **Mary** por ser una muy buena compañera de tesis, por todos los días enteros de trabajo, los desvelos y preocupaciones vividas, muchas gracias por estar a mi lado.

No puedo dejar de mencionarte a ti **Yaniel** por ser tan especial conmigo, sobrellevarme y enseñarme tanto.

A ti **Eslavy** que de más está decirte que formas parte de mi historia aquí en la UCI, por ser tan lindo conmigo, por demostrarme ser un hombre de verdad y sobre todo ser mi amigo.

No podía dejar de mencionarte a ti **Osvaldo**, que aunque me has demostrado no tener memoria para nada, eres un buen amigo para mí.

También agradecerle mucho a todas las muchachas y muchachos del **4102** con los cuales pasé casi toda mi carrera y ahora los actuales del **4502**.A todas las amistades que he conocido en la escuela, a todos esos profesores que me ayudaron a llegar hasta aquí. Pero sobre todo a ti **Liuver** que fuiste uno de mis mejores profesores en la UCI y a pesar de no estar aquí, sigues siendo un buen amigo.

Y muy especialmente a **mi tutora Delmys** por apoyarme tanto, soportar mi quisquilla con las cosas de la tesis. Por ser tutora y amiga y sobre todas las cosas por ser la tutora que yo siempre había soñado tener. Una persona que se llevara bien conmigo, con la cual tuviera confianza y me guiara bien en el desarrollo de mi tesis. Sabes que conmigo puedes contar para lo que sea. Nunca te voy a olvidar.

Y agradecerte también a ti **Mairelis** que a pesar de tu embarazo en la recta final nos fuiste de gran ayuda. Te deseo lo mejor con tu bebita. También agradecerle a **Violena** por su apoyo.

Lisney

Este trabajo está dedicado principalmente a **mi mamá**, que es la luz que ilumina todos mis días y la razón por la cual hoy soy lo que soy y puedo tener esta oportunidad. Porque mimi tú me enseñaste a ser mejor persona cada día. Quiero que me disculpes si en algún momento te he hecho sentir mal con mis palabras. Tú sabes que tu lugar nadie lo va a ocupar, esté con quien esté y que siempre vas a ser para mí la persona más linda y buena del mundo. Por todo eso eres alguien especial.

A **mi papá**, tu sabes papi que te quiero mucho y te agradezco todo lo que hiciste por mí en este año. Por haberte portado tan bien conmigo y por aguantar todo lo que te digo algunas veces, tú sabes que te quiero.

A **mis hermanos** a todos los quiero bien grande. **Nao** y **Rudy** ustedes saben que pueden contar conmigo, que a pesar que no vivamos juntos porque ya los dos lograron hacer su familia. Aún recuerdo cuando me decían que me iban a robar por la noche o cuando se ponían a secretear delante de mí para molestarme jaja. Nunca voy a olvidar esos recuerdos y no se preocupen que todo en la vida llega. Ustedes se merecen lo mejor y yo siempre los voy a querer porque son mis hermanitos más grandes.

A **mi hermanito Marlito** que tanto me ha costado estar lejos de él, solo yo lo sé. Mi (ñú) aunque no te guste que te lo diga no sabes lo difícil que se me hacía venir para la escuela y saber que me demoraría días para volverte a ver. Te quiero mucho.

También a **mis otros hermanitos Dayi, Dayansito** y **Marquito** ustedes sabe que para mí son una parte importante quiero que siempre recuerden que su hermanita los quiere un montón.

A mi padrastro **Naldito** a pesar de las cosas que sucedieron contigo y con mimi yo siempre diré que eres mi padrastro. Porque tú jugaste un papel muy importante en toda mi vida. Siempre te voy a estar muy agradecida, también te quiero mucho.

A **mi tati**, tú sabes **Abe** que para mí eres alguien especial. Cada segundo aquí en la escuela lo he logrado soportar porque estás a mi lado. Nunca voy a olvidar todo el apoyo incondicional que me brindaste en los momentos más difíciles. Quisiera que nunca olvides que te quiero mucho y que a pesar que soy bien malcriada tengas paciencia conmigo como siempre lo has hecho, tati gracias.

A **mis abuelos Mami** y **Papi** y fundamentalmente a **mi abuelita Raque,** que de verme hoy aquí estaría muy orgullosa y feliz. Pero tristemente no está hoy conmigo, pero yo

sé que desde algún lugar ella me está viendo. A veces te extraño mucho, quisiera poder verte y al menos darte un abrazo.

A **mis tíos** también están dedicados todos mis esfuerzos, los quiero mucho.

A todos **mis primos** principalmente a mi primo **Albertico**, yo sé que tu vas a triunfar en la vida. Eres la persona más buena que alguien puede tener como primo, tú sabes que para mí eres más que eso, te quiero.

A toda la **familia de mi novio** que siempre ha estado al tanto de mis cosas, a ustedes también los quiero bastante.

A todos **mis amigos** que han estado a mi lado siempre, a **Roxy**, mi flaqui tú sabes que te quiero grande y no te preocupes que todo un día llega. **Arleen, Yordy, Yany, Queto, Yusni**, todos para mí son muy importantes.

A todos los **amigos** que he logrado hacer en la Universidad, a **Liset Hidalgo, Randy, Yoe, Liset Morejón, a la Puchy, a Carlito, Magdanis, Yelenis** y todas las **muchachitas del apartamento.** Sobre todo a mi compañera de tesis, **Lisny** en este tiempo que hemos estado juntas compartiendo los días enteros realizando la tesis me han servido para darme cuanta que eres una verdadera amiga. Desearía que no tuviéramos que separarnos, de verdad creo que hacemos un buen team jajaj.

A ti **Delmys** por haber sido no solo mi tutora sino también una amiga que siempre estuvo a nuestro lado tratando de que fuéramos cada día mejor, por ser tan paciente. Nunca olvidaré como te portaste con nosotras, te quiero mucho.

A ti **Mairelis** por haber sido tan dedicada y preocuparte por nosotras. Te deseo que todo esté bien con tu embarazo y que sea una niña bien saludable, te quiero.

Mary

DEDICATORIA

Le dedico todo mi esfuerzo durante estos cinco años de universidad a mi mamá, mi papá, mi abuelita Edith y mi abuelito Chichín que han sido las personas que me han enseñado todo en la vida, los que siempre me han apoyado y me han guiado por el buen camino.

Y muy especialmente a mi abuelito Mongo que a pesar de no estar ya junto a mí siempre supo darme mucho amor y comprensión y eso nunca lo voy a olvidar.

Y a ti Lismy que eres una de las cosas más grandes que tengo en mi vida y has sido mi inspiración de seguir adelante siempre, para darte un buen ejemplo.

A todas mis buenas amigas y amigos que han estado siempre a mi lado apoyándome y a mi novio que ha sabido ser todo para mí en estos meses de tanta tención.

Lisney

DEDICATORIA

Le dedico esta tesis a la Revolución por haberme dado la oportunidad de estar hoy en la Universidad y sobre todo a Fidel, por hacer mi sueño realidad.

A mi mimiti, de no ser por ella hoy no estuviera aquí y no sería la persona que soy, ella es mi inspiración.

A mis hermanitos, a ellos les pido que sigan luchando por sus sueños que tarde o temprano se realizarán.

A mi novio por ser la persona más comprensiva del mundo y por quererme tanto.

A mi papá por demostrarme que sí me quiere.

A todos mis amigos siempre los voy a recordar aunque no nos podamos ver más.

Y sobre todo esta tesis se la dedico a mi abuelita que no está conmigo, pero de alguna forma la siento a mi lado.

Mary

RESUMEN

Las pruebas de penetración son una técnica de pruebas de seguridad empleadas para explotar vulnerabilidades cuando el software está completamente elaborado. Actualmente muchas personas emplean este tipo de prueba para aplicaciones web como su técnica de comprobación de seguridad principal. Se han desarrollado herramientas de prueba de penetración para automatizar el proceso.

El presente trabajo de diploma hace un estudio de las pruebas de penetración que se le realizan a las aplicaciones web, debido a la necesidad que tiene actualmente la Universidad de las Ciencias Informáticas (UCI) en sus proyectos productivos, en los cuales no se hacen pruebas de seguridad sino pruebas de funcionalidad, volumen, cargas y stress, realizando además una investigación de las herramientas que se utilizan para efectuar estas pruebas, para posteriormente seleccionar las herramientas más óptimas y eficaces. Se abordarán temas relacionados con las pruebas de seguridad a las aplicaciones web, su impacto en la actualidad, tipos de técnicas utilizadas. Se realizará una propuesta de un procedimiento para pruebas de penetración en aplicaciones web, el cual aportará grandes beneficios a los proyectos productivos de la Universidad, este estará unido a la confección de un manual de usuario de las herramientas que seleccionamos.

Palabras claves: Pruebas de Seguridad, Prueba de Penetración, Aplicaciones web, Calidad.

TABLA DE CONTENIDOS

INTRODUCCIÓN

El desarrollo de software, como se practica actualmente, es lento, costoso y sujeto a errores, usualmente obteniendo productos con un gran número de defectos, causando serios problemas de usabilidad, disponibilidad, rendimiento, seguridad y otras calidades de servicio (1). El mismo es un trabajo intensivo, donde sus productos obviamente ofrecen un valor significante a los consumidores. Esto no significa que los consumidores queden totalmente satisfechos con el software que les es entregado, o con la manera en que se le entregue. Sólo significa que ellos valoran el software, que está dispuestos a soportar grandes riesgos y pérdidas en orden de intentar alcanzar los beneficios que el mismo produce (2).

Entre los beneficios más importantes durante la última década se encuentran los lenguajes de código byte[1], patrones y métodos ágiles. Aparte de estos avances, se sigue desarrollando software de la misma manera que se hacía diez años atrás. Los métodos y prácticas de control de la calidad del software no han cambiado mucho, y tampoco lo han hecho los costos y riesgos asociados (3).

Para controlar la calidad del software es necesario, ante todo, definir los parámetros, indicadores o criterios de medición (4). Las cualidades para medir la calidad del software son definidas por innumerables autores, los cuales las denominan y agrupan de formas diferentes. Por ejemplo, John Wiley (5) define métricas[2] de calidad y criterios, donde cada métrica se obtiene a partir de combinaciones de los diferentes criterios. Otros autores identifican la calidad con el nivel de complejidad del software y definen dos categorías de métricas: de complejidad de programa o código, y de complejidad de sistema o estructura. Todos los autores coinciden en que el software posee determinados índices medibles que son las bases para la calidad, el control y el perfeccionamiento de la productividad. Lograr el éxito en la producción de software es hacerlo cada día mejor.

[1] **Código de Byte**: El compilador de Java compila su código fuente en un código de byte. El término de código de byte viene del hecho que cada parte de su programa en Java se reduce a una secuencia de bytes que representan instrucciones en una máquina virtual.

[2] **Métrica:** Es una medida cuantitativa del grado en el cual un sistema, componente o proceso posee un atributo dado.

Una idea general sobre un software de calidad es aquel que debiera cumplir con los requerimientos explícitos e implícitos además de ser confiable y aceptable. Veamos cada uno de las principales características que hacen a un software de calidad (6):

Mantenibilidad: El software debe ser diseñado de tal manera, que permita ajustarlo a los cambios en los requerimientos del cliente. Esta característica es crucial, debido al inevitable cambio del contexto en el que se desempeña un software.

Confiabilidad: Incluye varias características además de la confiabilidad, como la seguridad, control de fallos, etc.

Eficiencia: Tiene que ver con el uso eficiente de los recursos que necesita un sistema para su funcionamiento.

Usabilidad: El software debiera ser utilizado sin un gran esfuerzo por los usuarios para los que fue diseñado, documentado.

Corrección: El grado en que una aplicación satisface sus especificaciones y consigue los objetivos encomendados por el cliente.

Fiabilidad: El grado que se puede esperar de una aplicación lleve a cabo las operaciones especificadas y con la precisión requerida. Se refiere a la capacidad del sistema de funcionar permanentemente sin fallos y de mantener la integridad de los datos.

Portabilidad: La capacidad del producto de software para ser transferido de un entorno a otro. El entorno puede incluir la organización, hardware o software.

Integridad: El grado con que puede controlarse el acceso al software o a los datos a personal no autorizado y se mide la capacidad del sistema a resistir ataques contra su seguridad.

Facilidad de uso: El esfuerzo requerido para aprender el manejo de una aplicación, trabajar con ella, introducir datos y conseguir resultados.

Seguridad: Consiste en asegurar que los recursos del sistema de información (material informático o programas) de una organización sean utilizados de la manera que se decidió y que el acceso a la información allí contenida, así como su modificación sólo sea posible a las personas que se encuentren acreditadas y dentro de los límites de su autorización.

Como puede observarse, las diversas características con las que se desea que cumpla un software de calidad varían ampliamente. Algunas tienen que ver con el usuario que interactúa con el sistema, otras con el líder del proyecto y diseñadores, otras características parecen muy abstractas y hasta indefinidas.

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 (7).

La obtención de un software con calidad implica la utilización de metodologías[3] 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.

En la actualidad, 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 (8). 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 mas eficaces para dar seguridad a las aplicaciones.

La seguridad, como parte necesaria del desarrollo de software, es un tema que hasta hace muy 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.

Es cierto que aquellas compañías que se dedican a desarrollar sistemas con altos índices de seguridad han tenido mayores éxitos, han alcanzado la confianza de los clientes y se han establecido como líderes en la industria de la informática. Debido a esto ha surgido un mayor interés por parte de estas compañías en desarrollar productos y servicios informáticos que manejen y mantengan un mejor control de la calidad.

[3] **Metodología:** Métodos de investigación que se siguen para alcanzar una gama de objetivos en una ciencia.

En la Universidad de las Ciencias Informáticas (UCI) la información es un recurso con valor incalculable, teniendo en cuenta que el centro tiene una amplia conexión interna y acceso pleno a internet por la mayoría del personal. Además se encuentra en estos momentos inmersa en el desarrollo de una gama de sistemas de software muy importantes a partir de convenios firmados con diferentes empresas, tanto del ámbito nacional como internacional. Muchos de estos productos que se implementan, manejarán un volumen considerable de información de clientes. El objetivo de estos sistemas de desarrollo de software es incrementar la calidad del software a través de una 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 sensible almacenada en formato electrónico. Por esta razón a partir del año 2005 es creado el Laboratorio Industrial de Pruebas de Software (LIPS) en la UCI con el fin de certificar con un sello de calidad los productos que estén listos para ser comercializados internacionalmente. También se mide la calidad de los productos que se están produciendo en la universidad. Además se crearon en algunas facultades grupos de calidad con el objetivo de garantizar que exista documentación, lograr mejoras en el proceso de desarrollo de software y garantizar que se prueben las aplicaciones que realizan en cada facultad.

Sin embargo, la **situación problémica** que se presenta actualmente en el Laboratorio Industrial de Pruebas de Software (LIPS) es que las pruebas que se realizan al software son: pruebas funcionales, de volumen, de carga, de stress y regresión pero no se efectúan pruebas de penetración, que es una técnica de prueba de seguridad, donde se buscan incidencias de seguridad en el software después de elaborado. Además existe este mismo problema en los proyectos productivos de la universidad, los cuales solo realizan las pruebas de funcionalidad.

A partir de esta situación problémica surge el siguiente **problema:** ¿Cómo desarrollar pruebas de penetración en aplicaciones web?

Por lo cual, el **objeto de estudio** lo constituyen las pruebas de penetración a aplicaciones Web y el **campo de acción** son los procesos de pruebas de penetración en el Laboratorio Industrial de Pruebas de Software (LIPS).

Se persigue con esta investigación lograr el siguiente **objetivo general:** Proponer un procedimiento para realizar pruebas de penetración en aplicaciones web.

Para lo que se proponen las siguientes tareas a realizar**:**

- ✓ Realizar un estudio previo del estado del arte sobre los principales temas relacionados con la seguridad en el desarrollo de aplicaciones web y los problemas que afectan el buen desarrollo de esta.
- ✓ Estudiar las pruebas de seguridad que se le realizan al software.
- ✓ Estudiar las pruebas de penetración que se le realizan al software.
- ✓ Proponer un procedimiento para las pruebas de penetración en las aplicaciones web para el Laboratorio Industrial de Pruebas de Software (LIPS).
- ✓ Proponer herramientas para la automatización de las pruebas de seguridad.
- ✓ Realizar los manuales de usuarios correspondientes a las herramientas seleccionadas.
- ✓ Validar el procedimiento.

El presente documento está estructurado en tres capítulos.

En el **Capítulo 1** “Fundamentación teórica”, se presenta un estudio acerca de los conceptos principales de las pruebas de seguridad en aplicaciones web, los principios, técnicas de pruebas de seguridad, los tipos de pruebas y un resumen de las herramientas mediante tablas comparativas.

En el **Capítulo 2** “Propuesta de un procedimiento para pruebas de penetración en aplicaciones web”, se explica detalladamente el procedimiento para pruebas de penetración.

En el último capítulo de este documento, el **Capítulo 3** “Validación del procedimiento”, se valida el procedimiento utilizando el método de Delphi.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA.

Introducción:

En éste capítulo se exponen los temas fundamentales que sustentan la investigación. Se mencionan conceptos de seguridad, pruebas seguridad. Se aborda de manera general el proceso de pruebas de software, enfatizando las pruebas de seguridad en aplicaciones web, específicamente el tema de las pruebas de penetración a aplicaciones web como tema inicial para este trabajo. Se muestra un estudio de las herramientas que existen para la ejecución de pruebas de seguridad, así como comparaciones entre ellas.

1.1Seguridad de Software.

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 aplica los principios de la seguridad de información al desarrollo de software. 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. La misma requiere más manejo y riesgo de mitigación, de la que requiere la tecnología. Una vez que se identifiquen los riesgos, identificar medidas de seguridad apropiadas llega a ser manejable.

Además de esto la seguridad del código y el proceso de software deben de ser considerados durante la fase del diseño y desarrollo. Además, la misma debe de ser preservada durante la operación y el mantenimiento para asegurar la integridad de una parte del software. Esta se origina en dos problemas fundamentales: Los sistemas que son teóricamente seguros pueden ser inseguros en la práctica. Además los sistemas son cada vez más complejos, la complejidad proporciona más oportunidades para los ataques. Es mucho más fácil probar que un sistema es inseguro que demostrar que uno es seguro.

Para mantener la seguridad es necesario garantizar la confidencialidad, integridad y disponibilidad.

Confidencialidad: La confidencialidad es la propiedad de prevenir la divulgación de información a personas o sistemas no autorizados.

Existen infinidades de posibles ataques contra la privacidad, especialmente en la comunicación de los datos. La transmisión a través de un medio presenta múltiples oportunidades para ser interceptada y copiada. Algunos de los ejemplos más conocidos están relacionados con las líneas intervenidas, la intercepción o recepción electromagnética no autorizada o la simple intrusión directa en los equipos donde la información está físicamente almacenada.

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 mitigar[4] 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.

[4] **Mitigar**: Reducción de la vulnerabilidad, es decir la atenuación de los daños potenciales sobre la vida y los bienes causados por un evento de carácter.

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.

1.1.1 Fallos de Seguridad.

Se puede hacer un análisis agrupando los fallos de seguridad que se pueden dar en el software. De una forma simplista, se pueden dividir en tres bloques:

• Fallos debidos a errores desconocidos en el software, o conocidos sólo por terceras entidades hostiles.

• Fallos debidos a errores conocidos pero no arreglados en la copia en uso del software.

• Fallos debidos a una mala configuración del software, que introduce vulnerabilidades en el sistema.

El primero de ellos se puede atribuir a la calidad del código, el segundo a la capacidad y celeridad de arreglo de los errores descubiertos en el código por parte del proveedor del mismo y a la capacidad del administrador de recibir e instalar nuevas copias de este software actualizado. El tercer tipo de vulnerabilidades puede atribuirse, a una falta de documentación del software o una falta de formación adecuada de los administradores para hacer una adaptación correcta del mismo a sus necesidades.

1.1.2 Clasificación de Amenazas en Aplicaciones Web.

En las aplicaciones web, existen importantes amenazas, que si no se tienen en cuenta podría ocasionar grandes daños en las mismas, estas amenazas se dividen en tres grupos.

Atendiendo al origen:

- ✓ Externas. Se originan desde fuera de la organización.
- ✓ Internas. Se producen desde dentro de la propia organización y suelen tener un mayor impacto sobre la misma.

Atendiendo al área de efecto:

- ✓ Físicas. Afectan a los elementos físicos de las organizaciones.
- ✓ Lógicas**.** Su área de efecto engloba a todos los activos de naturaleza digital.
- ✓ Humanas. Se centran en vulnerar la seguridad utilizando como medio a las personas.

Atendiendo al efecto que provoca:

- ✓ Interrupción. Esta amenaza afecta la continuidad de los servicios, pudiendo hacer que estos queden inutilizados o no disponibles.
- ✓ Interceptación. El propósito de esta clase se basa en el acceso no autorizado de un elemento a un determinado objeto.
- ✓ Modificación. Implica que un elemento no autorizado, además de acceder al objeto ha conseguido modificarlo.
- ✓ Fabricación. El objetivo de esta eventualidad es el de crear un objeto similar al atacado de tal manera que pueda suplantar al original.

1.1.3 Clasificación de Vulnerabilidades en aplicaciones Web.

Conocer las vulnerabilidades que hacen insegura una aplicación es un tema de gran relevancia, ya que esta es una vía para que el atacante acceda al sistema y haga lo que quiera según sus intereses. Estas vulnerabilidades se encuentran desglosadas en dos grupos, los que se muestran a continuación.

Errores de Diseño

Este tipo de errores se producen cuando el diseño del flujo operacional de la propia aplicación es inseguro. Los motivos que originan este tipo de errores suelen ser la ignorancia, negligencia o simplemente el desconocimiento de algunos conceptos mínimos de seguridad.

Autenticación.

- ✓ Autenticación insuficiente. Se presenta cuando una aplicación web permite a un atacante el acceso contenido privilegiado o funcionalidades sin haberse autenticado.

- ✓ Validación débil en la recuperación de contraseñas. Se puede utilizar este ataque cuando la aplicación permite a un atacante obtener o modificar la contraseña de otro usuario.

Autorización.

- ✓ Predicción de credenciales. Este método permite a un atacante suplantar la identidad de un usuario legítimo. A todos los efectos, interactúa con la aplicación como si fuera el usuario comprometido.

- ✓ Expiración[5] de sesión insuficiente. Se produce cuando el tiempo de expiración de las credenciales es tal que permite a un atacante reutilizar credenciales suyas o de otros usuarios para autenticarse.

- ✓ Fijación de sesión. Si una aplicación web asigna siempre las mismas credenciales o ID a un usuario, estas pueden ser aprovechadas por un atacante para suplantarlo siempre que este las conozca y el usuario comprometido se encuentre autenticado dentro de la aplicación.

Ataques lógicos.

- ✓ Abuso de funcionalidad .Mediante este error un atacante puede aprovechar las propias capacidades y funcionalidades de una aplicación web para evadir los mecanismos de autenticación.

- ✓ Validación de procesos insuficientes. Este ataque se presenta cuando una aplicación permite a un atacante evadir o engañar el flujo de control de la misma.

Errores de Programación y Configuración.

El desarrollo de una aplicación no es más que la codificación del diseño realizado. El problema radica en que algunas veces muchas decisiones de diseño se toman en el mismo momento de la implementación.

La celeridad genera que dichas decisiones no estén lo suficientemente estudiadas.

Autenticación.

[5] **Expiración:** Es cuando algún objeto cosa o ser viviente pierde valor, o de alguna forma fallece.

- ✓ Fuerza Bruta. Este ataque consiste en obtener información probando todas las combinaciones posibles mediante un proceso automatizado.

Ataques en la parte cliente.

- ✓ Suplantación de contenido. Mediante esta técnica un atacante consigue hacerle creer al usuario objetivo que cierto contenido de una aplicación web es legítimo, cuando no lo es.

- ✓ Cross-Site-Scripting (XSS). Este método fuerza a la aplicación web a repetir código que ha insertado el atacante, de tal manera que el navegador del cliente lo interprete y ejecute.

Ejecución de comandos.

- ✓ Desbordamientos de buffer. Este tipo de técnica permite alterar el flujo de un programa a través de la modificación de contenidos de memoria.

- ✓ Inyección SQL. Este ataque permite alterar los contenidos de las sentencias SQL que se generan a través de entradas de la aplicación web.

- ✓ Inyección LDAP. Este ataque permite alterar los contenidos de las sentencias LDAP que se generan a través de entradas de la aplicación web.

Obtención de información.

- ✓ Indexación[6] de directorios. Esta función permite que el servidor liste los ficheros de un directorio determinado.

- ✓ Pathtrasversal. Esta técnica permite el acceso a ficheros, directorio o incluso ejecutables que residen fuera del directorio raíz del servidor.

- ✓ Recursos predecibles. Mediante este proceso un atacante intenta acceder a recursos y funcionalidades que suelen estar ocultas pero accesibles en un servidor web cuando este no está configurado correctamente.

1.1.4 Las 10 vulnerabilidades más comunes de una aplicación Web.

[6] **Indexación:** En informática, tiene como propósito la elaboración de un índice que contenga de forma ordenada la información, esto con la finalidad de obtener resultados de forma sustancialmente más rápida y relevante al momento de realizar una búsqueda.

Existen 10 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…).

- ✓ 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[7] 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

[7] **Tokens**: También llamado componente léxico es una cadena de caracteres que tiene un significado coherente en cierto lenguaje de programación.

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.

- ✓ Staks de los componentes.
- ✓ No ambiente Java.

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. Llamadas a:

- ✓ System.
- ✓ Exec.
- ✓ Fork.
- ✓ Runtime.exec.
- ✓ Solicitudes SQL.

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[8] para proteger información y credenciales. Estas funciones y

[8] **Criptografía**: Arte o ciencia de cifrar y descifrar información mediante técnicas especiales y es empleada frecuentemente para permitir un intercambio de mensajes que sólo puedan ser leídos por personas a las que van dirigidos y que poseen los medios para descifrarlos.

el código que integran a ellas han sido difíciles de codificar adecuadamente, lo cual frecuentemente redunda en una protección débil.

- ✓ Fallar al encriptar información crucial.
- ✓ Almacenamiento inseguro de llaves, certificados y contraseñas.
- ✓ Almacenamiento incorrecto de secretos en memoria.
- ✓ Fuentes pobres de aleatoriedad[9].
- ✓ Elección pobre de algoritmo.
- ✓ Intentar inventar el nuevo algoritmo de encriptación.
- ✓ Fallar al incluir soporte para cambios en las llaves de encriptación.

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.

- ✓ Fallas de seguridad no parchadas en el software del servidor.
- ✓ Fallas de seguridad en el software del servidor o malas configuraciones que permiten ataques de listado de directorio o cross directory.
- ✓ Innecesarios archivos por defecto, de respaldo o de ejemplo, incluyendo scripts, aplicaciones, archivos de configuración y páginas web.
- ✓ Permisos no adecuados en archivos y directorios.

[9] **Aleatoriedad**: Procesos aleatorios quedan englobados dentro del área del cálculo de probabilidad y, en un marco más amplio en el de la estadística.

- ✓ Servicios innecesarios habilitados, incluyendo manejo de contenido y administración remota.
- ✓ Cuentas por defecto con contraseñas por defecto.
- ✓ Funciones administrativas o de depuración[10] que son habilitadas o accesibles.
- ✓ Certificados SSL y opciones de encriptación mal configurados.
- ✓ Uso de certificados por defecto.
- ✓ Autenticación inadecuada con sistemas externos.

1.1.5 Situación Actual de la Seguridad de Software.

El software de seguridad se ha vuelto cada vez más necesario con el pasar de los años tanto en empresas como en lugares públicos y todo tipo de sitios en los cuales puede llegar a ocurrir una violación a la seguridad del mismo. Ahora bien, se debe tener en cuenta que existen diferentes tipos de software de seguridad, pero se debe señalar que no siempre se emplean los mismos programas considerando que no es igual instalar un paquete de software de seguridad para una empresa que para cualquier otra institución. Por eso a la hora de decidir cual puede llegar a ser más conveniente, se debe pedir consejo a algún experto en el tema ya que muchas veces sucede que los software de seguridad que se compran no son lo suficientemente eficientes como se desearía, por eso es importante no arriesgarse si no se conoce del tema.

Por otro lado también es muy importante determinar cual será la función del software de seguridad, es decir, qué tipo de seguridad se quiere que el mismo brinde. Para entender a lo que se refiere, es importante tener en cuenta que, por ejemplo, muchas empresas suelen contratar paquetes especializados de software para reforzar la seguridad en sus sistemas, ya sea por precaución hacia el espionaje competitivo o debido a otra amenaza.

Como bien se dijo anteriormente, existen muchos tipos de paquetes de software de seguridad que se aplican a diferentes situaciones y contextos. Los cuales en algunos casos suelen no ser tan complejos como parecen, de todos modos, cuando se trata de

[10] **Depuración:** El proceso de identificar y corregir errores de programación.

operar con algún tipo de software de seguridad siempre es mejor que lo haga un especialista para así ahorrarse el riesgo que se correría de cometer algún error y producir alguna falla en el sistema entero de la empresa. Es importante destacar que si bien se mencionaron las dos razones principales por las cuales las empresas instalan software de seguridad, se debe decir que la mayoría de las veces lo hacen para cuidarse del espionaje competitivo. Es muy común que mediante algún documento infiltrado, una empresa logre ingresar dentro del sistema de otra, y así poder tener acceso no solo a los datos que maneja sino también, a todos los planes y estrategias que puede llegar a tener la misma. Estas ocasiones se dan especialmente cuando se trata de empresas que se mueven en mercados muy acaudalados.

Cuba es un país donde el desarrollo del software es aun incipiente. Es por ello que una de las principales tareas del Gobierno Cubano es desarrollar la Industria del Software. Lo que ha traído consigo la creación de variadas estrategias con el fin de elevar la producción y la calidad del software cubano, logrando así una seguridad en el mismo. En el año 2001, existían en el Ministerio de la Informática y las Comunicaciones (MIC) 24 empresas o entidades que tenían en su objeto social la producción del software y servicios informáticos, esta estructura sufrió diversas reorganizaciones. Actualmente hay 8 entidades en el MIC que desarrollan esta actividad.

La Industria Cubana del Software (ICSW) está llamada a convertirse en una significativa fuente de ingresos para el país, como resultado del correcto aprovechamiento de las ventajas del alto capital humano disponible. La promoción de la industria cubana del software en el ámbito internacional ha tenido como línea estratégica aprovechar la enorme credibilidad que tiene Cuba en sectores tales como la salud, la educación y el deporte. El continuar la producción sostenida de software de alta calidad en prestaciones, imagen y soporte, para satisfacer las necesidades nacionales en estos sectores, tendrá una positiva repercusión en el incremento de la exportación.

Se ha desarrollado software para todos los aspectos de la economía cubana, incluyendo las telecomunicaciones, educación. Para fomentar el desarrollo de la industria se prioriza en el país la enseñanza masiva de la computación en todo el sistema educacional, se cuenta con 26 politécnicos de informática, parte de los cuales son nuevos o fueron completamente remozados[11], donde se preparan alrededor de 40

[11] **Remozados:** Dar un aspecto nuevo o moderno a una persona o cosa haciendo reformas en ella:

mil técnicos medios en informática, a los que se unen mas de 11 800 estudiantes a nivel superior, de ellos 10 mil de la Universidad de la Ciencias Informáticas (UCI).

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.

1.1.6 Situación Actual de la Seguridad de Software en las Aplicaciones Web.

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[12] 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 (9).

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

[12] **Hacking:** Utilizado para referirse a un experto en varias o alguna rama técnica relacionada con la informática: programación, redes de computadoras, sistemas operativos.

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 (10).

A pesar de esto y de las inversiones en materia de seguridad, existen noticias que sitios de empresas, medios de comunicación, entidades gubernamentales, organizaciones políticas y civiles, entre otras, son presas de ataques a la seguridad de sus sitios web.

Las organizaciones invierten recursos, principalmente, en la seguridad informática, en seguridad perimetral[13] con la instalación de firewalls, en Sistemas de Detección de Intrusos (IDS), Sistemas de Prevención de Intrusiones (IPS) y firewalls de aplicaciones, todo apoyado en Sistemas de Administración de Eventos e Incidentes (IEAS). Todos estos controles de seguridad bloquean ataques muy bien identificados, sin embargo, todos ellos descuidan la seguridad en las aplicaciones y la información que en ella se contienen, de tal manera que la seguridad se deja única y exclusivamente a la programación realizada. Sin embargo aquellas fallas de seguridad no contempladas o conocidas, quedan vulnerables a usuarios malintencionados, empleados enojados o en el peor de los casos, a un hacker. Además los ataques a las aplicaciones no siempre son malintencionados, un usuario por error puede lograr tener acceso a información a segmentos de la aplicación que no estaban destinados para su perfil de usuario.

Adicional a esto ha habido un incremento de sitios que proporcionan herramientas para la exploración y explotación de vulnerabilidades en sitios web, lo que ha permitido el crecimiento de la población de estos usuarios malintencionados. Una de las mejores prácticas para mantener seguras las aplicaciones es “nunca asumir que la información ingresada por el usuario será confiable”, esto evita que la aplicación acepte información o datos que no están permitidos y que podrían convertirse en datos potencialmente peligrosos para la seguridad de la misma.

En la Universidad de las Ciencias Informáticas (UCI) una de las formas de controlar la seguridad en las aplicaciones web es mediante la autenticación, de esta forma se otorga o no los privilegios según el rol que ocupan dentro de la aplicación. Los administradores de estas aplicaciones deben ser personas con un alto nivel de

[13] **Seguridad perimetral:** Asume la integración de elementos y sistemas, tanto electrónicos como mecánicos, para la protección de perímetros físicos, detección de tentativas de intrusión y/o disuasión de intrusos en instalaciones especialmente sensibles.

compromiso para evitar que se puedan manejar los datos internos de la aplicación por personas no autorizadas.

1.2 Pruebas de Software.

La prueba es un proceso de ejecución de un programa con la intención de descubrir errores. Un buen caso de prueba es aquel que tiene una alta probabilidad de mostrar un error no descubierto hasta entonces (11).

La mayoría de las grandes organizaciones asumen la responsabilidad del control de calidad y prueba de software a tal medida que en la producción se incluyen desarrolladores de sistemas (analistas, programadores) y un grupo dedicado a la prueba de software para que estos grupos antes mencionados trabajen en conjunto cumpliendo el control de calidad (prevención) y la prueba de software (detección) logrando una tarea exitosa.

1.2.1 Objetivos de las Prueba.

- ✓ Encontrar y documentar los defectos que puedan afectar la calidad del software.
- ✓ Validar que el software trabaje como fue diseñado.
- ✓ Validar y probar los requisitos que debe cumplir el software.
- ✓ Validar que los requisitos fueron implementados correctamente.

1.2.2 Puntos clave.

- ✓ La prueba no puede asegurar la ausencia de defectos, solo pueden demostrar que existen defectos en el software.
- ✓ Cada prototipo que se quiera entregar al final de una iteración debe ser probado y evaluado.

1.2.3 Niveles de Prueba.

La Prueba es aplicada para diferentes tipos de objetivos, en diferentes escenarios o niveles de trabajo (12).

Se distinguen los siguientes niveles de pruebas:

Prueba de Unidad: Pruebas a nivel de clases.

Prueba de Integración: Pruebas a nivel de componentes. Donde existen dos tipos: las ascendentes que son basadas en hilos y las descendentes con regresión basadas en uso.

Prueba de sistema: Pruebas al software funcionando como un todo.

Prueba de desarrollador: Diseñada e implementada por el equipo de desarrollo.

Prueba independiente: Diseñada e implementada por alguien independiente del grupo de desarrolladores.

Prueba de aceptación: Pruebas pilotos realizadas por los clientes.

Las pruebas del software se encuentran en cada etapa del desarrollo del software. En la medida que el trabajo de los desarrolladores va aumentando el volumen de la aplicación, las pruebas van cambiando de estrategias y técnicas, presentándose como una nueva etapa, estas están definidas en niveles que tienen nuevos objetivos, entornos y resultados.

1.2.3 Tipos de Pruebas.

Cada tipo de prueba tiene un objetivo específico y una técnica que lo soporte. La siguiente tabla muestra los tipos de pruebas basado en dimensiones de calidad (13).

