TESIS DE MAESTRÍA

Maestría en Ingeniería en Sistemas de Información

Título:
“MÉTODO DE INCLUSIÓN DE HACKING ÉTICO EN EL PROCESO DE TESTING DE SOFTWARE”

Autor: Lic. Ariel Orlando Giannone
Director de Tesis: Dr. Dario Rodriguez
Co Director de Tesis: Mg. Hernan Amatriain

Buenos Aires - 2018

AGRADECIMIENTOS

Gracias a mi madre y a mi padre por estar en los malos y buenos momento y no permitirme bajar los brazos.

A mi mujer Melanie por creer en mí y por el apoyo incondicional.

A Hernán por la paciencia y la ayuda para poder culminar este trabajo.

RESUMEN

Como ocurre con la mayoría de los avances tecnológicos, el crecimiento explosivo de Internet tiene un lado oscuro: los hackers. La escalada natural de amenazas ofensivas contra las medidas defensivas ha demostrado una y otra vez que no existen sistemas prácticos que se puedan construir que sean invulnerables a los ataques.

Las organizaciones informatizadas se dieron cuenta de que una de las mejores formas de evaluar la amenaza de intrusión sería tener profesionales independientes de seguridad informática intentando entrar en sus sistemas. Estos "hackers éticos" emplean las mismas herramientas y técnicas que los intrusos, pero sin dañar el sistema de destino ni robar información. En su lugar, permiten evaluar la seguridad de los sistemas e informar a los propietarios sobre las vulnerabilidades encontradas junto con las instrucciones de cómo remediarlos. Este proceso debe ser planificado con antelación. Todos los aspectos técnicos, de gestión y estratégicos deben estar sumamente cuidados. Estas etapas deben realizarse en un marco de control, gestión y supervisión constante. Es allí donde apunta este proyecto, poder incluir de manera segura y metódica la fase de revisión por hacking ético dentro del proceso de Testing de software.

ABSTRACT

As with most technological advances, the explosive growth of the Internet has a dark side: the hackers. The natural escalation of offensive threats against defensive measures has shown again and again that there are no practical systems that can be built that are invulnerable to attacks. The organizations realized that one of the best ways to assess the threat of intrusion would be to have independent IT security professionals trying to enter their systems. These "ethical hackers" employ the same tools and techniques as intruders, but without damaging the system or stealing information. Instead, they allow you to evaluate the security of the systems and inform the owners about the vulnerabilities found along with instructions on how to remedy them. This process must be planned in advance. All the technical, management and strategic aspects must be extremely careful. These stages must be carried out in a framework of constant control, management and supervision. This is where this project aims to be able to include in a safe and methodical way the review phase for ethical hacking within the Software Testing process.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

ÍNDICE DE FORMULAS

ABREVIATURAS

ASCII	Código de Caracteres
BCRA	Banco Central de la República Argentina
BD	Base de Datos
CAG	Cantidad de Accesos Ganados
CP	Casos de Prueba
CPE	Calendario de Pruebas y Esfuerzo
DNS	sistema de nombres de dominio
ESA	Agencia Espacial Europea
FAA	Autoridad Federal de Aviación de los Estados Unidos
FTP	File Transfer Protocol
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
IA	Informe de Amenazas
IAG	Informe de Accesos Ganados
IAM	Informe de Accesos Mantenidos
IAR	Informe de Arquitectura de Red
IAR	Índice de Acceso Real
IC	Informe de Criterios
ID	Informe de Dominio
IDP	Índice de Detección de Puertos
IDS	Sistema de Detección de Intrusos
IEEE	Institute of Electrical and Electronics Engineers
IES	Informe de Escaneo de Sistema
IFTHE	Informe Final de Testeo por Hacking Ético
IGRVV	Informe General de Riesgos, Validaciones y Verificaciones
IIS	Internet Information Server
IP	Internet Protocol
IPA	Índice de Posibilidad de Acceso
IPE	Informe de Pruebas Eliminadas
ISO	Organización Internacional de Normalización
ISTQB	International Software Testing Qualifications Board
IV	Informe de Vulnerabilidades
IVC	Informe de Vulnerabilidades Críticas
IVV	Informe de Verificación y Validación
KLOC	Kilo Lines Of Code
LCAR	Listado de Criterios de Aprobación y Rechazo
LDAP	Protocolo Ligero de Acceso a Directorios
OSI	Open System Interconnection
OVAL	Open Vulnerability and Assessment Language
OWASP	Open Web Application Security Project
PAD	Puertos Activos Detectados
PAR	Puertos Activos Reales
PC	Prioridades de Criticidad
PITHE	Plan Integral de Testeo por Hacking Ético
PMO	Oficina de Gestión de Proyectos
PSI	Política de Seguridad de la Información

SO	Sistema Operativo
SSL	Secure Sockets Layer
TCP	Protocolo de control de transmisión
UNICEF	Fondo de las Naciones Unidas para la Infancia
URL	Uniform Resource Locator
USB	Universal Serial Bus
VA	Vulnerability Assesment
XML	Extensible Markup Language

1. INTRODUCCION

En este Capítulo se plantea el contexto de la tesis (sección 1.1), se establecen sus objetivos (sección 1.2), y se resume la estructura de la tesis (sección 1.3).

1.1. CONTEXTO DE LA TESIS

El crecimiento explosivo de Internet ha traído muchas cosas buenas tales como el comercio electrónico, facilidad en el acceso a grandes cantidades de almacenamiento de material de referencia, computación colaborativa, e-mail, nuevas vías para la publicidad, información distribuida, por nombrar unos pocos. Como ocurre con la mayoría de los avances tecnológicos, también hay un lado oscuro: los hackers. Los gobiernos, las empresas, y ciudadanos privados de todo el mundo están ansiosos por ser parte de esta revolución, pero tienen miedo que algún intruso (hacker) irrumpa en su servidor Web y reemplace su logotipo con pornografía, lea su correo electrónico, robe su número de tarjeta de crédito de un sitio de compras en línea, o implante software encubierto para transmitir secretos de su organización a la Internet abierta. [Palmer, 2001] El software inseguro está debilitando las finanzas, salud, defensa, energía, y otras infraestructuras críticas. A medida que la infraestructura digital se hace cada vez más compleja e interconectada, la dificultad de lograr la seguridad en aplicaciones aumenta exponencialmente. [OWASP, 2013] La información para la organización es un activo que debe ser protegido del acceso no autorizado de personas y el mal uso de algunas otras cuestiones ilegales, la piratería es muy común en Internet y tiene afectado a la organización en términos de dinero, la pérdida de recursos y pérdida de imagen. [Sheoran & Singh , 2014]
Si la historia sirve de indicio, la comunidad de tecnología de la información fue incapaz de construir sistemas de información en red que pueden prevenir consistentemente ataques con éxito [Evans 2001]. La escalada natural de amenazas ofensivas contra las medidas defensivas ha demostrado una y otra vez que no hay sistemas prácticos que se puedan construir que sean invulnerables a los ataques. Incluso una organización tal como el Departamento de Defensa de EE.UU. en la red ha demostrado de forma continua el grado de susceptibilidad que posee ante los ataques.

Se supone que el factor principal que contribuye a la mala situación de la seguridad en Internet es la falta de pruebas de software de calidad. La complejidad intelectual asociada con el diseño de software, codificación, y prueba, prácticamente asegura la presencia de "errores" en

el software que puede ser explotada por atacantes. La mayoría del software hoy en día es la prueba de errores por el enfoque penetración-parche; cuando alguien encuentra una seguridad explotable "agujero" del fabricante de software emite un parche. Este enfoque ha demostrado ser insuficiente, ya que después de los hechos de seguridad deja abiertas las vulnerabilidades de errores hasta que sean explotados. Sin embargo, los fabricantes de software sostienen que este enfoque es económicamente atractivo, ¿por qué invertir tiempo y dinero en las pruebas de control si los consumidores no están dispuestos a pagar una prima por software seguro?. Otra variable es el tiempo de salida al mercado dicta que el software se libera en la forma más temprana como sea posible, a menudo con graves defectos no detectados de seguridad [Zimmerman, 2001]. El problema presentado por falta de pruebas de calidad también se agrava ante ataques automatizados, la homogeneidad del sistema operativo y las malas prácticas. [Schneier, 2000].

Algunos informáticos tienen la impresión equivocada de que su sitio Web no sería un objetivo citando numerosas razones, tales como "No tiene nada interesante en él" o "los hackers nunca han oído hablar de mi empresa". Lo que estas personas no se dan cuenta es que cada sitio web es un objetivo. El objetivo de muchos hackers es simple: hacer algo espectacular y luego asegurarse de que todos sus amigos sepan que hizo. Mismo, a muchos hackers simplemente no les importa que empresa u organización piratear, solo lo hacen porque pueden. Por ejemplo, los administradores Web de UNICEF (Naciones Unidas para la Infancia) podrían bien haber pensado que ningún pirata informático los atacaría. Sin embargo, en enero de 1998, su página fue completamente desfigurada. [Palmer, 2001].

Inicialmente las intrusiones en computadoras eran bastante inocentes, siendo el mayor daño el robo de tiempo de procesamiento. Otras veces, estos ataques serian en forma de bromas. Sin embargo, estas intrusiones no permanecieron así por mucho tiempo. En ocasiones, los intrusos menos talentosos, o menos cuidadosos, accidentalmente "bajaban" un sistema o dañaban sus archivos, y los administradores de sistemas tendrían que reiniciar o hacer reparaciones. Otras veces, cuando se descubrieron sus actividades y se les negó el acceso, los intrusos reaccionaron con acciones aun más destructivas. Cuando el número de estas intrusiones informáticas destructivas se hicieron "famosas", debido a la visibilidad del sistema o la magnitud del daño infligido, se convirtieron en noticias y los medios de comunicación recogieron estas historias. En lugar de usar el término preciso de "criminal informático", los medios de comunicación comenzaron a usar el término "Hacker" (*Sustantivo*. Persona que disfruta el aprendizaje de los detalles de los sistemas informáticos y cómo ampliar sus

capacidades, en contraste con la mayoría de los usuarios de computadoras, que prefieren aprender sólo el mínimo cantidad necesaria.) [Raymond, 1991] para referirse a las personas que irrumpen en los ordenadores para la diversión, la venganza, o su propia ganancia. Dado que llamar a alguien "hacker" fue originalmente concebido como un cumplido, profesionales de la seguridad informática prefieren utilizar el término "cracker" o "intruso" para los hackers que giran hacia el lado oscuro de la informática, la piratería. [Palmer, 2001].

En general, las políticas de seguridad de la información o los controles por sí solos no garantizan la protección total de la información, ni de los sistemas de información, servicios o redes. Después de los controles que se han implementado, vulnerabilidades residuales probablemente permanezcan haciendo ineficaz la seguridad de la información y por lo tanto los incidentes son aun mas posibles. Esto puede llegar a tener efectos negativos tanto directos e indirectos sobre las operaciones de negocio de una organización. Además, es inevitable que se produzcan nuevos casos de amenazas no identificadas previamente. Una preparación insuficiente por una organización para hacer frente a este tipo de incidentes hará cualquier respuesta menos efectiva, y aumentar así el grado de impacto comercial potencial adverso. [ISO/IEC 27035:2011]

1.2. OBJETIVOS DE LA TESIS

En esta sección se introduce el objetivo general de la tesis (sección 1.2.1) y los objetivos específicos derivados de la misma (sección 1.2.2).

1.2.1. OBJETIVO GENERAL DE LA TESIS

El presente trabajo de investigación propone la incorporación del método de hacking ético para la evaluación de vulnerabilidades dentro del procedimiento mismo de Testeo de un sistema. Se intenta, de esta manera, aportar a los encargados de testing en sectores de Seguridad Informática de un grupo de actividades, herramientas sugeridas y forma de actualizaciones de las mismas que les brinde el soporte necesario para poder prevenir los problemas que en la actualidad son de creciente interés por las pérdidas económicas que conllevan.

El método propuesto para la inclusión del hacking ético, se limita a contemplar las posibles intrusiones que pueden llevarse a cabo en el entorno de los sistemas civiles tanto de uso

comercial como privado. Esto significa que no se incluye dentro de la propuesta la evaluación de aspectos relacionados con sistemas críticos como gubernamentales, militares, etc.

1.2.2. OBJETIVOS ESPECIFICOS DE LA TESIS

- Identificar y describir las falencias que existen en el proceso actual de testeo sobre las vulnerabilidades de hackeo.

- Detallar métodos y sugerir herramientas que ayudaran en la detección de vulnerabilidades en el proceso de testeo.

- Efectuar una prueba comparativa de concepto para demostrar la validez de su aplicación.

1.3. VISIÓN GENERAL DE LA TESIS

En el Capítulo Introducción se plantea el contexto de la tesis, se establecen sus objetivos y se resume la estructura de la tesis.

En el Capítulo Estado del Arte se presentan distintas teorías y técnicas que son concurrentes con los objetivos de esta tesis. Se presenta la teoría de testeo de software donde se describen los diferentes tipos de prueba que existen en el testeo de software, el proceso de testeo en los diferentes modelos de software y las diferentes técnicas de pruebas. Se exponen los conceptos de seguridad informática tales como clasificación de ataques, amenazas, vulnerabilidades, políticas de seguridad, buenas prácticas y evaluación de la seguridad. Por ultimo se presenta la teoría del hacking ético, donde se exponen entre otros detalles el perfil de conocimientos, los niveles de ataques, la evaluación de seguridad anti hackeo y las etapas del ciclo de hacking.

En el Capítulo Descripción del Problema se presenta la importancia que tiene la actividad de testeo con hacking ético, se caracteriza al profesional de seguridad informática y la forma en que se vincula la prueba de software con el testeo de vulnerabilidades, se aborda el problema de investigación a partir de la desorganización e informalidad en la aplicación de herramientas y métodos de hacking ético, se caracteriza el problema abierto y se concluye con un sumario de investigación donde se exponen las preguntas sin respuestas abordadas.

En el Capítulo Solución se presenta: un modelo de proceso de aplicación técnica de hacking ético, donde se emprenden las cuestiones generales de mayor relevancia, se presenta la propuesta de dicho modelo y se describe a partir de su estructura general, detallando las 3 fases que componen el método (fase de planificación, fase de ejecución y fase de mantenimiento).
Junto con las fases, se describen las etapas y actividades que deben realizarse para concretarlas, detallando minuciosamente los insumos necesarios y productos que se obtienen con la implementación de cada etapa. Por último se detallan las métricas a utilizar para poder mensurar la aplicación del método en cuestión.

Dentro del capítulo Caso de Validación 1, se presenta una aplicación de tipo web para la aplicación de las técnicas asociadas a las tareas del modelo de proceso de testeo por hacking ético, a los efectos de implementar las tareas correspondientes a cada una de las fases. Se analiza un caso correspondiente a un sistema de Gestión de archivos.

En el capitulo Caso de Validación 2, se pone en ejecución el método sobre una aplicación web del tipo blog.
En los últimos dos capítulos descriptos, se mencionan una serie de herramientas, las cuales son las más importantes a nivel de uso y performance al momento de escribir esta tesis, sin embargo el proceso debe trascender a las herramientas.

En el Capítulo Conclusiones se presentan las aportaciones de este trabajo y se destacan las futuras líneas de investigación que se consideran de interés en base al problema abierto que se presenta en este trabajo de tesis.

Dentro del capítulo Referencias bibliográficas, se exponen todas referencias que sirvieron como base para este trabajo de tesis.

En el capitulo Publicaciones realizadas se presentan las publicaciones del tesista vinculadas a las investigaciones realizadas en el desarrollo de la tesis y se resume la estructura de la tesis.
Y por ultimo en el capitulo Anexos, se adiciona toda la documentación obtenida en los casos de validación ejecutados.

2. ESTADO DE LA CUESTIÓN

En este capítulo se presenta el estado de la cuestión sobre distintas teorías y técnicas que son concurrentes con los objetivos de esta tesis. Describiendo la teoría de Testeo de Software (sección 2.1), bases teóricas de la Seguridad Informática (sección 2.2) y la teoría del Hacking Ético (sección 2.3).

2.1. TESTEO DE SOFTWARE

2.1.1. ¿Qué es una prueba?

Existe una gran diferencia entre la percepción sobre lo que se entiende como prueba y lo que realmente se busca al probar un sistema.
Los desarrolladores siguen definiciones tales como [Myers, 2004]:

- "El propósito de las pruebas es demostrar que un sistema realiza las funciones indicadas correctamente".
- "Las pruebas son el proceso de demostrar que no hay errores presentes"

Ahora bien, Myers expresa que estas afirmaciones están mal planteadas, dado que cuando se prueba un sistema se quiere aportar un valor añadido a lo que estamos probando, elevar la calidad y fiabilidad y esto nos lleva a tener que encontrar y eliminar los errores en el sistema, sosteniendo todo esto en la siguiente frase:

"La prueba es el proceso de ejecución de un sistema con la intención de encontrar errores" [Myers, 2004]

Esto quiere decir que no se debe probar un sistema para demostrar que funciona, sino que es esencial partir de la suposición de que el sistema contendrá errores.

Esta forma de dirigir las pruebas, nos evitará caer en la selección de casos de prueba con baja probabilidad de causar que el sistema falle, ya que si nuestro principal objetivo es demostrar que el sistema tiene fallos, existirá una mayor posibilidad de encontrar errores.

Otros autores y hasta organizaciones apoyan este lineamiento de ideas con las siguientes definiciones:

"Las pruebas de software pueden ser una manera muy eficaz de mostrar la presencia de errores, pero son totalmente inadecuadas para mostrar su ausencia." [Dijstra, 1972]

"El proceso que consiste en todas las actividades del ciclo de vida, tanto estáticas como dinámicas relacionadas con la planificación, preparación y evaluación de productos de software y productos relacionados con el trabajo para determinar que cumplen los requisitos especificados, para demostrar que son aptos para el propósito y para detectar defectos" [ISTQB, 2002]

En todas estas definiciones hay una clara tendencia que demuestra que el proceso de testing se centra en mayor o menor manera en la detección de errores.

Ahora bien, para terminar de comprender a lo que se refiere con la detección de errores, debemos revisar la diferencia existente entre tres palabras, que a menudo se entienden como sinónimos, pero que indican conceptos muy diferentes. Estas palabras son "Error", "Defecto" y "Fallo".

Un ejemplo trae muy buena claridad sobre estos conceptos relacionados entre sí, pero con significados totalmente diferentes:

"Un error de programación se puede dar al momento en que el desarrollador asigna 2 valores a una misma variable, o cometido en la lógica de programación. En el momento en que se compila el código, se arma la versión y se instala en un ambiente, ese software contiene defectos, ¿Cuáles?, no lo sabemos si no hasta ejecutar nuestras pruebas, en el momento en que el sistema falla, se manifiesta mediante un mensaje de error el cual capturamos para reportar un fallo"

Ahondando en la definición de cada término se puede determinar la diferencia a la que se hace referencia.

Error: Acción humana que produce un resultado incorrecto [ISTQB, 2002]

Defecto: Una imperfección o deficiencia en un producto el cual no cumple con sus requisitos o especificaciones. [ISTQB, 2002]

Fallo: Un evento en el que un componente o sistema no realiza una función requerida dentro de los límites especificados. [ISTQB, 2002]

2.1.2. ¿Por qué son importantes las pruebas?

Habiendo expuesto lo que significa la prueba de software, la siguiente pregunta que aparece es, ¿para qué son importantes pruebas?.

En la sociedad en la que vivimos, estamos sometidos diariamente a una multitud de sistemas en múltiples dispositivos tales como, coches, smartphones, computadoras, etcétera. Y a los que día a día, gracias al llamado internet de las cosas, se agregan tales como lavarropas, aspiradoras y demás dispositivos cotidianos. Sin dejar de lado el avance tecnológico digital producido en materia de salud y de servicios.
Como ejemplo y a modo de respuesta a la importancia de las pruebas, se narra el conocido suceso del lanzamiento del Ariane 5, considerado como el fallo de programación más caro de la historia.

El 4 de junio de 1996 el primer Ariane 5 tenía que haber puesto en órbita las cuatro sondas de la misión Cluster, pero en lugar de eso explotó a los 37 segundos de despegar. Según el informe de la Agencia Espacial Europea (ESA), el fallo fue debido a que el Ariane 5G perdió de forma completa la guía e información de orientación. Esta pérdida de información se debió a que usaba el mismo sistema de navegación inercial que el Ariane 4 pero nadie había tenido en cuenta que la mayor potencia del Ariane 5 iba a hacer que al poco de despegar adquiriera mucha más velocidad horizontal que el 4 y que esto podía afectar al funcionamiento del sistema guía. [Dalmau & Gigou, 1997]
En esos casi 40 segundos fatales la pérdida fue de más de 500 millones de dólares, y las posteriores conclusiones dejaron ver que con un correcto análisis y pruebas, se podría haber evitado el fallo.

Ahora bien si irnos a más de 20 años atrás parece mucho, hace apenas 3 años en el año 2015, la Autoridad Federal de Aviación de los Estados Unidos (FAA) publicó una orden de efecto

inmediato que obligaba a apagar y encender todos los Boeing 787 matriculados en los Estados Unidos al menos una vez cada 120 días.
Esto es porque la propia Boeing detectó un error en la programación de las unidades de control de los cuatro generadores eléctricos principales que poseen estas aeronaves, que hace que si estas permanecen encendidas durante 248 días entren en un modo de protección contra fallos que pararía los cuatro generadores.
Con los cuatro generadores parados los pilotos podrían perder el control del avión, pues este fallo se puede producir en cualquier fase del vuelo.

Esto no hace más que demostrarnos que el no probar adecuadamente el software que antes podía acarrear un problema económico mínimo, en nuestros tiempos puede producir desde pérdidas gigantes de dinero, daños irreparables de información, lesiones físicas personales y hasta eventos catastróficos.

2.1.3. ¿Cuál es el objetivo de las pruebas?

Ahora bien, las pruebas de sistemas buscan estos errores con un único y, más que importante, fin como es el aporte de calidad al software desarrollado.

Una de las primeras definiciones aseguraba que "la calidad de un programa o sistema se evaluaba de acuerdo al número de defectos por cada mil líneas de código. (KLOC: Kilo Lines Of Code)". [Pressman, 1993]
Este concepto fue cambiando a medida que fueron madurando los métodos y herramientas, hasta llegar a definiciones tales como las del ISTQB, la IEEE y las normas ISO.
Para el ISTQB la calidad del software es la siguiente:
"El grado en que un componente, sistema o proceso cumple con los requisitos especificados y/o las necesidades y expectativas del usuario/cliente." [ISTQB, 2002]

La definición de la calidad del software según la IEEE, Std. 610-1990, es:
"El grado con el que un sistema, componente o proceso cumple los requerimientos especificados y las necesidades o expectativas del cliente o usuario" [IEEE, 1990]

Y por otro lado la International Standards Organization, ISO en la norma 8402:1994, la define como:

“La totalidad de propiedades y características de un producto, proceso o servicio que le confiere su aptitud para satisfacer unas necesidades expresadas o implícitas.” [ISO, 1994]
En la actualización de esta norma, la 9000:2000, la definición se actualizó dejando en curso lo siguiente:
“Grado en el que un conjunto de características inherentes cumple con los requisitos”. [ISO, 2000]

En conclusión en todas las definiciones existen 2 puntos principales, el cumplimiento de los requisitos del producto y la satisfacción de las necesidades del cliente.

Sobre este último punto la norma ISO/IEC 25010 dice que la calidad del producto software se puede interpretar como el grado en que dicho producto satisface los requisitos de sus usuarios aportando de esta manera un valor. Son precisamente estos requisitos (funcionalidad, rendimiento, seguridad, portabilidad, etc.) los que se encuentran representados en el modelo de calidad, el cual categoriza la calidad del producto en características y sub características.

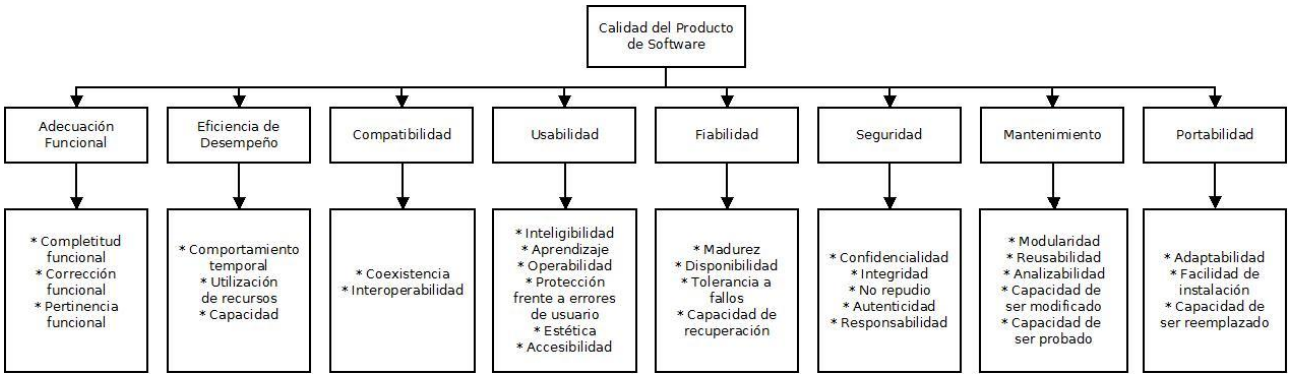

Figura 1. Modelo de calidad de producto de software.

2.1.4. Principios fundamentales de las pruebas

En la búsqueda de lograr la máxima calidad en los productos de software desarrollados, gracias a experiencias, se plantearon siete principios, también llamados los 7 mandamientos del testing, los cuales deben intentar aplicarse siempre para conseguir la tan ansiada satisfacción del cliente.

Aquí se detallan los siete principios con sus respectivas explicaciones según el [ISTQB, 2002].

- Principio 1: Las pruebas demuestran la presencia de errores

La prueba puede mostrar que los defectos están presentes, pero no puede probar que no hay defectos. La prueba reduce la probabilidad de los defectos no descubiertos restantes en el software pero, incluso si no se encuentran defectos, no es una prueba de corrección

- Principio 2: Es imposible probarlo todo

Probar todo, como todas las combinaciones de entradas y de precondiciones, no es factible a excepción de casos triviales. En vez de la prueba exhaustiva, se utiliza el riesgo y las prioridades para enfocar los esfuerzos de prueba.

- Principio 3: Cuanto antes se comience a probar mejor

Las actividades de prueba deberían comenzar tan pronto como sea posible en el ciclo de vida del desarrollo del software o del sistema y deberían estar enfocados en los objetivos definidos.

- Principio 4: Agrupamiento de defectos

Se concentran en determinados puntos de un sistema software, no manteniendo una distribución uniforme a lo largo del mismo.

Una posible estrategia puede ser centrarse en mejorar las pruebas de aquellos componentes para los que se han reportado un número mayor de errores, para ser más eficaces a la hora de detectarlos en fases tempranas.

- Principio 5: La paradoja del pesticida

Si las mismas pruebas se repiten una y otra vez, eventualmente el mismo conjunto de casos de prueba, no se encontrarán nuevos errores (bugs). Para superar esta "paradoja del pesticida", los casos de prueba necesitan ser repasados y revisados regularmente, armando nuevas y diversas pruebas para someter diferente partes del software para encontrar más defectos.

- Principio 6: Las pruebas son dependientes del contexto

Las pruebas se hacen diferenciadamente en diversos contextos. Si el producto se centra en el ámbito de la seguridad se deberán adaptar los casos de prueba para intentar forzar situaciones o posibles escenarios no amistosos.

Además, hay que tener en cuenta que los recursos en los proyectos son siempre escasos, con lo que en el inicio del proyecto hay que plantearse qué estrategia debemos seguir para

encontrar y corregir lo antes posible los errores en las funcionalidades de mayor valor para nuestros usuarios.

- Principio 7: La falacia de la ausencia de errores

Encontrar y arreglar defectos no ayuda si el sistema construido es inutilizable y no cumple las necesidades y expectativas de los usuarios.

2.1.5. Tipos de prueba

Existen diferentes tipos de prueba de software. Las que buscan probar funcionalidades del software, las que buscan probar características no funcionales, como puede ser la carga, usabilidad, etc.

Por otro lado se encuentran aquellas pruebas que buscan probar la estructura o arquitectura del software, las que se aplican sobre errores corregidos para corroborar la solución y las pruebas que se aplican de modo preventivo o correctivo.

Sobre estas diferencias, las pruebas se pueden dividir en cinco grandes grupos:

- Pruebas funcionales

La prueba funcional es conocida también como basado en la especificación o de caja blanca, ya que tienen en cuenta el comportamiento externo del software.

El objetivo de la prueba funcional es validar cuando el comportamiento observado del software probado cumple o no con sus especificaciones. La prueba funcional toma el punto de vista del usuario [Beizer, 1990].

Según la ISO 25000, la adecuación funcional se subdivide en las siguientes características:

  - Completitud funcional. Grado en el cual el conjunto de funcionalidades cubre todas las tareas y los objetivos del usuario especificados.
  - Corrección funcional. Capacidad del producto o sistema para proveer resultados correctos con el nivel de precisión requerido.
  - Pertinencia funcional. Capacidad del producto software para proporcionar un conjunto apropiado de funciones para tareas y objetivos de usuario especificados.

  Además de las pruebas sobre los módulos y funciones, pueden realizarse pruebas en áreas especializadas como Pruebas de Seguridad y Pruebas de Interoperabilidad.

- Pruebas no funcionales

Este tipo de pruebas tienen en cuenta el comportamiento externo del software, es decir cómo funciona el sistema, y se realiza mediante técnicas de diseño de caja negra. Al igual que las características funcionales, las características no funcionales tienen que estar definidas en las especificaciones del producto.

Las pruebas no funcionales pueden ser divididas por sus características:

- Pruebas de carga: consiste en la medición del comportamiento del sistema para aumentar la carga del mismo, ya sea mediante el número de peticiones que se realizan al mismo tiempo, el número de usuarios que trabajan simultáneamente, etc.
- Pruebas de rendimiento: en estas pruebas se medirán la velocidad de procesamiento y el tiempo de respuesta del sistema.
- Pruebas de volumen: se mide la capacidad del sistema para procesar gran cantidad de datos, como procesar archivos con tamaños muy grandes.
- Pruebas de esfuerzo: se realizan pruebas donde se sobrecarga el sistema y se analiza la capacidad de recuperación.
- Pruebas de seguridad: se realizan diferentes pruebas de accesos no autorizados, ataque de denegación de servicio, etc.
- Pruebas de estabilidad, eficiencia, robustez: se realiza una medición de la respuesta del sistema a los errores de funcionamiento.
- Pruebas de compatibilidad: son pruebas del funcionamiento del sistema con los diferentes sistemas operativos, plataformas de hardware, etc., con los que puede interactuar el programa.
- Pruebas de usabilidad: se mide la facilidad de uso, efectividad y satisfacción, siempre dentro de un grupo específico de usuarios.

- Pruebas estructurales

Las Pruebas Estructurales es el término usado por ISTQB para las pruebas de “Caja Blanca”. Estas se realizan aplicando técnicas de pruebas estructurales y técnicas estáticas, en lugar de técnicas basadas en especificación.

Aparece el concepto de “Cobertura” para definir la extensión con la cual la estructura ha sido cubierta por el conjunto de pruebas, expresado como un porcentaje del elemento probado, donde si la cobertura no es del 100%, se pueden diseñar pruebas adicionales.

- Pruebas de regresión

Las pruebas de regresión son aplicadas luego de que un defecto es identificado y corregido, con la finalidad de verificar que el defecto ya no se presenta, para esto es sumamente necesario tener claridad de las piezas de software que resultan afectadas por el cambio.

Deben buscarse nuevos defectos tanto en el componente que se está probando cómo otros componentes afectados por el cambio y en estas se incluyen pruebas funcionales, no funcionales y estructurales.

El mayor desafío se encuentra en que las pruebas deben ser repetibles si han de usarse para pruebas de confirmación y regresión.

Dado repetición de estas pruebas, son candidatas a la automatización de pruebas por medio de herramientas.

- Pruebas de mantenimiento

Este tipo de prueba se aplica sobre sistemas que están operativos en ambiente de producción y teniendo como disparadores modificaciones, migraciones o desincorporación de software.

En estas se incluyen mejoras planificadas, correctivas o de emergencia, así como cambios en el entorno de sistema operativo, bases de datos, actualizaciones o parches.

Las pruebas de mantenimiento pueden ser divididas por sus características:

- Pruebas de Migración: incluyen pruebas operativas del nuevo entorno (Sistema operativo, base de datos, etc.) así como pruebas sobre el software modificado. Ante alguna migración o conversión de datos, también serán necesarias pruebas sobre estos.
- Pruebas por Desincorporación: incluyen pruebas de migración de datos o su archivo si se requieren largos períodos de retención.
- Pruebas de regresión: sobre las partes del sistema que no se están cambiando.
  Este tipo de pruebas suelen ser difíciles de realizar si las especificaciones están desactualizadas o no existen, o si no se cuenta con Testers con conocimiento del sistema.

2.1.6. Proceso fundamental del testing

Habiendo revisado los 7 mandamientos del testing, y para lograr la máxima eficiencia y eficacia, los planes deben incluir la planificación de las pruebas, el análisis y diseño de los casos de

prueba, la preparación de la ejecución y la evaluación final del estado. Para este orden se definió el llamado proceso fundamental, el cual está dividido en 5 etapas, las cuales poseen diferentes actividades para ser cumplimentadas.

Donde las etapas son las siguientes:

- Planificación y Control

La planificación de pruebas es la tarea de definir los objetivos de las pruebas y la especificación de las actividades de pruebas con el objetivo de cumplir los objetivos y misión establecidos. El Control de las pruebas es la actividad permanente de comparar el progreso real contra el plan, y comunicar el estado actual de las pruebas, incluyendo las desviaciones del plan. Implica tomar las medidas necesarias para cumplir la misión y los objetivos del proyecto.

La tarea de planificación de prueba tienen las siguientes actividades principales:

- Identificar los objetivos de la prueba.
- Determinar el alcance y los riesgos.
- Determinar la cobertura de la prueba.
- Determinar los recursos requeridos.
- Establecer la política de prueba y estrategia de prueba.
- Programar la implementación, ejecución y evaluación de prueba.
- Determinar los criterios de salida.

Las tareas de control de prueba tienen las siguientes actividades:

- Medir y analizar los resultados.
- Monitorear, documentar y comunicar el progreso, la cobertura de prueba y los criterios de salida.

- Análisis y Diseño

El análisis y diseño de pruebas es la tarea durante la cual los objetivos de las pruebas generales se transforman en condiciones de prueba y casos de prueba tangibles.

Teniendo las siguientes tareas a realizar:

- Revisar la documentación base de la prueba a realizar.
- Identificar y priorizar las condiciones de prueba.

- Diseñar las pruebas.
- Identificar los datos necesario para la prueba
- Evaluar el índice de testeo de los requisitos y del sistema.
- Identificar y configurar el entorno de prueba junto a la infraestructura y herramientas requeridas.

- Implementación y Ejecución

La implementación y ejecución de pruebas es la actividad donde las condiciones de prueba son transformadas en casos de prueba y el entorno es configurado.

Se recomiendan las siguientes actividades en la etapa de implementación:

- Desarrollar y priorizar casos de prueba, crear datos de prueba, escribir procedimientos de prueba y, opcionalmente, preparar scripts de prueba automatizados.
- Crear conjuntos de pruebas de los casos de prueba para la ejecución de la prueba eficientemente. Un conjunto de pruebas es una colección lógica de casos de prueba que, naturalmente, trabajan juntos. Los conjuntos de pruebas a menudo comparten datos y un alto nivel común de un conjunto de objetivos.
- Establecer un calendario de ejecución de las pruebas.
- Implementar y verificar el ambiente.

Por otro lado la ejecución de las pruebas tiene las siguientes tareas principales:

- Ejecutar los bancos de pruebas y casos de prueba individuales, siguiendo los procedimientos de pruebas.
- Registrar el resultado de la ejecución de pruebas y registrar la identidad y las versiones del software en las herramientas de pruebas.
- Comparar los resultados reales con los resultados esperados.

- Evaluación de criterios de salida y reporte

Evaluar los criterios de salida es la actividad donde la ejecución de la prueba es evaluada contra los objetivos definidos.

Esta evaluación de criterios tiene las siguientes tareas principales:

- Comprobar los registros de prueba contra los criterios de salida especificados en la planificación de prueba.

- Evaluar si más pruebas son necesitadas o si los criterios de salida especificados deberían ser cambiados.
- Escribir un reporte de resumen de prueba para las partes interesadas.

- Actividades de cierre de pruebas

Las actividades de cierre de prueba recolectan datos de las actividades de prueba completadas para consolidar experiencia, hechos y números.

Las actividades de cierre de prueba incluyen las siguientes tareas principales:

- Comprobar cuales entregables planeados han sido entregados, el cierre de los informes de incidentes o el aumento de registros de cambio para cualquiera que permanezca abierto y la documentación de la aceptación del sistema.
- Finalizar y archivar los casos, el entorno y la infraestructura de prueba para reutilización posterior.
- Entrega de los reportes a la organización de mantenimiento.
- Analizar las lecciones aprendidas para futuros lanzamientos y proyectos y la mejora de la madurez de pruebas.

2.1.7. La prueba en los modelos de desarrollo de software

El proceso de prueba de software no existe en forma aislada, sino que estas actividades se encuentran inmersas en el ciclo de vida de desarrollo de software. Los diferentes modelos de ciclo de vida de desarrollo necesitan diferentes enfoques hacia la prueba.

- Modelo en Cascada

Una de las metodologías más antiguas en lo que es el ciclo de vida de un modelo informático, es el modelo de cascada. Esta metodología es lineal y consta de algunas fases que hay que seguir y completar para poder avanzar a la fase siguiente. No es precisamente la mejor metodología, pero si se utiliza de forma correcta los resultados pueden ser muy buenos.

Está compuesta por las siguientes fases:

- Requerimientos
- Diseño
- Desarrollo
- Integración

- **Pruebas**
- Instalación
- Mantenimiento

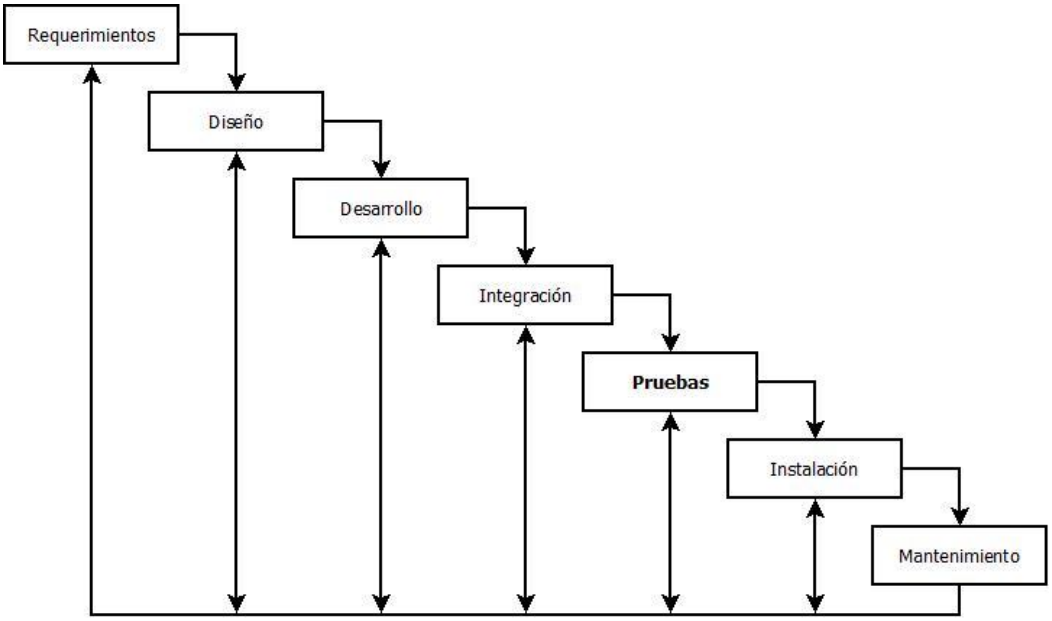

Figura 2. Fases y flujo del modelo de desarrollo en cascada.

El ciclo de vida de un software realizado bajo esta metodología, es extenso y muy estructurado. Una fase no comienza hasta que termine la fase anterior y generalmente se incluye la corrección de los problemas encontrados en fases previas.

En décadas pasadas, las críticas al mismo han ocasionado que incluso sus más fieles seguidores hayan puesto en duda su eficiencia, pues para trabajar con un modelo en cascada, es necesario tener en cuenta algunos aspectos que de alguna forma se constituyen en sus desventajas. [Pressman, 2005]

- Modelo incremental

El modelo incremental permite una secuencia no lineal de los pasos de desarrollo. Aplica secuencias lineales de forma escalonada, mientras progresa el tiempo en el calendario. Cada secuencia lineal produce un "incremento" del software. En el modelo incremental se va creando el sistema de software y se van añadiendo componentes funcionales al sistema. En cada paso sucesivo se actualiza el sistema con nuevas funcionalidades o requisitos, es decir,

cada versión o refinamiento parte de una versión previa y le añade nuevas funciones. [Piattini, 2000]

Cada ciclo de desarrollo (incremento) se divide en seis fases:

- Requerimientos
- Diseño
- Desarrollo
- **Pruebas**
- Instalación
- Mantenimiento

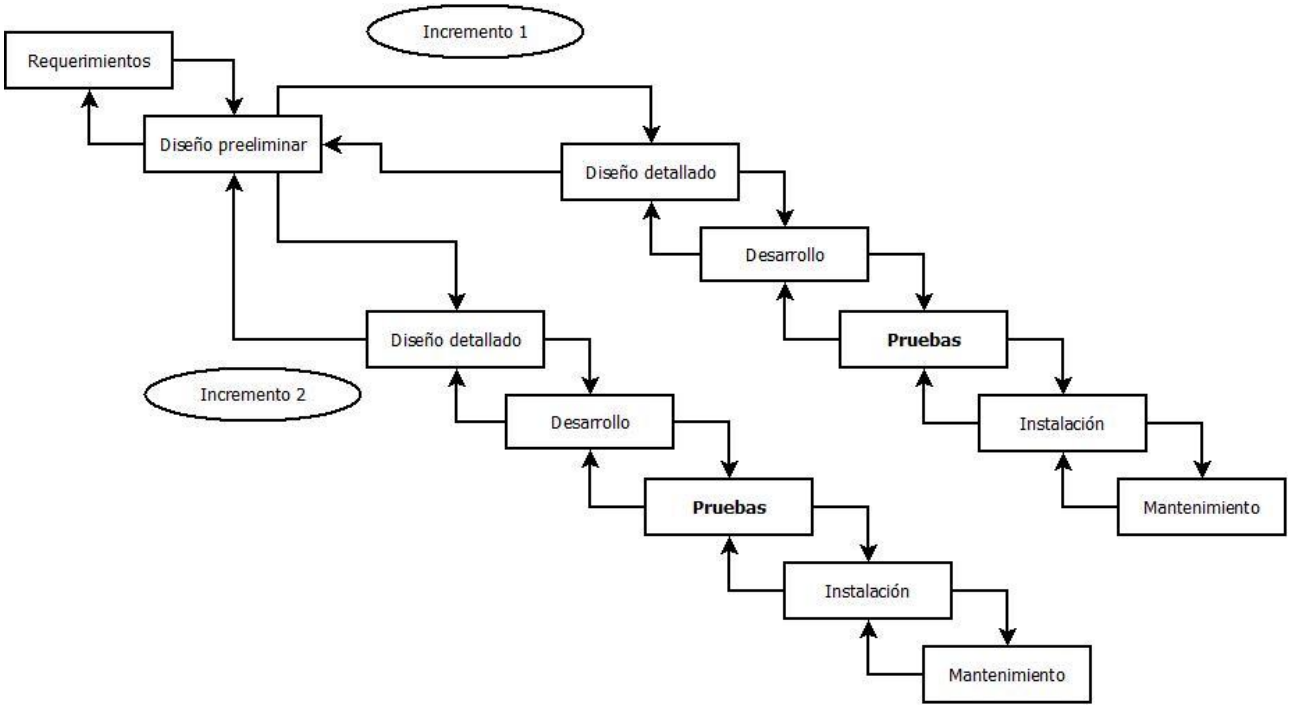

Figura 3. Fases y flujo del modelo de desarrollo incremental.

El modelo incremental se ajusta a entornos de alta incertidumbre, por no tener la necesidad de poseer un conjunto exhaustivo de requisitos o especificaciones.

- Modelo en Espiral

El modelo de desarrollo en espiral propuesto por [Boehm, 1988]. Se representa como un espiral, en lugar de una serie de actividades sucesivas con retrospectiva de una actividad a otra.

Puede verse como un modelo evolutivo que conjuga la naturaleza iterativa del modelo incremental con los aspectos controlados y sistemáticos del Modelo Cascada, con el agregado de gestión de riesgo.

Cada ciclo de desarrollo se divide en cuatro fases:

- Definición de objetivos: Se definen los objetivos. Se definen las restricciones del proceso y del producto. Se realiza un diseño detallado del plan administrativo. Se identifican los riesgos y se elaboran estrategias alternativas dependiendo de estos.
- Análisis de riesgos: Se realiza un análisis detallado de cada riesgo identificado. Pueden desarrollarse prototipos para disminuir el riesgo de requisitos dudosos. Se llevan a cabo los pasos para reducir los riesgos.
- Desarrollo y Pruebas: Se escoge el modelo de desarrollo después de la evaluación del riesgo. El modelo que se utilizará (cascada, sistemas formales, evolutivo, etc.) depende del riesgo identificado para esa fase.
- Planificación: Se determina si continuar con otro ciclo. Se planea la siguiente fase del proyecto.

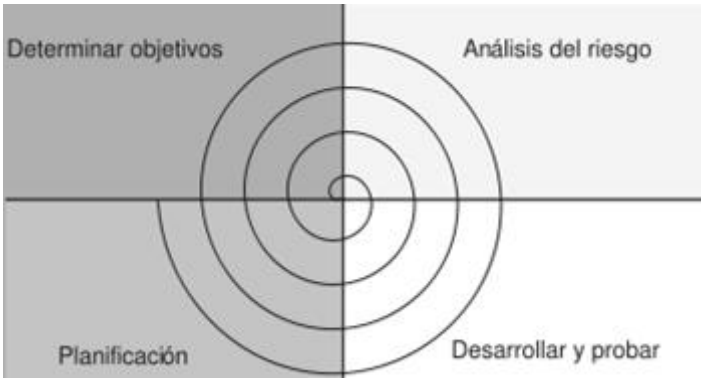

Figura 4. Flujo del modelo de desarrollo en espiral.

Este modelo a diferencia de los otros toma en consideración explícitamente el riesgo, esta es una actividad importante en la administración del proyecto.

- Modelo Ágil

Los métodos ágiles nacen a principios de la década de los 90 en contraposición a lo que representaban los métodos tradicionales. Esta explosión de metodologías llevó a que, en febrero del 2001, tras una reunión celebrada en Utah, USA, se acuñara formalmente el término “ágil” aplicado al desarrollo de software. En esta misma reunión participó un grupo

de 17 expertos de la industria del software, incluyendo algunos de los creadores o impulsores de metodologías de software, con el objetivo de esbozar los valores y principios que deberían permitir a los equipos desarrollar software rápidamente y respondiendo a los cambios que pudieran surgir a lo largo del proyecto.

Método	Acrónimo	Autor	Año
Microsoft solutions framework	MSF	Microsoft	1994
Scrum	Scrum	Sutherland	1994
Rapid development	RAD	McConnell	1996
Dynamic solutions delivery model	DSDM	Stapleton	1997
Cristal methods	CM	Cockbum	1998
Agile RUP	dX	Booch, Martin, Newkirk	1998
eXtreme Programming	XP	Beck	1999
Adaptive software development	ASD	Highsmith	2000
Feature-driven development	FDD	Charett, Poppendieck	2001
Agile modeling	AM	Ambler	2002

Tabla 1. Diferentes lineamientos de Metodología agiles.

El 2001, en Utah, USA, se crea el Manifiesto ágil [Agile, 2014], documento firmado por los principales exponentes de la corriente de desarrollo ágil a nivel mundial, marcando un hito en la historia de este tipo de métodos. Estos métodos se centran en otras dimensiones, distintas que en los métodos tradicionales, como por ejemplo el factor humano o el producto software.

