FACULTAD DE INGENIERÍA Y CIENCIAS APLICADAS

ANÁLISIS DE VULNERABILIDADES EN EL PORTAL WEB DE UNA INSTITUCIÓN DE EDUCACIÓN SUPERIOR DEL ECUADOR MEDIANTE HACKING ÉTICO

Autores

John Paúl Añazco Bedón
Bryan Fernando Ortiz Rodriguez

Año
2018

FACULTAD DE INGENIERÍA Y CIENCIAS APLICADAS

ANÁLISIS DE VULNERABILIDADES EN EL PORTAL WEB DE UNA INSTITUCIÓN DE EDUCACIÓN SUPERIOR DEL ECUADOR MEDIANTE HACKING ÉTICO

Trabajo de Titulación presentado en conformidad con los requisitos establecidos para optar por el título de Ingeniero en Sistemas de Computación e informática

Profesor Guía

Mgt. Eddy Mauricio Armas Pallasco

Autores

John Paúl Añazco Bedón

Bryan Fernando Ortiz Rodriguez

Año

2018

DECLARACIÓN DEL PROFESOR GUÍA

"Declaro haber dirigido el trabajo, “Análisis de Vulnerabilidades en el Portal Web de una Institución de Educación Superior del Ecuador Mediante Hacking Ético” a través de reuniones periódicas con los estudiantes John Paul Añazco Bedón y Bryan Fernando Ortiz Rodriguez, en el semestre 2018-2, orientando sus conocimientos y competencias para un eficiente desarrollo del tema escogido y dando cumplimiento a todas las disposiciones vigentes que regulan los Trabajos de Titulación".

_____

Eddy Mauricio Armas Pallasco
Magister en Gerencia de Sistemas y TI
CC: 171171580-3

DECLARACIÓN DEL PROFESOR CORRECTOR

"Declaro haber revisado este trabajo, “Análisis de Vulnerabilidades en el Portal Web de una Institución de Educación Superior del Ecuador Mediante Hacking Ético”, de los estudiantes John Paul Añazco Bedón y Bryan Fernando Ortiz Rodriguez, en el semestre 2018-2, dando cumplimiento a todas las disposiciones vigentes que regulan los Trabajos de Titulación".

_____

Pedro Manuel Nogales Cobas

Máster en Ciencias

CC: 175676028-4

DECLARACIÓN DE AUTORÍA DE ESTUDIANTES

“Declaramos que este trabajo es original, de nuestra autoría, que se han citado las fuentes correspondientes y que en su ejecución se respetaron las disposiciones legales que protegen los derechos de autor vigentes.”

_____	_____
John Paul Añazco Bedón	Bryan Fernando Ortiz Rodriguez
CC: 172623544-1	CC: 172211569-6

AGRADECIMIENTO

Agradezco a mis padres por que han estado siempre apoyándome en todas las situaciones que he encontrado. También agradezco a Jessie, Daniel y Jonathan porque en gran medida, ayudaron en mi crecimiento personal y sin ellos tampoco hubiera llegado tan lejos o entendido más cosas. Agradezco a Juan José porque fue la persona que apoyo mi crecimiento laboral confiando en mi habilidad. Agradezco a Bryan por que sin él no hubiera podido concretar esta tesis.

John

AGRADECIMIENTO

A la suerte de encontrarme con vida y en la situación actual de poder compartir con mis seres queridos un día más, a mi primera maestra que recuerdo y que me presentó uno de los libros más completos y con ello cimentó las bases para ser curioso. Por último, a mis seres queridos que siempre estarán a mi lado, a pesar de la distancia.

Bryan Fernando

DEDICATORIA

A mi ma y pa por todo su esfuerzo vertido en mí y a mi mejor compañía Kapito.

John

DEDICATORIA

A quien corresponda, el leer la tesis y tratar de tomarse un tiempo en comprender la dedicación y el esfuerzo que John y mi persona pusimos en este trabajo. No siempre se culmina un proyecto, pero por suerte tuve la oportunidad de hacerlo junto a ellos.

Bryan Fernando

RESUMEN

Los portales web de las Instituciones de Educación Superior en Ecuador se enfrentan a un aumento exponencial de los ataques informáticos, es necesario contar con herramientas y metodologías adecuadas para hacer frente a los riesgos informáticos. Utilizando técnicas de *Ethical Hacking*, se analizan y detectan las vulnerabilidades de los portales web, y se mide y cuantifica el riesgo para determinar el impacto. Por último, se propone un plan general de mitigación de ataques informáticos.

ABSTRACT

The web portals of Higher Education Institutions in Ecuador are facing an exponential increase in computer attacks, it is necessary to have the right tools and methodologies to face the computer risks. Using Ethical Hacking techniques, web portal vulnerabilities are analyzed and detected, and risk is measured and quantified to determine impact. Finally, a general plan for mitigating computer attacks is proposed.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

1. Introducción

1.1. Antecedentes

La ciberseguridad dentro de las empresas corporativas e institucionales ha ido evolucionando debido al aumento exponencial de ciberdelitos, tales como; fraudes informáticos, manipulación de datos y daños de sistemas computacionales (Borghello, 2001). Todo ello conlleva a que se sigan una serie de procedimientos para mitigar, controlar o aceptar el riesgo inherente de usar sistemas informáticos, tales como herramientas de auditoría, pruebas de *PenTesting* y Creación de Equipos de Respuesta ante Incidentes Computacionales "*CSIRT*" (ISACA, 2015 y Freire y Lennin, 2017).

Por todo ello el cibercrimen es sin duda una preocupación creciente y las empresas están buscando formas viables y ágiles de hacer frente a diversas vulnerabilidades y amenazas. Para lo cual, procesos como el Hacking Ético ayudan a las empresas a detectar amenazas tanto internas como externas en diferentes escenarios.

Por otra parte, la creación de Equipos de Respuesta ante Incidentes Computacionales, ayudan a las empresas a cuidar activos del negocio ante riesgos y vulnerabilidades en los sistemas de información, respondiendo oportuna y correctamente ante eventualidades provocadas por atacantes, infecciones de virus informáticos y robo de información (Freire y Lennin, 2017).

En la actualidad y en América Latina el porcentaje anual de ataques informáticos es de un 30% según datos expuestos en la 7ma Cumbre Latinoamericana de Analistas de Seguridad (Kaspersky Lab., 2017) dicho informe estadístico muestra el nivel de seguridad de las aplicaciones infectadas en América latina, demostrando el estado urgente de realizar análisis de seguridad en aplicaciones web.

Según el *ITRC Data Breach Reports* el porcentaje de ataques por tipo de instituciones estuvo distribuido en el 2017 de la siguiente manera; A empresas

privadas un 55.1%, instituciones de salud un 23.7%, Instituciones de Educación un 8.0% e Instituciones gubernamentales un 4.7% (Identity Theft Resource Center, 2017). Lo cual indica que las Instituciones de Educación son un grupo vulnerable para ataques informáticos.

La situación actual de las páginas web de las Instituciones de Educación Superior del Ecuador al público en general es de vital importancia para el negocio, debido a que se concentran en la web del portal alrededor del 68.50% del público interesado en acceder a información de matriculación y estudio en modalidades presencial, semipresencial y a distancia (Alexa Internet Inc., 2018).

A su vez la página web principal es una de las primeras herramientas que tienen las Instituciones de Educación Superior para atender al público y dar a conocer todos sus servicios y beneficios que ofrece a nivel nacional e internacional, es por lo que se vuelve indispensable el precautelar la integridad, seguridad y disponibilidad de su principal acceso a estudiantes nuevos y antiguos.

1.2. Alcance

Este tema de titulación tendrá como objetivo principal el evaluar vulnerabilidades en el portal web principal de una Institución de Educación Superior y recomendar acciones para la mitigación de ataques de hackeo considerando las generalidades de los portales web.

En este proceso se usa herramientas de penetración web abarcando entre ellas las referidas en el capítulo cuatro de la guía Abierta de Seguridad para las Aplicaciones Web (*Open Web Application Security Protocol*) u OWASP, específicamente las de Obtención de Información (*Information Gathering*), Pruebas de Validación de Información (*Data Validation Testing*) y Pruebas de Denegación de Servicio (*Testing for Denial of Service*) (The OWASP Foundation, 2014) cumpliendo con la metodología de Prueba de Penetración a Ciegas (Blind PenTesting).

Para la creación de un Plan de Mitigación de Ataques Informáticos se tendrá en cuenta la guía de NIST (*Computer Security Incident Handling Guide*", la cual abarca los principales pasos para la preparación, identificación, actuación e informe ante incidentes computacionales. (NIST, 2012).

A su vez, este proyecto no abarca la explotación de las vulnerabilidades encontradas ni la implementación de un *CSIRT*, esto compete a los administradores de los sistemas y las políticas aplicables al negocio. Para cumplir con el objetivo del trabajo de titulación se emplea lo aprendido y adquirido en las materias de Seguridad Informática y Auditoria informática.

1.3. Justificación

Una vulnerabilidad en la Institución de Educación Superior puede afectar a todos sus estudiantes, como resultado de ello se necesita que la infraestructura y los sistemas en los que interactúan estén seguros ante posibles ataques criminales.

Dichos ataques tienen como objetivo reconocer vulnerabilidades y amenazas sobre sistemas críticos, los mismos que son usados por hackers para alterar, obtener y vender información valiosa y sensible.

En la búsqueda de una mejor seguridad para la Institución y sus estudiantes se pone a disposición del público los resultados encontrados con las pruebas de Penetración realizadas en las páginas webs, verificando de esta manera el nivel de seguridad con la que se manejan estos servicios.

1.4. Objetivo General

Diseñar una propuesta de mitigación de ataques informáticos para disminuir los riesgos relacionados con las vulnerabilidades en la infraestructura del portal web de una Institución de Educación Superior del Ecuador.

1.5. Objetivos Específicos

- Analizar el estado actual del portal web principal de la Institución de Educación Superior, utilizando la metodología OWASP para la Obtención de Información, Validación de Información y Denegación de Servicios a objetivos específicos dentro de la página web principal, mediante el uso de Ethical Hacking.
- Realizar un análisis de probabilidad-impacto de las vulnerabilidades encontradas en base a la metodología OWASP.
- Recomendar el curso de acciones que se debe tomar para mitigar cada vulnerabilidad encontrada mediante el uso de OWASP.

1.6. Metodología

Para el desarrollo del presente proyecto se usa la metodología deductiva basada en OWASP *Blind PenTesting* y las recomendaciones de NIST para la respuesta ante Incidentes Computacionales, identificando prioridades, activos, riesgos y amenazas. Por último, se presenta una propuesta de mitigación de ataques informáticos basados en las recomendaciones de NIST (Computer Security Incident Handling Guide) el cual provee de cuatro fases ante la ocurrencia de incidentes informáticos (NIST, 2012).

2. Marco Teórico

En el presente capítulo se describe todas las metodologías, herramientas y recomendaciones realizadas en el presente trabajo.

2.1. Seguridad Informática

La información es uno de los activos más importantes para las empresas por lo que se necesita de políticas de seguridad de la información, las mismas que deben estar instauradas en un Sistema de Gestión de la Seguridad de la Información “ISO/IEC 27000”. Se acuña el termino de seguridad informática a toda acción, método, medida y política para proteger la infraestructura informática y de telecomunicaciones de posibles amenazas, ataques, vulnerabilidades y riesgos que puedan ocurrir. (Gascó, Romero, Ramada, y Pérez, 2017, p.7).

2.1.1. Vulnerabilidades Informáticas

Una vulnerabilidad informática es un defecto o debilidad en el diseño, implementación, operación o gestión de un sistema que podría explotarse para comprometer los recursos del sistema. (The OWASP Fundation, 2018, p. 27).

2.1.2. Amenazas Informáticas

Según el libro, Seguridad Informática: “Una amenaza es cualquier entidad o circunstancia que atente contra el buen funcionamiento de un sistema informático” (Gascó et al., 2017, p.9). Entre las amenazas más comunes se tiene Intrusos como; Hackers, Crackers, Sniffers, Spammers entre otros y Códigos Maliciosos (*Malware*) como; Virus, Troyanos, Gusanos (*Worms*), Programas espía (*Spyware*), *Creepware,* entre otros.

a. Hackers y Hackeo

Los hackers son intrusos que se dedican a conocer los aspectos administrativos y técnicos del funcionamiento de los sistemas informáticos, sin pretensiones de hacer daño, por consecuente su actividad puede ser considerada como delito por comprometer datos sensibles y confidenciales de la empresa (Gómez, 2011, p. 196). La acción de vulnerar a un sistema informático sin la autorización o sin el consentimiento de la empresa es considerado como hackeo.

b. Hacking Ético

Según la ISECOM (*The Institute for Security and Open Methodologies*) en su manual digital OSSTMM 3 (*Open Source Security Testing Methodology Manual*) se aplica el termino Hacking Ético (*Ethical Hacking*) para referirse a una prueba de penetración en donde el objetivo atacado no conoce de antemano las pruebas a realizar, mientras que el atacante, con sus habilidades y conocimientos, trata de vulnerar el sistema. A este tipo de prueba se lo conoce con el nombre de auditoría ciega o *Blind PenTesting*. (ISECOM, 2010, p. 37).

2.1.3. Ataques Informáticos

Según La Biblia del Hacker, existen 4 categorías de ataques informáticos: Ataques de Monitorización, Ataques de Validación, Ataques de denegación de servicios y Ataques de modificación. (Pérez, Pérez, Matas, Picouto y Ramos, 2006). Estos pasan por una serie de fases; reconocimiento, donde el atacante recaba toda la información de la empresa, exploración, se revisa todos los posibles componentes que pueden fallar, obtención de acceso, se intenta explotar alguna vulnerabilidad, mantener el acceso, para futuros ataques y por último, borrar huellas, para evitar detecciones del sistema o del administrador. (Seguridad informática, 2017, p. 10).

a. Ataques de Monitorización

Estos tipos de ataques se ejecutan para observar a la víctima y su sistema, obteniendo de esta manera información muy valiosa, con el objetivo de enumerar y relacionar debilidades con formas de acceso al sistema. (Pérez et al., 2006, p. 71). Entre los más comunes se encuentran; *Shoulder Surfing*, *Decoy*, Escaneo TCP/SYN/FIN/fragmentado, *Eavesdropping*.

b. Ataques de Validación

El objetivo de estos ataques es suplantar al usuario, mediante la obtención de sus credenciales de acceso, suplantando su identidad en la máquina. (Pérez et al., 2006, p. 73). Los tipos de ataques más comunes son; suplantación de identidad, robo de IP, DNS, *IP Splicing*, *Hijacking*, creación de *backdoors* y *exploits*.

c. Ataques de Denegación de Servicios

Con este tipo de ataques lo que se pretende es desbordar los recursos del objetivo de tal forma que los servicios ofrecidos por el mismo sean interrumpidos (Pérez et al., 2006, p. 75). Los tipos de ataque más comunes son; *Flodding, SYN Flood, Connection Flood, LAND Attack, Broadcast Storm,* Desbordamiento NET, *supernuke*, *Teardown one y two*, spam.

d. Ataques de Modificación

Este tipo de ataques tienen como objetivo realizar modificaciones no autorizadas de datos y archivos de información en los sistemas objetivo (Pérez et al., 2006, p. 77). Los ataques más comunes son; *Tampering*, Borrado de Registros, Ataque de *Java Applets*, *JavaScript* o *Visual Script*, Ataques *ActiveX*.

2.1.4. Riesgos Informáticos

El riesgo es una medida de probabilidad de que una amenaza se materialice, como el riesgo no se puede evitar se tienen medidas de coste/riesgos asumibles por una organización. Para realizar una clasificación correcta de los riesgos se necesita de una priorización o clasificación de activos y recursos de la empresa. (Gascó et al., 2017, p. 12).

Por lo tanto, es necesario un equipo de Gestión de Riesgos, el cual tiene como objetivo principal proteger a la organización y poder llevar a cabo su misión, no sólo centrado en los activos de TI. El proceso de gestión de riesgos no debe ser tratado principalmente como una función técnica llevada a cabo por expertos de Tecnologías de la Información, sino como una función de gestión esencial de una organización. (NIST, 2002).