Dimensión de Calidad/Riesgos de Calidad	Tipos de prueba
Usabilidad.	Prueba enfocada 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.
Fiabilidad.	**Integridad:** Enfocada a la valoración de la robustez (resistencia a fallos). **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. **Stress:** Enfocada a evaluar cómo el sistema responde bajo condiciones anormales (extrema sobrecarga, insuficiente memoria, servicios y hardware no disponible, recursos compartidos no disponible).
Rendimiento.	**Benchmark:** Es un tipo de prueba que compara el rendimiento de un elemento nuevo o desconocido a uno de carga de trabajo de referencia conocido. **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). **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. **Performance profile:** 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.
Soportabilidad.	**Configuración:** Enfocada a asegurar que funciona en diferentes configuraciones de hardware y software. Esta prueba es

	implementada también como prueba de rendimiento del sistema. **Instalación:** Enfocada a asegurar la instalación en diferentes configuraciones de hardware y software bajo diferentes condiciones (insuficiente espacio en disco, etc.).
Funcionalidad.	**Función:** Pruebas fijando su atención en la validación de las funciones, métodos, servicios, caso de uso. **Seguridad:** Asegurar que los datos o el sistema solamente es accedido por los actores deseados. **Volumen:** Enfocada en verificando las habilidades de los programas para manejar grandes cantidades de datos, tanto como entrada, salida o residente en la BD.

1.2.5 Pruebas de Seguridad en Aplicaciones Web.

Una buena práctica de seguridad es la realización de pruebas para verificar y comprobar las mejoras en el diseño y programación. En la etapa de pruebas, de igual manera existen dos formas de ser realizadas, manualmente o mediante una herramienta que automatice estas pruebas.

La seguridad y garantía de la información que reside en las aplicaciones web no es un tema que se deje a la ligera, requiere de toda la atención ya que existe todo un ejército de personas que están dispuestas a tomarse el tiempo de encontrarle las fallas al sitio que escojan y de esa forma poner en riesgo a las empresas, independientemente de su tamaño. 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.

En Cuba no se realizan las pruebas de seguridad a aplicaciones web, lo que sí hacen los desarrolladores es tratar de producir un software seguro, pero no utilizan técnicas para probar esto.

En la Universidad de las Ciencias Informáticas (UCI) de igual forma se cumple esto, es decir, tampoco se realizan pruebas a aplicaciones web.

1.2.6 Tipos de pruebas de seguridad para aplicaciones Web.

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:** Las inspecciones 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.

- ✓ **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.

- ✓ **Revisión de Código:** 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. 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 a comprobar el software cerrado de terceras partes, como sistemas operativos, cuando se realizan pruebas de aplicaciones web, 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 pruebas o análisis, como la prueba de penetración, haciendo del análisis de código la técnica preferida para las comprobaciones técnicas. Con el código fuente, la persona comprobándolo puede determinar con exactitud que está pasando (o qué se supone que está pasando), y eliminar el trabajo de adivinar la prueba de caja negra. Algunas de sus ventajas: eficacia e integridad, precisión y rapidez , dentro de sus desventajas tenemos que requiere desarrolladores de seguridad altamente competentes, no puede detectar errores en tiempo de ejecución con facilidad y el código fuente realmente en uso puede ser diferente del que está siendo analizado.

- ✓ **Pruebas de Penetración:** Este tipo de prueba se ha convertido desde hace muchos años en una técnica común empleada para comprobar la seguridad de una red. También son conocidos comúnmente como pruebas de caja negra o hacking ético. Las pruebas de penetración 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. Mientras que las pruebas de penetración han

demostrado ser efectivas en seguridad de redes, la técnica no se traslada de forma natural al caso de aplicaciones. Cuando se realizan pruebas de penetración 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 penetración en el campo de aplicaciones web es más similar a investigación pura. 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.

1.2.7 Pruebas de seguridad

Dentro de la técnica de penetración se realizan una serie de pruebas de seguridad las cuales se muestran a continuación:

Prueba de Intrusión de Aplicación Web.

Una prueba de intrusión o penetración 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.

El proceso conlleva un análisis activo de la aplicación en busca de cualquier debilidad, fallos técnicos o 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.

Pruebas de Firma Digital de Aplicaciones Web.

La determinación de la firma digital de aplicaciones 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[14] adecuados a emplear durante las pruebas.

Pruebas de Gestión de Configuración de la Infraestructura.

[14] **Exploit:** Programa o técnica que aprovecha una vulnerabilidad.

A menudo, los análisis de la infraestructura y topología de la arquitectura pueden revelar una gran cantidad de datos sobre la aplicación web. Se pueden obtener así informaciones como el código fuente, métodos HTTP permitidos, funcionalidad administrativa, sistemas de autenticación y configuraciones de infraestructura.

Claramente, concentrarse tan solo en la aplicación web no será una prueba exhaustiva. No puede ser tan completo como la información recopilada mediante un análisis más amplio de la infraestructura.

Pruebas de SSL/TLS

SSL y TLS 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.

Teniendo en cuenta la criticidad de estas implementaciones de seguridad, es importante verificar el uso de un algoritmo de cifrado robusto, y su implementación adecuada.

Pruebas del Receptor de Escucha de la BBDD.

Durante la configuración de un servidor de base de datos, muchos administradores de bases de dato (BBDD) no toman en consideración adecuadamente la seguridad del componente receptor de escucha de la base de datos. El receptor puede revelar información sensible, así como ajustes de la configuración o instancias de la base de datos en ejecución, si está configurado de forma insegura, y se comprueba con técnicas manuales o automatizadas. La información desvelada a menudo nos será de utilidad, sirviéndonos como una puerta de entrada a otras pruebas derivadas, de mayor impacto.

Pruebas de la Gestión de Configuración de la Aplicación.

Las aplicaciones web, en ocasiones ocultan alguna información que no se toma en consideración normalmente durante el desarrollo o configuración de la propia aplicación. Estos datos pueden ser descubiertos en el código fuente, en archivos de registro o a través de los códigos de error por defecto de los servidores web.

Pruebas de Manejo de Extensión de Archivos.

Las extensiones de archivo presentes en un servidor o aplicación web hacen posible identificar las tecnologías que componen la aplicación objetivo, por ejemplo las extensiones jsp y asp. Las extensiones de archivo también exponen sistemas adicionales conectados a la aplicación.

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 (14).

Pruebas de Gestión del Caché de Navegación y de Salida de Sesión.

El objetivo de esta prueba es comprobar 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.

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 del servidor web, o bien de implementaciones a medida.

Pruebas de Cifrado y Vulnerabilidades por Reutilización de Testigos de Sesión.

La protección frente al acceso a la información se realiza habitualmente mediante el cifrado SSL, pero puede incorporar otro sistema de túnel o cifrado. Debe destacarse que el cifrado o el hash criptográfico de los identificadores de sesión deberá de considerarse por separado del cifrado del transporte, así que el identificador de sesión estará protegido en si mismo, y no los datos representados por él. Si el identificador de sesión puede ser presentado por un atacante para obtener acceso a la aplicación, entonces deberá de ser protegido en el tránsito para reducir este riesgo. Deberá por lo tanto garantizarse que el cifrado en ambos por defecto esté forzado para cualquier petición o respuesta donde se transmita el identificador de sesión, independientemente del mecanismo.

Inyección SQL.

Hablamos de pruebas de Inyección SQL cuando intentamos inyectar una determinada consulta SQL directamente en la base de datos, sin que la aplicación haga una validación adecuada de los datos. El objetivo es manipular los datos en la base de datos, un recurso vital para todas las empresas. Una inyección SQL explota el siguiente patrón: Entrada -> Consulta SQL == Inyección SQL.

Inyección XML.

Hablamos de pruebas de inyección XML cuando tratamos de inyectar un determinado documento XML en la aplicación: si el intérprete XML falla al realizar una validación adecuada de los datos, la prueba resultará positiva.

Una inyección XML explota el siguiente patrón:

Entrada -> documento XML == Inyección XML.

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 búfer.

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.

Pruebas de Denegación de Servicio.

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.

1.2.8 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. A continuación se cubren 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. (15)

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.

Piensa estratégicamente, no tácticamente.

En los últimos años, los profesionales de la seguridad han llegado a darse cuenta de la falacia 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 figura 1. 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 (16).

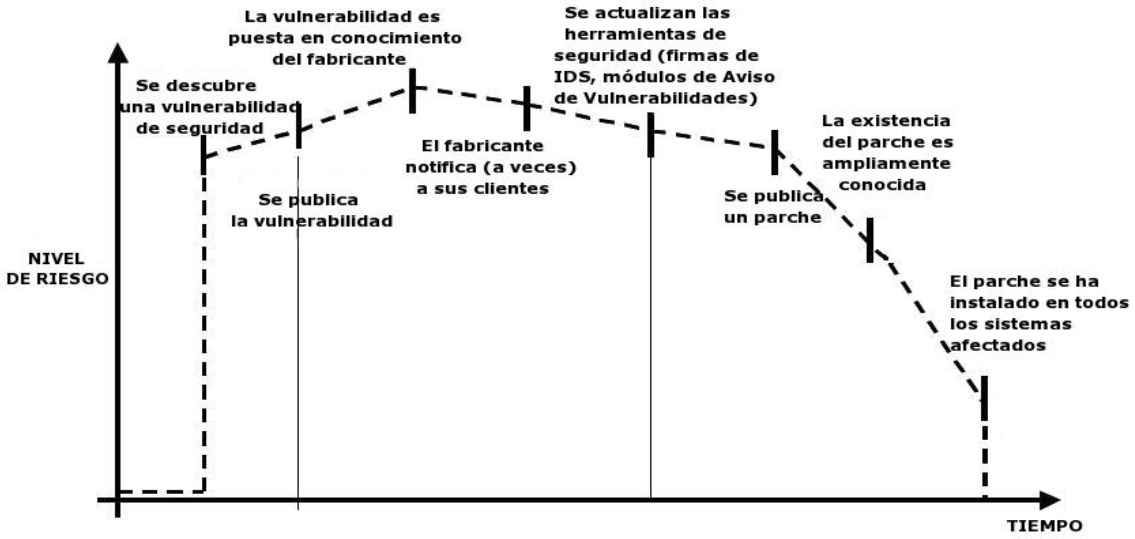


Figura 1: Ventana de exposición.

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. Teniendo en cuenta que mientras los nombres de las 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 de sus responsabilidades.

Comprende el alcance de la seguridad.

Es importante saber cuanta seguridad requerirá un proyecto determinado. A la información y activos[15] 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 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 creativamente puede ayudar a determinar que datos no esperados pueden causar que una aplicación falle de modo inseguro. Esta es una de las razones por las que las herramientas automatizadas son realmente malas en la

[15] **Activos:** Es el conjunto de bienes tangibles o intangibles que posee una empresa. Se considera activo a aquellos bienes que tienen una alta probabilidad de generar un beneficio económico a futuro y se pueda gozar de los beneficios económicos que el bien otorga.

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 tus aplicaciones y red (por ejemplo, sistemas IDS).

Utiliza las herramientas adecuadas.

Aunque ya hemos enunciado que no existe una herramienta bala de plata, las herramientas juegan un papel crítico en el programa de seguridad global. Hay toda una gama de herramientas de códigos abiertos y comerciales 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 cualquier falso positivo que pueda quedar en el informe. Informar de un resultado de seguridad hallado que sea incorrecto puede desacreditar el resto de mensajes válidos de un informe de seguridad. Debe tomarse cuidado en verificar que casa 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 penetración 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 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 del 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 OWASP Metrics y otras organizaciones podría ser una buena base de la que partir para empezar.

1.3 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 (17).

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.

Es importante mencionar que una página web puede contener elementos que permiten una comunicación activa entre el usuario y la información. Esto permite que el usuario acceda a los datos de modo interactivo, gracias a que la página responderá a cada una de sus acciones, como por ejemplo rellenar y enviar formularios, participar en juegos diversos y acceder a gestores de base de datos de todo tipo.

1.3.1 Ventajas.

- ✓ No se necesita Instalar nada (No depende de algún software), por lo general.
- ✓ Puedes acceder a ella desde cualquier ordenador con conexión a Internet.
- ✓ Brindan privacidad con acceso (usuario y contraseña) para acceder a tus datos.
- ✓ No necesita actualizarlo.
- ✓ No hay discriminación (generalmente) acerca del sistema operativo del usuario.
- ✓ Una empresa puede migrar de sistema operativo o cambiar el hardware libremente sin afectar el funcionamiento de las aplicaciones de servidor.
- ✓ No se requieren complicadas combinaciones de Hardware/Software para utilizar estas aplicaciones. Solo un computador con un buen navegador web.
- ✓ Se facilita el trabajo a distancia. Se puede trabajar desde cualquier PC o computador portátil con conexión a Internet.
- ✓ Actualizar o hacer cambios en el Software es sencillo y sin riesgos de incompatibilidades.
- ✓ Existe solo una versión en el servidor lo que implica que no hay que distribuirla entre los demás computadores. El proceso es rápido y limpio.
- ✓ Al funcionar en un navegador, se requiere un conocimiento básico de informática para utilizar una aplicación web (18).

1.4 Herramientas para Pruebas Automatizadas de Seguridad.

Las herramientas automatizadas dan soporte a las actividades de calidad, pruebas de integración, pruebas de sistema, diagnóstico o afinado de las aplicaciones en los

entornos de preproducción o certificación y los de producción. Generalmente son difíciles de realizar de una forma integrada, comprenden herramientas de análisis estático, automatización de pruebas funcionales, de carga, de rendimiento, de estrés.

La relación de herramientas de mayor utilidad para los entornos de pruebas es:

Shadow Security Scanneres (SSS).

En esta nueva generación de programas de alta tecnología que han sido realizados en el siglo XX y permanecen en uso en el nuevo milenio. Security Scanner es un completísimo y efectivo escáner de vulnerabilidad es para la plataforma de Windows nativa, aunque también examina servidores de cualquier otra plataforma revelando brechas en Unix, Linux, FreeBSD y Net BSD.

Por su arquitectura, SSS también descubre fallos en CISCO, HP y otros equipos de la red. Actualmente, los servicios analizados son: FTP, SSH, Telnet, SMTP, DNS, Finger, HTTP, POP3, IMAP, IRC, Terminal Service, NetBIOS, NFS, NNTP, SNMP, Squid, LDAP, HTTPS, SSL, TCP/IP, UDP y servicios del registro. Es importante destacar el excelente desempeño que esta herramienta posee analizando servicios TCP/IP, HTTPS, FTP, UDP, Registro, Windows Media Service, así como cualquier otro de los que soporta.

SSS ha sido desarrollado para proveer una detección segura, rápida y fiable de amplio rango de fallas de seguridad en los sistemas. Luego de completar el análisis del sistema, SSS analiza los datos recogidos, localiza las vulnerabilidades y posibles errores en las opciones de ajuste del servidor, y sugiere posibles formas de solucionar el problema, además de actualizarse automáticamente a través de Internet. Su nueva versión ya esta actualmente en el mercado Shadow Security Scanner 7.25. SSS emplea un algoritmo de análisis de sistemas bastante potente.

Es un analizador de seguridad en el sector del mercado analizando el rendimiento de muchas marcas famosas. SSS emplea un algoritmo de análisis de la seguridad de sistemas basado en un "núcleo intelectual" patentado. SSS realiza el análisis del sistema a una velocidad y precisión capaz de competir con servicios profesionales de seguridad de hackers, intentando ingresar en su red. Es además el único analizador comercial capaz de rastrear más de 4.000 auditorias por sistema (19).

Nessus.

Tal como se menciona en su sitio web oficial, el inicio del proyecto “Nessus”, data del año 1998, momento en el cual Renaud Deraison comenzara a trabajar en el, con la idea de dotar a la comunidad de un escáner remoto de seguridad el cual fuera fácil de utilizar y actualizar, lo suficientemente poderoso y por sobre todas las cosas libre y de código abierto.

Hoy, casi veinte años después del comienzo de este proyecto, Nessus ostenta un lugar de privilegio entre los productos de su tipo, es considerado un estándar de facto y se estima que el mismo es utilizado por 100.000 organizaciones alrededor del mundo. Sin lugar a dudas, muchos son los motivos que han hecho de Nessus, una herramienta “Estrella”, aunque probablemente gran parte de su éxito se deba a la maravillosa arquitectura sobre la cual se encuentra construido. Esta se basa en una serie de componentes fundamentales para su ejecución, entre los cuales se encuentran las porciones Cliente/Servidor y una serie de plugins específicamente desarrollados para lanzar las más diversas pruebas de seguridad.

El servidor es el encargado de comprobar la seguridad de un equipo y el cliente es el responsable de realizar las peticiones. Podríamos decir que el servidor es el motor y el cliente simplemente el entorno gráfico. Se puede utilizar ambas partes del programa en un único ordenador, de forma que la propia PC realice peticiones así mismo de análisis, o ejecutar el servidor en un equipo potente realizando las peticiones desde un ordenador menos preparado. Esto influye en la velocidad a la que se realizan las pruebas, aunque no a los resultados finales.

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 buffer overflow[16].

El escáner de vulnerabilidad Nessus es el líder mundial en escáneres activos, destacando el descubrimiento de alta velocidad, la revisión de configuración, el activo descubrimiento de datos copiador, sensible y el análisis de vulnerabilidad de su postura de seguridad .Nessus escáner puede ser distribuido en todas partes de una empresa entera y a través de redes físicamente separadas.

Es la herramienta de evaluación de seguridad "Open Source" de mayor renombre. Nessus es un escáner de seguridad remoto para Linux, BSD, Solaris y Otros Unix.

[16] **Overflow:** Desbordamiento, sobrecarga.

Está basado en plugins, tiene una interfaz basada en GTK, y realiza más de 1200 pruebas de seguridad remotas. Permite generar reportes en HTML, XML, LaTeX, y texto ASCII; también sugiere soluciones para los problemas de seguridad.

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.

Los plugins[17] son el "corazón" de Nessus. Ellos son las pruebas de seguridad, esto significa descubrir una vulnerabilidad determinada. NASL (Nessus Attack Sripting Language) es un leguaje recomendado para escribir pruebas de seguridad.

Nessus es un Scanner de Vulnerabilidades que más o menos Simula Ataques de Intrusión a nuestro Equipo.

Existen aproximadamente 20 familias de plugins: puertas traseras, denegación de servicio, lograr accesos remotamente. Como ya lo mencionamos, cada plugins reporta información (20).

Brutus.

Brutus es un excelente crackeador de contraseñas remoto on-line muy reconocido por su rapidez (puede llegar a 2500 palabras por segundo) y su eficaz y cómodo diseño. Se trata de un auténtico todo terreno en el mundo del Crack.

Brutus abarca una extensa lista de tipos de autenticación que puede llegar a crackear, tales como HTTP (Autentificación Básica), HTTP (Autenticación por Formulario HTML), POP3, FTP, SMB, Telnet, y otros tipos tales como IMAP, NNTP y NetBus.

Entre sus principales características se pueden enumerar las siguientes:

- ✓ Posee un motor gradual de la autentificación.

[17] **Plugins:** Podríamos traducirlo por añadido o conector. Se trata de un pequeño programa que proporciona alguna funcionalidad específica a otra aplicación mayor o más compleja.

- ✓ Permite más de 60 conexiones simultáneas (Se podría tirar abajo algún servicio).
- ✓ Obtiene parejas de usuario / contraseña mediante ataque simple con diccionario o mediante fuerza bruta.
- ✓ Dispone de un generador de diccionarios bastante completo.
- ✓ Guarda sesiones y las continúa posteriormente. Se pueden exportar e importar.
- ✓ La versión corriente de Brutus es ' Brutus AET2 ', fue liberado el 28 de enero de 2000. El tamaño de archivo es 331 kilobytes.

Nikto.

Nikto es un escaneador para servidores web realizado en lenguaje Perl, obviamente tiene licencia Open Source GPL y realiza todo tipo de pruebas de ataques y vulnerabilidades por medio de un extensible sistema de plugins.

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.

Este programa busca fallos en diferentes categorías, algunas de estas son:

1. Problemas de configuración.

2. Archivos por defecto y ejemplos.

3. Archivos y scripts inseguros.

4. Versiones desactualizadas de productos.

Es utilizado tanto para seguridad como para buscar vulnerabilidades en servidores. Nikto utilizado como herramienta de seguridad 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.

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.

Se han documentado dos vulnerabilidades recientes, basadas en el mismo concepto, que permitirían construir ataques de Cross-Site Scripting vía inyección de scripts en sistemas donde se empleen las aplicaciones de auditoría Nikto (UNIX y derivados).

El problema matriz deriva del como escáneres de servidores web, no hacen uso de un error de saneamiento del valor de " cabecera del servidor " en la respuesta HTTP que ofrecen las webs auditadas, de manera previa a la generación del informe de auditoría. Esto podría ser empleado para ejecutar código HTML arbitrario en la zona de seguridad local, una vez se visualicen los informes de la aplicación, siendo perfectamente factible la inserción de scripts de carácter malintencionado en el código del informe.

Un atacante podría construir un sitio destinado a responder maliciosamente a las peticiones HTTP de sendas aplicaciones de auditoría, y pese al carácter moderadamente crítico del problema, la explotación exitosa pasa únicamente por que el usuario lance el análisis de la web maliciosa de un modo intencionado desde la aplicación de auditoría, lo cual reduce la criticidad del problema.

Aunque algunas comprobaciones puedan ser encontradas en otros plugins, el scan_database.db contiene la mayor parte de la información de prueba de web para Nikto. Todos los usuarios pueden crear sus propias pruebas, para ello han de crear una prueba válida con un nombre específico, que ha de comenzar por “u”. Cuando Nikto procede a cargar las pruebas propias se cargarán también aquellas que se encuentren en el directorio plugins. Estos archivos se han de incorporar al archivo nikto_plugin_order.txt, para que sea llamado de forma correcta.

Paros.

Paros es una herramienta especializada en la seguridad web que hace poco presentó su versión 3.2.3. Paros 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) para detectar problemas de seguridad típicos de aplicaciones web.

Está programado en Java pero puede usarse para analizar aplicaciones web realizadas con cualquier tecnología. Licencia Clarified Artistic License. Paros proxy 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.

Finalmente es posible generar plugins propios para incluir dentro de Paros, un ejemplo es el plugins de BitWise de Mike Shema, que mediante inyección SQL en bases de datos con MS SQL Server permite enumerar los siguientes elementos:

- Nombre de la base de datos.
- Nombre del usuario que establece la conexión con la base de datos.
- Hash del usuario SA.
- Todas las tablas de la base de datos.

Es importante recordar que las herramientas que identifican vulnerabilidades de forma automática generan falsos positivos, por lo tanto será necesario revisar la información que nos proporciona Paros Proxy para garantizar que los resultados son correctos.

WebScarab

WebScarab es un marco para el análisis de las aplicaciones que se comunican a través de los protocolos HTTP y HTTPS. Está escrito en Java, y por lo tanto es portable a muchas plataformas. WebScarab tiene varios modos de funcionamiento, ejecutados por una serie de plugins. En su uso más común, WebScarab funciona como un proxy de interceptar, que permite al operador revisar y modificar las peticiones creada por el navegador antes de que sean enviados al servidor, y para revisar y modificar las respuestas de regresar de el servidor antes de que sean recibidos por el navegador. El operador también puede revisar las conversaciones (peticiones y respuestas) que han pasado por WebScarab.

WebScarab está diseñado para ser una herramienta que cualquiera que necesite exponer la funcionalidad de una aplicación basada en HTTP(S), ya sea para permitir al desarrollador depurar problemas difíciles o permitir a un especialista en seguridad identificar vulnerabilidades mientras la aplicación está siendo diseñada o implementada.

Un marco de trabajo sin ninguna función que no valga la pena, por supuesto, WebScarab provee un número de plugins, cuyo objetivo principal, por el momento, es agregar funcionalidad de seguridad. Estos plugins incluyen:

- Fragmentos - extraer los scripts y comentarios de las páginas HTML en el momento en que son vistos por el proxy y otros plugins.

- Proxy - Observa el trafico entre el navegador y el servidor Web. El proxy de WebScarab es capaz de observar tanto HTTP como tráfico HTTPS cifrado al negociar una conexión SSL entre WebScarab y el navegador en ves de simplemente conectar el navegador al servidor y permitir que un flujo de datos cifrado pase por él. Varios plugins del proxy has sido también desarrollados para permitir al operador controlar las peticiones y respuestas que pasan por el proxy.

- Intercepción Manual - permite al usuario modificar peticiones y respuestas HTTP y HTTPS, antes de que ellas alcancen el servidor o el navegador.

- Beanshell - permite la ejecución de operaciones arbitrarias complejas en las peticiones y respuestas. Cualquier cosas que pueda ser expresada en Java puede ser ejecutada.

- Revelar campos ocultos - algunas veces es mas fácil modificar un campo oculto en la página misma, más que interceptar la petición después que ha sido enviada. Este plugins cambia todos los campos ocultos encontrados en las páginas HTML a campos de texto, haciéndolos visibles y editables.

- Simulador de ancho de banda - permite al usuario emular una red mas lenta, de manera que observe como se desempeña su sitio con es accedido, por ejemplo, desde un modem.

- Araña (Spider) - identifica nuevas URLs en el sitio objetivo y obtiene e contenido cuando se le indica.

- Peticiones manuales - permite editar y reenviar peticiones anteriores o la creación de peticiones nuevas completas.

- Scripted - los operadores pueden usar BeanShell para escribir un script para crear peticiones y obtenerlas del servidor. El script puede entonces realizar algunos análisis en las peticiones, con todo el poder del modelo de objetos de peticiones y respuestas de WebScarab para simplificar las cosas.

- Ofuscador de parámetros - realiza la sustitución automatizada de valores en los parámetros que es probable que muestre una validación incompleta de parámetros que lleve a vulnerabilidades como secuencia de comandos en sitios cruzados (XSS) o inyección de SQL**.**

- Búsqueda - permite al usuario crear expresiones arbitrarias de BeanShell para identificar conversaciones que deben ser mostradas en la lista.

- Comparación - calcula la distancia de edición en el cuerpo de la respuesta de la conversación observada y una conversación predeterminada. La distancia de edición es el número de ediciones requeridas para transformar un documento en otro. Por razones de desempeño, las ediciones son calculadas usando testigos de palabras, más que byte por byte.

- SOAP - hay un plugins que interpreta WSDL, y presenta las varias funciones y los parámetros requeridos, permitiendo que sean editadas antes de que sean enviadas al servidor.

- Extensiones - automatiza las revisiones de archivos que fueron dejados por error en el directorio raíz del servidor .Las revisiones son realizadas en archivos y directorios. Las extensiones para archivos y directorios pueden ser editados por el usuario.

- XSS/CRLF - este plugins de análisis pasivo busca datos controlados por el usuario en los encabezados y cuerpo de las respuestas HTTP para identificar posibles inyecciones CRLF (partición de respuesta HTTP) y vulnerabilidades de secuencia de comandos en sitios cruzados (XSS).

1.4.1 El papel de las herramientas automatizadas.

Existen varias compañías 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.

Lo más importante a remarcar es que estas herramientas son genéricas es decir, que 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. Por experiencia en el tema, las incidencias de seguridad más serias son

aquellas que no son genéricas, sino profundamente intrincadas en tu lógica de negocio y diseño a medida de la aplicación.

Estas herramientas pueden también ser atractivas, ya que encuentran muchas incidencias potenciales. Aunque ejecutar estas herramientas no toma mucho tiempo, cada uno de los problemas potenciales toma su tiempo investigar y verificar. Si el objetivo es encontrar y eliminar los fallos más serios tan rápidamente como sea posible.

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.

1.4.2 Tabla Comparativa.

Aspectos/Herramientas	**Nessus**	**Nikto**	**Paros**	**WebScarab**
Tegnología que soporta.	IMAP,SMTPPOP3,	CGI, Perl		Java
Gratuita	SI	SI	SI	SI
Última versión	Nessus 3.x	V2.04	V3.2.3	WebScarab -NG
Tipos de Pruebas	P1 P2 P4	P3 P5	P3 P5	P3 P5

Leyenda:

P1: Prueba de intrusión de aplicación web.

P2: Pruebas de denegación de servicio.

P3: Inyección SQL.

P4: Firma digital de aplicaciones.

P5: Cross-Site Scripting.

1.5 Conclusiones del capítulo:

En el presente capítulo se han descrito los elementos teóricos sobre los cuales se sustentará la propuesta del procedimiento para pruebas de penetración en aplicaciones web. Se definen los conceptos generales de pruebas, sus tipos, su automatización. Se destaca un estudio de las principales herramientas, a través de una tabla comparativa, los tipos de pruebas de seguridad en aplicaciones web, los principios de pruebas de seguridad, etc. De la investigación realizada se puede concluir que no existe actualmente una guía para llevar a cabo las pruebas de penetración en aplicaciones web.

CAPÍTULO 2: PROPUESTA DEL PROCEDIMIENTO PARA PRUEBAS DE PENETRACIÓN.

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 una propuesta de solución para garantizar la realización de pruebas de penetración en aplicaciones web. Esta idea surge por la necesidad que hay en estos momentos en los proyectos productivos de la Universidad de las Ciencias Informáticas (UCI), 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 Laboratorio Industrial de Pruebas de Software (LIPS), donde el software llega completamente elaborado por lo cual el procedimiento será para la técnica de prueba de penetración.

2.1 Pruebas y herramientas que se van a incluir en la propuesta de procedimiento.

Categoría de Prueba	Descripción	Tipo de prueba	Descripción	Herramienta
Recopilación de Información	Comprobar si la aplicación brinda datos sensibles que puedan ser utilizados por cualquier atacante.	Firma digital de aplicaciones.	La determinación de la firma digital de aplicaciones es el primer paso del proceso de recopilación de información.	**Nessus**
		Descubrimiento de aplicaciones.	Es el proceso destinado a identificar aplicaciones web instaladas en una infraestructura dada.	
		Análisis de códigos	El factor más importante para esta actividad es enfocar la	

		de error.	atención en los errores encontrados en las aplicaciones.	
		Pruebas de SSL/TLS.	SSL y TLS son dos protocolos que proveen, con el apoyo de criptografía, para la protección, confidencialidad etc, de la información transmitida.	
Comprobación de las reglas del Negocio.	Comprobar las reglas del negocio definidas en la aplicación.	Comprobación de las reglas del Negocio	La lógica de negocio puede contener fallos de seguridad que permiten a un usuario hacer algo no permitido por el negocio.	**Nessus**
Comprobación de la autenticación.	Poner a prueba el sistema de autenticación de la aplicación.	Fuerza bruta.	Consiste en averiguar el usuario y contraseña válidos de un individuo registrado en el sistema, mediante el intento reiterado de diferentes combinaciones de usuarios y contraseñas.	**Brutus**

		Recordatorio de contraseñas y Pwd reset.	Esta prueba usa solo características funcionales de la aplicación y código HTML que siempre está disponible al cliente.	
Validación de datos.	Verificar que todas las entradas de datos estén validadas.	Inyección SQL.	El objetivo es manipular los datos en la base de datos, un recurso vital para todas las empresas.	**Nessus**
		Inyección de procedimientos almacenados.	El simple uso de procedimientos almacenados no asiste en la mitigación de inyecciones SQL. Si no se tratan adecuadamente, las consultas dinámicas de SQL en los procedimientos almacenados pueden ser tan vulnerables a inyecciones SQL como lo son las consultas dinámicas desde una página web.	
		Inyección ORM (herramienta para mapear objetos	La inyección ORM es un ataque donde se utiliza inyección SQL contra un modelo de objeto de acceso a datos generado por	

		relacionales).	ORM.	
		Inyección LDAP (protocolo ligero de acceso a directorio).	Este ataque permite alterar los contenidos de las sentencias LDAP que se generan a través de entradas de la aplicación web.	
		Inyección XML.	Hablamos de pruebas de inyección XML cuando tratamos de inyectar un determinado documento XML en la aplicación.	
		Inyección 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.	

		Inyección de código.	Estas pruebas pueden tener como 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 como se trabaja con estas dos herramientas ver los manuales de usuario que se elaboraron. En el Anexo 1 para Nessus y el Anexo 2 para Brutus.

2.2 Procedimiento de Pruebas de Penetración.

Descripción: Este procedimiento definirá de forma detallada los pasos para llevar a cabo las pruebas de seguridad a una aplicación Web.

Alcance: Es aplicable para ser utilizado por los probadores del LIPS y por los probadores de los grupos de calidad de la facultad.

Código	Actividad	Descripción
A.1	Planificación de las pruebas.	Coordinar el comienzo de las pruebas. Se desarrollará un plan de pruebas de Seguridad.
A.2	Diseño de las pruebas.	Diseñar las listas de chequeo y los casos de prueba.
A.3	Ejecución de la pruebas.	Ejecutar las pruebas mediante el uso de las listas de chequeo y los casos de prueba.
A.4	Documentación e Informe de	Documentar y realizar un informe de las

	los resultados.	no conformidades encontradas.
A.5	Depuración de Errores.	Solución de los errores encontrados.

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

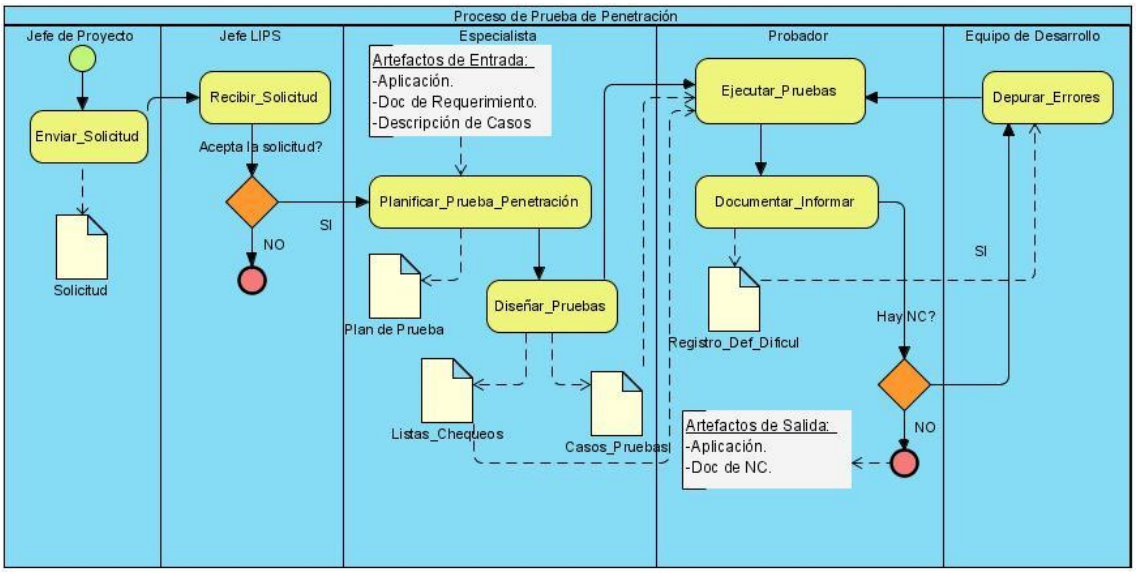


Figura 2: Flujo gráfico del procedimiento.

2.2.1Descripción 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.

2.2.2 A.1 Planificación de las Pruebas

En este paso se planificarán las pruebas que se le 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, calendario, 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” derivada de la plantilla Plan de Prueba de RUP, la cual se define en el siguiente epígrafe. 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.

1. Introducción

Este documento se confecciona con el objetivo de definir el plan de las pruebas de penetración a la aplicación [nombre de la aplicación].

1.1 Objetivos

Los objetivos de este Plan de Pruebas de Penetración 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.
- ✓ Definir el cronograma de las pruebas.

1.2 Alcance

Este procedimiento va a ser utilizado por los estudiantes del laboratorio de pruebas de liberación para realizar pruebas de penetración a aplicaciones web.

2. Recursos

2.1 Roles y responsabilidades

En la siguiente tabla se muestran los roles, y responsabilidad que serán empleados en la realización de las pruebas.

Rol	Cantidad	Responsabilidad

Especialistas de Calidad.	[Total de especialistas asignados].	Elaborar lista de chequeo para realizar las pruebas. 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.
Probador.	[Total de probadores del equipo de pruebas].	Ejecutar las pruebas. Registrar no conformidades.
Desarrollador.	[Total de miembros del equipo de desarrollo 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 o una imagen hecha en Visio, este tópico es responsabilidad del equipo de desarrollo].

2.3 Recursos de Sistema.

Recurso	Tipo
Servidores	
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 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 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 informes diarios.

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

En la figura 3 se muestra de forma general el flujo de trabajo.

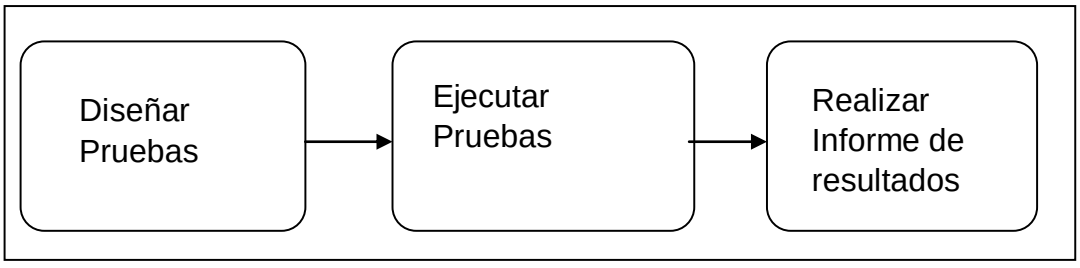


Figura 3: Descripción del Flujo de Trabajo.