Esta es la filosofía de los métodos ágiles, las que dan mayor valor al individuo, a la colaboración con el cliente y al desarrollo incremental del software con iteraciones muy cortas. Este enfoque está mostrando su efectividad en proyectos con requisitos muy cambiantes y cuando se exige reducir drásticamente los tiempos de desarrollo pero manteniendo una alta calidad. [Leiva & Villalobos, 2015]

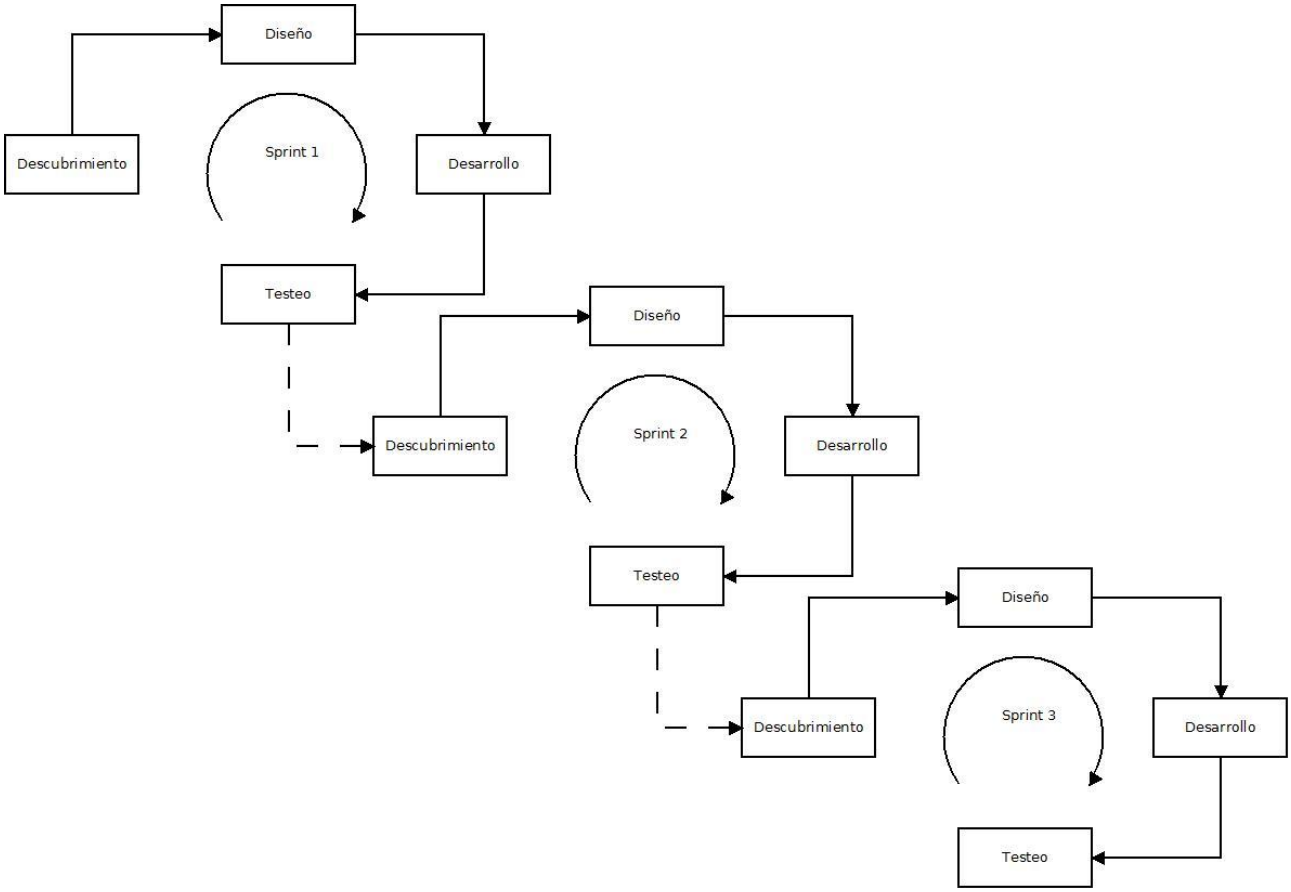

Figura 5. Flujo resumido del modelo de desarrollo mediante metodologías agiles.

2.1.8. Niveles de pruebas

Con mucha facilidad suelen confundirse los niveles de pruebas con los tipos de prueba, y a pesar de que se encuentren íntimamente relacionadas, tienen connotaciones diferentes en el proceso.
Para cada instancia del desarrollo, independientemente del tipo de prueba que se seleccione, se define un nivel o, también llamada, estrategia de prueba, donde debe asegurarse la verificación y validación de la pieza de software. [Pressman, 2005]
Antes de continuar es importante detallar que se entiende como verificación y validación de software.

El proceso de validación y verificación es un conjunto de procedimientos, actividades, técnicas y herramientas que se utilizan paralelamente al desarrollo de software, con el fin asegurar que un producto resuelve el problema inicialmente planteado. [Sommerville, 2005]
La verificación comprueba la consistencia del software con respecto a especificaciones y requisitos, es decir, intenta responder a la pregunta: ¿se ha construido correctamente el software? [Bohem, 1979]
Ayuda a determinar si los productos resultantes de una fase del ciclo de vida software cumplen los requisitos establecidos en la fase anterior y si el producto resultante es completo, consistente y correcto para comenzar la siguiente fase. [Pressman, 2005]

La validación, por su parte, tiene como objetivo comprobar si lo que se ha especificado e implementado es lo que el usuario realmente desea, es decir, si responde a la pregunta: ¿se ha construido el software correcto? [Bohem, 1979]
Buscando determinar si el software cumple su especificación y si se comporta como lo espera el cliente de acuerdo a sus expectativas. [Pressman, 2005]

Ahora bien se pueden seleccionar diferentes estrategias de pruebas respecto al desarrollo que se ha llevado a cabo, aquí se detallan los niveles mas conocidos y utilizados.

- Prueba unitaria

  Este puede llamarse el primer nivel de las pruebas, también llamadas prueba de componentes.
  Consiste en la verificación de módulos del software de forma aislada, es decir, probar el correcto funcionamiento de una unidad funcional de código.
  La unidad funcional de código es una porción de programa, como una función o método de una clase que es invocada desde fuera de la unidad y que puede invocar otras unidades. Es por ello que hay que probar que cada unidad funcione separada de las demás unidades de código [Pressman, 2005].
  Estas pruebas suelen ser realizadas por los desarrolladores, ya que es recomendable conocer el código fuente del programa y generalmente se analizará el código para comprobar que cumple con las especificaciones del componente.
  Sin embargo, no se persigue realizar solamente pruebas de la estructura del código sino que también se generarán casos de prueba funcionales para comprobar el funcionamiento completo del componente.

Para esto, el bloque puesto a prueba debe implementar una función independiente simple y puede ser probado al cien por cien por separado.

- Prueba de integración

Es el proceso en el cual los componentes son agregados para crear componentes más grandes. Se busca mostrar que aunque los componentes hayan pasado satisfactoriamente las pruebas unitarias, la combinación de componentes puede ser incorrecta o insatisfactoria [Beizer, 1990].

Existen diferentes tácticas para llevar adelante las pruebas de integración [Schaefer, et al; 2014].

  - Integración descendente: Se trata de un enfoque incremental para la construcción de la arquitectura del software. La prueba se iniciará con el componente de más alto nivel del sistema que llama a otros componentes del sistema pero no a sí mismo. La integración avanza con componentes de nivel inferior.
  - Integración ascendente: Comienza con los componentes elementales del sistema que no requieren componentes adicionales. Los subsistemas más grandes se ensamblan a partir de los componentes probados.
  - Integración Ad-hoc: Los componentes se van integrado en el orden en que están terminados. Cuando un componente ha pasado la prueba de componentes, se inicia la prueba de integración para ver si encaja con otro componente ya probado y se inicia la prueba de integración.
  - Integración del esqueleto: Cuando un esqueleto o columna vertebral del sistema se ha terminado, los componentes se integran gradualmente en él.

- Prueba de sistema

La prueba del sistema refiere al comportamiento del sistema entero. La mayoría de las faltas funcionales deben haber sido identificadas ya durante las pruebas de unidad e integración. La prueba del sistema generalmente se considera apropiada para probar los requerimientos no funcionales del sistema, tales como seguridad, desempeño, exactitud, y confiabilidad. Las interfaces externas, los dispositivos de hardware, o el ambiente de funcionamiento también se evalúan a este nivel [IEEE, 2004].

A continuación se describen los tipos de prueba del sistema [Myers, 2004]:

- Prueba de Volumen: Este tipo de prueba del sistema somete el software a grandes volúmenes de datos. El propósito es demostrar que el software puede o no manejar el volumen de datos especificados en sus objetivos. El volumen a probar puede requerir recursos significativos, en términos del tiempo de procesador y de personas.
- Pruebas de Estrés: Estas pruebas someten al programa a cargas pesadas. Una carga pesada es un volumen máximo de datos, o de actividad, en un plazo corto de tiempo.
- Pruebas de Usabilidad: Intentan encontrar problemas de usabilidad del sistema en su interacción con humanos. El análisis de factores humanos es una cuestión altamente subjetiva.
- Pruebas de Seguridad: Intentan subvertir los chequeos de seguridad del sistema. Una forma de pensar tales casos de prueba es estudiar problemas conocidos de seguridad en sistemas similares.
- Pruebas de Desempeño: Muchos programas tienen objetivos específicos de desempeño o de eficiencia, indicando características tales como tiempos de respuesta y rendimiento bajo ciertas condiciones de carga de trabajo y configuración.
- Pruebas de Almacenamiento: Algunos programas tienen establecidos límites en el almacenamiento, que indican, por ejemplo, la cantidad de memoria principal y secundaria que el programa utiliza. Se diseñan los casos de prueba para demostrar que estos objetivos de almacenamiento no se han resuelto.
- Pruebas de Configuración: Implican probar el programa con distintas configuraciones de hardware y software posibles
- Pruebas de Compatibilidad/ Configuración/ Conversión: La mayoría de los programas que se desarrollan no son totalmente nuevos; son a menudo reemplazos de un sistema existente. Algunas veces, estos programas tienen objetivos específicos referentes a su compatibilidad y procedimientos de conversión del sistema existente. Se diseñan los casos de prueba para demostrar que los objetivos de la compatibilidad no se han resuelto y que los procedimientos de conversión no funcionan.
- Pruebas de Instalación: La prueba de instalación es una parte importante del proceso de prueba del sistema, particularmente en un sistema empaquetado con instalación automatizada.

- Pruebas de Confiabilidad: Estos tipos de prueba tienen como objetivo la mejora de la confiabilidad del programa, probando que las declaraciones específicas sobre confiabilidad se cumplen.
- Pruebas de Recuperación: Algunos programas tienen requerimientos específicos de recuperación que indican cómo el sistema se recupera de errores de programación, de faltas del hardware, y de errores en los datos. Un objetivo de la prueba del sistema es demostrar que estas funciones de recuperación no funcionan correctamente
- Pruebas de Mantenimiento: Algunos programas tienen objetivos de mantenimiento, como por ejemplo: requerimientos específicos sobre las ayudas que se proporcionarán al sistema, los procedimientos de mantenimiento, y la calidad de la documentación de la lógica interna.

- Pruebas de validación o aceptación

Estas pruebas, a diferencia de las anteriores, son responsabilidad del cliente y pueden ser la única parte de las pruebas en donde estén involucrados. Estas pruebas se llevan a cabo cuando el producto está listo para implantarse y antes de que el programa se ponga en funcionamiento y tiene que satisfacer las expectativas del cliente. [ISTQB, 2002]

El usuario debe ser el que realice las pruebas, ayudado por personas del equipo de pruebas, siendo deseable, que sea el mismo usuario quien aporte los casos de prueba.

- Pruebas de aceptación del contrato: Toman como base los criterios de aceptación previstos en el contrato realizado al principio del proyecto. Estas pruebas son muy importantes, ya que en el análisis los requerimientos pudieron ser malinterpretados, eso conlleva a un error en los criterios de aceptación y no tener en cuenta aspectos que sí tiene en cuenta el cliente. Por esto, el programa se entrega al cliente para que realice sus pruebas. Estas pruebas suelen ser las mismas que se realizaron en las pruebas de sistema, con la salvedad de que se realizan en el entorno del cliente, característica muy importante, ya que pueden aparecer errores que antes no aparecieron en los entornos propios de desarrollo.
- Pruebas de aceptación del usuario: Hay casos en los que el cliente y el usuario final son diferentes y lo que le parece válido a un usuario final, puede ocurrir que no le parezca válido a otro. Por este motivo, es fundamental realizar pruebas con todos los usuarios finales.
- Pruebas operativas: Estas son llevadas a cabo por los administradores del sistema que mantendrán el software en producción. En estas pruebas se incluyen tareas complejas

tales como copia de seguridad/restauración, recuperación de desastres, gestión de usuarios, comprobación de vulnerabilidades de seguridad, carga de datos y tareas de mantenimiento.

- Pruebas alfa y beta: Esta pruebas son utilizadas cuando se presenta el caso en que un programa sea utilizado por diferentes usuarios finales y que la cantidad de usuarios finales sea muy amplia. Cuando esto ocurre no es práctico realizar pruebas de aceptación con cada uno de estos clientes, para lo cual se utilizan las pruebas de alfa y beta que van a descubrir errores que sólo el usuario final va a encontrar. Las pruebas alfa son las que se ejecutan en las oficinas del desarrollador del producto por un grupo de personas que representa al cliente final. En estas pruebas, el desarrollador estará junto a estos usuarios registrando errores y problemas de uso. Las pruebas beta se realizan en las oficinas de un cliente o en varias situaciones específicas, ya que el usuario final será todos los clientes a los que se les entrega el producto. En estas pruebas el desarrollador no está presente. Se genera un canal donde los clientes le hacen llegar al desarrollador todos los problemas registrados durante las pruebas.

2.1.9. Técnicas de prueba

Para continuar con el objetivo de que el producto termine con la calidad deseada, además de saber qué estrategia de prueba y los tipos de pruebas a realizar es necesario definir qué técnicas de prueba que se pueden aplicar a la hora de realizar las pruebas.
Las técnicas tienen el objetivo de identificar y conformar condiciones, casos y datos de la prueba.

- Técnicas estáticas

Este tipo de técnicas son las primeras comprobaciones que se aplican al software. Su objetivo es ayudar a los ingenieros a reconocer y arreglar sus propios errores en etapas tempranas del proceso de desarrollo sin ejecutar código. [Sommerville, 2005]
Estas técnicas definen las revisiones sistemáticas aplicables al desarrollo, operación y mantenimiento del software [IEEE 1028, 1990]
Existen diferentes tipos de revisiones, las cuales dependen del grado de formalidad de las mismas.

- Revisión informal: En este tipo de técnica, los revisores carecen de instrucciones escritas, ni produce documentos de resultados. Se intenta obtener defectos con bajo costo.
- Revisión guiada: Son llevadas a cabo por el autor de un documento del proyecto y el objetivo principal es encontrar defectos y establecer un entendimiento común.
- Revisión técnica: Aquí el equipo técnico debate, toma decisiones, evalúa alternativas y comprueba la conformidad con las especificaciones, estándares y normativas.
- Revisión de gestión: Sirven para controlar el progreso y detectar inconsistencias de los planes con la programación y en los requisitos.
- Inspecciones: Son las revisiones más formales. Se basa en el examen visual de documentos para detectar defectos como puede ser el no cumplimiento de estándares de desarrollo.
- Recorrido (walkthrough): Inspecciones conducidas únicamente por miembros del grupo de desarrollo que examinan una parte específica del producto.
- Auditorias: Son un tipo de revisión del software la cuál es llevada a cabo por uno o más interventores, los cuales no pertenecen a los miembros de la organización. Se trata de un examen independiente de un producto, de un proceso, o de un sistema software con el fin de determinar su conformidad con las especificaciones, los estándares, los acuerdos contractuales u otros criterios.

- Técnicas dinámicas

Este tipo de técnica es la que se realiza con la ejecución de la aplicación, en estos tipos recaen la mayoría de las pruebas.

- Técnica de caja blanca

  Esta técnica, conocida también como “caja de cristal” es aquella donde los casos de prueba se derivan a partir de la estructura del sistema. Requieren conocer el código del programa a probar.

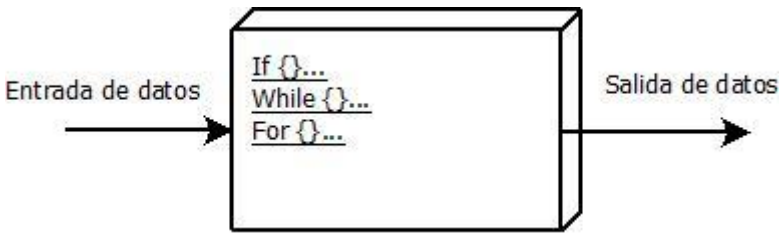

Figura 6. Representación de la técnica de caja blanca.

- Basadas en el flujo de control: Los criterios basados en el flujo de control o la estructura del programa, cubren todas las sentencias o bloques de sentencias en un programa, o combinaciones especificadas de ellas. Se han propuesto varios criterios de cobertura. El más fuerte de los criterios es el de flujo de control, que ejecuta todas las trayectorias del flujo del control de la entrada a la salida. Puesto que la prueba de la trayectoria no es generalmente factible debido a los bucles, otros criterios menos rigurosos son utilizados en la práctica, por ejemplo de cobertura de sentencia, de condiciones y de decisión. La adecuación de tales pruebas se mide en porcentajes; por ejemplo, se dice haber alcanzado una cobertura de sentencia del 100% cuando todos las sentencias han sido ejecutadas por lo menos una vez por las pruebas [IEEE, 2004].
- Basadas en el flujo de los datos: La prueba en el flujo de datos usa la información sobre cómo se definen, se utilizan, y se destruyen las variables del programa. El criterio más fuerte, all definition-use paths, requiere que para cada variable, cada segmento de la trayectoria del flujo del control desde una definición de esa variable hasta su uso sea ejecutado. Para reducir el número de las trayectorias requeridas se utilizan estrategias más débiles como all-definitions y all-uses [IEEE, 2004].
- Mutantes: Un mutante es una versión levemente modificada del programa a probar, diferenciado de él por un cambio sintáctico pequeño. Cada caso de prueba ejercita el programa original y todos los mutantes generados: si un caso de prueba identifica la diferencia entre el programa y un mutante, se dice que el mutante fue "matado". Puede ser usado para evaluar un conjunto de prueba o como criterio de prueba en sí mismo. En este último caso, las pruebas se diseñan específicamente para matar a mutantes que sobreviven. Para que la técnica sea eficaz, se deben derivar automáticamente de una manera sistemática una gran cantidad de mutantes [IEEE, 2004].

En muchas ocasiones se pone tanto énfasis en la estructura del código que se ignora la especificación del programa, convirtiendo al testing en una tarea un tanto desprolija e inconsistente [Ghezzi, Jazayeri & Mandrioli, 1991].

- Técnica de caja negra

Se denominan técnicas de caja negra o de prueba funcional a aquellas donde los casos de prueba derivan a partir de la especificación funcional como no funcional, sin tener en cuenta la estructura interna del programa [Myers, 2004]. A diferencia de las pruebas de caja blanca, estas suelen realizarse durante las últimas etapas de las pruebas.

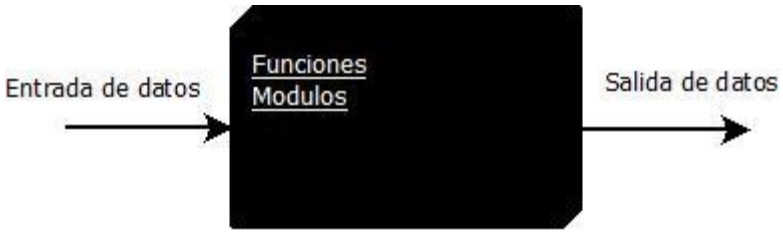

Figura 7. Representación de la técnica de caja negra.

Para llevar adelante este proceso de pruebas, se pueden utilizar estas principales técnicas:

- Partición de equivalencia: En estas pruebas, se intenta seleccionar un conjunto de valores de entrada del programa o del sistema que vayan a tener un comportamiento similar y con una alta probabilidad de encontrar la mayoría de errores pudiendo ser procesados de la misma forma. De esta forma se reduce el número de casos de prueba a desarrollar, buscando cubrir todos los casos posibles de prueba. [ISTQB, 2005].
- Análisis del valor límite: Tomando como premisa que el mayor número de errores se presenta en los límites del dominio de estrada que en el centro [Pressman, 2005], en esta técnica se intenta explorar dichas condiciones límites, obteniendo una rentabilidad mayor [Myers, 2004].
- Tablas de decisión: Si la lógica a probar está basada en decisiones, este tipo de pruebas son las que se deben llevar adelante. Se generan tablas donde se indican las condiciones y acciones de entradas, donde cada columna corresponde a una regla de negocio, que define a una combinación única de las condiciones que dan lugar a la ejecución de las acciones asociadas a esa regla. [ISTQB, 2005].
- Máquinas de estado finito: Un programa puede tener diferentes respuestas dependiendo de condiciones o estados particulares, mostrándose como una máquina de estado. Es por esto que viéndolo de esta forma, las pruebas se

pueden seleccionar para cubrir diferentes estados y sus transiciones.[IEEE, 2004].

- Grafo causa efecto: El grafo causa efecto ayuda a seleccionar de una manera sistemática los casos de prueba. Una causa es una condición de entrada, mientras que un efecto es una condición de salida o una transformación del sistema. La segunda iteración de esta prueba es convertirse en una tabla de decisión, donde cada columna representa un caso de prueba.
- Pruebas de casos de uso: Un caso de uso expresa todas las maneras de utilizar un sistema para alcanzar un objetivo particular para el usuario final. En conjunto, los casos de uso le proporcionan todos los caminos útiles de usar el sistema e ilustran el valor que este proporciona. [Bittner & Spence, 2002]. En particular un caso de uso es una secuencia de acciones que un sistema realiza con el fin de lograr un resultado de valor para quien interactúa con el sistema de forma particular. Los casos de prueba derivados de los casos de uso resultan muy útiles a la hora de descubrir defectos en todos los caminos útiles durante el uso real del sistema. [ISTQB, 2005].

- Técnicas basadas en la experiencia

Son aquellas en las que las pruebas se derivan de la habilidad e intuición del probador y de su experiencia con aplicaciones y tecnologías similares [ISQTB, 2012].

Las técnicas principales basadas en la experiencia son:

- Predicción de error: En este tipo de pruebas, los casos de prueba se diseñan en base a la experiencia del probador. Logrando así anticipar los errores en base a la experiencia de quien prueba el sistema o componente y diseñar pruebas específicas en base a esa experiencia para encontrar los errores.
- Exploratorias: Las pruebas exploratorias son un estilo de pruebas que hacen foco en la libertad personal y la responsabilidad del probador para optimizar continuamente el valor de su trabajo mediante el aprendizaje de las pruebas, diseño de pruebas, ejecución de las pruebas y la interpretación de los resultados de las pruebas como actividades que se apoyan mutuamente y que se ejecutan paralelamente a lo largo del proyecto [Kaner, 1983].

En el testing exploratorio siempre se debe tomar nota de lo que se hizo y lo que sucedió [Kaner, 1999]. Los resultados del testing exploratorio no son necesariamente diferente de aquellos obtenidos de la prueba con diseño previo y ambos enfoques para las pruebas son compatibles [Bach, 2001].

2.1.10. Plan de pruebas

El plan de pruebas de software se elabora para atender los objetivos de calidad en un desarrollo de sistemas, encargándose de definir aspectos como por ejemplo los módulos o funcionalidades sujeto de verificación, tipos de pruebas, entornos, recursos asignados, entre otros aspectos.[PMO,2016]

Este plan define diez pasos a seguir:

- Analizar los requerimientos de desarrollo de software

Para elaborar un plan de pruebas de software lo primero a realizar es entender los requerimientos de usuario que componen la iteración o proyecto, que son el sujeto de la verificación de calidad que se va a realizar.

Se analizará toda la información de la ingeniería de requisitos, incluyendo la matriz de trazabilidad, especificaciones y diseño funcional, requisitos no funcionales, casos de uso, historias de usuario (si se está trabajando con metodologías ágiles), entre otra documentación.

También es muy importante realizar entrevistas con el equipo encargado de la ingeniería de requisitos para aclarar dudas y ampliar la información que sea necesaria.

- Identificar las funcionalidades nuevas a probar

A partir de la documentación del análisis de requisitos y de las entrevistas con el equipo de ingeniería de requisito y desarrollo, se identifican e incluyen en el plan de pruebas de software la lista de las funcionalidades (características) totalmente nuevas.

Si se está trabajando con un sistema informático nuevo, no será problema discernir, pues todas serán nuevas.

En el caso de desarrollos de software integrados a un sistema existente es necesario revisar con los analistas de negocio y también con los arquitectos de software las funcionalidades que forman parte del desarrollo de software, en todas las capas de la arquitectura.

- Identificar las funcionalidades de sistemas existentes que deben probarse

Se debe identificar las funcionalidades existentes que estén siendo impactadas por el desarrollo de alguna forma, considerando todos los componentes afectados en todas las capas de la arquitectura de software.

- Definir la estrategia de pruebas

Consiste básicamente en seleccionar cuáles son los tipos de pruebas de software que se deben realizar. Es recomendable seguir un marco de referencia para determinar los tipos de prueba, como por ejemplo los tipos de pruebas de software definidos por el ISTQB.

- Definir los criterios de inicio, aceptación y suspensión de pruebas

Para la definición de los criterios de aceptación o rechazo, es necesario precisar el nivel de tolerancia a fallos de calidad. Si la tolerancia a fallos es muy baja puede definirse como criterio de aceptación que el 100% de los casos de prueba estén sin incidencias. Lograr este margen en todos los casos de prueba principales y casos borde será muy difícil, y podría comprometer los plazos del proyecto (incrementa los riesgos), pero asegura la calidad del producto.

Por otra parte, puede ser que la intención sea realizar un Soft Launch, o un mínimo producto viable, en ese caso se podría definir como criterio de aceptación el 100% de los casos de prueba principales (considerados clave) y 20% de casos de prueba no principales (casos borde).

Una vez logradas las condiciones, se darán por aceptadas las pruebas y el desarrollo de software.

- Identificar los entornos (ambientes) requeridos

Se definen y documentan las características de los entornos de Hardware y Software necesarios para realizar la ejecución de las pruebas de software.

Esta información se obtiene a partir del equipo de desarrollo y de los arquitectos de software, quienes pueden suministrar los requisitos mínimos y óptimos para la operación del sistema.

Como mejor práctica, el ambiente de pruebas de software debería ser lo más similar posible al ambiente de producción, sin embargo, no siempre es posible debido a limitaciones de recursos (financieros). En estos casos debe estudiarse cuales son los requisitos que aseguran un mínimo de confiabilidad de estas pruebas respecto al entorno de producción.

- Determinar necesidades de personal y entrenamiento

Debe completarse previamente la estimación del esfuerzo de pruebas a partir del diseño de casos de prueba.

Si aún no se cuenta con la estimación, se puede comenzar por definir los tipos de perfiles de habilidades y conocimientos en Software Testing que se necesitan.

- Establecer la metodología y procedimientos de prueba

La metodología de pruebas de software dependerá de la que se esté utilizando para la gestión del proyecto.

Si se está utilizando una metodología predictiva, las pruebas de software comenzaran con la estimación del esfuerzo de pruebas, diseño y luego la ejecución de las pruebas.

Si se están utilizando metodologías ágiles de desarrollo de software, se deben considerar las diferencias de las pruebas ágiles de software respecto al enfoque predictivo, por lo que la metodología debe estar alineada con el manifiesto ágil.

Luego se seleccionar la metodología de referencia, se documentan los procedimientos para diseño y ejecución, siguiendo el orden de los pasos definidos, flujos de procesos, condiciones para tomar decisiones, y demás aspectos.

- Elaborar la planificación de las pruebas

La planificación de las pruebas requiere una descripción de responsabilidades ya que las tareas de pruebas deben estar alineadas con las habilidades y conocimientos de cada persona.

También se debe elaborar un cronograma a partir de la estimación de las actividades de Software Testing realizada por el equipo donde se especifican las premisas necesarias que deben cumplirse para que el cronograma sea realizable, determinadas a partir de la documentación de entornos y de los requisitos de personal.

- Identificar los riesgos y definir planes de respuesta

Para identificar los riesgos es necesario enumerar cada una de estas dependencias y por medio de mesas de trabajo y tormentas de ideas pensar en las posibilidades de que algo salga mal (u oportunidades para que salga bien).

Luego de la identificación, es necesario también definir planes de respuesta, los cuales deben ser específicos para cada situación particular y riesgo.

2.2. SEGURIDAD INFORMATICA

2.2.1. Seguridad

Podemos entender como seguridad una característica de cualquier sistema (informático o no) que nos indica que ese sistema está libre de todo peligro, daño o riesgo, y que es, en cierta manera, infalible. Como esta característica, particularizando para el caso de sistemas operativos o redes de computadores, es muy difícil de conseguir (según la mayoría de expertos, imposible), se suaviza la definición de seguridad y se pasa a hablar de fiabilidad (probabilidad de que un sistema se comporte tal y como se espera de él) más que de seguridad; por tanto, se habla de sistemas fiables en lugar de hacerlo de sistemas seguros [Villalon Huerta, 2002].

Los objetivos principales aspecto a garantizar por la seguridad informática son los siguientes:

- Disponibilidad y accesibilidad: Es un requisito necesario para garantizar que el sistema trabaje puntualmente, con prontitud y que no se deniegue el servicio a ningún usuario autorizado. La disponibilidad protege al sistema contra determinados problemas como intentos deliberados o accidentales de realizar un borrado no autorizado de datos, de causar cualquier tipo de denegación del servicio o de acceso a los datos y de los intentos de utilizar el sistema o los datos para propósitos no autorizados. La disponibilidad, frecuentemente, es uno de los objetivos de seguridad más importante de toda organización.
- Integridad: Se encarga de garantizar que la información del sistema no haya sido alterada por usuarios no autorizados, evitando la perdida de consistencia, teniendo dos grandes grupos:
    - Integridad de datos: Es la propiedad de que los datos no hayan sido alterados de forma no autorizada, mientras se almacenan, procesan o transmiten.
    - Integridad del sistema: Es la cualidad que posee un sistema cuando realiza la función deseada. De manera no deteriorada y libre de manipulación no autorizada. La integridad normalmente, es el objetivo de seguridad más importante después de la disponibilidad.
- Confidencialidad de datos e información del sistema: Es el requisito que intenta que la información privada o secreta no se revele a individuos no autorizados. La protección de la confidencialidad se aplica a los datos almacenados durante su procesamiento, mientras se transmiten y se encuentra, frecuentemente, detrás de la disponibilidad y de la

integridad en términos de importancia. Para algunos sistemas y para tipos específicos de datos, como los de autenticación, la confidencialidad es de extrema importancia.

- Responsabilidad (registros de auditoria): Es el requisito que permite que puedan trazarse las acciones de una entidad de forma única. A menudo, es un requisito de la política de la organización y soporta de forma directa el no repudio, la disuasión, el aislamiento de fallos, la detección y la prevención de intrusiones.
- Confiabilidad: Es la garantía en que los cuatro objetivos anteriores se han cumplido adecuadamente. Es la base de la confianza en que las medidas de seguridad, tanto técnicas, como operaciones, funcionan tal y como se idearon para proteger el sistema y la información que procesa.

2.2.2. ¿Qué se busca proteger?

La información es un activo que tiene un valor fundamental para la organización y debe ser protegida de un modo adecuado. Así, la seguridad de la información protege a ésta de un amplio rango de amenazas para asegurar la continuidad del negocio, minimizar los daños a la organización y maximizar las oportunidades de negocio.

2.2.3. Seguridad informática vs seguridad de la información

La seguridad informática y la seguridad de la información pueden parecer lo mismo. Sobre todo si se tiene en cuenta que el desarrollo y la evolución de la tecnología tiende hacia el modelo de digitalizar y manejar cualquier tipo de información mediante un sistema informático. No obstante, aunque se encuentran destinadas a vivir en armonía y trabajar de forma conjunta, cada una de las áreas de seguridad tiene objetivos y actividades diferentes.

La seguridad informática se describe como la distinción táctica y operacional de la seguridad, mientras que la seguridad de la información es la línea estratégica de la seguridad.

Se debe tener en cuenta la definición de la seguridad de la información como la disciplina que se encarga de la implementación técnica de la protección de la información, el despliegue de las tecnologías que establecen de forma que se aseguran las situaciones de fallas parciales o totales, cuando la información es el activo que se encuentra en riesgo.

Esta disciplina se encarga de llevar a cabo las soluciones técnicas de protección de la información.

Por otro lado, la seguridad de la información es la disciplina que nos habla de los riesgos, de las amenazas, de los análisis de escenarios, de las buenas prácticas y los esquemas normativos, que nos exigen niveles de aseguramiento de procesos y de tecnología para elevar el nivel de confianza en la creación, utilización, almacenaje, transmisión, recuperación y disposición final de la información.

Se encarga esencialmente de proporcionar la evaluación de riesgos y amenazas, trazar el plan de acción y adecuación para minimizar los riesgos, bajo la normativa o las buenas prácticas con el objetivo de asegurar la confidencialidad, integridad y disponibilidad del manejo de la información de activos.

La seguridad de la información es la disciplina que se encarga de garantizar la confidencialidad, integridad y disponibilidad de la información. Para conseguir el objetivo se apoya a la seguridad informática, es decir, a pesar de ser disciplinas diferentes, la una no puede ir sin la otra. De forma que la seguridad de la información será la encargada de regular y establecer las pautas a seguir para la protección de la información.

Es habitual que la seguridad de la información se apoye en la política de seguridad que se desarrolla mediante la elaboración de un plan director de seguridad. La dirección será la encargada de marcar todas las líneas de actuación en materia de seguridad y mediante el plan director para determinar las medidas tanto técnicas como procedimentales que garantice los objetivos marcados por la política de seguridad.

Muchas veces escuchamos hablar de seguridad de la información y seguridad informática indistintamente. Por una parte, la seguridad informática protege el sistema informático, tratando de asegurar la integridad y la privacidad de la información que contiene. Por lo tanto, podríamos decir, que se trata de implementar medidas técnicas que preservaran las infraestructuras y de comunicación que soportan la operación de una empresa, es decir, el hardware y el software empleados por la empresa.

La seguridad de la información va mucho más allá, puesto que intenta proveer de medidas de seguridad a otros medios donde se localiza la información como:

- Impresos en papel.
- Discos duros.
- Medidas de seguridad respecto de las personas que la generan o que acceden a ella.

Se encuentra orientado no solo a preservar la información, sino además a mejorar los procesos de negocio. Se deben añadir a las medidas técnicas, otras organizativas o legales que permitan a la organización asegurarse una mayor solidez de la confidencialidad, integridad y disponibilidad de los sistemas de información.

La seguridad de la información integra toda la información independientemente del medio en el que esté. La seguridad informática atiende sólo a la protección de las instalaciones informáticas y de la información en medios digitales. [Cano, 2011]

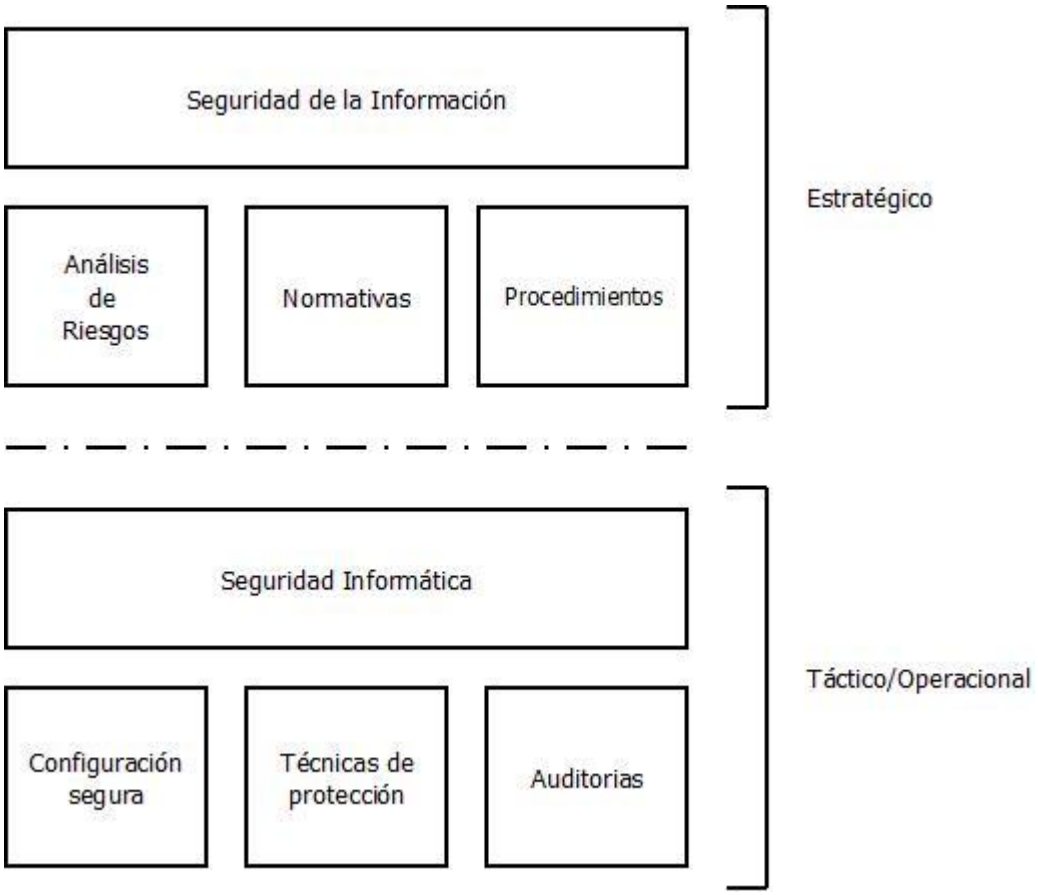

Figura 8. Seguridad de la información versus seguridad informática.

2.2.4. Clasificación de ataques

Los ataques que alteran buscan alterar la seguridad de las aplicaciones pueden clasificarse en dos grandes grupos:

- Ataques Activos

  Los ataques activos suelen modificar la información, los datos o los mensajes. Pueden consistir, por ejemplo, en el cambio de la identidad de un emisor/receptor, la manipulación de datos, denegación de servicios, encaminamiento incorrecto o incluso la repetición.

- Ataques Pasivos

  Pueden consistir, básicamente en:

  - Observación de mensajes (simple acceso a la información)

- Análisis de tráfico (no se accede a la información, que suele ir cifrada, sino que se roba información sobre el tráfico: tipo de frecuencias de envío, identificación de usuarios, y en general, características del intercambio entre sistemas que pueden usarse después en acciones como el reemplazo de IPs en cachés de servidores DNS).

2.2.5. Tipos de ataques

Ahora bien los ataques suelen tener diferentes modalidades tales como:

- Interrupción

Las consecuencias de este tipo de ataque son típicamente la destrucción y/o inutilización de la información. Es un ataque, pues, sobre la disponibilidad.

Ejemplos: destrucción de un elemento hardware (ataques físicos), corte de una línea de comunicaciones (deliberada o accidentalmente), borrado de ficheros, registros, bases de datos, programas. En caso de un fallo de este tipo, es necesario detectarlo convenientemente, evaluarlo y sobre todo actuar rápidamente, momento en el cual intervendrán diversos factores, entre ellos el económico.

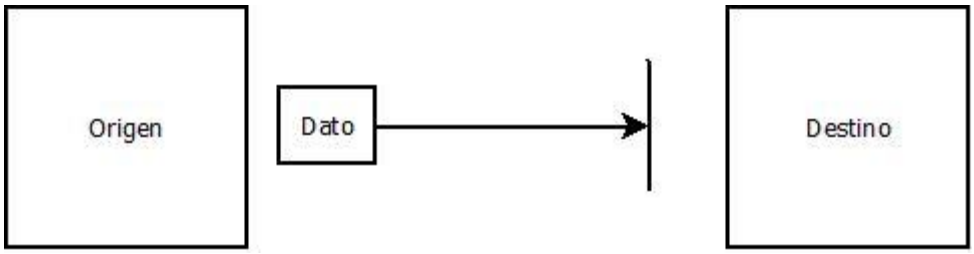

Figura.9. Ataque por interrupción.

- Intercepción

La intercepción consiste en la participación sin autorización realizada por personas, computadoras o en general cualquier tipo de entidad, en la comunicación entre la fuente y el destino de la información.

Es el ataque más difícil de detectar, ya que todo funciona bien, no como en el caso anterior; aquí se está atentando contra la confidencialidad. Ejemplos: sniffers, copia de software, etc.

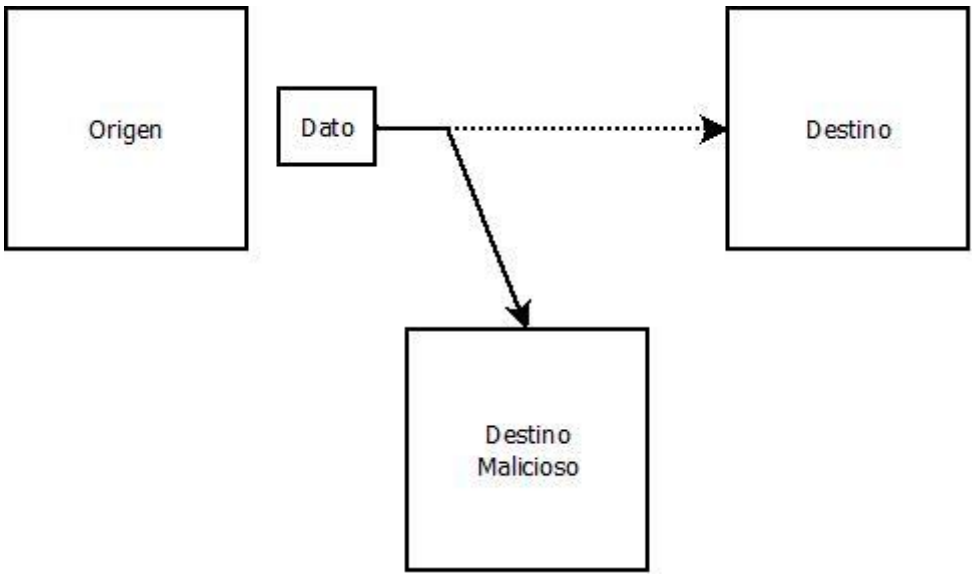

Figura 10. Ataque por intercepción.

- Modificación

En un ataque de modificación se captura la información, que es tratada y posteriormente reenviada al destino

Es un ataque contra la integridad de los datos, por lo que resulta muy peligroso. Ejemplos: cambios en valores de BD, programas, modificación de mensajes, troyanos.

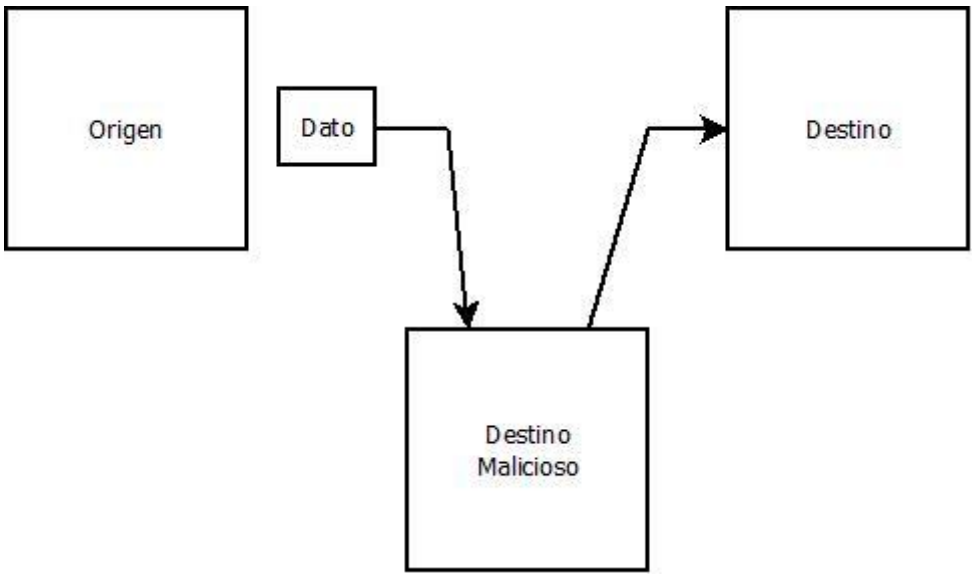

Figura 11. Ataque por modificación.

- Fabricación

En ocasiones como esta, el intruso o elemento subversivo intenta hacerse pasar por la fuente, siendo, pues, un ataque sobre la autenticidad. Se introducen en el sistema objetos o entidades fabricadas.

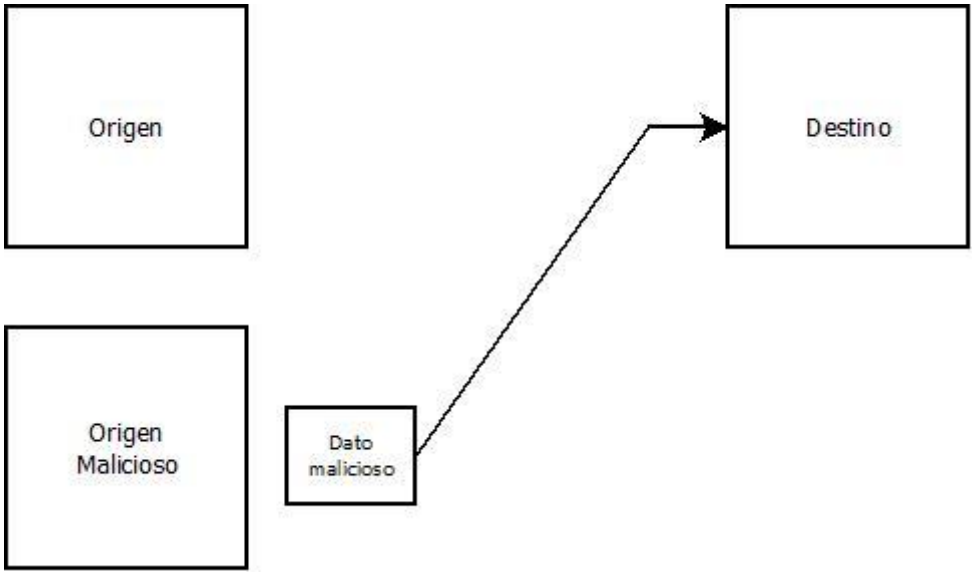

Figura 12. Ataque por fabricación.

2.2.6. Vulnerabilidades

En un sistema informático lo que se busca proteger son sus activos, es decir, los recursos que forman parte del sistema y que se pueden agrupar en:

- Hardware

Elementos físicos del sistema informático, tales como procesadores, electrónica y cableado de red, medios de almacenamiento.

- Software

Elementos lógicos o programas que se ejecutan sobre el hardware, tanto si es el propio sistema operativo como las aplicaciones.

- Datos

Comprenden la información lógica que procesa el software haciendo uso del hardware. En general serán informaciones estructuradas en bases de datos o paquetes de información que viajan por la red.

Dentro de estos activos, el más vulnerable son los datos. Tanto el hardware como el software se pueden reponer con facilidad, pero los datos dependen de que la empresa tenga una buena política de copias de seguridad y sea capaz de reponerlos en el estado más próximo al momento en que se produjo la pérdida. Esto puede suponer para la empresa, por ejemplo, la dificultad o imposibilidad de reponer dichos datos con lo que conllevaría de pérdida de tiempo y dinero.

Se entiende por vulnerabilidad a la debilidad de cualquier tipo que compromete la seguridad del sistema informático.
Pueden ser agrupadas en función a:

- Diseño
  Debilidad en el diseño de protocolos utilizados en las redes.
  Políticas de seguridad deficientes e inexistentes.
- Implementación
  Errores de programación.
  Existencia de "puertas traseras" en los sistemas informáticos.
  Descuido de los fabricantes.
- Uso
  Mala configuración de los sistemas informáticos.
  Desconocimiento y falta de sensibilización de los usuarios y de los responsables de informática.
  Disponibilidad de herramientas que facilitan los ataques.
  Limitación gubernamental de tecnologías de seguridad.

2.2.7. Vulnerabilidades más conocidas

A medida que avanzan los desarrollos y la detección de amenazas, se pueden comenzar a registrar las modalidades más comunes para contar con detecciones tempranas.
A continuación se detallan las más conocidas:

- Vulnerabilidad de desbordamiento de buffer.
  Si un programa no controla la cantidad de datos que se copian en buffer, puede llegar un momento en que se sobrepase la capacidad del buffer y los bytes que sobran se almacenan en zonas de memoria adyacentes.

En esta situación se puede aprovechar para ejecutar código que nos de privilegios de administrador.

Vulnerabilidad de condición de carrera.

Si varios procesos acceden al mismo tiempo a un recurso compartido puede producirse este tipo de vulnerabilidad. Es el caso típico de una variable, que cambia su estado y puede obtener de esta forma un valor no esperado.

- Vulnerabilidad de Cross Site Scripting (XSS).

  Es una vulnerabilidad de las aplicaciones web, que permite inyectar código VBSript o JavaScript en páginas web vistas por el usuario. El phishing es una aplicación de esta vulnerabilidad. En el phishing la víctima cree que está accediendo a una URL (la ve en la barra de direcciones), pero en realidad está accediendo a otro sitio diferente. Si el usuario introduce sus credenciales en este sitio se las está enviando al atacante.

- Vulnerabilidad de denegación del servicio.

  La denegación de servicio hace que un servicio o recurso no esté disponible para los usuarios. Suele provocar la pérdida de la conectividad de la red por el consumo del ancho de banda de la red de la víctima o sobrecarga de los recursos informáticos del sistema de la víctima.

- Vulnerabilidad de ventanas engañosas (Window Spoofing).

  Las ventanas engañosas son las que dicen que eres el ganador de tal o cual cosa, lo cual es mentira y lo único que quieren es que el usuario de información.

2.2.8. Políticas de seguridad

El objetivo de la Política de Seguridad de Información de una organización es, por un lado, mostrar el posicionamiento de la organización con relación a la seguridad, y por otro lado servir de base para desarrollar los procedimientos concretos de seguridad.

La empresa debe disponer de un documento formalmente elaborado sobre el tema y que debe ser divulgado entre todos los empleados. No es necesario un gran nivel de detalle, pero tampoco ha de quedar como una declaración de intenciones. Lo más importante para que estas surtan efecto es lograr la concienciación, entendimiento y compromiso de todos los involucrados.

Las políticas deben contener claramente las prácticas que serán adoptadas por la compañía. Y estas políticas deben ser revisadas, y si es necesario actualizadas, periódicamente.