En la Tabla 1 se presenta un ranking de las 10 amenazas más importantes de la Web proporcionada por OWASP, las cuales sirven como guía para las pruebas relevantes de OWASP.

Tabla 1.
Comparación Top 10 de Vulnerabilidades OWASP.

OWASP Top 10 2017
A1: 2017 – Inyección
A2: 2017 – Pérdida de Autenticación y Gestión de Sesiones
A3: 2017 – Exposición de Datos Sensibles
A4: 2017 – Entidad Externa de XML (XXE)
A5: 2017 – Pérdida de Control de Acceso
A6: 2017 – Secuencia de Comandos en Sitios Cruzados (XSS)
A7: 2017 – Secuencia de Comandos en Sitios Cruzados (XSS)
A8: 2017 – Deserialización Insegura
A9: 2017 – Uso de Componentes con Vulnerabilidades Conocidas
A10: 2017 – Registro y Monitoreo Insuficientes

Adaptado de (The OWASP Foundation, 2017).

2.1.5. Análisis de Amenazas y Vulnerabilidades para calificar los Riesgos

Para el análisis de Amenazas y Vulnerabilidades se toma en cuenta OWASP *Risk Rating Methodology.* La cual toma en cuenta Factores de Probabilidad y Factores de Impacto para determinar la gravedad del Riesgo *(*The OWASP Fundation, 2018). En la siguiente Figura se muestra los pasos seguidos para la calificación de Riesgo.

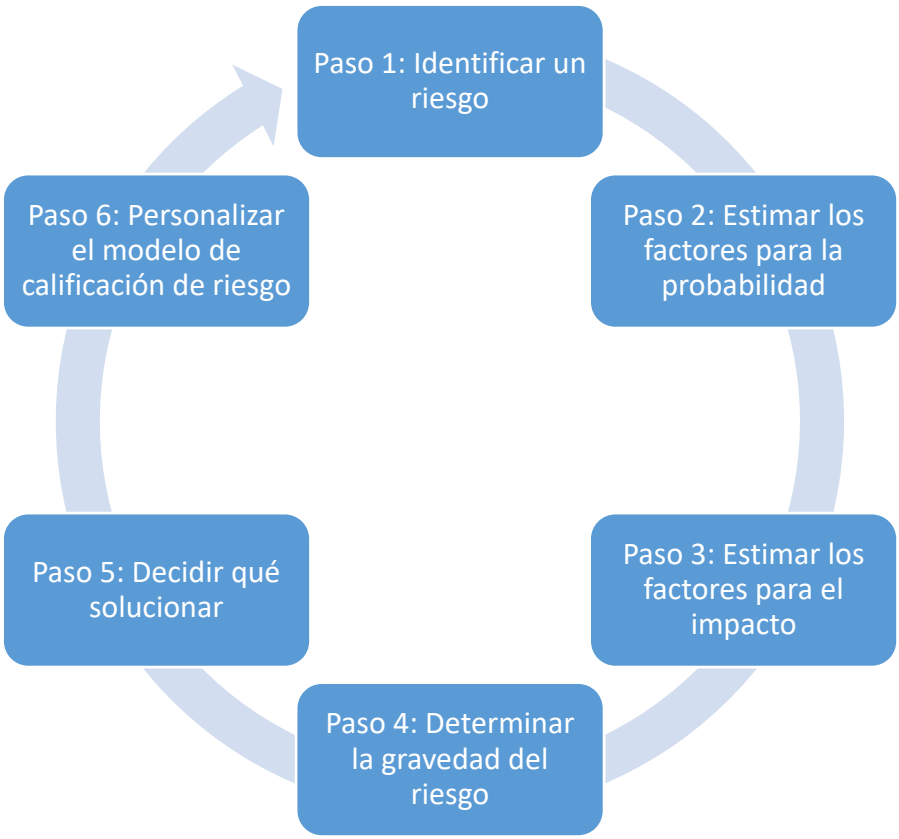

Figura 1. Diagrama de Calificación de Riesgo.
Adaptado de (The OWASP Fundation, 2018).

a. Estimación de factores de probabilidad

Los factores de probabilidad son aquellos que se deben juntar para que la vulnerabilidad sea ejecutada, en el modelo de Riesgos serán: Factores de Amenaza y Factores de Vulnerabilidad.

- **Factores de Amenaza**

Están relacionados con el agente de amenaza involucrado. El objetivo aquí es estimar la probabilidad de un ataque exitoso por parte de un grupo de agentes de amenaza. Se usa el peor de los casos posibles Tabla 2.

Tabla 2.
Factores de Amenaza.

Factor de Amenaza	Descripción	Puntuaciones
Nivel de Habilidad	Capacitación técnica del grupo	1. Sin Conocimientos Técnicos 3. Algunas Habilidades Técnicas 5. Usuario Avanzado de computación 6. Conocimiento de programación y redes 9. Habilidades de Penetración de la seguridad
Motivos	Motivaciones del grupo para encontrar y explotar la vulnerabilidad	1. Baja o ninguna recompensa 4. Posible Recompensa 9. Recompensa Alta
Oportunidad	Recursos necesarios y nivel de oportunidad que se requieren para que este grupo, encuentre y explote la vulnerabilidad	0. Acceso completo a la aplicación o recursos costosos requeridos 4. Acceso especial o recursos requeridos 7. Algunos Accesos o recursos requeridos 9. No requiere acceso o recursos
Tamaño	Tamaño del grupo	2. Desarrolladores, administradores de sistemas

Factor de Amenaza	Descripción	Puntuaciones
		4. usuarios de la intranet 5. Socios o *Partners* 6. Usuarios Autenticados 7. Usuarios anónimos de Internet

Adaptado de *(The OWASP Fundation, 2015)*.
Nota: Grupo, se refiere al Agente de Amenaza. Las puntuaciones son en base al modelo de *Risk Rating Template* de OWASP.

- **Factores de Vulnerabilidad**

Están relacionados con la vulnerabilidad en cuestión. El objetivo es estimar la probabilidad de que una vulnerabilidad en particular implicada sea descubierta y explotada Tabla 3:

Tabla *3.*
Factores de Vulnerabilidad basado en OWASP.

Factor de Vulnerabilidad	Descripción	Puntuaciones
Facilidad de Descubrimiento	Facilidad del grupo de para descubrir esta vulnerabilidad	1. Prácticamente imposible 3. Difícil 7. Fácil 9. Herramientas Automatizadas disponibles
Facilidad de Explotación	Facilidad del grupo para explotar la vulnerabilidad	1. Teórico 3. Difícil 5. Fácil 9. Herramientas automatizadas disponibles
Concienciación	Conocimiento del grupo sobre esta vulnerabilidad	1. Desconocido 4. Oculto 6. Obvio 9. Conocimiento Público

Detección de Intrusión	Probabilidad de que se detecte una intrusión	1. Detección activa en la aplicación 3. Registrado y revisado 8. Registrado sin revisión 9. No registrado

Adaptado de *(The OWASP Fundation, 2015)*.
Nota: Al referirse a este grupo, se refiere al agente de Amenaza. Las puntuaciones son tomadas en base al modelo de Risk Rating Template de OWASP.

b. Estimación de factores de impacto

Los factores de Impacto son aquellos a los que la vulnerabilidad afecta en menor o mayor medida, con respecto al negocio y a la aplicación estos son: Impactos Técnicos e Impacto al Negocio.

- **Impactos Técnicos**

Son aquellos factores que afectan a la tecnología o Infraestructura y están alineados con las áreas de seguridad tradicionales de interés para el negocio, como: confidencialidad, integridad, disponibilidad y responsabilidad. El objetivo es estimar la magnitud del impacto en el sistema si este fuera vulnerado. Tabla 4.

Tabla 4.
Impactos Técnicos.

Factor de Vulnerabilidad	Descripción	Puntuaciones
Pérdida de Confidencialidad	Información que se puede divulgar y cuan sensible es	2. Mínima Revelación de Datos No Sensibles 4. Divulgación Mínima de datos Críticos, divulgación extensa de datos no sensible 5. Amplia divulgación de datos críticos

Factor de Vulnerabilidad	Descripción	Puntuaciones
		9. Todos los datos revelados
Pérdida de Integridad	Daño sobre los datos y su afectación	1. Mínimo de datos ligeramente corruptos 3. Mínimo de datos seriamente corruptos 5. Datos extensos ligeramente corruptos 7. Datos Extensos y gravemente corruptos 9. Todos los datos están totalmente corruptos
Pérdida de Disponibilidad	Pérdida de servicio y su importancia para el negocio	1. Servicios secundarios mínimos interrumpidos 5. Servicios primarios mínimos interrumpidos, servicios secundarios extensos interrumpidos 7. Servicios primarios extensos interrumpidos 9. Todos los Servicios completamente perdidos
Pérdida de Trazabilidad del Atacante	Las acciones pueden ser rastreadas hasta un individuo	1. Totalmente Trazable 7. Posiblemente rastreable 9. Completamente anónimo

Adaptado de *(The OWASP Fundation, 2015)*.
Nota: Las puntuaciones son tomadas en base al modelo de *Risk Rating Template* de OWASP.

- **Impactos de Negocio**

El impacto en el negocio se deriva del impacto técnico, este requiere de un profundo conocimiento de lo que es importante para la empresa que ejecuta la aplicación. En general, el objetivo de este impacto es valorar el riesgo en el negocio, sobre todo a nivel ejecutivo. El impacto en el negocio es lo que justifica la inversión en la solución de problemas de seguridad. Muchas compañías tienen una guía de clasificación de activos y/o una referencia de impacto en el negocio para ayudar a formalizar lo que es importante para su negocio. Estas normas ayudan a concentrarse en lo que es realmente importante para la seguridad. Los factores siguientes son áreas comunes para muchas empresas (The OWASP Fundation, 2015).

Tabla 5.
Impactos de Negocio.

Factor de Vulnerabilidad	Descripción	Puntuaciones
Daño Financiero	Daño financiero aplicado al negocio	1. Menos que el costo de arreglar la vulnerabilidad 3. Efecto menor en el beneficio anual 7. Efecto Significativo en el beneficio anual 9. Bancarrota
Daños a la reputación	Daño a la reputación aplicado al negocio	1. Daño mínimo 4. Pérdida de las principales cuentas 5. Pérdida de prestigio y clientela 9. Daño a la marca
Incumplimiento	Exposición si se sigue incumpliendo	2. Violación menor 5. Violación clara

Factor de Vulnerabilidad	Descripción	Puntuaciones
Violación de Privacidad	Información personal divulgada	7. Violación de alto perfil 3. Un Individuo 5. Cientos de Personas 7. Miles de Personas 9. Millones de Personas

Adaptado de *(The OWASP Fundation, 2015)*.
Nota: Las puntuaciones son tomadas en base al modelo de Risk Rating Template de OWASP.

c. Determinar la gravedad del Riesgo

Con los parámetros obtenidos, se realiza la matriz de Riesgo por cada Vulnerabilidad y su grado para materializarse. La fórmula que se utiliza es la siguiente:

```
Riesgo = Probabilidad * Impacto
```

Figura 2. Fórmula de Riesgo Aplicada al Proyecto.
Adaptado de (The OWASP Fundation, 2017).

Se explica su matriz en Gravedad Global:

Gravedad Global del Riesgo = Probabilidad x Impacto

Impacto	ALTO	Medio	Alto	Crítico
	MEDIO	Bajo	Medio	Alto
	BAJO	Nota	**Bajo**	Medio
		BAJO	MEDIO	ALTO
		Probabilidad		

Figura 3. Matriz de Probabilidad x Impacto.
Adaptado de (The OWASP Fundation, 2017).

Para considerar la severidad de los riesgos (Figura 4). Se usan valores categóricos, dependiendo de la gravedad del Impacto al negocio o su probabilidad.

Niveles de probabilidad e impacto	
0 a <3	BAJO
3 a <6	MEDIO
6 a 9	ALTO

Figura 4. Categorización de Gravedad del Riesgo.
Adaptado de (The OWASP Fundation, 2017).

2.2. Metodologías para *PenTesting*

Las metodologías de *PenTesting* son necesarias para la organización, el desarrollo y mantenimiento de aplicaciones web, su principal objetivo debe ser la clasificación de riesgos informáticos, basados en las amenazas y su probabilidad de ocurrencia. Entre las metodologías de PenTesting más conocidas se encuentran:

- *OWASP Testing Guide V4*: Proyecto de código abierto encargado de valorar el nivel de inseguridad en las aplicaciones web, el equipo de *OWASP* está orientado a crear artículos, metodologías y documentación. Considerando las dimensiones de: personas, procesos y tecnologías. Su principal documentación es la de *OWASP Testing Guide v4*.
- *Payment Card Industry Data Security Standard Penetration Testing guide*: Es un comité conformado por las compañías de tarjetas de débito y crédito más importantes, su guía ayuda a las organizaciones que procesan, almacenan y/o transmiten datos de titulares de tarjeta, asegurando dichos datos, con el fin de evitar fraudes que involucren tarjetas de pago de débito y crédito.

- *Penetration Testing Execution Standard*: Es un estándar diseñado para proporcionar tanto a las empresas como a los proveedores de servicios de seguridad, un lenguaje común al alcance de todos para realizar pruebas de penetración. Comenzó a principios de 2009 después de una discusión entre algunos de los miembros fundadores sobre el valor de las pruebas de penetración en la industria.
- NIST 800-115: El Instituto Nacional de Estándares y de Tecnología (NIST), es responsable de desarrollar normas y directrices, incluidos los requisitos mínimos para proporcionar seguridad de información adecuada para todas las operaciones y activos de la agencia de los Estados Unidos de América, sin incluir los sistemas de seguridad nacional de dicho país.
- *Penetration Testing Framework* de Toggmeister y Lee Lawson: Ofrece un recorrido paso a paso de diferentes aspectos de una prueba de penetración de red, como el uso de herramientas especiales y comandos que se utilizan en cada herramienta.
- *Information Systems Security Assessment Framework* (ISSAF): Es un marco estructurado revisado por pares que categoriza la evaluación de seguridad del sistema de información en varios dominios (Unix/ Sistemas Linux, Seguridad en Base de Datos, etc.) detallando criterios específicos de evaluación y prueba para cada uno de estos dominios.
- *Open Source Security Testing Methodology Manual* (“OSSTMM 3”): Es un proyecto desarrollado por la ISECOM con una comunidad *open-source*, y sujeto a revisión interdisciplinaria y de pares. Protegido bajo la *Open Methodology License 3.0* que aplica a la protección que se otorga a Secretos Comerciales.

2.2.1. Ambiente de PenTesting

Es un sistema operativo centrado en la seguridad que ayuda a descubrir debilidades de los sistemas informáticos o de las redes informáticas. (Shakeel, 2017)**.**

a. Comparación de Ambientes de *PenTesting*

Existen diferentes distribuciones Linux para la realización de pruebas de PenTesting, diseñadas para casos específicos, por ejemplo; para Hackeo Web, Hackeo de Redes, Computación forenses, Hackeo Ético, etc. Estos ambientes de PenTesting abarcan una variedad de herramientas, por lo cual su selección dependerá de los activos de la empresa. (Shakeel, 2017). En la Tabla 2 se comparan las principales distribuciones de Linux existentes en el mercado y sus características para pruebas de *PenTesting* y hackeo ético:

Tabla *6.*
Comparación Distribuciones Linux para PenTesting.

Distribución	*Ethical Hacking*	*Pen Test*	Comp. Forense	Criptografía	*Web Testing*	Soporte Amazon Web Services
Kali Linux	*	*	*	*	*	*
Parrot Security OS	*	*	*	*	*	-
BackBox	*	*	-	-	-	-
Samurai Web Testing Framework	*	-	-	-	*	-
Pentoo Linux	-	*	*	*	-	-
DEFT Linux	*	*	*	*	-	-
C.A.I.N.E.	*	*	*	-	-	-
Network Security Toolkit (NST)	*	*	-	-	-	-
BlackArch Linux	-	*	*	*	*	-
BugTraq	*	*	*	-	-	-

Adaptado de (Shakeel, 2017).
Nota: (*) Se Considera como característica suficiente.