Descripción de las estrategias y tipos de pruebas.

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

- ✓ **Primera Fase:**

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

- ✓ **Segunda Fase:**

1. 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.

2. Realización de la categoría de pruebas Comprobación de las reglas del Negocio. Su objetivo es verificar las reglas del negocio definidas en la aplicación.

3. Realización de la categoría de pruebas Comprobación de la Autenticación. Su objetivo es poner a prueba el sistema de autenticación de la aplicación.

4. Realización de la categoría de pruebas Validación de Datos. Su objetivo es verificar que todas las entradas de datos estén validadas.

Cronograma de trabajo.

Cronograma Planificado.

[Cronograma de actividades a desarrollar].

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

3						

Cronograma Real.

[Cronograma según se van realizando las pruebas].

No.	Tarea	Fecha	Responsable	Participantes	Etapa	Observaciones
	Actividad a desarrollar.	Fecha de inicio y fecha de fin de la actividad.	Responsible.			
1						
2						
3						

2.2.3 A.2 Diseño de las pruebas.

En esta actividad se diseñarán los casos los casos de pruebas correspondientes a las categorías de pruebas Comprobación de las Reglas del Negocio y Validación de Datos. Además se elaborarán las listas de chequeo que se utilizarán en las categorías de pruebas Recopilación de Información y Comprobación de la Autenticación. A continuación se muestra la estructura de las listas de cuequeo 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.

Estas listas de chequeos y casos de pruebas que se explicarán a continuación medirán de forma general los aspectos que se deben tener en cuenta. En caso de surgir nuevos aspectos a medir estos se adicionarán.

Lista de chequeo para la Recopilación de Información:

Indicadores a Evaluar	Descripción	Resultado Esperado	Resultado Real	Herramienta
1.1 ¿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 esta en el plan de prueba.		Nessus
1.2 ¿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.		Nessus
1.3 ¿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.		Nessus

1.4 ¿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.		Nessus
1.5 ¿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.		Nessus

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.

Indicadores a Evaluar	Descripción	Resultado Esperado	Resultado Real	Herramienta
¿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 válidos de un individuo registrado en el sistema.	Obtener el usuario y la contraseña de un usuario privilegiado en la aplicación.		Brutus
¿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.		--------
¿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 primero	Enviar la contraseña por correo mediante el sistema.		--------

	una pregunta 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.			
¿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.		--------
¿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	Una vez respondida las preguntas la aplicación envía los datos por correos.		--------

	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.			
¿Se pueden responder las preguntas a través de una búsqueda en internet o mediante un ataque de ingería social?	Busca preguntas que tengan pocas 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.	Responder las preguntas a través de una búsqueda en internet.		--------

¿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.		--------
¿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.		--------
¿Luego de responder las preguntas la aplicación obliga al usuario a cambiar inmediatamente la contraseña?	Una vez se ha encontrado una respuesta correcta a la pregunta.	Cambiar la contraseña una vez que responda las preguntas.		--------

Caso de prueba para la Comprobación de las Reglas del Negocio:

Para realizar este caso de prueba lo primero que se hace es realizar una matriz de privilegios.

Funcionalidad	Rol 1	Rol 2	Rol n
Se adicionan las funcionalidades que brinda la aplicación.	Se marca con una X en caso de que el rol puede realizar esta funcionalidad.	Se marca con una X en caso de que el rol puede realizar esta funcionalidad.	Se marca con una X en caso de que el rol puede realizar esta funcionalidad.

Luego se crea un documento de caso de prueba por rol y en el se pone un escenario para cada funcionalidad a la que el rol no tenga permisos realizar para comprobar si esta funcionalidad puede ser ejecutado ilegalmente por un rol de usuario sin privilegios o con privilegios mínimos.

1. Descripción General.

Acceso.

<Resumen de las funcionalidades a las que el Rol puede acceder.>

No Acceso.

<Resumen de las funcionalidades a las que el Rol no puede acceder.>

Usuario y contraseña.

<Usuario y contraseña en caso que el sistema no esté conectado al dominio uci.>

Nota: Se pone usuario y contraseña en caso que la aplicación no sea para la UCI, es decir que sea para otra institución. En caso de ser así se tendrían que autenticar los probadores que trabajarán con la aplicación.

2. Condiciones de Ejecución.

[Precondiciones.]

3. Funcionalidades a probar para cada rol.

[Para cada rol listar todos los escenarios en los que tiene privilegios y los que no. Comenzar desde la autentificación. Verificar que cada rol accede estrictamente a lo que especifican los requerimientos. Verificar si el rol puede acceder a funcionalidades a las que no tiene privilegios].

Nombre del Rol	Escenarios	Descripción de la funcionalidad a probar	URL	Resultado esperado	Respuesta del sistema
< Nombre del Rol>	<EC 1.1: Nombre del Escenario (solo flujos).	<Descripción de la Funcionalidad.>		<Realizar la funcionalidad a través de páginas no sincronizadas.>	
	EC 1.2: Nombre del Escenario (solo flujos.	<Descripción de la Funcionalidad.>		<Realizar la funcionalidad a través de páginas no sincronizadas.>	
	EC 1.n: Nombre del Escenario (solo flujos básicos).	<Descripción de la Funcionalidad.>		<Realizar la funcionalidad a través de páginas no sincronizadas.>	

Caso de prueba para la Validación de Datos.

La plantilla para el diseño de los casos de pruebas para la validación de datos es la misma que se utiliza actualmente en el LIPS. Esta plantilla tiene el siguiente formato:

- ✓ **Introducción.**
- ✓ **Propósito.**
- ✓ **Alcance.**

[Proyectos con los que se involucra].

- ✓ **Definiciones, Acrónimos y Abreviaturas.**

[Definiciones y acrónimos utilizados en el documento. Deberán escribirse utilizando viñetas y con la palabra a definir o el acrónimo en negrita].

- ✓ **Referencias.**

[Lista de documentos a los que se hace referencia].

Código	Título
[1]	Documento 1.
[2]	Documento 2.
[3]	Modelo de Diseño - Módulo de Administración v0.0.

- ✓ **Desarrollo.**
- ✓ **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.>

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.**

No.	Nombre del campo.	Clasificación.	Valor Nulo.	Descripción.
[1]<Se enumera todos los campos ó variable, descrito en el Caso de Uso.>	<Nombre del campo de entrada>	<La clasificación es según el componente de diseño utilizado][[ejemplo: campo de texto, lista desplegable o campo de selección.>	<Se especifica si el campo puede ser nulo o no> Para ello solo se pone Sí o No.	<Una breve descripción de los datos que deben introducirse (Reglas que tiene que cumplir el campo.).>
[2]				

- ✓ **Matriz de Datos.**

SC 1<Nombre de la Sección. >

Escenario	Variable 1 *(Nombre de la variable)*	Variable 2 *(Nombre de la variabl*	Variable N *(Nombre de la variable)*	Respuesta del Sistema	Resultado de la Prueba	Flujo Central

		e)				
<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.]

2.2.4 A.3 Ejecución de la pruebas.

Se ejecutan 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 Comprobación de la Autenticación, Comprobación de las Reglas del Negocio y Validación de Datos en paralelo o en el

orden que el especialista de prueba decida. El probador ejecuta las pruebas y al mismo tiempo va registrando las no conformidades (NC) detectadas.

2.2.5 A.4 Documentación e Informe de los resultados.

Como se mencionó anteriormente los probadores ejecutan las pruebas y en paralelo van a ir registrando las no conformidades encontradas en el registro de defectos y dificultades. A continuación se muestra la estructura de dicho registro. Además luego que se haya registrado todas las no conformidades encontradas en la ejecución de las pruebas el especialista al frente de las pruebas las revisa y las guarda en el informe de resultados que también se describe a continuación.

Informe de los resultados.

Registro de defectos y dificultades detectados

Elemento	No	No conformidad	Aspecto correspondiente	Etapa de detección	Significativa	No Significativa	Recomendación	Estado NC	Respuesta del Equipo Desarrollo
<Nombre del Elemento>	< 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:	[Esta columna se comienza a llenar a partir de la 2da iteración, y es responsabilidad del equipo de desarrollo, quien especifica la conformidad

								Resuelta PD:Pendiente NP:No Procede	con lo encontrado o no y en caso de no proceder la no conformidad explica por qué.]

2.2.6 A.5 Depuración de los errores.

El equipo de desarrollo se reúne, ven las no conformidades encontradas en la fase anterior y las resuelve.

2.3 Conclusiones del capítulo.

Durante el presente capítulo se desarrolló detenidamente la propuesta de 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 esta constituido el procedimiento, además del diseño de las listas de chequeos y casos de pruebas. Este puede ser utilizado por los probadores del LIPS y por los probadores de los grupos de calidad de la facultad.

CAPÍTULO 3: VALIDACIÓN DEL PROCEDIMIENTO.

Introducción:

En el presente capítulo se evaluará el procedimiento para pruebas de penetración en aplicaciones web a través del método Delphi. Se seleccionarán un grupo de expertos a los cuales se les realizará una encuesta para comprobar la efectividad y eficiencia del procedimiento. Para la validación del procedimiento se empleó la entrevista como método para obtener la información referente al tema, el criterio de los expertos para la validación y aceptación del procedimiento mediante el uso de técnicas propuestas por el método Delphi este es uno de los métodos subjetivos de pronosticación más confiables, su origen parte de la década de los 60, con el objetivo de elaborar pronósticos referentes a posibles acontecimientos en varias ramas de la ciencia, la técnica y la política, además constituye un procedimiento para confeccionar un cuadro de la evolución de situaciones complejas, a través de la elaboración estadística de las opiniones de un grupo de expertos en el tema tratado. Debido a lo anterior es que se ha decidido el uso de este método. Para aplicar el método se siguieron tres etapas fundamentales: (22)

- ✓ Elección de expertos.
- ✓ Elaboración del cuestionario para la validación de la propuesta.
- ✓ Desarrollo práctico y explotación de resultados.

3.1 Entrevista.

La entrevista es una conversación planificada entre el investigador y el entrevistado para obtener información. Su uso constituye un medio para el conocimiento cualitativo de los fenómenos o sobre características personales del entrevistado y puede influir en determinados aspectos de la conducta humana por lo que es importante una buena comunicación. La entrevista puede ser estructurada o no estructurada. La primera se basa en un cuestionamiento fijo, determinado y es aplicado a personas que no son especialistas en el tema, la no estructurada es más abierta que la estructurada, prevé el tema pero no lleva un cuestionario rígido y puede variar de una persona a otra, es mas flexible. Se aplica a especialistas en el tema, es una forma de obtener criterios de expertos.

Para realizar una entrevista es necesario que esta contenga estas tres etapas.

Introducción: Debe comenzar con la puntualidad en la hora prevista para comenzar la entrevista, esto le indica al entrevistado la importancia de la labor que se realiza, la apariencia personal del entrevistador debe adaptarse a las condiciones del entrevistado y tenerse en cuenta la edad y el sexo del mismo, pues la similitud de edad y la diferencia de sexo facilita la comunicación.

Desarrollo: Durante la entrevista el entrevistador debe actuar con naturalidad, no ser dominante ni discutir con el entrevistado, saber escuchar y siempre tener presente que su responsabilidad es captar la mayor información posible, para lo que es necesario hablar poco, observar hasta el último detalle y estimular al entrevistado a que hable.

Conclusión: Cuando se finaliza la entrevista es necesario agradecer al entrevistado su disposición a conceder parte de su tiempo y de sus conocimientos, mostrar respeto por su cooperación y dejar abierto el camino para si es necesario una nueva información.

El éxito que se logre con la entrevista depende del nivel de comunicación que se alcance con el entrevistado, la preparación del investigador, la estructuración de las preguntas, la seguridad que tenga el entrevistado de que no se divulgue la información que esta brindando y sus condiciones sicológicas.

3.2 Selección del grupo de expertos.

Se seleccionaron 7 expertos, tomando como criterio de selección la efectividad de la actividad profesional que realizan, la experiencia que poseen en este tema de la seguridad en aplicaciones web y los años vinculados a esta actividad. De los 7 expertos seleccionados 3 pertenecen al proyecto de seguridad de la facultad 2 y los 4 restantes pertenecen a Laboratorio Industrial de Pruebas de Software (LIPS).

Para seleccionar el grupo de expertos se analizaron los siguientes criterios.

- ✓ Vinculación al desarrollo de productos informáticos.
- ✓ Experiencia en liderazgo de proyectos productivos.
- ✓ Conocimientos y habilidades en actividades de pruebas de software y seguridad.

Para validar el procedimiento para pruebas de seguridad en aplicaciones web se aplicó la encuesta mostrada en el (Anexo-3). Además es importante aclarar que cada buena práctica o consejo dado por estos expertos, está validado por su seriedad, honestidad, sinceridad, responsabilidad y experiencias adquiridas en el tema de la seguridad en aplicaciones web. La validación se realizó por 7 expertos seleccionados según la efectividad de la actividad profesional que realizan y mediante el empleo de técnicas del Método Delphi.

3.2.1 Encuesta para determinar el coeficiente de competencia de los expertos.

Compañero (a):

En la presente tesis, se desea someter a la valoración de un grupo de expertos una propuesta de procedimientos para pruebas de penetración en aplicaciones web, para garantizar una mejor seguridad del software que se produce en la universidad. Para ello necesitamos conocer el grado de dominio que usted posee sobre el tema de las pruebas de penetración en aplicaciones web y con ese fin se desea que responda lo que se le pide a continuación.

Nombre y Apellidos: _____

Labor que realiza: _____

Años de experiencia: _____ **Especialidad:** _____

Categoría docente: _____ **Categoría científica:** _____

1.- Marque con una cruz (X) el grado de conocimiento que UD. tiene sobre la temática que se investiga:

0	1	2	3	4	5	6	7	8	9	10

2.- Marque con una cruz (X) las fuentes que le han servido para argumentar el conocimiento que tiene UD. de la temática que se investiga.

No.	Fuentes de argumentación	Grado de Influencia.		
		Alto	Medio	Bajo
1.-	Análisis realizado por Ud.			
2.-	Experiencia.			
3.-	Trabajos de autores nacionales.			
4.-	Trabajos de autores extranjeros.			
5.-	Su propio conocimiento del tema.			
6.-	Su intuición.			

Gracias por su colaboración.

Resultado Final de la Validación de los indicadores propuestos.

Indicadores a evaluar en la encuesta realizada al grupo de expertos:

1. Criterios de mérito científico.
2. Criterios de implantación.
3. Criterios de flexibilidad.
4. Criterios de impacto.

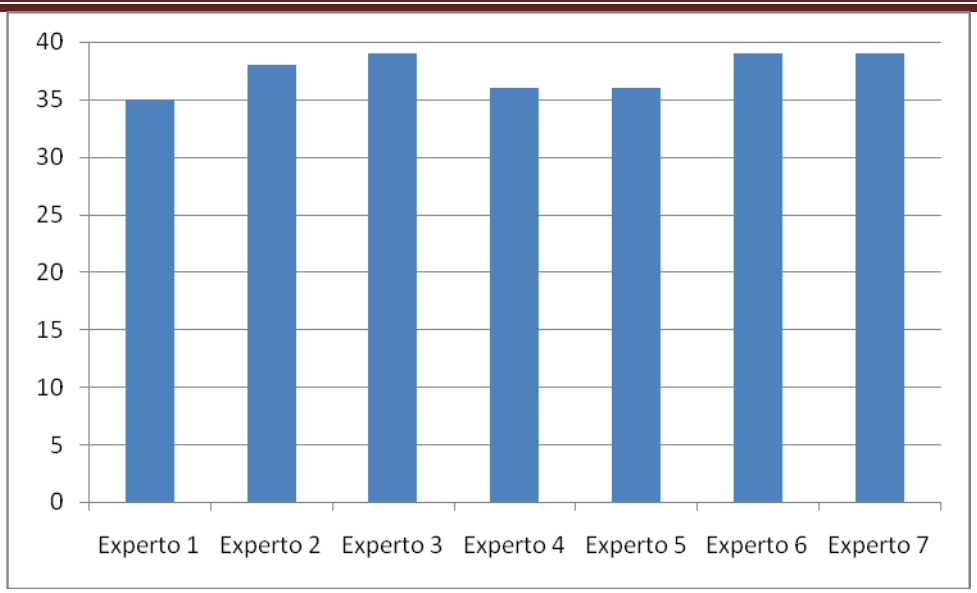


Figura 4: Criterios de mérito científico.

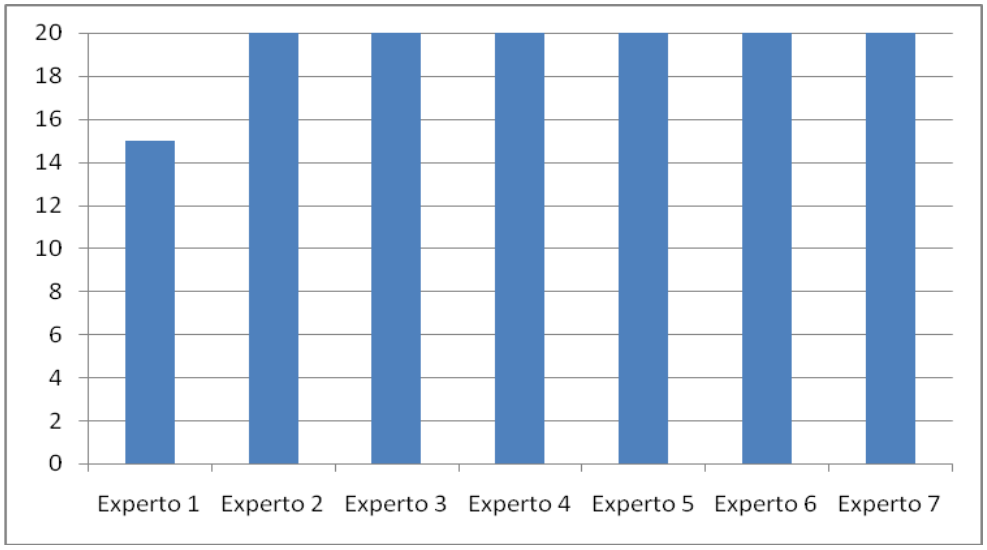


Figura 5: Criterios de implantación.

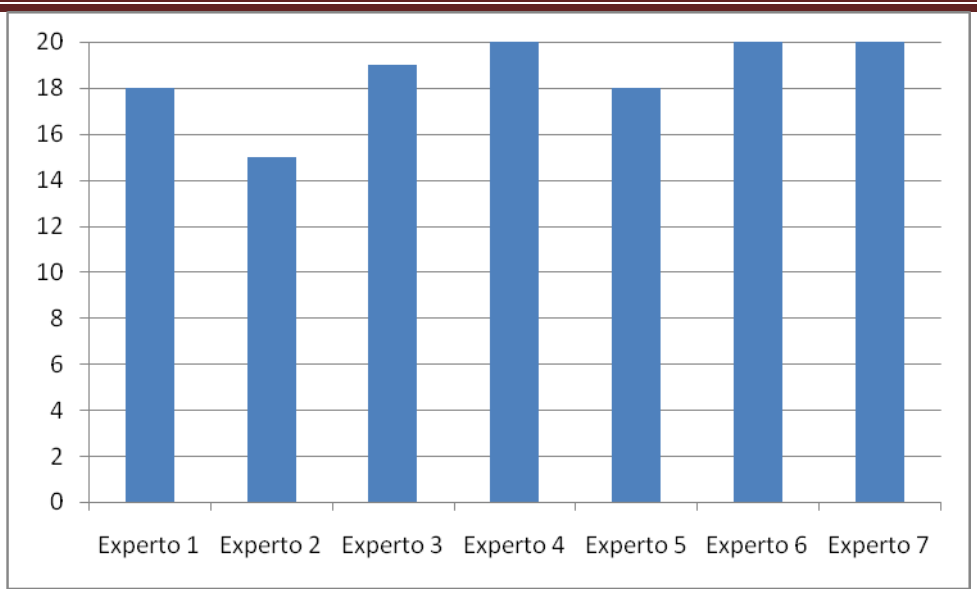


Figura 6: Criterios de flexibilidad.

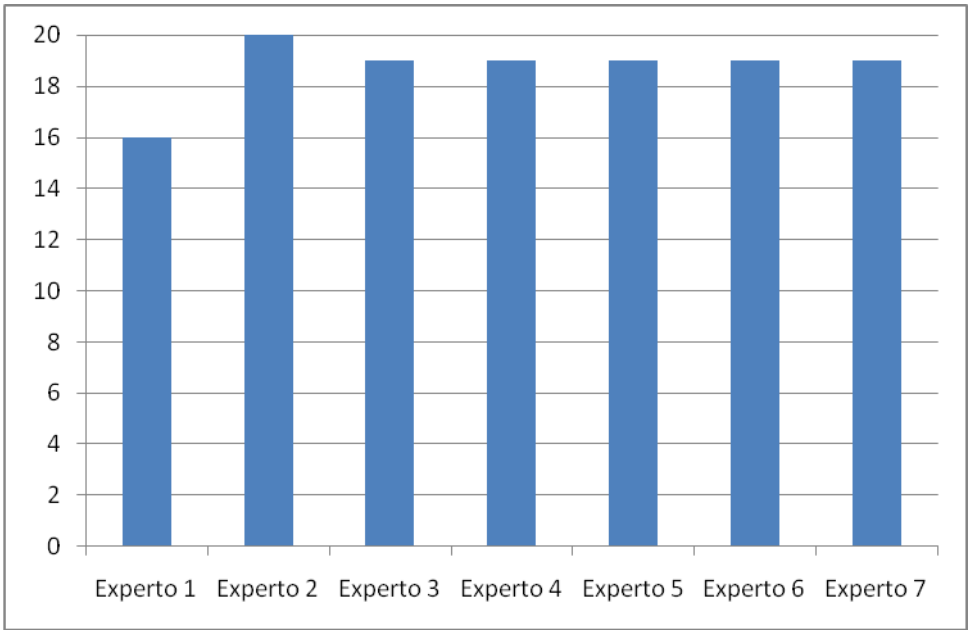


Figura 7: Criterios de impacto.

Resultado de la encuesta realizada a los expertos.

De los 7 expertos seleccionados 2 evaluaron el procedimiento de regular y los 5 restantes lo evaluaron de bueno, los datos se mostrarán en la figura 8.

Figura 8: Evaluación de los Expertos.

3.3 Conclusiones del capítulo.

El análisis de los resultados anteriores permite afirmar que de manera general el procedimiento para pruebas de penetración en aplicaciones web fue evaluado por los expertos como útil, correcto, completo y efectivo para desarrollar, incluyendo que dos de los expertos le dieron una evaluación de regular. Esta evaluación permite concluir que con la aplicación del procedimiento propuesto se aplicarán un conjunto de prácticas relacionadas con las pruebas de penetración. Las recomendaciones dadas por los expertos en las encuestas estuvieron alrededor de perfeccionar el conjunto de pruebas que se le realizan a las aplicaciones de manera que se pueda evaluar cuantitativamente la seguridad y el desempeño del procedimiento propuesto y realizar las mejoras necesarias.

CONCLUSIONES.

Con el cumplimiento de esta investigación se realizó un estudio de las pruebas de penetración que se le realizan a las aplicaciones web, los principales problemas que existe en la Universidad de las Ciencias Informáticas (UCI) actualmente la cual no cuenta en sus proyectos productivos con este tipo de pruebas sino con las pruebas funcionales, de volumen, carga y stress. Además se cumplió el objetivo general y las tareas trazadas alcanzando los siguientes resultados:

- ✓ Se desarrolló un estudio previo sobre los temas relacionados con la seguridad en el desarrollo de las aplicaciones web y los problemas que actualmente afectan a las mismas.

- ✓ Se realizó un estudio sobre las pruebas de penetración que se le realizan al software.

- ✓ Se propusieron dos herramientas automatizadas para la realización de estas pruebas de seguridad.

- ✓ Se desarrolló la propuesta de procedimiento para pruebas de penetración en aplicaciones web para el Laboratorio Industrial de Pruebas de Software (LIPS).

- ✓ Se elaboraron dos manuales de usuarios correspondientes a las dos herramientas propuestas.

- ✓ Se validó el procedimiento mediante el método de Delphi, por un conjunto de 7 expertos.

RECOMENDACIONES.

Existen algunos aspectos que se desprenden de este trabajo investigativo en los cuales serían útiles seguir profundizando y otros que se recomiendan tener en cuenta como son:

- ✓ Aplicar la propuesta del procedimiento para realizar Pruebas de Penetración a Aplicaciones Web en varios proyectos productivos, y se analicen los resultados como una validación práctica del mismo.

- ✓ Emplear las siguientes herramientas: Nessus y Brutus para pruebas de seguridad.

- ✓ Profundizar en el estudio de las pruebas de seguridad en aplicaciones web.

- ✓ Probar y evaluar nuevas herramientas para pruebas de seguridad, estableciendo comparaciones entre las mismas.

REFERENCIAS BIBLIOGRÁFICAS.

1. ESCOBAR YANVARY, F. R., MARTÍNEZ YURAIMA, YANEZ JOEL Desarrollo de software, 21 de Marzo de 2009.

2. VILLEGAS, A. A. Desarrollo de software bajo metodologías agiles, Mayo de 2007.

3. LESGUILLIER, W. Buenas practicas para el desarrollo de software, 4 octubre del 2006.

4. WIKIMEDIA FOUNDATION, I. Sistema de control de calidad de software, 14 diciembre 2008.

5. WILEY, J. Define métricas de calidad, 1807. [Disponible en: http://es.wikipedia.org/wiki/John_Wiley.

6. CAHUICH, L. Características que hacen a un software de calidad, 2009. [Disponible en: www. slideshare.net/lcahuich/calidad-del-software-presentation.

7. Calidad de software una gran preocupación. 14 abril 2009. [Disponible en: http://es.wikipedia.org/wiki/Calidad_de_software.

8. OSCAR M. FERNÁNDEZ, D. G. L. Y. A. B. B. Enfoque actual sobre la calidad del software, septiembre-diciembre, 1995. [Disponible en: http://bvs.sld.cu/revistas/aci/vol3_3_95/aci05395.htm.

9. DOMÍNGUEZ, G. Aplicaciones web seguras, 9 de Diciembre de 2008. [Disponible en: http://www.sg.com.mx/content/view/793.

10. ALIRETH_K. Un 90% de las aplicaciones web son inseguras., 10 de Septiembre de 2005.

11. La puerta a los servicios Web empresariales. 2003. [Disponible en: www.albasoft.com.

12. Pruebas realizadas al software. 25 marzo 2007. [Disponible en: http://lml.ls.fi.upm.es/ftp/ed2/0203/Apuntes/pruebas.

13. GARCERANT, I. Niveles de pruebas, 15-03-08.

14. OPTIMA TECHNOLOGY. Tipos de pruebas, 1991-2007. [Disponible en: www.optima.com.mx/pruebas.htm.

15. La Guía de OWASP para construir Aplicaciones Web Seguras. 2002. [Disponible en: http://www.owasp.org/index.php?title=Proyecto_Guia_de_OWASP&setlang=es

16. Pruebas de seguridad [cross-site scripting]. 30 de marzo de 2009. [Disponible en: http://googlewebmaster-es.blogspot.com/2009/03/practicas-recomendadas-contra-el.html.

17. SCHNEIER, B. Vulnerabilidades, 15 de Septiembre de 2000. [Disponible en: http://www.schneier.com/crypto-gram-0009.html.

18. http://www.wikilearning.com/articulo/actualidad_sobre_el_mundo_de_la_informatica_y_los_hackers-un_90_de_las_aplicaciones_web_son_inseguras/4447-4.

19. MONTALVO, C. Ventajas de las aplicaciones web, 26 de Octubre de 2008. [Disponible en: www.calinsoft.com/2008/08/aplicaciones-web-ventajas-y-desventajas.html.

20. LABS, S. Efectivo escáner de vulnerabilidad es para la plataforma de Windows nativa, 22 de Febrero de 2006. [Disponible en: www.wikilearning.com/tutorial/manual_sobre_shadow_security_scanner/4363.

21. DERAISON, R. Excelente escáner de vulnerabilidades, 1998. [Disponible en: http://shadow-security-scanner.softonic.com/.

22. El método Delphi. 2007.

BIBLIOGRAFÍA CONSULTADA.

1. . COMERCIAL, I. K.-I. *Presentación de la ténica Delphi*, 1990. [Disponible en: http://www.geocities.com/Pentagon/Quarters/7578/pros01.html.

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

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

4. *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.

5. *Seguridad de las aplicaciones Web*. 1999 - 2009. [Disponible en: http://www.sowre.es/wps/portal/enlace?WCM_GLOBAL_CONTEXT=/WebPublica/Servicios/Seguridad+Web/

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

7. UNIDOS, D. P. E. D. D. D. D. L. E., En los años 70. [Disponible en: http://euitio178.ccu.uniovi.es/wiki/index.php/Uso_de_Nikto_como_herramienta_de_seguridad.

8. ALVARO, E. M. *Niveles de pruebas*, 21 Agosto del 2008. [Disponible en: http://www.itbuilder.com.mx/blogs/edgar.alvarado/post/Niveles-de-Prueba-(Levels-of-Testing).aspx.

9. BAÑUELOS, E., 2009. [Disponible en: http://bitelia.com/2009/05/pruebas-de-seguridad-esta-segura-tu-computadora.

10. C., S. C. *Riesgos y seguridad en los sistemas de información*, 2007. [Disponible en: http://ciberconta.unizar.es/leccion/seguro/099.HTM.

11. C., M. V. *Prueba de Software y Seguridad en entornos distribuidos*.

12. DAWES, R. *Marco de trabajo para analizar aplicaciones web.* Disponible en: http://www.owasp.org/index.php/Proyecto_WebScarab_OWASP.

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

14. LAURACELDRANS. *Un pontente escáner de redes.*, 26/05/2005. [Disponible en: http://bulma.net/body.phtml?nIdNoticia=2193

15. LÓPEZ, J. A. P. and L. R. XIRGO. *Introducción al desarrollo de software.*, Marzo 2004. [Disponible en: http://ocw.uoc.edu/informatica-tecnologia-y-multimedia/introduccion-al-desarrollo-de-software/Course_listing**.**

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

17. RACCIATTI., H. M., Noviembre 2005. [Disponible en: http://www.slideshare.net/ernesto.jimenez/seguridad-en-aplicaciones-web.

18. RACCIATTI, H. M. *Seguridad en Aplicaciones Web*, Noviembre 2005. [Disponible en: http://www.hernanracciatti.com.ar/articles/HPP27_Seguridad_en_Aplicaciones_Web.pdf

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

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

21. SPAIN, V. *Pruebas de vulnerabilidad*, 2003-2005. [Disponible en: http://www.verisign.es/static/030193.pdf.

ANEXOS.

Anexos- 1: Manual de Usuario de Nessus.

Manual de usuario de la Herramienta para Pruebas de Seguridad: Nessus

Versión 1.0

Introducción:

Tal como se menciona en su sitio web oficial, el inicio del proyecto “Nessus”, data del año 1998, momento en el cual Renaud Deraison comenzara a trabajar en el, con la idea de dotar a la comunidad de un escáner remoto de seguridad el cual fuera fácil de utilizar y actualizar, lo suficientemente poderoso y por sobre todas las cosas libre y de código abierto.

Hoy, casi veinte años después del comienzo de este proyecto, Nessus ostenta un lugar de privilegio entre los productos de su tipo, es considerado un estándar de facto y se estima que el mismo es utilizado por 100.000 organizaciones alrededor del mundo. Sin lugar a dudas, muchos son los motivos que han hecho de Nessus, una herramienta “Estrella”, aunque probablemente gran parte de su éxito se deba a la maravillosa arquitectura sobre la cual se encuentra construido. Esta se basa en una serie de componentes fundamentales para su ejecución, entre los cuales se encuentran las porciones Cliente/Servidor y una serie de Plugins específicamente desarrollados para lanzar las más diversas pruebas de seguridad.

El servidor es el encargado de comprobar la seguridad de un equipo y el cliente es el responsable de realizar las peticiones. Podríamos decir que el servidor es el motor y el cliente simplemente el entorno gráfico. Se puede utilizar ambas partes del programa en un único ordenador, de forma que la propia PC realice peticiones así mismo de análisis, o ejecutar el servidor en un equipo potente realizando las peticiones desde un ordenador menos preparado. Esto influye en la velocidad a la que se realizan las pruebas, aunque no a los resultados finales.

Nessus utiliza plugins que son pequeños programas (también llamados exploits) que se aprovechan de un fallo en el diseño de los atacantes que están escuchando detrás de los puertos, para conseguir entrar al sistema. El escáner de vulnerabilidad Nessus es el líder mundial en escáneres activos, destacando el descubrimiento de alta velocidad, la revisión de configuración, el activo descubrimiento de datos copiador, sensible y el análisis de vulnerabilidad de su postura de seguridad .Nessus escáner puede ser distribuido en todas partes de una empresa entera, dentro DMZs, y a través de redes físicamente separadas. Note que Nessus 3.x es propietario, mientras Nessus 2.x es la fuente abierta, que el vendedor ha cometido al mantenimiento.

Es la herramienta de evaluación de seguridad "Open Source" de mayor renombre. Nessus es un escáner de seguridad remoto para Linux, BSD, Solaris y Otros Unix. Está basado en plugins, tiene una interfaz basada en GTK, y realiza más de 1200

pruebas de seguridad remotas. Permite generar reportes en HTML, XML, LaTeX, y texto ASCII; también sugiere soluciones para los problemas de seguridad.

Desde hace algún tiempo, se rumoreaba que la siguiente versión de Nessus (para aquellos que no lo conozcan, es un escáner de vulnerabilidades) dejaría de estar bajo la licencia GPL, y pasaría a un modelo cerrado, aunque al menos continúa siendo gratuito. Ahora esta noticia ha sido confirmada por el propio responsable del proyecto (Renaud Deraison), que anuncia la continuación de dos ramas estables:

Nessus 2: GPL, con actualizaciones regulares para soluciones agujeros de seguridad.
Nessus 3: Gratuito, con más funcionalidades.

Las razones para este cambio, parecen ser que la competencia de Nessus se está aprovechando de la libertad de código de éste. Es una gran pérdida para la comunidad open source el perder esta valiosa herramienta.

Soporta IMAP, SMTP y POP3, cuenta con una librería COM de fácil uso.

En operación normal, Nessus comienza escaneando los puertos con nmap o con su propio escaneador de puertos para buscar puertos abiertos y después intentar varios exploits para atacarlo. Las pruebas de vulnerabilidad, disponibles como una larga lista de plugins, son escritos en NASL (Nessus Attack Scripting Language, Lenguaje de Scripting de Ataque Nessus por sus siglas en inglés), un lenguaje scripting optimizado para interacciones personalizadas en redes.

Algunas de las pruebas de vulnerabilidades de Nessus pueden causar que los servicios o sistemas operativos se corrompan y caigan.

Los plugins son el "corazón" de Nessus. Ellos son las pruebas de seguridad, esto significa descubrir una vulnerabilidad determinada. NASL (Nessus Attack Sripting Language) es un leguaje recomendado para escribir pruebas de seguridad.

Existen aproximadamente 20 familias de Plugins: puertas traseras, denegación de servicio, lograr accesos root remotamente. Como ya lo mencionamos, cada plugins reporta información. Diciéndole qué esta incorrecto y que debería hacer usted para corregir el problema.

Operaciones:

NessusClient introduce un nuevo formato de archivo de Nessus para la exportación de exploración y la importación. El formato tiene las ventajas siguientes:

· **XML basado:** Para compatibilidad fácil avanzada y atrasada y puesta en práctica fácil.

· **Autosuficiente:** Un archivo solo contiene de Nessus la lista de objetivos, la política definida por el usuario, así como los resultados de exploración.

· **Seguro:** Las contraseñas no son salvadas en el archivo. En cambio una referencia a una contraseña almacenada en una posición segura sobre el host local es usada.

El proceso para crear el archivo de Nessus que contiene los objetivos, la política y resultados de exploración se debe primero generar la política y salvarlo, luego generar la lista de direcciones objetivo y finalmente, controlar una exploración. Una vez que la exploración es completa, toda la información puede ser salvada en el archivo de Nessus usando el " Salva Como " la opción del menú "de Archivo". Si usted salva a un archivo de Nessus antes generado, la información es añadida a la sección apropiada de aquel archivo.

Usted también puede generar el archivo de Nessus que contiene las direcciones objetivo y/o la política, pero ningunos resultados por salvar la información antes de la carrera de una exploración. Si usted controla una exploración en un tiempo posterior, la información será añadida a la sección "de Informes" del archivo Nessus.