Las políticas deben:

- Definir qué es seguridad de la información, cuáles son sus objetivos principales y su importancia dentro de la organización
- Mostrar el compromiso de sus altos cargos con la misma
- Definir la filosofía respecto al acceso a los datos
- Establecer responsabilidades inherentes al tema
- Instaurar la base para poder diseñar normas y procedimientos referidos a
    - Organización de la seguridad
    - Clasificación y control de los datos
    - Seguridad de las personas
    - Seguridad física y ambiental
    - Plan de contingencia
    - Prevención y detección de virus
    - Administración de los computadores

A partir de las políticas se podrá comenzar a desarrollar, primero las normas, y luego los procedimientos de seguridad que serán la guía para la realización de las actividades.

La política de seguridad comprende todas las reglas de seguridad que sigue una organización (en el sentido general de la palabra). Por lo tanto, la administración de la organización en cuestión debe encargarse de definirla, ya que afecta a todos los usuarios del sistema.

La seguridad informática de una compañía depende de que los empleados (usuarios) aprendan las reglas a través de sesiones de capacitación y de concienciación.

Sin embargo, la seguridad debe ir más allá del conocimiento de los empleados y cubrir las siguientes áreas:

- Un mecanismo de seguridad física y lógica que se adapte a las necesidades de la compañía y al uso de los empleados
- Un procedimiento para administrar las actualizaciones
- Una estrategia de realización de copias de seguridad planificada adecuadamente
- Un plan de recuperación luego de un incidente
- Un sistema documentado actualizado

Por lo tanto y como resumen, la política de seguridad es el documento de referencia que define los objetivos de seguridad y las medidas que deben implementarse para tener la certeza de alcanzar estos objetivos. [Mifsud, 2018]

2.2.9. Buenas prácticas

Sumado a las políticas de seguridad, a más bajo nivel, las buenas prácticas de seguridad mejoran de forma exponencial las posibles pérdidas de información.

- Mantener actualizado el sistema operativo y las aplicaciones

  Es importante mantener actualizados los sistemas operativos (SO) y las aplicaciones con sus correspondientes parches de seguridad, nombre que recibe el código que soluciona una debilidad en un SO o aplicación.

  En cuanto a este aspecto de la seguridad, las medidas prácticas de prevención se enfocan en:

  - No descargar actualizaciones desde sitios de dudosa reputación y hacerlo sólo desde sitios de confianza. Descargar las actualizaciones desde sitios no oficiales implica un potencial riesgo de infección.
  - Descargar las actualizaciones a través de los mecanismos ofrecidos por el fabricante. En el caso de las actualizaciones de productos de Microsoft, la disponibilidad de los mismos es informada el segundo martes de cada mes, aunque puede haber excepciones en casos de vulnerabilidades críticas.
  - En entornos corporativos, y sin importar la plataforma, se aconseja preparar políticas de gestión de actualizaciones claras, que permitan coordinar y administrar los parches de seguridad tanto de los sistemas operativos como de las aplicaciones. Lo ideal es que esta política de gestión forme parte de la PSI (Política de Seguridad de la Información)

- Aseguramiento del sistema operativo

  Otro de los aspectos importantes en materia de prevención, radica en configurar el sistema operativo para hacerlo más seguro. Entre las buenas prácticas que se pueden tener en cuenta se encuentran:

  - Deshabilitar las carpetas compartidas. Esto evita la propagación de gusanos que aprovechen ese vector como método de infección.
  - Utilizar contraseñas fuertes. El empleo de contraseñas fáciles de recordar es otra de las debilidades que los códigos maliciosos suelen aprovechar para propagarse por los recursos de información.

- Crear un perfil de usuario con privilegios restringidos. Por defecto, el usuario que crean las plataformas Windows al momento de su implementación posee privilegios administrativos. Esto es un factor que aumenta la probabilidad de infección.
- Deshabilitar la ejecución automática de dispositivos USB. Los dispositivos de almacenamiento removibles que se conectan al puerto USB constituyen un vector de ataque muy empleado por el malware para la propagación, sobre todo, de gusanos.
- De ser posible, migrar hacia plataformas (sistemas operativos) modernas. En la actualidad, los sistemas operativos antiguos (Microsoft Windows9x, NT) no cuentan con soporte técnico ni con actualizaciones de seguridad por parte de Microsoft, lo cual constituye un punto que permite la explotación de vulnerabilidades
- Configurar la visualización de archivos ocultos ya que la mayoría de los códigos maliciosos se esconden en el sistema con este tipo de atributos.
- Configurar la visualización de las extensiones de archivos para poder identificar las extensiones de los archivos descargados y no ser víctimas de técnicas como la doble extensión.

- Protección en el correo electrónico

  El correo electrónico constituye uno de los canales de propagación/infección de malware más utilizados por atacantes; por lo tanto es importante que los usuarios incorporen como hábito determinadas prácticas que permitan prevenir los ataques realizados a través de códigos maliciosos. En consecuencia, a continuación se presenta una serie de medidas preventivas orientadas a aumentar la seguridad durante el uso del correo electrónico.

  - Spam

    El spam es el correo electrónico que promociona diferentes productos y servicios a través de publicidad no solicitada, enviada masivamente a las direcciones de correo de los usuarios. Constituye uno de los principales medios de propagación de una importante cantidad de códigos maliciosos y por lo tanto se recomienda:

    - No confiar en correos spam con archivos adjuntos y explorar el archivo antes de ejecutarlo. Esto asegura que no se ejecutará un malware.

- Cuando se reciben adjuntos, prestar especial atención a las extensiones de los mismos, ya que suelen utilizar técnicas de engaño como la doble extensión o espacios entre el nombre del archivo y la extensión del mismo.
- Evitar publicar las direcciones de correo en sitios web de dudosa reputación como sitios pornográficos, foros, chats, entre otros. Esto minimiza la posibilidad de que la dirección se guarde en la base de datos de los spammers.
- Utilizar filtros anti-spam que permitan el filtrado del correo no deseado.
- No responder jamás el correo spam. Es preferible ignorarlos y/o borrarlos, ya que si se responde se confirma que la dirección de correo se encuentra activa.
- En lo posible, evitar el re-envío de mensajes en cadena (por lo general son hoax), ya que suelen ser utilizados para recolectar direcciones de correo activas.
- Si de todos modos se desea enviar mensajes en cadena, es recomendable hacerlo siempre Con Copia Oculta (CCO) para que quien lo recibe lea solo la dirección del emisor.
- Proteger la dirección de correo utilizando una cuenta alternativa durante algún proceso de registro en sitios web y similares. Esto previene que la dirección de correo personal sea foco del spam.
- Utilizar claves seguras y cambiar la contraseña con periodicidad si se utiliza webmail. Esto favorece la seguridad de la cuenta, evitando que sea descubierta a través de un proceso sencillo.
- Configurar la pregunta secreta, además, de una forma que no sea adivinable para fortalecer aún más la seguridad de la cuenta.
- Como medida de seguridad extra, bloquear las imágenes de los correos y descargarlas cuando se asegure de que el correo no es dañino.
- También es preferible que se evite ingresar el nombre de usuario y su respectiva contraseña en sitios de los cuales no se tenga referencia. De esta manera se preserva la privacidad de la cuenta de correo y, por ende, la información que se intercambia a través de la misma.

- Phishing

El phishing es una modalidad delictiva encuadrada en la figura de estafa realizada a través de Internet, y constituye otra de las amenazas de seguridad más propagadas a través del correo electrónico.

Entre las buenas prácticas de seguridad que se recomiendan a los usuarios, para que éstos eviten ser víctimas del phishing, están las siguientes:

- Tener en cuenta que las entidades bancarias y financieras no solicitan datos confidenciales a través de este medio, de esta manera se minimiza la posibilidad de ser víctima de esta acción delictiva.
- Desconfiar de los correos que dicen ser emitidos por entidades que brindan servicios y solicitan cambios de datos sensibles ya que suelen ser métodos de Ingeniería Social.
- No hacer clic sobre enlaces que aparecen en el cuerpo de los correos electrónicos, ya que pueden re direccionar hacia sitios web clonados o hacia la descarga de malware.
- Asegurarse de que la dirección del sitio web al cual se accede comience con el protocolo https. La ”s” final, significa que la página web es segura y que toda la información depositada en la misma viajará de manera cifrada.
- Verificar la existencia de un certificado digital en el sitio web. El certificado digital se despliega en pantalla al hacer clic sobre la imagen del candado.
- Revisar que el certificado digital no haya caducado, ya que el mismo podría haber sido manipulado intencionalmente con fines maliciosos.
- Comunicarse telefónicamente con la compañía para descartar la posibilidad de ser víctimas de un engaño, si se tiene dudas sobre la legitimidad de un correo.
- Jamás se debe enviar contraseñas, números de tarjetas de crédito u otro tipo de información sensible a través del correo electrónico, ya que la comunicación podría ser interceptada y robada.
- Habituarse a examinar periódicamente la cuenta bancaria, a fin de detectar a tiempo alguna actividad extraña relacionada con la manipulación de la cuenta o transacciones no autorizadas.

- Denunciar casos de phishing (dentro de lo posible) en la entidad de confianza, ya que además de cortar la actividad del sitio malicioso, se colabora con la seguridad general de la navegación en Internet.

- Seguridad en la navegación

  En los últimos años, Internet se ha transformado en una plataforma de ataque donde acciones delictivas se llevan a cabo a través de diferentes técnicas como por ejemplo el Drive-by-Download. En consecuencia, es fundamental navegar con cautela y tener presente las recomendaciones más importantes.

  Entre ellas:

  - Evitar el ingreso a sitios web con contenidos que, dependiendo el país, son ilegales, como aquellos que ofrecen cracks y programas warez; ya que constituyen canales propensos a la propagación de malware.
  - Impedir la ejecución de archivos desde sitios web sin verificar previamente que es lo que dice ser. Es importante no hacer clic sobre el botón ejecutar ya que esto provoca que el archivo se ejecute automáticamente luego de descargado, dejando al margen la posibilidad de verificar su integridad.
  - Descargar programas de seguridad solamente desde el sitio oficial del mismo, para evitar la descarga de archivos que pudieran ser previamente manipulados con fines delictivos.
  - Si es posible, leer atentamente las políticas de privacidad de las aplicaciones descargadas desde sitios web no oficiales o de dudosa reputación, antes de instalarlas.
  - No realizar la instalación de complementos extras como barras de tareas o protectores de pantallas sin verificar previamente su autenticidad.
  - Configurar el navegador web para minimizar el riesgo de ataques a través del mismo.
  - Instalar, en lo posible, un programa antivirus con capacidades proactivas que permita detectar códigos maliciosos incluso desconocidos y explorar con el mismo cada archivo descargado.
  - Disponer, además, de un Firewall personal que permita bloquear comunicaciones entrantes y salientes.

- Tratar de no acceder a servicios como Home-Banking desde lugares públicos (ciber, bibliotecas, cafés, hoteles, etc.).
- Si se navega desde sitios públicos, es recomendable eliminar los archivos temporales, caché, cookies, direcciones URL, contraseñas y formularios donde se haya ingresado datos.
- El bloqueo de determinados sitios considerados maliciosos, ya sea porque descargan malware o porque contienen material de dudosa reputación, es también otra de las mejores prácticas que ayudan a la prevención y refuerzan la seguridad del equipo.

- Seguridad en redes sociales

  En la actualidad, las redes sociales son muy populares y los usuarios las utilizan masivamente; estas características las transforman en importantes focos de propagación de malware. Por tal motivo, se torna necesario tener en cuenta y aplicar las siguientes medidas preventivas:

  - Intentar no publicar información sensible y confidencial, debido a que personas extrañas pueden aprovechar esta información con fines maliciosos.
  - También es recomendable evitar la publicación de fotografías propias y de familiares. Las fotografías pueden ser utilizadas para complementar actos delictivos, incluso fuera del ámbito informático.
  - Mantener la privacidad del perfil; es decir, configurar el perfil para que no sea público.
  - No responder las solicitudes de desconocidos, ya que pueden contener códigos maliciosos o pueden formar parte de actividades delictivas.
  - Ignorar los mensajes que ofrecen material pornográfico, pues usualmente a través de ellos suele canalizarse la propagación de malware, además de otras acciones ofensivas desde el punto de vista ético y moral.
  - No abrir contenidos con spam a través de este medio. De esta manera se evita formar parte del ciclo de vida del spam a través de este canal.
  - Cambiar periódicamente la contraseña para evitar que la misma sea descubierta fácilmente.
  - Antes de aceptar contactos espontáneos, es recomendable verificar su existencia y que realmente provienen de quien dice ser.

- Seguridad en redes P2P

  Las redes Punto a Punto, más conocidas como P2P, forman otro de los canales por donde se propagan diferentes amenazas informáticas y cuya relación con códigos maliciosos es muy activa. Esta situación obliga a tener en cuenta una serie de medidas preventivas tendientes a fortalecer la seguridad del sistema, entre las cuales se destacan:

  - Explorar con una solución antivirus de alta efectividad en la detección de amenazas conocidas y desconocidas, absolutamente todos los archivos que se descargan a través de esta red, sin importar su extensión.
  - Evitar el almacenamiento de información confidencial y sensible en la misma computadora donde se comparten archivos por redes P2P, para evitar que la misma sea robada.
  - Verificar que el programa cliente de intercambio de archivos no instale o descargue componentes extras, ya que en la mayoría de los casos son códigos maliciosos del tipo Adware/Spyware.
  - Asegurarse de que los archivos a descargar no se encuentren sometidos a métodos de engaño como doble extensión, debido a que se trata de una técnica muy empleada por el malware.
  - Controlar que exista coherencia entre el tamaño original del archivo descargado y el tamaño aproximado que debería tener, para descartar la posibilidad de que se esté en presencia de programas troyanos.
  - Chequear que la carpeta de intercambio de archivos contenga sólo los archivos que se desea compartir.
  - Revisar la configuración de seguridad del programa cliente. Esto ayuda a maximizar la seguridad durante el proceso de descarga de archivos.

- Seguridad en mensajería instantánea

  Otro medio de comunicación popular, y que se emplea masivamente, son los clientes de mensajería instantánea, que, en consecuencia, constituyen uno de los vehículos más explotados por diferentes amenazas, dentro de las cuales una de las más activas es el malware. Por tal motivo poner en ejecución medidas tendientes a volver más seguro el cliente de mensajería instantánea se transforma en una tarea casi obligada. Para prevenir ser víctimas de acciones maliciosas llevadas a cabo a través de esta tecnología, se recomienda aplicar alguna de las medidas de seguridad que a continuación se describen:

- Evitar aceptar como contacto cuentas desconocidas sin verificar a quién pertenece, ya que en la mayoría de los casos se trata de intentos de engaños con fines maliciosos.
- No descargar archivos sospechosos, sobre todo cuando vienen acompañados de mensajes genéricos o en otro idioma. Esto constituye una de las características principales de los códigos maliciosos que se propagan a través de este canal de comunicación.
- En caso de descargar archivos, explorarlos con una solución antivirus con capacidad proactiva antes de ser ejecutados, para verificar que se encuentren libre de amenazas.
- Configurar en el cliente de mensajería la exploración automática de archivos en el momento de su recepción. La mayoría de estos clientes contemplan la posibilidad de configurarlos con un antivirus.
- Es recomendable, al igual que con el correo electrónico, no hacer clic sobre los enlaces incrustados en el cuerpo del mensaje, ya que pueden direccionar a páginas con contenido malicioso o hacia la descarga de malware.
- Cuando se reciben mensajes conteniendo un enlace no esperado, es recomendable preguntar si la otra persona realmente lo ha enviado; de esta manera se puede verificar la autenticidad del mismo.
- No escribir los datos de autenticación en páginas que prometen ofrecer información de contactos bloqueados y similares. Estos sitios suelen comprometer la privacidad de la información que se aloja en los correos, además de utilizar la cuenta con otros fines delictivos.
- Cambiar la contraseña de manera periódica. Ayuda a maximizar el nivel de seguridad.
- No compartir la contraseña con nadie. El carácter de ésta es privado, con lo cual lo recomendable es que sólo la conozca el usuario que la ha creado.
- Cuando se accede al mensajero desde lugares públicos, es recomendable deshabilitar la opción de inicio automático para que no quede la dirección (ni la contraseña) grabada. Esto evita que terceros inicien sesión de manera automática.
- No compartir información confidencial a través de este medio ya que la misma puede ser interceptada y robada con fines delictivos.

- Seguridad en dispositivos removibles

Los dispositivos de almacenamiento removibles que se conectan a través del puerto USB (memorias, cámaras digitales, filmadoras, teléfonos celulares, etc.), constituyen otro de los mayores focos de propagación/infección de códigos maliciosos. Por lo tanto, es necesario tener presente alguna de las siguientes medidas que ayudan a mantener el entorno de información con un nivel adecuado de seguridad, ya sea en entornos corporativos como en entornos hogareños:

- Establecer políticas que definan el uso correcto de dispositivos de almacenamiento removibles. Esto ayuda a tener claro las implicancias de seguridad que conlleva el uso de estos dispositivos.
- Brindar acceso limitado y controlado de los usuarios que utilizan estos dispositivos, para controlar la propagación de potenciales amenazas y el robo de información.
- De ser necesario, registrar el uso de los mismos y/o habilitar/deshabilitar puertos del tipo USB. Esto permite un mayor control sobre el uso de dispositivos de este estilo.
- En casos extremos es recomendable bloquear, por medio de políticas de grupo, de dominio o corporativas, el uso de estos dispositivos.
- Si se transporta información confidencial en estos dispositivos, es recomendable cifrarla. De esta forma, en caso de robo o extravío, la información no podrá ser vista por terceros.
- Es recomendable explorar con el antivirus cualquier dispositivo que se conecte a la computadora para controlar a tiempo una posible infección.
- Deshabilitar la ejecución automática de dispositivos en los sistemas operativos Microsoft Windows, ya que muchos códigos maliciosos aprovechan la funcionalidad de ejecución automática de dispositivos de las plataformas Microsoft para propagarse a través de un archivo Autorun.inf. [Mieres, 2009]

2.2.10. Amenazas en entornos web

Se ha comprobado que en los últimos años, 75% o más de los ataques electrónicos fueron a nivel de aplicación (y no a nivel de host o de red).Esto se da ya que todo tipo de transacciones se realizan actualmente en la web, y cada vez en mayor proporción. Detalles de cuentas bancarias, tarjetas de crédito y todo tipo de información confidencial y de valor circulan en enormes

cantidades y continuamente. Es por ende lógico que la mayoría de los esfuerzos de hackers y demás atacantes se centre en vulnerar estas aplicaciones.

Las aplicaciones web están en parte definidas por su uso del protocolo HTTP como medio de comunicación entre cliente y servidor, el cual tiene puntos débiles tales como:

- Simpleza y basado en ASCII - no se requiere gran esfuerzo para generar pedidos y descifrar el contenido de las respuestas.
- Utiliza un puerto TCP bien conocido – de poco sirve un firewall para proteger una aplicación si tiene que admitir el tráfico a través del puerto 80
- No mantiene por sí mismo el estado de la sesión – un atacante no tiene que emular mecanismos de mantenimiento de sesión, basta con emitir un request para lograr el cometido. Mecanismos como el uso de cookies permiten simular una sesión virtual intercambiando información adicional en cada request/response, pero no son efectivas si no se las implementa bien, e introducen problemas adicionales de seguridad y privacidad.
- Existen muchas excepciones y variantes adicionales a estos elementos; en particular se utiliza ampliamente SSL como protocolo de encriptación a nivel de transporte en las comunicaciones cliente-servidor. Como explicaremos a continuación, esto está lejos de resolver todas las vulnerabilidades de la aplicación.

Aparte del protocolo que se utiliza existen otros detalles tales como:

- El usuario solamente enviará entradas esperadas - HTML admite el uso de tags que manipulan las entradas a la aplicación, por ejemplo si la aplicación utiliza campos ocultos para enviar información sensible estos pueden ser fácilmente manipulados desde el cliente.
- La validación puede realizarse únicamente del lado del cliente con JavaScript - si no se efectúa ninguna validación del lado del servidor, cualquier atacante que evite esta validación (para nada difícil de lograr) tendrá acceso total a toda la aplicación.
- El uso de firewalls es insuficiente, si el firewall tiene que habilitar los puertos 80 y/o 443 para que la aplicación sea accesible al exterior, no podrá hacer nada para detectar entradas maliciosas del cliente, y por supuesto no es protección contra ataques internos.

Los múltiples ataques externos a los que puede estar expuesto un sitio web son usualmente clasificados en seis categorías principales. Debajo se describen junto con ejemplos de tipos de ataques. [Scambray & Shema, 2002]

- Autenticación: explotan el método de validación de la identidad de un usuario, servicio o aplicación

  - Fuerza Bruta
  - Autenticación insuficiente
  - Débil validación de recuperación de Password
- Autorización: explotan el mecanismo de un sitio web de determinar si un usuario o servicio tiene los permisos necesarios para ejecutar una acción.
  - Predicción de Credenciales o Sesión
  - Autorización insuficiente
  - Expiración de Sesión insuficiente
  - Fijado de Sesión
- Ataques lógicos: explotan la lógica de la aplicación como flujos utilizados por la aplicación para efectuar cierta acción.
  - Abuso de funcionalidad
  - Denial of Service
  - Insuficiente Anti-Automatismo
  - Insuficiente validación de procesos
  - Manipulación de entradas (URL, campos)
- Ataques al cliente: atacan al usuario de la aplicación.
  - Content Spoofing
  - Cross-Site Scripting
- Ejecución de comandos : ataques diseñados para ejecutar comandos remotos en el servidor.
  - Buffer Overflow
  - Format String
  - LDAP Injection
  - Ejecución de Comandos
  - SQL Injection
  - SSI Injection
  - XPath Injection
- Robo de Información: ataques que apuntan a adquirir información específica sobre el sitio web.
  - Indexado de directorio
  - Caminos transversales
  - Predicción de ubicación de recursos
  - Escape de información

2.2.11. Evaluación de seguridad

Las organizaciones tienen como herramienta la auditoria para efectuar evaluación de seguridad en todos sus activos, procesos y sistemas.

La auditoria examina si el sistema está cumpliendo con los requerimientos de seguridad incluyendo políticas de la organización y del sistema. Dentro de las técnicas a emplear incluye investigación, observación y pruebas. Una auditoria puede variar ampliamente en alcance, examinar un sistema entero para el proceso de re acreditación o puede investigar un solo evento malicioso.

La auditoría puede ser interna o externa, la diferencia puede radicar en la objetividad con que se realice. La auditoría interna puede tener conflictos de intereses, o al contrario, estar motivado por el deseo de mejorar la seguridad del sistema, además de ser conocedores del sistema pueden encontrar problemas ocultos.

La auditoría toma documentación existente en cuanto a: políticas aplicables, análisis de riesgos, descripción de procesos, lista de controles. La ausencia de algún documento amerita la recomendación de la realización del mismo.

Las principales áreas de seguridad que se pueden evaluar en una auditoría de sistemas son:

- Evaluación de la seguridad física de los sistemas.

  La seguridad física se refiere por lo tanto, a la protección de los soportes de datos, equipos de redes, programas e instalaciones.

  Estos deben estar debidamente protegidos de factores físicos que los puedan dañar o disminuir su capacidad

- Evaluación de la seguridad lógica del sistema.

  La seguridad lógica, se refiere a la seguridad del uso de los datos, protegiendo los procesos y programas usados por las personas autorizadas por la gerencia.

- Evaluación de la seguridad del personal del área de sistemas.

  La seguridad del personal al momento de trabajar con los sistemas informáticos, deben estar en óptimas condiciones para que no causen ningún daño. En toda actividad se hace necesario, no solo planear y ejecutarlas, sino efectuar procedimientos de control que vayan encaminados a asegurar que dichas actividades han sido ejecutadas de acuerdo a los parámetros que se habían establecido con anterioridad.

- Evaluación de la seguridad de la información y las bases de datos.

  La seguridad de la información de la base de datos puede tener varios enfoques, confidencialidad, disponibilidad e integridad. Se debe tener en cuenta que esta

información se puede originar dentro o fuera de la entidad, siendo documentos referidos por el cliente, empleados o de cualquier otra índole.

- Evaluación de la seguridad en el acceso y uso del software.

  La evaluación de seguridad de los sistemas de información tiene como propósito la protección y resguardo de la información de la empresa. Para evaluar la seguridad en la operación del software, se deben identificar los aspectos que afecten la información como son:

  - Errores de aplicaciones.
  - Errores de sistemas operativos.
  - Rutinas de acceso no autorizados.
  - Servicios no autorizados.

- Evaluación de la seguridad en la operación del hardware.

  Para evaluar la seguridad en la operación del hardware, se deben identificar los aspectos que afecten la información como son:

  - Inapropiada operación.
  - Fallas en mantenimiento.
  - Inadecuada seguridad física.

- Evaluación de la seguridad en las telecomunicaciones.

  Debe reconocerse que los sistemas, redes y mensajes transmitidos y procesados son propiedad de la entidad y no deben usarse para otros fines no autorizados.

  Los usuarios tendrán restricciones de accesos según dominios, y únicamente podrán modificar los programas autorizados o variar las configuraciones siempre y cuando sean autorizados.

2.3. HACKING ETICO

2.3.1. ¿Porqué hacking?

Llamamos hacking a un conjunto de técnicas para acceder a un sistema informático sin autorización. Existe autorización cuando se dispone de un control de acceso mediante el uso de identificadores de usuario y passwords. Es un término tradicionalmente ligado a la libertad de Información de Internet. En sus códigos está el respetar la vida privada, pero eso después de aprender cómo funcionan los sistemas y dónde están los datos. Entre sus medios destacan los

Sniffers o escaneadores de puertos, programas que buscan claves, passwords y puertos abiertos.[De Miguel, 2007]

2.3.2. ¿Por qué ético?

Para emular la metodología de ataque de un intruso informático y no serlo, tiene que haber ética de por medio, más allá de todas las condiciones, términos y activos que haya alrededor del caso.
No entraremos en el terreno de lo ético como rama dentro de la filosofía práctica ni redactaremos un tratado sobre deontología profesional, no sólo por la cuestión de espacio, sino también por respeto a los griegos. Sí, en cambio, diremos que la ética implica que el trabajo y la intervención del profesional en seguridad informática o de la información no comprometen de ningún modo los activos de la organización, que son los valiosos datos con los que ella cuenta. [Tori,2008]

2.3.3. Perfil de conocimientos

Un Hacker Ético será, seguramente, un experto en informática y sistemas, tendrá certeros conocimientos sobre los sistemas operativos, sabrá sobre hardware, electrónica, redes, telecomunicaciones y también programación en lenguajes de alto y bajo nivel.
Además, entenderá sobre problemas relacionados con seguridad en temáticas tales como la criptografía, los sistemas de control de acceso, las aplicaciones, la seguridad física y la seguridad administrativa
Un Hacker Ético seguirá un estricto código de conducta, dado que de eso se trata la primera parte del concepto. Pero no todo termina allí, el perfil no es una cosa estática y maciza, sino que requiere de la constante renovación en busca de nuevos conocimientos, mucha investigación, prueba de herramientas, etc. Quien quiera alcanzar dicho nivel, además de dedicar el tiempo suficiente, deberá armarse de un alto grado de paciencia, perseverancia, y por sobre todo, una gran dosis de humildad. [Benchimol, 2011]

2.3.4. Diferencia entre hacker y cracker

Suelen confundirse los términos, pero existe una gran diferencia entre ellos, un hacker es un individuo con amplios conocimientos en sistemas informáticos, quienes acceden de forma remota, no autorizada, a traces de Internet. No suelen dejar huellas en cambio hay movimientos que dejan una marca o código que los identifica. Por otro lado los crackers son expertos o

aficionados de las nuevas tecnologías, que de forma consciente y voluntaria emplean sus conocimientos en la informática con fines maliciosos, antimorales y bélicos para el robo de información, fabricación de virus, piratería, acceso ilegal a sistemas gubernamentales, con el objetivo de beneficiarse, lucrarse e incluso causar daños en un objetivo. [Gimenez, 2011]

2.3.5. Niveles de ataque

Los ataques producidos por los hackers, suelen tener un mismo objetivo, pero no suelen ser de la misma naturaleza.

Se detallan a continuación los niveles de ataques más comunes:

- Sistema operativo

  Los ataques al sistema operativo constituyen un punto clásico de la seguridad. Desde esta perspectiva, la búsqueda de fallas se realizará en lo concerniente al propio sistema base de todo el resto del software, de tal modo que, muchas veces, independientemente de lo que se encuentre por encima, se podrá explotar y tomar control del sistema en caso de que sea vulnerable. En la actualidad tenemos tres líneas principales: los sistemas del tipo Windows, los del tipo Linux o derivados de UNIX, y los sistemas MAC OSX, los cuales, si bien están basados en UNIX, a esta altura presentan entidad propia. En el caso de los primeros, desde su origen fueron objeto de ataque dada su masificación y la relativa simplicidad con que se pudo acceder históricamente al núcleo del sistema, incluso, sin contar con su código fuente. Para el caso de Linux la situación es quizá peor, ya que, al poseer el código fuente, es posible detectar problemas también a nivel de código. Y en cuanto a OSX, la velocidad con la que ha acaparado mercado de múltiples plataformas en los últimos años, sumado a que los controles de seguridad implementados no son suficientes frente a las amenazas actuales, hacen que el sistema operativo de MAC sea un blanco cada vez más buscado por los atacantes. Pese a lo que se cree, la estadística de cantidad de vulnerabilidades de Windows no supera anualmente a la de Linux; en general, la diferencia ha sido la velocidad con la que aparecían las soluciones en cada caso, llevando aquí Linux la delantera.

- Aplicación

  En este caso, la variedad es mayor. Existen miles y miles de piezas de software y programas de todo tipo y tamaño, disponibles en el mundo. Por supuesto, entre tantos millones de líneas de código, necesariamente se producen errores. Para los ataques a las

aplicaciones también se tendrá en cuenta la masividad de uso. Esto implica que un programa manejado por millones de personas para leer archivos del tipo PDF será mejor objetivo que uno empleado por unos pocos para editar cierto tipo de archivos específicos de un formato menos conocido por los usuarios

Las aplicaciones amplían entonces la superficie de ataque de un sistema, por lo que se recomienda siempre evitar la instalación de aquellas que no se requieran, siguiendo el principio de seguridad que sugiere el minimalismo. La idea de atacar la implementación de algo en vez del software en sí mismo también vale para este caso. Muchos son los programas que realizan las mismas funciones, solo que algunos podrían hacerlo de manera tal que puedan encontrarse fallos en dicha operatoria, lo que comprometería al software, y con él, al sistema completo. Justamente esta es otra de las problemáticas. Dependiendo de los privilegios con los cuales se ejecute cierto programa, si es comprometido, podría afectar de forma directa al sistema, ya que se utilizaría el mismo nivel de permisos para atacarlo desde adentro, y tal vez, hasta escalar privilegios para llegar al máximo nivel, tema que analizaremos más adelante.

- Configuración

  El caso de las configuraciones, ya sean del sistema operativo o de las aplicaciones, también constituye un punto sensible, dado que por más seguro que sea un software, una mala configuración puede tornarlo tan maleable como un papel. Pensemos en un ejemplo muy elemental, como sería un antivirus: su configuración deficiente podría hacer que cumpliera su función de manera poco efectiva, provocando que una buena herramienta terminara por traducirse en una mala solución y, por ende, en una brecha de seguridad. Aquí reside el peligro; ni siquiera las herramientas de protección y seguridad son fiables en sí mismas solo por su función. Esto podría producir algo muy grave, pero que suele darse con frecuencia tanto en el ambiente corporativo como en el personal: una falsa sensación de seguridad.

  Si bien con el paso del tiempo las empresas han incorporado cada vez más medidas de seguridad en sus configuraciones de fábrica, un atacante, como primera medida, tratará de aprovecharse de las configuraciones estándar, ya sean aplicaciones, equipos informáticos, dispositivos de red, etcétera. Por ejemplo, si un panel de administración web se instala con un conjunto de credenciales de acceso por defecto y estas no son modificadas, cualquiera que conozca dichas credenciales podrá acceder. No perdamos de vista que en Internet existe una gran cantidad de sitios que presentan contraseñas por defecto de aplicaciones y dispositivos, por ejemplo, http://cirt.net/ passwords. En este

sitio podremos encontrar, clasificados por fabricante, una gran variedad de dispositivos con sus claves predefinidas. La solución más efectiva a estos problemas, sin dudas, es el hardening. Este proceso consiste en utilizar las propias características de dispositivos, plataformas y aplicaciones para aumentar sus niveles de seguridad. Cerrar puertos que no son imprescindibles, deshabilitar protocolos y funciones que no se utilicen, cambiar parámetros por defecto y eliminar usuarios que no sean necesarios son solo algunos ejemplos sencillos de un proceso de hardening. En el ámbito corporativo, como resultado de este proceso y luego de un análisis exhaustivo de sus propios sistemas, surge una serie de configuraciones mínimas indispensables para obtener el mejor nivel de seguridad sin perder de vista los requerimientos de negocio de la organización. Este conjunto de configuraciones se documenta y recibe el nombre de baseline, ya que describe cuáles son las necesarias para que los equipos y las aplicaciones implementen las recomendaciones propuestas por las buenas prácticas de seguridad y, a su vez, estén alineadas con los objetivos de negocio. En función de lo comentado previamente, podemos notar que deben existir diversos baselines, uno por cada aplicación o sistema. De esta forma, por ejemplo, tendremos un baseline para sistemas Microsoft Windows 2008, otro para Ubuntu Server, otro para routers Cisco, etc. Pero a su vez, también podríamos tener un baseline para MS SQL 2005, que deberá contemplar todos los puntos del baseline de Windows Server 2008; uno para servidores de correo sendmail, que deberá contemplar los puntos de baseline de Ubuntu Server, y otros más. La implementación de baselines permite garantizar que todos los sistemas estén estandarizados en sus configuraciones y que posean el mejor nivel de seguridad en función de los requerimientos del negocio.

- Protocolos

  Otro problema, tal vez más grave pero menos frecuente con el que podemos enfrentarnos, es que los errores estén directamente en los protocolos. Esto implica que, sin importar la implementación, el sistema operativo, ni la configuración, algo que se componga de dicho protocolo podría verse afectado. El ejemplo clásico es el Transmission Control Protocol/Internet Protocol (TCP/IP), una suite de protocolos tan efectiva y flexible, que, luego de más de tres décadas de existencia, aún perdura y continúa en uso. El problema aquí es que, en su momento, a principios de los años 70, su diseño no obedecía a aspectos de seguridad por determinados motivos propios de su objetivo de uso, y con toda razón. Con el tiempo, su utilización se extendió a tal punto, que comenzó a ser implementado de maneras que el propio esquema permitía, pero para fines que no había sido pensado en un principio, de modo que se transformó en un arma de doble filo. A pesar de esto, TCP/IP nunca ha estado en dudas, ya que todos los fallos se han ido corrigiendo o bien se

mitigaron sus efectos a partir de las mejoras realizadas por las implementaciones; incluso, el modelo de referencia Open System Interconnection (OSI) se basó en él. Por otro lado, la masividad que tiene el protocolo hace que su reemplazo sea imposible en la práctica. Como podemos imaginar, dado que existen centenares de protocolos, hay, a la vez, muchas posibilidades de encontrar fallos en ellos. El problema más grave es que un error en el diseño de uno implica que las situaciones sean potencialmente incorregibles, y que deben realizarse modificaciones a distintos niveles para resolverlo, incluyendo a veces su variación total o parcial, o su reemplazo por otro más seguro. Dentro de esta rama de errores, también incluimos los protocolos y algoritmos criptográficos, que, como veremos, tienen un alto nivel de complejidad y pueden producir huecos de seguridad realmente muy grandes dada la función de protección para la que son utilizados. [Jara & Pacheco, 2012]

2.3.6. Evaluación de seguridad anti hackeo

La evaluación de seguridad de una organización se realiza en función de la profundidad y el alcance que se le quiera dar. Existen diferentes tipos herramientas evaluadoras que ayudan a llevar adelante esta decisión, entre ellas las más conocidas son:

- Vulnerability Assessment

  El concepto de Vulnerability Assessment (VA) o evaluación de vulnerabilidades es utilizado en un sinfín de disciplinas y se refiere a la búsqueda de debilidades en distintos tipos de sistemas. En este sentido, no solo se remite a las tecnologías informáticas o a las telecomunicaciones, sino que incluye áreas como, por ejemplo, sistemas de transporte, sistema de distribución de energía y de agua, procesos de biotecnología, energía nuclear, y otros. De esta manera, se busca determinar las amenazas, los agentes de amenaza y las vulnerabilidades a los que está expuesto el sistema en su conjunto. Estas debilidades suelen referirse a todas aquellas de carácter técnico que dependen de las cualidades intrínsecas del sistema que se esté evaluando. En nuestro caso, teniendo en cuenta lo antedicho, vamos a hablar sobre Vulnerability Assessment cuando nos refiramos a un análisis técnico sobre las debilidades de una infraestructura informática y de telecomunicaciones. Puntualmente, se analizarán vulnerabilidades asociadas a distintos servidores, dispositivos, sistemas operativos, aplicaciones y un largo etcétera vinculado a todas las deficiencias técnicas posibles. Es importante destacar que este tipo de

evaluaciones solo identifica potenciales vulnerabilidades, pero no confirma que estas existan. Dicho de otra forma, cuando se detecta una vulnerabilidad en un equipo o sistema, no se trata de explotarla para confirmar su existencia, sino que, simplemente, se la reporta. Por lo general, las diferentes normativas exigen efectuar determinada cantidad de evaluaciones de vulnerabilidades en forma anual. Por ejemplo, PCI-DSS requiere cuatro evaluaciones en el año.

En relación a este tipo de evaluaciones, se desarrolló el Open Vulnerability and Assessment Language (OVAL), un estándar internacional de seguridad de la información abierto, cuyo objetivo es promocionar y publicar contenido de seguridad y normalizar la transferencia de este por todo el espectro de herramientas y servicios de seguridad. Incluye un lenguaje desarrollado en XML utilizado para codificar los detalles de los sistemas y una colección de contenido relacionado alojado en distintos repositorios, mantenidos por la comunidad OVAL. Su sitio oficial es: http://oval.mitre.org.

- Análisis de brecha de cumplimiento

  Otro tipo de evaluación que está creciendo en popularidad en estos últimos años es el análisis de brecha, o GAP Analysis. Debemos tener en cuenta que su objetivo es medir la distancia entre el estado actual de cumplimiento de una organización frente a los requisitos planteados por una regulación o estándar. Si bien parece bastante sencillo en su definición, muchas veces no lo es debido a las particularidades de cada una de las organizaciones. Así, en función de lo mencionado en párrafos anteriores, una entidad podría estar interesada en determinar cuál es la brecha que la separa del cumplimiento de la normativa PCI, de la normativa ISO/IEC 27.001, de la Ley de Protección de Datos Personales o de la comunicación A-4609 del BCRA, entre otras. Desde ya, esta brecha se traduce en un conjunto de requerimientos a los cuales se debe dar cumplimiento si se desea acceder a la certificación en el caso de PCI e ISO, o a no tener implicancias legales en el caso de la Ley de Protección de Datos Personales. Si bien es importante la verificación técnica por parte del profesional que lleva adelante la tarea o por algún colaborador, a fin de comprobar que lo relevado mediante entrevistas sea correcto, este paso no es usualmente requerido, ya que este tipo de evaluaciones suele basarse en la información que brinda la organización o que es relevada a partir de la serie de entrevistas realizadas. A modo de complementar el punto anterior, es útil realizar también una evaluación de vulnerabilidades o test de intrusión para verificar técnicamente el cumplimiento de aquellos requisitos relacionados con los sistemas de información, redes, etc.

- Autotesteo y contratación

  Por un lado, no todas las organizaciones poseen personal especializado que esté en condiciones de implementar esta clase de evaluaciones. En algunos casos, solo se cuenta con personal de sistemas o tecnología, que además realiza algunas tareas de seguridad solamente por ser el que sabe un poco más. Esta persona, por lo general, tiene conocimientos básicos en materia de seguridad de la información, por lo que sería impensado que pudiera llevar adelante un proyecto de evaluación de tal magnitud.

- Penetration test

  Si extendemos el concepto de Vulnerability Assessment y nos enfocamos en todos los procesos que involucren el manejo de la información de una organización, independientemente del medio en que esta se encuentre, nos acercamos a las evaluaciones del tipo Penetration Test o test de intrusión. A diferencia de los análisis de vulnerabilidades, estas pruebas no solo identifican las vulnerabilidades potenciales, sino que también tratan de explotarlas y, así, confirmar su existencia y el impacto real que podrían tener en la organización. Este punto es importante, ya que en muchas ocasiones, una vulnerabilidad reportada como crítica por el fabricante de la aplicación vulnerable no siempre es igualmente crítica en el contexto de una organización en particular. Puede darse el caso de que, incluso existiendo la vulnerabilidad, a partir de la presencia de otros controles compensatorios (recordemos otra vez el concepto de defensa en profundidad), la explotación de dicha vulnerabilidad se hace dificultosa o bien imposible. En esa situación, el riesgo que implica la presencia de la vulnerabilidad para esta organización puede no ser alto. Supongamos, por ejemplo, la existencia de una vulnerabilidad en un servidor de acceso remoto que, de ser explotada con éxito, permitiera tomar control total del equipo a distancia. Sin dudas, sería una vulnerabilidad crítica para cualquier organización. Ahora complejicemos un poco el escenario. Imaginemos, además, que este servidor está en una red segmentada, a la cual solo se puede acceder desde otra red integrada únicamente por equipos de confianza. Además, el equipo que posee el servidor vulnerable tiene definido un conjunto de reglas de filtrado que únicamente permiten el acceso desde dos ubicaciones. Este ejemplo sencillo nos presenta un caso en el que tenemos una vulnerabilidad claramente crítica, ya que cualquier atacante podría tomar control del equipo en forma remota. Sin embargo, existe un conjunto de controles que dificultan en gran medida la explotación de esta vulnerabilidad: en nuestro caso, el doble nivel de filtrado que debería pasar un atacante para poder explotar con éxito la vulnerabilidad. De esta manera, vemos que resulta importante no solo verificar la posible

existencia de vulnerabilidades, sino también evaluar cuál sería el impacto real para el negocio de la organización en caso de que fueran explotadas con éxito. Hace unos años, un problema común que surgía al momento de encarar un test de intrusión como de contratarlo era la falta de estandarización en cuanto a la metodología utilizada para llevarlo adelante y a las etapas en las cuales se dividía el proceso. A partir de nuestra experiencia y la compartida con colegas, pero siempre apoyada en metodologías y estándares internacionales, hoy podemos dividir el proceso en cinco etapas conceptuales.

2.3.7. Etapas del ciclo de hackeo

Tal como se explica en esta sección, el hacking ético analiza los sistemas y programas informáticos corporativos, asumiendo el rol de un ciberdelincuente y simulando ataques con el objetivo de evaluar el estado real de seguridad. Para llevar a cabo este proceso, se enumeran las mismas etapas de un hackeo real, las cuales son:

- Reconocimiento

  Si recordamos las clasificaciones de las evaluaciones de seguridad de la etapa anterior, tendremos presentes aquellas que son del tipo caja negra. Estas evaluaciones son las que más se acercan al proceso que lleva adelante un atacante real, ya que este no tiene conocimiento previo sobre la organización objetivo, más allá de su nombre. Por esta razón, es fácil intuir que la fase de reconocimiento es la que más tiempo insume dentro de la planificación de un ataque. En esta etapa, el atacante busca definir al objetivo con el mayor nivel de detalle posible, y a partir de eso, obtener la mayor cantidad de información. Cuando nos centramos en las personas físicas, algunos ejemplos de información que podemos obtener de ellas son direcciones de correo electrónico, direcciones postales, información personal, etcétera. Desde la perspectiva corporativa, la información que se buscará obtener abarca direcciones IP, resolución de nombres DNS, y otros datos. En esta etapa, también denominada Information Gathering (recopilación de información), el atacante se basa en distintas técnicas para llevarla a cabo; las más utilizadas son footprinting, ingeniería social y dumpster diving o trashing.

- Escaneo

  El escaneo corresponde a la segunda fase de la etapa de relevamiento. A partir de las direcciones IP y del resto de la información de los sistemas objetivos obtenidas como resultado de la etapa anterior, el objetivo final será encontrar todas las posibles fallas,

errores y vulnerabilidades, de modo tal de definir los vectores de ataque de cara a que, en una fase posterior, puedan ser explotados y, así, ganar acceso al sistema. Como pasos intermedios dentro de esta etapa, incluiremos la identificación de servicios, y la detección del sistema operativo y de las aplicaciones con el mayor nivel de detalle posible. Por ejemplo, en el caso del sistema operativo, es conveniente identificar no solo la plataforma, sino también la familia. Es decir, saber si el sistema es Microsoft o UNIX y, también, si es Windows 2003 o Windows 2008, y si tiene o no el último Service Pack instalado. En el próximo capítulo veremos el porqué de este nivel de detalle.

Por otro lado, respecto a las aplicaciones, en primera instancia el objetivo es determinar qué servicio se está brindando y, luego, cuál es la aplicación que lo hace y la versión específica. Por ejemplo, si el equipo estuviese funcionando como servidor web, habrá que determinar si está corriendo Apache o un Internet Information Server (IIS) y, adicionalmente, qué versión del servidor es. No es lo mismo que sea IIS 5.0, IIS 6.0 o IIS 7.0, ya que las medidas de seguridad implementadas y, sobre todo, las vulnerabilidades existentes entre las distintas versiones son fundamentales. Para esto utilizaremos varios métodos de escaneo, dependiendo del tipo de servicio o dispositivo que se quiera analizar, del sistema operativo y también de la aplicación que se esté ejecutando en el momento. Finalmente, se analizará con detalle la etapa de escaneo de vulnerabilidades, y veremos aquellas consideraciones que son preciso tener en cuenta antes de lanzar el ataque.

- Ganancia de acceso

  Esta es una de las fases para el hacker ya que es la fase en la que aplicara la estrategia planteada luego de que en la fase anterior haya encontrado las vulnerabilidades, para esto el hacker deberá hacer uso de todas sus habilidades mejor aún si usa herramientas que existen justamente para lo que desea realizar el hacer. El acceso puede ser localmente o de un medio externo, a través de secuestro de sesión (esto consiste en falsificar la identidad de un ordenador conocido para el ordenador de la víctima, y confundirla haciéndose pasar por esta), incluso tratando de descifrar la contraseña del ordenador de la víctima. Esta fase es decisiva ya que el hacker podrá ver el alcance de éxito que pueda tener su penetración.

- Mantenimiento de acceso

Esta fase consiste en mantener el acceso que gano en el sistema tratando de usar distintas herramientas como los sniffers que son usados para capturar el tráfico de red, este tráfico de red le servirá al hacker para poder obtener información sobre con que ordenadores interactúa su objetivo, lo que le servirá para poder hacer una falsificación de identidad haciéndose pasar por una de la direcciones conocidas y de confianza de sus objetivo, en esta fase debe iniciar sesiones telnet y FTP. En esta fase es importante que el hacker permanezca como indetectable para el objetivo, para esto debe remover el rastro de evidencia que dejo su penetración y haciendo uso de Backdoor y Troyanos para de esta manera intentar conseguir acceso con altos niveles de privilegio es decir como un administrador, como también podrían usar caballos de Troya para transferir información como nombres de usuario, passwords y cuentas de banco que podrían estar almacenadas en el sistema del objetivo.

- Borrado de huellas

  Una vez que el atacante ha sido capaz de ganar y mantener el acceso al sistema, cubre las huellas para evitar ser detectado por el personal de seguridad, para poder seguir usando el sistema comprometido, para eliminar evidencias de la violación al sistema y/o para evitar acciones legales. Es decir, trata de eliminar todos los rastros del ataque, como archivos de registro (log) o alarmas del Sistema de Detección de Intrusos (IDS). Ejemplos de actividades llevadas a cabo durante esta fase del ataque son la esteganografía (steganography), el empleo de protocolos de tunneling y la modificación de archivos de registro (log). [Mieres, 2010]

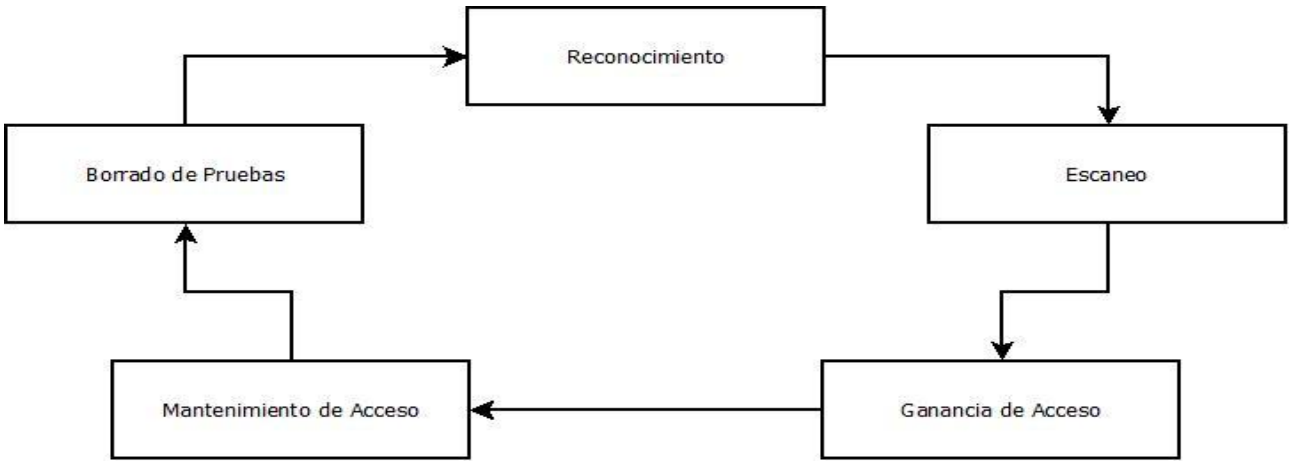

Figura 13. Fases de Hacking Ético

3. DESCRIPCIÓN DEL PROBLEMA

En este capítulo se desarrolla la identificación del problema de investigación a partir de la incorporación del método de testeo por hacking ético en las organizaciones (sección 3.1), caracterizando el problema abierto (sección 3.2) y concluyendo con un sumario de investigación exponiendo las debilidades y oportunidades de mejora identificadas (sección 3.3).