Como se puede observar Kali Linux y Parrot Security OS son los principales candidatos para realizar pruebas de PenTesting, la diferencia que existe entre estos dos ambientes es el soporte cloud de AWS para Kali Linux 2018 Anexo 1.

b. Ambiente de *PenTesting* seleccionado: Kali Linux

Kali Linux fue lanzado el 13 de marzo de 2013 como una completa reconstrucción de BackTrack Linux, es una distribución Linux basada en Debian destinada a realizar Pruebas de *PenTesting* y Auditorías de Seguridad avanzadas. Kali contiene diferentes herramientas que se orientan hacia la seguridad de la información, investigación de seguridad, computación forense e ingeniería inversa. Kali Linux está desarrollado, financiado y mantenido por Offensive Security (Offensive Security, 2018).

A continuación, se presentan las características más importantes de Kali Linux:

- Soporte de más de 600 herramientas para pruebas de penetración.
- Código Fuente Abierto.
- Compatible con FHS o Estándar de Jerarquía de Sistemas de Archivos.
- Compatibilidad con gran variedad de dispositivos inalámbricos.
- Kernel Personalizable con parches de inyección.
- Paquetes y repositorios firmados.
- Compatibilidad con dispositivos ARMEL y ARMHF.

2.2.2. Herramientas OWASP que se utilizan

En el presente proyecto se utiliza las herramientas y metodologías descritas en *OWASP Testing Guide V4*, las cuales al ser documentación de software libre y con una comunidad activa, permiten que las vulnerabilidades y técnicas usadas estén acorde con la demanda actual en seguridad de la información. A continuación, se presentan las pruebas realizadas en el presente proyecto:

a. Pruebas para Obtener de Información (*Information Gathering*)

- *Conduct Search Engine Discovery/Reconnaissance for Information Leakage (OTG-INFO-001):* Búsqueda de índices y contenido asociado a las cachés, recopilación de información sensible de diseño y configuración.

- *Fingerprint Web Server (OTG-INFO-002):* Conocer la versión y el tipo de servidor web en ejecución.
- *Review Web Server Metafiles for Information Leakage (OTG-INFO-003):* Fuga de información del directorio o carpeta de la aplicación web, listado de directorios.
- *Enumerate Applications on Web Server (OTG-INFO-004):* Enumerar aplicaciones en el Servidor Web.
- *Review Web Page Comments and Metadata for Information Leakage (OTG-INFO-005):* Enumerar los comentarios e información guardada como metadatos.
- *Identify application entry points (OTG-INFO-006):* Entender cómo se forman las solicitudes y las respuestas típicas de la aplicación.
- *Map execution paths through application (OTG-INFO-007):* Comprender la estructura y funcionamiento del código.
- *Fingerprint Web Application Framework (OTG-INFO-008):* Entender el lenguaje en el que está desarrollado el *framework* de la aplicación web.
- *Fingerprint Web Application (OTG-INFO-009):* Identificar la aplicación web y la versión para determinar las vulnerabilidades conocidas y los *exploits* apropiados para usar durante las pruebas.

b. Pruebas de Validación de Datos (*Input Validation Testing*)

- *Testing for Reflected Cross Site Scripting (OTG-INPVAL-001):* Inyección de código ejecutable del navegador dentro de una única respuesta HTTP.
- *Testing for Stored Cross Site Scripting (OTG-INPVAL-002):* Recopilación de información maliciosa, para su uso posterior.
- *Testing for HTTP Verb Tampering (OTG-INPVAL-003):* Respuesta inesperada a otras peticiones estándar GET y POST.
- *Testing for HTTP Parameter Pollution (OTG-INPVAL-004):* Interpretación imprevista a varios parámetros HTTP.
- *Testing for SQL Injection (OTG-INPVAL-005):* Inserción total o completa de sentencias SQL desde el cliente.

- *Testing for LDAP Injection (OTG-INPVAL-006):* Ataque del lado del servidor, modificación o insertando información sensible sobre usuarios y hosts representados en una estructura LDAP.
- *Testing for ORM Injection (OTG-INPVAL-007*): Inyección SQL contra un modelo de objetos de acceso a datos generado por ORM (*Object Relational Model*).
- *Testing for XML Injection (OTG-INPVAL-008):* Inyección de un documento XML en la aplicación.
- *Testing for SSI Injection (OTG-INPVAL-009):* Posibilidad de inyectar datos en la aplicación que serán interpretados por mecanismos SSI (*Server-Side Includes*).
- *Testing for XPath Injection (OTG-INPVAL-010):* Inyectar sintaxis *XPath* en una petición interpretada por la aplicación, permitiendo a un atacante ejecutar consultas *XPath* controladas por el usuario.
- *Testing for IMAP/SMTP Injection (OTG-INPVAL-011):* Capacidad de inyectar comandos IMAP/SMTP arbitrarios en los servidores de correo.
- *Testing for Code Injection (OTG-INPVAL-012):* Introducir código como entrada en una página web y ejecutarlo por el servidor web.
- *Testing for Command Injection (OTG-INPVAL-013):* Inyectar comandos del sistema operativo a través de una petición HTTP.
- *Testing for Buffer Overflow (OTG-INPVAL-014):* Comportamiento imprevisto por acceso indebido a la memoria de la aplicación, puede ser *Heap* (almacenamiento de datos asignados dinámicamente y variables globales), *Stack* (pila de programas sin verificación de límites) y *Format String* (entrada de usuario no filtrada como parámetro de cadena de formato).
- *Testing for Incubated Vulnerability (OTG-INPVAL-015):* Conocido como ataques persistentes. Método complejo que necesita más de una vulnerabilidad de *Data Validation* para funcionar.
- *Testing for HTTP Splitting/Smuggling (OTG-INPVAL-016):* Ataques sobre características del protocolo HTTP, explotando debilidades de la aplicación web o “peculiaridades” que los agentes interpretan en mensajes HTTP.

c. Pruebas del lado del Cliente para Denegación de Servicio (*Client Side Testing - Denial of Service*)

- *Testing for DOM based Cross Site Scripting (OTG-CLIENT-001):* Errores *Cross Site Scripting* que son el resultado de contenido activo del lado del navegador en una página, ejecutando código malicioso.
- *Testing for JavaScript Execution (OTG-CLIENT-002):* Inyectar código JavaScript arbitrario, ejecutado dentro del navegador de la víctima.
- *Testing for HTML Injection (OTG-CLIENT-003):* Inyección de código HTML arbitrario en una página web vulnerable.
- *Testing for Client Side URL Redirect (OTG-CLIENT-004):* Redirección URL del lado del cliente, también conocida como redirección abierta.
- *Testing for CSS Injection (OTG-CLIENT-005):* Inyección de código CSS arbitrario en el contexto de un sitio web de confianza, renderizado dentro del navegador de la víctima.
- *Testing for Client Side Resource Manipulation (OTG-CLIENT-006):* Capacidad de controlar URLs que enlazan a algunos recursos presentes en una página web.
- *Test Cross Origin Resource Sharing (OTG-CLIENT-007):* Configuraciones inseguras del tipo "*cross-domain*", valores de cabecera *Access-Control-Allow-Origin*.
- *Testing for Cross Site Flashing (OTG-CLIENT-008):* Vulnerabilidades en patrones de validación basado en *ECMAScript* utilizado en aplicaciones *Flash.*
- *Testing for Clickjacking (OTG-CLIENT-009):* Problema de seguridad del lado del cliente al interactuar con la página web, por ejemplo, haciendo clic, con algo diferente a lo que el usuario cree que está interactuando.
- *Testing WebSockets (OTG-CLIENT-010):* Permiso restringido a los clientes de enviar y recibir datos de forma asíncrona (en segundo plano sin actualizar la página) al servidor.
- *Test Web Messaging (OTG-CLIENT-011):* Desarrollo inseguro para recibir comunicación desde diferentes fuentes (entre *iframes*, pestañas y ventanas).

- *Test Local Storage (OTG-CLIENT-012):* Manipulación de datos almacenados en el cliente, para manipulación de datos.

2.3. Recomendaciones para Planes de Mitigación de Ataques Informáticos

Para mitigar ataques informáticos se debe contar con un grupo de contención de Incidentes Computacionales o CSIRT (*Computer Security Incident Response Team*), encargado de que todas las actividades se lleven con el debido procedimiento legal, siguiendo normas establecidas por analistas de sistemas e informática forense, de tal manera que las personas encargadas de peritos informáticos tengan evidencias respaldadas ante organismos competente (NIST, 2012). Entre las fuentes recomendadas para crear un plan de Mitigación de Ataques Computacionales para respuesta de incidentes computacionales, existen:

- *The Incidence Handlers Handbook*: Realizado por *SANS Institute InfoSec Reading Room*, recomienda seguir los pasos de; Preparación, Identificación, Contención, Erradicación, Recuperación y Lecciones Aprendidas (SANS Institute, 2011).
- *OWASP Top 10 Considerations For Incident Response*: Realizado por *OWASP Team Foundation* y que realiza los pasos: Auditar y ser diligente, crear un Equipo de Respuesta, crear un documento de Plan de Respuesta de Incidentes (basado en estándares ISO/IEC 27000), categorizar factores que desencadenen incidentes, Investigar el Problema, Clasificar y Priorizar Incidentes, Recuperación, Documentación y Reporte, Revisión y por último práctica continua (The OWASP Fundation, 2015).
- *Computer Security Incident Handling Guide*: Elaborado por NIST y que propone un ciclo continuo de vida para Incidentes, compuesto por: Preparación, Detección y Análisis, Mitigación y Prevención y Actividades Post-Incidente. A su vez recomienda 11 pasos en caso de no disponer de un CSIRT (NIST, 2012).

- *Incidence Response & Computer Forensics*: Elaborado por Jasson T. Luttgens, Kevin Mandia y Matthew Pepe, los cuales parten desde el punto de vista práctico y actual del Internet de las cosas (*IoT*), las cuales pueden convertirse en potenciales vectores de ataques para la empresa. Su metodología se basa en 5 pasos cíclicos y consecuentes al descubrimiento de indicios de vulnerabilidades; Creación de Indicadores Comprometidos (*IOC*), desarrollo de *IOC*, Identificar Sistemas de Interés, Recolectar Evidencia y Analizar Datos (Mandia, et al., 2014).

2.3.1. Descripción de las recomendaciones de NIST sobre Incidentes Computacionales

La Respuesta a Incidentes Computacionales es la reacción de una organización a acciones ilegales o inaceptables que involucran una o varias computadoras y un componente de red, esta utiliza enfoques sistemáticos y bien organizados para reaccionar (NIST, 2012).

Los incidentes deben ser manejados por un Equipo de Respuesta a Incidentes de Seguridad Computacional (*Computer Security Incident Response Team, CSIRT),* el cual está compuesto por personal con diferentes perfiles especializados, con experiencia legal y técnica en seguridad de la información, que son necesarios durante el proceso de respuesta. El *CSIRT* coordina la respuesta apropiada ante un incidente, proporcionando efectivamente las capacidades de contestación ante incidentes de seguridad informática (NIST, 2012).

Algunos de los beneficios clave que un *CSIRT* organizado ofrece son, por ejemplo, una confirmación ya sea que haya o no ocurrido un incidente, una detección rápida, contención y recuperación de incidentes (NIST, 2012).

Las recomendaciones describen el proceso de Respuesta a Incidentes, utilizando cuatro fases diferentes, estas son:

a. Preparación de Pre-incidentes

La preparación de pre-incidentes es una fase continua, y en realidad es la única que tiene lugar incluso antes de que ocurra un incidente. Su objetivo es preparar a la organización y al CSIRT para un posible incidente. La preparación de la organización puede implicar la implementación de medidas de seguridad basadas en la red y en el servidor, por ejemplo, la implementación de un sistema de detección de intrusos (NIST, 2012).

b. Detección y Análisis de Incidentes

Para la detección automática de incidentes, es necesaria la instalación de un Sistema de Detección de Intrusos o IDS, con el objetivo de analizar el tráfico de la red (*NIDS*) o del comportamiento del servidor (*HIDS*). Existen soluciones de software libre y de pago, entre los más populares se tiene; Snort, Fail2Ban, Bro, etc. (NIST, 2007). Otras opciones teóricas se basan en el uso eficiente de energía y recursos utilizando redes neuronales y arboles bayesianos (Viegas, Santin, França, Jasinski, Pedroni, y Oliveira, 2017). También es recomendable tener un servidor de antivirus como: ClamAV, Malwarebytes, Avast, entre otros (NIST, 2012).

En caso de no poder realizar una detección automática de incidentes, una alternativa son las pruebas de *PenTest*. El *CSIRT* o la persona encargada en la seguridad de los sistemas, puede implementar pruebas de Penetración adecuando parámetros a las necesidades del negocio. El análisis debe incluir la criticidad de los sistemas o datos afectados, tipo de ataque y daño general recibido o esperado. Debido a que un incidente también puede inducir una acción legal o administrativa, el desarrollo de la estrategia tiene que involucrar a las personas responsables de las respectivas áreas, lo que significa, ejecutivos de alta dirección, departamento de Recursos Humanos, entre otros. (NIST, 2012).

c. Mitigación, Contención y Recuperación de Ataques Informáticos

Toda organización debe asumir un riesgo tolerable en sus sistemas informáticos, ante lo cual se debe tener una estrategia de mitigación de ataques informáticos. Con la información recabada en la detección y análisis del sistema se debe guardar evidencia necesaria con respecto al ataque y su prevención tal como:

- Ubicación, Número de Vulnerabilidad CVE, El nombre de host, Direcciones MAC y direcciones IP de la o las computadoras.
- Gravedad del incidente.
- Prioridad de respuesta/resolución de incidente y asignar tareas y roles de mitigación.
- Nombre, cargo y número de teléfono de cada individuo que recolecta o manipula la evidencia del ataque, durante la investigación y Hora y fecha (incluida la zona horaria).
- Ubicaciones donde se almacena la evidencia del ataque.

Dependiendo de la tecnología disponible generar varios escenarios de mitigación de ataques informáticos, estos son; actualizar tecnologías, cambiar los *plugins*, comprar licencias, cambiar el sistema por completo, etc. (NIST, 2012).

d. Actividades Post Incidente

El objetivo del proceso de post-incidente es establecer o proponer mejoras en el actual proceso de contención de incidentes informáticos, restableciendo el sistema a la normalidad al eliminar vulnerabilidades conocidas (NIST, 2012).

2.4. Descripción de la Metodología en este proyecto

La metodología del presente proyecto se basa en OWASP para pruebas de PenTesting y recomendaciones de NIST 800-61 para la creación de un plan de mitigación de ataques informáticos. Los acuerdos de confidencialidad y patrocinios por parte de la Institución de Educación Superior se realizaron bajo

términos de confidencialidad [Anexo 2]. La metodología de OWASP de *Blind PenTesting*, incluye el modo pasivo y modo activo de pruebas, para la evaluación de Riesgos Informáticos se usa *OWASP Risk Rating Methodology*, el cual provee la priorización de riesgos en base a los activos y escenarios del negocio. Por último, la propuesta de mitigación de ataques informáticos usa recomendaciones de Respuesta ante Incidentes Computacionales descritas por NIST.

El modo Pasivo se refiere a las irregularidades encontradas mediante la exploración de la página web y su información encontrada mediante el uso continuo de la plataforma, esto indica la necesidad de descubrir la gravedad de las vulnerabilidades y el grado de afectación a los internautas, así como para la Institución de Educación Superior. El modo Activo, se realiza mediante las pruebas recabadas en *Information Gathering*, *Input Validation Testing* y *Client Side Testing,* entregando una guía general para una Institución de Educación Superior. En la figura se muestran las metodologías resumidas y el curso de acciones seguidas para proponer un plan de mitigación de ataques informáticos.