Descarga de Nessus:

Para instalar Nessus debe conectarse primeramente a su página principal: http://www.nessus.org/download/

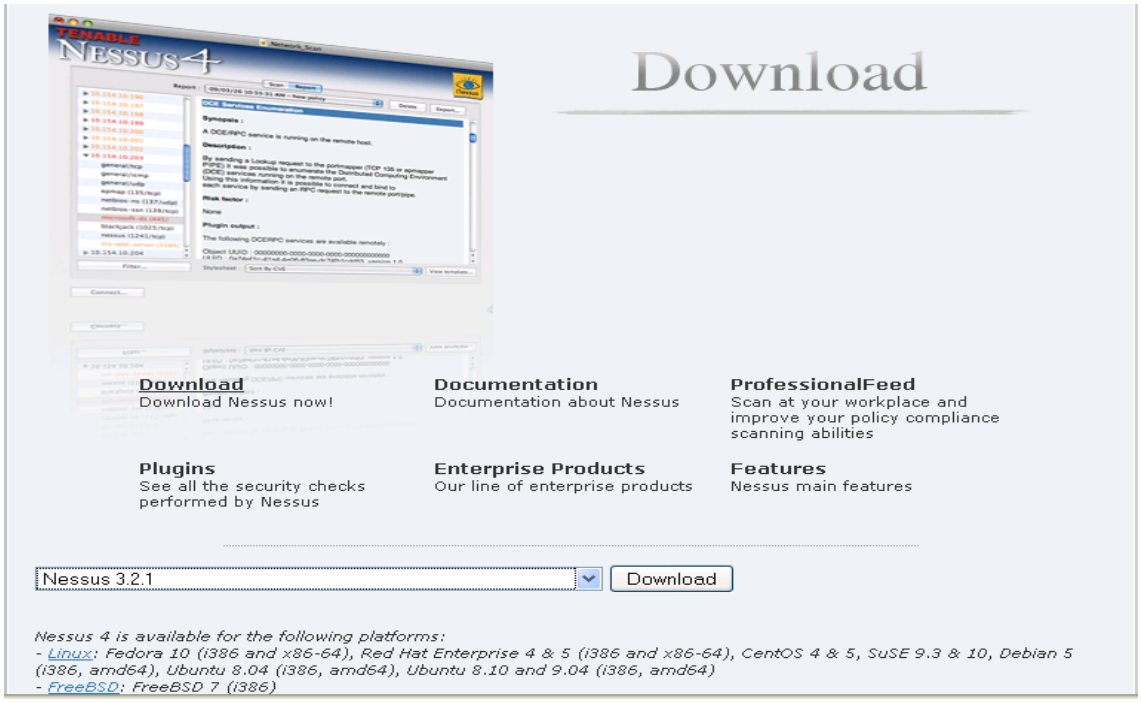


Fig. 1 Descargando Nessus.

Posteriormente escoges el tipo de SO que tengas instalado en tu máquina para de esta forma descargar Nessus, ya que el instalador viene para cada unos de los SO y el

usuario solo debe escoger el que el tenga en su PC. A continuación se muestra lo antes explicado.

You must have an activation code to be able to use Nessus. Please click here to obtain one.

Red Hat ES 5 / CentOS 5	Nessus-3.2.1-es5.i386.rpm	11860 KB
Fedora 8	Nessus-3.2.1-fc8.i386.rpm	11559 KB
Ubuntu 8.04 (32 bits)	Nessus-3.2.1-ubuntu804_i386.deb	13645 KB
Windows XP, 2003, Vista & 2008 (32 bits)	Nessus-3.2.1.1.exe	20570 KB
Debian 4.0 (64 bits)	Nessus-3.2.1-debian4_amd64.deb	13745 KB
Debian 4.0	Nessus-3.2.1-debian4_i386.deb	13687 KB
Red Hat ES 4 / CentOS 4	Nessus-3.2.1-es4.i386.rpm	11009 KB
Fedora 7	Nessus-3.2.1-fc7.i386.rpm	11501 KB
Fedora 9	Nessus-3.2.1-fc9.i386.rpm	11570 KB
SuSE 9.3	Nessus-3.2.1-suse9.3.i586.rpm	8733 KB
SuSE 10	Nessus-3.2.1-suse10.0.i586.rpm	8690 KB
Ubuntu 8.04 (64 bits)	Nessus-3.2.1-ubuntu804_amd64.deb	13744 KB
Solaris 9 & 10 (sparc)	Nessus-3.2.1-solaris-sparc.pkg.gz	10102 KB
Red Hat ES 3 / CentOS 3	Nessus-3.2.1-es3.i386.rpm	10909 KB
FreeBSD 7 (32 bits)	Nessus-3.2.1-fbsd7.tbz	7749 KB
Red Hat ES 5 (64 bits) / CentOS 5	Nessus-3.2.1-es5.x86_64.rpm	12006 KB
Mac OS X (10.4 and 10.5)	Nessus-3.2.1.dmg.gz	10972 KB
Ubuntu 7.10	Nessus-3.2.1-ubuntu710_i386.deb	13677 KB

- GPG Signed MD5s of these packages

Fig. 2 Seleccionado el SO.

Una ves seleccionado el SO de clic sobre el instalador y guarde este en el escritorio, de esta forma se podrá comenzar a instalar a Nessus.

Instalando Nessus:

De doble clic sobre el ejecutable, que debe encontrarse en la carpeta que usted seleccionó para guardarlo. Se mostrara una ventana de la siguiente forma.

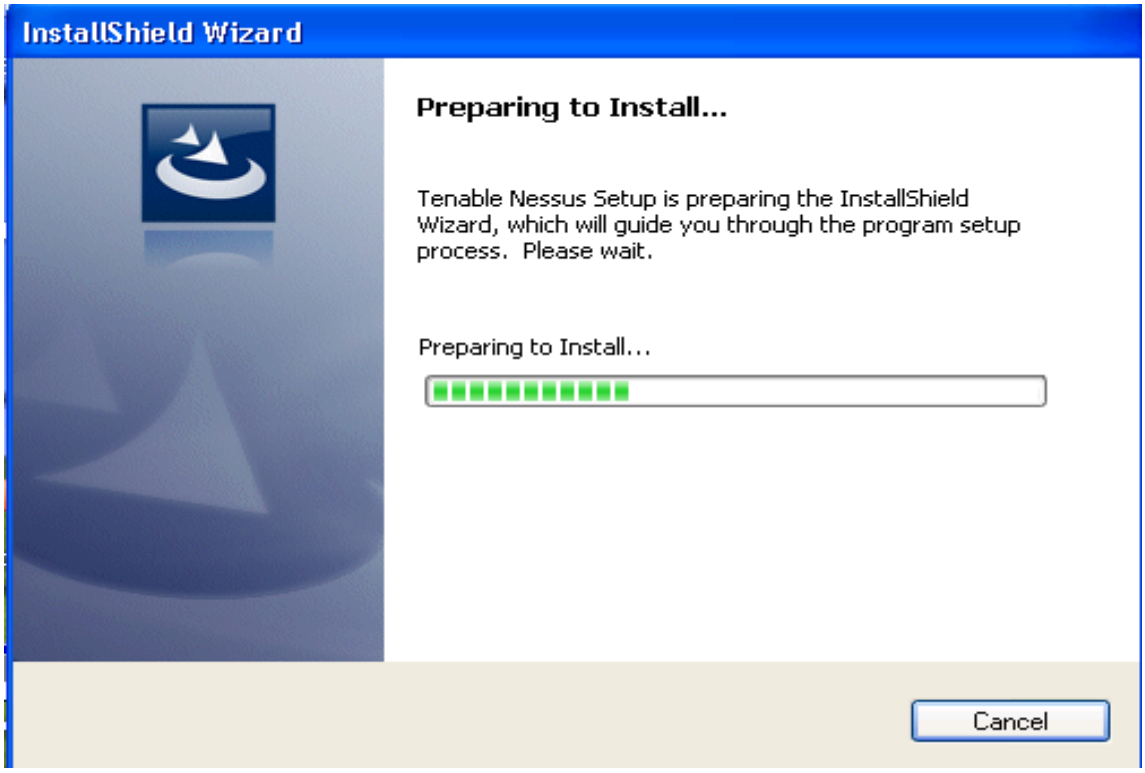


Fig. 3 Instalación de Nessus.

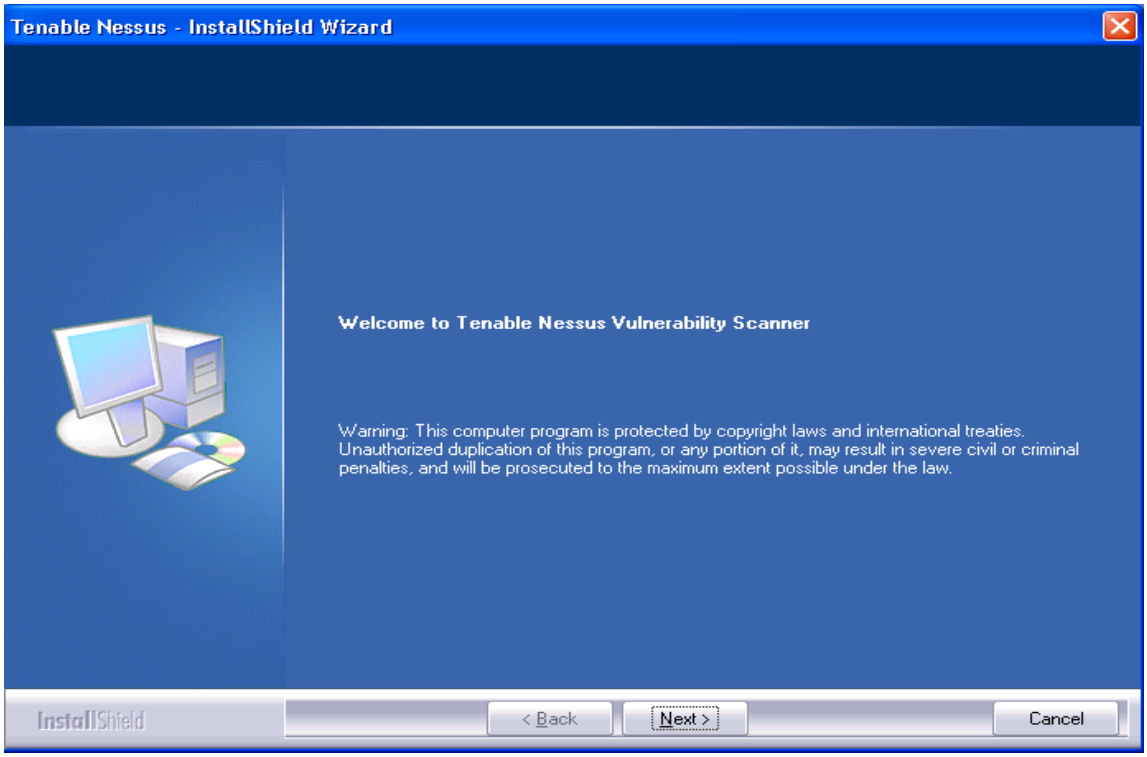


Fig. 4 Instalación de Nessus.

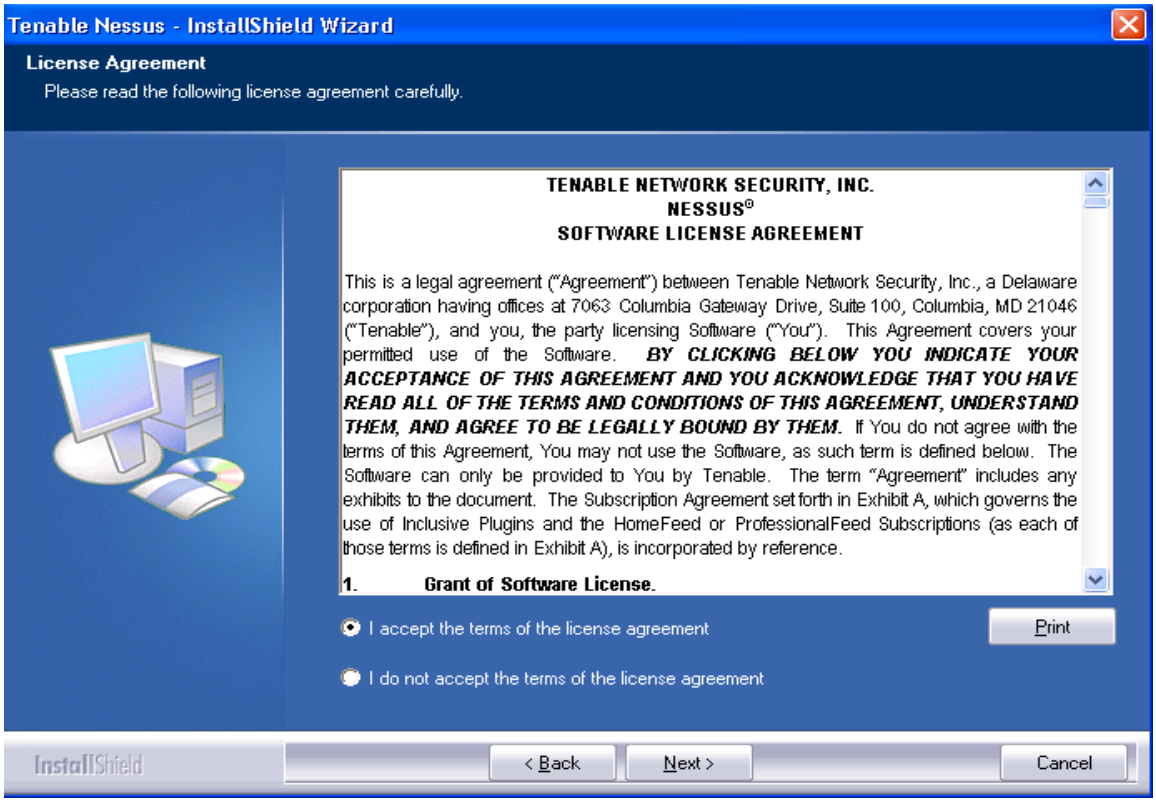


Fig. 5 Instalación de Nessus.

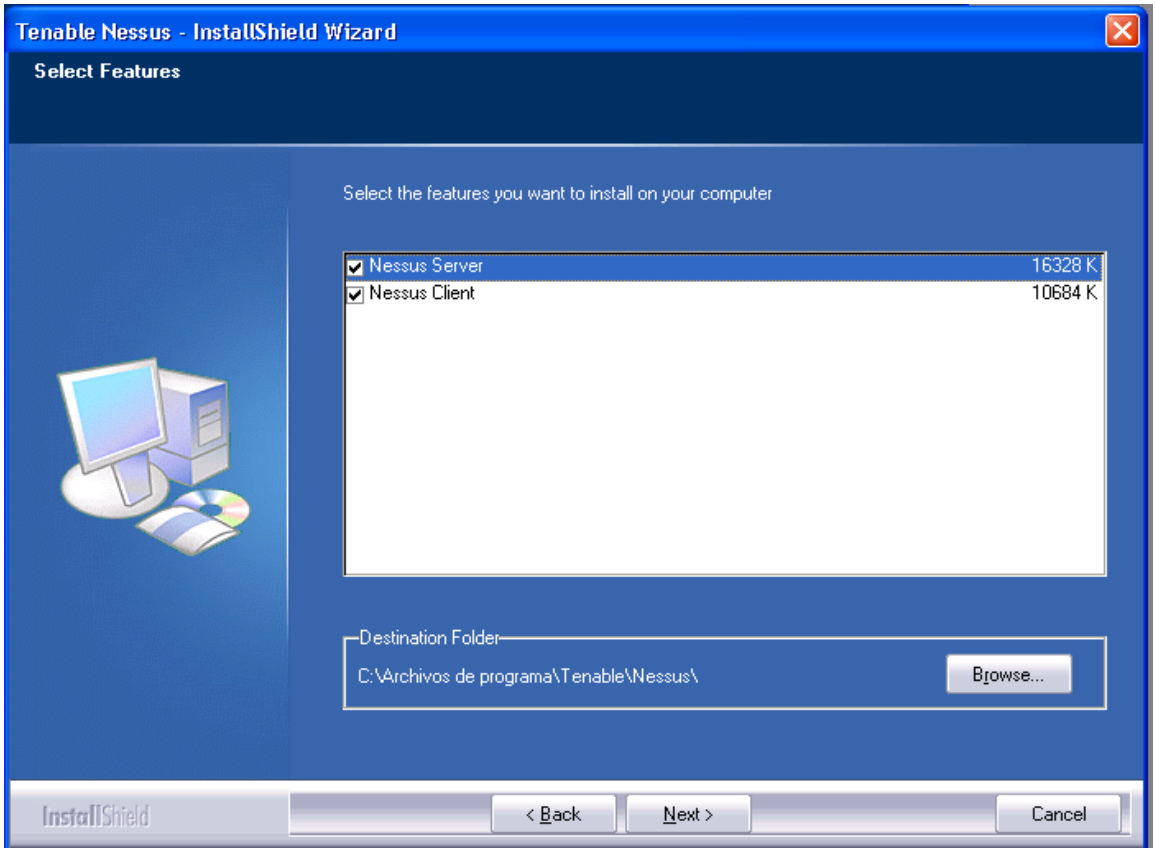


Fig. 6 Instalación de Nessus.

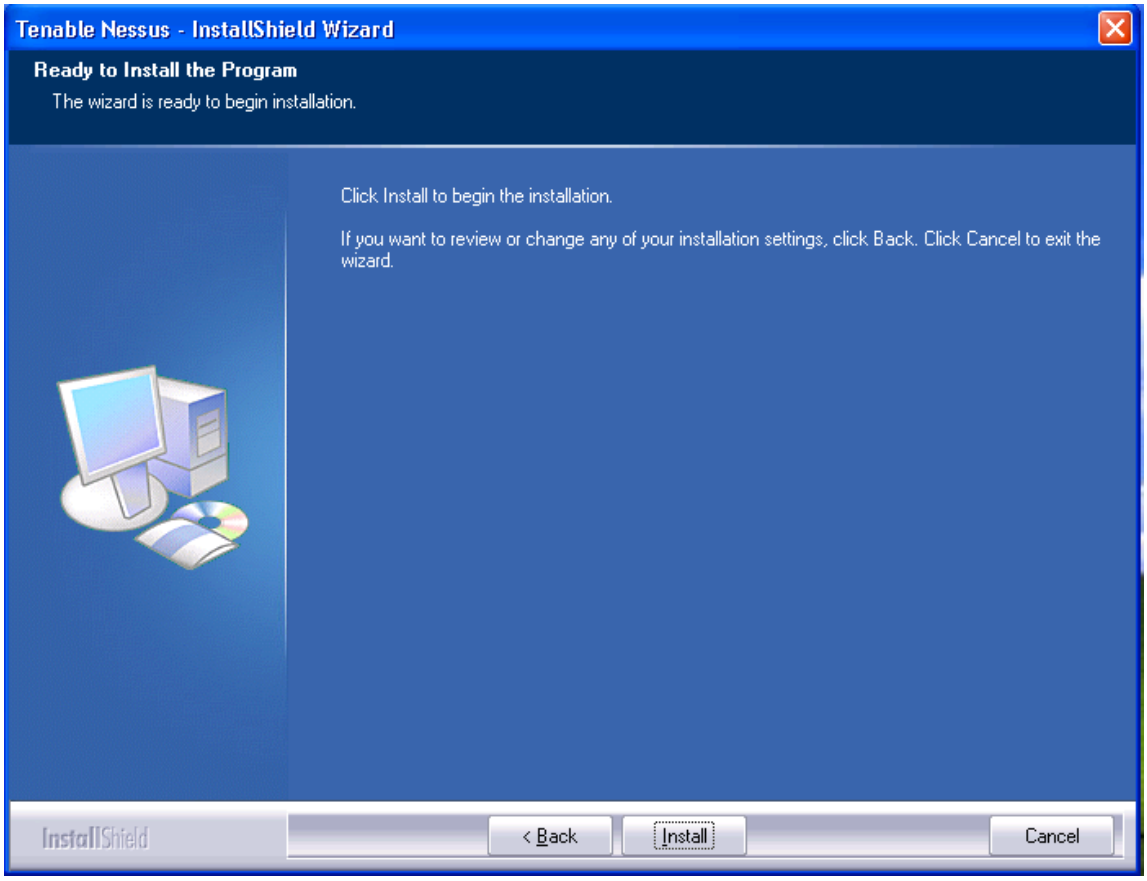


Fig. 7 Instalación de Nessus.

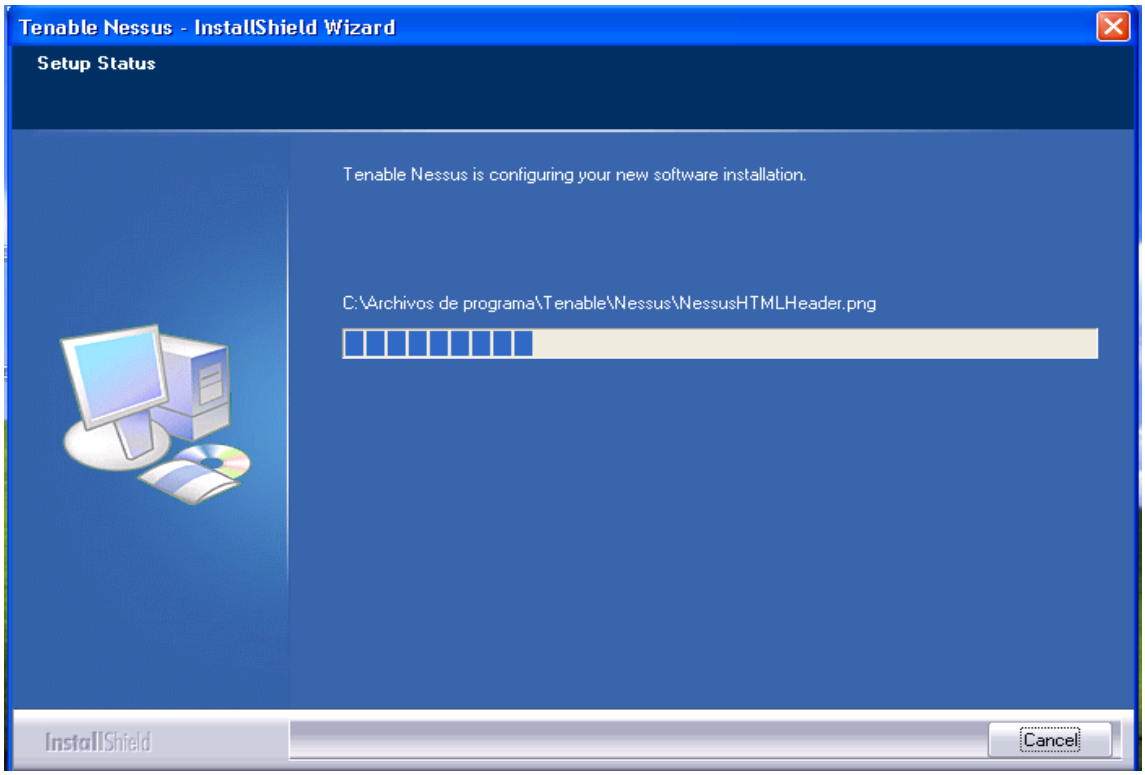


Fig. 8 Instalación de Nessus.

Posterior a esto debe introducir el código de activación el mismo está disponible en la siguiente dirección http://www.nessus.org/plugins/?view=register-info donde debe poner sus datos dentro de ellos poner su dirección de correo para enviarle este código al mismo.

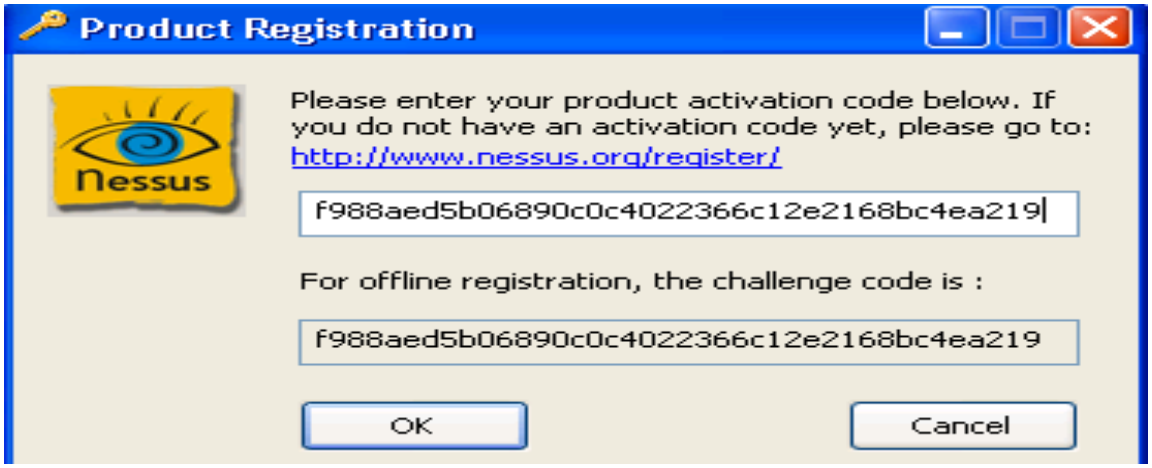


Fig. 9 Instalación de Nessus.

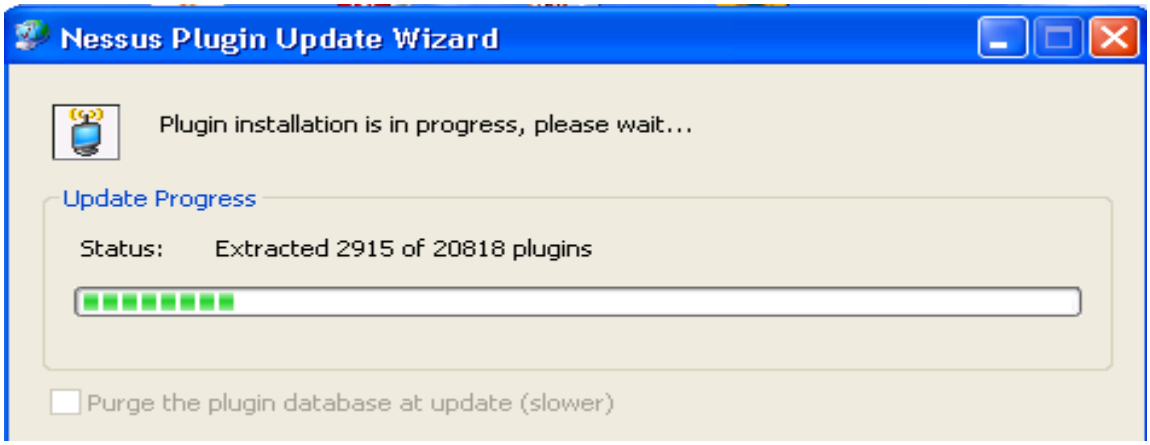


Fig. 10 Instalación de Nessus.

Una vez terminado la instalación le aparecerá el NessusClient y el NessusServer.

NessusServer.

Pulsar sobre el icono del NessusServer sobre el escritorio y aparecerá la siguiente ventana.

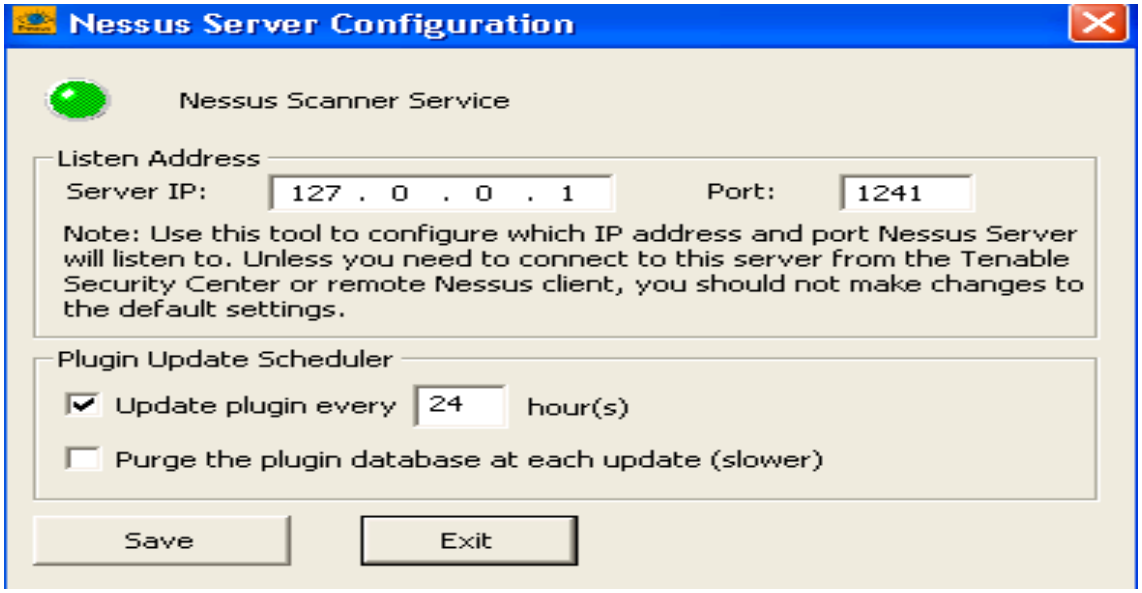


Fig. 11 Ventana principal del NessusServer.

NessusClient

Pulsar sobre el icono " Nessus Cliente " sobre el escritorio.

Se mostrará la siguiente ventana:

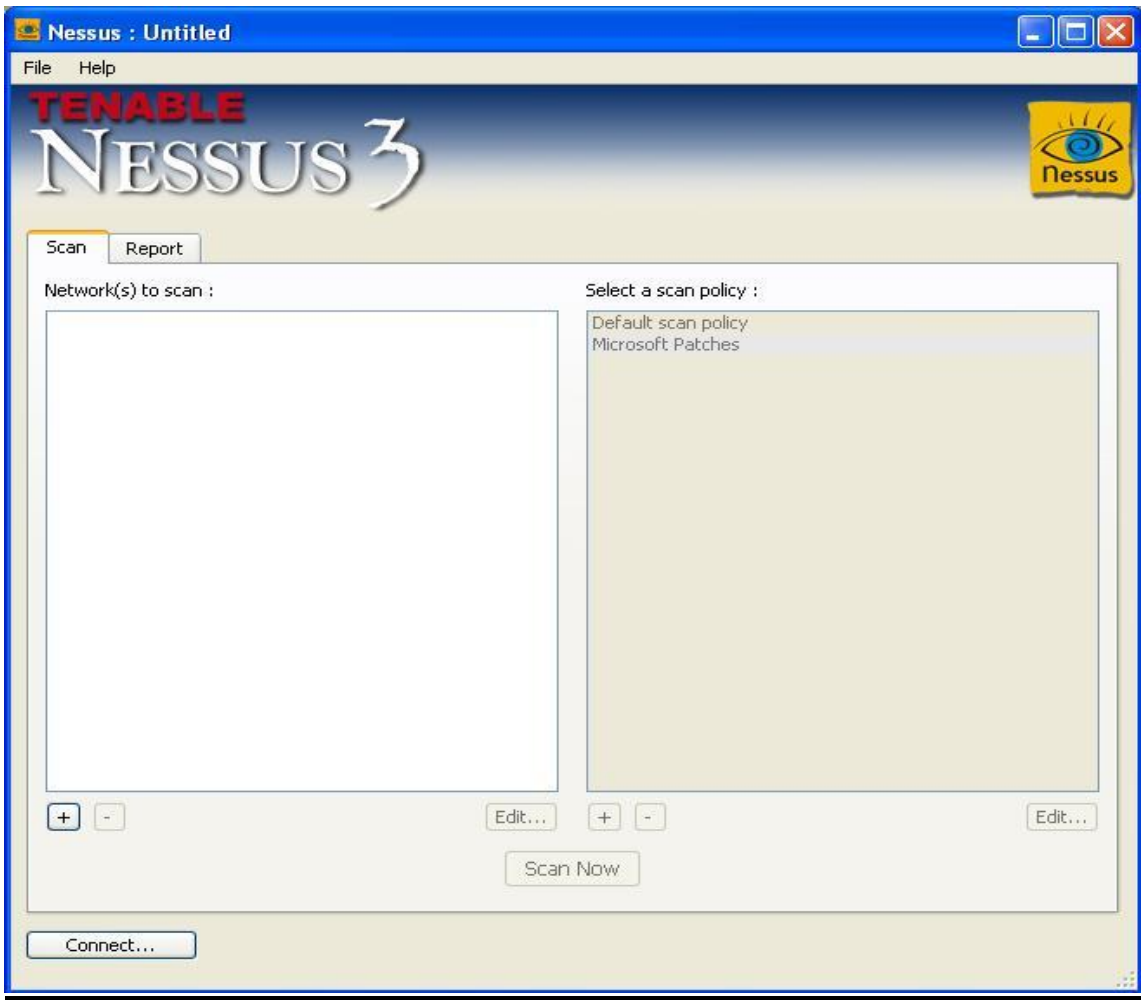


Fig. 12 Pantalla principal.

Usted primero tiene que conectarse al servidor de Nessus pulsando sobre el botón "Conectar" en la esquina inferior izquierda de la pantalla. Esto crea una ventana con los escáneres disponibles de Nessus a los cuales usted puede conectarse como se muestra debajo:

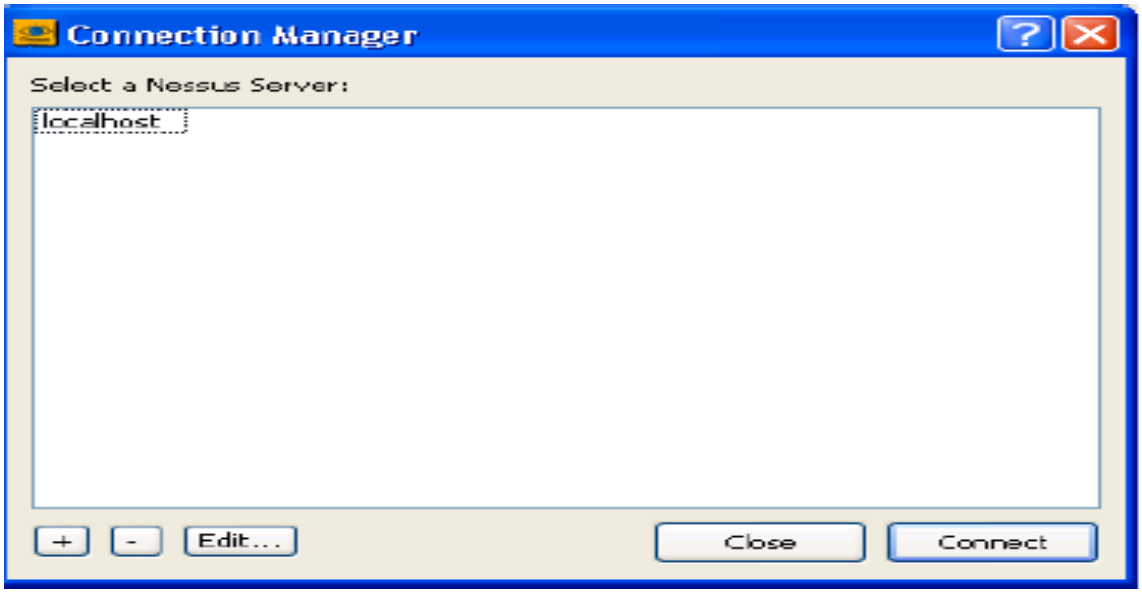


Fig. 13 Conectar con el servidor

Sobre la instalación, el único escáner de Nessus disponible es el que está instalado sobre el host local. La versión de Windows del NessusClient es pre configurada con la conexión del servidor de Nessus y la contraseña. Para Linux (XWindow el Sistema) las versiones del NessusClient que también tienen el servidor de Nessus sobre el host local, usted tendrá que añadir la conexión de Nessus y la información de contraseña que fue generada por el controlador el/opt/nessus/sbin/nessus-adduser durante la instalación del servidor. Haga clic en el botón de Edit si usted tiene que corregir la información de conexión o cambiar la conexión de Nessus o la contraseña. Haga clic sobre el botón "Salvar" para salvar la configuración de conexión.