3.1. IDENTIFICACIÓN DEL PROBLEMA DE INVESTIGACIÓN

En el capítulo previo se ha llevado a cabo una revisión de los diferentes ciclos de vida propuestos para el testeo de software más conocidos y en general, las políticas de seguridad de la información o los controles por sí solos no garantizan la protección total de la información, ni de los sistemas de información, servicios o redes. Después de los controles que se implementan, vulnerabilidades residuales probablemente permanezcan haciendo ineficaz la seguridad de la información y por lo tanto los incidentes son aún más posibles. Esto puede llegar a tener efectos negativos tanto directos e indirectos sobre las operaciones de negocio de una organización. Además, es inevitable que se produzcan nuevos casos de amenazas no identificadas previamente. Una preparación insuficiente por una organización para hacer frente a este tipo de incidentes hará cualquier respuesta menos efectiva, y aumentar así el grado de impacto comercial potencial adverso. [ISO/IEC 27035:2011]

En su búsqueda de una manera de abordar el problema, las organizaciones informatizadas se dieron cuenta de que una de las mejores formas de evaluar la amenaza de intrusión a sus intereses sería tener profesionales independientes de seguridad informática intentando entrar en sus sistemas. Este esquema es similar a tener auditores independientes entrando en una organización para verificar sus registros de contabilidad. En el caso de seguridad informática, estos "hackers éticos" emplean las mismas herramientas y técnicas que los intrusos, pero sin dañar el sistema de destino ni robar información. En su lugar, permiten evaluar la seguridad de los sistemas de destino e informar de a los propietarios sobre las vulnerabilidades encontradas junto con las instrucciones de cómo remediarlos.

En resumidas palabras, la evaluación de la seguridad de un sistema por parte de un hacker ético busca responder 3 preguntas básicas:

¿Qué puede ver un intruso en los sistemas atacados?

¿Qué puede hacer un intruso con esa información?

¿Hay alguien en el sistema atacado que se dé cuenta de los ataques o éxitos del intruso?

Este proceso debe ser planificado con antelación. Todos los aspectos técnicos, de gestión y estratégicos deben estar sumamente cuidados. La planificación es importante para cualquier todas las pruebas, ya sea desde un simple análisis de contraseña a una prueba de penetración completa en una aplicación web [Mayorga Jácome et al, 2015; Santos Castañeda, 2016; Onofa Calvopiña et al, 2016; López Vallejo, 2017]. El resguardo de datos debe garantizarse, de lo contrario la prueba puede volverse en contra si alguien afirma que nunca se autorizaron las pruebas. Por lo tanto, un alcance bien definido implica la siguiente información:

- Sistemas específicos para probar.
- Estimar los riesgos que están involucrados.
- Tiempo que llevara la prueba y evaluación del calendario general.
- Recoger y explorar el conocimiento de los sistemas que tenemos antes de la prueba.
- Entrega de informes específicos incluyendo informes de evaluación de la seguridad y un informe de nivel superior describiendo las vulnerabilidades generales que deben abordarse, junto con las medidas correctivas que se deben implementar.

Claro que el profesional de seguridad, al llevar a cabo un test de penetración como parte de su trabajo de hacking ético, necesita contar con ese tipo de lógica y tiene que aplicarla, más allá de utilizar las técnicas y herramientas open source [Comunidad Linux,2014; OISSG, 2012; GNU, 2014], comerciales o privadas [Tenable Network Security, 2014], dado que necesita imitar un ataque de la mejor manera y con el máximo nivel posible [Coronel Suarez, 2016; Hurtado Sandoval et al, 2016; López Alvarez, 2016; López Vallejo, 2017]. Para eso, tendrá que emplear todos los recursos de inteligencia que tenga a su alcance, utilizar al extremo sus conocimientos, poder de deducción y análisis mediante el razonamiento y así determinar qué es lo mejor que puede intentar, cómo, dónde y con qué. Por ejemplo, saber si un pequeño dato, por más chico o insignificante que parezca, le será útil y cómo proseguir gracias a él. Continuamente se deberá enfrentar a etapas que le demanden la mayoría de estas aptitudes [Tori, 2008].

- Definir patrones de conducta y acción.
- Hacer relevamientos pasivos de información.
- Interpretar y generar código y cifrado de datos.
- Descubrir manualmente descuidos en el objetivo.
- Descubrir vulnerabilidades presentes de todo el escenario técnico.
- Proyectarse sobre la marcha en modo abstracto, táctica y estratégicamente.

- Ser exhaustivo, pero a la vez saber cuándo es el momento de recurrir a la distensión para no agotar la mente.

Ahora bien, estas etapas deben realizarse en un marco de control, gestión y supervisión constante la cual otorgue tranquilidad y seguridad tanto al profesional que se "coloca" en los pies del criminal como a la organización en su totalidad [Tamayo Veintimilla, 2016; Onofa Calvopiña et al, 2016; Paillacho Pozoet al, 2016].

3.2. PROBLEMA ABIERTO

Luego de lo expuesto anteriormente, el problema abierto identificado es la necesidad de ordenar, estructurar y organizar el proceso de testeo por hacking ético llevado a cabo dentro del proceso mismo de testing general de software.

Diversos autores mencionan la necesidad de la planificación y ordenamiento en las acciones a llevar adelante en el testeo de software.

El proceso a llevar a cabo por el profesional de seguridad informática debe ser planificado con antelación. Todos los aspectos técnicos, de gestión y estratégicos deben estar sumamente cuidados. La gran debilidad encontrada es allí donde hará foco este trabajo de Tesis,

3.3. SUMARIO DE INVESTIGACIÓN

De lo anteriormente expuesto aparecen las siguientes preguntas de investigación:

Pregunta 1: ¿Se puede plantear un modelo de proceso que permita planificar, ejecutar y evaluar las vulnerabilidades explotables de un producto de software desarrollado en el contexto de un proyecto, ubicando a este modelo dentro del proceso mismo de testeo de un sistema?

Pregunta 2: De ser posible la pregunta anterior
¿Cómo dividir el proceso en fases de dicho proceso, las etapas vinculadas a cada fase y las actividades a llevar a cabo por los profesionales de seguridad?

Pregunta 3: ¿Se pueden clasificar y medir los tipos y grados de vulnerabilidades?

Se proponen soluciones a los interrogantes planteados y su correspondiente validación en los capítulos siguientes.

4. SOLUCIÓN

En este capítulo se presenta: un modelo de proceso de aplicación técnica de Hacking Ético (sección 4.1), del cual se abordan las cuestiones generales de mayor relevancia (sección 4.1.1), se presenta la propuesta de dicho modelo (sección 4.1.2) la que se describe a partir de su estructura general (sección 4.1.2.1). Luego se explican en detalle las tres fases que componen el modelo (fase de Planificación de testeo, fase de Ejecución de testeo y fase de Mantenimiento), junto con las actividades que deben realizarse para llevar a cabo estas fases, los insumos de las mismas, métricas a obtener y los productos que se adquieren con la implementación de las mencionadas tareas (sección 4.1.2.2).

4.1. MODELO DE PROCESO DE APLICACIÓN TECNICA DE HACKING ETICO

En esta sección se presenta una propuesta de modelo de proceso de Testeo por Hacking Ético estructurada en dos partes: generalidades (sección 4.1.1), Propuesta del Modelo de Proceso de Testeo por Hacking Ético (sección 4.1.2).

4.1.1. Generalidades

Luego del análisis realizado en el capítulo anterior correspondiente a la Descripción del Problema, se citará nuevamente el problema que se aborda en este trabajo de Tesis, recordando que el mismo se focaliza en el “caos” e “informalidad” presente en la aplicación de herramientas y métodos para la aplicación de hacking ético dentro del proceso mismo de Testeo. Esto genera aplicaciones más vulnerables y sometidas a un mantenimiento casi constante de penetración-parche.

El proceso a llevar a cabo por el profesional de seguridad informática debe ser planificado con antelación. Todos los aspectos técnicos, de gestión y estratégicos deben estar sumamente cuidados. La planificación es importante para cualquier tipo de prueba que se lleve adelante, ya sea desde un simple análisis de contraseña a una prueba de penetración completa en una aplicación web [Mayorga Jácome et al, 2015; Santos Castañeda, 2016; Onofa Calvopiña et al, 2016; López Vallejo, 2017].

Estas etapas deben realizarse en un marco de control, gestión y supervisión constante la cual otorgue tranquilidad y seguridad tanto al profesional que se "coloca" en los pies del criminal como a la organización en su totalidad [Tamayo Veintimilla, 2016; Onofa Calvopiña et al, 2016; Paillacho Pozoet al, 2016], es aquí donde entra la solución que se propone en esta Tesis.
La cual consiste en desarrollar e incorporar un método de hacking ético para la evaluación de vulnerabilidades dentro del procedimiento mismo de Testeo de un sistema. Suministrando, de esta manera, a los encargados de testing en sectores de Seguridad Informática un grupo de actividades y herramientas que les brinde el soporte necesario para poder prevenir los problemas que en la actualidad son de creciente interés por las pérdidas económicas que conllevan.

4.1.2. Propuesta del Modelo de Proceso de Testeo por Hacking Ético

En esta sección se describe la estructura general y ordenada del proceso de testeo mediante hacking ético (sección 4.1.2.1) junto con su modo de funcionamiento y detalle de etapas, actividades, insumos y productos (sección 4.1.2.2). Además se detallan todos los productos generados en cada etapa (sección 4.1.2.3) y las métricas que se obtienen luego de la etapa de ejecución (sección 4.1.2.4)

4.1.2.1. Estructura General del Proceso de Testeo por Hacking Ético

El mecanismo de testeo mediante Hacking ético se realiza mediante 3 fases que se integran en el Proceso de Testeo general de software.

- Una primera fase de ***Planificación***, cuyo objetivo se centra en la documentación, modelado, ordenamiento y planeamiento de las pruebas a las que se someterá el software desarrollado.
- Una segunda fase de ***Ejecución***, donde el propósito consiste en la prueba ordenada y fundamentada con herramientas específicas buscando lograr obtener un software de calidad y libre de vulnerabilidades críticas.
- Y por último una tercera fase de ***Mantenimiento***, el cual tiene como objetivo someter, periódicamente, al software desarrollado e implementado a pruebas de vulnerabilidades.

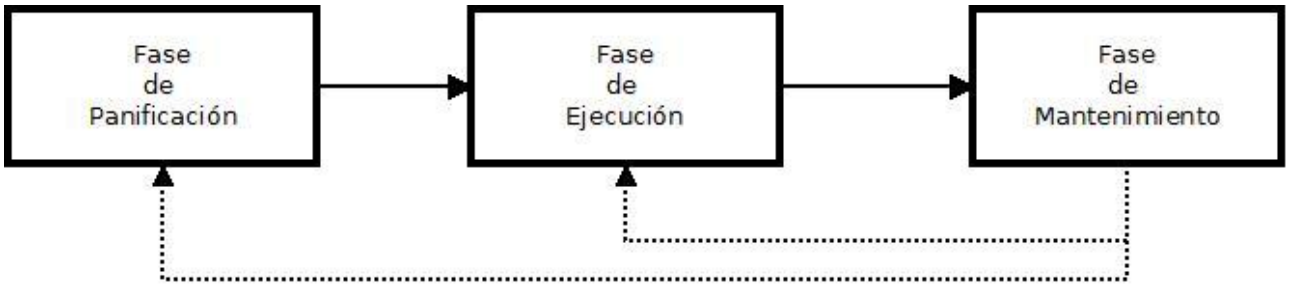

Figura 14. Estructura General del "Proceso de Testeo por Hacking Ético".

Este proceso transcurre dentro del procedimiento mismo de Testeo de un sistema, siendo aplicable a cualquier modelo de desarrollo de software. Teniendo como punto de partida los requisitos y las funcionalidades desarrolladas y probadas en su totalidad.

Proporcionando como salida un informe final de testeo de hacking ético (IFTHE), el cual resumirá los riesgos, criticidades, casos erróneos y consejos a seguir para el software puesto a prueba.

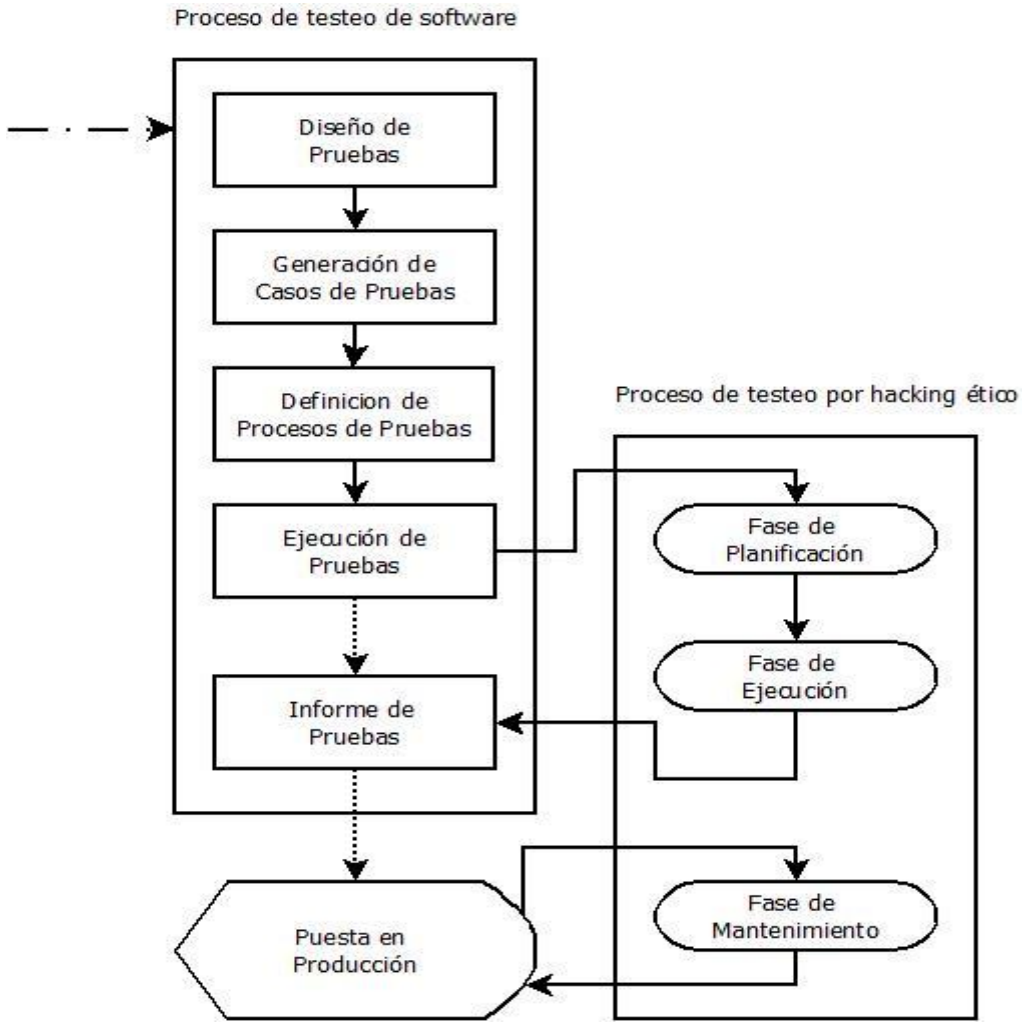

Figura 15. Proceso de Testeo por Hacking Ético dentro de la fase de Pruebas

En este trabajo aparecerán descriptas diferentes herramientas que al momento de escribir esta tesis, se encuentran en el auge de su uso. El proceso planteado debe trascender a las herramientas que se utilicen en cada etapa. Y por lo que es sumamente necesario que al momento de aplicar este proceso se realice una revisión de las herramientas en auge para la ejecución del mismo.
Algo similar ocurre con los listados de vulnerabilidades presentados, los cuales son los expuestos por la OWASP al momento de escritura de la tesis, al comenzar la fase de planificación debe revisarse la última documentación presentada por esta organización.

4.1.2.2. Componentes detallados del Proceso de Testeo por Hacking Ético

El proceso se compondrá de tres fases principales, las cuales se conforman mediante etapas, donde se encuentran agrupadas las tareas o actividades que mediante herramientas y elementos de entrada, proporcionan los correspondientes productos de salida.

Siguiendo estos lineamientos, el modelo de proceso propuesto para el Testeo por Hacking Ético se encuentra dividido en tres fases principales: Fase de Planificación del testeo, Fase de Ejecución del testeo y Fase de Mantenimiento.

La Fase de Planificación del Testeo, tiene como objetivo interpretar la aplicación, identificar amenazas y vulnerabilidades, planificar y calendarizar las pruebas, entre otras acciones. Posee como entradas principales los Análisis de Requerimientos previos al desarrollo y la Documentación de la aplicación a someter al testeo, generando como producto final el Plan Integral de Testeo por Hacking Ético (PITHE).

Esta fase se encuentra conformada por cuatro etapas: Recopilación de información, Análisis de vulnerabilidades, Modelado de amenazas y Planeamiento de pruebas.
En la etapa de Recopilación de Información, se encuentran las tareas correspondientes al entendimiento e interpretación de la aplicación que se someterá al testeo, por lo que se plantean las siguientes actividades: Documentación del tipo de aplicación, Establecimiento de límites de componentes a someter a pruebas, Delimitación del alcance de la prueba e Identificación de primeros riesgos asociados. Los insumos requeridos en esta etapa son el Análisis de requerimientos y la Documentación de la aplicación, dando como producto de salida el Informe de Dominio (ID).

Dentro de la etapa de Análisis de vulnerabilidades, aparecen las actividades que ayudan a identificar las vulnerabilidades a las que la aplicación sería más propicia. Las tareas que la conforman son: Listado de vulnerabilidades, Determinación de probabilidad de vulnerabilidad y Realización de ranking de vulnerabilidades más propicias. Como insumo se requiere el producto de la etapa anterior, el informe de dominio, junto con el Análisis de requerimientos y la Documentación de la aplicación. Obteniendo como producto el Informe de vulnerabilidades (IV).

Continúa la etapa la llamada Modelado de Amenazas, donde se realiza una descomposición detallada y minuciosa de la aplicación, identificando y documentando la criticidad de las vulnerabilidades.

Para llevar esto a cabo se plantean las siguientes actividades: Identificación de la información sensible dentro de la aplicación, Descripción de la arquitectura, Descomposición de la aplicación, Identificación de las vulnerabilidades críticas, Documentación de las vulnerabilidades y Asignación de prioridades de las vulnerabilidades críticas, las cuales tienen como insumos los mismos que su etapa antecesora sumando, además, el producto de aquella, el Informe de Vulnerabilidades. Los productos obtenidos en esta etapa son el Informe de Amenazas y las Prioridades de Criticidad.

La ultima tarea de esta primera fase es el Planeamiento de Pruebas, la cual tiene como objetivo el ordenamiento, armado y planificación de las pruebas a realizar, esta es constituida por las actividades: Planificación de pruebas, Armado de casos de pruebas, Establecimiento de criterios de aprobación y rechazo y de tipo de acción, Determinación de grados de criticidad, Calendarización de hitos, Estimación de esfuerzos y por último, la Elaboración del plan integral de testeo por hacking ético. Esta etapa al ser la final de fase, posee como insumos todos los productos de las tareas antecesoras, dando como productos finales: Casos de prueba, Calendario de pruebas y esfuerzo e Informe de Criterios.

Habiendo cumplido con todas estas tareas y armado de productos, obtendremos como salida final el antes mencionado Plan Integral de Testeo por Hacking Ético (PITHE).
Siendo precisamente el input para la fase siguiente que es la llamada Fase de Ejecución del Testeo, la cual tiene como objetivo principal la realización de las pruebas de forma ordenada,

pautada y regida por las herramientas que se proponen para llevar adelante y arribar de la mejor forma al Informe Final del Testeo (IFTHE).

Para lograr lo descripto anteriormente, esta fase es compuesta por cinco etapas complementarias: Reconocimiento, Escaneo, Ganancia de Acceso, Mantenimiento de Acceso y Eliminación de pruebas.
Estas etapas son las 5 etapas recomendadas en la mayoría de las bibliografías que mencionan al Hacking Ético.

En la actividad de Reconocimiento, se busca entender, descubrir y recopilar toda la información que se estará exponiendo al exterior sobre la aplicación. Para lograr esto, se plantean estas actividades: Recopilación de información inicial, Determinación del tamaño de la red, Identificación de las máquinas activas, Descubrimiento de puertos abiertos y puntos de acceso, Rastreo del sistema operativo y Mapeo de la red completa, dando como producto el Informe de Arquitectura de Red.

La etapa que continúa es la llamada Escaneo, donde se busca descubrir y determinar todas las características propias del sistema, las tareas que la conforman son: Detección de sistemas vivos en la red y Descubrimiento de puertos activos, sistema operativo, servicios en ejecución y presentes en el sistema y direcciones IPs, utilizando como documentación de entrada la salida de la tarea anterior y generando el Informe de escaneo del sistema.
Esta etapa nos otorgará el listado de los accesos que se encuentran expuestos.
Luego del escaneo, se deberá llevar adelante la Ganancia de acceso, donde se plantean las siguientes tareas: Escaneo de direcciones, Investigación de puertos y Explotación de servicios y sistemas. Como ingreso a la etapa se tienen el Plan Integral de Testeo por Hacking Ético y los Informes de arquitectura de red y escaneo del sistema; y por salida se obtendrá el Informe de accesos ganados.

En esta etapa se intenta determinar en qué medida nuestro sistema se encuentra abierto a posibles accesos indeseados.

Luego de ganar los accesos, un hacker buscará mantenerlos, por esto tenemos la etapa llamada: Mantenimiento de acceso, la cual es integrada por las tareas: Mantenimiento de acceso, Aseguramiento de acceso exclusivo, Carga, descarga y manipulación de datos, aplicaciones y

configuraciones en el sistema y Utilizar el sistema para lanzar más ataques. Necesitando como entrada todos los informes generados hasta esta etapa, para poder dar como producto el Informe de Accesos Mantenidos.

Por último aparece la etapa denominada Eliminación de Pruebas, la cual tiene como objetivo probar que nuestro sistema, en el caso de haber sido penetrado, no permitirá el ocultamiento de huellas y la apertura para futuros accesos al sistema.

Para esto tiene como tareas las siguientes: Ocultamiento de actos maliciosos, Continuación del acceso al sistema víctima, Sobre escritura del servidor, los sistemas y el registro de aplicaciones y el armado del informe final de Testeo por Hacking Ético.
Todas estas etapas, tareas, actividades y herramientas, nos llevan a obtener el Informe Final de Testeo por Hacking Ético (IFTHE).

Este informe contiene la información de que tan crítico, vulnerable o protegido esta nuestro sistema próximo a salir a Producción.

Es en esta etapa donde se recurre a la medición de la vulnerabilidad de nuestro sistema aplicando las métricas de vulnerabilidad explicadas más adelante.

Como complemento final aparece la última, y no menos importante, Fase de Mantenimiento, la cual vela por que los sistemas que no son vulnerables en su puesta productiva, no sean posibles objetivos ante nuevas amenazas. Para lograr esto se proponen 3 etapas.

En primer lugar la etapa de Control de Vulnerabilidades, busca constantemente actualizarse con las vulnerabilidades nuevas que surgen y mide que tan críticas son en nuestras aplicaciones. Para esto tiene las siguientes actividades ordenadas: Listado de vulnerabilidades nuevas, Determinación de grado de aplicabilidad. Para estos análisis, son necesarios los siguientes insumos: Informe Final de Testeo por Hacking Ético, Listado de Actualización de Vulnerabilidades y Documentación de la aplicación, generando como producto el Informe de Vulnerabilidades Criticas.

La segunda etapa es la más importante de esta fase, esta es la Determinación de Criterios, en ella las tareas Determinación del grado de criticidad, Establecimiento de criterios de aprobación y

rechazo y Listado del ranking de vulnerabilidades más riesgosas, buscan determinar que tan crítico y con qué prioridad se deben corregir las vulnerabilidades. Junto con el Informe de vulnerabilidades y la Documentación de la aplicación, se puede obtener el Listado Criterios de Aprobación y Rechazo.

Por último aparece la etapa de Verificación y Validación, donde las tareas a realizarse son: Ejecución de pruebas de verificación, Comparación con criterios establecidos, Determinación de la criticidad del riesgo, Establecimiento de las acciones a tomar, Validación de los resultados y la Elaboración del informe.

Esta etapa final tiene como insumos todos los informes y documentación disponible y da como producto el Informe de Verificación y Validación.

A continuación se ilustra este proceso completo con sus fases, etapas y actividades.

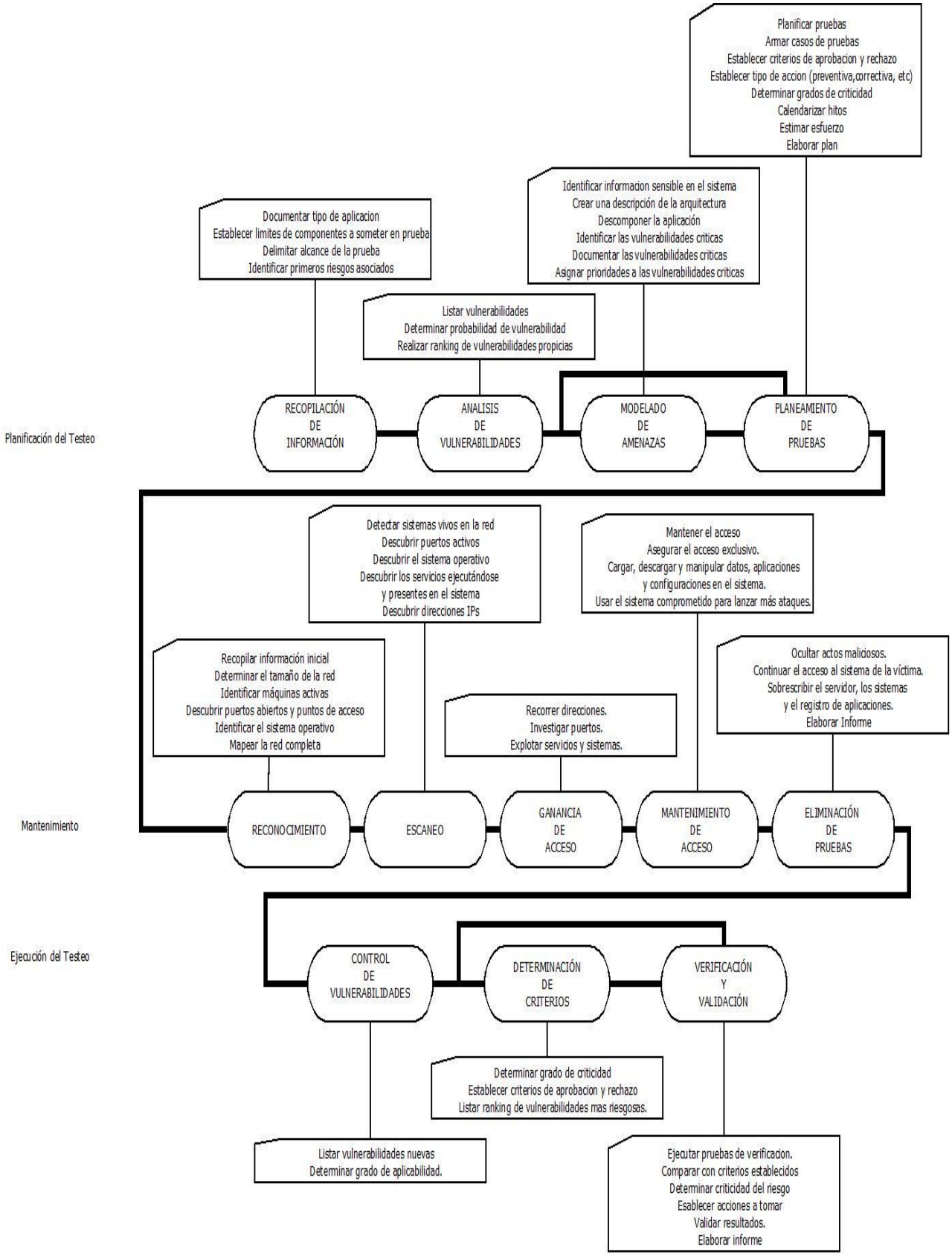

Figura 16. Estructura General del "Proceso de Testeo por Hacking Ético" con sus fases, etapas y actividades.

La figura a continuación ilustra este proceso completo con sus fases y los respectivos insumos y productos:

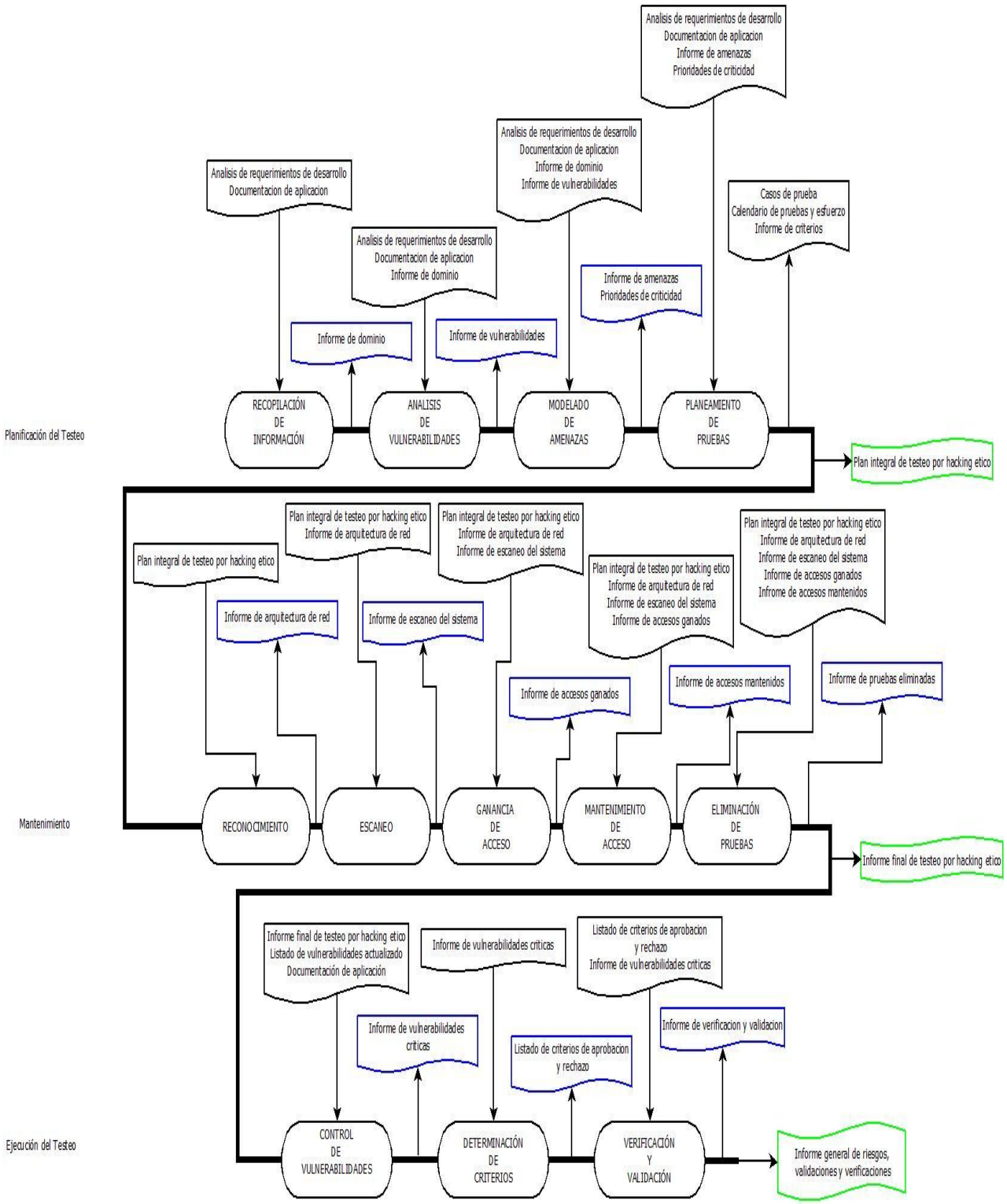

Figura 17. Estructura General del "Proceso de Testeo por Hacking Ético" con sus fases y los elementos de entrada y salida

Proceso de testeo de hacking ético			
FASE I: PLANIFICACION DE TESTEO			
ETAPAS	**ACTIVIDADES**	**INSUMOS**	**PRODUCTOS**
RECOPILACION DE INFORMACION	1) Documentar tipo de aplicación 2) Establecer límites de componentes a someter en prueba 3) Delimitar alcance de la prueba 4) Identificar primeros riesgos asociados	• Análisis de requerimientos de desarrollo • Documentación de aplicación	• Informe de dominio
ANALISIS DE VULNERABILIDADES	5) Listar vulnerabilidades 6) Determinar probabilidad de vulnerabilidad 7) Realizar ranking de vulnerabilidades propicias	• Análisis de requerimientos de desarrollo • Documentación de aplicación • Informe de dominio	• Informe de vulnerabilidades
MODELADO DE AMENAZAS	8) Identificar información sensible en el sistema 9) Crear una descripción de la arquitectura 10) Descomponer la aplicación 11) Identificar las vulnerabilidades criticas 12) Documentar las vulnerabilidades criticas 13) Asignar prioridades a las vulnerabilidades criticas	• Análisis de requerimientos de desarrollo • Documentación de aplicación • Informe de dominio • Informe de vulnerabilidades	• Informe de amenazas • Prioridades de criticidad
PLANEAMIENTO DE PRUEBAS	14) Planificar pruebas 15) Armar casos de pruebas 16) Establecer criterios de aprobación y rechazo 17) Establecer tipo de acción (preventiva, correctiva, etc.) 18) Determinar grados de criticidad 19) Calendarizar hitos 20) Estimar esfuerzo 21) Elaborar plan	• Análisis de requerimientos de desarrollo • Documentación de aplicación • Informe de amenazas • Prioridades de criticidad	• Casos de prueba • Calendario de pruebas y esfuerzo (Gantt • Informe de criterios
PRODUCTO FINALDE LA FASE: PLAN INTEGRAL DE TESTEO POR HACKING ETICO			
FASE II: EJECUCION DEL TESTEO			
ETAPAS	**ACTIVIDADES**	**INSUMOS**	**PRODUCTOS**
RECONOCIMIENTO	22) Recopilar información inicial 23) Determinar el tamaño de la red 24) Identificar máquinas activas 25) Descubrir puertos abiertos y puntos de acceso 26) Identificar el sistema operativo 27) Mapear la red completa	• Plan integral de testeo por hacking ético	• Informe de arquitectura de red
ESCANEO	28) Detectar sistemas vivos en la red 29) Descubrir puertos activos 30) Descubrir el sistema operativo 31) Descubrir los servicios ejecutándose y presentes en el sistema 32) Descubrir direcciones IPs	• Plan integral de testeo por hacking ético • Informe de arquitectura de red	• Informe de escaneo del sistema
GANANCIA DE ACCESO	33) Recorrer direcciones 34) Investigar puertos 35) Explotar servicios y sistemas	• Plan integral de testeo por hacking ético • Informe de arquitectura de red • Informe de escaneo del sistema	• Informe de accesos ganados
MANTENIMIENTO DE ACCESO	36) Mantener el acceso 37) Asegurar el acceso exclusivo 38) Cargar, descargar y manipular datos, aplicaciones y configuraciones en el sistema 39) Usar el sistema comprometido para lanzar más ataques	• Plan integral de testeo por hacking ético • Informe de arquitectura de red • Informe de escaneo del sistema • Informe de accesos ganados	• Informe de accesos mantenidos
ELIMINACION DE PRUEBAS	40) Ocultar actos maliciosos. 41) Continuar el acceso al sistema de la víctima. 42) Sobrescribir el servidor, los sistemas y el registro de aplicaciones. 43) Elaborar Informe	• Plan integral de testeo por hacking ético • Informe de arquitectura de red • Informe de escaneo del sistema • Informe de accesos ganados • Informe de accesos mantenidos	• Informe de pruebas eliminadas
PRODUCTO FINALDE LA FASE: INFORME FINAL DE TESTEO POR HACKING ETICO			
FASE III: MANTENIMIENTO			
ETAPAS	**ACTIVIDADES**	**INSUMOS**	**PRODUCTOS**
CONTROL DE VULNERABILIDADES	44) Listar vulnerabilidades nuevas 45) Determinar grado de aplicabilidad	• Informe final de testeo por hacking ético • Listado de vulnerabilidades actualizado	• Informe de vulnerabilidades criticas
DETERMINACION DE CRITERIOS	46) Determinar grado de criticidad 47) Establecer criterios de aprobación y rechazo 48) Listar ranking de vulnerabilidades mas riesgosas	• Informe de vulnerabilidades criticas	• Listado de criterios de aprobación y rechazo
VERIFICACION Y VALIDACION	49) Ejecutar pruebas de verificación 50) Comparar con criterios establecidos 51) Determinar criticidad del riesgo 52) Establecer acciones a tomar 53) Validar resultados 54) Elaborar informe	• Listado de criterios de aprobación y rechazo • Informe de vulnerabilidades criticas	• Informe de verificación y validación
PRODUCTO FINALDE LA FASE: INFORME GENERAL DE RIESGOS, VALIDACIONES Y VERIFICACIONES			

Tabla 2. Detalle del Proceso de Testeo por Hacking Ético con sus fases y los elementos de entrada y salida

4.1.2.3. Productos detallados de las etapas del proceso de Testeo por Hacking Ético

A continuación se detalla cada uno de los productos que se generarán en cada una de las fases y etapas mencionadas anteriormente.

Fase de Planificación

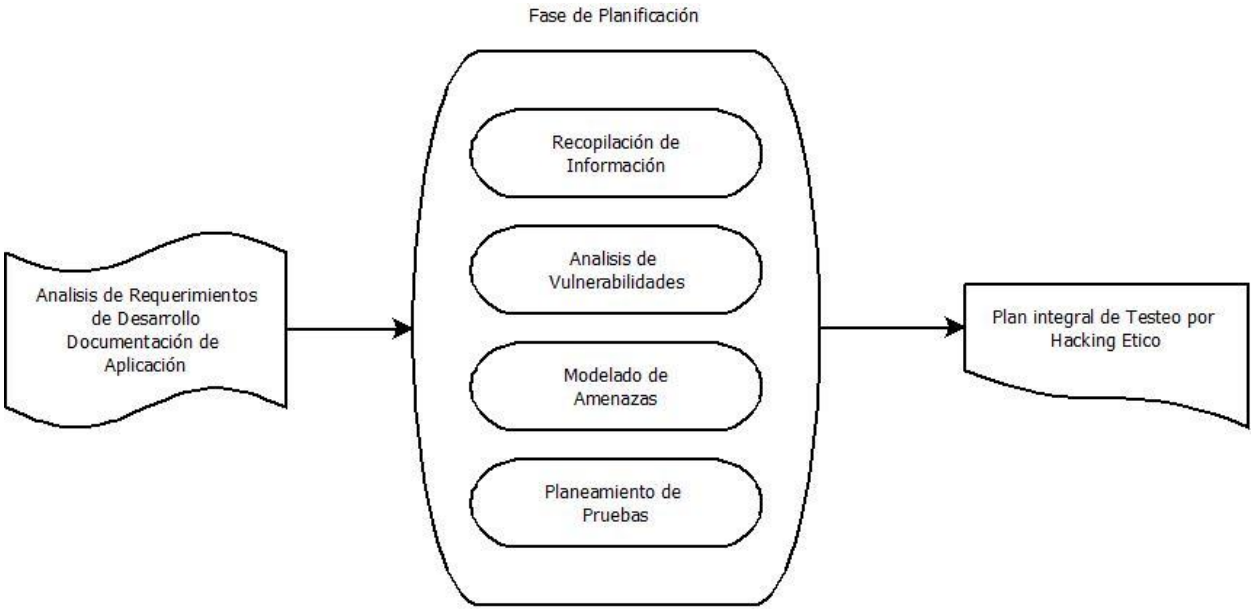

Figura 18. Elementos de entrada y salida de la fase de Planificación.

ENTRADAS

- Análisis de Requerimientos de Desarrollo
- Documentación de la Aplicación

SALIDA

- Plan Integral de Testeo por Hacking Ético - PITHE

- Etapa de Recopilación de Información

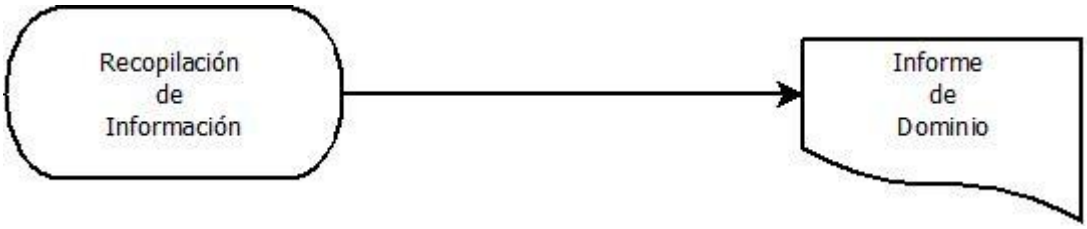

Figura 19. Producto generado por la etapa de Recopilación de Información**.**

Informe de dominio - ID

a) Portada
    a. Nombre proyecto/aplicación
    b. Histórico de versiones
    c. Responsables del documento
b) Índice
c) Objetivo y alcance del proyecto
    a. Responsabilidades
    b. Objetivo del proyecto
    c. Funcionalidades afectadas
    d. Descripción funcional
d) Evaluación de posibles riesgos asociados.
    a. Preguntas
    b. Nivel de riesgo
    c. Respuesta

- Etapa de Análisis de Vulnerabilidades

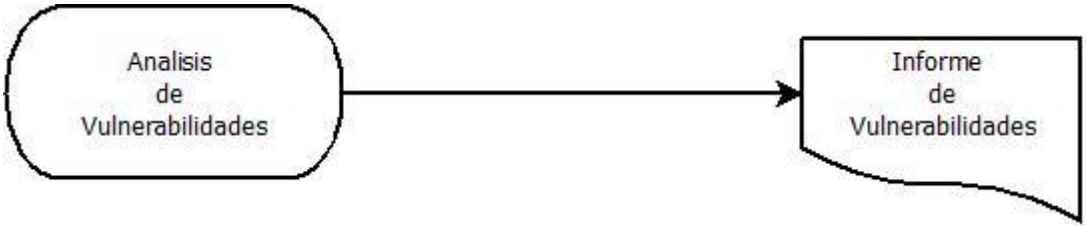

Figura 20. Producto generado por la etapa de Análisis de Vulnerabilidades**.**

Informe de vulnerabilidades - IV

a) Portada
    a. Nombre proyecto/aplicación
    b. Histórico de versiones
    c. Responsables del documento
b) Índice
c) Listado de vulnerabilidades Web
    a. Actualización
    b. Titulo vulnerabilidad
    c. Descripción vulnerabilidad
    d. Sistema propicio
d) Listado de vulnerabilidades Mobile
    a. Actualización
    b. Titulo vulnerabilidad
    c. Descripción vulnerabilidad
    d. Sistema propicio
e) Otras vulnerabilidades detectadas
    a. Actualización
    b. Titulo vulnerabilidad
    c. Descripción vulnerabilidad
    d. Sistema propicio

- Etapa de Modelado de Amenazas

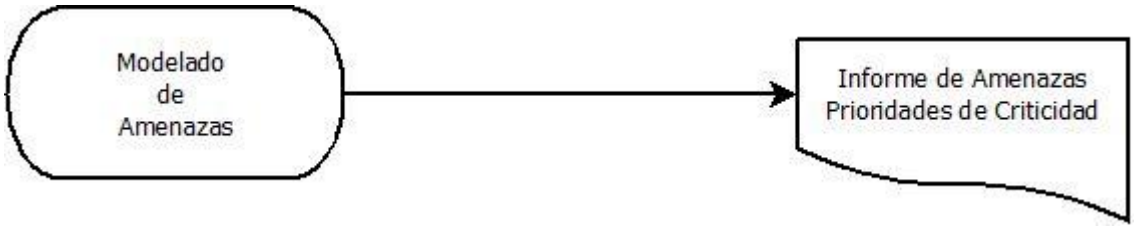

Figura 21. Productos generados por la etapa de Modelado de Amenazas.

Informe de amenazas - IA

a) Portada
    a. Nombre proyecto/aplicación
    b. Histórico de versiones
    c. Responsables del documento
b) Índice
c) Listado de amenazas Web junto a nivel de criticidad y aplicabilidad
    a. Actualización
    b. Título de amenaza
    c. Criticidad
    d. Aplicabilidad
d) Listado de amenazas Mobile junto a nivel de criticidad y aplicabilidad
    a. Actualización
    b. Título de amenaza
    c. Criticidad
    d. Aplicabilidad
e) Listado de otras amenazas detectadas
    a. Actualización
    b. Título de amenaza
    c. Criticidad
    d. Aplicabilidad

Prioridades de criticidad - PC

*Este documento es anexo del informe de amenazas, donde se colocan las amenazas que evaluadas anteriormente se encuentran entre nivel 3 y 5.

a) Portada
    a. Nombre proyecto/aplicación
    b. Histórico de versiones
    c. Responsables del documento
b) Índice
c) Prioridades de amenazas Web
    a. Actualización
    b. Prioridad de criticidad
    c. Título de amenaza
    d. Criticidad
    e. Aplicabilidad
d) Prioridades de amenazas Mobile
    a. Actualización
    b. Prioridad de criticidad
    c. Título de amenaza
    d. Criticidad
    e. Aplicabilidad
e) Prioridades de otras amenazas detectadas
    a. Actualización
    b. Prioridad de criticidad
    c. Título de amenaza
    d. Criticidad
    e. Aplicabilidad

- Etapa de Planeamiento de Pruebas

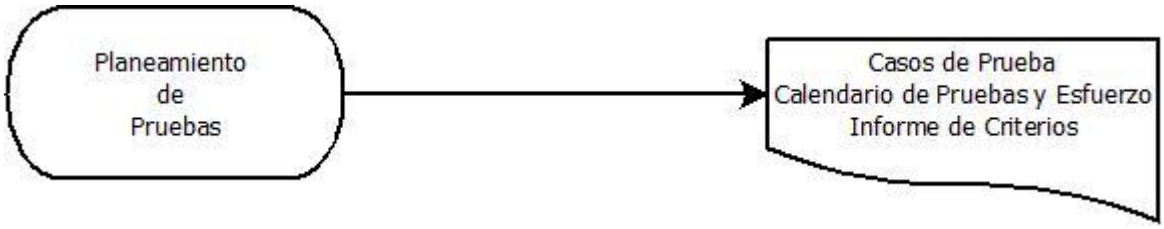

Figura 22. Productos generados por la etapa de Planeamiento de Pruebas**.**

Casos de prueba - CP

a) Portada
    a. Nombre proyecto/aplicación
    b. Histórico de versiones
    c. Responsables del documento
b) Índice
c) Casos de prueba
    a. Código de caso
    b. Título del caso
    c. Condiciones
    d. Datos de entrada
    e. Resultados esperados
    f. Etapa de fase de ejecución
d) Criticidad
    a. Código de caso
    b. Nivel de criticidad
    c. Observaciones
e) Estimación de esfuerzo
    a. Código de caso
    b. Esfuerzo aproximado
    c. Observaciones

Calendario de pruebas y esfuerzo - CPE

a) Portada
    a. Nombre proyecto/aplicación
    b. Histórico de versiones
    c. Responsables del documento
b) Índice
c) Calendarización
    a. Código caso
    b. Inicio
    c. Duración (días)
    d. Fin
d) Diagrama Gantt

Informe de criterios - IC

a) Portada
    a. Nombre proyecto/aplicación
    b. Histórico de versiones
    c. Responsables del documento
b) Índice
c) Listado de criterios de aprobación
    a. Criterios
    b. Aprobación
d) Listado de criterios de rechazo
    a. Criterios
    b. Rechazo

Fase de Ejecución

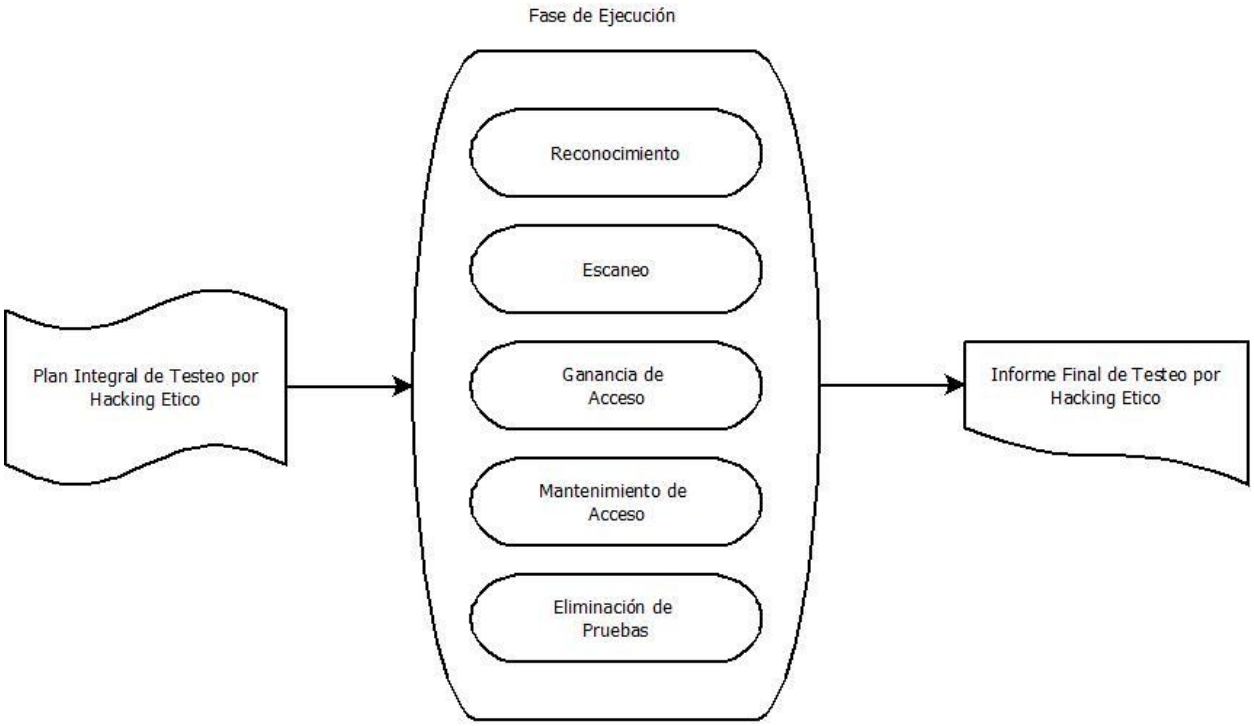

Figura 23. Elementos de entrada y salida de la fase de Ejecución.