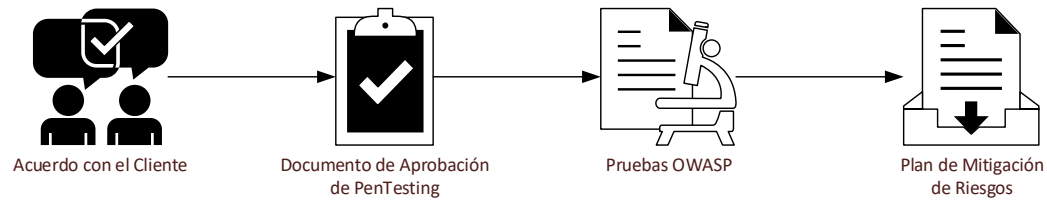

Figura 5. Diagrama de Metodología Usada.

3. Information Gathering

En el presente capítulo se realizarán las pruebas Iniciales, utilizando el ambiente de *PenTesting* Kali Linux 7.2 en *VMware*, basados en la fase 2 del Capítulo 4 de *OWASP Web Application Security Testing*, se realiza la recolección de información (*Information Gathering*), Test de Validación de Datos de Entradas (*Input Validation*) y Test desde el Lado del Cliente (*Client-Side Testing*). Después se incluye el riesgo total, de las aplicaciones que están expuestas. En la Figura 6 se presenta el curso de la metodología OWASP para aplicaciones Web.

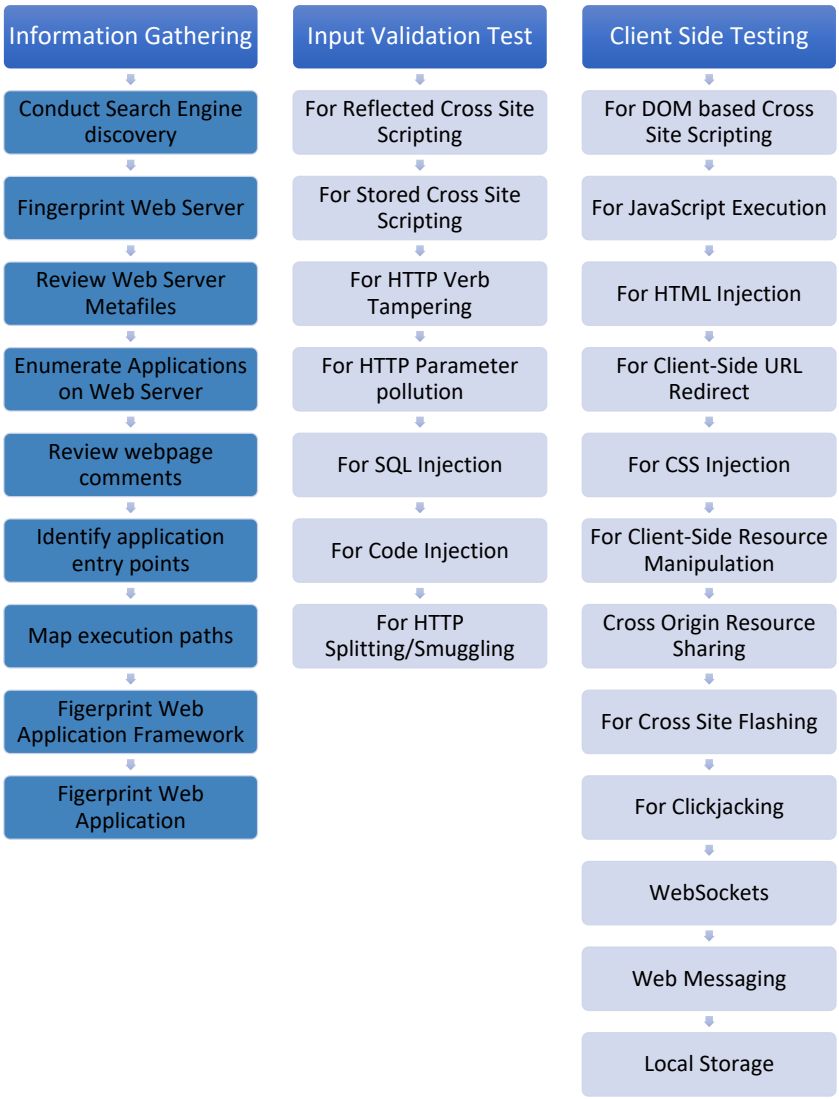

Figura 6. Diagrama de Pruebas de *Information Gathering* de OWASP.

3.1. Conduct search engine discovery/reconnaissance for information leakage (OTG-INFO-001)

Objetivo: Recabar información sensible de diseño y configuración de la aplicación.

Herramienta: SiteDigger, Búsqueda en la caché de Google para detectar vulnerabilidades, errores, problemas de configuración, información patentada y fragmentos de seguridad.

Enlace: https://www.mcafee.com/uk/downloads/free-tools/sitedigger.aspx

Requisitos: Windows con .NetFramework *3.5* o superior.

Uso: Introducir el dominio y seleccionar el tipo información que se desea buscar en los checkboxs que se muestran a la izquierda.

Acciones Adicionales: Es necesario depurar la información manualmente.

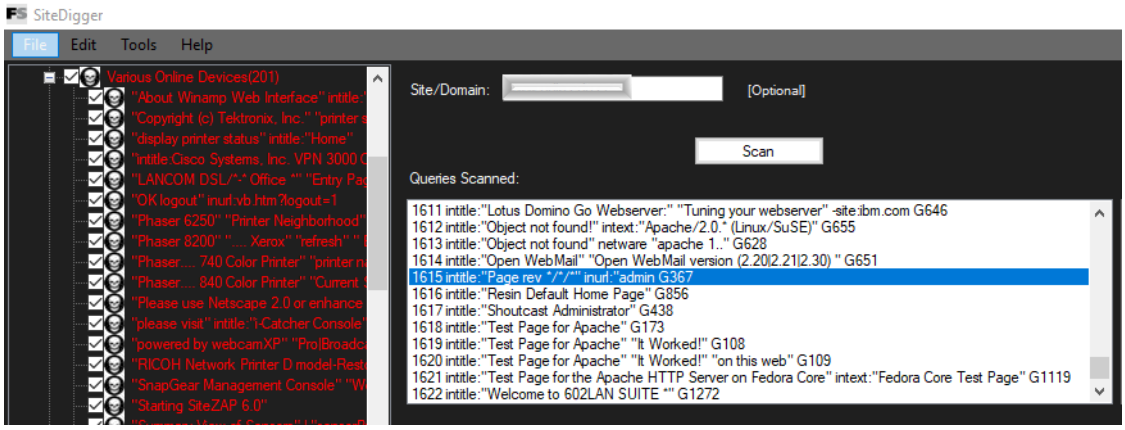

Figura 7. Resultados de SiteDigger.

Resultados iniciales: Aquí no se encuentran resultados valiosos ya que el sitio usa un CMS el cual mediante reglas de .htaccess bloquea cualquier ingreso directo a la raíz del sitio siempre redirigiendo al index.php del mismo.

3.2. Fingerprint Web Server (OTG-INFO-002)

Objetivo: Conocer la versión y el tipo del servidor web en ejecución permitiendo determinar vulnerabilidades conocidas y exploits apropiados para utilizar durante las pruebas.

Herramienta: NetCraft, herramienta online que proporciona análisis de cuota de mercado de servidores y alojamiento web, incluyendo detección de servidores web y sistemas operativos.

Enlace: https://www.netcraft.com/

Requisitos: Navegador web Firefox, Google Chrome o Microsoft Explorer.

Uso: Ingreso de la URL en el buscador NetCraft.

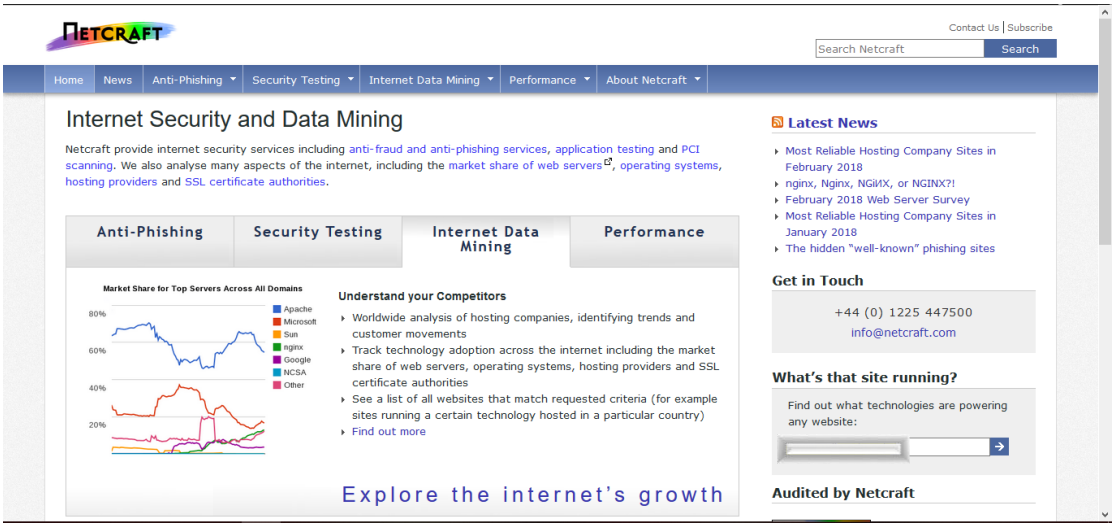

Figura 8. Página de Inicio de NetCraft.

Resultados iniciales:

Tabla 7.
Resultados de NetCraft.

Tipo de Servidor Web	Apache 2.2.15
OS	Linux CentOS
Compañía de Hosting	Microsoft - US East (Virginia) datacenter
Información WhoIs	Bloqueada
IP	[redacted]
CMS	Wordpress
Tecnología de la Plataforma Web	Azure
Nombre NameServer	[redacted]

Nota: Los valores ocultos en la tabla representan datos personales de la Institución de Educación Superior.

3.3. Review Web Server Metafiles for Information Leakage (OTG-INFO-003)

Objetivo: Detectar fugas de información del directorio de la carpeta de la aplicación web.

Herramienta Para Usar: DomainTools, herramienta online que toma muestras de la red, incluye dominios e IPs, y su vinculación con casi todos los dominios activos en Internet, muestra información relevante sobre el DNS.

Enlace: http://whois.domaintools.com/

Requisitos: Navegador web Firefox, Google Chrome o Microsoft Explorer.

Uso: Ingreso del dominio a investigar en el buscador de DomainTools.

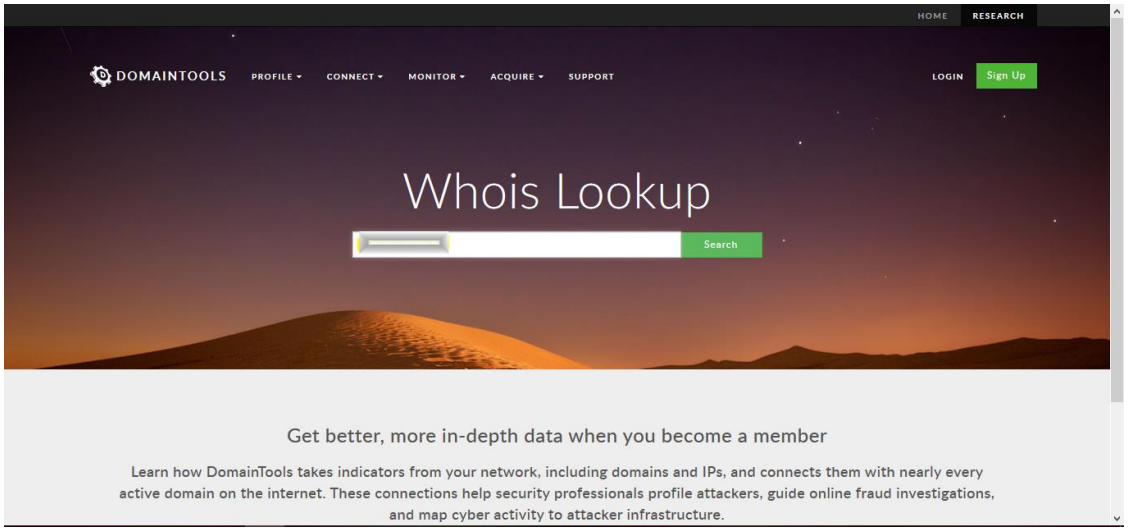

Figura 9. Página Principal de DomainTools.

Resultados iniciales: Los registros DNS se encuentran bloqueados al público, se pudo obtener los estados iniciales de la página con imágenes históricas guardadas en diferentes fechas.

3.4. Enumerate Applications on Web Server (OTG-INFO-004)

Objetivo: Averiguar qué aplicaciones concretas están alojadas en un servidor web.

Herramienta: NMap para escaneo comprehensivo, utilidad gratuita y de código abierto para la detección de redes y auditoría de seguridad, utiliza paquetes IP sin procesar.

Enlace: https://nmap.org/download.html

Requisitos: Sistema Operativo Windows o Linux.

Uso: Ingresar la dirección o IP que se requiere en la barra Objetivo.

Acciones Necesarias: Configuración sobre el tipo de escaneo en la barra perfil y revisión manual de los registros devueltos.

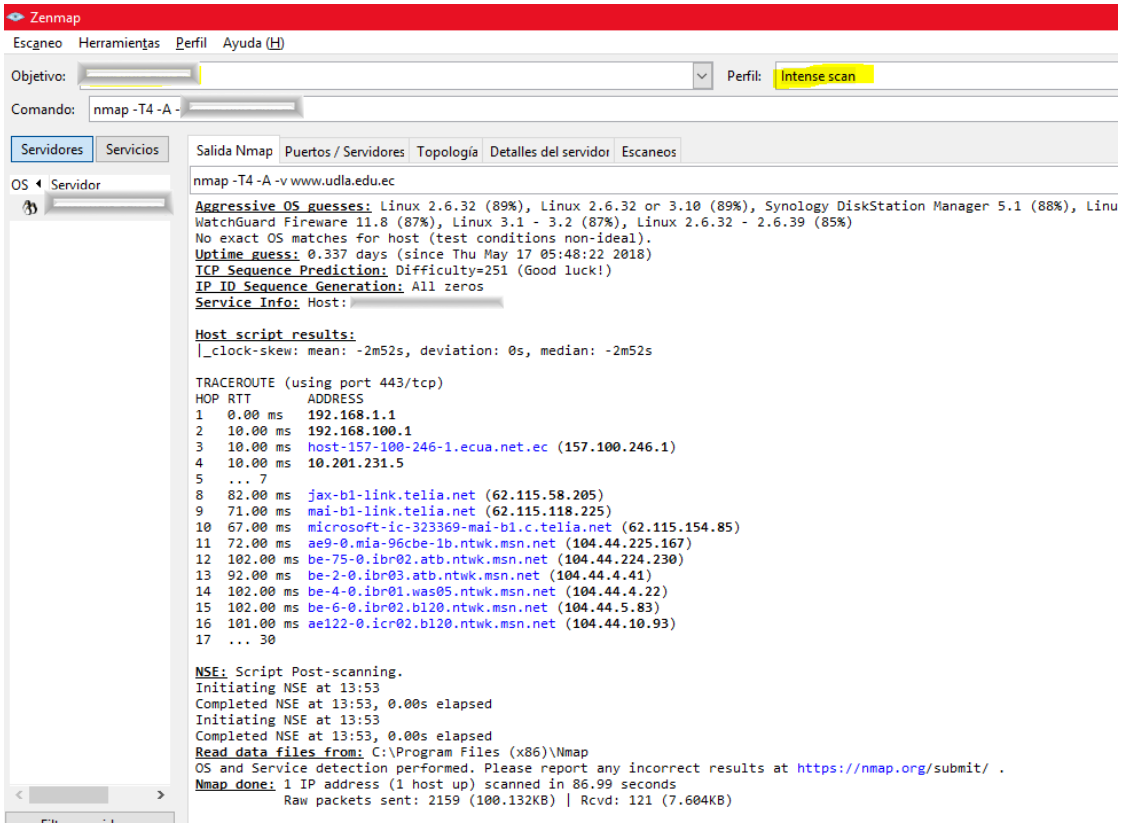

Figura 10. NMAP Escaneo sobre la página web.