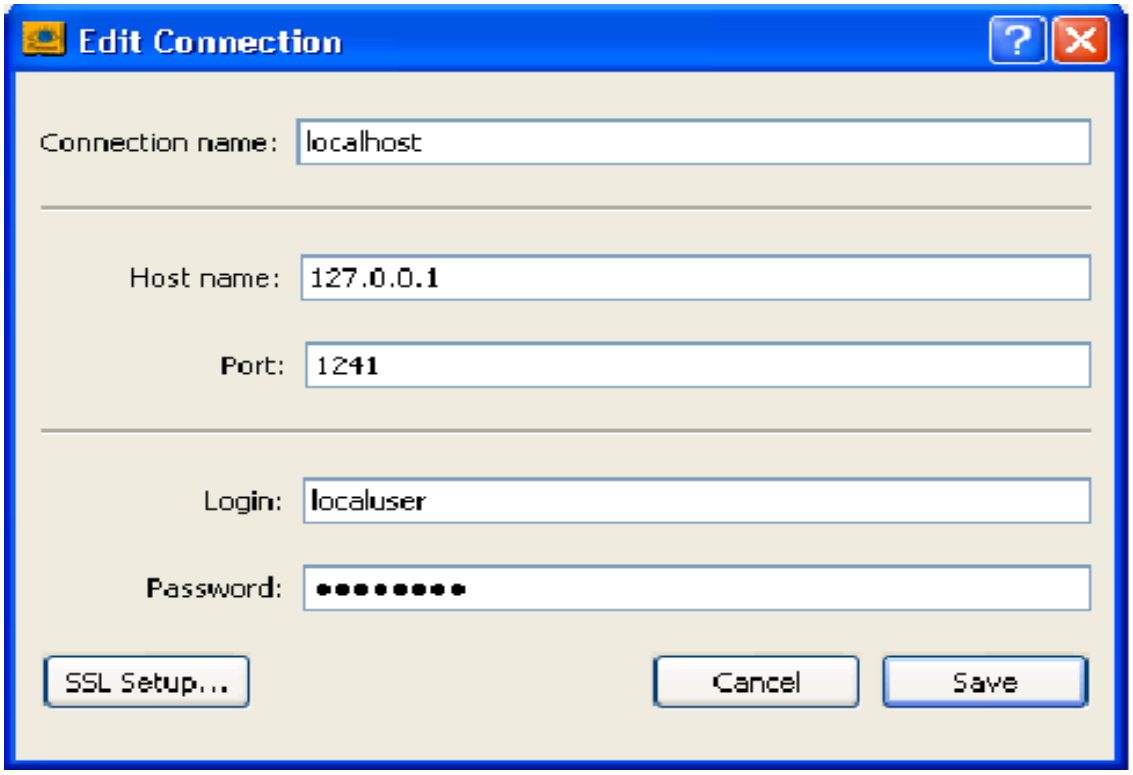


Fig. 14 Instalación del servidor.

Para añadir una conexión a Nessus, pulse sobre el botón del signo Más ("+") y llénese la información apropiada de la forma siguiente:

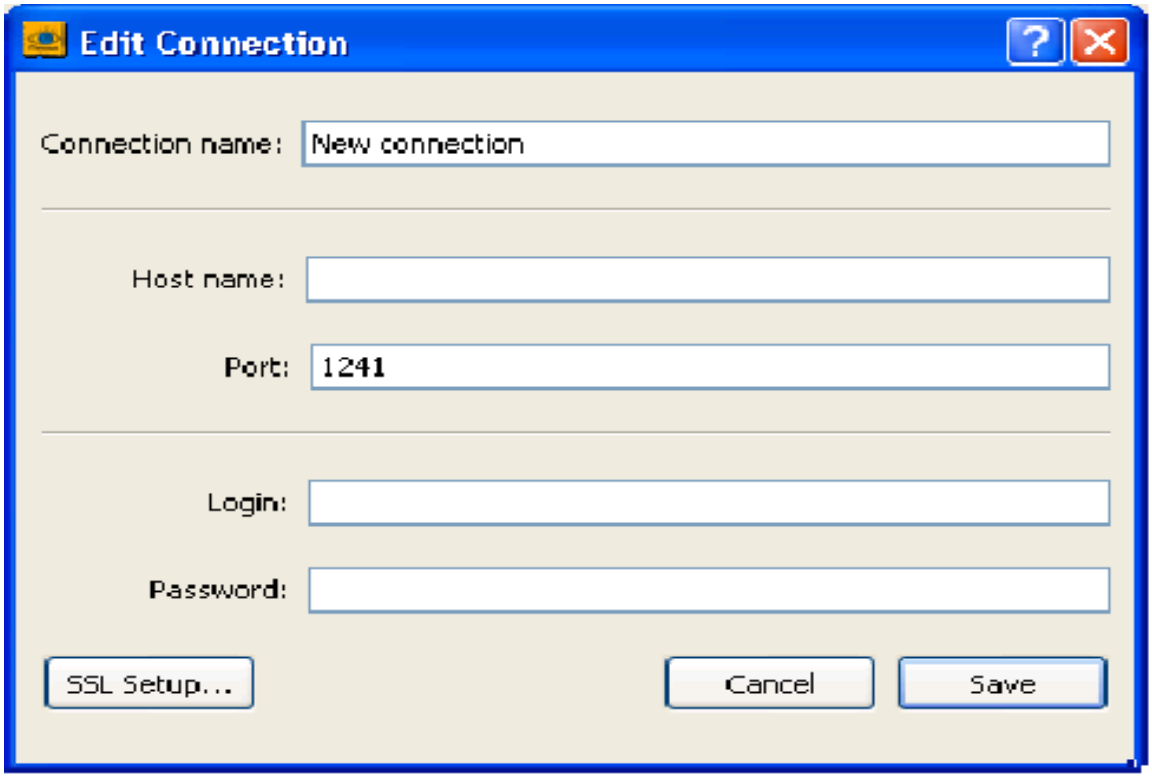


Fig. 15 Añadir una nueva conexión.

Nombre de la conexión (Connection name): En este campo se pone el nombre a la nueva conexión que se realizará.

Nombre del host (Host name): En este campo se pone la dirección del host.

Puerto (Port): En este campo se pone el puerto.

Usuario (Login): En este campo se pone el usuario local.

Contraseña (Password): En este campo se pone la contraseña del usuario local.

Salve la configuración y haga clic sobre el nombre del escáner mostrado para seleccionarlo, luego pulse sobre " Connect". Una ventana aparecerá mostrando lo siguiente:

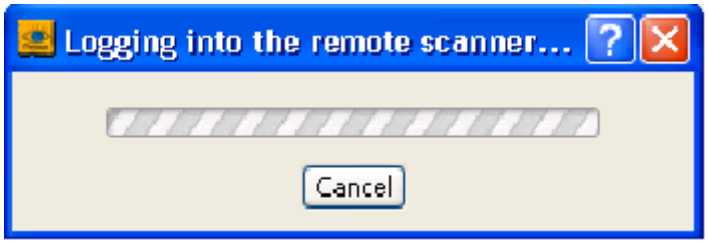


Fig. 16 Salvar la conexión.

Creación de una Política

Una vez que usted se ha unido a un servidor de Nessus, usted puede crear una política de encargo pulsando sobre el signo "+" (Añada la Política) el botón con el título " Selecciona una política de exploración: ". " La Política de Revisión " la ventana será mostrada como sigue:

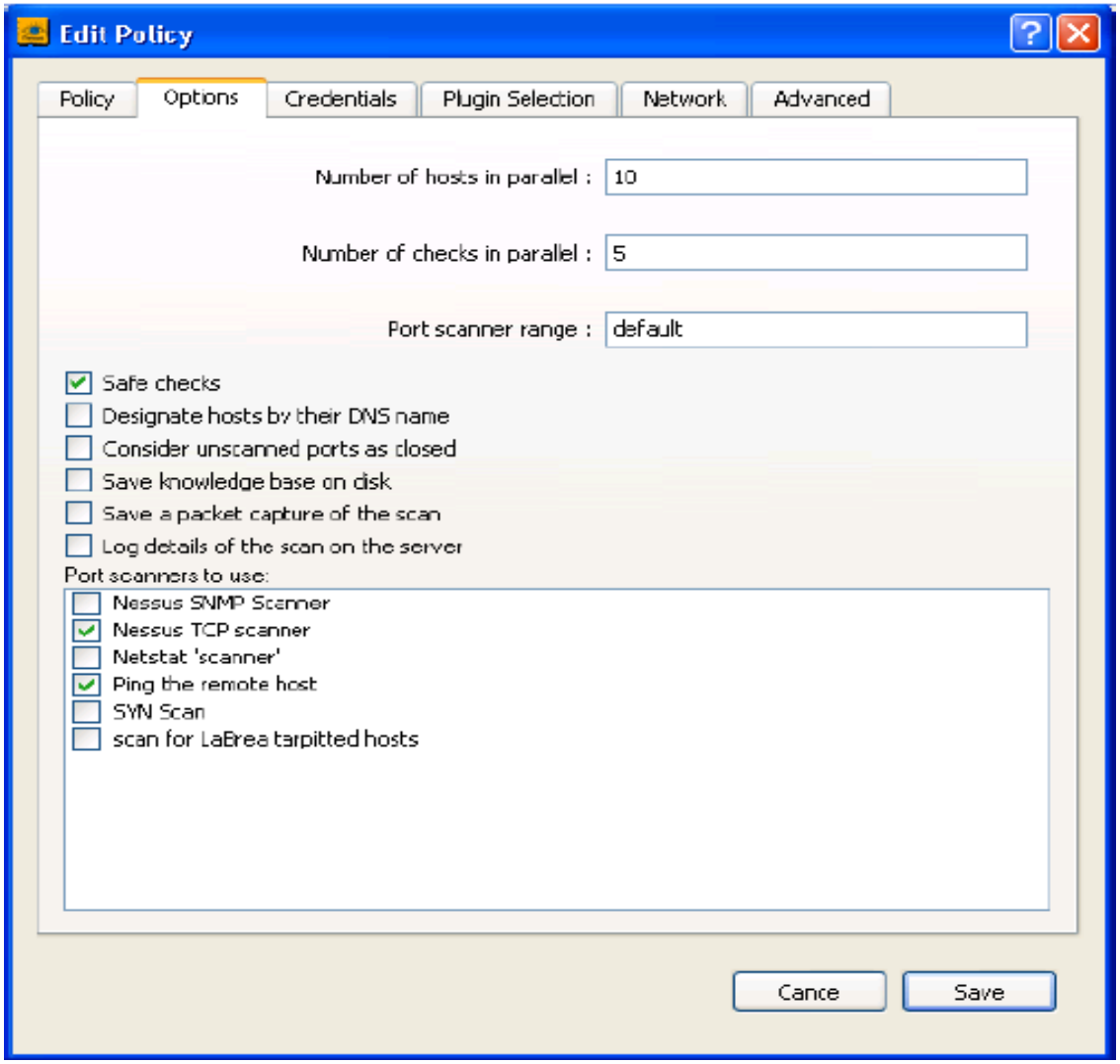


Fig.17 Creación de una política de seguridad.

Note que hay seis etiquetas de configuración: Política, Opciones, Credenciales, Plugins, Red, y Avanzado. Estas etiquetas son descritas debajo.

Para salvar la " Política de Revisión " se oprime en el botón "Salvar", esta debe ser guarda en un archivo de Nessus, si esta política no es salvada en este archivo, no estará disponible después de que usted cierra la sesión del NessusClient.

Política

La etiqueta de política le permite nombrar la política y determinar como la política es salvada.

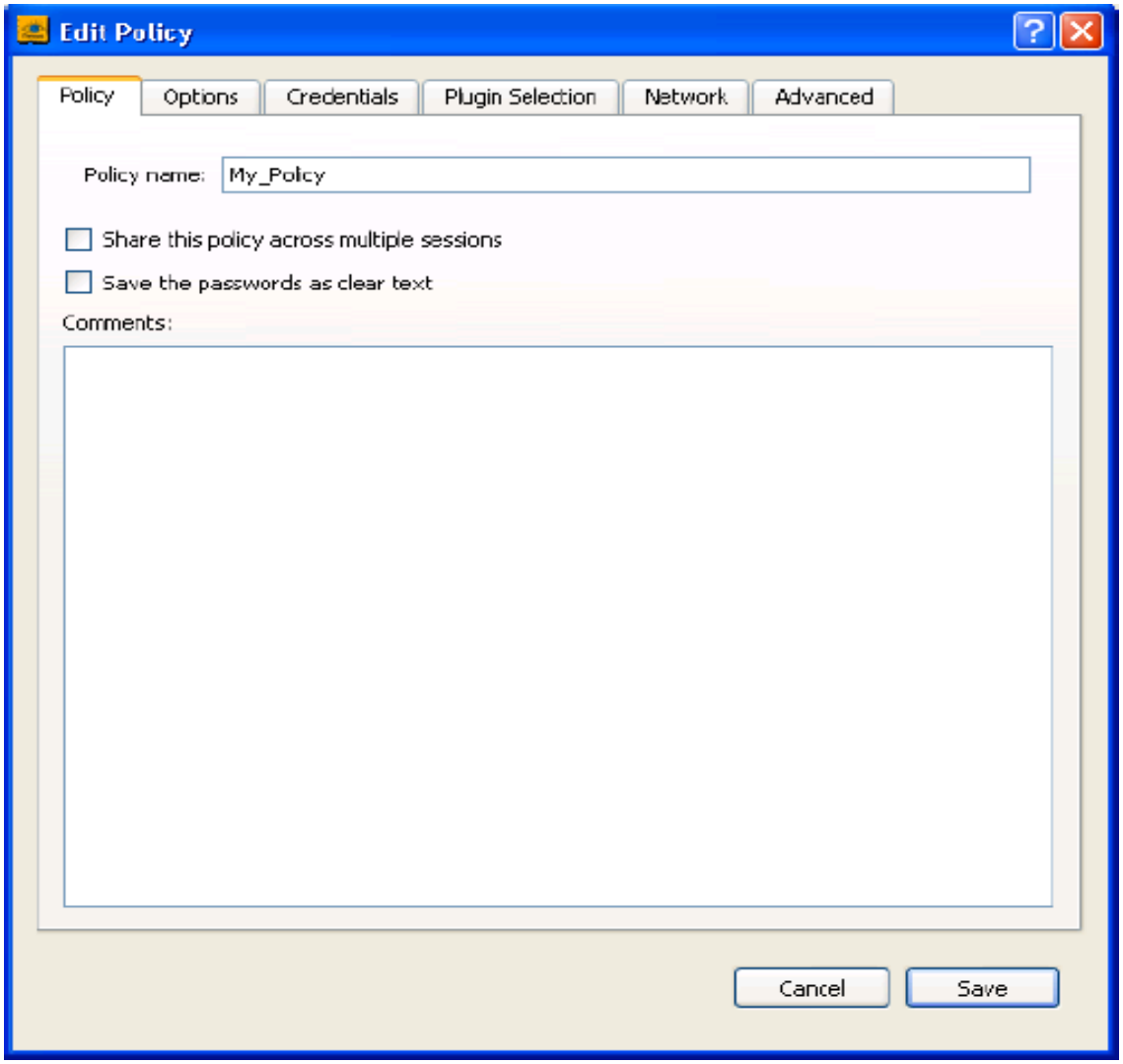


Fig. 18 Creación de una política de seguridad (Política)

Use el campo " el nombre de política " para poner el nombre que será usado en el NessusClient para identificar la política.

La opción de comprobación " **Comparte esta política a través de múltiples sesiones** " se refiere sólo a sesiones Nessus sobre el terminal de trabajo local, y sólo para el usuario corriente.

La utilización de esta opción quiere decir que esta política será mostrada como una de la política de falta, catalogada siempre para que el NessusClient sea comenzado o siempre que la opción " la Nueva Sesión " sea seleccionada del menú principal. Para

este ajuste una política debe ser salvada de la ventana principal NessusClient, a través del menú principal ("Salvar" o " Salvar Como " de la opción de " Archivo").

Por defecto, todas las contraseñas asociadas con la política son cifradas. Si la política es salvada el archivo de .nessus entonces es copiado a NessusClient de forma diferente, todas las contraseñas en la política serán inutilizables antes del segundo escáner de Nessus, que será incapaz de descifrarlos.

Cuando la opción " **Salvar las contraseñas como proporcionan el texto claro**" es seleccionada, la política del archivo de texto claro de .nessus es salvada y todas las contraseñas serán salvadas en el mismo archivo. La política puede ser copiada a un segundo NessusClient y luego modificada para cifrar las contraseñas para la seguridad.

Opciones.

La etiqueta de Opciones le permite poner parámetros globales para el plugins siendo controladas por Nessus.

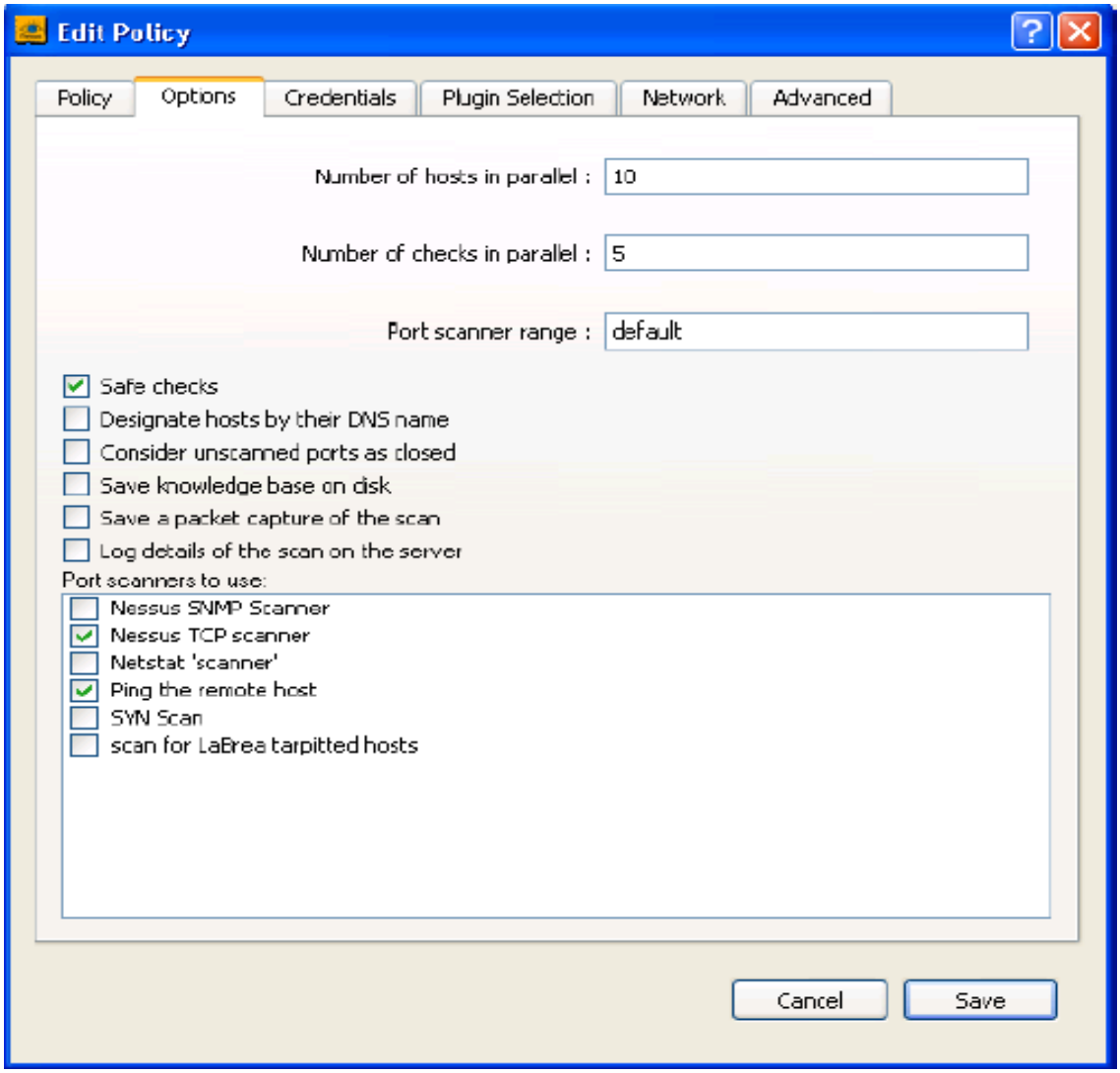


Fig. 19 Creación de una política de seguridad (Opciones).

La tabla siguiente describe las opciones disponibles:

Opciones	Descripción
El número de Host en paralelo(Number of hosts in parallel)	Pone el número máximo de los hosts que serán explorados
El número de facturaciones en paralelo (Number of checks in parallel)	Pone el número máximo de plugins que será controlado sobre cada host simultáneamente. Nessus puede correr sus exploraciones a altas velocidades. Debido a limitaciones de red, en particular sobre WANS, usted puede

	tener que reducir la marcha de las exploraciones para optimizar el funcionamiento de Nessus y evitar el impacto adverso a su red.
El rango de escáner de puerto(Port scanner range)	Específica que puertos explorar. Esta opción es útil para explorar vulnerabilidades particulares sobre puertos específicos. El rango de puerto debe explorar puertos TCP definidos en el archivo de nessus-servicios. Usted puede usar un rango como " 137-139 " y separar puertos individuales o rangos con una coma "137-139, 445,80" que excluye las cotizaciones sobre cada ejemplo.
Salvas de Comprobaciones(Safe checks)	Especifican que los dispositivos que han sido identificados para ser desfavorablemente afectados por la exploración no son explorados. Por ejemplo, una exploración de una impresora puede causar que la impresora que tiene que ser explorada comience de nuevo.
Designe a Host por su nombre de DNS (Designate hosts by their DNS name)	Permite especificar una lista de DNS al activo llamado como " la Red para explorar " sobre la etiqueta de Exploración más bien que una dirección de IP sola o rango de dirección de IP.
Considere puertos inexplorados como cerrado(Consider unscanned ports as closed)	Usado para explorar vulnerabilidades sobre puertos particulares para decir al escáner de Nessus que todos los otros puertos están cerrados.
Salve la base de conocimiento sobre el disco(Save knowledge base on disk)	Esta opción dice al escáner de Nessus salvar la información de exploración a la

	base de conocimiento de servidor de Nessus para el empleo posterior.
Ahorre una captura de paquete de la exploración(Save a packet capture of the scan)	Le permite salvar el tráfico de paquete TCP durante la exploración. Estos registros pueden ser útiles si usted recibe resultados anormales durante exploraciones y sospecha problemas de red. Los archivos pueden ser analizados con una utilidad como PVS de Tenable para Windows.
Los detalles del escáner sobre el servidor(Log details of the scan on the server)	Salvan los detalles de la exploración sobre el servidor de Nessus. El archivo antes de pasar puede ser comprobado para confirmar que plugins en particular fue usado y los host que fueron explorados.
Escáneres de puerto para usar(Port scanners to use)	Esta sección de opciones le permite escoger el modo que usted desea y preguntar sus objetivos de exploración.
El escáner Nessus SNMP(Nessus SNMP scanner)	Esta opción explorará objetivos que buscan una respuesta SNMP. Nessus intentará adivinar los ajustes durante una exploración. Si conocen el ajuste y está configurado bajo el artículo de menú de ajustes SNMP de la Etiqueta Avanzada, esto facilitará plugins que buscan vulnerabilidades SNMP y produce resultados más detallados de auditoria.
El Escáner de Nessus TCP(Nessus TCP Scanner)	Esta opción contrata a Nessus con los objetivos de escanear puertos abiertos TCP. Este escáner es optimizado. La configuración remota para este escáner puede ser puesta bajo el artículo de menú de escáner de Nessus de la

	Etiqueta Avanzada TCP.
'El escáner' de Netstat (Netstat 'scanner')	Esta opción usa netstat para comprobar puertos abiertos. Esto confía en el puerto de netstat que es disponible o en una conexión SSH. Este tipo de exploración es requerida para sistemas A BASE DE UNIX.
El ping de host remote (Ping the remote host)	Esta opción permite el ping de host remotos sobre múltiples puertos para determinar si ellos están vivos.
La Exploración SYN(SYN Scan)	Hace que el escáner envíe un paquete SYN al puerto y espera una respuesta de ACK. Si esto no recibe la respuesta dentro de un rango de tiempo definido, se considerará el puerto cerrado. Esto es en particular útil, explorando por un cortafuego.
La exploración para el host LaBrea tarpitted(Scan for LaBrea tarpitted hosts)	LaBrea tarpits es una forma de un honeypot. Ellos típicamente son desplegados para hacer más lento los escáneres y presentar a falsos host. Con este rasgo se permite que Nessus intente identificar tales sistemas dentro de ciertos parámetros y no explorarlos.

Credenciales.

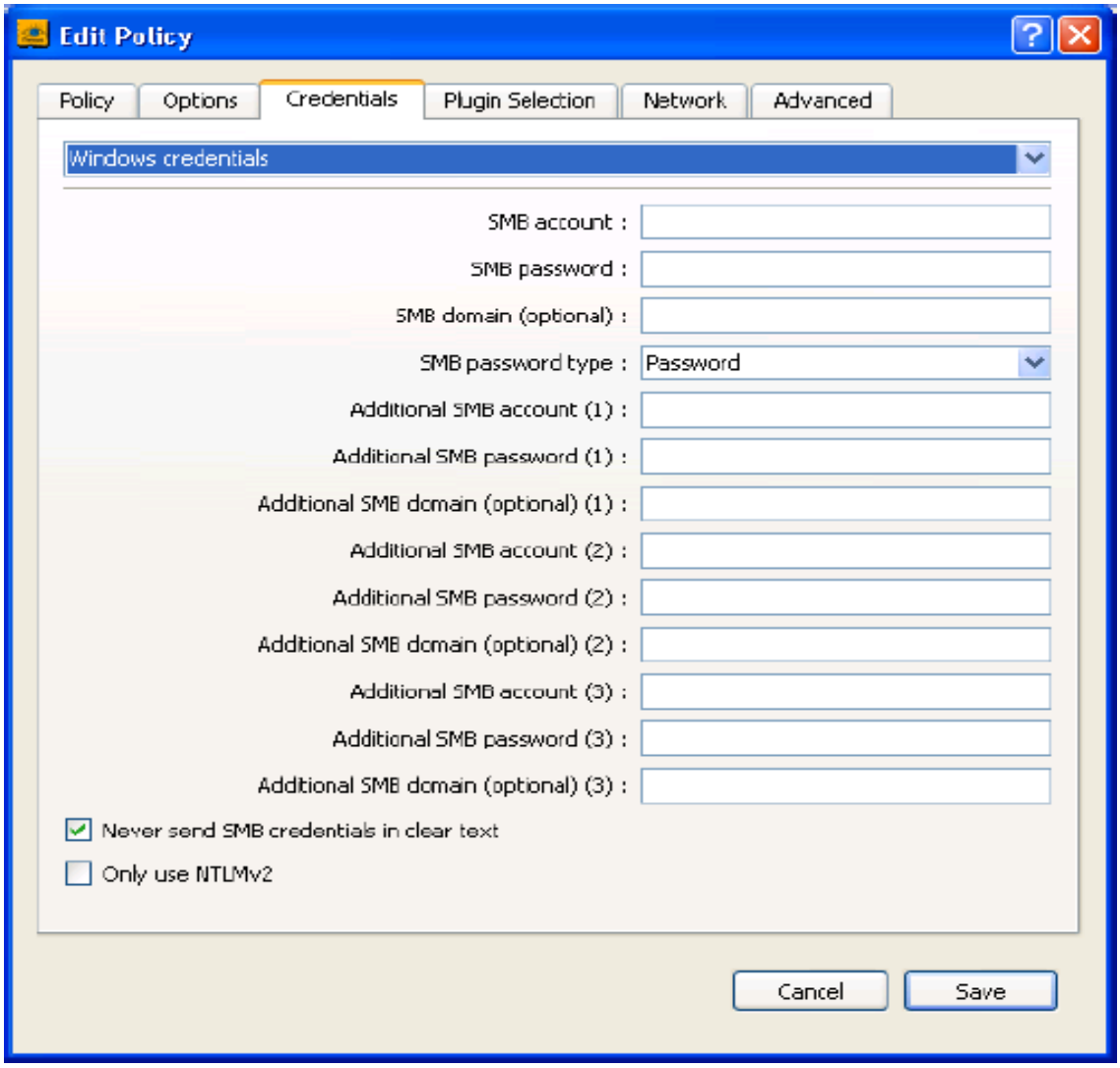


Fig. 20 Creación de una política de seguridad (Credenciales).

Menú desplegable:

Este menú está constituido por las siguientes opciones, las cuales serán expuestas más adelante.

- ✓ **Windows Credentials.**
- ✓ **SSH Setings.**
- ✓ **Oracle Setings.**
- ✓ **Kerberos Configuration.**
- ✓ **Cleartext Protocols Settings.**

Windows Credentials

Bloque de Mensaje de Servidor (SMB): Es un archivo que comparte el protocolo que permite a ordenadores compartir la información transparentemente a través de la red. “Las credenciales de Windows " el artículo de menú desplegable tienen ajustes para proveer a Nessus de la información como el nombre de cuenta de SMB, la contraseña, y el nombre de dominio. El suministro de esta información le permitirá a Nessus encontrar la información local de un host de ventanas remoto. Por ejemplo, usando cartas credenciales permite a Nessus determinar que parches de seguridad importantes han sido aplicados. El personal de seguridad experto, debería modificar otros parámetros SMB de ajustes de falta.

Si la cuenta de un mantenimiento SMB es creado con privilegios de administrador limitados, Nessus fácilmente puede explorar múltiples dominios.

Se recomienda que los administradores de red piensen crear cuentas de dominio específicas para facilitar pruebas. Nessus incluye una variedad de comprobaciones de seguridad para el Windows NT, 2000, el Servidor 2003, y XP que es más exacto si proporcionan una cuenta de dominio.

Nessus realmente intenta con varias facturaciones en la mayor parte de los casos si no proporcionan ninguna cuenta.

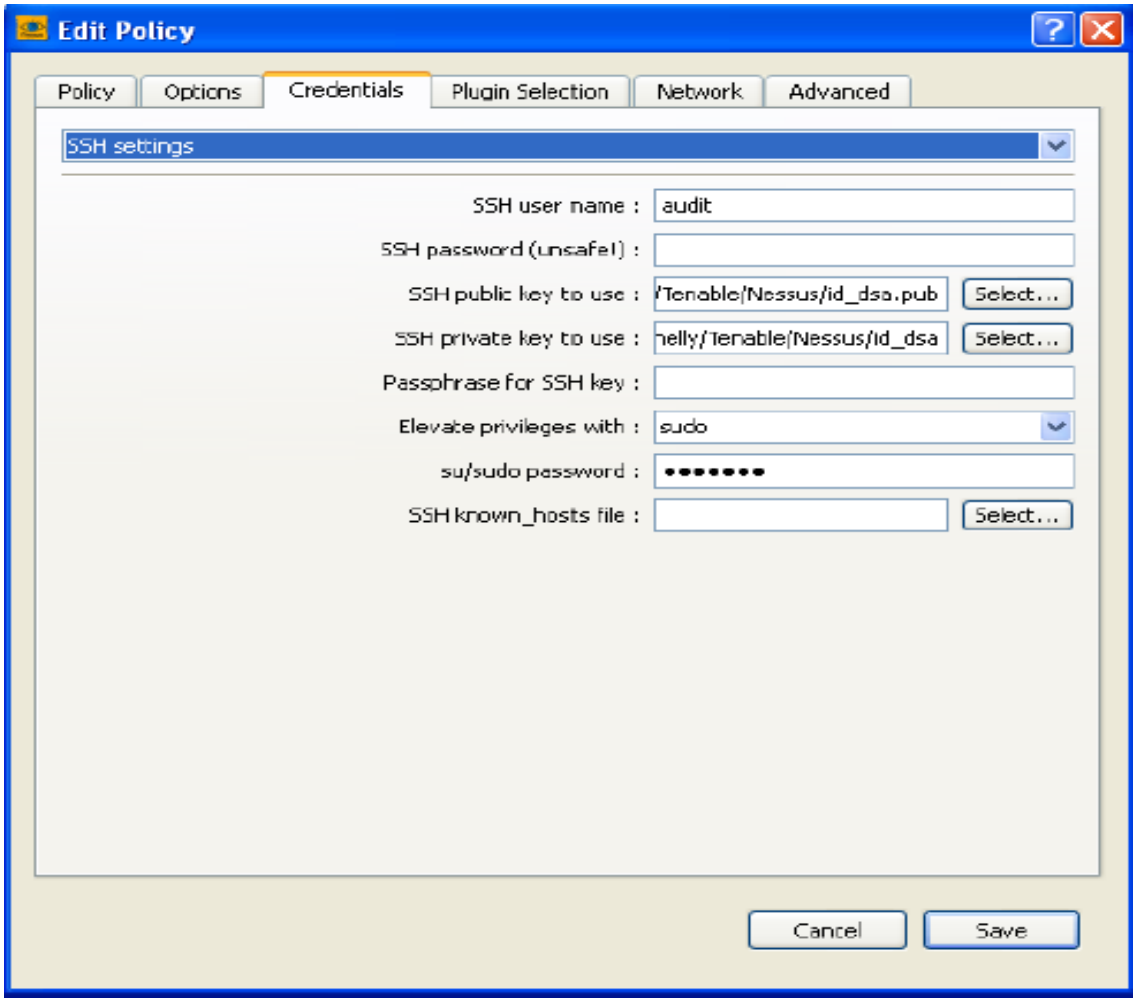


Fig. 21 Creación de una política de seguridad (Credenciales)

SSH Setings

Los usuarios pueden seleccionar " SSH Setings " del menú desplegable y entrar en Credenciales para explorar sistemas de UNIX. Estas Credenciales son usadas para obtener la información local de usuarios remotos que pueden seleccionar "SSH ajustes" del menú desplegable y entrar en Credenciales para explorar sistemas de UNIX. Estas Credenciales son usadas para obtener la información local del host remoto.

Hay un campo para entrar en el nombre de usuario SSH, para la cuenta que realizará las comprobaciones sobre el sistema de UNIX, con la contraseña SSH o con el par SSH clave pública y clave privada. Hay también un campo para entrar en el Passphrase para la llave SSH, si lo requieren.

Las exploraciones de credenciales más eficaces son cuando las Credenciales suministradas tienen privilegios "de raíz". Para usar este rango, la cuenta de usuario para ser usado para la exploración debe tener llaves SSH establecidas para ser usadas en la conjunción con el "su" o la contraseña "sudo".

Una captura de pantalla de ejemplo para usar "sudo" en la conjunción con llaves SSH es cuando, la cuenta de usuario es de " auditoria", que ha sido añadido al archivo/etc/sudoers sobre el sistema para ser explorado. La contraseña proporcionada es la contraseña para la cuenta de " auditoria", no la contraseña raíz. Las llaves SSH están en correspondencia con llaves generadas para la cuenta de " auditoria":

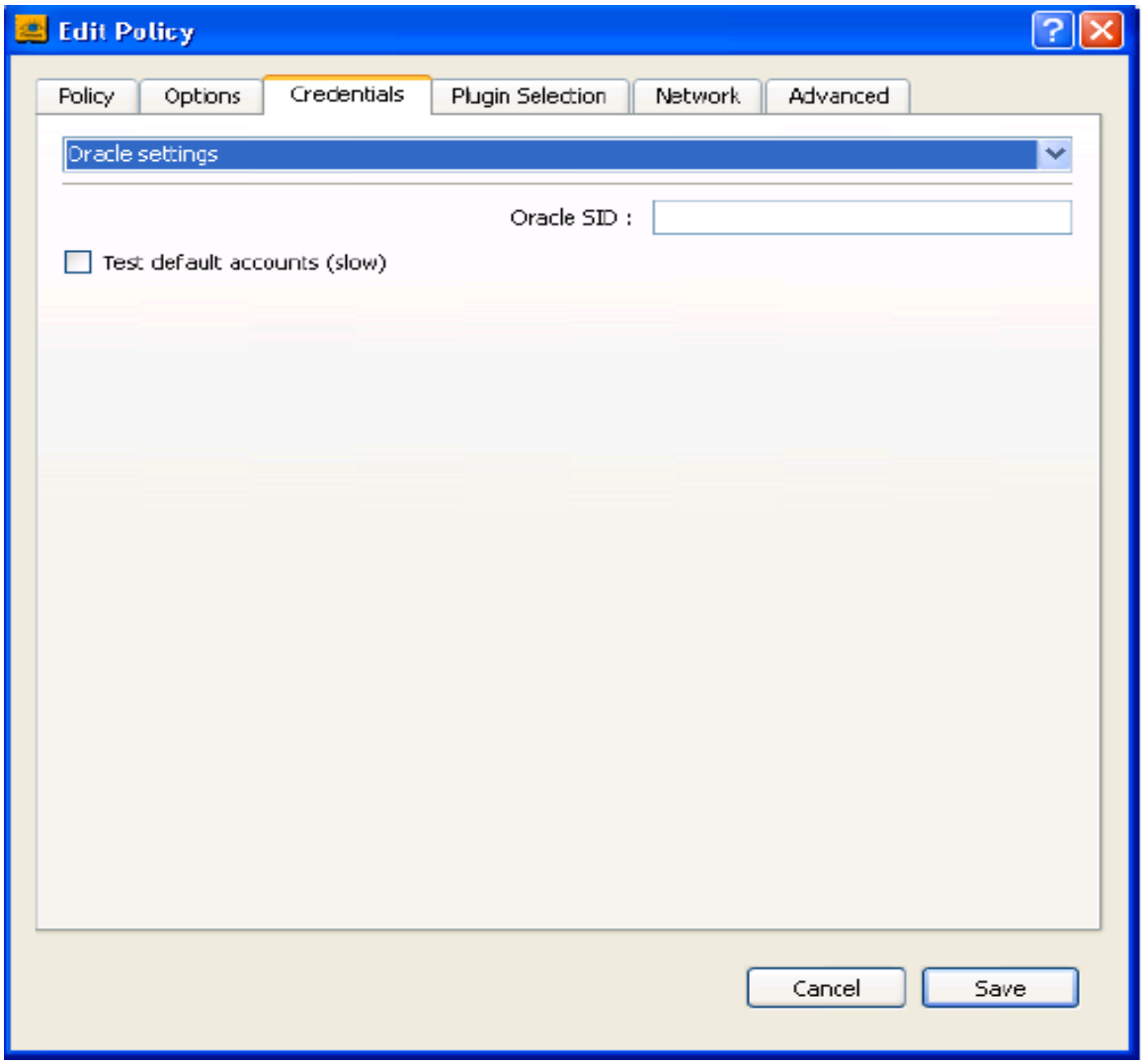


Fig. 22 Creación de una política de seguridad (Credenciales)

Oracle settings

La etiqueta de Credenciales también proporciona una opción en el menú desplegable para configurar " ajustes de Oráculo ", así como una opción para " la configuración de Kerberos".