ENTRADA

- Plan Integral de Testeo por Hacking Ético - PITHE

SALIDA

- Informe Final de Testeo por Hacking Ético - IFTHE

- Etapa de Reconocimiento

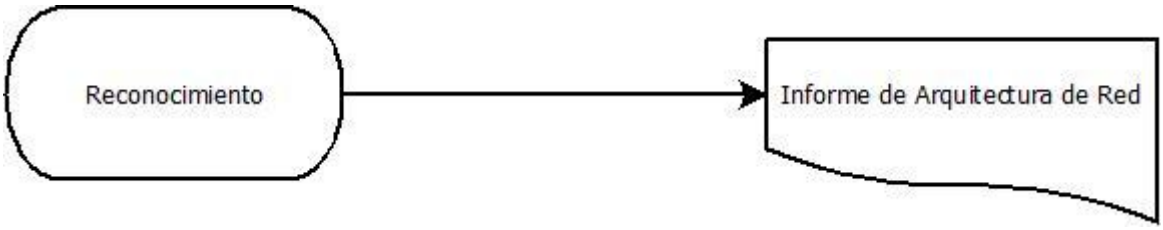

Figura 24. Producto generado por la etapa de Reconocimiento.

Informe de arquitectura de red - IAR

a) Portada
    a. Nombre proyecto/aplicación
    b. Histórico de versiones
    c. Responsables del documento
b) Índice
c) Sistema operativo
d) Maquinas activas
e) Listado de puertos y puntos de acceso
    a. Puerto
    b. Descripción
f) Dns
g) Dominios activos
h) Casos de prueba particulares
    a. Código de caso
    b. Título del caso
    c. Condiciones
    d. Datos de entrada
    e. Resultados esperados
    f. Etapa de fase de ejecución
    g. Resultado

- Etapa de Escaneo

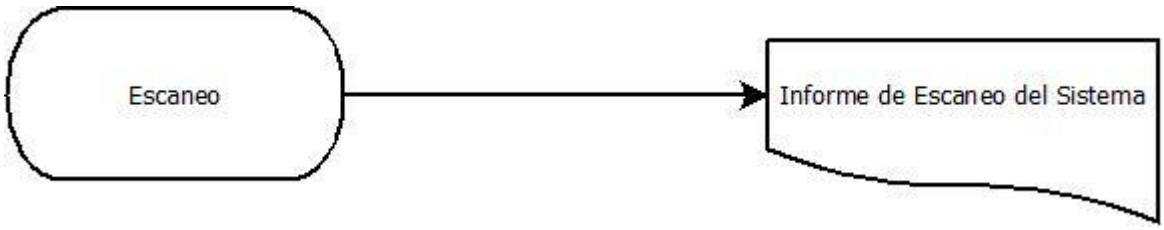

Figura 25. Producto generado por la etapa de Escaneo.

Informe de escaneo del sistema - IES

a) Portada
    a. Nombre proyecto/aplicación
    b. Histórico de versiones
    c. Responsables del documento
b) Índice
c) Listado de puertos activos y puntos de acceso
    a. Puerto
    b. Tipo
    c. Estado
    d. Servicio
d) Dns y dominios
    a. Dns
    b. Proveedor
    c. Ip
e) Casos de prueba particulares
    a. Código de caso
    b. Título del caso
    c. Condiciones
    d. Datos de entrada
    e. Resultados esperados
    f. Etapa de fase de ejecución
    g. Resultado

- Etapa de Ganancia de Acceso

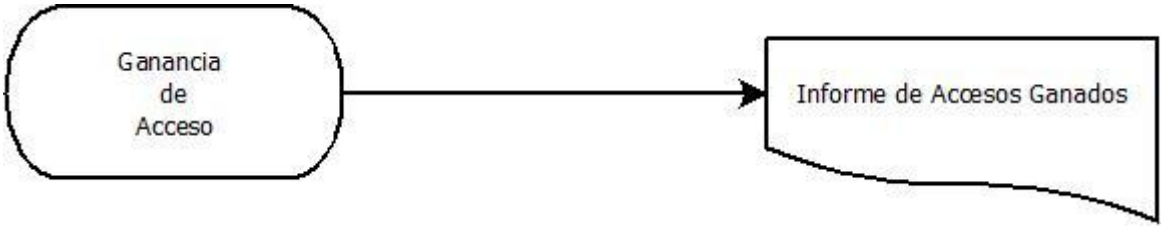

Figura 26. Producto generado por la etapa de Ganancia de Acceso.

Informe de accesos ganados - IAG

a) Portada
    a. Nombre proyecto/aplicación
    b. Histórico de versiones
    c. Responsables del documento
b) Índice
c) Listado de puertos activos y puntos de acceso
    a. Puerto
    b. Tipo
    c. Estado
    d. Servicio
    e. Accedido
    f. Ganado

- Etapa de Mantenimiento de Acceso

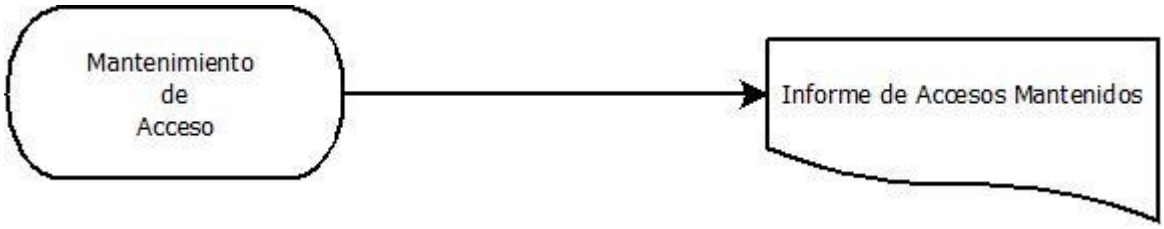

Figura 27. Producto generado por la etapa de Mantenimiento de Acceso.

Informe de accesos mantenidos - IAM

a) Portada
    a. Nombre proyecto/aplicación
    b. Histórico de versiones
    c. Responsables del documento
b) Índice
c) Listado de puertos activos y puntos de acceso
    a. Puerto
    b. Tipo
    c. Estado
    d. Servicio
    e. Accedido
    f. Ganado
d) Esploit instalado
    a. Titulo
    b. Puerto canalizado
    c. Acceso ininterrumpido

- Etapa de Eliminación de pruebas

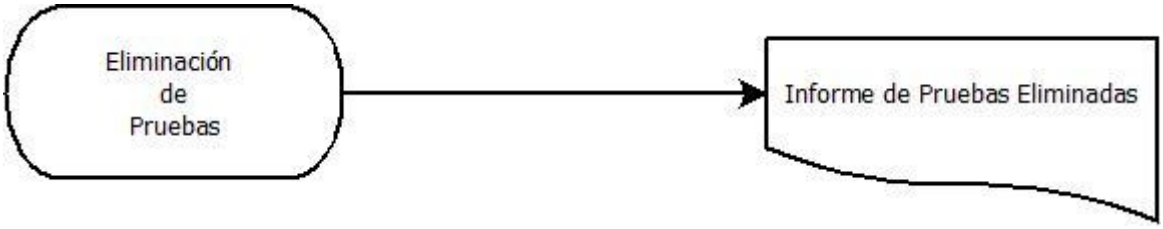

Figura 28. Producto generado por la etapa de Eliminación de Pruebas.

Informe de pruebas eliminadas - IPE

a) Portada
    a. Nombre proyecto/aplicación
    b. Histórico de versiones
    c. Responsables del documento
b) Índice
c) Eliminación de logs
    a. Herramienta utilizada
    b. Ubicación
    c. Objetivo
    d. Resultado
d) Recuperación de logs
    a. Herramienta forense
    b. Objetivo
    c. Resultado

Fase de Mantenimiento

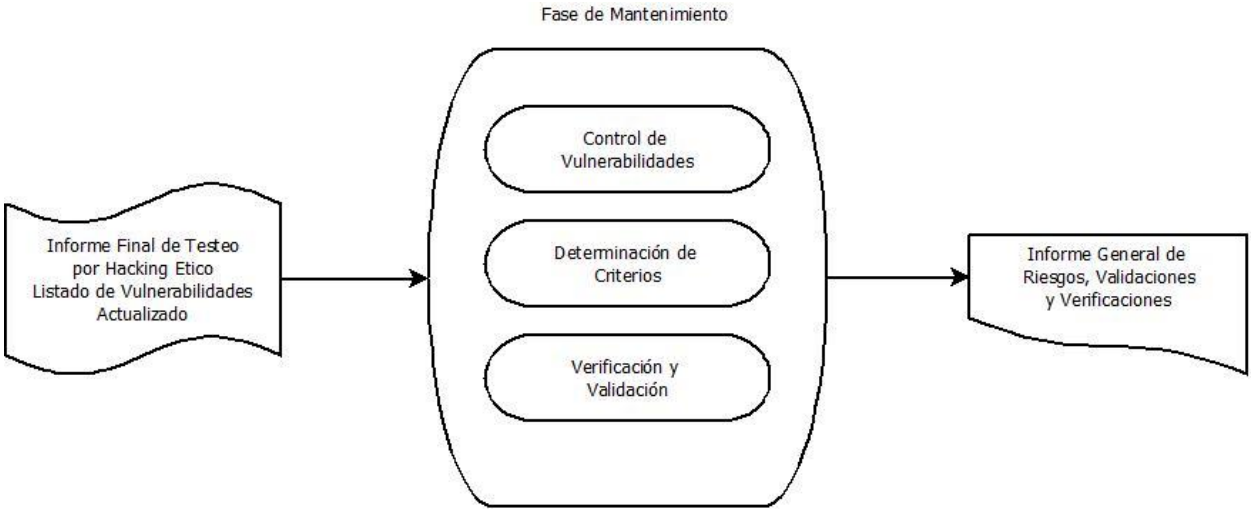

Figura 29. Elementos de entrada y salida de la fase de Mantenimiento.

ENTRADA

- Informe Final de Testeo por Hacking Ético - IFTHE
- Listado de Vulnerabilidades Actualizado

SALIDA

- Informe General de Riesgos, Validaciones y Verificaciones - IGRVV

- Etapa de Control de Vulnerabilidades

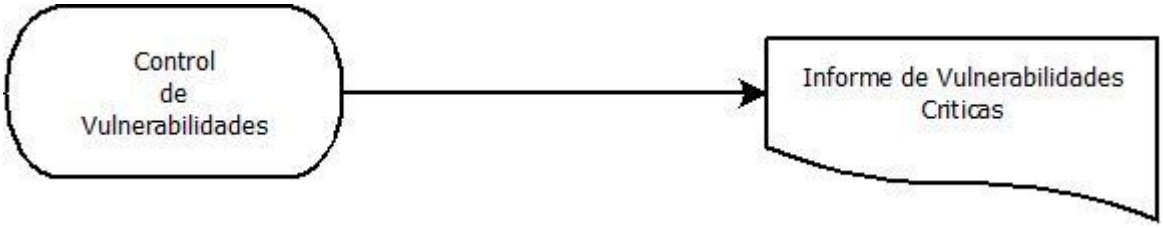

Figura 30. Producto generado por la etapa de Control de Vulnerabilidades.

Informe de vulnerabilidades críticas - IVC

a) Portada
    a. Nombre proyecto/aplicación
    b. Histórico de versiones
    c. Responsables del documento
b) Índice
c) Listado de vulnerabilidades Web
    a. Actualización
    b. Titulo vulnerabilidad
    c. Descripción vulnerabilidad
    d. Sistema propicio
d) Listado de vulnerabilidades Mobile
    a. Actualización
    b. Titulo vulnerabilidad
    c. Descripción vulnerabilidad
    d. Sistema propicio
e) Otras vulnerabilidades detectadas
    a. Actualización
    b. Titulo vulnerabilidad
    c. Descripción vulnerabilidad
    d. Sistema propicio

- Etapa de Determinación de Criterios

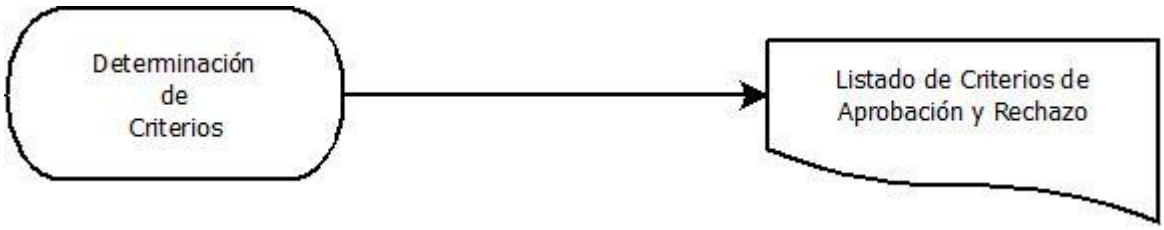

Figura 31. Producto generado por la etapa de Determinación de Criterios.

Listado de criterios de aprobación y rechazo - LCAR

a) Portada
    a. Nombre proyecto/aplicación
    b. Histórico de versiones
    c. Responsables del documento
b) Índice
c) Listado de amenazas Web junto a nivel de criticidad y aplicabilidad
    a. Actualización
    b. Título de amenaza
    c. Criticidad
    d. Aplicabilidad
d) Listado de amenazas Mobile junto a nivel de criticidad y aplicabilidad
    a. Actualización
    b. Título de amenaza
    c. Criticidad
    d. Aplicabilidad
e) Prioridades de amenazas Web
    a. Actualización
    b. Prioridad de criticidad
    c. Título de amenaza
    d. Criticidad
    e. Aplicabilidad
f) Prioridades de amenazas Mobile
    a. Actualización

   b. Prioridad de criticidad
   c. Título de amenaza
   d. Criticidad
   e. Aplicabilidad

g) Listado de criterios de aprobación
   a. Criterios
   b. Aprobación

h) Listado de criterios de rechazo
   a. Criterios
   b. Rechazo

- Etapa de Verificación y Validación

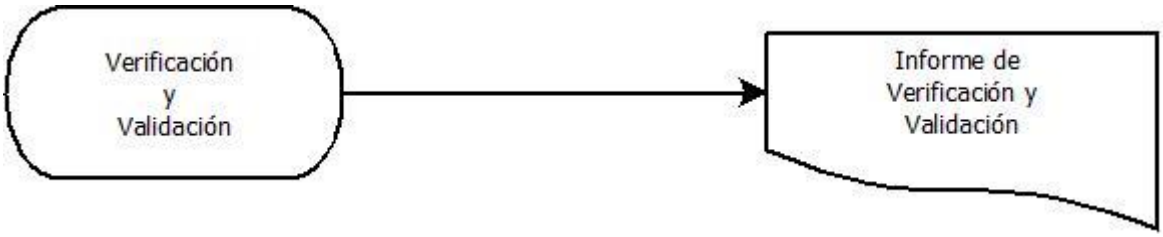

Figura 32. Producto generado por la etapa de Verificación y Validación.

Informe de verificación y validación - IVV

a) Portada
    a. Nombre proyecto/aplicación
    b. Histórico de versiones
    c. Responsables del documento
b) Índice
c) Casos de prueba
    a. Código de caso
    b. Título del caso
    c. Condiciones
    d. Datos de entrada
    e. Resultados esperados
    f. Validación de resultados
d) Acciones a tomar
    a. Código de caso
    b. Acción
    c. Resultado exigido

4.1.2.4. Métricas de vulnerabilidad

Durante la fase de Ejecución del Testeo se plantea el uso de métricas que nos indiquen el grado de vulnerabilidad del sistema al ser desarrollado. Las métricas propuestas son las siguientes:

- Índice de posibilidad de acceso (IPA): es directamente proporcional a la cantidad de accesos ganados (CAG) e inversamente proporcional a la cantidad de puertos activos detectados (PAD). Es un índice que tiene un valor entre 0 y 1, siendo los valores cercanos a 0 los que nos dan una medida de un sistema más seguro y un valor cercano a la unidad un sistema más vulnerable.

$$IPA = \frac{CAG}{PAD} \qquad 0 <= IPA <= 1 \qquad IPA\ \% = \frac{CAG}{PAD} * 100\ \% \qquad IPA << 1$$

Fórmula 1. Índice de posibilidad de acceso

- Índice de acceso real (IAR): directamente proporcional a la cantidad de accesos ganados (CAG) e inversamente proporcional a los puertos activos reales (PAR) del sistema. Es un índice que tiene un valor entre 0 y 1, siendo los valores cercanos a 0 los que nos dan una medida de un sistema más seguro y un valor cercano a la unidad un sistema más vulnerable.

$$IAR = \frac{CAG}{PAR} \qquad 0 <= IAR <= 1 \qquad IAR\ \% = \frac{CAG}{PAR} * 100\ \% \qquad IAR << 1$$

Fórmula 2. Índice de acceso real

- Índice de detección de puertos (IDP): directamente proporcional a la cantidad de puertos activos detectados (PAD) e inversamente proporcional a los puertos activos reales (PAR) del sistema. Es un índice que tiene un valor entre 0 y 1, siendo los valores cercanos a 0 los que nos dan una medida de un sistema más seguro y un valor cercano a la unidad un sistema más vulnerable.

$$IDP = \frac{PAD}{PAR} \qquad 0 <= IDP <= 1 \qquad IDP\ \% = \frac{PAD}{PAR} * 100\ \% \qquad IDP << 1$$

Fórmula 3. Índice de detección de puertos

5. CASO DE VALIDACIÓN 1

En este capítulo se presentan un caso de validación de una aplicación de tipo web con características para la aplicación de las técnicas asociadas a las tareas del modelo de proceso de testeo por hacking ético, a los efectos de implementar las tareas correspondientes a cada una de las fases. Se analizará un sistema de Gestión de archivos.
En la sección 5.1 se aplica la fase de planificación de testeo, en la sección 5.2, se aplica la fase de ejecución del testeo y por ultimo en la sección 5.3 se aplican las actividades correspondientes a la fase de mantenimiento.

5.1. APLICACIÓN DE LAS ACTIVIDADES DE LA FASE DE PLANIFICACION DE TESTEO

En esta sección se aplican al caso de validación las etapas correspondientes a la fase de Planificación: Recopilación de Información (sección 5.1.1), Análisis de Vulnerabilidades (sección 5.1.2), Modelado de Amenazas (sección 5.1.3) y el Planeamiento de Pruebas (sección 5.1.4). Obteniendo como producto final de la fase el Plan Integral de Testeo por Hacking Ético (PITHE) (sección 5.1.5).

A continuación se procede a aplicar las etapas de esta primera fase, siguiendo los pasos especificados en la tabla 4.1 y que se describen con detalle en la figura 4.5 de la sección 4.1.2.2 del capítulo 4. La etapa se inicia con el análisis de requerimientos de desarrollo obtenido en la fase inicial del proceso de software junto con la documentación de la aplicación que se genera durante todo el proceso de desarrollo.

5.1.1. Aplicación de la etapa de Recopilación de Información

La aplicación de la etapa de recopilación de información, permite documentar el tipo de aplicación a testear, establecer límites de componentes a someter en prueba, delimitar el alcance de la prueba e identificar los primeros riesgos asociados. Para llevar a cabo este proceso se cuenta con el Análisis de requerimientos de desarrollo y la Documentación de la aplicación como productos de entrada, y se obtiene el Informe de dominio (ID) como producto de salida.
La figura 33 sintetiza la aplicación de la etapa de recopilación de información para el caso de estudio:

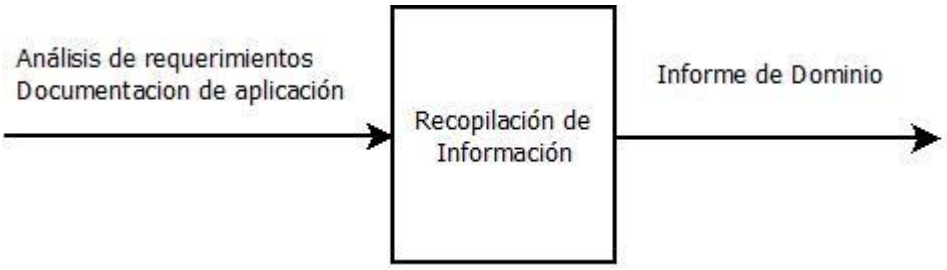

Figura 33. Resumen de la aplicación de la etapa de Recopilación de Información con sus productos de entrada y de salida

PRODUCTOS DE ENTRADA para la Recopilación de Información

- Análisis de requerimientos de desarrollo

- Documentación de la aplicación
  - Manuales técnicos.
  - Manuales de trazabilidad.
  - Elementos creados y/o modificados.
  - Documentación de requisitos relevados

PRODUCTO DE SALIDA para la Recopilación de Información

Paso 1. *Identificar a los responsables del proyecto:* Sumado a los dos productos de entrada (Análisis de requerimientos de desarrollo y documentación de la aplicación), es de suma importancia registrar a los responsables del proyecto de todas las áreas involucradas para dinamizar consultas o evacuación de dudas durante el nuevo proceso de testeo.

Dentro del producto generado en esta etapa, el Informe de Dominio, se encuentra el apartado para completar esta información.

1.1 Responsabilidades

Responsables de proyecto		
Nombre y Apellido	Cargo	Sector
Ariel Giannone	Desarrollador	Sistemas
Sergio Gonzalez	Analista	Sistemas
Raul Perez	Analista	Web

Tabla 3. Tabla de responsabilidades del producto informe de dominio

Paso 2. *Documentar el objetivo del proyecto:* En primera instancia, se describe el objetivo del proyecto para que todos los profesionales involucrados puedan consultarlo.

1.2 Objetivo del proyecto:
El objetivo de este proyecto es incorporar un canal web para permitir el acceso a los diferentes documentos que existen actualmente o que puedan añadirse en un futuro. Se debe tener como consideración especial que esta documentación es únicamente para usuarios mayoristas, donde minoristas no deben tener acceso allí.

Figura 34. Captura del objetivo del proyecto del producto informe del dominio

Paso 3. *Describir las funcionalidades afectadas:* El testeo no siempre es sobre un proyecto completo, sino que se puede someter a pruebas solo algunas funcionalidades, esto se describe para dejar en conocimiento el alcance de las pruebas.

1.3 Funcionalidades afectadas:
La funcionalidad que se ve afectada es la navegación actual de la página en cuestión.

Figura 35. Captura de funcionalidades afectadas del proyecto del producto informe del dominio

Paso 4. *Resumir el alcance funcional de la solución:* Este apartado ayudará en etapas futuras, donde se deban armar casos de pruebas, contando con un resumen del alcance funcional del proyecto esa tarea será facilitada de gran manera.

1.4 Descripción funcional:
El canal de acceso debe ser vía navegador web y sin necesidad de contar con ninguna instalación adicional, ni utilizar componentes que puedan fallar en navegadores conocidos.
Es requisito fundamental que la descarga solo pueda realizarse mediante un acceso a usuarios registrados, ya que no deben estar habilitadas para el público general. Este login será con un e-mail registrado y una contraseña alfanumérica de 8 caracteres.
Al ingresar 3 veces erróneamente la clave, se debe bloquear el ingreso, sin necesidad de bloquear el usuario y emitiendo el mensaje “Se han superado los intentos de ingreso”.
Si la paridad de usuario y clave concuerdan, se debe habilitar la descarga de los archivos que se encuentren cargados.

Figura 36. Captura de descripción funcional del proyecto del producto informe del dominio

Paso 5. *Evaluación de primeros riesgos asociados*: Dentro de este conocimiento de la aplicación, se encuentra un cuestionario sencillo el cual posee una valoración y según las respuestas nos dará un panorama cuantos riesgos posee la aplicación que estamos sometiendo a pruebas.

2. Evaluación de riesgos asociados:

El nivel de riesgo se valora entre 1 y 5. Se consideran los siguientes aspectos para determinar que una iniciativa es de riesgo alto (4 ó 5)

Preguntas	**Nivel de riesgo**	**Respuesta**
¿Se usa una nueva tecnología?	Riesgo 4	No
¿La operativa propuesta ha provocado algún incidente en la industria?	Riesgo 4	Si
¿Se utiliza un nuevo mecanismo de autenticación?	Riesgo 5	Si
¿Hay externalización de datos, personas o procesos?	Riesgo 5	No
¿Hay salida de datos? ¿De dónde sale? ¿Qué tipo de datos son?	Riesgo 5	Si, Archivos en formato PDF o XLS.
¿Se habilita un nuevo tipo de acceso a recursos informáticos?	Riesgo 4	No
¿La iniciativa se apoya sobre un componente/operativa declarado como riesgo activo?	Riesgo 5	No
¿Quién es el usuario consumidor?		Usuario Final
¿Desde dónde se accede a la información?		Por web
¿Qué información se maneja?		Listas de precios mayoristas
¿La iniciativa contempla comunicación directa? Correo electrónico, SMS, Push, etcétera.	Riesgo 3	No

Tabla 4. Tabla de riesgos asociados del producto informe de dominio

Cumpliendo estos ítems tendremos generado el documento llamado Informe de Dominio (ID) (Ver informe de dominio en sección 10.1 Anexos caso de validación 1)

5.1.2. Aplicación de la etapa de Análisis de Vulnerabilidades

La aplicación de la etapa de análisis de vulnerabilidades, permite listar vulnerabilidades, determinar probabilidad de vulnerabilidad y realizar un ranking de vulnerabilidades propicias. Para llevar a cabo este proceso se cuenta con el Análisis de requerimientos de desarrollo, la Documentación de la aplicación y el Informe de Dominio generado en la etapa predecesora

como productos de entrada, obteniendo el Informe de Vulnerabilidades (IV) como producto de salida.

La figura 37 sintetiza la aplicación de la 2° etapa aplicada para el caso de estudio:

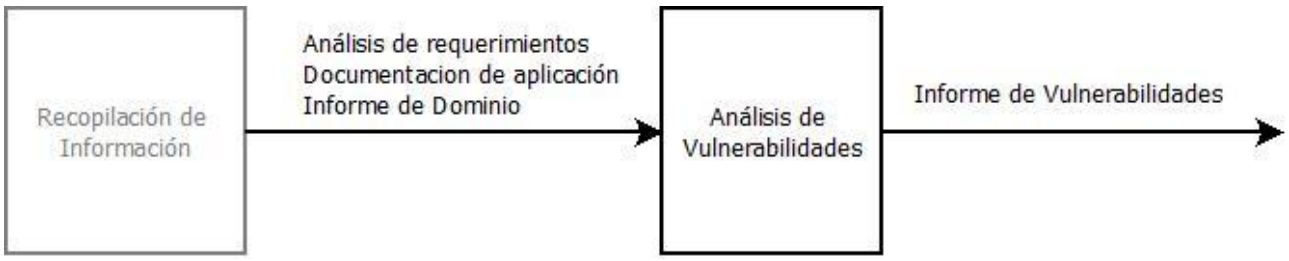

Figura 37 . Resumen de la aplicación de la etapa de Análisis de Vulnerabilidades con sus productos de entrada y de salida

PRODUCTOS DE ENTRADA para el Análisis de Vulnerabilidades

- Análisis de requerimientos de desarrollo
- Documentación de la aplicación
- Informe de Dominio

PRODUCTO DE SALIDA para el Análisis de Vulnerabilidades

Paso 1. *Revisar vulnerabilidades:* Junto con lo analizado en la etapa anterior, se sabrá si las vulnerabilidades a revisar son de tipo web o móvil. Se listarán las amenazas mas criticas dadas por la fundación OWASP, la cual al momento de escribir esta tesis, confecciona un documento de alto nivel que se centra sobre las vulnerabilidades más críticas de las aplicaciones web y móviles. Existe en el producto un apartado para completar ante vulnerabilidades específicas.

Paso 2. *Indicar vulnerabilidades propicias:* Teniendo listadas las amenazas mas criticas y recurrentes, procederemos a marcar cuales de estas pueden atacar directamente a nuestro sistema. Obteniendo de esta forma un ranking de vulnerabilidades propicias.

1. Listado de vulnerabilidades Web

Actualización	OWASP_Top_10-2017

Titulo	Sistema Propicio
Inyección	Si
Pérdida de Autenticación y Gestión de Sesiones	No
Cross-Site Scripting (XSS)	No
Rotura de control de acceso	Si
Security Misconfiguration	No
Sensitive Data Exposure	No
InsufficientAttackProtection	No
Cross-Site Request Forgery (CSRF)	Si
Using Components with Known Vulnerabilities	No
UnderprotectedAPIs	No

Tabla 5. Tabla de vulnerabilidades web del producto informe de vulnerabilidades

2. Listado de vulnerabilidades Mobile

Actualización	Mobile_Top_10-2016

Titulo	Sistema Propicio
Uso inadecuado de la plataforma	No
Almacenamiento de datos inseguros	No
Comunicación insegura	No
Autenticación no segura	No
Criptografía insuficiente	No
Autorización insegura	No
Calidad del código del cliente	No
Manipulación del código	No
Ingeniería inversa	No
Funcionalidad Extraña	No

Tabla 6. Tabla de vulnerabilidades mobile del producto informe de vulnerabilidades

Cumpliendo con estos pasos, obtendremos el producto de salida de la etapa llamado Informe de Vulnerabilidades (IV). (Ver informe de vulnerabilidades en sección 10.1 Anexos caso de validación 1)

5.1.3. Aplicación de la etapa de Modelado de Amenazas

La aplicación de la etapa de análisis de vulnerabilidades, permite identificar información sensible, describir la arquitectura, descomponer la aplicación, identificar y documentar vulnerabilidades criticas y asignar prioridades de ataque a las vulnerabilidades mas criticas. Para llevar a cabo este proceso se cuenta con los documentos de entrada de la fase junto con el

generado en la etapa predecesora, obteniendo el Informe de Vulnerabilidades (IA) y las prioridades de criticidad (PC) como productos de salida.
La figura 38 sintetiza la aplicación de la 3° etapa para el caso de estudio:

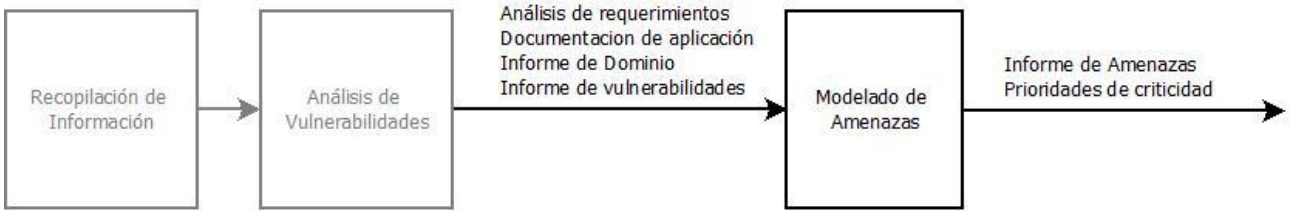

Figura 38. Resumen de la aplicación de la etapa de Modelado de Amenazas con sus productos de entrada y de salida

PRODUCTOS DE ENTRADA para el Modelado de Amenazas

- Análisis de requerimientos de desarrollo
- Documentación de la aplicación
- Informe de Dominio
- Informe de Vulnerabilidades

PRODUCTOS DE SALIDA para el Modelado de Amenazas

Paso 1. *Analizar nivel de criticidad de amenazas:* Junto con el documento generado de la etapa de Análisis de Vulnerabilidades, se deberá indagar sobre cada una de las amenazas propicias y asignarle un valor de criticidad. Esta valoración es de 1 a 5, siendo 1 Baja criticidad y 5 Alta Criticidad.

Paso 2. *Analizar nivel de aplicabilidad de amenazas:* De la misma forma que se evalúa que tan crítica puede ser una vulnerabilidad para el tipo de sistema a analizar, se identifica el nivel de aplicabilidad, diciéndonos que tanto aplica esa amenaza al sistema puesto a prueba. Esta valoración es de 1 a 5, siendo 1 Baja aplicabilidad y 5 Alta Aplicabilidad.

Estos dos últimos pasos se reflejan en el documento de salida de la siguiente forma:

1. Listado de Amenazas Web

Actualización	OWASP Top 10 - 2017

El nivel de criticidad y aplicabilidad se valora entre 1 y 5.

Titulo	Criticidad	Aplicabilidad
Inyección	4	4
Pérdida de Autenticación y Gestión de Sesiones	-	-
Cross-Site Scripting (XSS)	-	-
Rotura de control de acceso	5	3
Security Misconfiguration	-	-
Sensitive Data Exposure	-	-
InsufficientAttackProtection	-	-
Cross-Site Request Forgery (CSRF)	2	1
Using Components with Known Vulnerabilities	-	-
UnderprotectedAPIs	-	-

Tabla 7. Tabla de listado de amenazas web del producto informe de amenazas

2. Listado de Amenazas Mobile

Actualización	Mobile Top 10 2016-Top 10

Titulo	Criticidad	Aplicabilidad
Uso inadecuado de la plataforma	-	-
Almacenamiento de datos inseguros	-	-
Comunicación insegura	-	-
Autenticación no segura	-	-
Criptografía insuficiente	-	-
Autorización insegura	-	-
Calidad del código del cliente	-	-
Manipulación del código	-	-
Ingeniería inversa	-	-
Funcionalidad Extraña	-	-

Tabla 8. Tabla de listado de amenazas mobile del producto informe de amenazas

Contando con estos pasos, obtendremos el producto de salida de la etapa llamado Informe de Amenazas (IA). (Ver informe de amenazas en sección 10.1 Anexos caso de validación 1)

Paso 3. *Priorizar amenazas críticas:* Todas las amenazas valoradas en los pasos anteriores se deben colocar en prioridad para que los profesionales sepan cómo y qué puntos encarar con mayor urgencia. Esto se realiza colocando en formato de ranking las amenazas evaluadas en los puntos anteriores.

1. Prioridades de amenazas Web

Actualización	OWASP Top 10 - 2017

Prioridad de criticidad	Titulo	Criticidad	Aplicabilidad
1	Inyección	4	4
	Pérdida de Autenticación y Gestión de Sesiones	-	-
	Cross-Site Scripting (XSS)	-	-
2	Rotura de control de acceso	5	3
	Security Misconfiguration	-	-
	Sensitive Data Exposure	-	-
	InsufficientAttackProtection	-	-
3	Cross-Site Request Forgery (CSRF)kl	2	1
	Using Components with Known Vulnerabilities	-	-
	UnderprotectedAPIs	-	-

Tabla 9. Tabla de prioridades de amenazas web del producto prioridades de criticidad

2. Prioridades de amenazas Mobile

Actualización	Mobile Top 10 2016-Top 10

Prioridad de criticidad	Titulo	Criticidad	Aplicabilidad
-	Uso inadecuado de la plataforma		
-	Almacenamiento de datos inseguros		
-	Comunicación insegura		
-	Autenticación no segura		
-	Criptografía insuficiente		
-	Autorización insegura		
-	Calidad del código del cliente		
-	Manipulación del código		
-	Ingeniería inversa		
-	Funcionalidad Extraña		

Tabla 10. Tabla de prioridades de amenazas mobile del producto prioridades de criticidad

Cumplimentando estos pasos, obtendremos el segundo producto de salida de esta etapa llamado Prioridades de criticidad (PC). (Ver prioridades de criticidad en sección 10.1 Anexos caso de validación 1)

5.1.4. Aplicación de la etapa de Planeamiento de Pruebas

La aplicación de la etapa de planeamiento de pruebas, permite documentar pruebas, armar casos de pruebas, establecer criterios de aprobación y rechazo, determinar grados de criticidad, calendarizar hitos, estimar esfuerzos y finalmente elaborar el plan. Esta etapa cuenta con todos

los documentos generados durante la fase, generando los Casos de pruebas (CP), los calendarios de pruebas y esfuerzo (CPE) y el Informe de Criterios (IC).

La figura 39 sintetiza la aplicación de la 4° etapa aplicada en el caso de estudio:

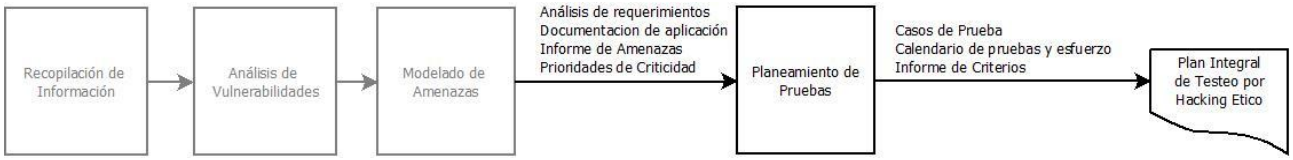

Figura 39. Resumen de la aplicación de la etapa de Planeamiento de Pruebas con sus productos de entrada y de salida

PRODUCTOS DE ENTRADA para el Planeamiento de Pruebas

- Análisis de requerimientos de desarrollo
- Documentación de la aplicación
- Informe de Amenazas
- Prioridades de criticidad

PRODUCTOS DE SALIDA para el Planeamiento de Pruebas

Paso 1. *Armado de casos de prueba:* Teniendo priorizadas las amenazas críticas y aplicables, se puede proceder a crear los diferentes casos de prueba que satisfagan las pruebas en esos puntos sensibles dentro de la aplicación. En cada caso se deberá indicar un titulo, las condiciones, los datos de entrada y los resultados esperados. Y junto a cada uno de los casos, se indicará en que etapa de la fase siguiente se realizaran las pruebas correspondientes.

1. Casos de prueba

Código caso	*Titulo*	*Condiciones*	*Datos de entrada*	*Resultados esperados*	*Etapa de fase de ejecución*
1	Revisión general de funcionamiento de login	Encontrarse en la página de acceso a listas de precios	Usuario y contraseña valida	Consulta en la BBDD correcta con contraseña encriptada	Escaneo
2	Inyección SQL	Estar dentro del login y tener habilitados los campos de usuario y contraseña	Ingresar una sentencia valida de SQL, tal como “select * from usuarios”	Ningún resultado de consulta en BBDD o error que pueda dar indicio de datos	Reconocimiento
3	Lectura de tráfico de contraseñas	Haber realizado un login		Contraseña encriptada desde la entrada y no solo en el motor	Escaneo
4	Prueba de bloqueo por accesos fallidos			Bloqueo de login ante 3 intentos fallidos	Reconocimiento
5	Intento de penetración en hosting de pagina		Relevamientos previos	Puertos y servicios monitoreados	Todas

Tabla 11. Tabla de casos de prueba del producto casos de prueba

Paso 2. *Priorizar casos de prueba:* Una vez identificados todos los casos de prueba, se procede a catalogarlos por su criticidad. Utilizando la misma valoración que en etapas anteriores, donde el valor 1 significa Baja Criticidad y 5 Alta Criticidad.

2. Criticidad

Código de caso	Nivel de criticidad	Observaciones
1	2	
2	4	
3	5	
4	3	
5	5	

Tabla 12. Tabla de criticidad del producto casos de prueba

Paso 3. *Estimar esfuerzo:* Por cada caso indicaremos que esfuerzo medido en horas/hombre insumirán.

3. Estimación de esfuerzo

Código de caso	Esfuerzo aproximado	Observaciones
1	4 hs/h	
2	5 hs/h	
3	6hs/h	
4	1 hs/h	
5	16hs/h	

Tabla 13. Tabla de estimación de esfuerzos del producto casos de prueba

Paso 4. *Calendarizar pruebas:* Teniendo registrado el esfuerzo de cada caso a probar, se puede realizar la calendarización de la prueba completa. Esto cobra muchísimo sentido en proyectos de gran envergadura y donde hay casos que prueba con dependencias cruzadas o muchos profesionales involucrados por ejemplo.

1. Calendarización

Casos	Inicio	Duración (días)	Fin
1	10/7/2018	0,5	10/7/2018
2	11/7/2018	0,625	11/7/2018
3	12/7/2018	0,75	12/7/2018
4	16/7/2018	0,125	16/7/2018
5	17/7/2018	2	19/7/2018

Tabla 14. Tabla de calendarización del producto calendario de pruebas y esfuerzo

2. Gantt

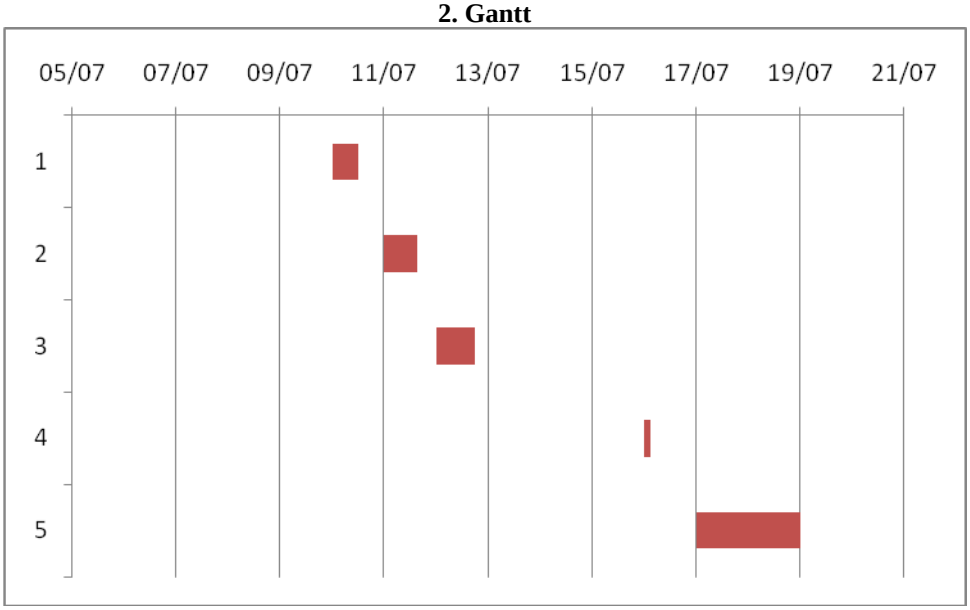

Figura 40. Captura de grafico de Gantt del producto casos de prueba

Cumplido estos pasos se obtienen como productos de salida los Casos de prueba (CP) y el Calendario de pruebas y esfuerzo (CPE). (Ver casos de prueba y el calendario de pruebas y esfuerzo en sección 10.1 Anexos caso de validación 1)

Paso 5. *Listar criterios de aprobación y rechazo:* Para poder aprobar o rechazar la prueba de un caso es necesario contar con criterios. Estos quedan registrados y pueden reutilizarse en aplicaciones similares.

2. Listado de Criterios de aprobación

Se aprobará al menos que…

Criterio	Aprobación
Contraseñas seguras	La clave debe encriptarse antes de la llamada Post
Firewall	El firewall no expondrá servicios sensibles y/o direcciones internas
Pruebas de acceso	Se instalen diferentes logs servers para que no sea sencilla el borrado de pruebas
Casos de prueba	Los casos de prueba sean satisfactorios en su totalidad.

Tabla 15. Tabla de listado de criterios de aprobación del producto informe de criterios

3. Listado de Criterios de Rechazo

Se rechazará cuando…

Criterio	Rechazo
Contraseñas seguras	Si la clave puede leerse con un lector de trafico
Inyección SQL	Si ante una sentencia SQL se emite algún indicio del motor de BBDD
Bloqueo al tercer intento	Al tercer intento se debe bloquear la prueba de acceso
Puertos abiertos	Si hay puertos abiertos que no pertenezcan a la aplicación o funcionamiento propio del servidor

Tabla 16. Tabla de listado de criterios de rechazo del producto informe de criterios

Al culminar este ítem obtenemos el Informe de Criterios (IC). (Ver informe de criterios en sección 10.1 Anexos caso de validación 1)

5.1.5. Obtención documento "Plan Integral de Testeo por Hacking Ético (PITHE)"

La aplicación de la todas las etapas que conforman la 1° fase de este proceso y compilando todos los productos obtenidos, se obtiene el documento final de la fase.

5.2. APLICACIÓN DE LAS ACTIVIDADES DE LA FASE DE EJECUCION DE TESTEO

En esta sección se procederá a aplicar al caso de validación las etapas correspondientes a la fase de Ejecución: Reconocimiento (sección 5.2.1), Escaneo (sección 5.2.2), Ganancia de acceso (sección 5.2.3), Mantenimiento de Acceso (sección 5.2.4) y Eliminación de pruebas (sección 5.2.5). Obteniendo como producto final de la fase el Informe Final de Testeo por Hacking Ético (IFTHE) (sección 5.2.6) y aplicación de métricas (sección 5.2.7).

A continuación se procede a aplicar las etapas de esta fase de ejecución, siguiendo los pasos especificados en la tabla 4.1 y que se describen con detalle en la figura 4.5 de la sección 4.1.2.2 del capítulo 4. La etapa es iniciada con todos los productos generados en la fase anterior (Planificación), y finalizando con la obtención del informe final del testeo.

En esta sección aparecerán descriptas diferentes herramientas que al momento de escribir esta tesis, se encuentran en el auge de su uso. El proceso planteado debe trascender a las herramientas que se utilicen en cada etapa.