Resultados iniciales: Topología de red y Anexo 3.

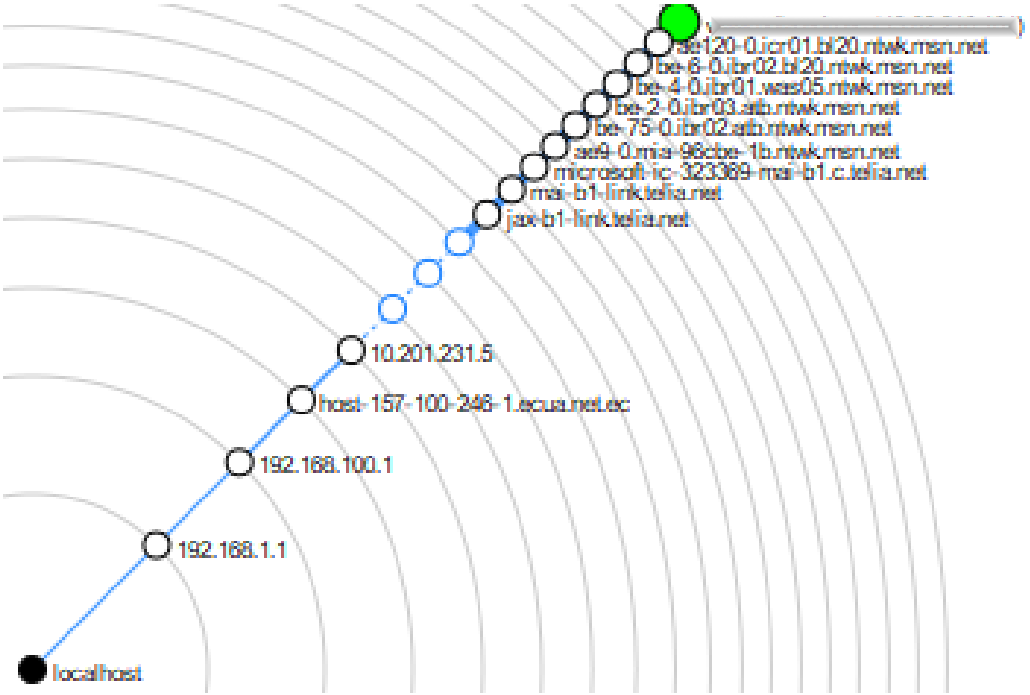

Figura 11. NMAP topología de saltos.

Información Extra:

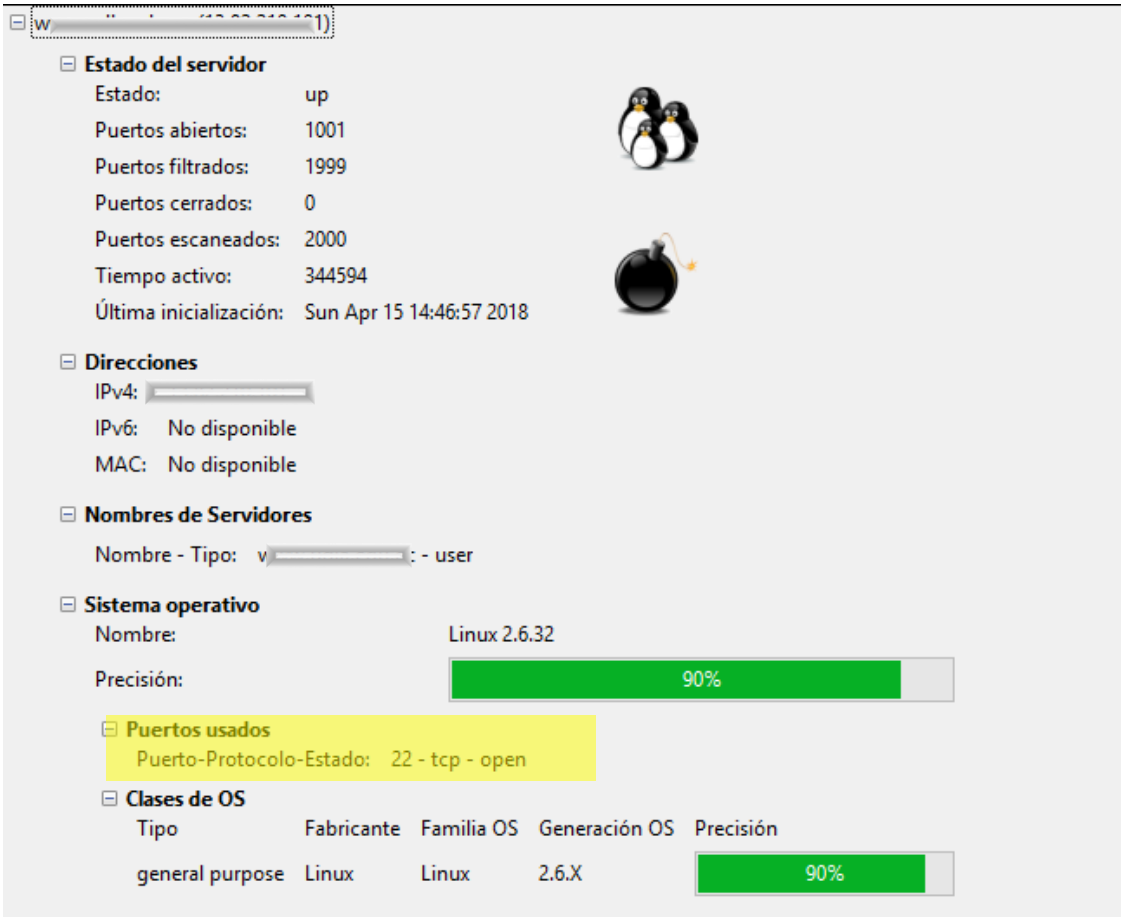

Figura 12. Estado del Servidor.

Solo está abierto el puerto 22, la configuración del funcionamiento de red de la página web es difícil de determinar debido a su estructura en la nube.

3.5. Review webpage comments and metadata for information leakage (OTG-INFO-005)

Objetivo: Búsqueda de comentarios detallados, metadatos y código fuente en el aplicativo.

Herramienta: OWASP ZAP, herramienta para encontrar automáticamente vulnerabilidades de seguridad en aplicaciones web.

Enlace: https://github.com/zaproxy/zaproxy/wiki/Downloads

Requisitos: Sistema Operativo Windows, Linux o Mac.

Uso: Poner la URL a atacar y luego cambiar las preferencias a opción “ATTACK Mode”, para que se muestre información completa sobre el sitio y poder listar las posibles vulnerabilidades y código fuente.

Acciones Adicionales: Revisión de Logs y de Falsos-Positivos.

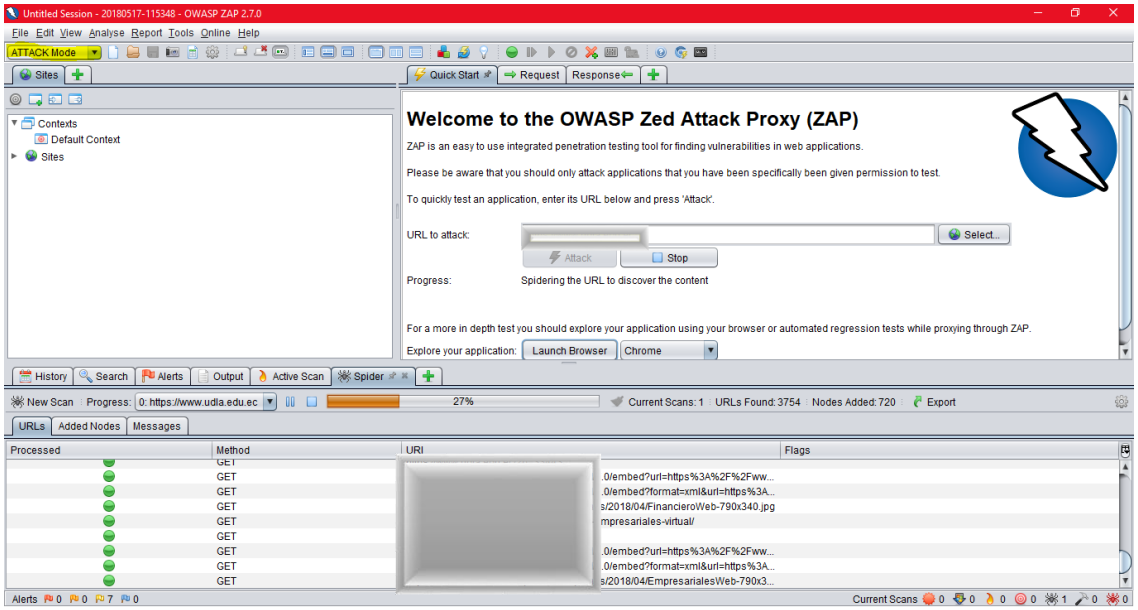

Figura 13. OWASP *Zed Attack Proxy*, página de Bienvenida.

Resultados iniciales: Se encontró la siguiente información relevante:

- Posible código de fuente de la aplicación divulgado por el servidor web – SQL, esta información, debe ser validada para encontrar el posible fallo de seguridad:
  - Se encontró la siguiente sentencia: **SELECT * FROM carrera WHERE 1 and code like.**
- Posible código fuente de la aplicación divulgado por el servidor web – ActiveVFP:

- Se divulga información del código fuente en uno de los forms usados en el sitio, con lo cual se puede iniciar ataques de ingeniería inversa para obtener más información.

Evidencia: Revisar el Anexo 4.

3.6. Identify application entry points (OTG-INFO-006)

Objetivo: Comprender cómo se forman las solicitudes y las respuestas típicas de la aplicación.

Herramienta: OWASP ZAP.

Resultados Iniciales: La aplicación al ser un CMS controla los *POST* y *GET* por lo cual es difícil tratar de hacer cambios en los mismo mediante la interceptación de estas solicitudes.

3.7. Map execution paths through application (OTG-INFO-007)

Objetivo: Mapear la aplicación objetivo y entender los flujos de trabajo principales.

Herramienta Para Usar: OWASP ZAP.

Resultados iniciales: Mapeo de la aplicación, encontrando el número aproximado de (14000) links además de concluir que se ejecuta una aplicación CMS WordPress, como informativo se encuentra una aplicación extra en *ReactJS*, su enlace está obsoleto.

3.8. Fingerprint Web Application Framework (OTG-INFO-008)

Objetivo: Definir el tipo de framework web utilizado para tener una mejor comprensión de la metodología de pruebas de seguridad.

Herramienta: WAPPALYZER, utilidad multiplataforma que descubre las tecnologías utilizadas en los sitios web. Detecta sistemas de gestión de contenido, plataformas de comercio electrónico, marcos de trabajo web, software de servidor, herramientas de análisis, entre otras.

Enlace: https://www.wappalyzer.com/download

Requisitos: Navegador web Firefox, Google Chrome o Microsoft Explorer.

Uso: Visitar el sitio y en la barra de URL del navegador verificar las tecnologías que usa el sitio.

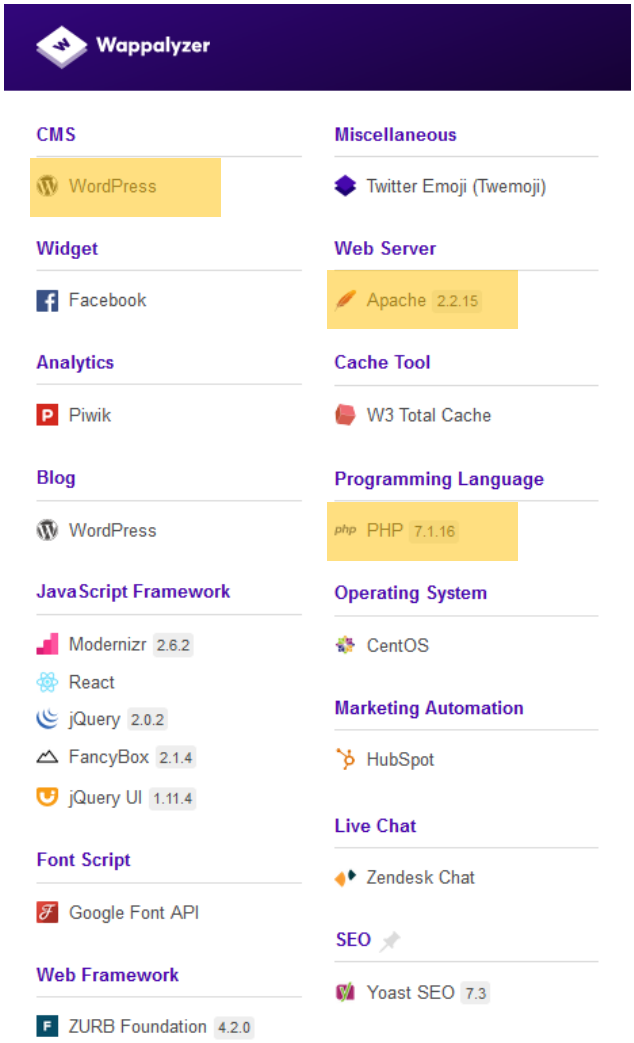

Figura 14. Listado de Tecnologías del Sitio.

Resultados iniciales: Lo importante de este análisis es la presencia de WordPress y la versión de PHP, el tipo y la versión del Web Server.

3.9. Fingerprint Web Application (OTG-INFO-009)

Objetivo: Identificar la aplicación web y la versión para determinar las vulnerabilidades conocidas y los exploits apropiados para utilizar durante las pruebas.

Herramienta: WAPPALYZER y Nikto el cual es un escáner de sitios web que comprueba la existencia de archivos/CGIs peligrosos, software de servidor obsoleto y otros problemas en los servidores web. Realiza comprobaciones genéricas y específicas del tipo de servidor. También captura e imprime las *cookies* recibidas.

Enlace: Solo para Linux: https://cirt.net/Nikto2

Requisitos: Nikto se encuentra instalado por defecto en Kali Linux.

Uso: Por medio de comandos; “nikto -host” para agregar una IP o URL y --output el cual guarda todo en un archivo.

Ejemplo: nikto -host https://www.abc.com --output pres.txt

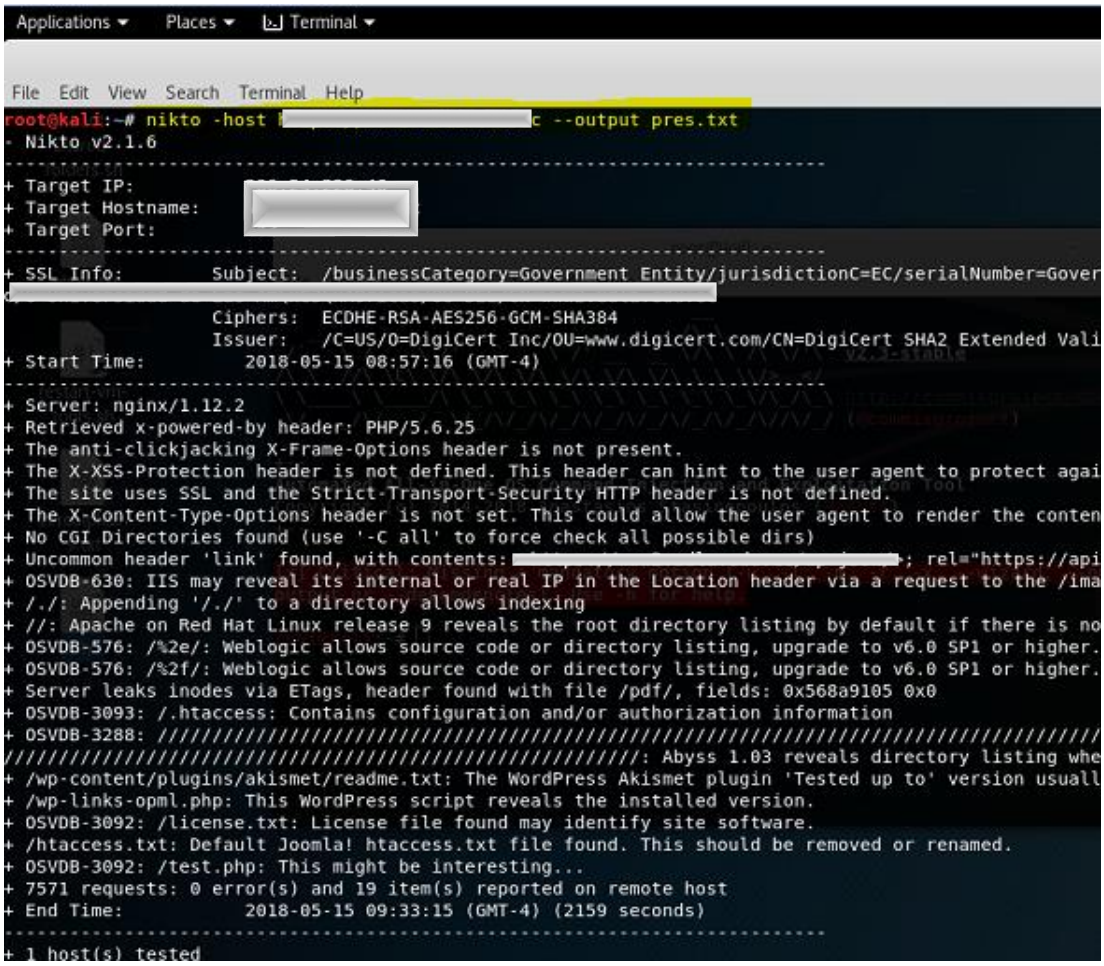

Figura 15. Uso por comandos de Nikto.