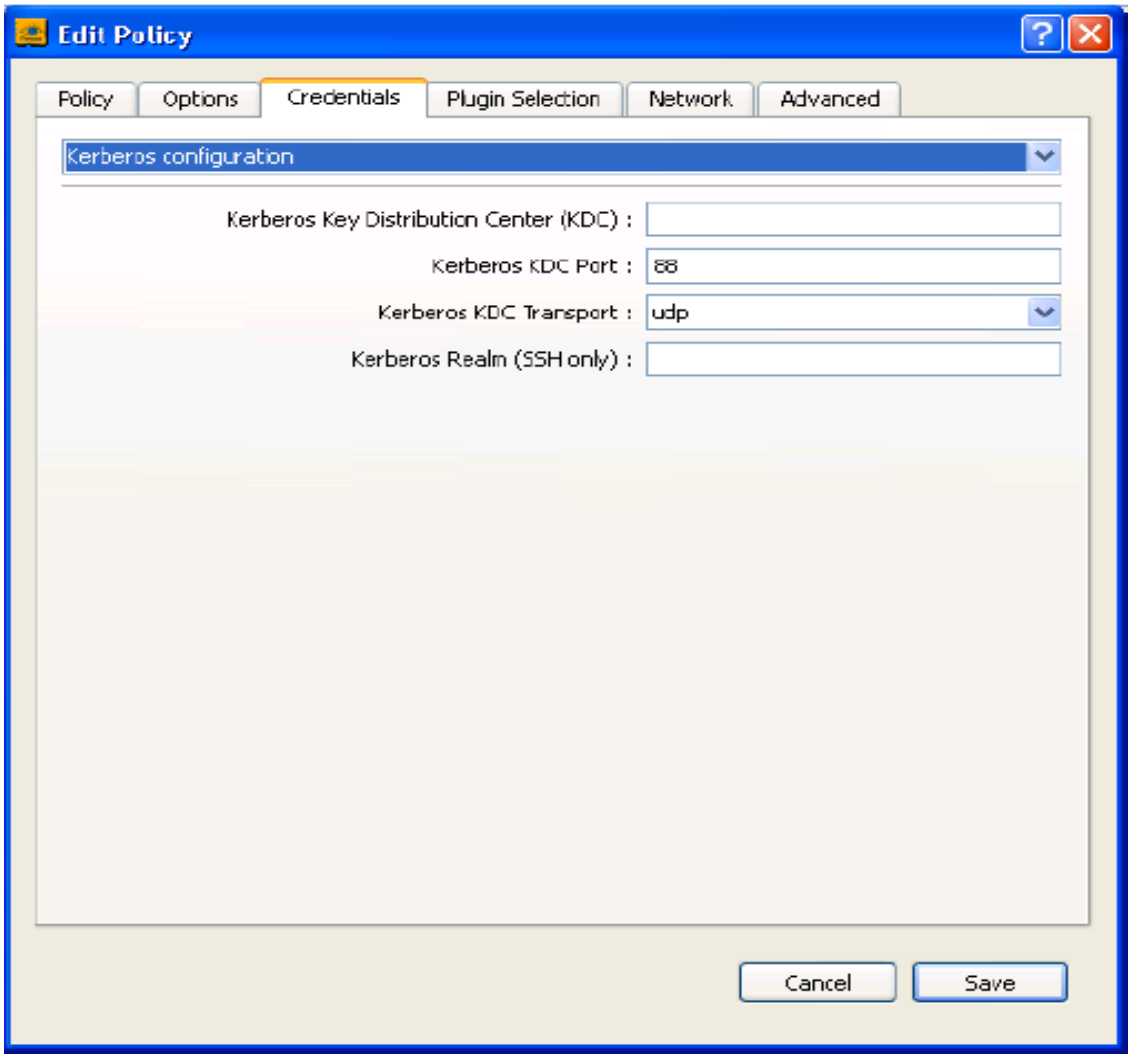


Fig. 23 Creación de una política de seguridad (Credenciales)

Finalmente, si un método seguro para realizar comprobaciones de credenciales no está disponible, los usuarios pueden forzar a Nessus para tratar de realizar comprobaciones sobre protocolos inseguros configurando el artículo de menú desplegable " Cleartext para ajustes de protocolo". Los protocolos cleartext apoyados para esta opción son telnet, rsh, y rexec.

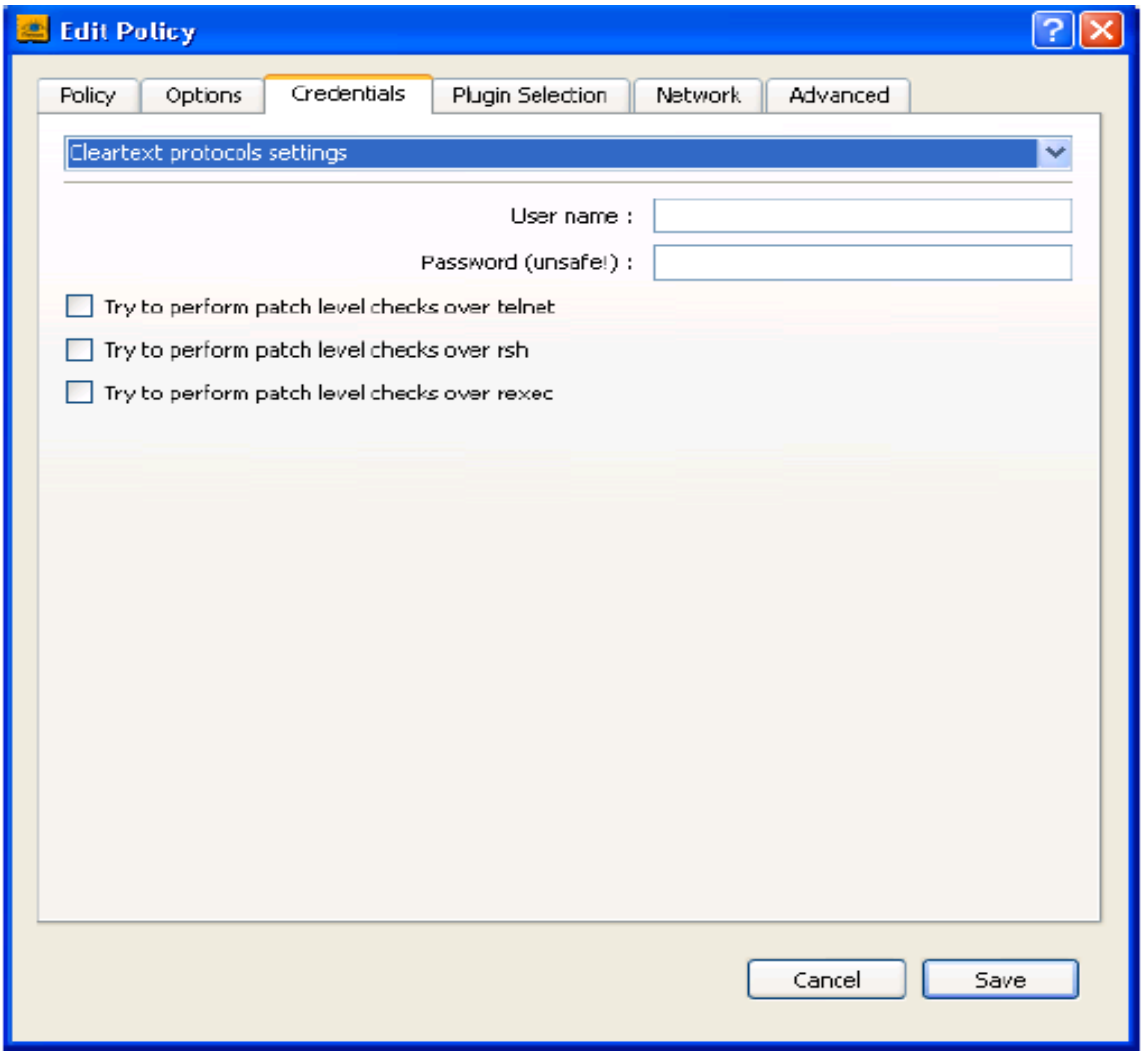


Fig. 24 Creación de una política de seguridad (Credenciales)

Seleccionar los plugins.

La etiqueta de Selección de Plugins permite al usuario escoger comprobaciones de seguridad específicas por "la familia" o comprobaciones individuales.

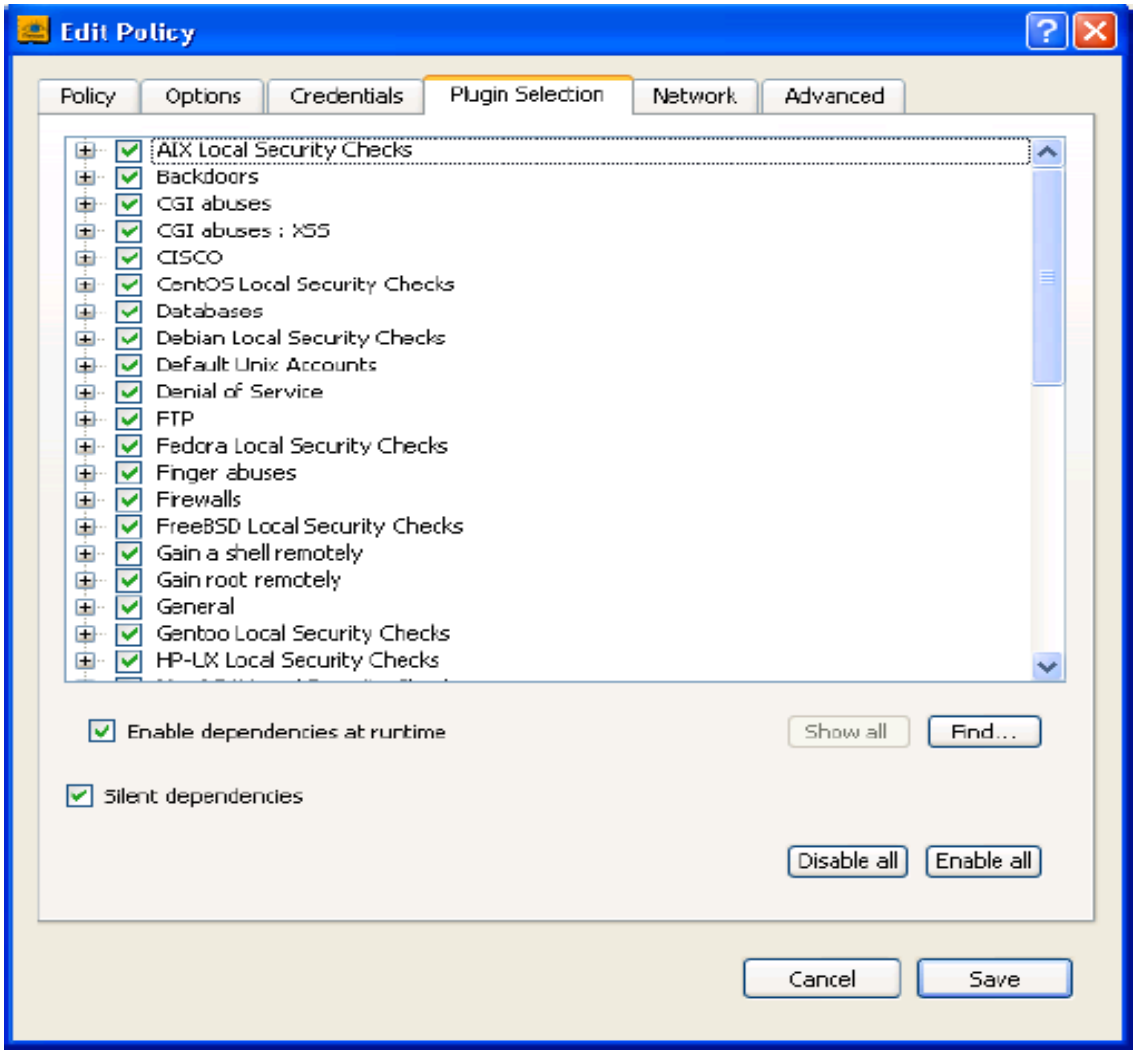


Fig. 25 Creación de una política de seguridad (Plugins)

Seleccionando plugins específico, Nessus mostrará una lista de menú de todas las familias disponibles y plugins individual que comprende aquella familia. El clic sobre el signo "+" es para ampliar la familia de plugins y ver sus plugins. El clic sobre el signo "-" es para destruir la familia de plugins y ocultar el plugins.

Hay un botón al lado de cada familia de plugins para indicar como aquella familia de plugins es usada por Nessus. Por ejemplo, si el botón al lado del artículo de la familia de plugins muestra una comprobación, entonces permite a la familia y todo sus plugins completamente. Si el botón está vacío, entonces aquella familia, así como todo el plugins dentro de ella, es inutilizado. A continuación se muestran los plugins de Nessus.

- ✓ **AIX Comprobaciones de Seguridad Locales.**

El host remoto omite AIX PTF U823830 que es relacionado con la seguridad del paquete bos.aixpert.cmds .Controlado ' suma-x-a RqType=Security ' sobre el sistema remoto .El factor de riesgo es alto.

- ✓ **CentOS Comprobaciones de Seguridad Locales.**

El host remoto omite una actualización de seguridad. El sistema remoto CentOS omite una actualización de seguridad que ha sido documentada. Además mejora los paquetes más recientes que están por hacer. El factor de riesgo es alto.

- ✓ **CGI abuses.**

El servidor web remoto contiene un uso PHP que es afectado por una vulnerabilidad de descubrimiento de la información. El filtro de TeX incluido con la versión instalada de Moodle puede ser abusado para revelar el contenido de archivos sobre el host remoto, sujeto a los privilegios bajo los cuales el servidor web funciona. El factor de riesgo es bajo.

- ✓ **CGI abuses: XSS.**

El servidor web remoto contiene una escritura que es afectada por múltiples sitios. La versión de esta escritura encontrada sobre el host remoto falla en esterilizar la entrada suministrada por un usuario a su 'Base de datos', 'Usuario', y parámetros 'de Contraseña' antes de la utilización de ello para generar el contenido dinámico. Un atacante remoto inautenticado puede ser capaz de inyectar el código HTML arbitrario o el código de escritura en el navegador de un usuario para ser ejecutado dentro del contexto de seguridad del sitio afectado. El factor de riesgo es medio.

- ✓ **CISCO.**

CiscoWorks Interconectan a una red la versión 2.6 del Monitor de Funcionamiento (IPM) para el Solaris y los sistemas operativos Windows de Microsoft contienen una vulnerabilidad que permite a usuarios remotos, inautenticado ejecutar órdenes arbitrarias. CiscoWorks IPM calibra el tiempo de respuesta de red y la disponibilidad. El factor de riesgo es crítico.

- ✓ **Databases.**

La versión de MySQL 6.0 instalado sobre el host remoto es afectada según una negación de vulnerabilidad de servicio. Expresamente, un usuario autenticado puede causar un fracaso que conduce a un choque de servidor .Mejora la versión 6.0.10 del Servidor de Comunidad MySQL. El factor de riesgo es medio.

- ✓ **Falta de Cuentas Unix.**

Una cuenta sobre el host remoto usa una contraseña conocida. La cuenta sobre el host remoto tiene la contraseña 'toor'. Una solución es cambiar la contraseña para esta cuenta o incapacítela. El factor de riesgo es crítico.

- ✓ **Negación de Servicio.**

El servidor remoto LDAP es afectado según una negación de vulnerabilidad de servicio. La versión instalada es más vieja que 6.3.1, y el servidor por poderes incluido con tales versiones según se informa es afectado según una negación de vulnerabilidad de servicio. El factor de riesgo es medio.

- ✓ **DNS.**

El servidor remoto DNS podría estar acostumbrado a una negación distribuida de ataque de servicio. El servidor remoto DNS contesta a cualquier petición. Es posible preguntar a los servidores y conseguir una respuesta que es más grande que la petición original. Un atacante remoto puede lanzar una negación de ataque de servicio contra un host de tercero que usa al servidor remoto DNS. Una solución puede ser restringir el acceso a su servidor DNS de la red pública o configurarlo de nuevo para rechazar tales preguntas. No tiene ningún factor de riesgo.

- ✓ **Comprobaciones de Seguridad Locales.**

Este plugins proporciona la estructura de datos que se maneja para C, envolturas de transportabilidad, e interfaces para tal funcionalidad de tiempo de ejecución como un lazo de acontecimiento, hilos, la carga dinámica, y un sistema de objeto. Este paquete proporciona la versión 2 de FÁCIL. El factor de riesgo es alto.

- ✓ **Finger abuses.**

Nessus fue capaz de enviar una petición demasiado larga. Este defecto es probablemente un desbordamiento de buffer y podría ser explotable para controlar el

código arbitrario contra esta máquina. Una solución podría ser incapacitarla aplicando los últimos buffer o un software más seguro. El factor de riesgo es alto.

- ✓ **Firewalls.**

La dirección remota de IP parece unirse a diferentes hosts mediante NAT.El NAT es una tecnología que deja a múltiples servicios públicos de oferta de ordenadores sobre diferentes puertos por medio de la misma dirección de IP.Basado en resultados que toman las huellas digitales, parecen diferentes sistemas operativos que escuchan sobre diferentes puertos remotos. Note que este comportamiento también puede indicar la presencia de un poder de interceptación, una carga balanceada o un tráfico .Una solución puede ser asegurarse que este sistema tenga la autorización de su política de seguridad. No tiene ningún factor de riesgo.

- ✓ **FreeBSD Comprobaciones de Seguridad Locales.**

El host remoto omite una actualización al sistema y el paquete siguiente es afectado: pivote-weblogs. Una solución puede ser poner al día el paquete sobre el host remoto.

- ✓ **FTP.**

El servidor remoto FTP es propenso a un ataque de inyección SQL. La versión FTP Xlight instalado sobre el host remoto es vulnerable a un ataque de inyección SQL durante la conexión. Esto permite a un atacante ejecutar órdenes arbitrarias de SQL en el contexto del servidor FTP. Las instalaciones que no usan la autenticación externa ODBC no son afectadas por esta vulnerabilidad. El factor de riesgo es alto.

- ✓ **Gain a shell remotely.**

El servicio de antivirus remoto es afectado por múltiples publicaciones. Tales versiones son afectadas por una o varias de las publicaciones siguientes: Un defecto de segmentación puede ocurrir procesando archivos corrompidos LZH. Errores de acceso de memoria inválidos en 'libclamav/chmunpack.c' procesando archivos mal formados CHM pueden conducir a un choque. El factor de riesgo es alto.

- ✓ **Gain root remotely.**

El servidor web remoto es afectado por una vulnerabilidad de desbordamiento de buffer. El host remoto controla el Directory, un software de servicio de directorio de

Novell. El componente Monitor incluido con la versión instalada es afectado por una vulnerabilidad de desbordamiento de buffer. El factor de riesgo es crítico.

- ✓ **General.**

El servidor web remoto puede estar acostumbrado a identificar el sistema operativo del host. Saben que estos sistemas son muy inseguros, y un intruso fácilmente puede romperse en ello para usarlo como una plataforma de lanzamiento para otros ataques. Además, estas unidades transportan con varias cuentas con una contraseña en blanco o fácilmente adivinada. Sin embargo, Nessus no ha comprobado nada para ello. No tiene ningún factor de riesgo.

- ✓ **Gentoo Local Security Checks.**

El host remoto es afectado por la vulnerabilidad descrita en GLSA-200903-41. Un atacante local podría atraer a un usuario en un ambiente de trabajo, posiblemente causando la ejecución de código arbitrario con los privilegios del usuario que controla el uso. El factor de riesgo es medio.

- ✓ **MacOS X Local Security Checks.**

El host remoto OS X contiene un uso que es afectado por una vulnerabilidad de descubrimiento remota de la información. La versión remota de iTunes es afectada por una vulnerabilidad de descubrimiento remota de la información. Engañando a un usuario sobre el host afectado en la autenticación a un podcast malévolo, un atacante podría adelantar iTunes del usuario que considera la información. El factor de riesgo es medio.

- ✓ **Mandrake Local Security Checks.**

El host remoto omite el pedazo para MDKA-2007:133 consultivo. Puesto al día los paquetes que proporcionan los sistemas de Linux que no contienen la nueva información del 2007. El factor de riesgo es alto.

- ✓ **Misc.**

El servicio de antivirus remoto es afectado por una vulnerabilidad de evasión de exploración. Tales versiones fallan en manejar ciertos ficheros mal archivados de

'RAR', y de ahí puede ser posible para ciertos ficheros archivados evadir la detección del motor de exploración. El factor de riesgo es medio.

- ✓ **Backdoors.**

El host remoto puede ser comprometido. Esto es probablemente una puerta trasera. Un atacante puede usar esta vulnerabilidad para robar datos confidenciales, impedir a su sistema trabajar correctamente, o lanzar ataques contra otras máquinas sobre su red. La solución a esto es desinfecte o instale de nuevo su sistema operativo.

- ✓ **Netware.**

El servidor es configurado con la contraseña por defecto. Los usuarios podrían ser privados de acceder a un recurso determinado. Además esto le puede permite a algún atacante ver la configuración del servidor y localizar a otros servidores sobre la red. El factor de riesgo es alto.

- ✓ **NIS.**

Si un atacante usa este tipo de plugins y proporciona la dirección correcta del cliente, entonces él recuperará su dominio NIS del servidor. Una vez que el atacante descubre el nombre de dominio NIS, este fácilmente puede conseguir su archivo de contraseña NIS.La solución a esto es filtrando el tráfico entrante para prevenir conexiones al servidor o desactivar este servicio. El factor de riesgo es alto.

- ✓ **Peer-To-Peer File Sharing.**

El host remoto puede ser afectado por múltiple vulnerabilidades. Un atacante podría explotar este defecto para robar las credenciales del usuario.

- ✓ **Policy Compliance**.

Usa las credenciales suministradas, este plugins realiza una comprobación de cumplimiento contra la política dada.

- ✓ **Port scanner**.

Es posible conseguir los puertos abiertos controlando el orden de netstat remotamente. La técnica usada es la del ping TCP, es decir esta escritura envía al host remoto un paquete con la bandera SYN, y el host contestará con un RST O UN SYN/ACK.

- ✓ **Red Hat Local Security Checks**.

En la comprobación de seguridad de redes hat sus paquetes están disponibles para el ÑU Ghostscript, que fija una vulnerabilidad encontrada durante la interpretación.

- ✓ **Remote file access**.

El host remoto controla a un servidor vulnerable que puede permitir a un atacante autenticarse sin credenciales apropiadas. Además hay un desbordamiento de buffer en algún sitio web, que permiten a un intruso remoto ejecutar cualquier acción sobre el host remoto.

- ✓ **RPC**.

Esta vulnerabilidad fue descubierta en 1995 y fijada entonces. Si usted no usa este servicio, esto puede hacerse una amenaza de seguridad en el futuro, si una vulnerabilidad es descubierta.

- ✓ **SCADA**.

El host remoto contiene un uso que es afectado por una vulnerabilidad de desbordamiento de buffer.

- ✓ **Service detection**.

El host remoto controla al servidor.

- ✓ **Settings**.

El servicio de registro no podía ser parado después de la exploración. Mientras Nessus satisfactoriamente comenzó el servicio de registro, esto no puede ser pararlo después de la exploración, podría querer inutilizarlo.

- ✓ **Slackware Local Security Checks**.

Un desbordamiento ha sido encontrado en la dirección que maneja el código.

- ✓ **SMTP problems**.

El servidor de correo remoto es vulnerable a una negación local de ataque de servicio. Un atacante local puede tener acceso al descriptor para lanzar una negación de ataque de servicio. Parece haber un desbordamiento de buffer en el servidor remoto SMTP.

- ✓ **SNMP**.

Inutiliza el servicio de SNMP sobre el host remoto, si usted no lo usa, o filtra paquetes entrantes UDP que van a este puerto. Un atacante puede usar esta información para ganar más conocimiento sobre el host remoto, o cambiar la configuración del sistema remoto.

- ✓ **Solaris Local Security Checks**.

Permite a atacantes remotos tener acceso archivos o ejecutar el código arbitrario.

- ✓ **Ubuntu Local Security Checks**.

Estos paquetes remotos omiten parches de seguridad.

- ✓ **Useless services**.

El host remoto controla un ident. El servicio de 'ident' proporciona la información sensible a atacantes potenciales. Es diseñado para decir cuales cuentas controlan que servicios. Esto ayuda a atacantes enfocar servicios de valor.

- ✓ **Web Servers**.

El servidor web remoto permite el acceso inautenticado al personal administrativo. Permitir el acceso de alguna persona sin una previa autenticación, podría hacer cosas que no están dentro de su perfil.

- ✓ **Windows**..

El host remoto tiene un control de ActiveX que es afectado por una vulnerabilidad de desbordamiento de buffer. Además que es posible obtener el nombre de red del host remoto.

- ✓ **Windows: Microsoft Bulletins**.

El host remoto es vulnerable a DNS y ataques de spoofing. Esta vulnerabilidad permite a un atacante enviar los paquetes mal formados que van a utilizar el 100 % del CPU, haciéndolo casi inutilizable para los usuarios legítimos.

- ✓ **Windows: User management.**

Es posible recuperar la política de contraseña del host remoto a través de la utilización de las credenciales suministradas.

Opciones	**Descripción**
Todas Inutilizables.	Inutiliza todos los plugins y sus familias.
Todas Permitidas.	Esto es un modo fácil de crear una exploración que comprobará todas las vulnerabilidades posibles. Note que algún plugins requiere opciones remotas de configuración.
Hallazgo.	Este rasgo le permite encontrar plugins por ID, familia, o el nombre de plugins. Por ejemplo, si usted quiere comprobar todas las formas de vulnerabilidades de las ventanas, usted puede seleccionar "el nombre" y en la caja de texto, escribir "Ventanas". Esto estrechará la lista de plugins seleccionado a lo que usted ha buscado.
Muestre Todo.	Seleccionar esta opción negará cualquier plugins que usted ha seleccionado por medio del botón "de Hallazgo", y devuelve su vista para mostrar todo el plugins en

	un estado no seleccionado. Esto es básicamente "un deshacer" para la opción "de Hallazgo".
Permita dependencias en el tiempo de ejecución.	Por defecto, si una escritura tiene dependencias, aquella escritura no será controlada a no ser que las dependencias catalogadas hayan sido completadas. Esta opción anula este comportamiento y hará que el escáner cargue y ejecuten cualquiera de las escrituras que requieren para el plugins que usted ha seleccionado.
Dependencias silenciosas.	Si esta opción es comprobada, la lista de dependencias no es incluida en el informe. Si usted quiere incluir la lista de dependencias en el informe, compruebe la caja.

Red

La etiqueta de Red es muy útil para ayudar a los ajustes para resultados máximos con la interferencia de red mínima. Por defecto Nessus usará cualquier tratamiento y la interconexión que impulsa el hardware lo proveerá. Esto a veces puede causar la sobrecarga de sistema y reducir la marcha en tiempos de respuesta. Estos ajustes ayudan a depurar las ineficiencias de Nessus para maximizar la eficacia.

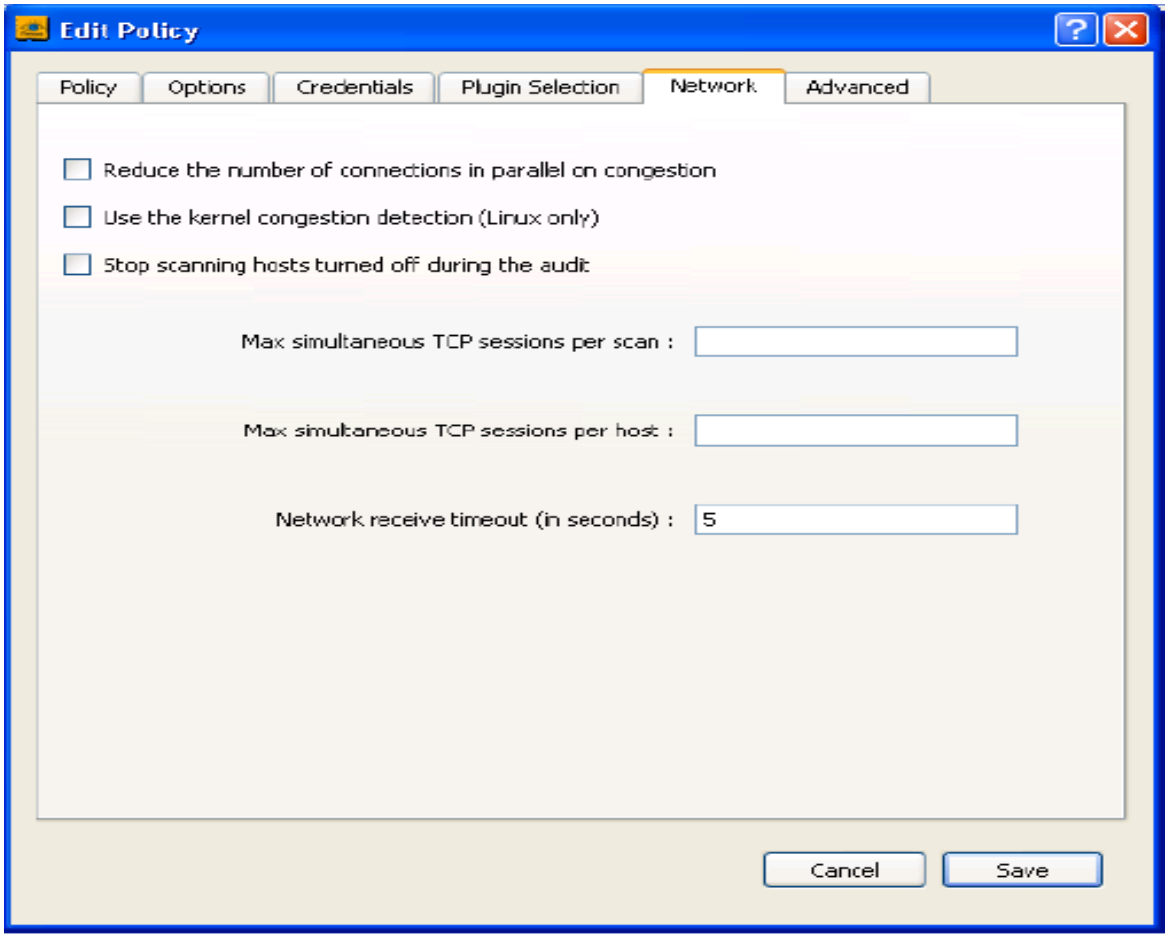


Fig. 26 Creación de una política de seguridad (Red)

La tabla debajo describe los ajustes Red:

Opciones	Descripción
Reduzca el número de conexiones en paralelo sobre la congestión.	Usado cuando Nessus envía demasiados paquetes a la vez. La comprobación de esta opción permite a Nessus descubrir cuando el tubo de red se acerca a la capacidad y el regulador atrás de la exploración para acomodar y aliviar la congestión. Una vez que la congestión es por delante, Nessus automáticamente intentará usar el espacio disponible dentro del tubo de red otra vez.
Use la detección de congestión (sólo	Permite a Nessus supervisar el CPU y

para Linux).	otros funcionamientos internos para la congestión. Nessus siempre intentará usar todo lo que este disponible. Este rasgo está sólo disponible para Escáneres de Linux basados en Nessus.
Pare el escaneó de los host durante la revisión de cuentas.	Es sólido descubrir cuando un host vivo se ha hecho insensible. Esto puede ocurrir si los usuarios apagan sus ordenadores personales durante una exploración o un plugins ha dejado de responder después de una negación de servicio de plugins. La continuación de exploraciones sobre estas máquinas enviará el tráfico innecesario a través de la red y retrasará la exploración. Compruebe esta opción para permitir a Nessus descubrir cuando estos sistemas se hacen insensibles e interrumpen la exploración de estos host.
Más sesiones TCP simultáneas por escaneo.	Limitan el número de los paquetes TCP que son enviados del escáner al mismo tiempo para la exploración entera. Poniendo esto a un ajuste inferior puede mejorar resultados, especialmente cuando la tarifa que limita el engranaje de red puede interferir con la exploración.
Más sesiones TCP simultáneas por host.	Es el mismo que la sección anterior, excepto los paquetes TCP que son limitados por el host. Esto puede mejorarse. Es resultado del host específicos y puede impedir al objetivo dejar caer paquetes debido a limitaciones sobre el host y la velocidad de paquetes entrantes.
La red recibe la interrupción.	Esto es el tiempo que Nessus esperará una respuesta de un host a no ser que no sea especificado dentro de un plugins. Si

	usted explora sobre una conexión lenta usted puede desear poner esto a un número más alto de segundos.

Etiqueta Avanzada.

La etiqueta avanzada incluye el medio para el control granular de ajustes de exploración. El seleccionar un artículo del menú desplegable mostrará remotos artículos de configuración para la categoría seleccionada. Note que esto es una lista dinámica de las opciones de configuración que es dependiente sobre los plugins a la cual el escáner conectado de Nessus tiene el acceso. Un escáner con una serie de plugins directos tendrá opciones de configuración avanzadas más disponibles que un escáner configurado con una serie de plugins certificados. Esta lista también puede cambiarse como son añadidos o modificados los plugins.

Opción	Descripción
No explorar. **Ajustes globales variables.**	Provee dispositivos frágiles, como impresoras de red y Novell host de la exploración de red. Use los artículos siguientes en este artículo de abajo del menú para poner variables globales para su exploración.
Servicios de sonda contra cada puerto.	Intenta trazar un mapa de cada puerto abierto con el servicio que corre sobre aquel puerto. Note que en algunos casos extremos, esto podría interrumpir algunos servicios y causar efectos secundarios imprevistos.
No conéctese con cuentas de usuario no especificadas en la política.	Es factible prevenir cierres de cuenta si su política de contraseña es puesta para cerrar la puerta sobre cuentas después de varias tentativas inválidas.
Permita exploración de CGI.	Activa la comprobación de CGI. La incapacitación de esta opción acelerará la revisión de cuentas de una red local.