5.2.1. Aplicación de la etapa de Reconocimiento

La aplicación de la etapa de reconocimiento, permite recopilar información, determinar el tamaño de la red, identificar maquinas activas, descubrir puertos abiertos y puntos de acceso, identificar el sistema operativo y realizar los casos de pruebas identificados para esta etapa. Para llevar adelante este proceso se cuenta con el Plan integral de testeo por hacking ético obtenido en la etapa predecesora como productos de entrada obteniendo el Informe de Arquitectura de Red (IAR) como producto de salida.
La figura 41 sintetiza la aplicación de la 2° etapa aplicada para el caso de estudio:

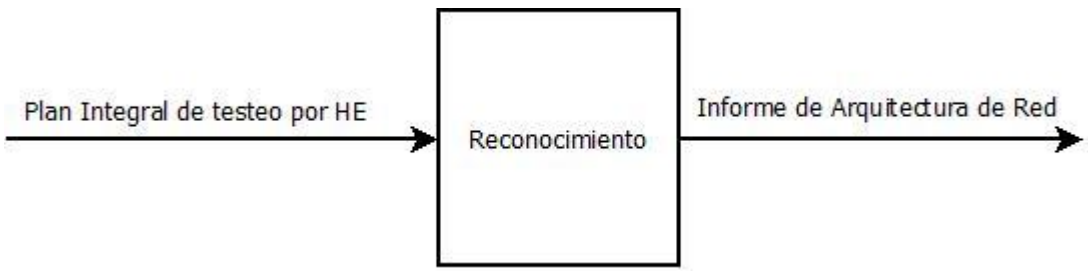

Figura 41. Resumen de la aplicación de la etapa de Reconocimiento con sus productos de entrada y de salida

PRODUCTOS DE ENTRADA para la etapa de Reconocimiento

- Plan Integral de Testeo por Hacking Ético

PRODUCTOS DE SALIDA para la etapa de Reconocimiento

Paso 1. *Descubrir Sistema Operativo:* Gracias a la herramienta nmap se puede determinar el Sistema Operativo que se utiliza en destino, esto ayuda a los atacantes a saber con qué comandos y que vulnerabilidades explotar. Dentro del informe se indicará esto.

```
nmap -O antirrobosalto.com.ar
```

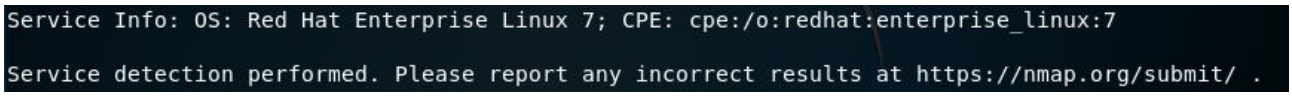

Figura 42. Captura de ejecución para descubrir el sistema operativo

Paso 2. *Identificar maquinas involucradas en la aplicación:* Utilizando la herramienta nmap se pueden obtener las maquinas activas junto con nombres de host y direcciones IP.

```
nmap -sP antirrobosalto.com.ar
```

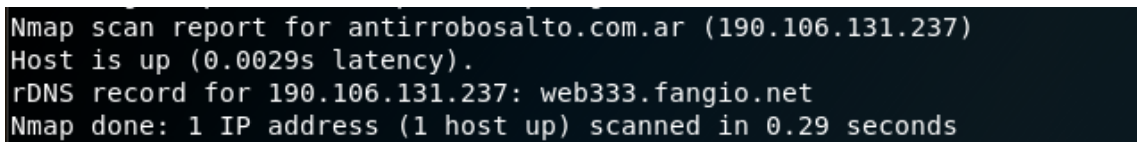

Figura 43. Captura de ejecución para identificar maquinas involucradas

Paso 3. *Identificar los puertos y puntos de acceso:* Junto con la misma herramienta se obtiene el listado de los puertos y posibles puntos de acceso abiertos en el destino.

```
nmap -v antirrobosalto.com.ar
```

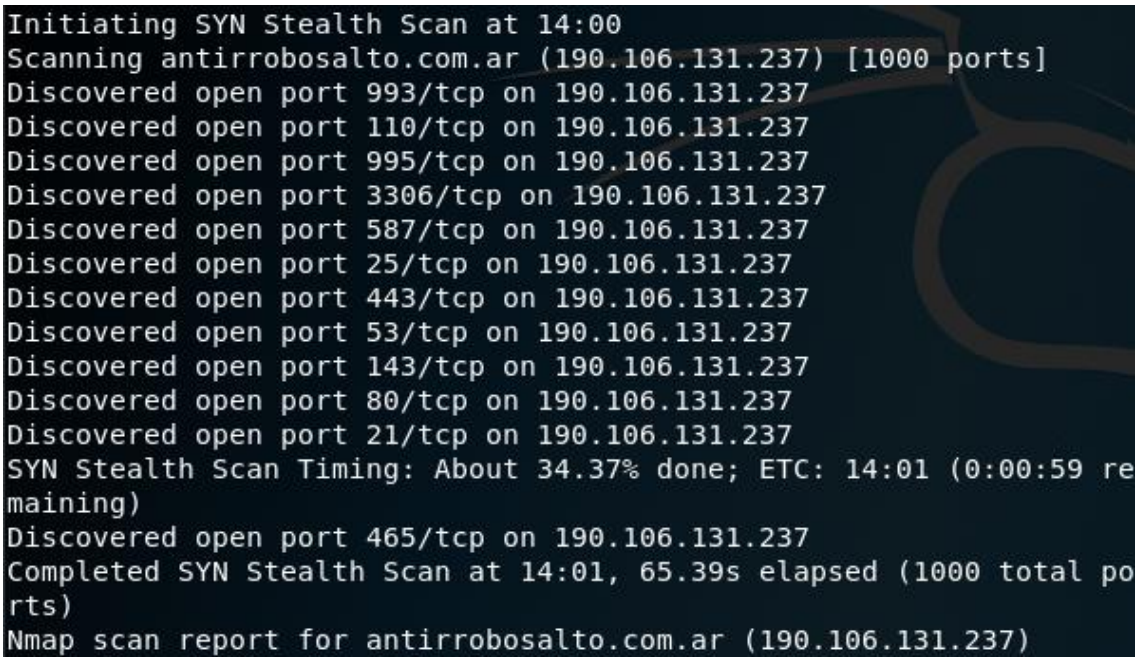

Figura 44. Captura de ejecución para identificar puertos y puntos de acceso

Paso 4. *Identificar DNS:* Ejecutando los mismos parámetros sobre la herramienta se obtiene el listado de dns configurados.

```
nmap -v antirrobosalto.com.ar
```

Figura 45. Captura de ejecución para identificar DNS

Paso 5. *Identificar Dominios activos:* Ejecutando los mismos parámetros sobre la herramienta se obtiene el listado de dns configurados.

El DNS obtenido en el paso anterior es el único dominio activo.

Paso 6. *Realizar los casos de prueba identificados:* En la etapa de armado de casos de prueba se identifica en que etapa de esta fase deben realizarse. Para este caso de prueba se encuentran los casos con código 2 y 4.

2	Inyección SQL	Estar dentro del login y tener habilitados los campos de usuario y contraseña	Ingresar una sentencia valida de SQL, tal como “select * from usuarios”	Ningún resultado de consulta en BBDD o error que pueda dar indicio de datos	Reconocimiento
4	Prueba de bloqueo por accesos fallidos		Usuario y contraseña invalida	Bloqueo de login ante 3 intentos fallidos	Reconocimiento

Tabla 17. Tabla de casos de prueba a ejecutar en la etapa de reconocimiento

Caso 2: *Inyección SQL*

Para poder comprobar este caso de prueba se utilizará la herramienta Wireshark, la misma es un analizador de protocolos utilizado para realizar análisis. La funcionalidad que provee es similar a la de tcpdump, permite ver todo el tráfico que pasa a través de una red.

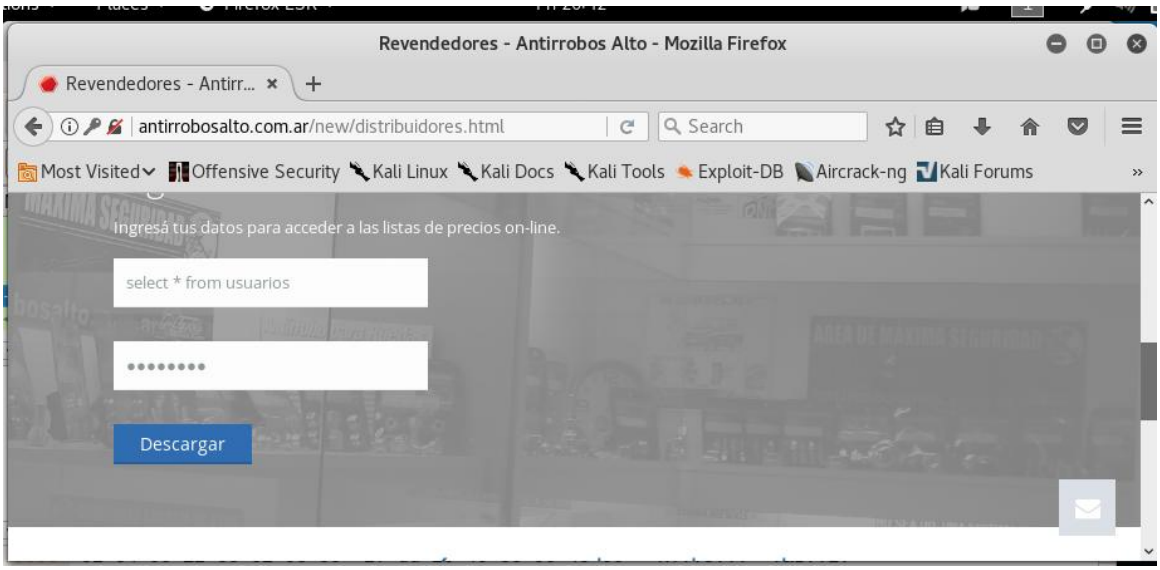

Figura 46. Captura de pantalla de ingreso de inyección SQL

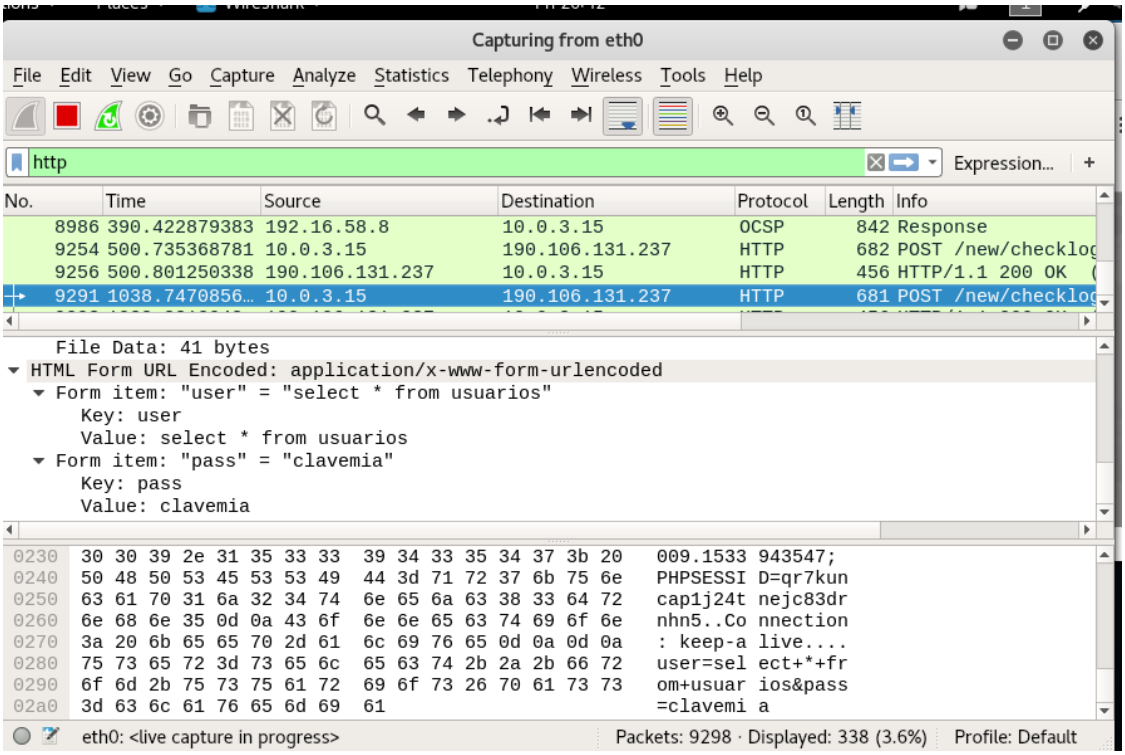

Figura 47. Captura de herramienta de lectura de trafico de ingreso de inyección SQL

Caso 4: *Accesos fallidos*

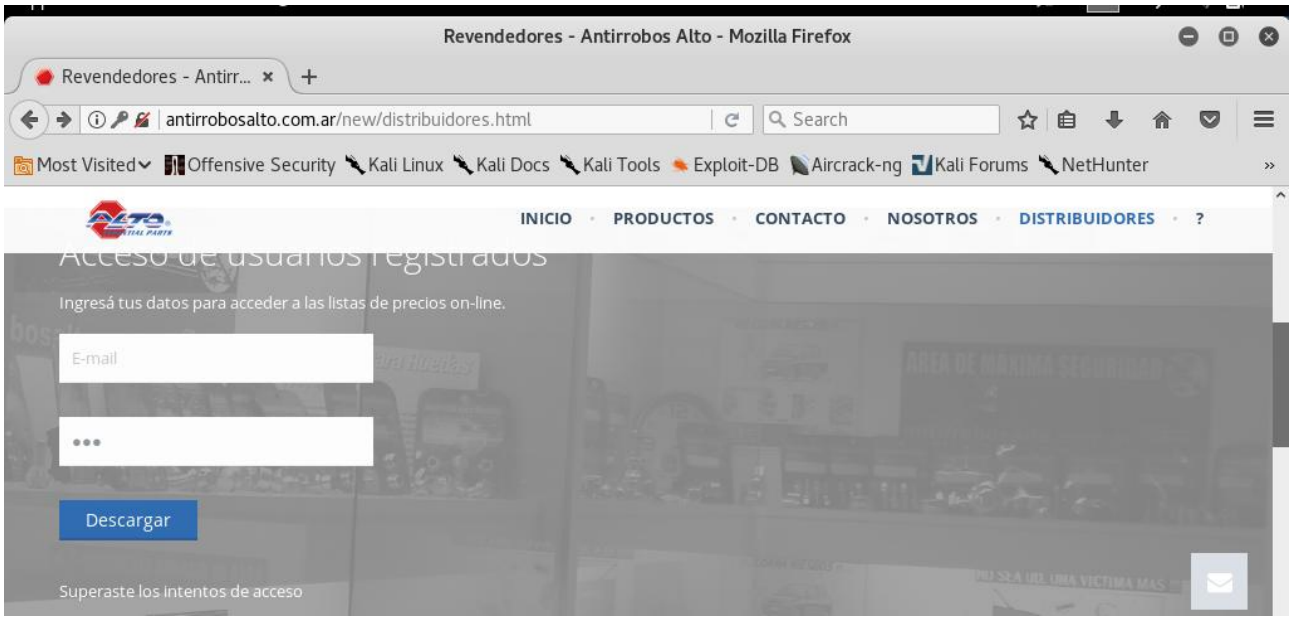

Figura 48. Captura de pantalla de prueba de intentos fallidos

Contando con todos estos pasos, obtendremos el producto de salida de la etapa llamado Informe de Arquitectura de Red (IAR). (Ver informe de arquitectura en sección 10.1 Anexos caso de validación 1)

5.2.2. Aplicación de la etapa de Escaneo

La etapa de escaneo, permite detectar sistemas vivos en la red, descubrir puertos abiertos, encontrar servicios activos y en ejecución y realizar los casos de pruebas identificados para esta etapa. Para llevar adelante este proceso se cuenta con el Plan integral de testeo por hacking ético obtenido en la etapa predecesora como productos de entrada obteniendo el Informe de Escaneo (IE) como producto de salida.
La figura 49 sintetiza la aplicación de la 2° etapa aplicada para el caso de estudio:

Figura 49. Resumen de la aplicación de la etapa de Escaneo con sus productos de entrada y de salida

PRODUCTOS DE ENTRADA para la etapa de Escaneo

- Plan Integral de Testeo por Hacking Ético
- Informe de Arquitectura de red

PRODUCTOS DE SALIDA para la etapa de Escaneo

Paso 1. *Descubrir Puertos abiertos y servicios:* Continuando con la herramienta nmap se pueden detallar los puertos abiertos y que servicios son los que los utilizan.

```
nmap –-osscan -guess antirrobosalto.com.ar
```

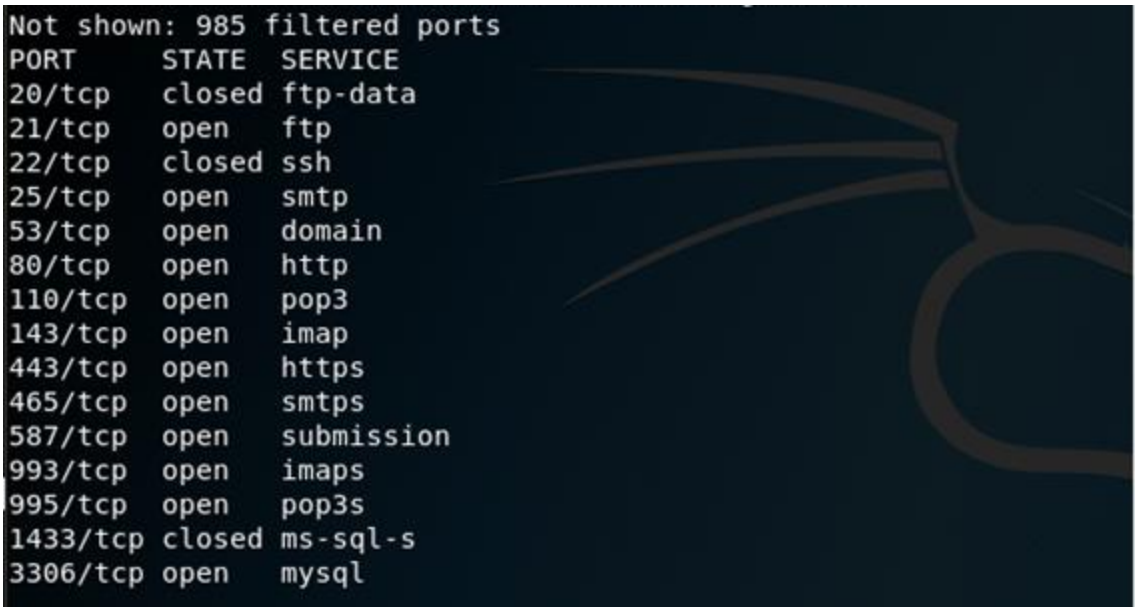

Figura 50. Captura de ejecución para descubrir puertos y servicios

2. Listado de Puertos activos y puntos de acceso

Puerto	Tipo	Estado	Servicio
20	Tcp	Cerrado	ftp-data
21	Tcp	Abierto	ftp
22	Tcp	Cerrado	Ssh
25	Tcp	Abierto	Smtp
53	Tcp	Abierto	Domain
80	Tcp	Abierto	http
110	Tcp	Abierto	Pop3
143	Tcp	Abierto	Imap
443	Tcp	Abierto	https
465	Tcp	Abierto	Smtps
587	Tcp	Abierto	Submission
993	Tcp	Abierto	Imaps
995	Tcp	Abierto	Pop3s
1433	Tcp	Cerrado	Ms-sql-s
3306	Tcp	Abierto	mysql

Tabla 18. Tabla de puertos activos y puntos de acceso del producto casos de prueba

Paso 2. *Descubrir dominios y DNS:* Continuando con la herramienta nmap se pueden descubrir los DNS y dominios mapeados.

```
nmap --osscan -guess antirrobosalto.com.ar
```

```
rDNS record for 190.106.131.237: web333.fangio.net
```

Figura 51. Captura de ejecución para descubrir dominios y dns

2. Dns y dominios

Dns	Proveedor	Ip
antirrobosalto.com.ar	**web333.fangio.net**	**190.106.131.237**

Tabla 19. Tabla de dns y dominios del producto casos de prueba

Paso 3. *Realizar los casos de prueba identificados:* En la etapa de armado de casos de prueba se identifican en qué etapa de esta fase deben realizarse. Para este caso de prueba se encuentran los casos con código 1 y 3.

1	Revisión general de funcionamiento de login	Encontrarse en la página de acceso a listas de precios	Usuario y contraseña valida	Consulta en la BBDD correcta con contraseña encriptada	Escaneo
3	Lectura de tráfico de contraseñas	Haber realizado un login	Usuario y contraseña valida	Contraseña encriptada desde la entrada y no solo en el motor	Escaneo

Tabla 20. Tabla de casos de prueba a ejecutar en la etapa de escaneo

Caso 1: *Prueba positiva*

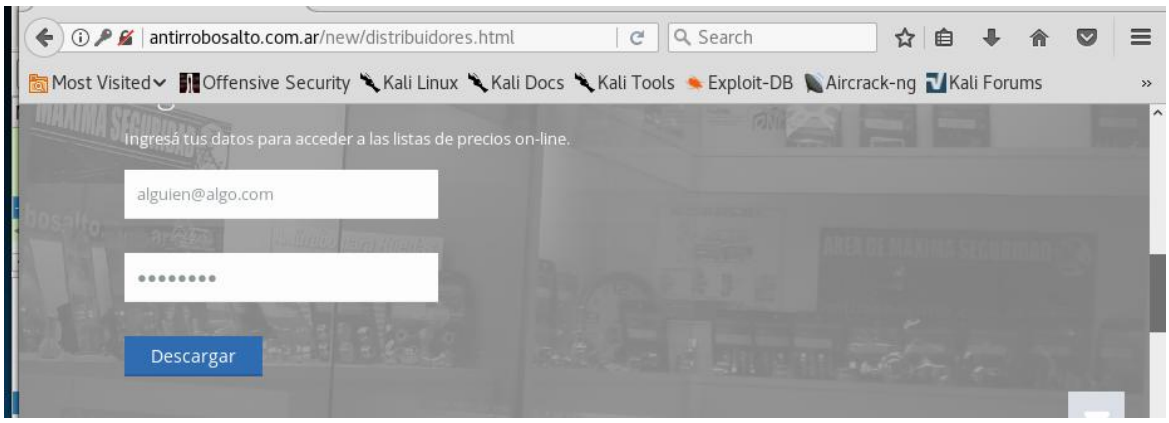

Figura 52. Captura de pantalla de ingreso por prueba positiva

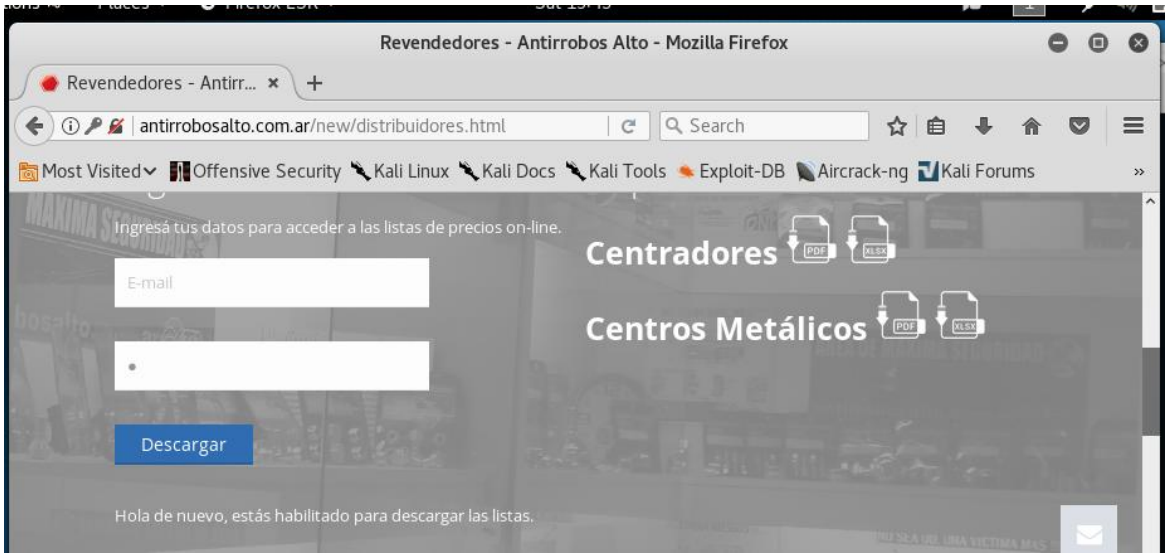

Figura 53. Captura de pantalla de acceso por prueba positiva

Caso 1: Prueba negativa

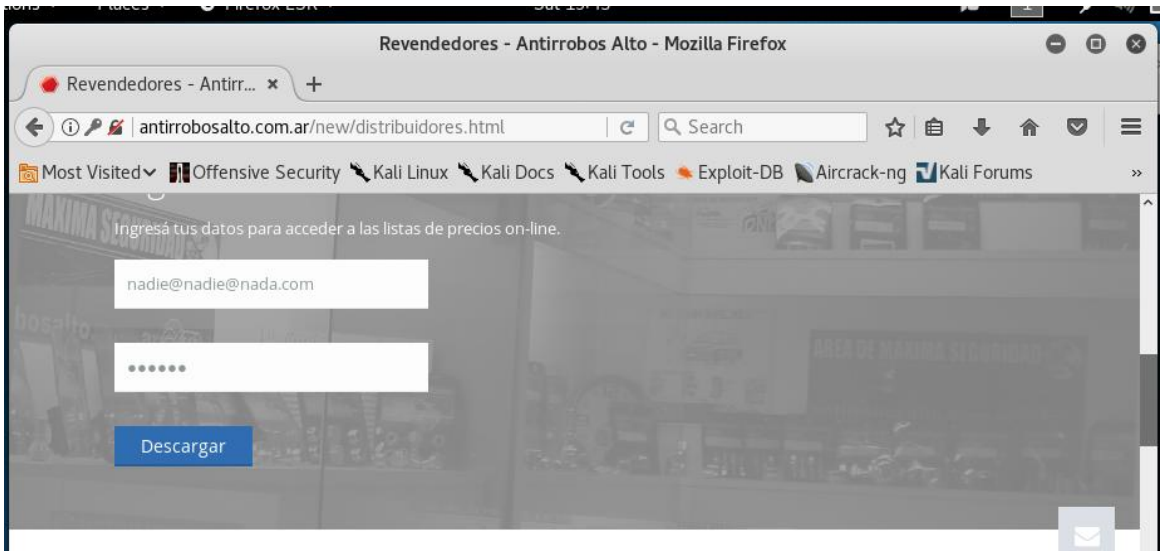

Figura 54. Captura de pantalla de ingreso por prueba negativa

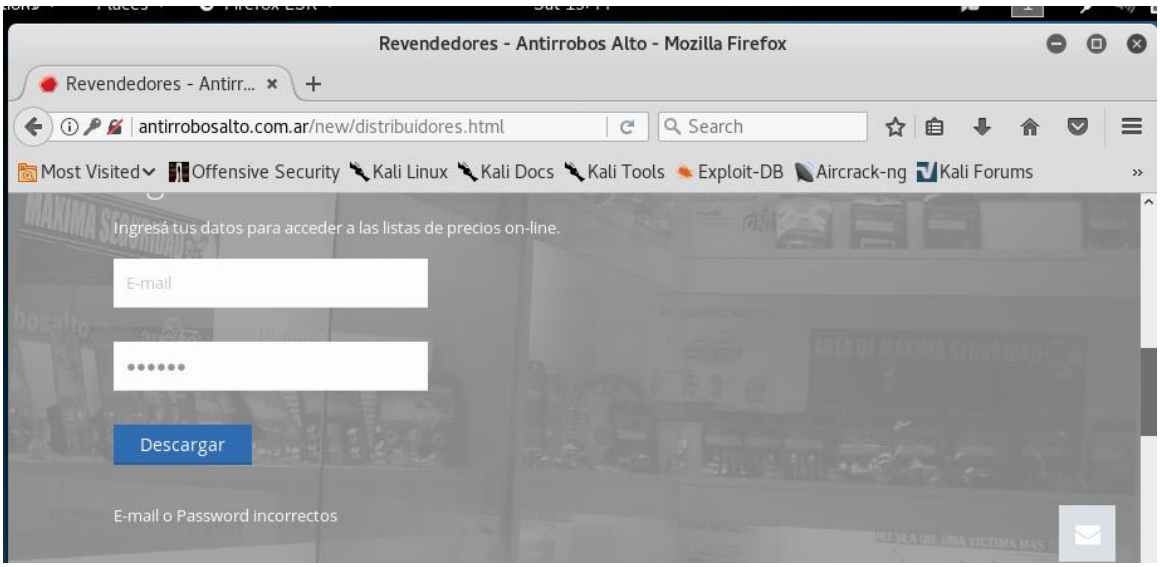

Figura 55. Captura de pantalla de falla por prueba negativa

Caso 3: *Lectura de tráfico de contraseñas*

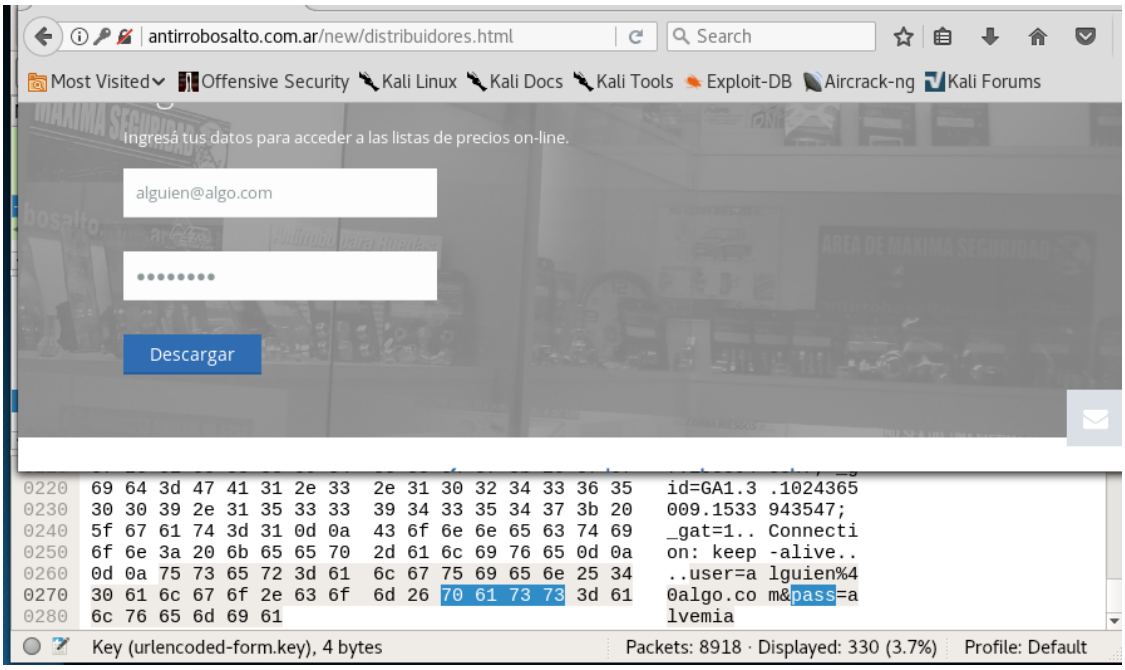

Figura 56. Captura de pantalla de lectura de contraseñas

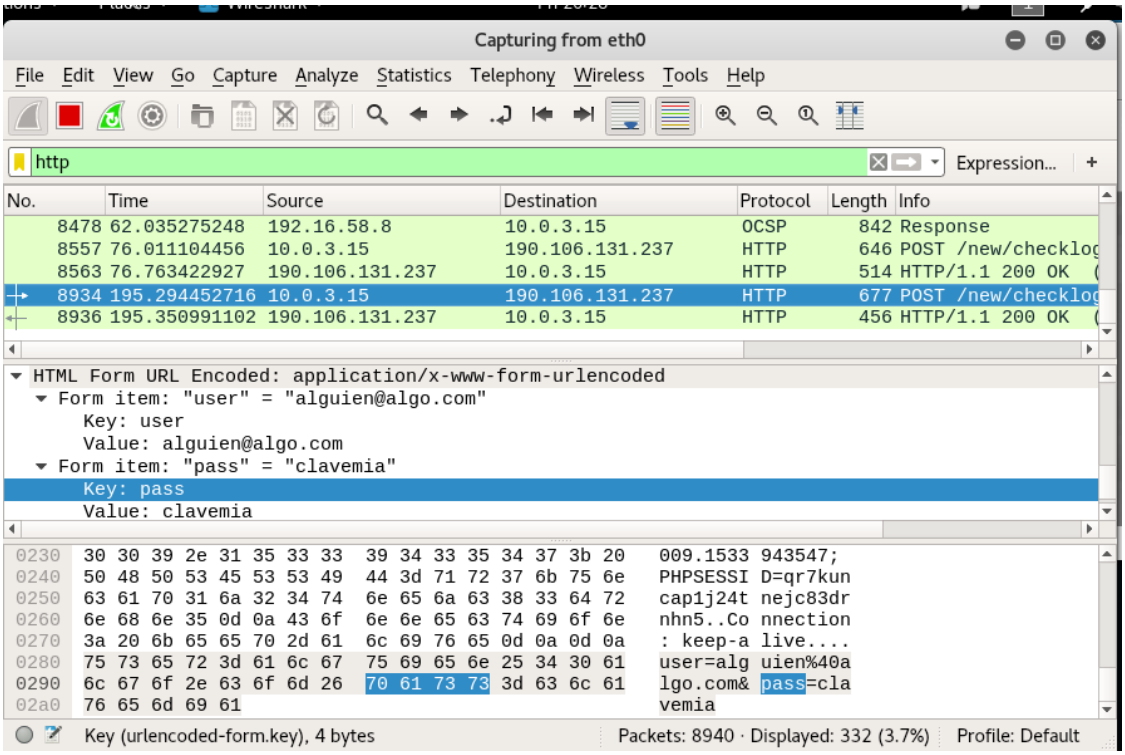

Figura 57. Captura de pantalla de herramienta para lectura de trafico

Es en este caso donde se encuentra una vulnerabilidad expuesta de tipo grave, ya que con un lector de tráfico podrían obtenerse un par de datos reales y validos de usuario y contraseña.

Contando con todos estos pasos, obtendremos el producto de salida de la etapa llamado Informe de Escaneo del Sistema (IES). (Ver informe de escaneo de sistema en sección 10.1 Anexos caso de validación 1)

5.2.3. Aplicación de la etapa de Ganancia de Acceso

La etapa de ganancia de acceso, existe para explotar las direcciones, puertos y servicios, buscando obtener un punto de acceso a la maquina víctima. Para llevar adelante este proceso se cuenta con las investigaciones de datos obtenidos en los productos anteriores y obteniendo el Informe de Accesos ganados (IAG) como producto de salida.
Estas etapas y lo mismo con su sucesora, son etapas propias del ciclo del hacking; dentro de este proceso, se forzaran ambas para demostrar que son factibles y que luego siguiendo con puntos de buenas prácticas, pueden mitigarse algunos riesgos.

La figura 58 demuestra la aplicación de la 3° etapa aplicada para el caso de estudio:

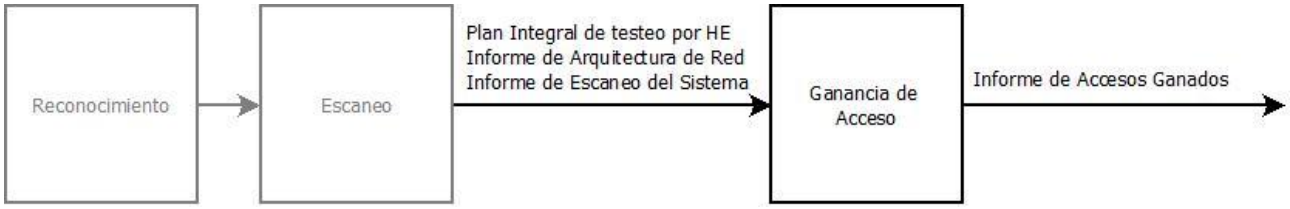

Figura 58. Resumen de la aplicación de la etapa de Ganancia de acceso con sus productos de entrada y de salida

PRODUCTOS DE ENTRADA para la etapa de Ganancia

- Plan Integral de Testeo por Hacking Ético
- Informe de Arquitectura de red
- Informe de Escaneo de Sistema

PRODUCTOS DE SALIDA para la etapa de Ganancia

Paso 1. *Explotar vulnerabilidad*: Junto con el listado de los puertos descubiertos y vulnerabilidades detectadas, se busca un exploit que permita obtener el control.

Herramientas recomendadas:

- MetaSploit
- Core Impact
- Immunity Canvas
- Milw0rm

Paso 2. *Elevar privilegios*: Para poder ampliar el control y llevarlo a ser total se debe intentar llevar los privilegios a administrador o root, esto se logra con otros tipos de aplicaciones exploits.

Paso 3. *Ejecución remota*: Obteniendo control y privilegios, se ejecutan comandos o aplicaciones de forma remota para mantener el canal.

Herramienta recomendada:

- PsExec

Paso 3. *Generar Informe*: Ante el caso necesario de poner en práctica esta etapa, se elaborará el Informe de Accesos Ganados (IAG).

5.2.4. Aplicación de la etapa de Mantenimiento de Acceso

La etapa de mantenimiento de acceso, existe para mantener el acceso ganado como exclusivo, cargar, descargar y manipular datos, aplicaciones y configuraciones del sistema, intentar usar al sistema para lanzar más ataques. Para llevar adelante este proceso se cuenta con los informes y obteniendo el Informe de Accesos mantenidos (IAM) como producto de salida.
La figura 59 demuestra la aplicación de la 4° etapa aplicada para el caso de estudio:

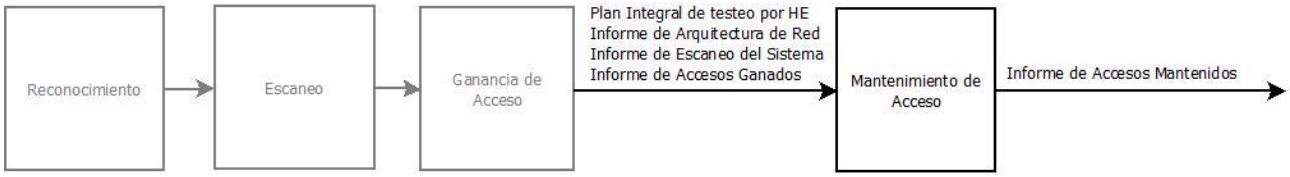

Figura 59. Resumen de la aplicación de la etapa de Ganancia de acceso con sus productos de entrada y de salida

PRODUCTOS DE ENTRADA para la etapa de Mantenimiento

- Plan Integral de Testeo por Hacking Ético
- Informe de Arquitectura de red
- Informe de Escaneo de Sistema
- Informe de Accesos Ganados

Paso 1. Instalar aplicación servidora de túnel: La mayoría de las herramientas listadas, trabajan instalándose en la maquina víctima, permitiendo conectarse las veces que sean necesarias.

Paso 2. *Tunelizar Puertos abiertos y servicios*: Junto con el listado de los puertos descubiertos, se intenta establecer un acceso end to end para conseguir una vía de intercambio de archivos y/o comandos.

La mayoría de las herramientas listadas, trabajan instalándose en la maquina víctima y pueden conectarse las veces que sean necesarias.

Herramientas recomendadas:

- PowerSploit

- Sbd
- Weevely
- http-tunnel
- dns2tcp

Paso 3. *Generar Informe*: Ante el caso necesario de poner en práctica esta etapa, se elaborará el Informe de Accesos Mantenidos (IAM)

5.2.5. Aplicación de la etapa de Eliminación de Pruebas

La etapa de borrado de pruebas tiene como objetivo el ocultamiento de actos maliciosos, sobrescribir registros y logs de sistema y aplicaciones y la elaboración del informe final. Para llevar adelante este proceso se cuenta con todos los datos obtenidos en los productos anteriores y se tendrá el Informe de Pruebas Eliminadas (IPE) como producto de salida.

La figura 60 demuestra la aplicación de la última etapa aplicada para el caso de estudio en cuestión:

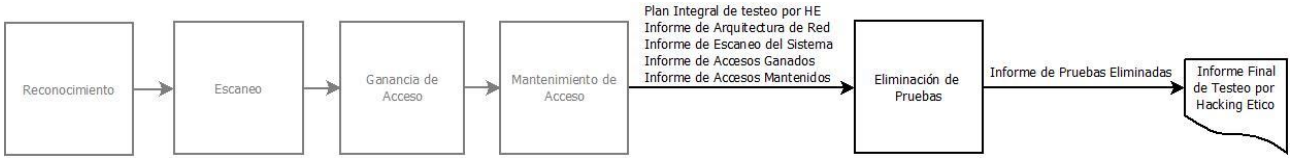

Figura 60. Resumen de la etapa de Eliminación de pruebas con sus productos de entrada y de salida

PRODUCTOS DE ENTRADA para la etapa de Eliminación de pruebas

- Plan Integral de Testeo por Hacking Ético
- Informe de Arquitectura de red
- Informe de Escaneo de Sistema
- Informe de Accesos Ganados
- Informe de Accesos Mantenidos

PRODUCTOS DE SALIDA para la etapa de Eliminación de pruebas

Paso 1. *Revisión de mantenimiento de logs:* Manteniendo el acceso ganado, se buscaran eliminar todos los registros de actividades realizadas. Aquí como en las etapas anteriores, a diferencia del proceso de hacking real, el hackeo ético, debe probar que no todos los logs pueden ser eliminados.

- Configuración de logs según SO
- Contar con varios logs servers

Herramientas recomendadas:

- Elogger
- Logsign
- Log360

Paso 2. *Ejecución de herramientas forenses:* En esta instancia del proceso, se busca contraprobar que es posible ejecutar herramientas forenses y recuperar huellas de lo sucedido en los servidores.

Herramientas recomendadas:

- Foremost
- Log2timeline

Cumplimentando estos pasos, obtendremos el producto de salida de esta etapa llamado Informe de Pruebas Eliminadas (IPE).

5.2.6. Obtención documento "Informe final de Testeo por Hacking Ético (IFTHE)"

La aplicación de la todas las etapas que conforman la 1° fase de este proceso y compilando todos los productos obtenidos, se obtiene el documento final de la fase, el cual contiene los casos de prueba positivos y negativos, siendo este documento el que puede enviar el desarrollo a ser modificado o el que indique que es un software apto para ser implementado en producción.

5.2.7. Aplicación de métricas en caso de validación 1

Tomando la información relevada en las secciones anteriores de la etapa de ejecución, se obtienen los siguientes índices de vulnerabilidad

- Índice de posibilidad de acceso

  CAG = 2
  PAD = 12

  Resultado = 2/12 = 16,7 %

- Índice de acceso real

  CAG = 2
  PAR = 15

  Resultado = 2/15 = 13 %

- Índice de detección de puertos

  PAD = 12
  PAR = 15

  Resultado = 12/8 = 80 %

5.3. APLICACIÓN DE LAS ACTIVIDADES DE LA FASE DE MANTENIMIENTO

En esta sección se aplicará al caso de validación corriente las etapas correspondientes a la fase de Mantenimiento: Control de Vulnerabilidades (sección 5.3.1), Determinación de Criterios (sección 5.3.2) y Verificación y Validación (sección 5.3.3). Obteniendo como producto final de la fase el Informe General de Riesgos, Validaciones y Verificaciones (IGRVV) (sección 5.3.4).

5.3.1. Aplicación de la etapa de Control de Vulnerabilidades

La aplicación de la esta etapa sobre el producto mismo en el ambiente de producción, permite listar, si existen, vulnerabilidades nuevas y determinar su grado de aplicabilidad al sistema. Para llevar a cabo este proceso se cuenta con el Informe obtenido en la fase de Ejecución y el Listado de nuevas vulnerabilidades, obteniendo el Informe de Vulnerabilidades Criticas (IVC) como producto de salida.

La figura 61 sintetiza la aplicación de la etapa de control de vulnerabilidades para el caso de estudio:

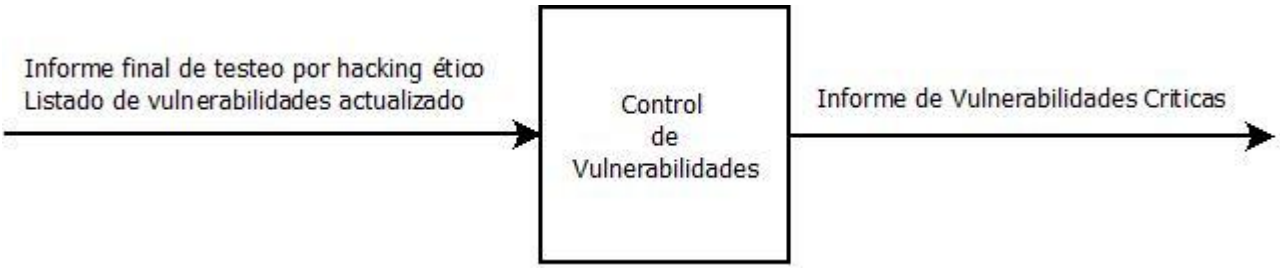

Figura 61. Resumen de la aplicación de la etapa de Control de Vulnerabilidades con sus productos de entrada y de salida

PRODUCTOS DE ENTRADA para el Control de Vulnerabilidades

- Informe Final de Testeo por Hacking Ético
- Listado de vulnerabilidades actualizado

PRODUCTO DE SALIDA para el Control de Vulnerabilidades

Paso 1. *Revisar vulnerabilidades:* Con esta etapa se busca revisar el cambio en las vulnerabilidades críticas o en ataques más frecuentes. Se listarán las amenazas mas criticas dadas por la fundación OWASP, la cual al momento de escribir esta tesis, confecciona un documento de alto nivel que se centra sobre las vulnerabilidades más críticas de las aplicaciones web y móviles.

Paso 2. *Indicar vulnerabilidades propicias:* Teniendo listadas las amenazas mas criticas y recurrentes, procederemos a marcar cuales de estas pueden atacar directamente a

nuestro sistema. Obteniendo de esta forma un ranking de vulnerabilidades propicias. Si en este documento no aparecen nuevas vulnerabilidades o si son las mismas a las que ya el sistema fue sometido a pruebas, la etapa de mantenimiento no debería continuar.

1. Listado de vulnerabilidades Web

Actualización	OWASP_Top_10-2017

Titulo	Sistema Propicio
Inyección	Si
Pérdida de Autenticación y Gestión de Sesiones	No
Cross-Site Scripting (XSS)	No
Rotura de control de acceso	Si
Security Misconfiguration	No
Sensitive Data Exposure	No
InsufficientAttackProtection	No
Cross-Site Request Forgery (CSRF)	Si
Using Components with Known Vulnerabilities	No
UnderprotectedAPIs	No

Tabla 21. Tabla de vulnerabilidades web del producto informe de vulnerabilidades

2. Listado de vulnerabilidades Mobile

Actualización	Mobile_Top_10-2016

Titulo	Sistema Propicio
Uso inadecuado de la plataforma	No
Almacenamiento de datos inseguros	No
Comunicación insegura	No
Autenticación no segura	No
Criptografía insuficiente	No
Autorización insegura	No
Calidad del código del cliente	No
Manipulación del código	No
Ingeniería inversa	No
Funcionalidad Extraña	No

Tabla 22. Tabla de vulnerabilidades mobile del producto informe de vulnerabilidades

Cumpliendo con estos pasos, obtendremos el producto de salida de la etapa llamado Informe de Vulnerabilidades Criticas (IVC). (Ver informe de vulnerabilidades críticas en sección 10.1 Anexos caso de validación 1)

5.3.2. Aplicación de la etapa de Determinación de Criterios

La aplicación de la etapa de determinación de criterios, ayuda a determinar grado de criticidad, establecer criterios de aprobación y rechazo y obtener un ranking de vulnerabilidades más riesgosas. Para llevar a cabo este proceso se cuenta con el Informe de vulnerabilidades críticas como producto de entrada, contando con el Listado de Aprobación y Rechazo (LAR) como producto de salida.

La figura 62 sintetiza la aplicación de la 2° etapa aplicada para el caso de estudio:

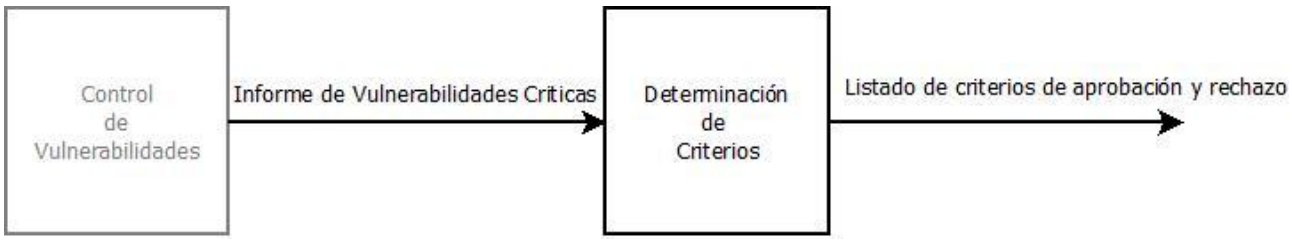

Figura 62 . Resumen de la aplicación de la etapa de Determinación de Criterios con sus productos de entrada y de salida

PRODUCTOS DE ENTRADA para la Determinación de Criterios

- Informe de Vulnerabilidades criticas

PRODUCTO DE SALIDA para la Determinación de Criterios

Paso 1. *Analizar nivel de criticidad de amenazas:* Junto con el documento generado de la etapa de Análisis de Vulnerabilidades, se deberá indagar sobre cada una de las amenazas propicias y asignarle un valor de criticidad. Esta valoración es de 1 a 5, siendo 1 Baja criticidad y 5 Alta Criticidad.

Paso 2. *Analizar nivel de aplicabilidad de amenazas:* De la misma forma que se evalúa que tan critica puede ser una vulnerabilidad para el tipo de sistema a analizar, se identifica el nivel de aplicabilidad, diciéndonos que tanto aplica esa amenaza al sistema puesto a prueba. Esta valoración es de 1 a 5, siendo 1 Baja aplicabilidad y 5 Alta Aplicabilidad.

Estos dos últimos pasos se reflejan en el documento de salida de la siguiente forma:

1. Listado de Amenazas Web

Actualización	OWASP Top 10 - 2017

El nivel de criticidad y aplicabilidad se valora entre 1 y 5.

Titulo	Criticidad	Aplicabilidad
Inyección	4	4
Pérdida de Autenticación y Gestión de Sesiones	-	-
Cross-Site Scripting (XSS)	-	-
Rotura de control de acceso	5	3
Security Misconfiguration	-	-
Sensitive Data Exposure	-	-
InsufficientAttackProtection	-	-
Cross-Site Request Forgery (CSRF)	2	1
Using Components with Known Vulnerabilities	-	-
UnderprotectedAPIs	-	-

Tabla 23. Tabla de listado de amenazas web del producto listado de criterios de aprobación y rechazo

2. Listado de Amenazas Mobile

Actualización	Mobile Top 10 2016-Top 10

Titulo	Criticidad	Aplicabilidad
Uso inadecuado de la plataforma	-	-
Almacenamiento de datos inseguros	-	-
Comunicación insegura	-	-
Autenticación no segura	-	-
Criptografía insuficiente	-	-
Autorización insegura	-	-
Calidad del código del cliente	-	-
Manipulación del código	-	-
Ingeniería inversa	-	-
Funcionalidad Extraña	-	-

Tabla 24. Tabla de listado de amenazas web del producto listado de criterios de aprobación y rechazo

Paso 3. *Priorizar amenazas críticas:* Todas las amenazas valoradas en los pasos anteriores se deben colocar en prioridad para que los profesionales sepan cómo y qué puntos encarar con mayor urgencia. Esto se realiza colocando en formato de ranking las amenazas evaluadas en los puntos anteriores.