Resultados Iniciales: Acceso al readme del CMS de la página web el cual se encuentra habilitado, dando por consiguiente la versión de este:

Figura 16. Versión del CMS.

Además, se tiene acceso a toda la información de PHP del servidor.

Evidencia: El output del programa Nikto muestra la información relevante, línea 31 del Anexo 5.

3.10. Resultados de *Information Gathering*

En la Tabla 8 se presenta un resumen con las pruebas realizadas, con el código correspondiente de cada prueba.

Tabla 8.
Resultados de Information Gathering.

OTG-INFO	Herramienta	Resultado
001	SiteDigger	No se encontraron resultados relevantes, el sitio usa un CMS el cual mediante reglas del .htaccess bloquea cualquier ingreso directo a la raíz del sitio siempre redirigiendo al index.php del mismo.
002	NetCraft	Tipo de Servidor Web, OS, Compañía de Hosting, Información WhoIs, IP, CMS Tecnología de la Plataforma Web Nombre *NameServer*
003	DomainTools	Registros DNS se encuentran bloqueados al público, se pudo obtener estados iniciales de la página con imágenes históricas guardadas en diferentes fechas.
004	NMap	Solo está abierto el puerto 22, la configuración del funcionamiento de red de la página web es difícil de determinar debido a su estructura en la nube.

OTG-INFO	Herramienta	Resultado
005	OWASP Zap	Posible código fuente de la aplicación divulgado por el servidor web – SQL Se encuentra la siguiente sentencia: SELECT * FROM carrera WHERE 1 and code like Posible código fuente de la aplicación divulgado por el servidor web – ActiveVFP Se divulga información del código fuente de uno de los forms usados en el sitio, con los cuales se puede iniciar ataques de ingeniería inversa para obtener más información.
006	OWASP Zap	La aplicación al ser un CMS controla los *POST* y *GET* por lo cual es difícil tratar de hacer cambios en los mismo mediante la interceptación de estas solicitudes.
007	OWASP Zap	Mapeo de la aplicación, encontrando el número aproximado de (14000) links además de concluir que se ejecuta una aplicación CMS WordPress, como informativo se encuentra una aplicación extra en *ReactJS*, su enlace está obsoleto.
008	WAPPALYZER	La presencia de WordPress y la versión de PHP, el tipo y versión del Web Server.
009	WAPPALYZER y Nikto	Archivo de configuración de php, shellshock, readme y versión de WordPress.

4. Data Test Validation

Se pretende conocer si la aplicación web valida correctamente las entradas provenientes del cliente y del entorno virtual antes de usarlas. Estas debilidades conllevan a las principales vulnerabilidades en las aplicaciones web, como scripts de sitios cruzados (*Cross Site Scripting* XSS), inyección de SQL (*SQL Injection*), desbordamientos de búfer *(Buffer Overflow)*, etc. Se prueba todas las formas posibles de *inputs* para comprender si la aplicación valida los datos antes de usarlos.

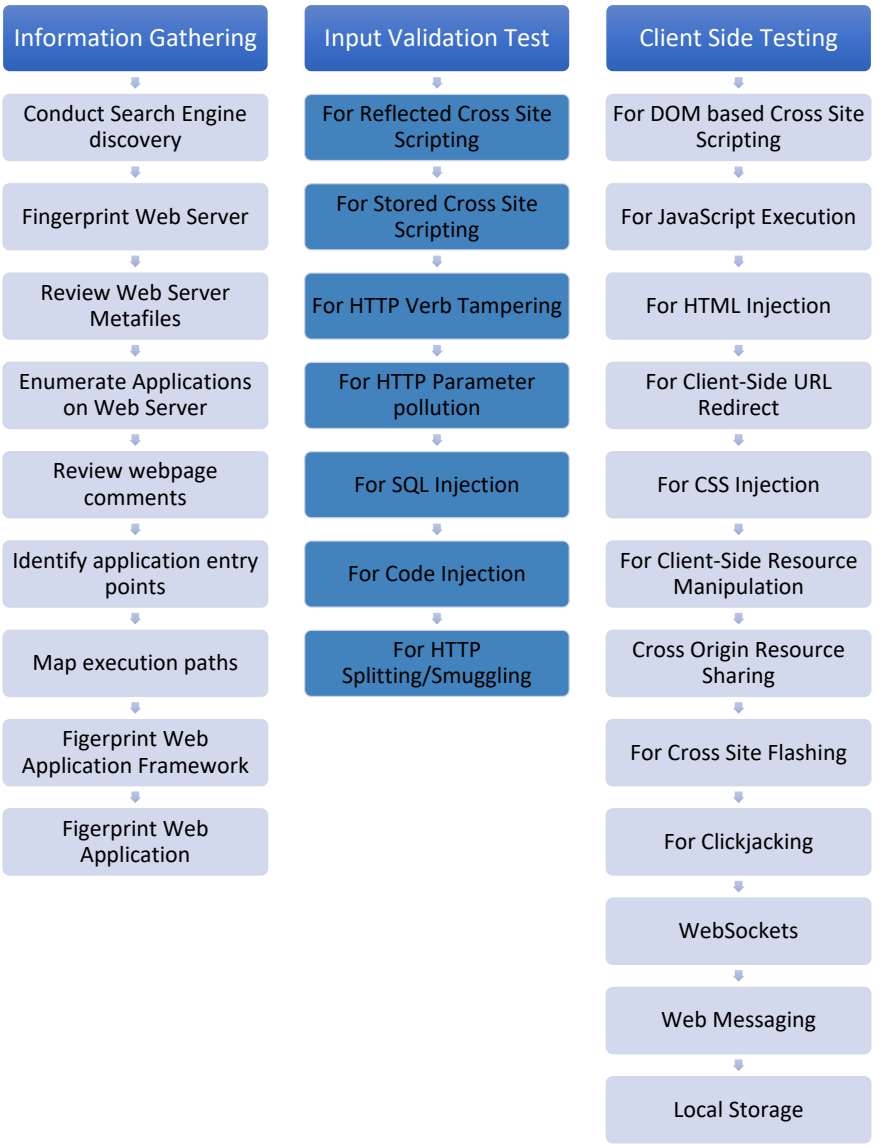

*Figura 17.*Diagrama de Pruebas de *Input Validation* de OWASP.

4.1. Test Validados y realizados

Las siguientes pruebas se realizaron en concordancia con la estructura e información encontrada en el capítulo 3, por lo cual la probabilidad de encontrar resultados es mayor en las pruebas de *Input Validation* de OWASP.

4.1.1. Testing for Reflected Cross Site Scripting (OTG-INPVAL-001)

¿Qué es?: Ataque utilizado para ejecutar código HTML y JavaScript en la sesión del usuario, para mostrar a la víctima contenido falso, con fines de guardar información confidencial del mismo sin su autorización.

Objetivo de la prueba: Vulnerar la página web para que ejecute código JavaScript, afectando a la sesión de los usuarios, realizado por medio de enlaces maliciosos.

Herramienta: Navegador Web, barra de búsqueda.

Diagrama de Ataque:

1. Un enlace es enviado, por ejemplo, por correo electrónico a los usuarios habituales del sitio web, indicándoles que deben actualizar sus contraseñas, para el registro correcto.
2. Al momento de ingresar a la página web, el sitio web, envía la respuesta habitual y la parte del *script* se genera en la sesión del cliente, debido a que el sitio web, no elimina dicha parte del *script* lo ejecuta como si fuera una búsqueda corriente.
3. El usuario recibe la página web y llena los datos mostrados o realiza alguna acción en la página web.
4. Este evento es el que se puede guardar en el sitio del atacante y con ello se puede robar información valiosa. En la Figura 18 se ilustra el ataque.

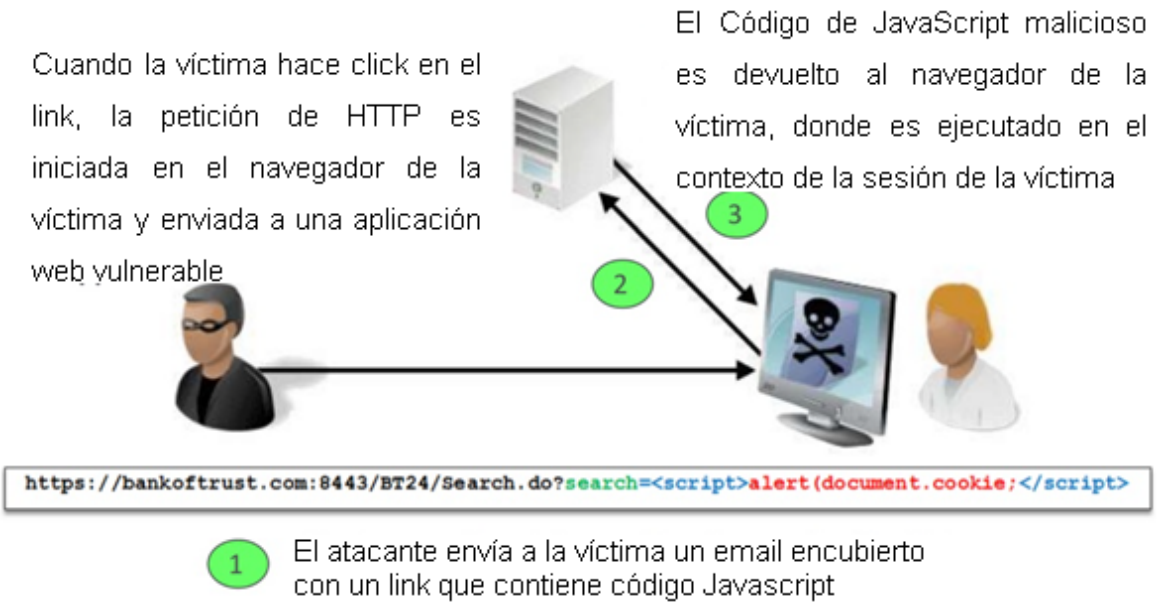

Figura 18. Cross Site Script esquema de ataque.
Adaptado de (Codecamp Romania, 2012).

Extracto de Script:

```
';alert(String.fromCharCode(88,83,83))//\';alert(String.fromCharCode(88,83,83))//";al
ert(String.fromCharCode(88,83,83))//\";alert(String.fromCharCode(88,83,83))//--
></script>">'><script>alert(String.fromCharCode(88,83,83))</script>
```

Figura 19. Extracto de Script para verificar *Cross Site Script*.

Explicación de Script: Se usa la palabra String.fromCharCode(88,83,83) para que no se filtren caracteres especiales de “<”, de esta manera se realiza un baipás de filtros, realizando la ejecución del *script*, sin que el CMS lo filtré.

Evidencia:

Figura 20. XSS en página principal.

Resultado: El sitio es vulnerable a *Reflected XSS*, por lo que puede ser posible un ataque como robo de cookies o la sesión de usuarios e información de ingreso del CMS.

Impacto negativo para el negocio: Alto, debido a que se puede realizar robo de sesiones de usuario.

4.1.2. Testing for Stored Cross Site Scripting (OTG-INPVAL-002)

¿Qué es?: Verificar las restricciones de los formularios y su forma de guardado en la base de datos, si su programación es adecuada o se pueden cargar archivos o scripts maliciosos en la base de datos, tales como registros, o *scripts* desde los formularios de las aplicaciones web, con la finalidad de que, si otras personas llaman a ese registro, se ejecute el código malicioso.

Objetivo: Verificar si el servidor de la aplicación valida los datos de formularios, debido a que las entradas almacenadas pueden no estar correctamente filtradas. Como consecuencia, los datos maliciosos aparecen como parte del sitio web y se ejecutan dentro del navegador del usuario bajo los privilegios de la aplicación web.

Herramienta: Navegador Web y Beef.

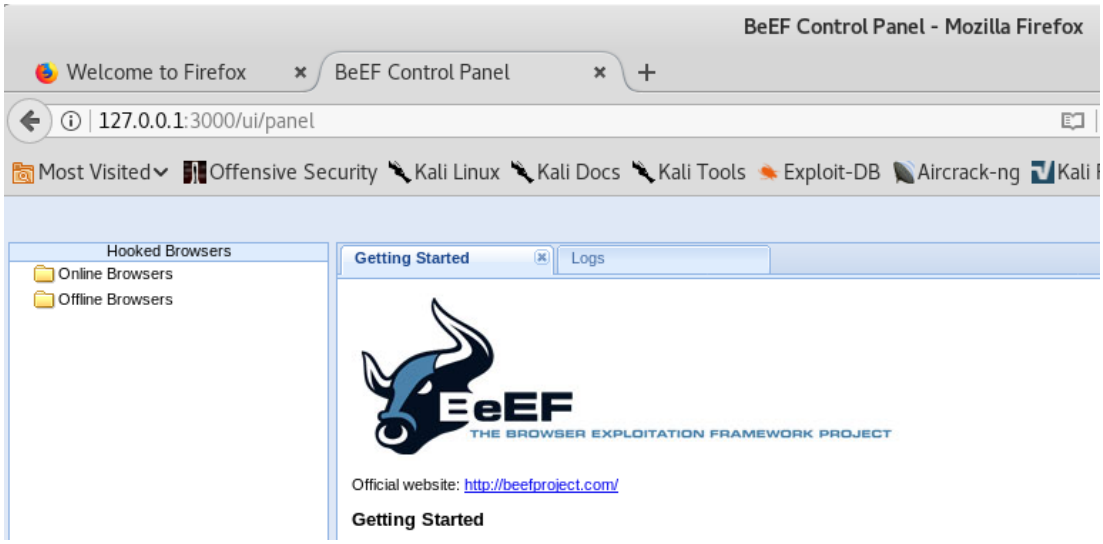

Figura 21. Página de Inicio del Servidor Web Beef.

Diagrama de Ataque:

1. El atacante descubre que un formulario en la aplicación web no tiene las validaciones correspondientes, por lo cual se prepara a guardar un *script* en los campos del formulario.
2. Este código se guarda en la base de datos, si este no cuenta con la sanitización adecuada para quitar caracteres especiales o en su defecto los guarda tal como el atacante desea, se convierte en un *script* ejecutable.
3. La víctima hace una consulta a la base de datos, la cual puede ser el listado de comentarios o las entradas que se realizaron en el formulario.
4. El servidor al no considerar que la entrada es código malicioso, sino solo texto, lo envía dentro de la consulta sin filtro de *script*.
5. El *script* es ejecutado en la sesión del cliente, el cual cree que es parte de la página web y puede aceptar el pedido para avanzar a la aplicación. Estos mensajes pueden ser, por ejemplo: “Su celular podría estar en riesgo: ¿Desea Instalar el Antivirus?”.
6. La víctima ante el mensaje informativo puede aceptar para que se desaparezca dicho mensaje. Sin saber que ahora su sesión ha sido robada y sus datos corren peligro. A continuación, se muestra en la figura el diagrama de ataque paso a paso.