Tipo de red.	Le permite especificar si usted usa el público routable IPs, la no Internet privada routable IPs, o una mezcla de estos. Seleccione "Mixto" si usted usa RFC 1918 direcciones y tiene múltiples encaminadores dentro de su red.
Permita escrituras experimentales.	El plugins es considerado experimental para ser usado en la exploración. No deberían permitir este ajuste explorando una red de producción.
Pruebas cuidadosas (lento).	Hace que varias escrituras NASL trabajen más difícil. Por ejemplo, examinando SMB partes de archivo, un NASL puede analizar 3 niveles profundamente en vez de 1. Esto podría causar mucho más tráfico de red y análisis en algunos casos. También note que por ser más cuidadoso, la exploración será más intrusa y con mayor probabilidad interrumpe la red.
Verbosidad de informe.	En algunos casos, Nessus remotamente no puede determinar si un defecto está presente o no. Si la paranoia de informe es puesta "a Paranoide" entonces un defecto será relatado, siempre hay una duda sobre el host remoto siendo afectado. A la inversa, un ajuste de paranoia que " Evita la falsa alarma " ,haga que Nessus no relate cualquier defecto siempre que haya una indirecta de incertidumbre sobre el host remoto. La opción de falta "Normal" será un medio del terreno entre estos dos ajustes.
Verbosidad de tronco.	Un ajuste más alto hará que la información más detallada sea proporcionada en el interior de

	exploración.
Nivel de ajuste.	Esta opción "1" para permitir eliminación de fallos para ayudar a la solución de una exploración de Nessus.
Agente-usuario HTTP.	Especifica que el tipo de navegador web de Nessus imitará explorando.
Dirección ICCP/COTP TSAP.	Este artículo de menú desplegable da expresamente con comprobaciones de SCADA. Esto determina una Conexión al Protocolo Orientado de Transporte (COTP) Puntos de Acceso de Servicio de Transporte (TSAP) el valor sobre un servidor ICCP intentando valores posibles. El principio y valores de parada son puestos " 8" por defecto.
Configuraciones de conexión.	En este artículo de menú desplegable usted puede configurar la información de conexión para los protocolos siguientes: HTTP, NNTP, FTP, POP2, POP3, y IMAP.
Misc información sobre servidor de Noticias.	Este artículo de menú desplegable puede ser usado para determinar si hay servidores de noticias que son capaces de retransmitir el bombardeo publicitario. Nessus intentará fijar un mensaje de noticias a un NNTP (el Protocolo de Transporte de Noticias de Red) el servidor, y puede probar si es posible fijar un mensaje de servidores de noticias.
De dirección.	La dirección que Nessus usará como ello intenta fijar un mensaje al servidor de noticias. Este mensaje se suprimirá automáticamente después de un período corto de tiempo.
Nombre de grupo de prueba regex.	El nombre puede ser especificado como

	una expresión regular de modo que el mensaje pueda ser fijado a múltiples grupos de noticias simultáneamente. ¿**P**or ejemplo, la falta valora " f" difundirá un mensaje de correo a todos los grupos de noticias con los nombres que comienzan con cualquier rango de ("a" a "z") y el final con ".tests" (o alguna variación que emparejó la cuerda). El signo de interrogación actúa como un carácter opcional.
Máximo crosspost.	El número máximo de noticias, los servidores que recibirán la fijación de prueba, independientemente del número de nombre. Por ejemplo, si Max crosspost es "7", el mensaje de prueba sólo será enviado a siete servidores de noticias, incluso si hay 2000 servidores de noticias que emparejan en este campo.
Distribución local.	Si esta opción es seleccionada, Nessus sólo intentará fijar un mensaje al servidor de noticias local. De otra manera, una tentativa será hecha para expedir el mensaje hacia adelante.
Ningún archivo.	Si esta opción es seleccionada, Nessus solicitará para no archivar el mensaje de prueba siendo enviado al servidor de noticias. De otra manera, el mensaje será archivado como cualquier otra fijación.
Modbus/TCP Acceso de Rollo.	Este artículo de menú desplegable dinámicamente es generado por el SCADA, plugins disponible. Modbus usa un código de función de 1 para leer "rollos" en un esclavo Modbus. Los rollos representan ajustes de salida binarios y típicamente son trazados en un mapa. La

	capacidad de leer rollos puede ayudar a un perfil de atacante de un sistema, identificar los rangos de registros para cambiar a través de " escriben el rollo " el mensaje. Las faltas para esto son "0" para el registro de Principio "y 16" para el registro de Final.
Nessus TCP Escáner.	En este artículo de menú desplegable, usted puede configurar opciones para el Nessus TCP el Escáner. Los puertos de exploración en la orden "Arbitraria ", son usados para engañar algunos Sistemas IDS más viejos. " La detección de Cortafuegos " en el menú proporciona ajustes para explorar las redes que pueden tener cortafuegos. Para seleccionar una opción, pulse sobre ello hasta que aparezca en el campo y luego pulsar sobre el botón "Salvar".
Automático (normal).	Esta opción puede ayudar a identificarse si un cortafuegos es localizado entre el escáner y el objetivo.
Ineficiente (más suave).	Incapacita el rasgo de detección de Cortafuegos.
No descubra la limitación de tarifa RST.	Incapacita la capacidad de supervisar como a menudo se reinicializa y se determina si hay una limitación configurada por un dispositivo de red.
No haga caso de puertos cerrados (agresivos).	Intentará controlar plugins incluso si el puerto aparece estar cerrado. Le recomiendan que esta opción no sea usada sobre una red de producción.
Ping al host remoto.	Nessus puede verificar si el host remoto es objetivo para la utilización de ARP, ICMP, TCP, UDP.

Puerto (s) TCP del ping destino.	Especifica la lista de puertos que serán comprobados a través del ping TCP.
Número de Reintentos (ICMP).	Le permite especificar el número de tentativas de intentar el ping del host remoto. La falta es puesta a 6.
Haga los host muertos aparecer en el informe.	Si esta opción es seleccionada, los host que no contestaron a la petición de ping serán incluidos en el informe de seguridad como host muertos.
Host existentes en el informe.	Seleccione esta opción para expresamente hacer un informe sobre la capacidad de un host remoto
Pruebe al host local de Nessus.	Esta opción le permite incluir o excluir al host local de Nessus de la exploración. Esto es usado cuando Nessus recibe caídas dentro del rango de red objetivo para la exploración.
SMB Alcance. **SMB usan el dominio SID para enumerar a usuarios.**	Si la opción que Solicita la información sobre el dominio esta puesta, entonces los usuarios de dominio serán llamados usuarios locales. Especifica el rango de SID para realizar una consulta inversa sobre user names sobre el dominio. Recomiendan al ajuste de falta.
SMB el empleo reciben a SID para enumerar a usuarios locales. **SMTP ajustes.**	Especifica el rango de SID para realizar una consulta inversa sobre user names local. Recomiendan al ajuste de falta. El SMTP (el Protocolo de Transporte de Correo Simple) las pruebas correrán sobre todos los dispositivos dentro del dominio explorado que controlan servicios SMTP. Nessus intentará retransmitir mensajes por el dispositivo especificado

	el dominio de tercero. El mensaje enviado al dominio de tercero debería ser rechazado por la dirección especificada en el para dirigirse al campo. Esto indicará la tentativa de bombardeo publicitario fracasada. Si el mensaje es aceptado, entonces el servidor SMTP se usaría para retransmitir el bombardeo publicitario.
Dominio de tercero.	Nessus intentará enviar el bombardeo publicitario por cada dispositivo SMTP a la dirección catalogada en este campo. Esta dirección de dominio de tercero debe ser fuera del rango del sitio siendo explorado o el sitio que realiza la exploración. De otra manera, la prueba podría ser rechazada por el servidor SMTP.
De dirección.	Los mensajes de prueba enviados al servidor SMTP aparecerán como si ellos provinieran de la dirección especificada en este campo.
Dirigirse.	Nessus intentará enviar mensajes dirigidos al recipiente de correo catalogado en este campo. La dirección de administrador de correos es el valor de falta, ya que esto es una dirección válida sobre la mayor parte de servidores de correo.
SNMP ajustes.	Le recomiendan que el SNMP (el Protocolo de Dirección de Red Simple) que la cuerda de comunidad sea configurada si la conocen. Si Nessus puede adivinarlo durante una exploración, será aplicado a comprobaciones subsecuentes, pero si puede ser pre configurado, una revisión de cuentas muy

SYN Exploración. **Detección de Servicio.**	detallada puede ser realizada. Una exploración de SYN permite a Nessus juntar la información sobre puertos abiertos sin completar el estrechamiento con el protocolo de comunicación TCP. Esta opción le permite poner el número de paquetes a enviar por segundos .Este menú proporciona opciones para la detección de servicio.
Número de conexiones hechas en paralela.	Número de conexiones simultáneas que pueden ser hechas.
Conexión de red.	Número de segundos para esperar una respuesta TCP.
Interrupción.	Esta opción interrumpe la conexión.
La red lee/escribe la interrupción.	Número de segundos para esperar una respuesta TCP sobre una conexión establecida antes de considerar la conexión para ser tiempo para salir de la misma.
Pruebe servicios basados en SSL.	Determina si servicios basados en SSL deben ser probados sobre puertos SSL. Todos los puertos, o ninguno. La comprobación SSL sobre cada puerto abierto puede ser quebrantadora para la red probada.
Comprobaciones de Cumplimiento de Unix.	Este artículo de menú desplegable proporciona la opción para seleccionar 5 archivos de revisión de cuentas de UNIX para asignar a la política. Al lado de cada artículo el botón "Escogido" que abrirá una ventana donde usted puede hojear y seleccionar el archivo de auditoría para

Tortura de argumentos desconocida CGIS.	usar con esa política. Este artículo "tortura" los argumentos de CGIS remoto (el Interfaz de Entrada Común) intentando pasar CGI, común para el programa de errores como argumentos.
Reflejar páginas Web.	Si la información en este menú desplegable es proporcionada, Nessus reflejará páginas web y luego probará las vulnerabilidades. Nessus reflejará páginas en orden secuencial. La página de arranque es el URL de la primera página que será probada, y el usuario también puede especificar el total el número de páginas para reflejar.
Comprobaciones de Cumplimiento de Ventanas.	Este artículo de menú desplegable proporciona la opción para seleccionar 5 archivos de revisión de cuentas de ventanas para asignar a la política. Al lado de cada artículo existe un botón "Escogido" que abrirá una ventana donde usted puede hojear y seleccionar el archivo de auditoría para usar con esta política.
Comprobaciones de Cumplimiento de Contenido de Archivo de Ventanas.	Este artículo de menú desplegable proporciona la opción para seleccionar 5 archivos de revisión de cuentas de Contenido de Archivo de Ventanas para asignar a la política. Al lado de cada artículo marcar el botón "Escogido" que abrirá una ventana donde usted puede hojear y seleccionar el archivo de auditoría para usar con esta política.

Creación de una Lista de Objetivo de Exploración

Crear un objetivo de exploración y dirigen la lista, hacer clic sobre el signo Más ("+") el botón titulado " Redes para Explorar ". " El Objetivo de Revisión ", el menú aparecerá incitando para la información sobre el objetivo de exploración. Hay cuatro opciones para entrar en el objetivo de exploración:

1. El host Solo - el host puede ser identificado como nombre de host o una dirección de IP en el formato de CIDR. Si una dirección de IP es usada, debe ser entrada en el formato punteado decimal (p.ej. 192.168.10.10 en vez de 1921681010). Si un nombre de host es usado esto debe ser una entrada válida que es resoluble sobre el servidor o el nombre de dominio totalmente calificado como nessus.tenable.com.
2. El rango – Un rango de IPS puede ser entrado. Entre la dirección de principio y la dirección final en los campos apropiados.
3. La Subred - La dirección de IP puede ser entrada con una máscara de red después de la dirección.
4. Host en el archivo - Un archivo con una lista de host puede ser usada pulsando sobre " el archivo Escogido… "para hojear el archivo. Seleccione el archivo y de clic en "Abierto".

Después de que usted ha entrado, de clic en el host y luego sobre "Save".

Por ejemplo, para explorar la máquina que controla Nessus, escoja la opción" al host Solo " y entre el IP interno 127.0.0.1.

Usted puede entrar múltiples objetivos de exploración en la dirección, cataloga y con criterio selectivo van marcando los que usted quiere usar.

Generación y Utilización del Archivos de .nessus

Una vez que usted ha creado una política y la lista de direcciones de exploración, usted puede salvar la configuración en el formato de archivo de .nessus de la ventana principal NessusClient seleccionando "el Archivo" y luego " Salvar Como " del menú principal.

Para tener acceso al archivo salvado de .nessus sobre futuras sesiones, simplemente vaya "al Archivo" y de clic sobre " Abierto". Sobre sistemas de ventanas, los archivos salvados .nessus son almacenados en C:\Documents y Ajustes \ <user name> \My Documents\Tenable\Nessus el Cliente. Sobre sistemas Linux, los archivos salvados .nessus son almacenados bajo el directorio principal del usuario (p.ej.

/root/my_policy.nessus).

Compartiendo los archivos de .nessus es útil si usted requiere la misma política contra escáner diferente de Nessus.

Note que una política que tiene esta política a través de múltiples sesiones la opción seleccionada no puede ser salvada al archivo de .nessus. La utilización de esta opción quiere decir que la política debe hacerse, una de la política de falta mostrada siempre que el NessusClient sea iniciado o siempre que la opción " la Nueva Sesión " sea seleccionada del menú principal.

Una vez que la opción " política a través de múltiples sesiones " halla sido seleccionada, debe ser salvada después de las instrucciones (a el archivo de .nessus), pero la política no será escrita al archivo de .nessus. En cambio la política es salvada al archivo Policies.xml que contiene los ajustes de falta de la instalación NessusClient. Sobre sistemas de ventanas, el archivo Policies.xml es almacenado en C:\Documents y Ajustes \ <user name> \Local Settings\Application Data\Tenable\Nessus Client\. Sobre Sistemas Linux, el archivo Policies.xml es almacenado en el directorio de .nessus-cliente del usuario, que está en el directorio de casa del usuario.

Lanzamiento de una Exploración

Para lanzar una exploración, simplemente seleccione la política y conecte la red que usted desea usar de la página principal y haga clic sobre el botón " la Exploración Salvar". La ventana "de Informe" será mostrada con el mensaje " la Exploración en curso " en el fondo de la ventana con un icono de actividad que indica que esto trabaja. Como los resultados se hacen disponibles, ellos pueden ser mostrados pulsando sobre "el +" en la opción al lado la dirección IP como se muestra en el ejemplo.

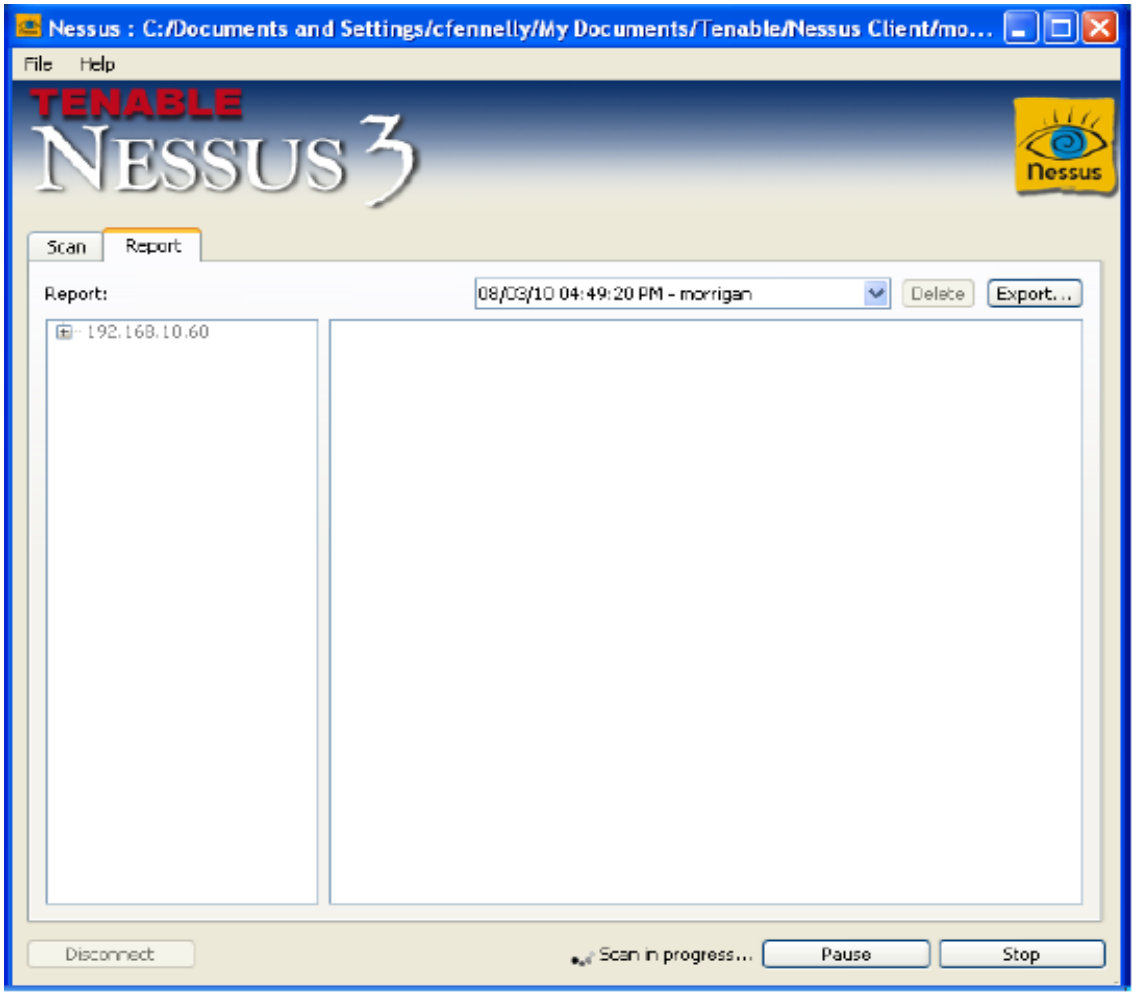


Fig. 27 Pantalla Principal (Reporte)

Informes

La parte de los cambios introducidos con el formato de archivo de .nessus es que los informes son salvados con la configuración de exploración en el archivo de .nessus. Para tener acceso a los informes generados usando una política específica que está contenida en el archivo .nessus, primero cargan el archivo de la ventana principal de nessus seleccionando "el Archivo", luego "Abren" y seleccionan el archivo apropiado de .nessus. Los informes archivados serán cargados con la política y objetivos de exploración asociados con los informes, como se muestra:

Fig. 28 Cargar los informes de archivos.

Una vez que el archivo de .nessus es cargado, hacer clic sobre la etiqueta "de Informe". Todos los informes almacenados en el archivo estarán disponibles en el menú de caída:

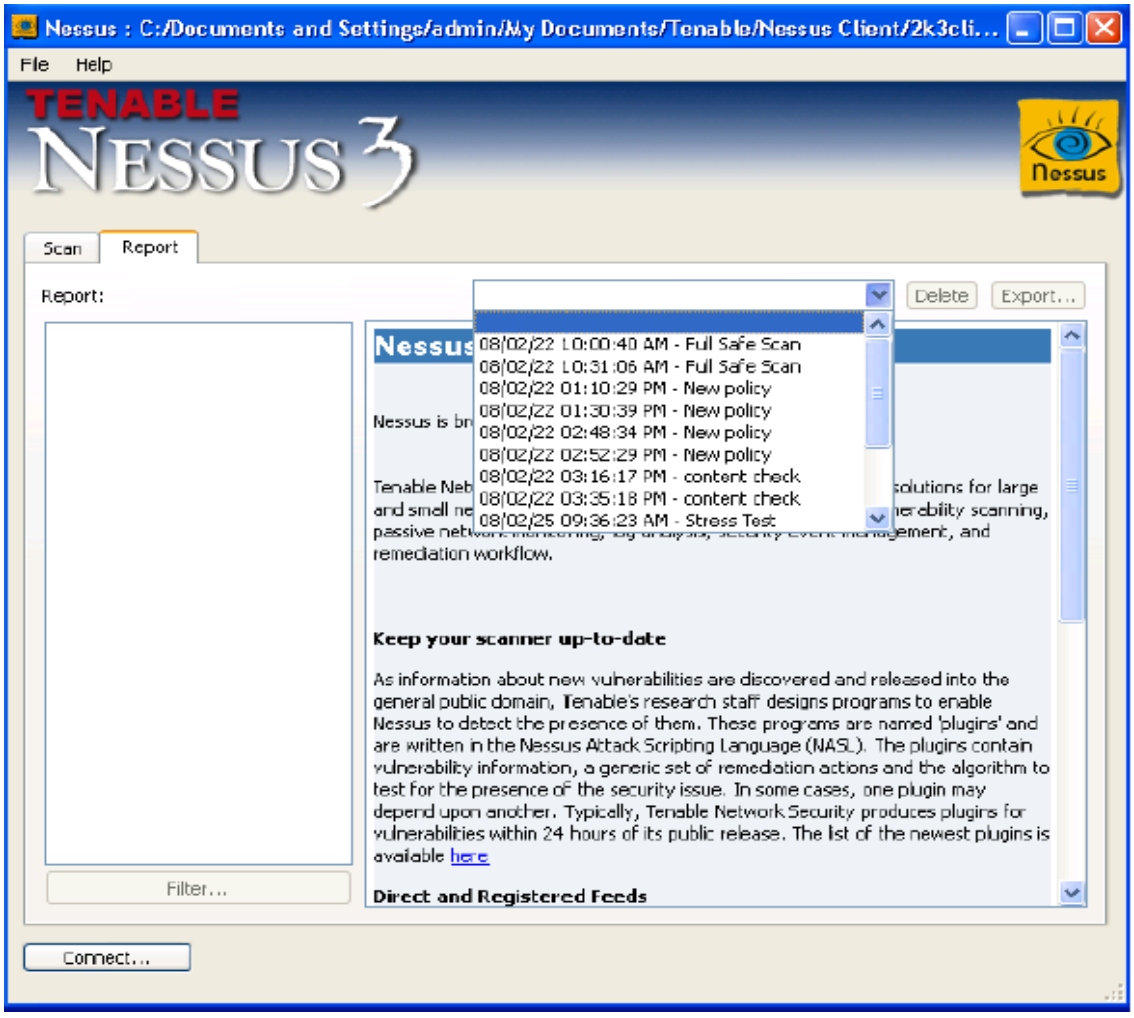


Fig. 29 Mostrar los archivos disponibles.

Los Filtros de Informe

Una vez que un informe es cargado el botón está disponible en el lado

Inferior izquierdo de la ventana. Clic sobre este botón presentará al usuario con un cuadro de diálogo que puede ser usado, crea una declaración simple o compleja con filtro. Se muestra a continuación.

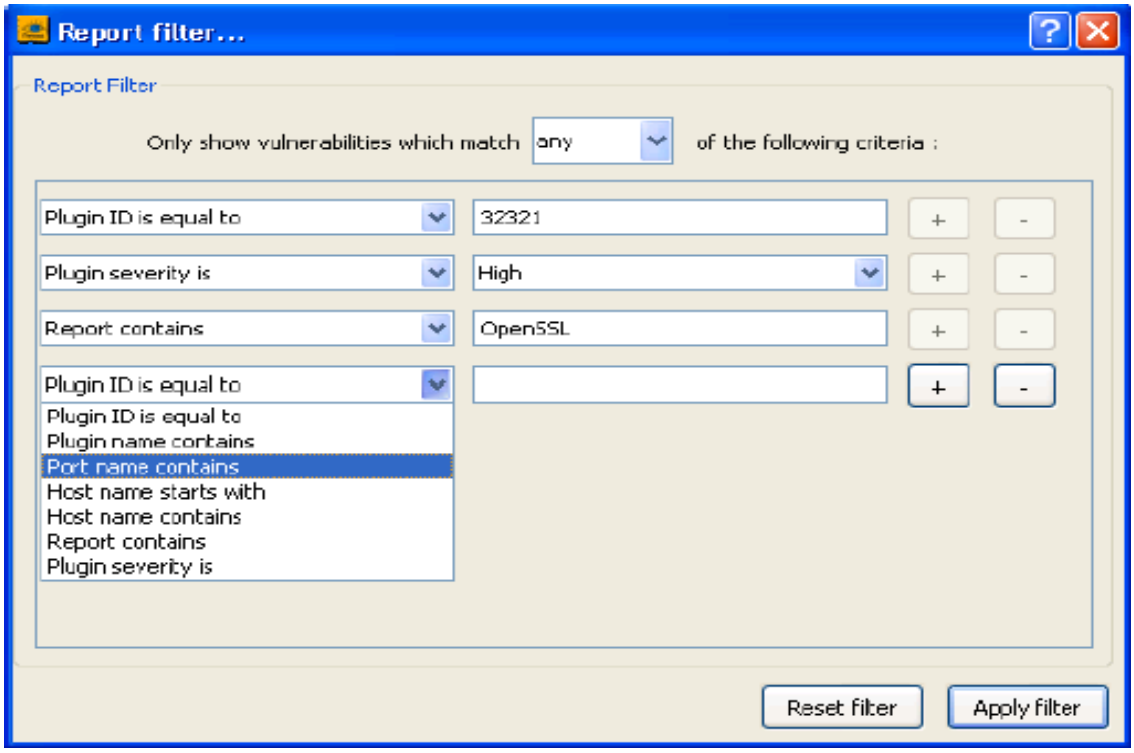


Fig. 30 Creando el reporte de filtro

Esta ventana permite al usuario de Nessus crear una serie de reglas donde alguna o todas de las condiciones siguientes son encontradas:

· ID de plugins

· Nombre de plugins

· Nombre de Puerto

· Nombre del host

· Contiene severidad

Todos los campos usan una caja de texto para entrar números excepto el nivel de severidad que deja al usuario escoger una lista de bajo, medio, o alto.

Por defecto, todas las opciones son puestas, así usted podría escoger los nombres de puerto de HTTP, HTTPs y SMTP para toda la web y enviar por correo electrónico vulnerabilidades del servidor. Si la opción "toda" es escogida, entonces sólo las vulnerabilidades que corresponde a los criterios enteros serán catalogadas. Tenga presente que si usted escoge dos filtros que crean reglas exclusivas como una regla de puerto de emparejar "HTTP" y una segunda regla de emparejar un nombre de

puerto de "SMTP" usted probablemente no tendrá ningún resultado de correspondencia.

Una vez que una declaración deseada con filtro es puesta, sólo los sistemas y las vulnerabilidades que hacen las reglas de filtrado son mostrados.

Los filtros que son en efecto también el control de los datos son enviados al .html.nsr, o formatos de archivo .nbe. Esto permite que usted seleccione qué tipo de datos entre en sus informes .html de web o esto sea exportado.

Para reinicializar el filtro, simplemente escoja el botón otra vez del “Filtro “ y reinicialice el filtro.

Exportación del Informe

Una vez cargado, un informe puede ser salvado a un archivo, exportándolo en el formato de HTML que usa el botón "De exportación":

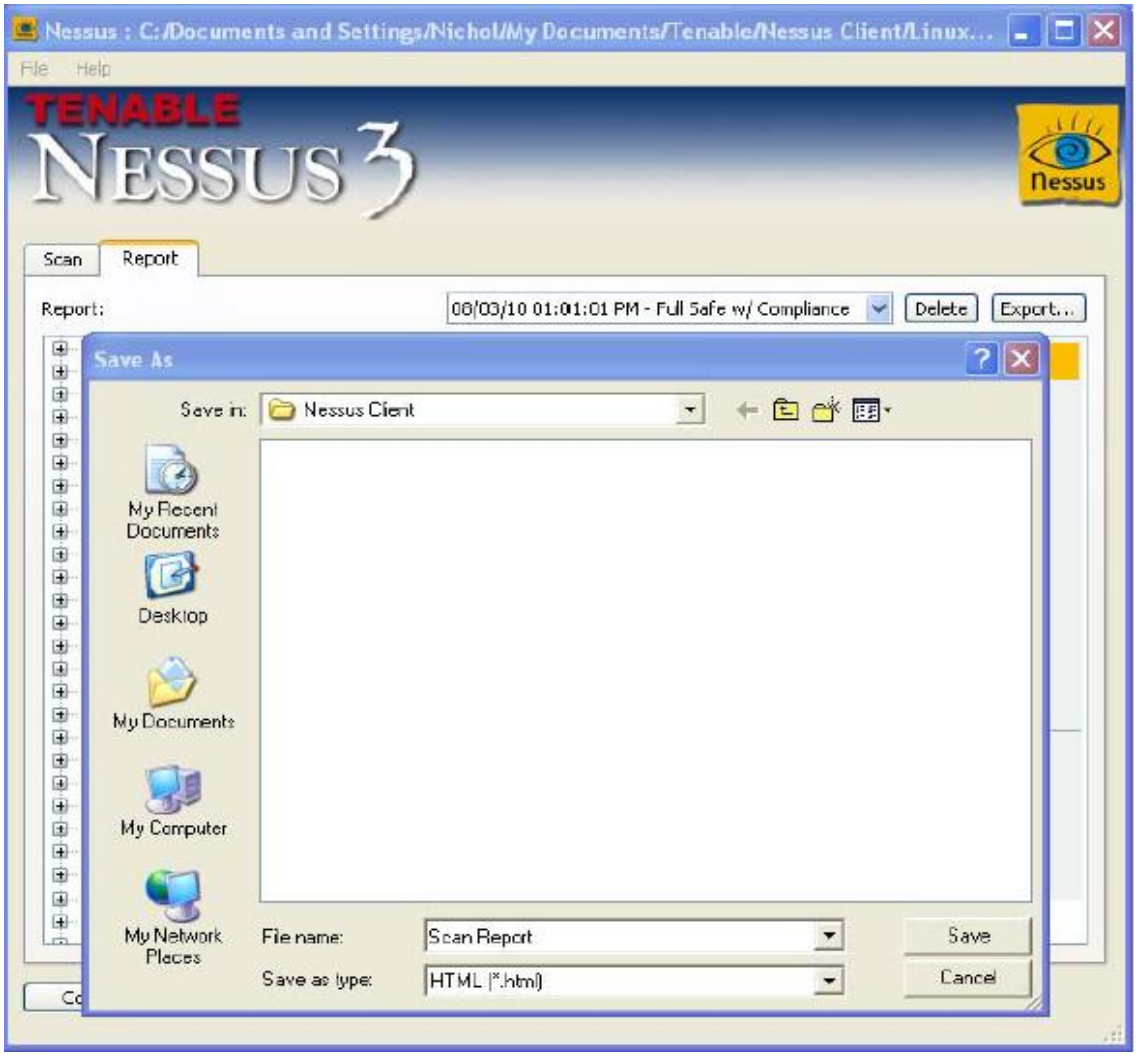


Fig. 31 Exportación de Informe.

El informe exportado es salvado en el mismo directorio que los archivos .nessus, C:\Documents y Ajustes \ <user name> \My Documents\Tenable\Nessus el Cliente.

Otros Clientes Nessus

Además del NessusClient GUI, apoyos Sostenibles existen dos métodos mas para comunicarse con el servidor Nessus: el interfaz de línea de mando de UNIX y el Centro de Seguridad.

Línea de mando que Usa Archivos Nessus

Argumento	Descripción
- el punto-nessus <el archivo>	Esto siempre es proporcionado como el primer parámetro binario pasado al nessus para indicar que el archivo de .nessus será usado. <el archivo> es la ubicación y el nombre del archivo de .nessus para ser usado.
- política <política>	El nombre de una política contenida en el archivo designado de .nessus. Proporcionan el parámetro de política lanzando una exploración de la línea de mando. Note que el nombre de política proporcionado debe ser el nombre exacto de política, incluyendo cotizaciones solas, como lo que es mostrado usando el " -listpolicies".
- la política de lista	Proporciona los nombres de toda la política de exploración contenida en el archivo designado de .nessus.
- los informes de lista	Proporcionan los nombres de todos los informes contenidos en el archivo designado de .nessus.
- el archivo de llegada <el archivo>	Anula los objetivos proporcionados en el archivo designado de .nessus y usa aquellos contenidos en el archivo especificado.

Anexo-2: Manual de usuario de Brutus.

Manual de usuario de la Herramienta para Pruebas de Seguridad: Brutus

Versión 1.0

Introducción:

Brutus es un excelente crackeador de contraseñas remoto on-line muy reconocido por su rapidez (puede llegar a 2500 palabras por segundo) y su eficaz y cómodo diseño. Se trata de un auténtico todo terreno en el mundo del Crack.

Brutus abarca una extensa lista de tipos de autenticación que puede llegar a crackear, tales como HTTP (Autentificación Básica), HTTP (Autenticación por Formulario HTML), POP3, FTP, SMB, Telnet, y otros tipos tales como IMAP, NNTP y NetBus.

Entre sus principales características se pueden enumerar las siguientes:

1. Posee un motor gradual de la autentificación.
2. Permite más de 60 conexiones simultáneas (Se podría tirar abajo algún servicio).
3. Obtiene parejas de usuario / contraseña mediante ataque simple con diccionario o mediante fuerza bruta.
4. Dispone de un generador de diccionarios bastante completo.
5. Guarda sesiones y las continúa posteriormente. Se pueden exportar e importar.

La versión corriente de Brutus es ' Brutus AET2 ', fue liberado el 28 de enero de 2000. El tamaño de archivo es 331 kilobytes.

Descargar Brutus.

Para descargar Brutus solo tiene que dirigirse a esta dirección http://www.hoobie.net/brutus/brutus-download.html y descargarlo en el escritorio de su máquina.

Ventana principal de Brutus.

Lo primero que debe hacer es abrir el Brutus AET2, y aparecerá una ventana como esta:

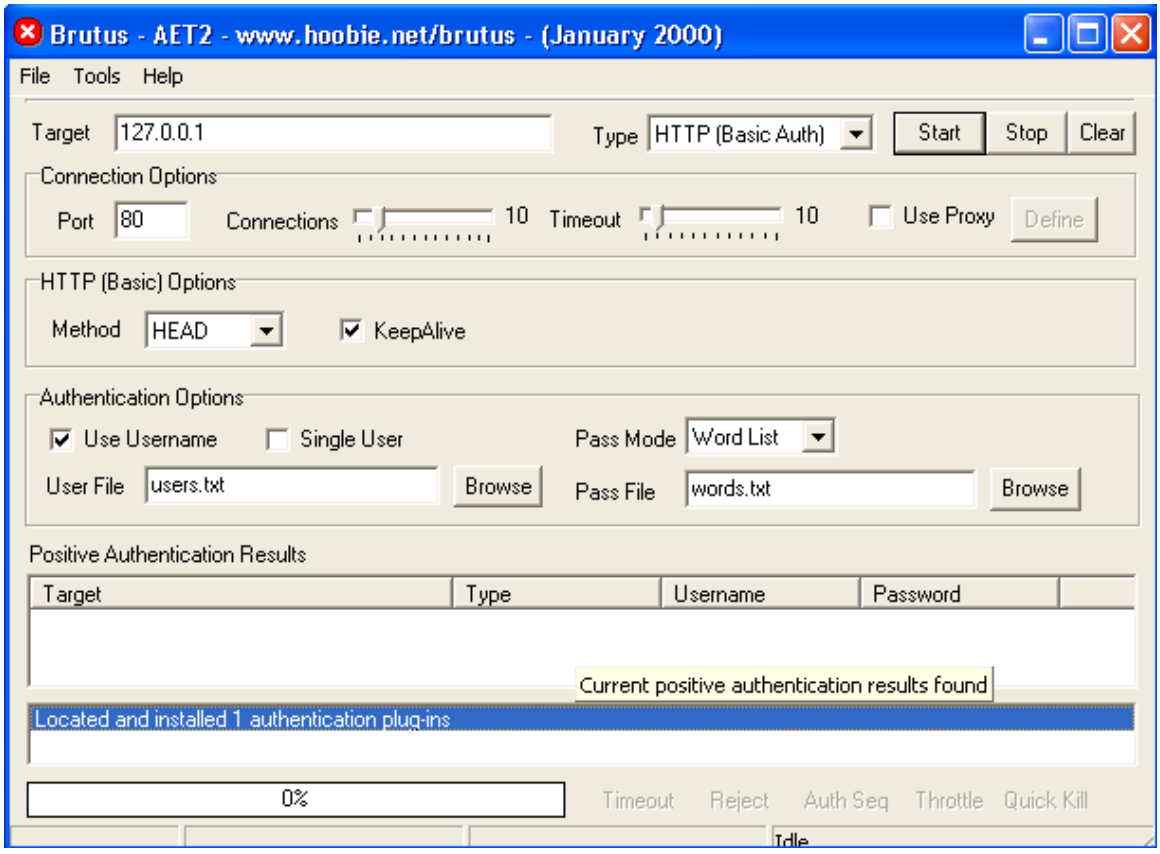


Fig. 1 Pantalla principal.

A continuación se explicará para que sirve cada sección del programa:

- En **"File"** , sale una ventana despegable con varias opciones, Import Service, que sirve para importar algún Servicio que no viene incluido en el programa como...HTTP (Autentificación Básica), HTTP (HTML Form/CGI), POP3, Ftp, SMB, Telnet, IMAP, NNTP, NetBus etc.

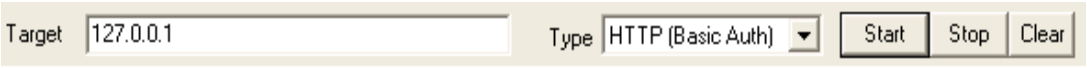
- **"Export Service"**, sirve para exportar si tiene algún servicio nuevo para el programa.

- **"Load Sesión"**, sirve para cargar alguna sesión que deje a medias.

- **"Save Sesión "** sirve para guardar la sesión que se esté ejecutando y poder volver a ella siempre que quiera.

- **"Restore Last "** sirve para restaurar todo lo que anteriormente halla elegido.

- En **"Tools"**, se encuentra Word List Generador que sirve como su nombre indica para generar, gestionar y optimizar tu Word List (lista de password).



- En **"Target"** sirve para poner la clase de Target que se quiera craquear o sea la IP, si es tipo HTML se pondrá la dirección URL y el programa la convertirá en IP, así mismo pasara si pone un FTP que va a craquear.

- En **"Type"** se selecciona el tipo de auto identificación que va a craquear. Se puede poner el protocolo POP 3, NetBus y FTP.