3. Prioridades de amenazas Web

Actualización	OWASP Top 10 - 2017

Prioridad de criticidad	Titulo	Criticidad	Aplicabilidad
1	Inyección	4	4
	Pérdida de Autenticación y Gestión de Sesiones	-	-
	Cross-Site Scripting (XSS)	-	-
2	Rotura de control de acceso	5	3
	Security Misconfiguration	-	-
	Sensitive Data Exposure	-	-
	InsufficientAttackProtection	-	-
3	Cross-Site Request Forgery (CSRF)kl	2	1
	Using Components with Known Vulnerabilities	-	-
	UnderprotectedAPIs	-	-

Tabla 25. Tabla de prioridades de amenazas web del producto listado de criterios de aprobación y rechazo

4. Prioridades de amenazas Mobile

Actualización	Mobile Top 10 2016-Top 10

Prioridad de criticidad	Titulo	Criticidad	Aplicabilidad
-	Uso inadecuado de la plataforma		
-	Almacenamiento de datos inseguros		
-	Comunicación insegura		
-	Autenticación no segura		
-	Criptografía insuficiente		
-	Autorización insegura		
-	Calidad del código del cliente		
-	Manipulación del código		
-	Ingeniería inversa		
-	Funcionalidad Extraña		

Tabla 26. Tabla de prioridades de amenazas mobile del producto listado de criterios de aprobación y rechazo

Paso 4. *Listar criterios de aprobación y rechazo:* Para poder aprobar o rechazar la prueba de un caso es necesario contar con criterios. Estos quedan registrados y pueden reutilizarse en aplicaciones similares.

5. Listado de Criterios de aprobación

Se aprobará al menos que…

Criterio	Aprobación
Contraseñas seguras	La clave debe encriptarse antes de la llamada Post
Firewall	El firewall no expondrá servicios sensibles y/o direcciones internas
Pruebas de acceso	Se instalen diferentes logs servers para que no sea sencilla el borrado de pruebas
Casos de prueba	Los casos de prueba sean satisfactorios en su totalidad.

Tabla 27. Tabla de listado de criterios de aprobación del producto listado de criterios de aprobación y rechazo

6. Listado de Criterios de Rechazo

Se rechazará cuando…

Criterio	Rechazo
Contraseñas seguras	Si la clave puede leerse con un lector de trafico
Inyección SQL	Si ante una sentencia SQL se emite algún indicio del motor de BBDD
Bloqueo al tercer intento	Al tercer intento se debe bloquear la prueba de acceso
Puertos abiertos	Si hay puertos abiertos que no pertenezcan a la aplicación o funcionamiento propio del servidor

Tabla 28. Tabla de listado de criterios de rechazo del producto listado de criterios de aprobación y rechazo

Cumplimentando estos pasos, obtendremos el producto de salida de esta etapa llamado Listado de Criterios de Aprobación y Rechazo (LCAR). (Ver listado de criterios de aprobación y rechazo en sección 10.1 Anexos caso de validación 1)

5.3.3. Aplicación de la etapa de Verificación y Validación

La aplicación de la etapa de verificación y validación, permite ejecutar pruebas de verificación, comparar criterios establecidos, determinar criticidad del riesgo y establecer acciones a tomar. Para llevar a cabo este proceso se cuenta con el documento generado en la etapa predecesora, el listado de aprobación y rechazo, junto con el informe de vulnerabilidades criticas, obteniendo el Informe de Verificación y Validación (IVV) como productos de salida.

La figura 63 sintetiza la aplicación de la última etapa para el caso de estudio:

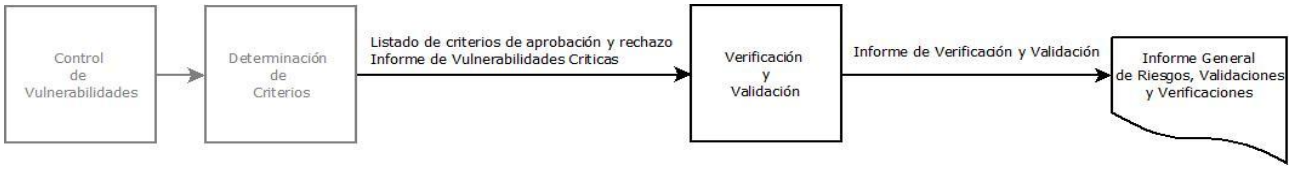

Figura 63. Resumen de la aplicación de la etapa de Modelado de Amenazas con sus productos de entrada y de salida

PRODUCTOS DE ENTRADA para la Verificación y Validación

- Informe de vulnerabilidades criticas
- Listado de Criterios de aprobación y rechazo

PRODUCTOS DE SALIDA para el Verificación y Validación

Paso 1. *Ejecutar pruebas de verificación:* Ante la actualización de vulnerabilidades, puede ser necesario re ejecutar pruebas para verificar nuevamente Si es necesario realizar nuevas pruebas sobre la aplicación, en este paso se plantean.
Se toma como basamento la etapa de planeamiento de pruebas para esto.

2. Casos de prueba

Código caso	*Titulo*	*Condiciones*	*Datos de entrada*	*Resultados esperados*	*Etapa de fase de ejecución*
1	Revisión general de funcionamiento de login	Encontrarse en la página de acceso a listas de precios	Usuario y contraseña valida	Consulta en la BBDD correcta con contraseña encriptada	Escaneo

Tabla 29. Tabla de casos de prueba ejemplo para el producto informe de verificación y validación

Paso 2. *Validar los resultados:* Luego de la ejecución de las pruebas necesarias, se validan respecto a los criterios estipulados.

Paso 3. *Acciones a tomar:* Dentro de este punto, se establecen que acciones se tornarán acerca de los riesgos asociados identificados.

Cumplimentando estos pasos, obtendremos el último producto de salida de esta etapa llamado Informe de Verificación y Validación (IVV).

5.3.4. Obtención documento “Informe General de Riesgos, Validaciones y Verificaciones (IGRVV)”

La aplicación de la todas las etapas que conforman la 1° fase de este proceso y compilando todos los productos obtenidos, se obtiene el documento final de la fase.

6. CASO DE VALIDACIÓN 2

En esta sección se analiza como caso de validación una web de posteos tipo blog. Al igual que el primer caso sometido a pruebas, en la sección 6.1 se aplica la fase de planificación de testeo, en la sección 6.2, se aplica la fase de ejecución del testeo y por ultimo en la sección 6.3 se aplican las actividades correspondientes a la fase de mantenimiento.

6.1. APLICACIÓN DE LAS ACTIVIDADES DE LA FASE DE PLANIFICACION DE TESTEO

En esta sección se aplicarán al caso de validación las etapas correspondientes a la fase de Planificación: Recopilación de Información (sección 6.1.1), Análisis de Vulnerabilidades (sección 6.1.2), Modelado de Amenazas (sección 6.1.3) y el Planeamiento de Pruebas (sección 6.1.4). Obteniendo como producto final de la fase el Plan Integral de Testeo por Hacking Ético (PITHE) (sección 6.1.5).

Se procede a aplicar la etapas de esta primer fase, siguiendo los pasos especificados en la tabla 4.1 y que se describen con detalle en la figura 4.5 de la sección 4.1.2.2 del capítulo 4. La etapa se inicia con el análisis de requerimientos de desarrollo obtenido en la fase inicial del proceso de software junto con la documentación de la aplicación que se genera durante todo el proceso de desarrollo.

6.1.1. Aplicación de la etapa de Recopilación de Información

La aplicación de la etapa de recopilación de información es la que permite documentar el tipo de aplicación a testear, establecer límites de componentes a someter en prueba, delimitar el alcance de la prueba e identificar los primeros riesgos asociados. Para llevar a cabo este proceso se cuenta con el Análisis de requerimientos de desarrollo y la Documentación de la aplicación como productos de entrada, y se obtiene el Informe de dominio (ID) como producto de salida.

PRODUCTOS DE ENTRADA para la Recopilación de Información

- Análisis de requerimientos de desarrollo

- Documentación de la aplicación
  - Manuales técnicos.
  - Manuales de trazabilidad.
  - Elementos creados y/o modificados.
  - Documentación de requisitos relevados

PRODUCTO DE SALIDA para la Recopilación de Información

Paso 1. *Identificar a los responsables del proyecto:* Sumado a los dos productos de entrada (Análisis de requerimientos de desarrollo y documentación de la aplicación), es de suma importancia registrar a los responsables del proyecto de todas las áreas involucradas para dinamizar consultas o evacuación de dudas durante el nuevo proceso de testeo.
Dentro del producto generado en esta etapa, el Informe de Dominio, se encuentra el apartado para completar esta información.

1.1 Responsabilidades

Responsables de proyecto		
Nombre y Apellido	Cargo	Sector
Pablo Herrera	Desarrollador	Sistemas
Sergio González	Analista	Sistemas
Damian Juarte	Analista Seguridad Informática	Tecnología

Tabla 30. Tabla de responsabilidades del producto informe de dominio

Paso 2. *Documentar el objetivo del proyecto:* En primera instancia, se describe el objetivo del proyecto para que todos los profesionales involucrados puedan consultarlo.

1.2 Objetivo del proyecto:
El objetivo de este proyecto es incorporar un canal web de intercambio de información entre moderadores y usuarios. Se pretende una plataforma del tipo blog, para permitir el intercambio de información dinámica. En principio habrá un moderador que es el administrador del sitio. Se desea que esta página de blog muestre de forma ascendente las entradas que se van produciendo, teniendo una búsqueda general.
Por otro lado, cada entrada puede recibir comentarios de los internautas que naveguen la página.

Figura 64. Captura del objetivo del proyecto del producto informe del dominio

Paso 3. *Describir las funcionalidades afectadas:* El testeo no siempre es sobre un proyecto completo, sino que se puede someter a pruebas solo algunas funcionalidades, esto se describe para dejar en conocimiento el alcance de las pruebas.

1.3 Funcionalidades afectadas:
La funcionalidad a realizar es nueva sobre la página y será colgada desde una nueva opción de menú, siendo completamente independiente de las funcionalidades existentes en el dominio.

Figura 65. Captura de funcionalidades afectadas del proyecto del producto informe del dominio

Paso 4. *Resumir el alcance funcional de la solución:* Este apartado ayudará en etapas futuras, donde se deban armar casos de pruebas, contando con un resumen del alcance funcional del proyecto esa tarea será facilitada de gran manera.

1.4 Descripción funcional:
El canal de acceso debe ser vía navegador web y sin necesidad de contar con ninguna instalación adicional, ni utilizar componentes que puedan fallar en navegadores conocidos.
Este nuevo desarrollo se dispondrá en una nueva opción de menú que se define como "Noticias".
Las entradas del blog pueden ser vistas por cualquier internauta que ingrese a la página. Asimismo podrá ingresar un comentario en respuesta a cualquier entrada. Estas entradas serán moderadas por el administrador, en búsqueda de no perder reputación de marca ante entradas que puedan ser susceptibles.
Se dispondrá también una búsqueda a modo de filtro para encontrar entradas antiguas.
Dentro del listado inicial solo se verá un párrafo de la entrada y luego, presionando el botón ver más, se verá la entrada completa.

Figura 66. Captura de descripción funcional del proyecto del producto informe del dominio

Paso 5. *Evaluación de primeros riesgos asociados*: Dentro de este conocimiento de la aplicación, se encuentra un cuestionario sencillo el cual posee una valoración y según las respuestas nos dará un panorama cuantos riesgos posee la aplicación que estamos sometiendo a pruebas.

2. Evaluación de riesgos asociados:
El nivel de riesgo se valora entre 1 y 5. Se consideran los siguientes aspectos para determinar que una iniciativa es de riesgo alto (4 ó 5)

Preguntas	**Nivel de riesgo**	**Respuesta**
¿Se usa una nueva tecnología?	Riesgo 4	No
¿La operativa propuesta ha provocado algún incidente en la industria?	Riesgo 4	Si
¿Se utiliza un nuevo mecanismo de autenticación?	Riesgo 5	No
¿Hay externalización de datos, personas o procesos?	Riesgo 5	No
¿Hay salida de datos? ¿De dónde sale? ¿Qué tipo de datos son?	Riesgo 5	No
¿Se habilita un nuevo tipo de acceso a recursos informáticos?	Riesgo 4	No

¿La iniciativa se apoya sobre un componente/operativa declarado como riesgo activo?	Riesgo 5	No
¿Quién es el usuario consumidor?		Usuario Final
¿Desde dónde se accede a la información?		Por web
¿Qué información se maneja?		Noticias de la empresa
¿La iniciativa contempla comunicación directa? Correo electrónico, SMS, Push, etcétera.	Riesgo 3	Si

Tabla 31. Tabla de riesgos asociados del producto informe de dominio

Cumpliendo estos ítems tendremos generado el documento llamado Informe de Dominio (ID). (Ver informe de dominio en sección 10.2 Anexos caso de validación 2)

6.1.2. Aplicación de la etapa de Análisis de Vulnerabilidades

La aplicación de la etapa de análisis de vulnerabilidades, permite listar vulnerabilidades, determinar probabilidad de vulnerabilidad y realizar un ranking de vulnerabilidades propicias. Para llevar a cabo este proceso se cuenta con el Análisis de requerimientos de desarrollo, la Documentación de la aplicación y el Informe de Dominio generado en la etapa predecesora como productos de entrada, obteniendo el Informe de Vulnerabilidades (IV) como producto de salida.

PRODUCTOS DE ENTRADA para el Análisis de Vulnerabilidades

- Análisis de requerimientos de desarrollo
- Documentación de la aplicación
- Informe de Dominio

PRODUCTO DE SALIDA para el Análisis de Vulnerabilidades

Paso 1. *Revisar vulnerabilidades:* Junto con lo analizado en la etapa anterior, se sabrá si las vulnerabilidades a revisar son de tipo web o móvil. Se listarán las amenazas mas criticas dadas por la fundación OWASP, la cual al momento de escribir esta tesis, confecciona

un documento de alto nivel que se centra sobre las vulnerabilidades más críticas de las aplicaciones web y móviles. Existe en el producto un apartado para completar ante vulnerabilidades específicas.

Paso 2. *Indicar vulnerabilidades propicias:* Teniendo listadas las amenazas mas criticas y recurrentes, procederemos a marcar cuales de estas pueden atacar directamente a nuestro sistema. Obteniendo de esta forma un ranking de vulnerabilidades propicias.

1. Listado de vulnerabilidades Web

Actualización	OWASP_Top_10-2017

Titulo	Sistema Propicio
Inyección	Si
Pérdida de Autenticación y Gestión de Sesiones	No
Cross-Site Scripting (XSS)	SI
Rotura de control de acceso	No
Security Misconfiguration	No
Sensitive Data Exposure	No
InsufficientAttackProtection	Si
Cross-Site Request Forgery (CSRF)	No
Using Components with Known Vulnerabilities	No
UnderprotectedAPIs	No

Tabla 32. Tabla de vulnerabilidades web del producto informe de vulnerabilidades

2. Listado de vulnerabilidades Mobile

Actualización	Mobile_Top_10-2016

Titulo	Sistema Propicio
Uso inadecuado de la plataforma	No
Almacenamiento de datos inseguros	No
Comunicación insegura	No
Autenticación no segura	No
Criptografía insuficiente	No
Autorización insegura	No
Calidad del código del cliente	No
Manipulación del código	No
Ingeniería inversa	No
Funcionalidad Extraña	No

Tabla 33. Tabla de vulnerabilidades mobile del producto informe de vulnerabilidades

Cumpliendo con estos pasos, obtendremos el producto de salida de la etapa llamado Informe de Vulnerabilidades (IV). (Ver informe de vulnerabilidades en sección 10.2 Anexos caso de validación 2)

6.1.3. Aplicación de la etapa de Modelado de Amenazas

La aplicación de la etapa de análisis de vulnerabilidades, permite identificar información sensible, describir la arquitectura, descomponer la aplicación, identificar y documentar vulnerabilidades criticas y asignar prioridades de ataque a las vulnerabilidades mas criticas. Para llevar a cabo este proceso se cuenta con los documentos de entrada de la fase junto con el generado en la etapa predecesora, obteniendo el Informe de Vulnerabilidades (IA) y las prioridades de criticidad (PC) como productos de salida.

PRODUCTOS DE ENTRADA para el Modelado de Amenazas

- Análisis de requerimientos de desarrollo
- Documentación de la aplicación
- Informe de Dominio
- Informe de Vulnerabilidades

PRODUCTOS DE SALIDA para el Modelado de Amenazas

Paso 1. *Analizar nivel de criticidad de amenazas:* Junto con el documento generado de la etapa de Análisis de Vulnerabilidades, se deberá indagar sobre cada una de las amenazas propicias y asignarle un valor de criticidad. Esta valoración es de 1 a 5, siendo 1 Baja criticidad y 5 Alta Criticidad.

Paso 2. *Analizar nivel de aplicabilidad de amenazas:* De la misma forma que se evalúa que tan crítica puede ser una vulnerabilidad para el tipo de sistema a analizar, se identifica el nivel de aplicabilidad, diciéndonos que tanto aplica esa amenaza al sistema puesto a prueba. Esta valoración es de 1 a 5, siendo 1 Baja aplicabilidad y 5 Alta Aplicabilidad.

Estos dos últimos pasos se reflejan en el documento de salida de la siguiente forma:

1. Listado de Amenazas Web

Actualización	OWASP Top 10 - 2017

El nivel de criticidad y aplicabilidad se valora entre 1 y 5.

Titulo	Criticidad	Aplicabilidad
Inyección	4	4
Pérdida de Autenticación y Gestión de Sesiones	-	-
Cross-Site Scripting (XSS)	3	2
Rotura de control de acceso	-	-
Security Misconfiguration	-	-
Sensitive Data Exposure	-	-
InsufficientAttackProtection	3	3
Cross-Site Request Forgery (CSRF)	-	-
Using Components with Known Vulnerabilities	-	-
UnderprotectedAPIs	-	-

Tabla 34. Tabla de listado de amenazas web del producto informe de amenazas

2. Listado de Amenazas Mobile

Actualización	Mobile Top 10 2016-Top 10

Titulo	Criticidad	Aplicabilidad
Uso inadecuado de la plataforma	-	-
Almacenamiento de datos inseguros	-	-
Comunicación insegura	-	-
Autenticación no segura	-	-
Criptografía insuficiente	-	-
Autorización insegura	-	-
Calidad del código del cliente	-	-
Manipulación del código	-	-
Ingeniería inversa	-	-
Funcionalidad Extraña	-	-

Tabla 35. Tabla de listado de amenazas mobile del producto informe de amenazas

Contando con estos pasos, obtendremos el producto de salida de la etapa llamado Informe de Amenazas (IA). (Ver informe de amenazas en sección 10.2 Anexos caso de validación 2)

Paso 3. *Priorizar amenazas críticas:* Todas las amenazas valoradas en los pasos anteriores se deben colocar en prioridad para que los profesionales sepan cómo y qué puntos encarar con mayor urgencia. Esto se realiza colocando en formato de ranking las amenazas evaluadas en los puntos anteriores.

1. Prioridades de amenazas Web

Actualización	OWASP Top 10 - 2017

Prioridad de criticidad	Titulo	Criticidad	Aplicabilidad
1	Inyección	4	4
	Pérdida de Autenticación y Gestión de Sesiones	-	-
2	Cross-Site Scripting (XSS)	3	2
	Rotura de control de acceso	-	-
	Security Misconfiguration	-	-
	Sensitive Data Exposure	-	-
3	InsufficientAttackProtection	3	3
	Cross-Site Request Forgery (CSRF)	-	-
	Using Components with Known Vulnerabilities	-	-
	UnderprotectedAPIs	-	-

Tabla 36. Tabla de prioridades de amenazas web del producto prioridades de criticidad

2. Prioridades de amenazas Mobile

Actualización	Mobile Top 10 2016-Top 10

Prioridad de criticidad	Titulo	Criticidad	Aplicabilidad
-	Uso inadecuado de la plataforma	-	-
-	Almacenamiento de datos inseguros	-	-
-	Comunicación insegura	-	-
-	Autenticación no segura	-	-
-	Criptografía insuficiente	-	-
-	Autorización insegura	-	-
-	Calidad del código del cliente	-	-
-	Manipulación del código	-	-
-	Ingeniería inversa	-	-
-	Funcionalidad Extraña	-	-

Tabla 37. Tabla de prioridades de amenazas mobile del producto prioridades de criticidad

Cumplimentando estos pasos, obtendremos el segundo producto de salida de esta etapa llamado Prioridades de criticidad (PC). (Ver prioridades de criticidad en sección 10.2 Anexos caso de validación 2)

6.1.4. Aplicación de la etapa de Planeamiento de Pruebas

La aplicación de la etapa de planeamiento de pruebas, permite documentar pruebas, armar casos de pruebas, establecer criterios de aprobación y rechazo, determinar grados de criticidad, calendarizar hitos, estimar esfuerzos y finalmente elaborar el plan. Esta etapa cuenta con todos

los documentos generados durante la fase, generando los Casos de pruebas (CP), los calendarios de pruebas y esfuerzo (CPE) y el Informe de Criterios (IC).

PRODUCTOS DE ENTRADA para el Planeamiento de Pruebas

- Análisis de requerimientos de desarrollo
- Documentación de la aplicación
- Informe de Amenazas
- Prioridades de criticidad

PRODUCTOS DE SALIDA para el Planeamiento de Pruebas

Paso 1. *Armado de casos de prueba:* Teniendo priorizadas las amenazas críticas y aplicables, se puede proceder a crear los diferentes casos de prueba que satisfagan las pruebas en esos puntos sensibles dentro de la aplicación. En cada caso se deberá indicar un titulo, las condiciones, los datos de entrada y los resultados esperados. Y junto a cada uno de los casos, se indicará en que etapa de la fase siguiente se realizaran las pruebas correspondientes.

2. Casos de prueba

Código caso	*Titulo*	*Condiciones*	*Datos de entrada*	*Resultados esperados*	*Etapa de fase de ejecución*
1	Revisión general de pagina creada	Encontrarse en la página de noticias	-	Vulnerabilidades detectadas	Reconocimiento
2	Inyección SQL	Estar dentro de la página y en la creación de comentario o búsqueda.	Ingresar una sentencia valida de SQL, tal como “select * from usuarios”	Ningún resultado de consulta en BBDD o error que pueda dar indicio de datos	Reconocimiento
3	Comentario trolls	Estar dentro del ingreso de comentario	Texto valido	Captcha anti trolls	Escaneo
4	Intento de penetración en hosting de pagina		Relevamientos previos	Puertos y servicios monitoreados	Todas

Tabla 38. Tabla de casos de prueba del producto casos de prueba

Paso 2. *Priorizar casos de prueba:* Una vez identificados todos los casos de prueba, se procede a catalogarlos por su criticidad. Utilizando la misma valoración que en etapas anteriores, donde el valor 1 significa Baja Criticidad y 5 Alta Criticidad.

3. Criticidad

Código de caso	Nivel de criticidad	Observaciones
1	4	
2	4	
3	5	
4	3	

Tabla 39. Tabla de criticidad del producto casos de prueba

Paso 3. *Estimar esfuerzo:* Por cada caso indicaremos que esfuerzo medido en horas/hombre insumirán.

4. Estimación de esfuerzo

Código de caso	Esfuerzo aproximado	Observaciones
1	8 hs/h	
2	5 hs/h	
3	6hs/h	
4	16 hs/h	

Tabla 40. Tabla de estimación de esfuerzos del producto casos de prueba

Paso 4. *Calendarizar pruebas:* Teniendo registrado el esfuerzo de cada caso a probar, se puede realizar la calendarización de la prueba completa. Esto cobra muchísimo sentido en proyectos de gran envergadura y donde hay casos que prueba con dependencias cruzadas o muchos profesionales involucrados por ejemplo.

1. Calendarización

Esta planilla se refleja desde el archivo Diagrama Gantt.xlsx

Casos	**Inicio**	**Duración (días)**	**Fin**
1	21/05/2018	1	22/05/2018
2	23/05/2018	0,625	23/05/2018
3	24/05/2018	0,75	24/05/2018
4	29/05/2018	2	31/05/2018

Tabla 41. Tabla de calendarización del producto calendario de pruebas y esfuerzo

2. Gantt

Este grafico se refleja desde el archivo Diagrama Gantt.xlsx

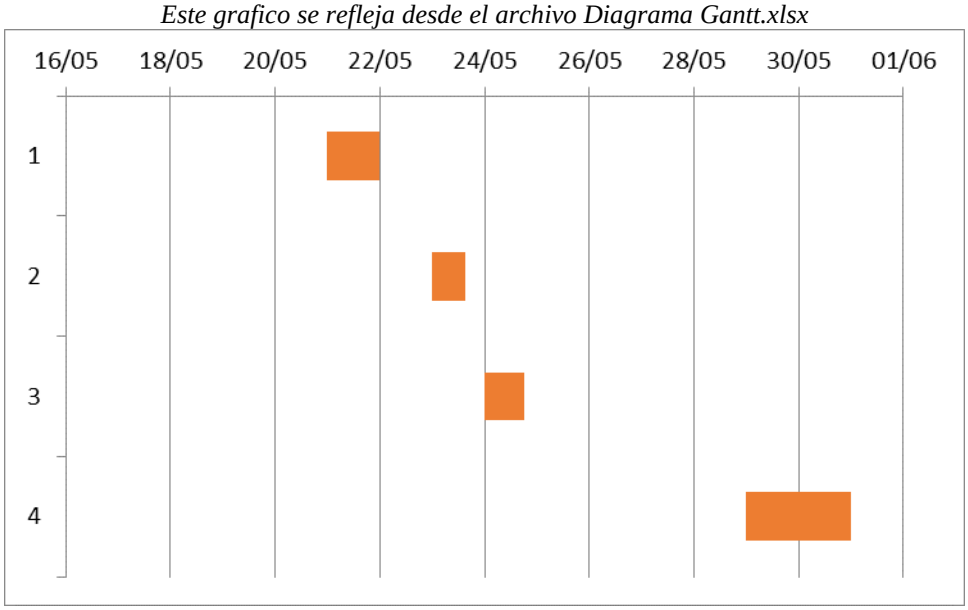

Figura 67. Captura de grafico de Gantt del producto casos de prueba

Cumplido estos pasos se obtienen como productos de salida los Casos de prueba (CP) y el Calendario de pruebas y esfuerzo (CPE). (Ver casos de prueba y el calendario de pruebas y esfuerzo en sección 10.2 Anexos caso de validación 2)

Paso 5. *Listar criterios de aprobación y rechazo:* Para poder aprobar o rechazar la prueba de un caso es necesario contar con criterios. Estos quedan registrados y pueden reutilizarse en aplicaciones similares.

2. Listado de Criterios de aprobación

Se aprobará al menos que…

Criterio	Aprobación
Comentarios de humanos	Los comentarios deben restringirse a ingresos hechos por humanos y no por trolls o automatismos
Firewall	El firewall no expondrá servicios sensibles y/o direcciones internas
Casos de prueba	Los casos de prueba sean satisfactorios en su totalidad.

Tabla 42. Tabla de listado de criterios de aprobación del producto informe de criterios

3. Listado de Criterios de Rechazo

Se rechazará cuando…

Criterio	Rechazo
Inyección SQL	Si ante una sentencia SQL se emite algún indicio del motor de BBDD
Puertos abiertos	Si hay puertos abiertos que no pertenezcan a la aplicación o funcionamiento propio del servidor
Escáner de vulnerabilidades	Si al menos hay un riesgo de tipo grave.

Tabla 43. Tabla de listado de criterios de aprobación del producto informe de criterios

Al culminar este ítem obtenemos el Informe de Criterios (IC). (Ver informe de criterios en sección 10.2 Anexos caso de validación 2)

6.1.5. Obtención documento "Plan Integral de Testeo por Hacking Ético (PITHE)"

La aplicación de la todas las etapas que conforman la 1° fase de este proceso y compilando todos los productos obtenidos, se obtiene el documento final de la fase.

6.2. APLICACIÓN DE LAS ACTIVIDADESD DE LA FASE DE EJECUCION DE TESTEO

En esta sección se procederá a aplicar al caso de validación las etapas correspondientes a la fase de Ejecución: Reconocimiento (sección 6.2.1), Escaneo (sección 6.2.2), Ganancia de acceso (sección 6.2.3), Mantenimiento de Acceso (sección 6.2.4) y Eliminación de pruebas (sección 6.2.5). Obteniendo como producto final de la fase el Informe Final de Testeo por Hacking Ético (IFTHE) (sección 6.2.6).

A continuación se procede a aplicar las etapas de esta fase de ejecución, siguiendo los pasos especificados en la tabla 4.1 y que se describen con detalle en la figura 4.5 de la sección 4.1.2.2 del capítulo 4. La etapa es iniciada con todos los productos generados en la fase anterior (Planificación), y finalizando con la obtención del informe final del testeo.

En esta sección aparecerán descriptas diferentes herramientas que al momento de escribir esta tesis, se encuentran en el auge de su uso. El proceso planteado debe trascender a las herramientas que se utilicen en cada etapa.

6.2.1. Aplicación de la etapa de Reconocimiento

La aplicación de la etapa de reconocimiento, permite recopilar información, determinar el tamaño de la red, identificar maquinas activas, descubrir puertos abiertos y puntos de acceso, identificar el sistema operativo y realizar los casos de pruebas identificados para esta etapa. Para llevar adelante este proceso se cuenta con el Plan integral de testeo por hacking ético obtenido en la etapa predecesora como productos de entrada obteniendo el Informe de Arquitectura de Red (IAR) como producto de salida.

PRODUCTOS DE ENTRADA para la etapa de Reconocimiento

- Plan Integral de Testeo por Hacking Ético

PRODUCTOS DE SALIDA para la etapa de Reconocimiento

Paso 1. *Descubrir Sistema Operativo:* Gracias a la herramienta nmap se puede determinar el Sistema Operativo que se utiliza en destino, esto ayuda a los atacantes a saber con qué comandos y que vulnerabilidades explotar. Dentro del informe se indicará esto.

```
nmap -O antirrobosalto.com.ar
```

```
Service Info: OS: Red Hat Enterprise Linux 7; CPE: cpe:/o:redhat:enterprise_linux:7

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
```

Figura 68. Captura de ejecución para descubrir el sistema operativo

Paso 2. *Identificar maquinas involucradas en la aplicación:* Utilizando la herramienta nmap se pueden obtener las maquinas activas junto con nombres de host y direcciones IP.

```
nmap -sP antirrobosalto.com.ar
```

```
Nmap scan report for antirrobosalto.com.ar (190.106.131.237)
Host is up (0.0029s latency).
rDNS record for 190.106.131.237: web333.fangio.net
Nmap done: 1 IP address (1 host up) scanned in 0.29 seconds
```

Figura 69. Captura de ejecución para identificar maquinas involucradas

Paso 3. *Identificar los puertos y puntos de acceso:* Junto con la misma herramienta se obtiene el listado de los puertos y posibles puntos de acceso abiertos en el destino.

```
nmap -v antirrobosalto.com.ar
```

```
Initiating SYN Stealth Scan at 14:00
Scanning antirrobosalto.com.ar (190.106.131.237) [1000 ports]
Discovered open port 993/tcp on 190.106.131.237
Discovered open port 110/tcp on 190.106.131.237
Discovered open port 995/tcp on 190.106.131.237
Discovered open port 3306/tcp on 190.106.131.237
Discovered open port 587/tcp on 190.106.131.237
Discovered open port 25/tcp on 190.106.131.237
Discovered open port 443/tcp on 190.106.131.237
Discovered open port 53/tcp on 190.106.131.237
Discovered open port 143/tcp on 190.106.131.237
Discovered open port 80/tcp on 190.106.131.237
Discovered open port 21/tcp on 190.106.131.237
SYN Stealth Scan Timing: About 34.37% done; ETC: 14:01 (0:00:59 re
maining)
Discovered open port 465/tcp on 190.106.131.237
Completed SYN Stealth Scan at 14:01, 65.39s elapsed (1000 total po
rts)
Nmap scan report for antirrobosalto.com.ar (190.106.131.237)
```

Figura 70. Captura de ejecución para identificar puertos y puntos de acceso

Paso 4. *Identificar DNS:* Ejecutando los mismos parámetros sobre la herramienta se obtiene el listado de dns configurados.

```
nmap -v antirrobosalto.com.ar
```

```
rDNS record for 190.106.131.237: web333.fangio.net
```

Figura 71. Captura de ejecución para identificar DNS

Paso 5. *Identificar Dominios activos:* Ejecutando los mismos parámetros sobre la herramienta se obtiene el listado de dns configurados.

El DNS obtenido en el paso anterior es el único dominio activo.

Paso 6. *Realizar los casos de prueba identificados:* En la etapa de armado de casos de prueba se identifican en que etapa de esta fase deben realizarse. Para este caso de prueba se encuentran los casos con código 1 y 2.

1	Revisión general de pagina creada	Encontrarse en la página de noticias	-	Vulnerabilidades detectadas	Reconocimiento
2	Inyección SQL	Estar dentro de la página y en la creación de comentario o búsqueda.	Ingresar una sentencia valida de SQL, tal como “select * from usuarios”	Ningún resultado de consulta en BBDD o error que pueda dar indicio de datos	Reconocimiento

Tabla 44. Tabla de casos de prueba a ejecutar en la etapa de reconocimiento

Caso 1: Revisión general de vulnerabilidades

Se utiliza la herramienta owasp zap, la cual simula ataques a sitios web y encuentra vulnerabilidades que pueden ser utilizadas por atacantes externos.

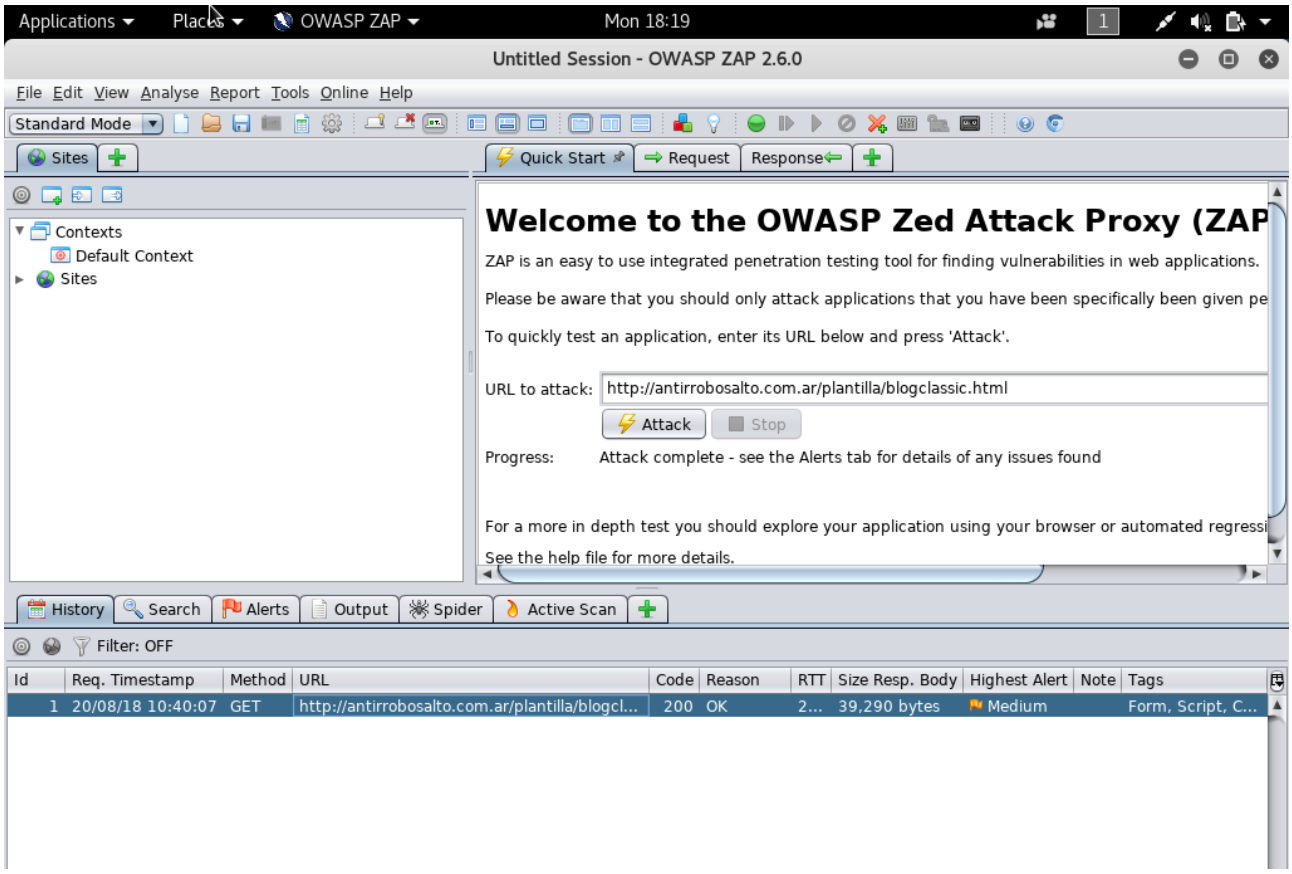

Figura 72. Captura de pantalla de herramienta busca vulnerabilidades

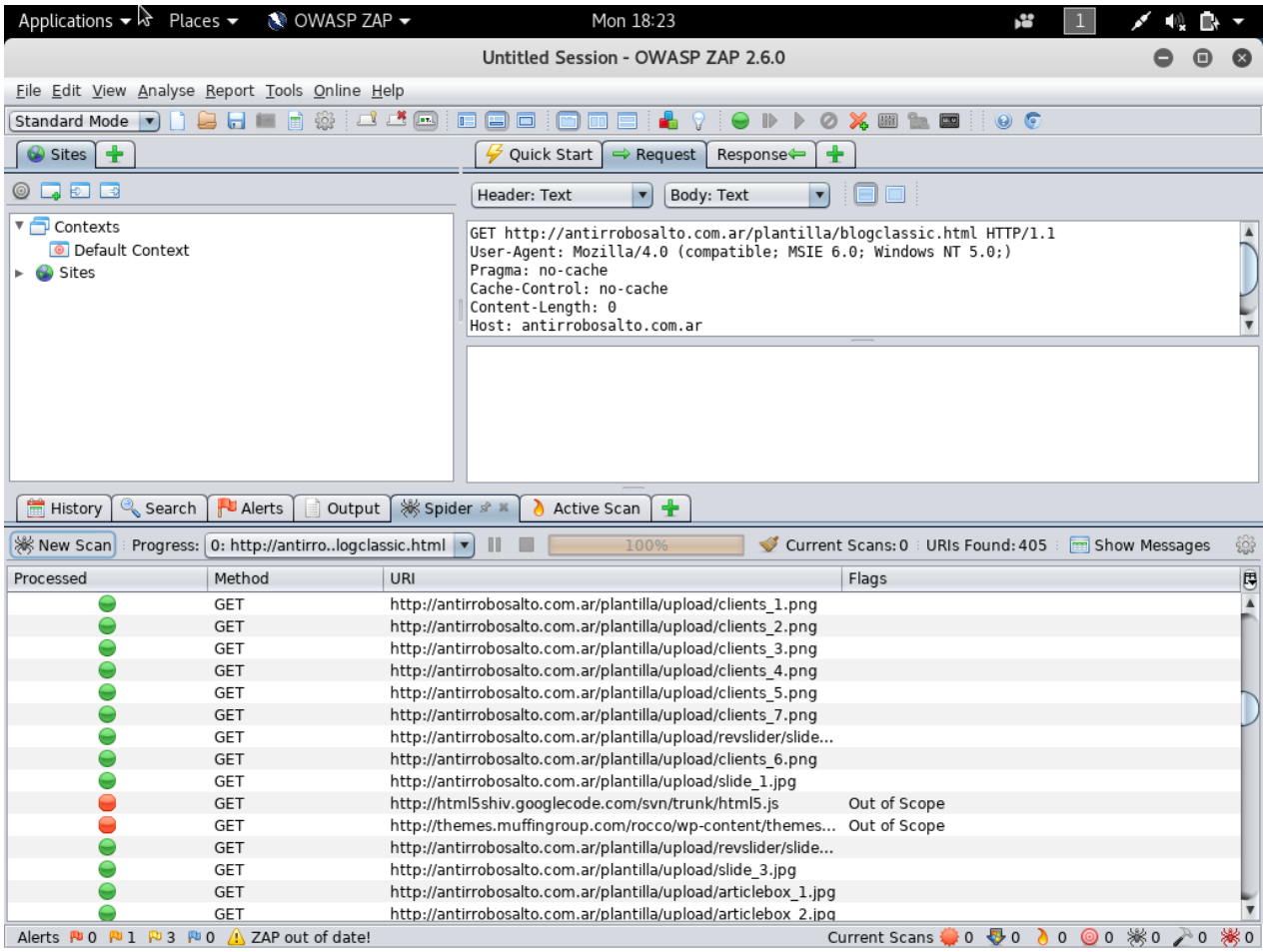

Figura 73. Captura de pantalla de ejecución herramienta busca vulnerabilidades

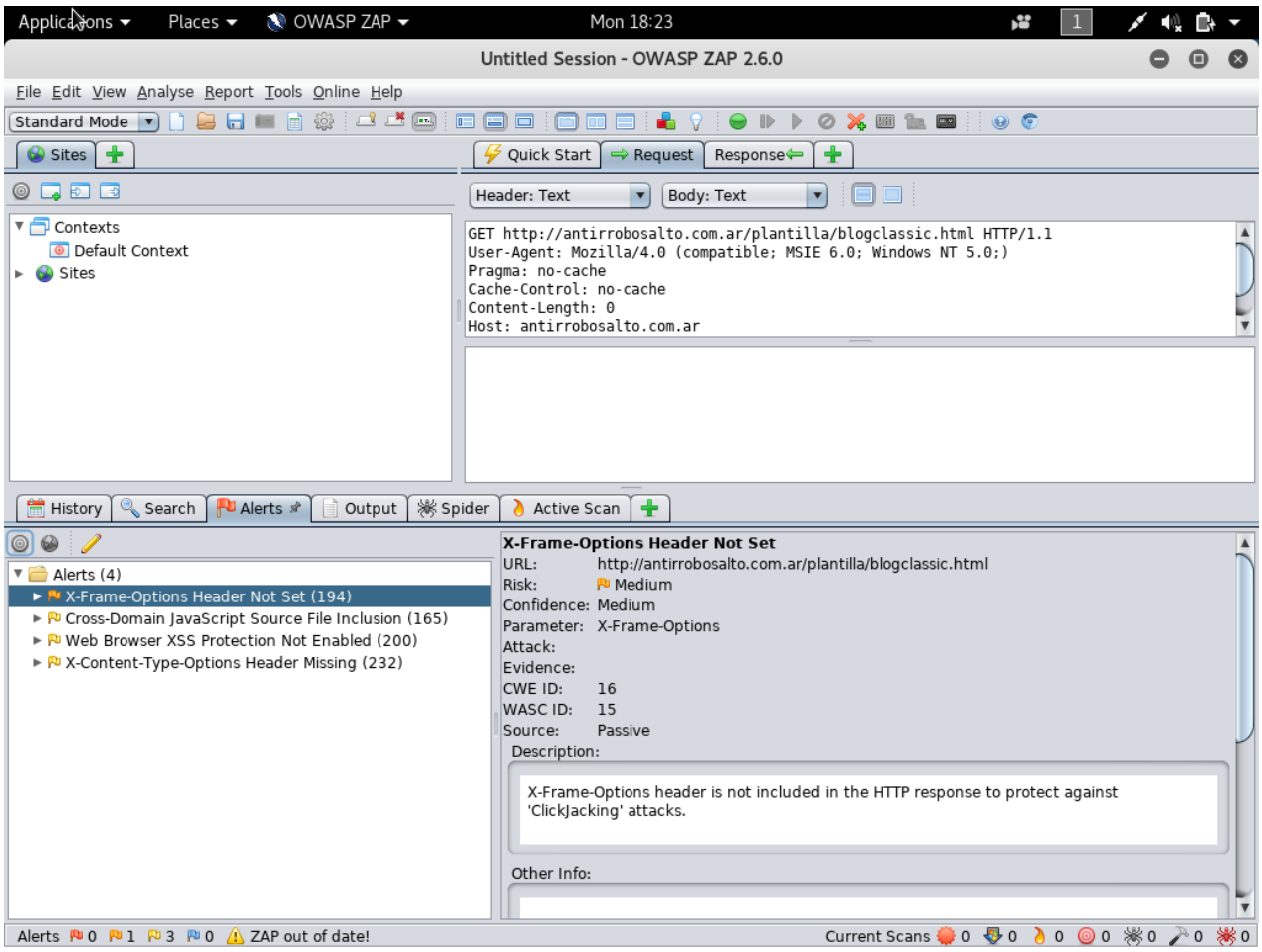

Figura 74. Captura de pantalla de resultado de ejecución herramienta busca vulnerabilidades

Aquí se observa que la herramienta arrojó cuatro alertas, donde una de ellas es del tipo crítico, la cual detectó que son posibles los ataques del tipo Clickjacking por el tipo de configuración en los Frames. Esta detección implicará devolver a instancias de desarrollo la aplicación web.

Por otro lado se detectaron tres vulnerabilidades de riego medio, las cuales deben solucionarse pero por los criterios de rechazo estipulados, no se corre el riego de no poder implementar el desarrollo.

Caso 2: Inyección SQL

Para poder comprobar este caso de prueba se utilizará la herramienta Wireshark, la misma es un analizador de protocolos utilizado para realizar análisis. La funcionalidad que provee es similar a la de tcpdump, permite ver todo el tráfico que pasa a través de una red.