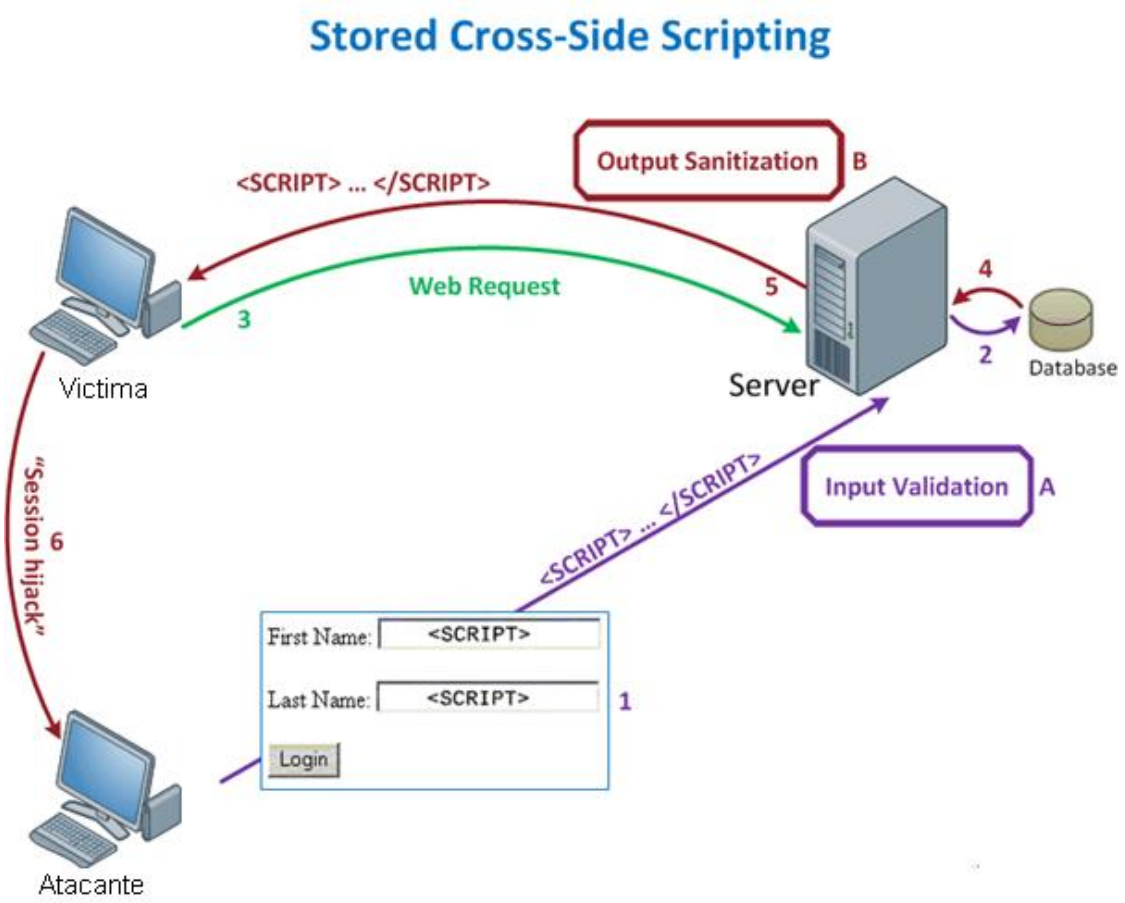

Figura 22. XSS Stored Scripting.
Adaptado de (Neo Kobo, 2014).

Extracto de Script Por Usar:

```
<script>alert(document.cookie)</script>
```

Figura 23. Script de *XSS Stored Scripting*.
Adaptado de (Nadji, Saxena, y Dawn, 2009).

Explicación de Script: Se guarda un script en la base de datos, permitiendo que el atacante pueda ejecutar comandos remotos y de esta manera obtener información de usuarios que requieren peticiones web al CMS.

Evidencia:

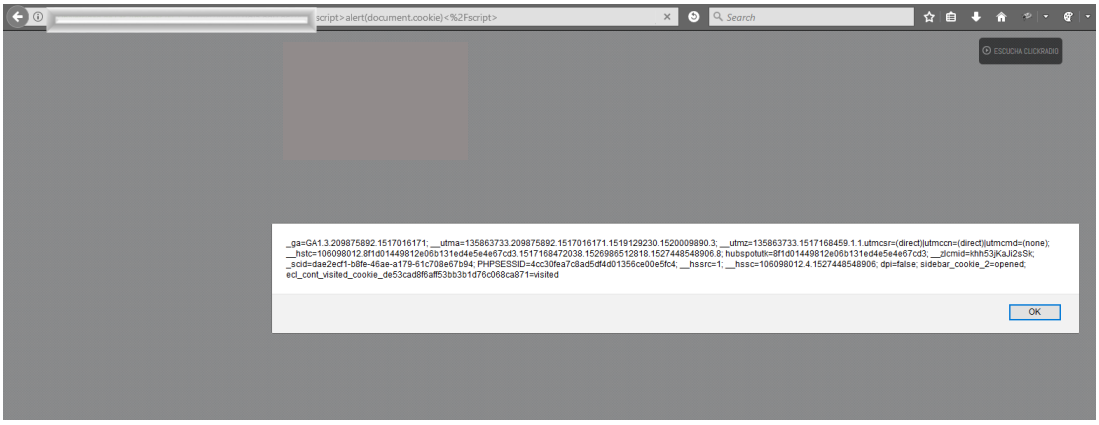

Figura 24. Evidencia de *XSS Stored Scripting*.

```
<div class="input" data-reactid=".hbspt-forms-0.0:$4.$mobilephone_1">
    <input id="mobilephone_1-6d0daeb6-ac26-44e0-9b86-f7b2440a4f01" class="hs-input" name
    reactid=".hbspt-forms-0.0:$4.$mobilephone_1.0" aria-required="true" autocomplete="of
  <script>
      1  alert(document.cookie)
    </script>
```

Figura 25. Extracto de *Script* introducido en un formulario.

Resultado: El sitio es teóricamente vulnerable a Stored Cross-Side Scripting, por lo que atacantes malintencionados pueden guardar archivos en el Payload del CMS, comprometiendo la seguridad de los usuarios y de la página web.

Impacto negativo para el negocio: Alto, debido a que se puede hacer phishing en la página web.

4.1.3. Testing for HTTP Verb Tampering (OTG-INPVAL-003)

¿Qué es?: Inclusión de métodos de petición distintos a las peticiones estándar *GET* y *POST* en un servidor web, mediante el uso de cambios en las tramas de los paquetes de peticiones *GET* y *POST*, utilizados en los servidores web.

Objetivo: Verificar si la aplicación requiere de métodos alternativos como *HEAD* u *OPTIONS*, estos métodos alternativos pueden activar acciones sin la autenticación adecuada o revelar información sensible, sobre el contenido de la aplicación web y su funcionamiento.

Herramienta: Consola Linux, *script* para uso de *netcat package*.

Diagrama de Ataque: El diagrama de ataque es sencillo y se basa en cambiar el tipo de petición en la cabecera de un paquete *GET* o *POST* y cambiarlo por *PUT*, *OPTIONS*, etc. Debido a la falta de seguridad, el servidor cree que es un método *GET* normal y envía información sensible al cliente.

Evidencia:

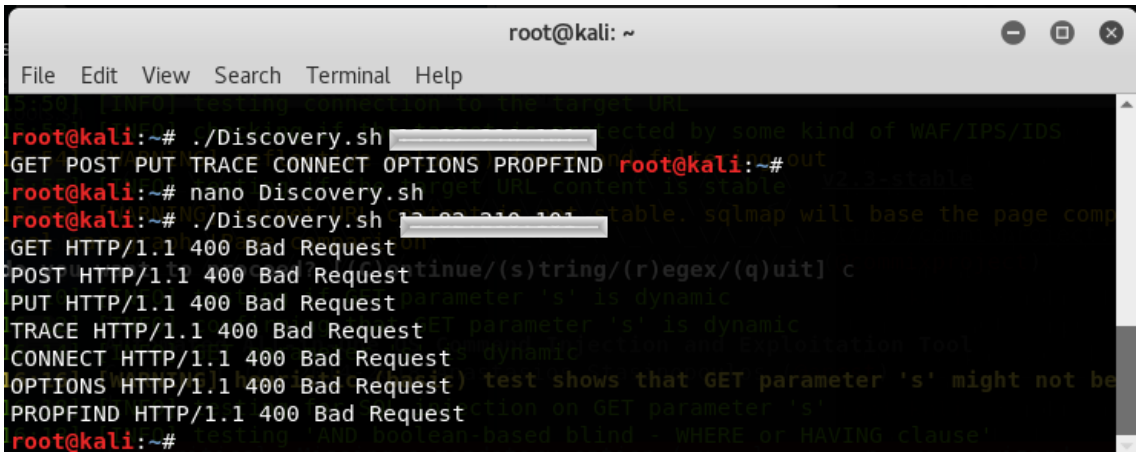

Figura 26. Script para validación de Métodos *"verb".*

Extracto de Script Por Usar:

```
1. #!/bin/bash
2. for webservmethod in GET POST PUT TRACE CONNECT OPTIONS PROPFIND;
3. do
   printf "$webservmethod ";
   printf "$webservmethod / HTTP/1.1\nHost: $1\n\n" | nc -q 1 $1 80 | grep "HTTP/1.1"
4. done
```

Figura 27. Extracto del *Script* usado.

Resultado: El sitio web no es propenso a este tipo de ataques. El CMS de WordPress se maneja por un archivo index.php, este ataque es recurrente en sitios web que no son manejados por CMS y tienen una estructura con index.html.

Impacto negativo para el negocio: Ninguno.

4.1.4. Testing for HTTP Parameter pollution (OTG-INPVAL-004)

¿Qué es?: Un tipo de ataque que se basa en proporcionar varios parámetros en la cabecera de HTTP, esperando que la aplicación web interprete erróneamente los valores y suceda un comportamiento imprevisto aprovechado por los atacantes.

Objetivo: Aprovechar dichos comportamientos, para ser capaz de pasar por alto validaciones de entrada, desencadenar errores en la aplicación web y modificar valores de variables internas, por ejemplo, contaminación de parámetros HTTP, además de permitir ataques del lado del servidor y del cliente.

Herramienta: OWASP ZAP.

Diagrama de Ataque:

1. Enviar una solicitud HTTP que contenga el nombre y valor del parámetro estándar, y registrar la respuesta HTTP. https://www.aaa.com.ec/? par1 = val1.
2. Se reemplaza el valor del parámetro con un valor alterado. https://www.aaa.com.ec/? par1 = HPP_TEST1.
3. Se envía una nueva solicitud combinando los pasos (1) y (2). https://www.aaa.com.ec/par1 = val1 & par1 = HPP_TEST1.
4. Se comparan las respuestas obtenidas durante todos los pasos anteriores. Si la respuesta de (3) es diferente de (1) y a su vez (3) también es diferente de (2), existe una discrepancia de impedancia que puede eventualmente abusarse para desencadenar vulnerabilidades de *HPP*.

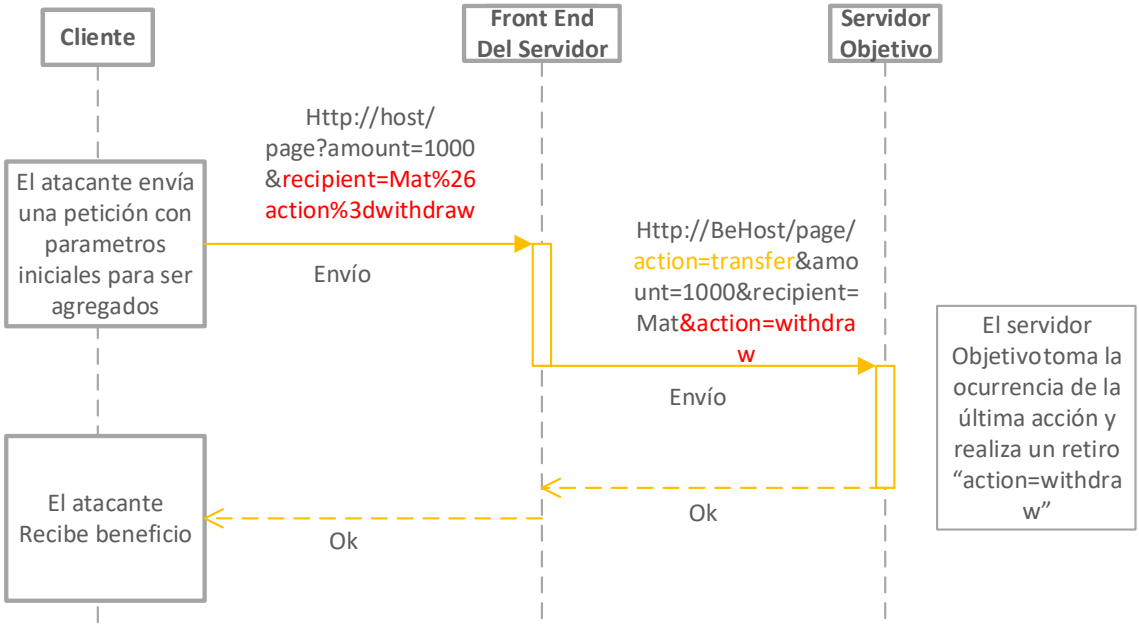

Figura 28. Diagrama de Ataque de *HTTP Parameter pollution.* Adaptado de *(Carettoni, 2009)*.

Resultados: OWASP ZAP no se encuentran resultados de este tipo, debido a que es un CMS el que tiene control total sobre este tipo de ataques.

Impacto negativo para el negocio: Ninguno.

4.1.5. Testing for SQL Injection (OTG-INPVAL-005)

¿Qué es?: Un ataque de inyección SQL, consiste en la inserción o "inyección" de una consulta SQL parcial o completa a través de la entrada de datos, transmitida desde el cliente (navegador) a la aplicación web.

Objetivo: Leer datos confidenciales de la base de datos, modificar datos de la base de datos (insertar/actualizar/eliminar), ejecutar operaciones de administración en la base de datos (como apagar el DBMS), recuperar el contenido de un archivo en el sistema de archivos del DBMS.

Herramienta: Sqlmap, herramienta de pruebas de penetración de código abierto que automatiza el proceso de detección y explotación de fallas de inyección SQL en servidores de bases de datos.

Diagrama de Ataque:

1. El atacante envía una consulta a la aplicación web, con código SQL enmascarado, de tal manera que la aplicación no lo detecte y pueda ser ejecutado en la base de datos.
2. La base de datos envía la respuesta de la consulta, debido a que se ejecuta en un contexto “seguro”, sin embargo, al no ser controlada la excepción puede mostrar información confidencial.
3. El atacante obtiene información sensible de la base de datos.