✓ Las 2 primeras opciones **HTTP** se utilizan para entrar en esas páginas que al intentar acceder sale una ventana pidiendo login y pass, utilizará uno u otro dependiendo del tipo de autentificación que requiera dicha web, se comprobará probando con ambos tipos., las HTTP "CGI application authentication" son las que usan programas CGIs, para autentificarse . Para utilizarlos no hay más que poner en target por ejemplo: www.victima.co m o víctima.com y selecciona en authentication options user name y pass mode el método que más convenga como norma general usará en este tipo de webs un diccionario del idioma en el que este dicha web.

✓ **POP3** se usará esta opción para tomar cuentas de correo de servidores que tenga pop3 activo. Seleccionará POP3 y en target escribirá POP3.víctima.com por ejemplo: (pop3.terra.es) o en su caso la ip, en authentication options marca user name y single user, y escribirá en la casilla de userID víctima.terra.es si la cuenta de la

víctima es (victima@terra.es). Lo escribirá con punto (.) no arroba (@) y de nuevo se usará el modo de pass que más le interese ya sea Word List, Combo List o Brute Force.

✓ **FTP** en este caso convendrá tener un nombre de usuario del que se tratará de sacar el pass aunque también se puede buscar nombres de usuario al azar, lo cual lógicamente le llevará mucho tiempo. Se selecciona **type ftp** y si tiene un nombre de usuario se marca user name y single user y se escribe el nombre de usuario por ejemplo: pardillo y se selecciona el método Word List, Combo List o Brute Force si no tiene nombre de usuario se selecciona Word List y se marca solo la casilla user name.

✓ **TELNET** es fácil se hará igual que con un ftp pero se selecciona type telnet.

✓ **SMB (NetBIOS)** este es otro servicio del cual también se puede sacar provecho usando Brutus si es que está protegido. NetBIOS se puede transportar sobre diferentes protocolos de red: NetBEUI,IPX/SPX y TCP/IP, etc. también se pueden ofrecer este servicio en otros sistemas, OS/2, BeOS etc., la única cosa que se tendrá que tener en cuenta a la hora de realizar el ataque, en **target** se escribirá el ip de la víctima seleccionando **type** SMB(NetBIOS) y ahora en **SMB options** si se trata de una máquina NT se escribirá el nombre del dominio si es solo una PC con win2 9x no se marcará esta opción, a continuación en authentication options se selecciona user name y el modo de pass que más le convenga.

- **NETBUS** esta es otra utilidad por si se le olvida el pass de alguna víctima de NetBus o simplemente quiera entrar en alguna que se ha encontrado y esta protegida con pass. Si conoce este troyano sabrá que cuando se protege con pass para entrar solo se requiere el pass no el login, por eso al seleccionar **type NetBus** no podrá seleccionar en authentication options la opción de user name o single user si no que solo podrá seleccionar la opción de pass (Word List, Combo List o Brute Force) .

- En **" Method"** se pondrá HEAD. Este se puede cambiar si lo desea.

- En **"Port"** esta opción cambiará en dependencia al tipo que seleccione en **"Type"**, por ejemplo si selecciona Telnet se pondrá automáticamente el puerto 23 y así sucesivamente con todos.

- El **"Use proxy"** es para utilizar SOCKS proxys a la hora de hacer el crackeo véase esta imagen:

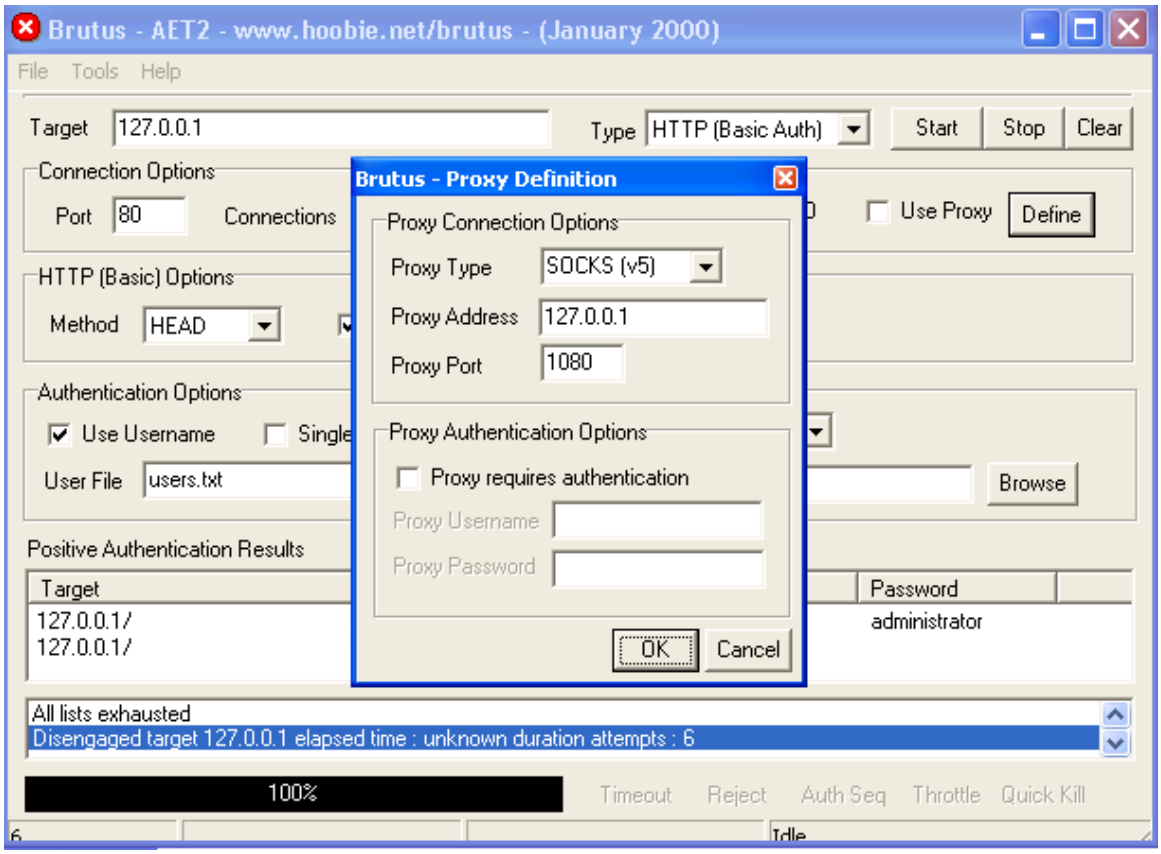


Fig. 2 Definición del proxy

Para trabajar con **"Proxy Type"** tiene que seleccionar el tipo de proxy, En **"Proxy Address"** el IP o en su defecto el host del Proxy y en **"Proxy Port"** el número del puerto que va a conectar recordando que, si escanea HTTP (Autentificación Básica) o HTTP tiene que usar el puerto 80 o 8080y si escanea, FTP, Telnet, etc. usar un SOCKET proxy o sea por el Puerto 1080.

En **"Authenticacions Options"** hay varias opciones, se comienza: Si no sabe el User name de la víctima ya sea al intentar hackear HTML, Telnet, FTP, etc, puede utilizar un archivo en el que tenga posibles nombres de usuarios entonces deja la opción **"User name"** activada y en el botón **"Browser"** busca en el disco duro el archivo que contenga posibles nombres de usuarios.

Pero si sabe el nombre del usuario pues entonces activa la opción

- "Single User" y la casilla **"User file"** se trasformara en **"User ID"** y ahí es donde tiene que poner el nombre de usuario que conoce.

En **"Pass Mode"** se selecciona el tipo de fichero que va a utilizar: **"Word List"** es la opción en la que se elige un archivo que tenga con posibles passwords y la busca en el disco duro con el botón **"Browser"** y aparecerá en la casilla **"Pass File"** la dirección completa donde tiene el archivo de Passwords.

➢ Puede seleccionar también la opción **"Combo List",** esa opción permite con una misma lista de passwords sacar el nombre de usuario y el password.

➢ La otra opción es **"Brute Force"** que esta no necesita ningún archivo con passwords ni nada, ósea, que lo hace por fuerza bruta, es algo mas lento que las demás opciones ya que tardará algo más en encontrar los pass, pero es más efectiva, si quieres configurar los rangos de passwords que va a utilizar por fuerza bruta entonces oprime el botón **"Range",** a continuación se muestra esta opción.

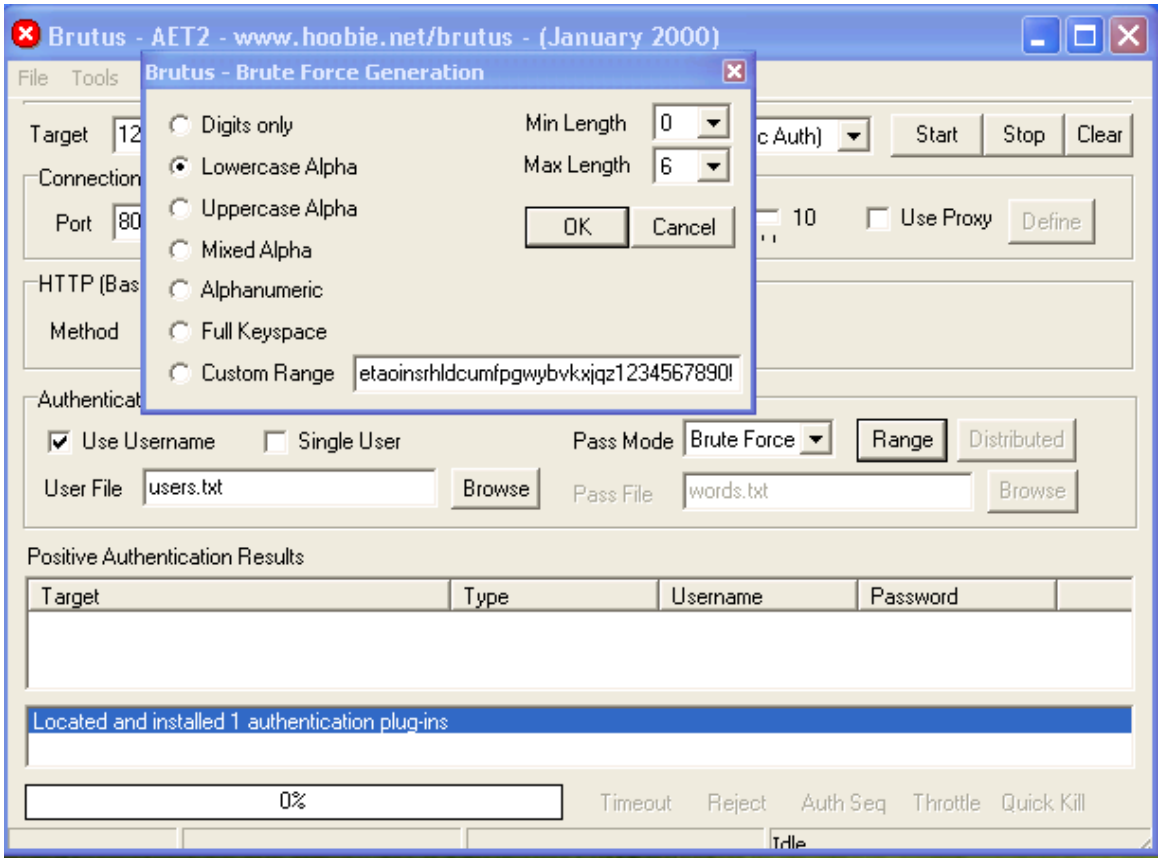


Fig. 3 Configurar los rangos de passwords

En la ventana podrá elegir que tipo de password va a utilizar por fuerza bruta, lo primero que tiene que hacer es elegir si va a empezar desde 0 hasta los dígitos que

ocupe el passwords, para eso va a **"Min Length"** y a **"Max Length"** y elige desde donde va a comenzar.

Ejemplo: si selecciona Min Length 0 y Max Length 6 pues empezará con un solo dígito o letra (desde "a" hasta "zzzzzz") e irá pasando por todas las letras y símbolos, si eliges Min Length 6 y Max Length 6 pues empezará con 6 dígitos o letras ("aaaaaa" hasta "zzzzzzz") y así sucesivamente con el número que elijas.

Luego en la parte izquierda es donde va a elegir el tipo de password que quiera craquear, ya sean letras, número, números y letras..... Esto se explicará a continuación:

"Digits Only" solo números o sea, que buscará pass por fuerza bruta que contengan solo números, ya sea una fecha, etc.

"Lowercase Alpha" esta opción son letras solo en minúscula.

"Aperchase Alpha" lo mismo que el anterior pero en mayúsculas.

"Mixed Alpha" buscaría letras pero no en orden alfabético.

"Alphanumeric" seria entre mezclando números y letras.

"Full Keyspace" como su nombre indica, utilizará todas las teclas del teclado, letras, símbolos, números etc.

"Custom Range" utiliza todo, letras, números, símbolos etc.

"Positive Authentication Results" es donde va a marcar como va el progreso de buscar Passwords y los que va encontrando, ver la Fig.6.

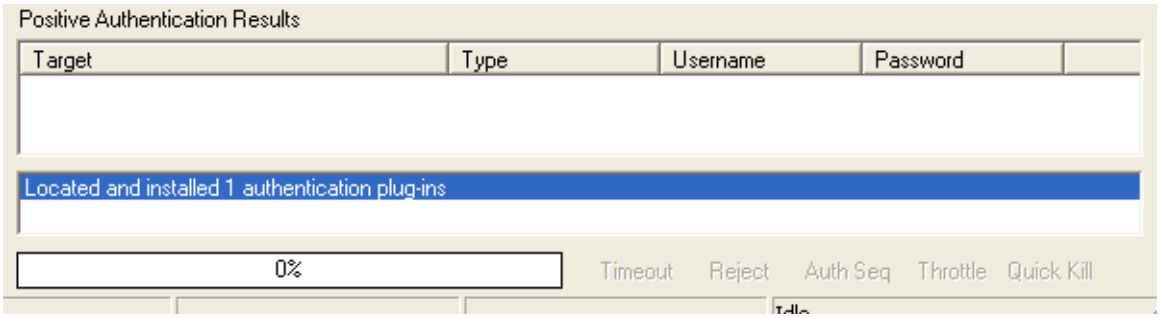


Fig. 4 Buscar el passwords.

Ejemplo malicioso del uso de Brutus.

Atacando la Autenticación Básica HTTP por fuerza bruta

El primer caso que se va a probar es extremadamente sencillo, pues el usuario y la clave estarán ambos compuestos por una palabra de únicamente tres letras, todas minúsculas.

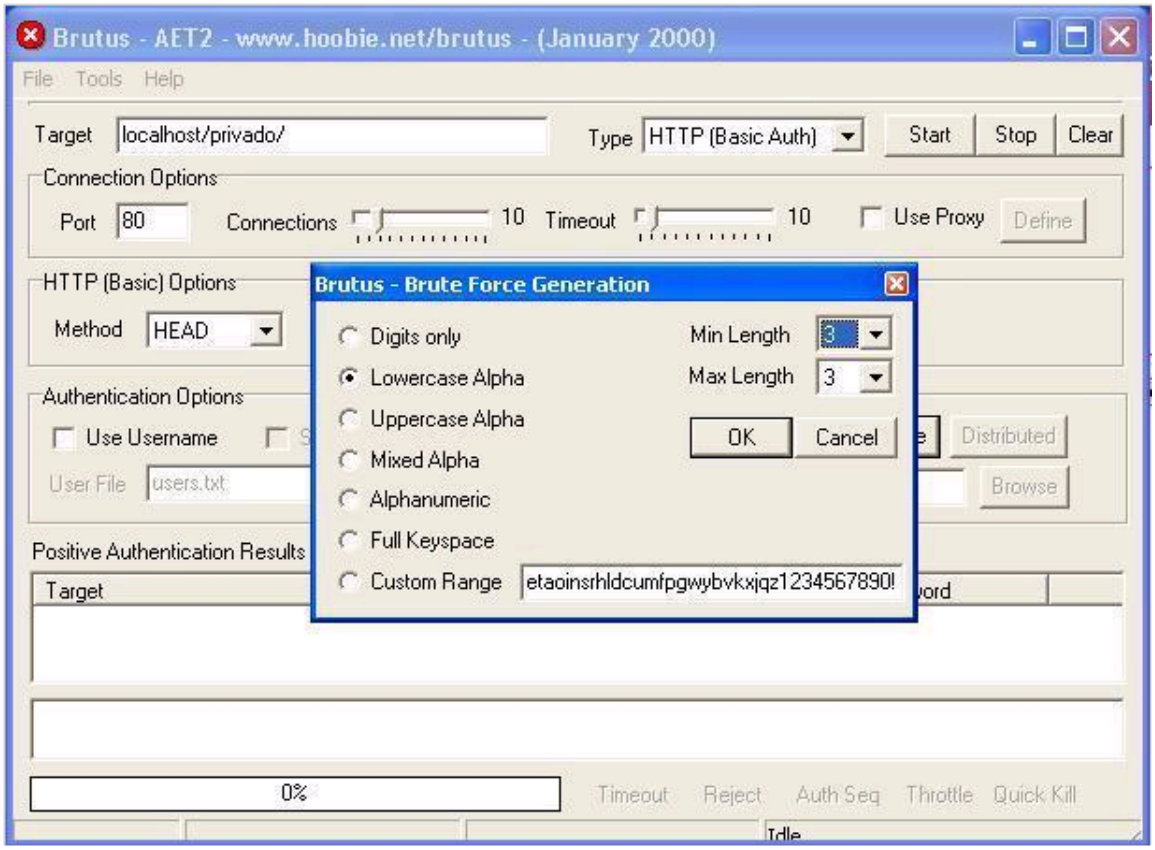


Fig. 5 Pantalla de opciones de fuerza bruta.

Y los resultados son los siguientes.

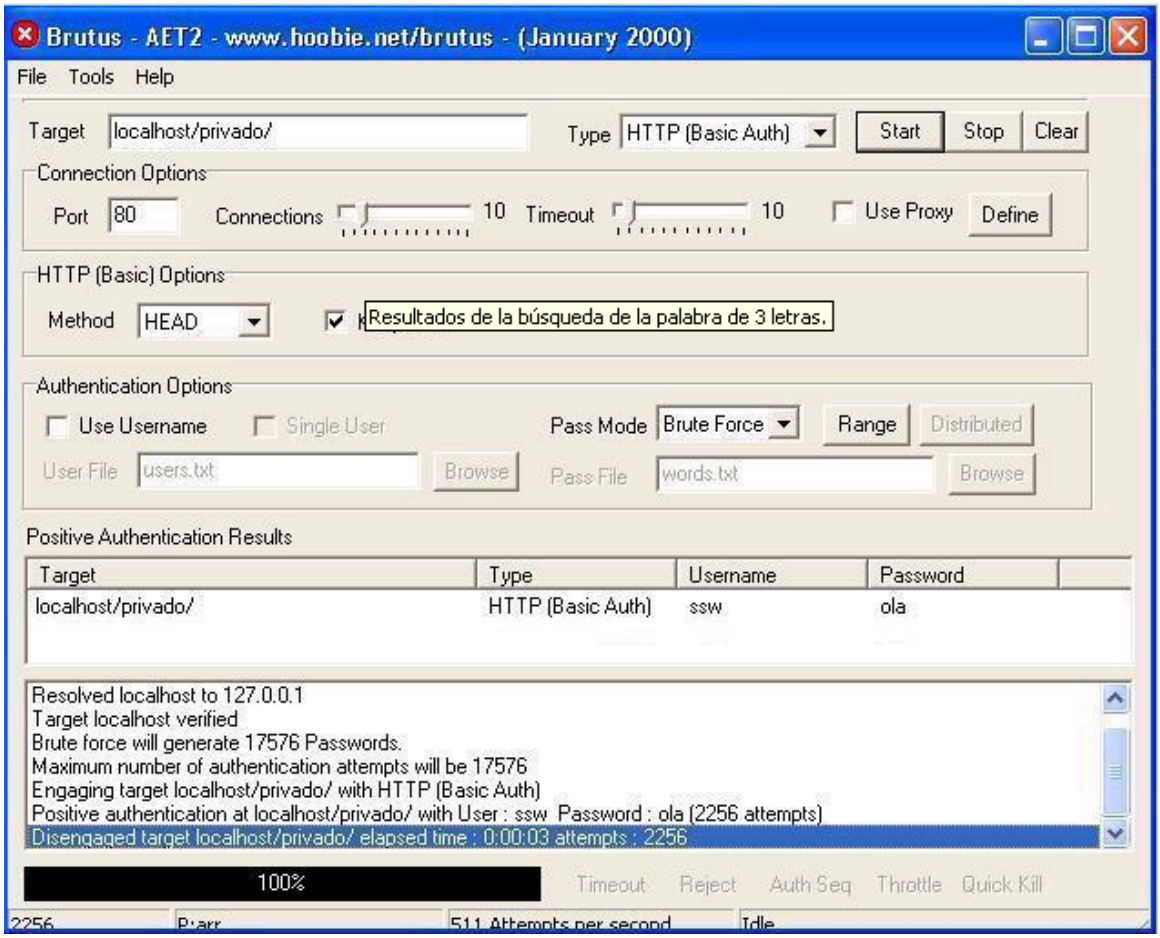


Fig. 6 Resultados de la búsqueda de la palabra de 3 letras.

Es recomendable que siempre que instalen el Brutus no lo hagan directamente, sino mediante el uso de un proxy, o cualquier otro método para mantener el anonimato porque todos los intentos que hagan quedarán registrados. Pueden reanudar su "ataque": ya que Brutus guarda el último ataque en autosave.bru, pueden reanudarlo solo con ir a File > Load Session, así como guardar sus ataques en **Save Session** con el nombre que quiera.bru.

Anexo-3: Encuesta realizada a los expertos.

Modelo para la recogida de información referente al peso de los criterios.

Guía para informar el peso de los criterios.

Fecha de recepción _/_/_

Fecha de entrega _/_/_

Nombre y Apellidos del evaluador_____.

Le otorgará un peso a cada criterio de acuerdo a su opinión y el peso total de cada grupo debe sumar:

Grupo No.1…… 40

Grupo No.2…… 20

Grupo No.3……..20

Grupo No.4……..20

Para que el peso total asignado sea 100. Cada Factor que se mida tiene un valor de 10 puntos.

Preguntas.

Grupo No 1: Criterios de mérito científico.

1. Valor científico de la propuesta.

Peso___

2. Calidad de la investigación

Peso___

3. Contribución científica.

Peso___

4. Responsabilidad científica y profesionalidad de los investigadores.

Peso___

Grupo No 2: Criterios de implantación.

5. Necesidad de empleo de la propuesta.

Peso___

6. Posibilidades de aplicación.

Peso___

Grupo No 3: Criterios de flexibilidad.

7. Adaptabilidad a entidades dedicadas a evaluar la calidad de los productos de software.

Peso ___

8. Capacidad del proceso de evaluación para la admisión de cambios que impliquen mejoras.

Peso___

Grupo No 4. Criterios de impacto.

9. Impacto en el área para la cual está destinada la guía.

Peso___

10. Organización en el proceso de desarrollo.

Peso___

Categoría final del procedimiento.

___ Excelente: Alta novedad científica, con aplicabilidad y resultados relevantes.

___ Bueno: Novedad científica, resultados destacados.

___ Regular: Suficientemente bueno con reservas.

___ Cuestionable: No tiene relevancia científica y los resultados son malos.

___ Malo: No aplicable.

Valoración final Sugerencias del experto para mejorar la calidad del proyecto.

Elementos críticos que deben mejorarse.

Descripción.

Anexo-4: Respuesta de los expertos.

1. Modelo para la recogida de información referente al peso de los criterios.

Guía para informar el peso de los criterios.

Fecha de recepción: 8/05/09

Fecha de entrega: 21/05/09

Nombre y Apellidos del evaluador: Asnier Enrique Góngora Rodríguez.

Le otorgará un peso a cada criterio de acuerdo a su opinión y el peso total de cada grupo debe sumar:

Grupo No.1…… 40

Grupo No.2…… 20

Grupo No.3……..20

Grupo No.4……..20

Para que el peso total asignado sea 100. Cada Factor que se mida tiene un valor de 10 puntos.

Grupo No 1: Criterios de mérito científico.

1. Valor científico de la propuesta.

Peso..... 9

2. Calidad de la investigación

Peso.....10

3. Contribución científica.

Peso..... 8

4. Responsabilidad científica y profesionalidad de los investigadores.

Peso.....8

Grupo No 2: Criterios de implantación.

5. Necesidad de empleo de la propuesta.

Peso..... 7

6. Posibilidades de aplicación.

Peso..... 8

Grupo No 3: Criterios de flexibilidad.

7. Adaptabilidad a entidades dedicadas a evaluar la calidad de los productos de software. Peso..... 9

8. Capacidad del proceso de evaluación para la admisión de cambios que impliquen mejoras.

Peso.....9

Grupo No 4. Criterios de impacto.

9. Impacto en el área para la cual está destinada la guía.

Peso..... 8

10. Organización en el proceso de desarrollo.

Peso.....8

Categoría final del procedimiento.

___ Excelente: Alta novedad científica, con aplicabilidad y resultados relevantes.

_**__**Bueno: Novedad científica, resultados destacados.

_x__ Regular: Suficientemente bueno con reservas.

___ Cuestionable: No tiene relevancia científica y los resultados son malos.

___ Malo: No aplicable.

Valoración final Sugerencias del experto para mejorar la calidad del proyecto.

Elementos críticos que deben mejorarse.

2. Modelo para la recogida de información referente al peso de los criterios.

Guía para informar el peso de los criterios.

Fecha de recepción: 8/05/09

Fecha de entrega: 16/05/09

Nombre y Apellidos del evaluador: Rogfel Thompson Martínez.

Le otorgará un peso a cada criterio de acuerdo a su opinión y el peso total de cada grupo debe sumar:

Grupo No.1…… 40

Grupo No.2…… 20

Grupo No.3……..20

Grupo No.4……..20

Para que el peso total asignado sea 100. Cada Factor que se mida tiene un valor de 10 puntos.

Grupo No 1: Criterios de mérito científico.

Valor científico de la propuesta.

Peso.....9

Calidad de la investigación

Peso.....9

Contribución científica.

Peso.....10.

Responsabilidad científica y profesionalidad de los investigadores.

Peso.....10.

Grupo No 2: Criterios de implantación.

Necesidad de empleo de la propuesta.

Peso.....10.

Posibilidades de aplicación.

Peso.....10.

Grupo No 3: Criterios de flexibilidad.

7. Adaptabilidad a entidades dedicadas a evaluar la calidad de los productos de software. Peso.....8.

8. Capacidad del proceso de evaluación para la admisión de cambios que impliquen mejoras.

Peso.....7.

Grupo No 4. Criterios de impacto.

9. Impacto en el área para la cual está destinada la guía.

Peso.....10.

10. Organización en el proceso de desarrollo.

Peso.....10.

Categoría final del procedimiento.

___ Excelente: Alta novedad científica, con aplicabilidad y resultados relevantes.

__X_ Bueno: Novedad científica, resultados destacados.

___ Regular: Suficientemente bueno con reservas.

___ Cuestionable: No tiene relevancia científica y los resultados son malos.

___ Malo: No aplicable.

Valoración final Sugerencias del experto para mejorar la calidad del proyecto.

Las concepciones del flujo de prueba están bien concebidas por lo que sirve de estructura para potenciar y mejor la metodología basando se en ella; sirve de base para futuras metodologías en otros tipos de software.

Elementos críticos que deben mejorarse.

Las herramientas para hacer las pruebas deben ser en modo de propuesta, ya que mañana puede ser que ya no tengan relevancia.

Se debe permitir la inclusión de nuevas pruebas

3. Modelo para la recogida de información referente al peso de los criterios.

Guía para informar el peso de los criterios.

Fecha de recepción: 8/05/09

Fecha de entrega: 15/05/09

Nombre y Apellidos del evaluador: Yuliesky Bello Chávez

Le otorgará un peso a cada criterio de acuerdo a su opinión y el peso total de cada grupo debe sumar:

Grupo No.1…… 40

Grupo No.2…… 20

Grupo No.3……..20

Grupo No.4……..20

Para que el peso total asignado sea 100. Cada Factor que se mida tiene un valor de 10 puntos.

Grupo No 1: Criterios de mérito científico.

Valor científico de la propuesta.

Peso……. 10

Calidad de la investigación

Peso……. 10

Contribución científica.

Peso…….9

Responsabilidad científica y profesionalidad de los investigadores.

Peso…….10

Grupo No 2: Criterios de implantación.

Necesidad de empleo de la propuesta.

Peso…….10.

Posibilidades de aplicación.

Peso…….10

Grupo No 3: Criterios de flexibilidad.

7. Adaptabilidad a entidades dedicadas a evaluar la calidad de los productos de software. Peso…….10

8. Capacidad del proceso de evaluación para la admisión de cambios que impliquen mejoras.

Peso…….9

Grupo No 4. Criterios de impacto.

9. Impacto en el área para la cual está destinada la guía.

Peso……. 9

10. Organización en el proceso de desarrollo.

Peso…….10

Categoría final del procedimiento.

___ Excelente: Alta novedad científica, con aplicabilidad y resultados relevantes.

_x__ Bueno: Novedad científica, resultados destacados.

___ Regular: Suficientemente bueno con reservas.

___ Cuestionable: No tiene relevancia científica y los resultados son malos.

___ Malo: No aplicable.

Valoración final Sugerencias del experto para mejorar la calidad del proyecto.

Elementos críticos que deben mejorarse.

4. Modelo para la recogida de información referente al peso de los criterios.

Guía para informar el peso de los criterios.

Fecha de recepción: 8/05/09

Fecha de entrega: 16/05/09

Nombre y Apellidos del evaluador: Roig Calzadilla Díaz.

Le otorgará un peso a cada criterio de acuerdo a su opinión y el peso total de cada grupo debe sumar:

Grupo No.1…… 40

Grupo No.2…… 20

Grupo No.3……..20

Grupo No.4……..20

Para que el peso total asignado sea 100. Cada Factor que se mida tiene un valor de 10 puntos.

Grupo No 1: Criterios de mérito científico.

1. Valor científico de la propuesta.

Peso……. 8

2. Calidad de la investigación

Peso……. 10

3. Contribución científica.

Peso……. 8

4. Responsabilidad científica y profesionalidad de los investigadores.

Peso……. 10

Grupo No 2: Criterios de implantación.

5. Necesidad de empleo de la propuesta.

Peso……. 10

6. Posibilidades de aplicación.

Peso……. 10

Grupo No 3: Criterios de flexibilidad.

7. Adaptabilidad a entidades dedicadas a evaluar la calidad de los productos de software.

Peso……. 10

8. Capacidad del proceso de evaluación para la admisión de cambios que impliquen mejoras.

Peso……. 10

Grupo No 4. Criterios de impacto.

9. Impacto en el área para la cual está destinada la guía.

Peso……. 9

10. Organización en el proceso de desarrollo.

Peso……. 10

Categoría final del procedimiento.

___ Excelente: Alta novedad científica, con aplicabilidad y resultados relevantes.

_x__ Bueno: Novedad científica, resultados destacados.

___ Regular: Suficientemente bueno con reservas.

___ Cuestionable: No tiene relevancia científica y los resultados son malos.

___ Malo: No aplicable.

Valoración final Sugerencias del experto para mejorar la calidad del proyecto.

Como sugerencia señalo la aplicación para constatar los resultados reales de la propuesta. Considero que es una buena propuesta.

Elementos críticos que deben mejorarse.

5. Modelo para la recogida de información referente al peso de los criterios.

Guía para informar el peso de los criterios.

Fecha de recepción: 8/05/09

Fecha de entrega: 17/05/09

Nombre y Apellidos del evaluador: Manuel Cheong.

Le otorgará un peso a cada criterio de acuerdo a su opinión y el peso total de cada grupo debe sumar:

Grupo No.1…… 40

Grupo No.2…… 20

Grupo No.3……..20

Grupo No.4……..20

Para que el peso total asignado sea 100. Cada Factor que se mida tiene un valor de 10 puntos.

Grupo No 1: Criterios de mérito científico.

Valor científico de la propuesta.

Peso.....9

Calidad de la investigación

Peso.....9

Contribución científica.

Peso.....8.

Responsabilidad científica y profesionalidad de los investigadores.

Peso.....10.

Grupo No 2: Criterios de implantación.

Necesidad de empleo de la propuesta.

Peso.....10.

Posibilidades de aplicación.

Peso.....9.

Grupo No 3: Criterios de flexibilidad.

7. Adaptabilidad a entidades dedicadas a evaluar la calidad de los productos de software. Peso.....9.

8. Capacidad del proceso de evaluación para la admisión de cambios que impliquen mejoras.

Peso.....9.

Grupo No 4. Criterios de impacto.

9. Impacto en el área para la cual está destinada la guía.

Peso.....9.

10. Organización en el proceso de desarrollo.

Peso.....10.

Categoría final del procedimiento.

___ Excelente: Alta novedad científica, con aplicabilidad y resultados relevantes.

___ Bueno: Novedad científica, resultados destacados.

__x_ Regular: Suficientemente bueno con reservas.

___ Cuestionable: No tiene relevancia científica y los resultados son malos.

___ Malo: No aplicable.

Valoración final Sugerencias del experto para mejorar la calidad del proyecto.

Elementos críticos que deben mejorarse.

6. Modelo para la recogida de información referente al peso de los criterios.

Guía para informar el peso de los criterios.

Fecha de recepción: 24/05/09

Fecha de entrega: 2/06/09

Nombre y Apellidos del evaluador: Tayché Capote García.

Le otorgará un peso a cada criterio de acuerdo a su opinión y el peso total de cada grupo debe sumar:

Grupo No.1……. 40

Grupo No.2……. 20

Grupo No.3…….20

Grupo No.4…….20

Para que el peso total asignado sea 100. Cada Factor que se mida tiene un valor de 10 puntos.

Grupo No 1: Criterios de mérito científico.

1. Valor científico de la propuesta.

Peso……. 9

2. Calidad de la investigación

Peso……. 10

3. Contribución científica.

Peso……. 10

4. Responsabilidad científica y profesionalidad de los investigadores.

Peso……. 10

Grupo No 2: Criterios de implantación.

5. Necesidad de empleo de la propuesta.

Peso……. 10

6. Posibilidades de aplicación.

Peso……. 10

Grupo No 3: Criterios de flexibilidad.

7. Adaptabilidad a entidades dedicadas a evaluar la calidad de los productos de software.

Peso……. 10

8. Capacidad del proceso de evaluación para la admisión de cambios que impliquen mejoras.

Peso……. 10

Grupo No 4. Criterios de impacto.

9. Impacto en el área para la cual está destinada la guía.

Peso……. 9

10. Organización en el proceso de desarrollo.

Peso……. 10

Categoría final del procedimiento.

___ Excelente: Alta novedad científica, con aplicabilidad y resultados relevantes.

_x__ Bueno: Novedad científica, resultados destacados.

___ Regular: Suficientemente bueno con reservas.

___ Cuestionable: No tiene relevancia científica y los resultados son malos.

___ Malo: No aplicable.

Valoración final Sugerencias del experto para mejorar la calidad del proyecto.

Elementos críticos que deben mejorarse.

7. Modelo para la recogida de información referente al peso de los criterios.

Guía para informar el peso de los criterios.

Fecha de recepción: 24/05/09

Fecha de entrega: 2/06/09

Nombre y Apellidos del evaluador: Delvis Echeverría.

Le otorgará un peso a cada criterio de acuerdo a su opinión y el peso total de cada grupo debe sumar:

Grupo No.1……. 40

Grupo No.2……. 20

Grupo No.3……..20

Grupo No.4……..20

Para que el peso total asignado sea 100. Cada Factor que se mida tiene un valor de 10 puntos.

Grupo No 1: Criterios de mérito científico.

1. Valor científico de la propuesta.

Peso……. 9

2. Calidad de la investigación

Peso……. 10

3. Contribución científica.

Peso……. 10

4. Responsabilidad científica y profesionalidad de los investigadores.

Peso……. 10

Grupo No 2: Criterios de implantación.

5. Necesidad de empleo de la propuesta.

Peso……. 10

6. Posibilidades de aplicación.

Peso……. 10

Grupo No 3: Criterios de flexibilidad.

7. Adaptabilidad a entidades dedicadas a evaluar la calidad de los productos de software.

Peso……. 10

8. Capacidad del proceso de evaluación para la admisión de cambios que impliquen mejoras.

Peso……. 10

Grupo No 4. Criterios de impacto.

9. Impacto en el área para la cual está destinada la guía.

Peso……. 9

10. Organización en el proceso de desarrollo.

Peso……. 10

Categoría final del procedimiento.

___ Excelente: Alta novedad científica, con aplicabilidad y resultados relevantes.

_x__ Bueno: Novedad científica, resultados destacados.

___ Regular: Suficientemente bueno con reservas.

___ Cuestionable: No tiene relevancia científica y los resultados son malos.

___ Malo: No aplicable.

Valoración final Sugerencias del experto para mejorar la calidad del proyecto.

Elementos críticos que deben mejorarse.

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.

Java Script: Lenguaje de programación interpretado.

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.

Website: Sitio web.