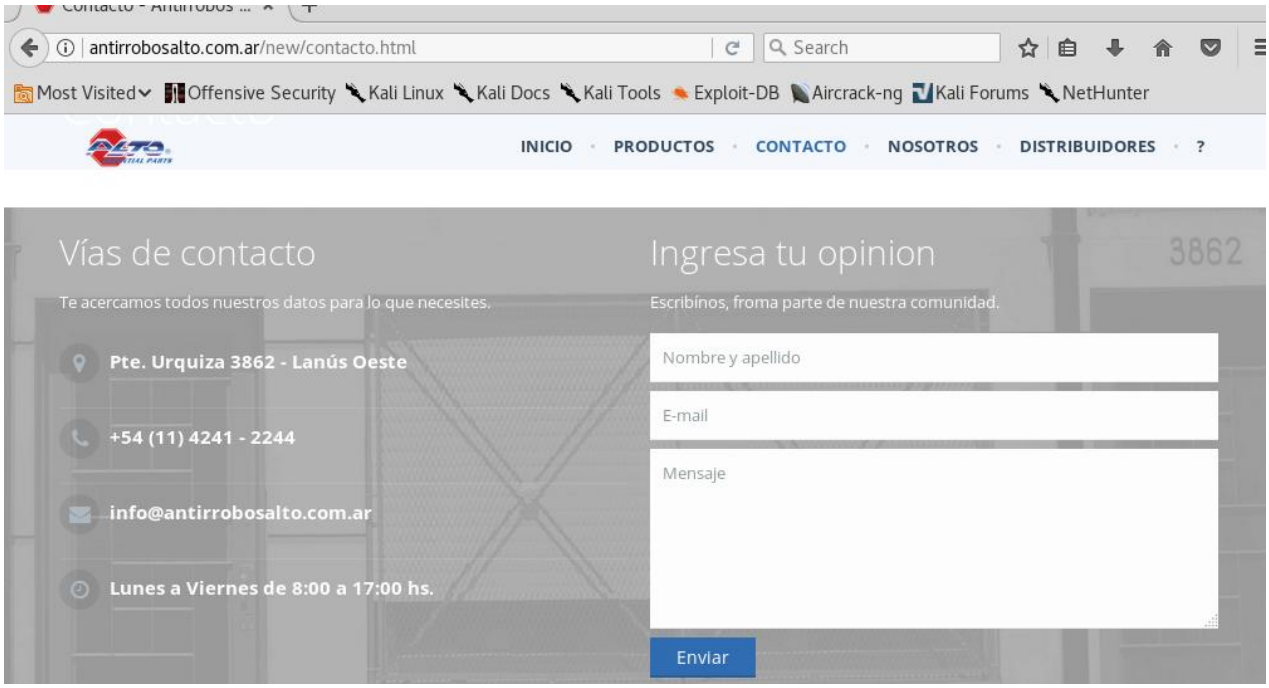

Figura 75. Captura de pantalla de preparación para inyección SQL

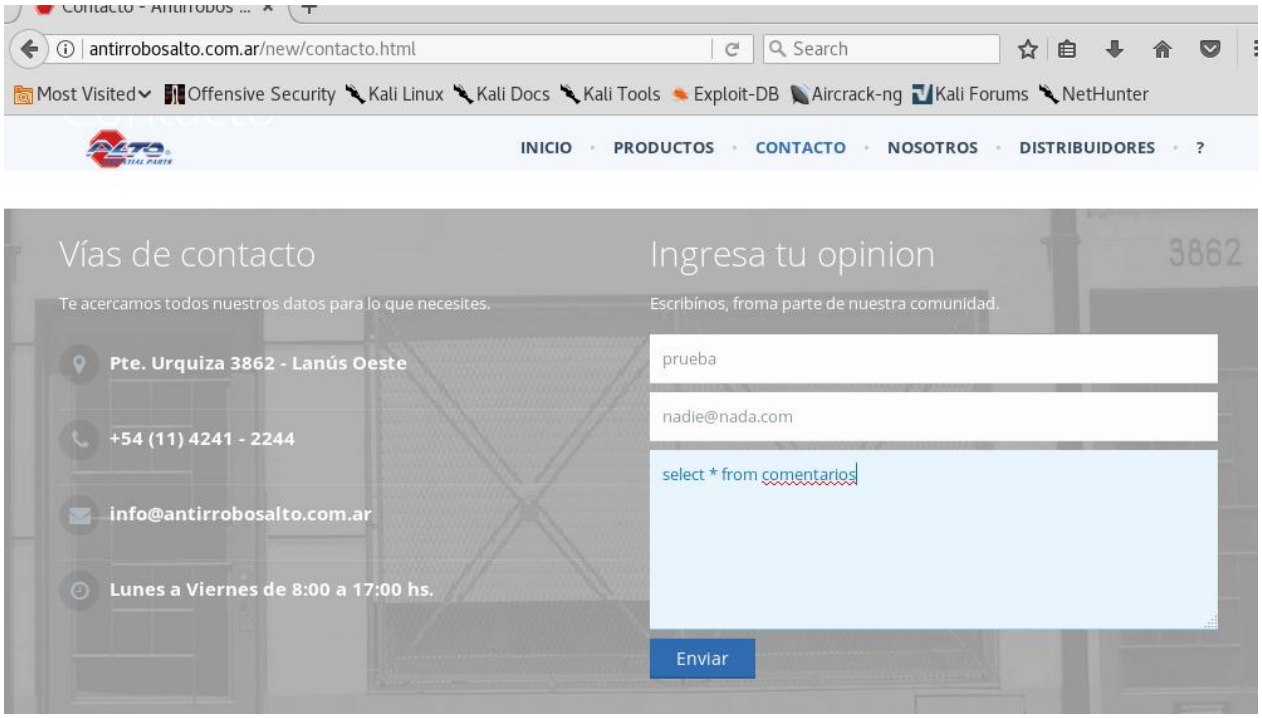

Figura 76. Captura de pantalla de ingreso de inyección SQL

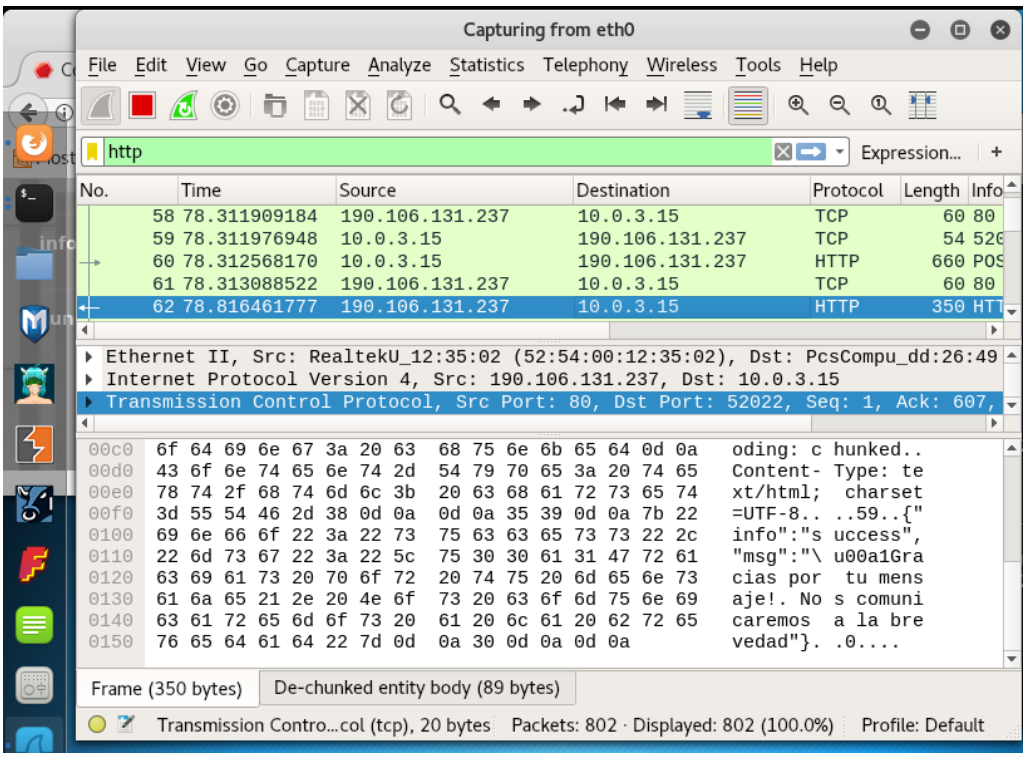

Figura 77. Captura de herramienta de lectura de trafico de ingreso de inyección SQL

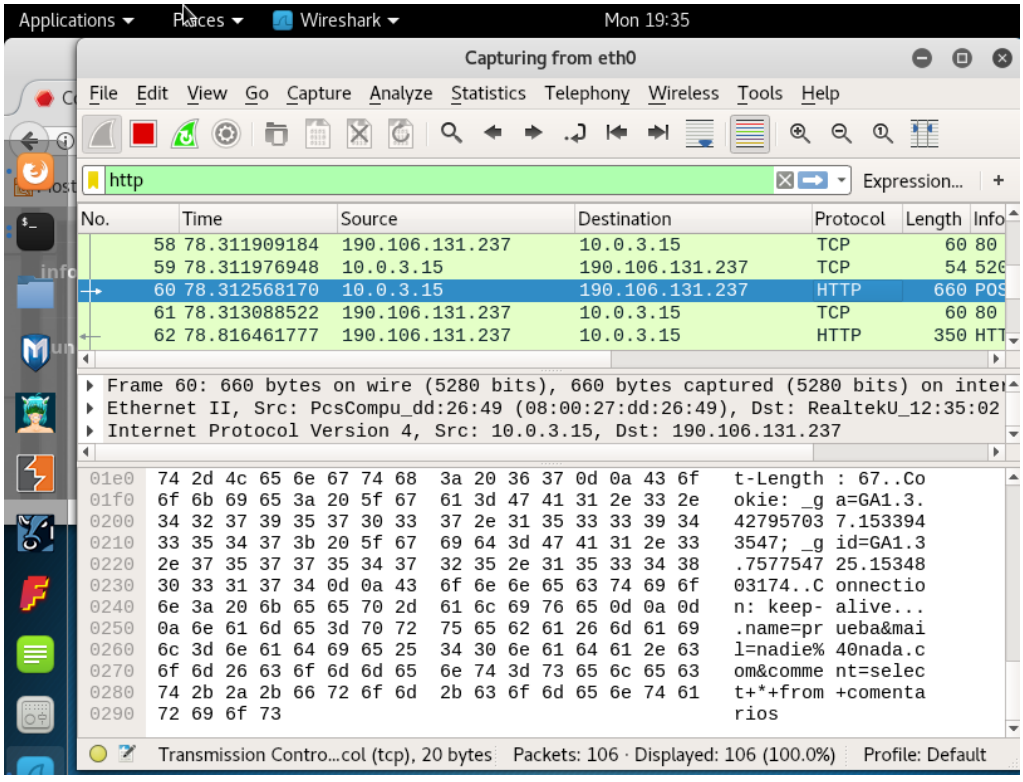

Figura 78. Segunda captura de herramienta de lectura de tráfico de ingreso de inyección SQL

Contando con todos estos pasos, obtendremos el producto de salida de la etapa llamado Informe de Arquitectura de Red (IAR). (Ver informe de arquitectura en sección 10.2 Anexos caso de validación 2)

6.2.2. Aplicación de la etapa de Escaneo

La etapa de escaneo, permite detectar sistemas vivos en la red, descubrir puertos abiertos, encontrar servicios activos y en ejecución y realizar los casos de pruebas identificados para esta etapa. Para llevar adelante este proceso se cuenta con el Plan integral de testeo por hacking ético obtenido en la etapa predecesora como productos de entrada obteniendo el Informe de Escaneo (IE) como producto de salida.

PRODUCTOS DE ENTRADA para la etapa de Escaneo

- Plan Integral de Testeo por Hacking Ético
- Informe de Arquitectura de red

PRODUCTOS DE SALIDA para la etapa de Escaneo

Paso 1. *Descubrir Puertos abiertos y servicios:* Continuando con la herramienta nmap se pueden detallar los puertos abiertos y que servicios son los que los utilizan.

```
nmap --osscan_guess antirrobosalto.com.ar
```

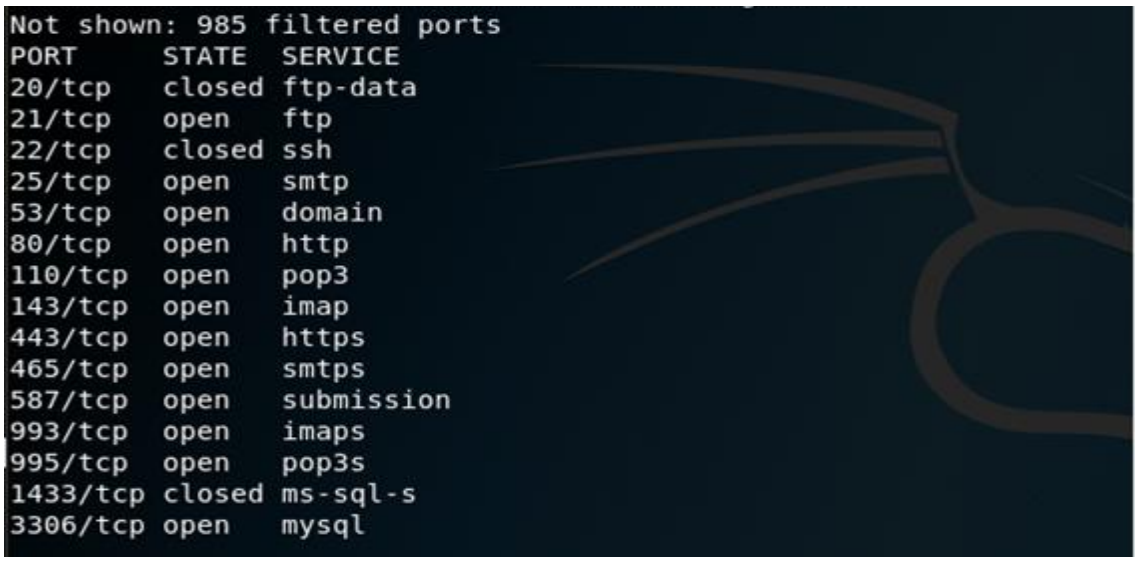

Figura 79. Captura de ejecución para descubrir puertos y servicios

2. Listado de Puertos activos y puntos de acceso

Puerto	Tipo	Estado	Servicio
20	Tcp	Cerrado	ftp-data
21	Tcp	Abierto	ftp
22	Tcp	Cerrado	Ssh
25	Tcp	Abierto	Smtp
53	Tcp	Abierto	Domain
80	Tcp	Abierto	http
110	Tcp	Abierto	Pop3
143	Tcp	Abierto	Imap
443	Tcp	Abierto	https
465	Tcp	Abierto	Smtps
587	Tcp	Abierto	Submission
993	Tcp	Abierto	Imaps
995	Tcp	Abierto	Pop3s
1433	Tcp	Cerrado	Ms-sql-s
3306	Tcp	Abierto	mysql

Tabla 45. Tabla de puertos activos y puntos de acceso del producto casos de prueba

Paso 2. *Descubrir dominios y DNS:* Continuando con la herramienta nmap se pueden descubrir los DNS y dominios mapeados.

```
nmap –-osscan -guess antirrobosalto.com.ar
```

```
rDNS record for 190.106.131.237: web333.fangio.net
```

Figura 80. Captura de ejecución para descubrir dominios y dns

2. Dns y dominios

Dns	Proveedor	Ip
antirrobosalto.com.ar	**web333.fangio.net**	**190.106.131.237**

Tabla 46. Tabla de dns y dominios del producto casos de prueba

Paso 3. *Realizar los casos de prueba identificados:* En la etapa de armado de casos de prueba se identifican en qué etapa de esta fase deben realizarse. Para este caso de prueba se encuentran los casos con código 3.

3	Comentario trolls	Estar dentro del ingreso de comentario	Texto valido	Captcha anti trolls	Escaneo

Tabla 47. Tabla de casos de prueba a ejecutar en la etapa de escaneo

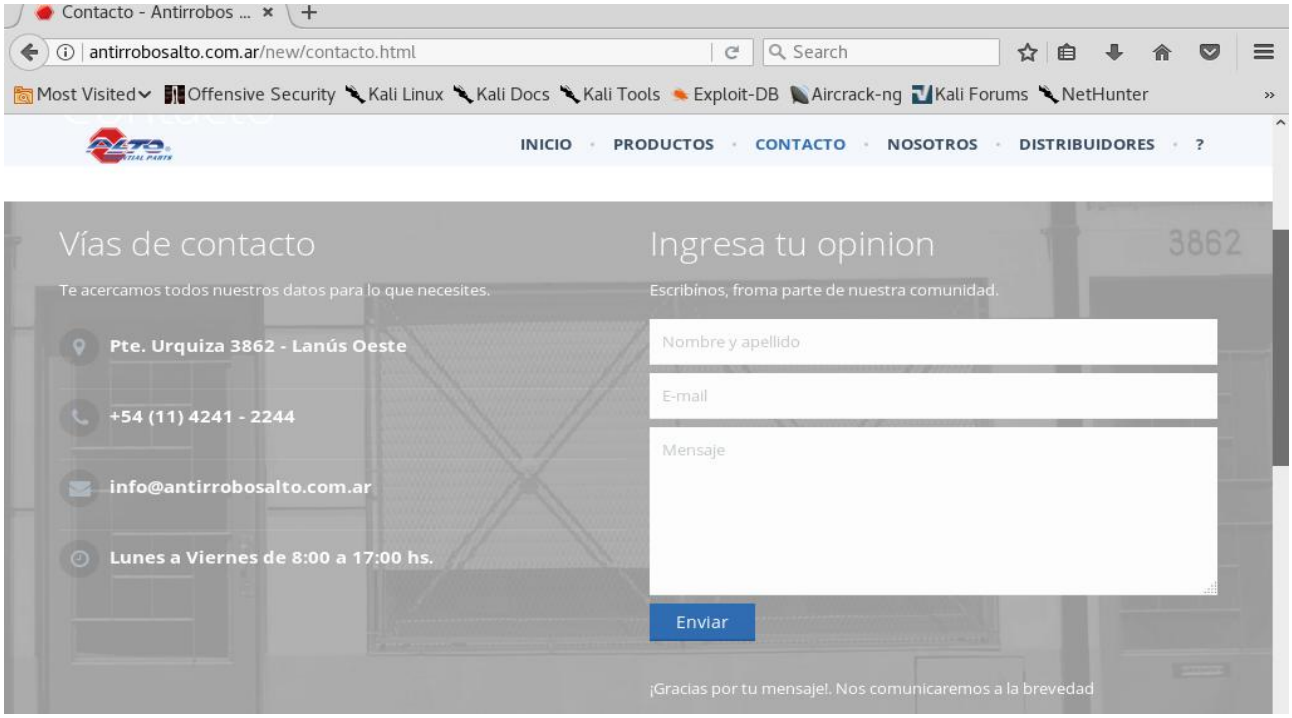

Figura 81. Captura de pantalla de ingreso por prueba anti trolls

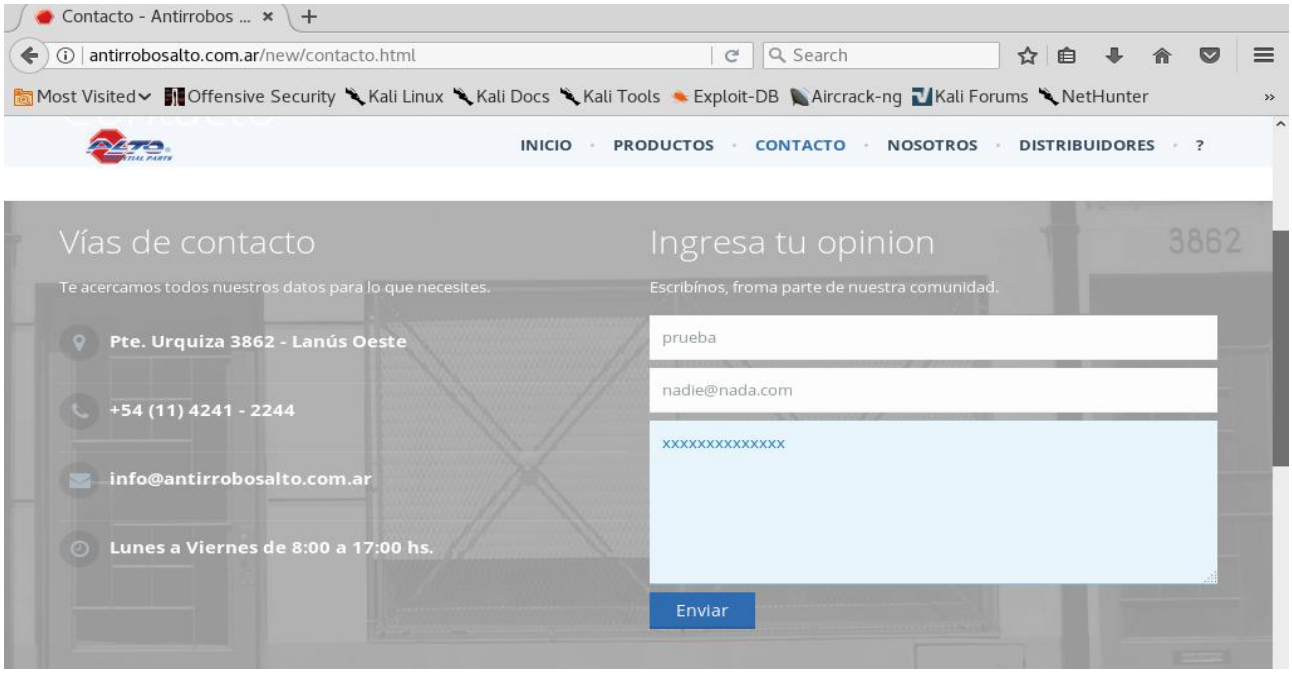

Figura 82. Segunda captura de pantalla de ingreso por prueba anti trolls

No existe ningún mecanismo de control que no permita el ingreso de robots a emitir comentarios.

Si bien los comentarios no son expuestos directamente, ya que se realizan mediante un moderador, el no tener control anti trolls, puede acarrear un problema de denegación de servicio.

Por este motivo, se aconseja agregar una intervención del tipo captcha.

Figura 83. Ejemplo de captcha

Es en este caso donde se encuentra una vulnerabilidad expuesta de tipo media, ya que se encuentra expuesta la página de contacto a ataques robotizados.

Contando con todos estos pasos, obtendremos el producto de salida de la etapa llamado Informe de Escaneo del Sistema (IES). (Ver informe de escaneo de sistema en sección 10.2 Anexos caso de validación 2)

6.2.3. Aplicación de la etapa de Ganancia de Acceso

La etapa de ganancia de acceso, existe para explotar las direcciones, puertos y servicios, buscando obtener un punto de acceso a la maquina víctima. Para llevar adelante este proceso se cuenta con las investigaciones de datos obtenidos en los productos anteriores y obteniendo el Informe de Accesos ganados (IAG) como producto de salida.
Estas etapas y lo mismo con su sucesora, son etapas propias del ciclo del hacking; dentro de este proceso, se forzaran ambas para demostrar que son factibles y que luego siguiendo con puntos de buenas prácticas, pueden mitigarse algunos riesgos.

PRODUCTOS DE ENTRADA para la etapa de Ganancia

- Plan Integral de Testeo por Hacking Ético
- Informe de Arquitectura de red

- Informe de Escaneo de Sistema

PRODUCTOS DE SALIDA para la etapa de Ganancia

Paso 1. *Explotar vulnerabilidad*: Junto con el listado de los puertos descubiertos y vulnerabilidades detectadas, se busca un exploit que permita obtener el control.

Herramientas recomendadas:

- MetaSploit
- Core Impact
- Immunity Canvas
- Milw0rm

Paso 2. *Elevar privilegios*: Para poder ampliar el control y llevarlo a ser total se debe intentar llevar los privilegios a administrador o root, esto se logra con otros tipos de aplicaciones exploits.

Paso 3. *Ejecución remota*: Obteniendo control y privilegios, se ejecutan comandos o aplicaciones de forma remota para mantener el canal.

Herramienta recomendada:

- PsExec

Paso 3. *Generar Informe*: Ante el caso necesario de poner en práctica esta etapa, se elaborará el Informe de Accesos Ganados (IAG).

6.2.4. Aplicación de la etapa de Mantenimiento de Acceso

La etapa de mantenimiento de acceso, existe para mantener el acceso ganado como exclusivo, cargar, descargar y manipular datos, aplicaciones y configuraciones del sistema, intentar usar al sistema para lanzar más ataques. Para llevar adelante este proceso se cuenta con los informes y obteniendo el Informe de Accesos mantenidos (IAM) como producto de salida.

PRODUCTOS DE ENTRADA para la etapa de Mantenimiento

- Plan Integral de Testeo por Hacking Ético

- Informe de Arquitectura de red
- Informe de Escaneo de Sistema
- Informe de Accesos Ganados

Paso 1. *Instalar aplicación servidora de túnel*: La mayoría de las herramientas listadas, trabajan instalándose en la maquina víctima, permitiendo conectarse las veces que sean necesarias.

Paso 2. *Tunelizar Puertos abiertos y servicios*: Junto con el listado de los puertos descubiertos, se intenta establecer un acceso end to end para conseguir una vía de intercambio de archivos y/o comandos.

La mayoría de las herramientas listadas, trabajan instalándose en la maquina víctima y pueden conectarse las veces que sean necesarias.

Herramientas recomendadas:

- PowerSploit
- Sbd
- Weevely
- http-tunnel
- dns2tcp

Paso 3. *Generar Informe*: Ante el caso necesario de poner en práctica esta etapa, se elaborará el Informe de Accesos Mantenidos (IAM)

6.2.5. Aplicación de la etapa de Eliminación de Pruebas

La etapa de borrado de pruebas tiene como objetivo el ocultamiento de actos maliciosos, sobrescribir registros y logs de sistema y aplicaciones y la elaboración del informe final. Para llevar adelante este proceso se cuenta con todos los datos obtenidos en los productos anteriores y se tendrá el Informe de Pruebas Eliminadas (IPE) como producto de salida.

PRODUCTOS DE ENTRADA para la etapa de Eliminación de pruebas

- Plan Integral de Testeo por Hacking Ético

- Informe de Arquitectura de red
- Informe de Escaneo de Sistema
- Informe de Accesos Ganados
- Informe de Accesos Mantenidos

PRODUCTOS DE SALIDA para la etapa de Eliminación de pruebas

Paso 1. *Revisión de mantenimiento de logs:* Manteniendo el acceso ganado, se buscaran eliminar todos los registros de actividades realizadas. Aquí como en las etapas anteriores, a diferencia del proceso de hacking real, el hackeo ético, debe probar que no todos los logs pueden ser eliminados.

- Configuración de logs según SO
- Contar con varios logs servers

Herramientas recomendadas:

- Elogger
- Logsign
- Log360

Paso 2. *Ejecución de herramientas forenses:* En esta instancia del proceso, se busca contraprobar que es posible ejecutar herramientas forenses y recuperar huellas de lo sucedido en los servidores.

Herramientas recomendadas:

- Foremost
- Log2timeline

Cumplimentando estos pasos, obtendremos el producto de salida de esta etapa llamado Informe de Pruebas Eliminadas (IPE).

6.2.6. Obtención documento "Informe final de Testeo por Hacking Ético (IFTHE)"

La aplicación de la todas las etapas que conforman la 1° fase de este proceso y compilando todos los productos obtenidos, se obtiene el documento final de la fase, el cual contiene los casos de prueba positivos y negativos, siendo este documento el que puede enviar el desarrollo a ser modificado o el que indique que es un software apto para ser implementado en producción.

6.3. APLICACIÓN DE LAS ACTIVIDADES DE LA FASE DE MANTENIMIENTO

En esta sección se aplicará al caso de validación corriente las etapas correspondientes a la fase de Mantenimiento: Control de Vulnerabilidades (sección 6.3.1), Determinación de Criterios (sección 6.3.2) y Verificación y Validación (sección 6.3.3). Obteniendo como producto final de la fase el Informe General de Riesgos, Validaciones y Verificaciones (IGRVV) (sección 6.3.4).

6.3.1. Aplicación de la etapa de Control de Vulnerabilidades

La aplicación de la esta etapa sobre el producto mismo en el ambiente de producción, permite listar, si existen, vulnerabilidades nuevas y determinar su grado de aplicabilidad al sistema. Para llevar a cabo este proceso se cuenta con el Informe obtenido en la fase de Ejecución y el Listado de nuevas vulnerabilidades, obteniendo el Informe de Vulnerabilidades Criticas (IVC) como producto de salida.

PRODUCTOS DE ENTRADA para el Control de Vulnerabilidades

- Informe Final de Testeo por Hacking Ético
- Listado de vulnerabilidades actualizado

PRODUCTO DE SALIDA para el Control de Vulnerabilidades

Paso 1. *Revisar vulnerabilidades:* Con esta etapa se busca revisar el cambio en las vulnerabilidades críticas o en ataques más frecuentes. Se listarán las amenazas mas criticas dadas por la fundación OWASP, la cual al momento de escribir esta tesis,

confecciona un documento de alto nivel que se centra sobre las vulnerabilidades más críticas de las aplicaciones web y móviles.

Paso 2. *Indicar vulnerabilidades propicias:* Teniendo listadas las amenazas mas criticas y recurrentes, procederemos a marcar cuales de estas pueden atacar directamente a nuestro sistema. Obteniendo de esta forma un ranking de vulnerabilidades propicias. Si en este documento no aparecen nuevas vulnerabilidades o si son las mismas a las que ya el sistema fue sometido a pruebas, la etapa de mantenimiento no debería continuar.

1. Listado de vulnerabilidades Web

Actualización	OWASP_Top_10-2017

Titulo	Sistema Propicio
Inyección	Si
Pérdida de Autenticación y Gestión de Sesiones	No
Cross-Site Scripting (XSS)	Si
Rotura de control de acceso	No
Security Misconfiguration	No
Sensitive Data Exposure	No
InsufficientAttackProtection	Si
Cross-Site Request Forgery (CSRF)	No
Using Components with Known Vulnerabilities	No
UnderprotectedAPIs	No

Tabla 48. Tabla de vulnerabilidades web del producto informe de vulnerabilidades

2. Listado de vulnerabilidades Mobile

Actualización	Mobile_Top_10-2016

Titulo	Sistema Propicio
Uso inadecuado de la plataforma	No
Almacenamiento de datos inseguros	No
Comunicación insegura	No
Autenticación no segura	No
Criptografía insuficiente	No
Autorización insegura	No
Calidad del código del cliente	No
Manipulación del código	No
Ingeniería inversa	No
Funcionalidad Extraña	No

Tabla 49. Tabla de vulnerabilidades mobile del producto informe de vulnerabilidades

Cumpliendo con estos pasos, obtendremos el producto de salida de la etapa llamado Informe de Vulnerabilidades Criticas (IVC).

6.3.2. Aplicación de la etapa de Determinación de Criterios

La aplicación de la etapa de determinación de criterios, ayuda a determinar grado de criticidad, establecer criterios de aprobación y rechazo y obtener un ranking de vulnerabilidades más riesgosas. Para llevar a cabo este proceso se cuenta con el Informe de vulnerabilidades críticas como producto de entrada, contando con el Listado de Aprobación y Rechazo (LAR) como producto de salida.

PRODUCTOS DE ENTRADA para la Determinación de Criterios

- Informe de Vulnerabilidades criticas

PRODUCTO DE SALIDA para la Determinación de Criterios

Paso 1. *Analizar nivel de criticidad de amenazas:* Junto con el documento generado de la etapa de Análisis de Vulnerabilidades, se deberá indagar sobre cada una de las amenazas propicias y asignarle un valor de criticidad. Esta valoración es de 1 a 5, siendo 1 Baja criticidad y 5 Alta Criticidad.

Paso 2. *Analizar nivel de aplicabilidad de amenazas:* De la misma forma que se evalúa que tan crítica puede ser una vulnerabilidad para el tipo de sistema a analizar, se identifica el nivel de aplicabilidad, diciéndonos que tanto aplica esa amenaza al sistema puesto a prueba. Esta valoración es de 1 a 5, siendo 1 Baja aplicabilidad y 5 Alta Aplicabilidad.

Estos dos últimos pasos se reflejan en el documento de salida de la siguiente forma:

1. Listado de Amenazas Web

Actualización	OWASP Top 10 - 2017

El nivel de criticidad y aplicabilidad se valora entre 1 y 5.

Titulo	Criticidad	Aplicabilidad
Inyección	4	4
Pérdida de Autenticación y Gestión de Sesiones	-	-
Cross-Site Scripting (XSS)	3	2
Rotura de control de acceso	-	-
Security Misconfiguration	-	-
Sensitive Data Exposure	-	-
InsufficientAttackProtection	3	2
Cross-Site Request Forgery	-	-

(CSRF)		
Using Components with Known Vulnerabilities	-	-
UnderprotectedAPIs	-	-

Tabla 50. Tabla de listado de amenazas web del producto listado de criterios de aprobación y rechazo

2. Listado de Amenazas Mobile

Actualización	Mobile Top 10 2016-Top 10

Titulo	Criticidad	Aplicabilidad
Uso inadecuado de la plataforma	-	-
Almacenamiento de datos inseguros	-	-
Comunicación insegura	-	-
Autenticación no segura	-	-
Criptografía insuficiente	-	-
Autorización insegura	-	-
Calidad del código del cliente	-	-
Manipulación del código	-	-
Ingeniería inversa	-	-
Funcionalidad Extraña	-	-

Tabla 51. Tabla de listado de amenazas web del producto listado de criterios de aprobación y rechazo

Paso 3. *Priorizar amenazas críticas:* Todas las amenazas valoradas en los pasos anteriores se deben colocar en prioridad para que los profesionales sepan cómo y qué puntos encarar con mayor urgencia. Esto se realiza colocando en formato de ranking las amenazas evaluadas en los puntos anteriores.

3. Prioridades de amenazas Web

Actualización	OWASP Top 10 - 2017

Prioridad de criticidad	Titulo	Criticidad	Aplicabilidad
1	Inyección	4	4
	Pérdida de Autenticación y Gestión de Sesiones	-	-
2	Cross-Site Scripting (XSS)	3	2
	Rotura de control de acceso	-	-
	Security Misconfiguration	-	-
	Sensitive Data Exposure	-	-
3	InsufficientAttackProtection	3	3
	Cross-Site Request Forgery (CSRF)kl	-	-
	Using Components with Known Vulnerabilities	-	-
	UnderprotectedAPIs	-	-

Tabla 52. Tabla de prioridades de amenazas web del producto listado de criterios de aprobación y rechazo

4. Prioridades de amenazas Mobile

Actualización	Mobile Top 10 2016-Top 10

Prioridad de criticidad	Titulo	Criticidad	Aplicabilidad
-	Uso inadecuado de la plataforma	-	-
-	Almacenamiento de datos inseguros	-	-
-	Comunicación insegura	-	-
-	Autenticación no segura	-	-
-	Criptografía insuficiente	-	-
-	Autorización insegura	-	-
-	Calidad del código del cliente	-	-
-	Manipulación del código	-	-
-	Ingeniería inversa	-	-
-	Funcionalidad Extraña	-	-

Tabla 53. Tabla de prioridades de amenazas mobile del producto listado de criterios de aprobación y rechazo

Paso 4. *Listar criterios de aprobación y rechazo:* Para poder aprobar o rechazar la prueba de un caso es necesario contar con criterios. Estos quedan registrados y pueden reutilizarse en aplicaciones similares.

5. Listado de Criterios de aprobación

Se aprobará al menos que…

Criterio	Aprobación
Comentarios de humanos	Los comentarios deben restringirse a ingresos hechos por humanos y no por trolls o automatismos
Firewall	El firewall no expondrá servicios sensibles y/o direcciones internas
Casos de prueba	Los casos de prueba sean satisfactorios en su totalidad.

Tabla 54. Tabla de listado de criterios de aprobación del producto listado de criterios de aprobación y rechazo

6. Listado de Criterios de Rechazo

Se rechazará cuando…

Criterio	Rechazo
Inyección SQL	Si ante una sentencia SQL se emite algún indicio del motor de BBDD
Puertos abiertos	Si hay puertos abiertos que no pertenezcan a la aplicación o funcionamiento propio del servidor
Escáner de vulnerabilidades	Si al menos hay un riesgo de tipo grave.

Tabla 55. Tabla de listado de criterios de rechazo del producto listado de criterios de aprobación y rechazo

Cumplimentando estos pasos, obtendremos el producto de salida de esta etapa llamado Listado de Criterios de Aprobación y Rechazo (LCAR).

6.3.3. Aplicación de la etapa de Verificación y Validación

La aplicación de la etapa de verificación y validación, permite ejecutar pruebas de verificación, comparar criterios establecidos, determinar criticidad del riesgo y establecer acciones a tomar. Para llevar a cabo este proceso se cuenta con el documento generado en la etapa predecesora, el listado de aprobación y rechazo, junto con el informe de vulnerabilidades criticas, obteniendo el Informe de Verificación y Validación (IVV) como productos de salida.

PRODUCTOS DE ENTRADA para la Verificación y Validación

- Informe de vulnerabilidades criticas
- Listado de Criterios de aprobación y rechazo

PRODUCTOS DE SALIDA para el Verificación y Validación

Paso 1. *Ejecutar pruebas de verificación:* Ante la actualización de vulnerabilidades, puede ser necesario re ejecutar pruebas para verificar nuevamente Si es necesario realizar nuevas pruebas sobre la aplicación, en este paso se plantean.
Se toma como basamento la etapa de planeamiento de pruebas para esto.

Paso 2. *Validar los resultados:* Luego de la ejecución de las pruebas necesarias, se validan respecto a los criterios estipulados.

Paso 3. *Acciones a tomar:* Dentro de este punto, se establecen que acciones se tornarán acerca de los riesgos asociados identificados.

Cumplimentando estos pasos, obtendremos el último producto de salida de esta etapa llamado Informe de Verificación y Validación (IVV).

6.3.4. Obtención documento “Informe General de Riesgos, Validaciones y Verificaciones (IGRVV)”

La aplicación de la todas las etapas que conforman la 1° fase de este proceso y compilando todos los productos obtenidos, se obtiene el documento final de la fase.

7. CONCLUSIONES

En este Capítulo se presentan las aportaciones de este trabajo (sección 7.1) y se destacan las futuras líneas de investigación que se consideran de interés en base al problema abierto que se presenta en este trabajo de tesis (sección 7.2).

7.1. APORTACIONES DE LA TESIS

La presente Tesis propone un modelo de proceso de hacking ético para la evaluación de vulnerabilidades dentro del procedimiento mismo de Testeo de un sistema. Este modelo se divide en tres fases, cada una de las cuales se organiza en etapas donde se propone una serie de actividades. Estas actividades, etapas y fases consumen ciertos insumos para generar una serie de productos que serán los informes y reportes que indicarán las vulnerabilidades y recomendaciones a seguir en cada caso.

El proceso se divide en las siguientes tres fases:

- Una primera fase de Planificación, cuyo objetivo se centra en la documentación, modelado, ordenamiento y planeamiento de las pruebas a las que se someterá el software desarrollado.
- Una segunda fase de Ejecución, donde el propósito consiste en la prueba ordenada y fundamentada con herramientas específicas buscando lograr obtener un software de calidad y libre de vulnerabilidades críticas.
- Y por último una tercera fase de Mantenimiento, el cual tiene como objetivo someter, periódicamente, al software desarrollado e implementado a pruebas de vulnerabilidades.

Este proceso transcurre dentro del procedimiento mismo de Testeo de un sistema, siendo aplicable a cualquier modelo de desarrollo de software. Teniendo como punto de partida los requisitos y las funcionalidades desarrolladas y probadas en su totalidad. Proporcionando como salida un informe final de testeo de hacking ético (IFTHE), el cual resumirá los riesgos, criticidades, casos erróneos y consejos a seguir para el software puesto a prueba.

Es importante resaltar, que si bien se requiere el uso de herramientas de análisis y penetración, las mismas no están atadas al proceso, siendo necesario seleccionar las herramientas de mayor auge en el momento de llevar a cabo el proceso de hacking ético.

Durante las pruebas realizadas se pudo comprobar la utilidad de planificar las tareas de testeo de hacking ético, al tener una guía de actividades y consulta de herramientas para las tareas de testeo, recolección de información, análisis y toma de medidas correctivas.

Algunas de estas tareas pueden llegar a pasarse por alto de no contar con una planificación adecuada, junto a un conjunto de medidas a tener en cuenta y herramientas para utilizar. Esta es la principal ventaja de contar con un modelo de proceso que sirva de guía para la planificación y ejecución de las actividades a desarrollar.

Si bien se han realizado solamente dos casos de validación, la utilización de las métricas de vulnerabilidad ofrece una visión general del estado inicial del sistema al ser desarrollado, en cuanto a vulnerabilidades se trate.

7.2. FUTURAS LÍNEAS DE INVESTIGACIÓN

Para continuar con el trabajo presentado en ésta Tesis, se recomienda continuar aplicando el modelo a proyectos de software para generar un mayor volumen de información. De esta manera puede, eventualmente, refinarse el modelo propuesto a través del análisis de estos resultados.

Particularmente, el refinamiento debería centrarse en dos puntos:

1) El desarrollo de métricas adicionales.
2) El desarrollo de índices.

Un indicador es una métrica o combinación de métricas que proporcionan una visión profunda del proceso, proyecto o producto software. Un indicador proporciona una visión profunda que permite al gestor del proyecto realizar los ajustes requeridos a tiempo para conseguir una mayor eficiencia y lograr mejores resultados [Pressman; 2005].

El empleo de estas métricas y análisis de más proyectos de desarrollo de software siguiendo el presente modelo de proceso para incorporar el hacking ético durante el testeo de software, harán posible el desarrollo de indicadores sobre estas métricas para, incluso, poder llegar a realizar una clasificación de sistemas de acuerdo a su grado de vulnerabilidad necesario y el que posee realmente. Así, un sistema de home banking tendrá un indicador de las métricas de vulnerabilidad mucho más ajustado que el de un blog hobista, por indicar un ejemplo.

8. REFERENCIAS BIBLIOGRAFICAS

Agile Manifesto, (2014). Disponible http://www.agilemanifesto.org/.

ANSI/IEEE, (2007). Draft IEEE Standard for software and system test documentation. ANSI/IEEE Std P829-2007.

Areitio J, (2008). "Seguridad de la información. Redes, informática y sistemas de información" Editorial Paraninfo, ISBN: 8497325028.

Argimón J, (2004). Métodos de Investigación Clínica y Epidemiológica. Elsevier España, S.A. ISBN 9788481747096.

Bach J, (2001). "What is Exploratory Testing? And How it differs from Scripted Testing" StickyMinds.

Basili, V. (1993). The Experimental Paradigm in Software Engineering. En Experimental Software Engineering Issues: Critical Assessment and Future Directions (Ed. Rombach, H., Basili, V., Selby, R.). Lecture Notes in Computer Science, Vol. 706. ISBN 978-3-540-57092-9.

Beizer B.(1990) "Software testing techniques (2nd ed.)", ISBN:0-442-20672-0, Van Nostrand Reinhold Co.

Benchimol D.(2011) Hacking desde cero. 1° edición - Buenos Aires: Fox Andina. ISBN 978-987-1773-03-9

Bittner K. & Spence I.(2002), Use-Case Modeling, Addison-Wesley Professional; 1° edición, ISBN-10: 0201709139

Boehm B. (1988) "A Spiral Model of Software Development and Enhancement", IEEE Computer, IEEE, 21(5):61-72.

Boehm W. (1979)"Software Engineering; R&D Trends and Defense Needs". In R. Wegner, ed. Research. Directions in Software Technology. Cambridge, MA:MIT Press.

Cano J.(2011), La Gerencia de la Seguridad de la Información: Evolución y Retos Emergentes. https://www.isaca.org/Journal/archives/2011/Volume-5/Pages/JOnline-La-Gerencia-de-la-Seguridad-de-la-Informacion-Evolucion-y-Retos-Emergentes.aspx Página Válida a 10/2018.

Creswell, J. (2002). Educational Research: Planning, Conducting, and Evaluating Quantitative and Qualitative Research. Prentice Hall. ISBN 10: 01-3613-550-1.

Dalmau J & Gigou J.(1997), ESA Bulletin Nr. 89, http://www.esa.int/esapub/bulletin/bullet89/dalma89.htm Página Válida a 10/2018.

De Miguel, María del Rosario(2007). Deontología y aspectos legales de la informática: cuestiones éticas, jurídicas y técnicas básicas. Valencia : Ed. UPV ISBN 978-84-8363-112-6

Dijkstra E.(1972), The humble programmer. ACM 15, 10: 859–866.

Dupuis R., Bourque P., Abran A., Moore J., and Tripp L.(2001). The SWEBOK Project: Guide to the software engineering body of knowledg. Stone Man Trial Version 1.00. http://www.swebok.org/ Página Válida a 10/2018.

Evans, Bob (2001). The Sorry State of Software. InformationWeek 112.

FAA https://s3.amazonaws.com/public-inspection.federalregister.gov/2015-10066.pdf Página Válida a 10/2018.

García Martínez, R., Britos, P. (2004). Ingeniería de Sistemas Expertos. Editorial Nueva Librería. ISBN 987-1104-15-4.

Ghezzi C, Jazayeri M., and Mandrioli D.(1991). Fundamentals of sofware engineering. Prentice Hall, Upper Saddle River, New Jersey.

Gimenez, V.(2011), Hacking y ciberdelito, Universidad Politécnica de Valencia.

Gregory, P. Computer Viruses for Dummies. Indianapolis, Indiana: Wiley Publishing, Inc.ISBN: 0-7645-7418-3

IEEE, (1997). IEEE Standard for Developing Software Life Cycle Processes. IEEE Std 1074-1997 (Revision of IEEE Std 1074-1995; Replaces IEEE Std 1074.1-1995)

IEEE, (1990) Standard Glossary of Software Engineering Terminology, IEEE STD 610-1990.

IEEE Computer Society (2004), Guide to the Software Engineering Body of Knowledge SWEBOK, 2004 version. http://www.swebok.org. Página Válida a 10/2018.

ISO 27035:2011 (2010). Information technology – Security techniques – Information security incident management. [Online]. http://www.iso.org/iso/catalogue_detail?csnumber=44379. Página Válida a 10/2018.

ISO 9000:2000, Sistemas de gestión de calidad – Conceptos y vocabulario. http://iso25000.com/index.php/normas-iso-25000/iso-25010 Página Válida a 10/2018

ISTQB Glosary - ISTQB Foundation – http://glossary.istqb.org/search/testing Página Válida a 10/2018.

ISQTB, (2005), Certified Tester Foundation Level Syllabus. http://www.istqb.org/fileadmin/media/SyllabusFoundation.pdf Página Válida a 10/2018.

Jacobson, I., Ng, P. W., McMah, P. E., & Jaramillo, C. M. Z. (2013). La esencia de la ingeniería de software: El núcleo de Semat. Revista Latinoamericana de Ingeniería de Software, 1(3), 71-78.

Jara H. y Pacheco F.(2012), Ethical Hacking 2.0, ISBN 978-987-1857-63-0.

Kaner C., Falk J., Nguyen H. (1999), "Testing Computer Software, 2nd Edition", ISBN: 0471358460, Wiley.

Leiva Mundaca, I; Villalobos Abarca, M. (2015) Método ágil híbrido para desarrollar software en dispositivos móviles. Ingeniare. Revista Chilena de Ingeniería, vol. 23, núm. 3, pp. 473-488 Universidad de Tarapacá, Chile

Levy, S (2001). Hackers. Capítulo: La ética del hacker. Ed. Penguin.

Mieres J. (2010). Certified ethical hacker review guide, Buenos Aires.

Mieres J.(2009), ESET Buenas prácticas en seguridad informática Analista de Seguridad de ESET.

Mifsud E.(2012.)Introducción a la seguridad informática - Políticas de seguridad.

Myers G.(2004). The art of software testing. Segunda edición. John Wiley & Sons.

Oktaba, H., Garcia, F., Piattini, M., Ruiz, F., Pino, F., Alquicira, C. (2007). Software Process Improvement: The Competisoft Project. IEEE Computer, 40(10): 21-28. ISSN 0018-9162.

OWASP Top 10 - 2013 (2013). [Online]. http://www.owasp.org. Página Válida a 10/2018.

Palmer, Charles (2001). Ethical hacking, IBM Systems Journal, Vol. 40, N°3

Piattini, M. (2000) Análisis y diseño detallado de Aplicaciones Informáticas de Gestión. Alfaomega.

Pressman, R,(2005) Ingeniería de Software 6ª Ed., McGraw Hill.

Raymond E (1991). The New Hacker's Dictionary, MIT Press, Cambridge, MA

Riveros, H. y Rosas, L. (1985). El Método Científico Aplicado a las Ciencias Experimentales. Editorial Trillas. México. ISBN 96-8243-893-4.

Royce, W.(1970), Managing the development of large software systems: concepts and technique, IEEE Westcon.

Rumbaugh, J., Jacobson, I., Booch, G. (1999). The Unified Modeling Language, Reference Manual. Addison Wesley, ISBN-10: 02-0130-998-X.

Sabato J, Mackenzie M. (1982). La Producción de Tecnología: Autónoma o Transnacional. Instituto Latinoamericano de Estudios Transnacionales - Technology & Engineering. ISBN 9789684293489.

Santos L.(2014) , Guia para la evaluación de seguridad en un sistema. Universidad de Pamplona

Scambray J., Schema M. (2002). Hacking Exposed Web Applications. McGraw-Hil/Osborne

Schneier, Bruce. (2000). Secrets & Lies: Digital Security in a Networked World. John Wiley & Sons.

Schaefer H., Linz T., Spillner A (2014).”Software testing foundation”. 4° edición. Rocky Nook.

Sheoran, Pankaj & Singh, Sukhwinder (2014). Applications of Ethical Hacking, International Journal of Enhanced Research in Science Technology & Engineering, ISSN: 2319-7463 Vol. 3 Issue 5, pp: (112-114), Impact Factor: 1.252, www.erpublications.com Page | 112 Página Válida a 10/2018.

Sommerville I. (2005), “Ingeniería del Software”, Editorial:
Pearson,7ª edición).

Tori C. (2008). Hacking Ético (1ra Ed). Buenos Aires: Mastroianni.

Villalón Huerta A.(2002), SEGURIDAD EN UNIX Y REDES Version 2.1.
www.rediris.es/cert/doc/unixsec/unixsec.pdf Página Válida a 10/2018.

Zimmerman, Christine. (2001). Race to Deploy May Magnify Software Bugs. InternetWeek 13.

9. PUBLICACIONES REALIZADAS

En este capítulo se mencionan las publicaciones donde fue presentado el tema abordado por esta tesis.

9.1. WICC 2018

WICC es el Workshop de Investigadores en Ciencias de la Computación (WICC), el mismo es organizado, a partir de 1999, por la Red de Universidades Nacionales con Carreras de Informática (RedUNCI) con el propósito de generar un foro para el intercambio de ideas entre investigadores en Ciencias de la Computación, de modo de fomentar la vinculación y potenciar el desarrollo coordinado de actividades de Investigación, Desarrollo e Innovación entre los mismos.

El artículo fue aprobado con el número 11122 - "INCLUSIÓN DE HACKING ÉTICO EN EL PROCESO DE TESTING DE SOFTWARE" perteneciente al Área "Ingeniería de Software"

El mismo fue expuesto los días 26 y 27 de Abril de 2018 en la Ciudad de Corrientes
Y se encuentra publicado en el libro de actas del WICC
RedUNCI - UNNE - ISBN 978-987-3619-27-4; Pagina 690.
http://wicc2018.unne.edu.ar/wicc2018librodeactas.pdf (pagina activa al día 31/8)

9.2. CACIC 2018

El Congreso Argentino de Ciencias de la Computación (CACIC) es organizado por la Red de Universidades Nacionales con carreras en Informática (RedUNCI). CACIC reúne desde 1995 a investigadores, docentes, profesionales y alumnos de grado y postgrado vinculados con la disciplina Informática. El Congreso cubre diferentes áreas a través de la organización de Workshops, coordinados por expertos en los temas del área. En estos Workshops se presentan trabajos científicos evaluados por investigadores del país y del exterior. El banco de evaluadores del Congreso es público.

El artículo se encuentra en proceso de aprobación con el número 11606 - "METODO DE INCLUSIÓN DE HACKING ÉTICO EN EL PROCESO DE TESTING DE SOFTWARE" perteneciente a la actividad "Workshop de Ingeniería de Software"

El mismo será llevado a cabo entre el 8 y el 12 de octubre en la ciudad de Tandil.

10. ANEXOS

En este capítulo se presentan los anexos correspondientes a los dos casos de validación ejecutados en el capítulo 5 de esta tesis. En la sección 9.1 se adjuntan todos los documentos correspondientes al caso analizado en la sección 5.1 y en la sección 9.2 la documentación obtenida en el caso de validación 5.2.

10.1. ANEXOS CASO DE VALIDACION 1

En esta sección se adjuntan los documentos correspondientes al caso analizado en el capítulo 5.

Los documentos son los siguientes:

- Informe de Dominio
- Informe de Vulnerabilidades
- Informe de Amenazas
- Prioridades de criticidad
- Casos de Prueba
- Calendario de pruebas y esfuerzo
- Planilla de Diagrama Gantt
- Informe de criterios
- Informe de Arquitectura de red
- Informe de Escaneo de sistema
- Informe de Vulnerabilidades Criticas
- Listado de criterios de aprobación y rechazo

10.2. ANEXOS CASO DE VALIDACION 2

En esta sección se adjuntan los documentos correspondientes al caso analizado en el capítulo 6.

Los productos son los siguientes:

- Informe de Dominio
- Informe de Vulnerabilidades
- Informe de Amenazas
- Prioridades de criticidad
- Casos de Prueba
- Calendario de pruebas y esfuerzo
- Planilla de Diagrama Gantt
- Informe de criterios
- Informe de Arquitectura de red
- Informe de Escaneo de sistema
- Informe de Vulnerabilidades Criticas
- Listado de criterios de aprobación y rechazo