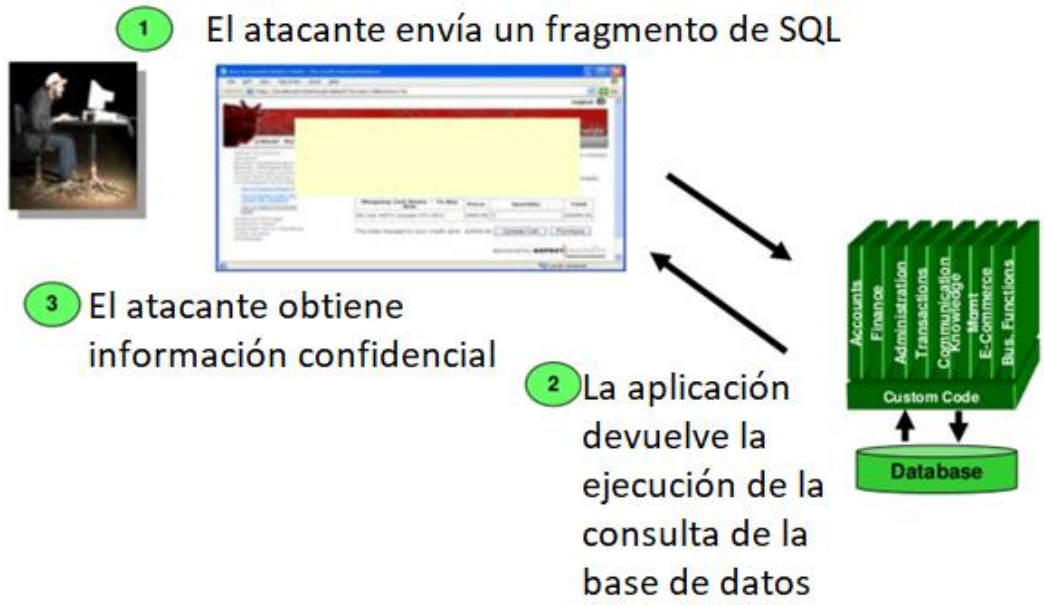

Figura 29. Diagrama de Ataque de *SQL Injection*.
Adaptado de *(Babulak, 2015)*.

Evidencia:

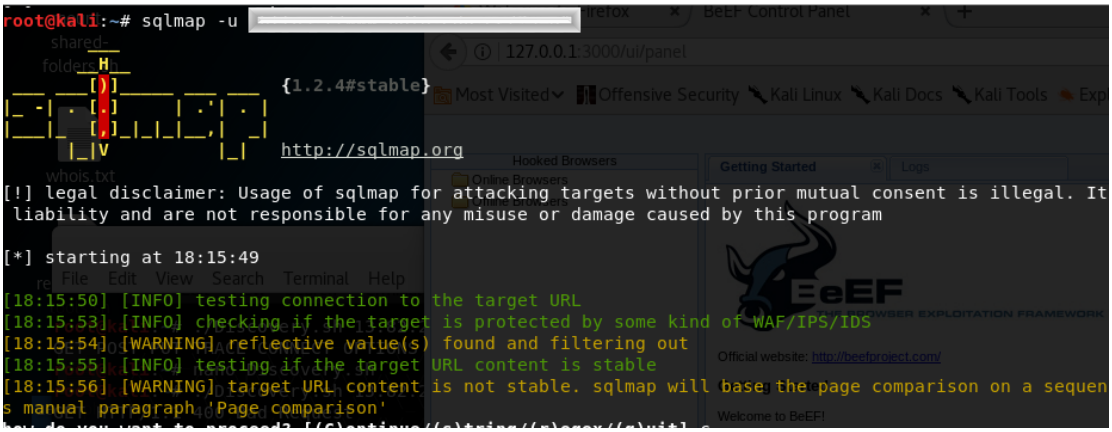

Figura 30. Evidencia de uso *SQLMap* para *SQLInjection.*

Enlace: http://sqlmap.org/ multiplataforma escrita en Python.

Requisitos: Sqlmap se encuentra instalado por defecto en Kali Linux.

Uso: Por medio de comandos; sqlmap - u “[host]” se genera toda la información encontrada en la consola. Ejemplo: sqlmap - u “https://www.abc.com/?s=a”.

Script Por Usar: Anexo 6.

Explicación de Script: SQLMap envía los parámetros de búsqueda por medio de scripts, después intenta buscar la versión de base de datos y por último trata de utilizar sentencias SQL.

Resultados: El sitio no es vulnerable a inyección SQL, se encuentra un enlace que cumple con las características de inyección SQL, pero esto no es suficiente para realizar un ataque de SQL Injection. Se usa las opciones adicionales del asistente para recabar más información sobre links vulnerables, sin encontrar ningún resultado relevante.

4.1.7. Testing for Code Injection (OTG-INPAL-012)

¿Qué es?: Ejecución de código como entrada para ser procesado en la aplicación web y ejecutado como código dinámico o como archivo incluido por el servidor web. Los tipos de Inyección de Código más comunes son:

- Inclusión Local de Archivos, proceso de inclusión de archivos desde la sesión de un usuario y su ejecución desde dentro de la aplicación.
- Inclusión Remota de Archivos, proceso de inclusión de archivos por medios remotos, se produce cuando un sitio web recibe como entrada la ruta al archivo que debe incluirse, lo que permite inyectar una URL externa.

Objetivo: Verificar la posibilidad de tener acceso a la aplicación mediante el uso de Fuerza Bruta, para verificar la inclusión de código malicioso y su consecuente ejecución.

Herramienta: Beef, wpscan, Navegador Web.

Diagrama de Ataque:

1. El atacante verifica que se pueda cargar scripts en formularios poco validados, pudiendo de esta manera cargar *scripts* para que sean ejecutados desde el servidor.
2. Estos *scripts* al ser ejecutados en el servidor pueden crear un "*backdoor shell*" el cual es una ventana de comandos que permiten al atacante tener acceso sobre el DBMS.
3. Una vez que el atacante haya creado dicho "*backdoor shell*" se puede realizar remotamente el cambio de todas las páginas web, usar el servidor como *bot DDoS* o extraer toda la información del sitio web.

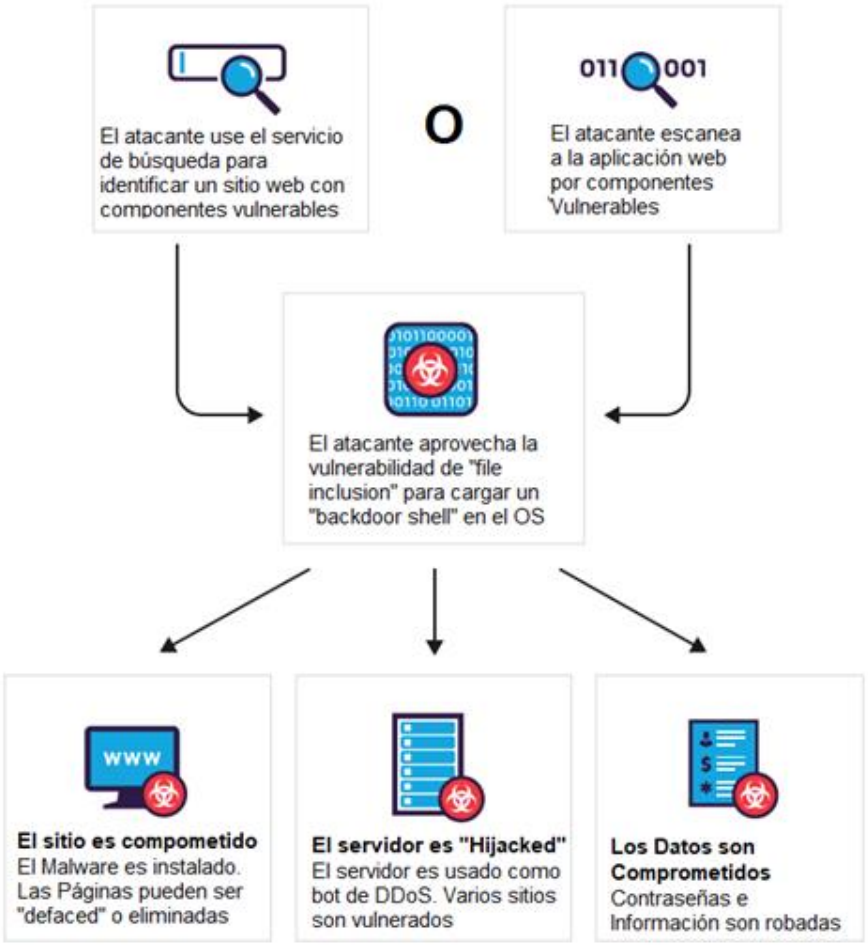

Figura 31. Diagrama de ataque de *Remote File Inclusion*
Adaptado de *(Imperva, 2018)*.

Resultado: En las pruebas realizadas a la aplicación web, no se pudo ejecutar ningún *script,* debido a la limitación de acceso al servidor, aunque teóricamente existe la posibilidad de este ataque por XSS, es necesario determinar con registros del CMS en la Base de Datos.

4.1.8. Testing for HTTP Splitting/Smuggling (OTG-INPVAL-016)

¿Qué es?: Es un tipo de ataque que permite a un intruso insertar caracteres CR (*ENTER*=Retrocede cursor al inicio de la línea actual) y LF (Avance de línea) en las cabeceras de la respuesta de la aplicación y hacer "*Split*" de esa respuesta en dos mensajes HTTP diferentes, este tipo de ataque requiere cierto nivel de

conocimiento sobre los diferentes agentes que manejan los mensajes HTTP (servidor web, *proxy*, cortafuegos).

Objetivo: Envenenamiento de la *caché* hasta realizar *Cross Site scripting*, análisis de mensajes HTTP especialmente diseñados para ser ejecutados por líneas de código e interpretación simultánea del navegador web, verificando la probabilidad de "*Smuggling HTTP*".

Herramienta: Navegador Web.

Diagrama de Ataque:

1. El atacante envía una petición *GET* al servidor web de la aplicación.
2. Pasar a las seguridades del *Proxy*.
3. El Servidor Web atiende la petición como si esta fuera única.
4. Envía una respuesta parcial, mientras espera a enviar la otra sección.
5. El atacante recibe la primera petición, a continuación, envía otra petición del contenido bienvenido.html.
6. Pasa el proxy.
7. y 8 Este se adjunta a la respuesta parcial y envía adjunto otro recurso "contenido_prohibido.html". Debido a que la petición de contenido_prohibido.html ahora está completa se adjunta al bienvenido.html y este es devuelto.

9. De esta manera cada vez que se haga una petición a bienvenido.html, se adjuntará contenido_prohibido.html, por envenenamiento de caché.

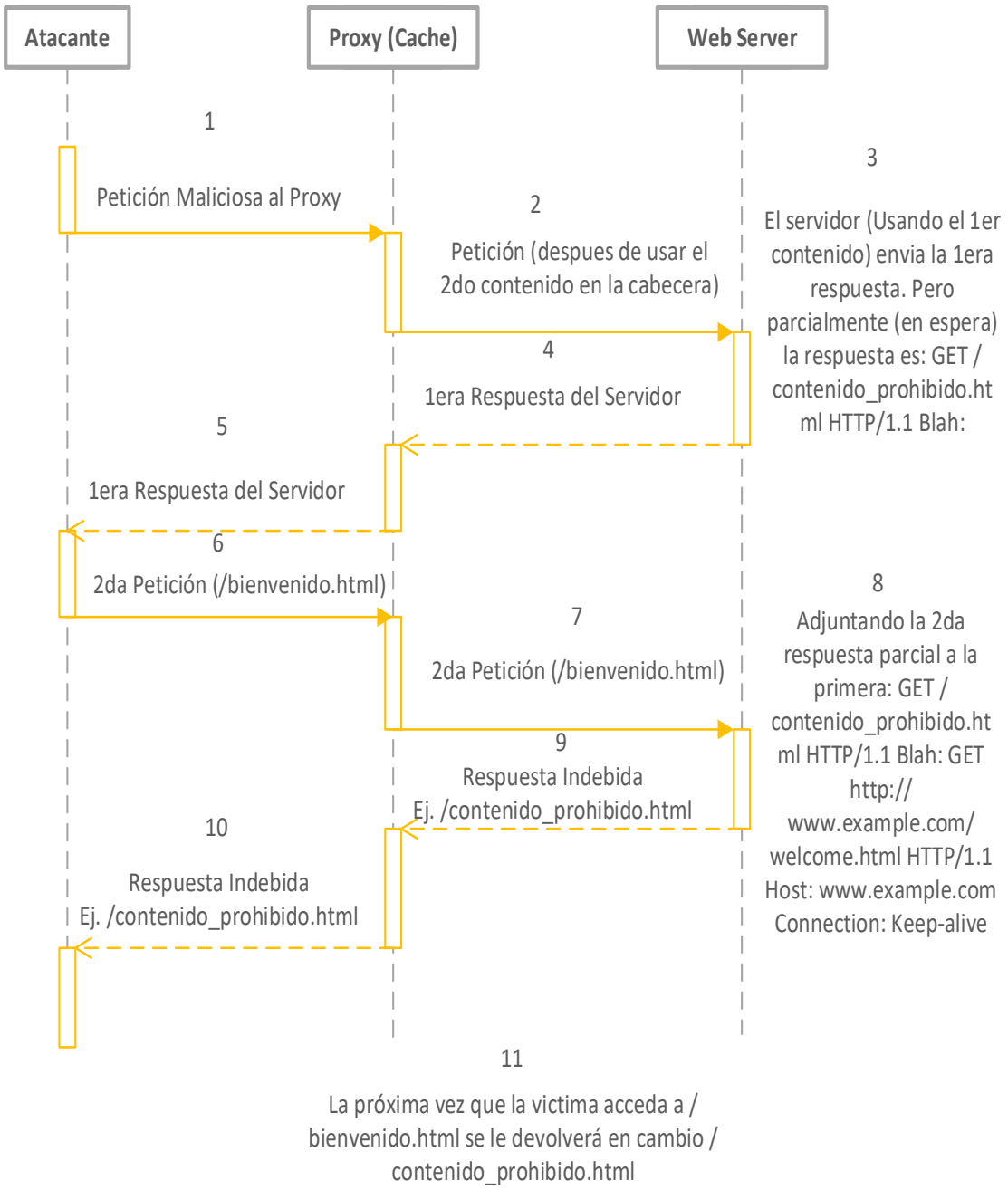

Figura 32. Diagrama de Ataque de *Web-cache poisoning*. Adaptado de *(Open Source, 2011)*.

Script Por Usar:

```
1.  ?name=Arupha%0d%0a%0d%0a<script>alert(document.domain)</script>
```

Figura 33. Extracto de *Script* para Http *Splitting.*

Evidencia:

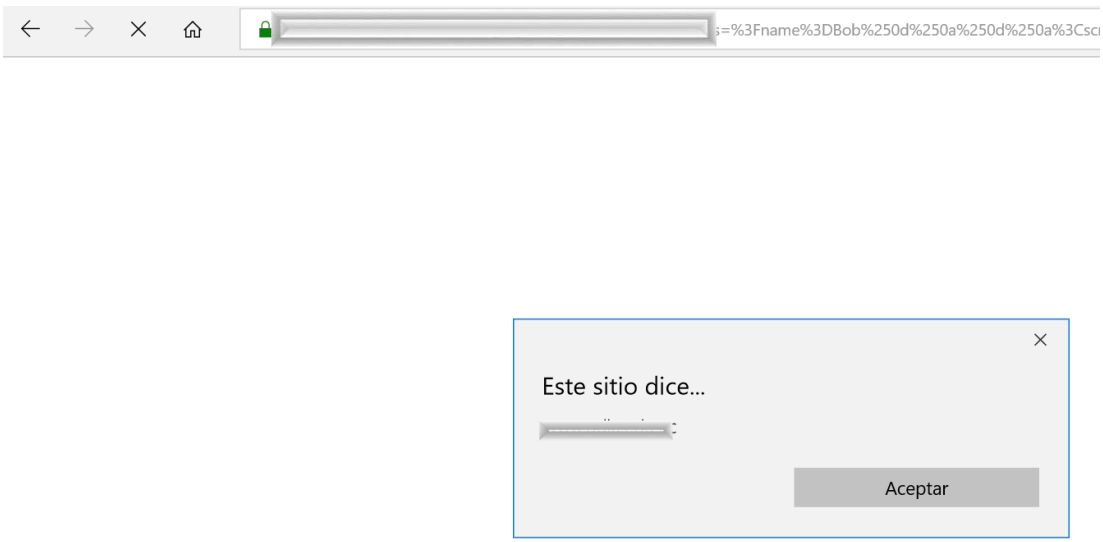

Figura 34. Uso de *HTTP Splitting* en la página web.

Resultado: La aplicación web muestra la búsqueda y ejecuta el *script* por separado, se puede realizar este tipo de ataque para desencadenar varios scripts parciales a la vez que se adjuntan a la petición de la página HTML.

Como se puede apreciar, existe alta posibilidad de que un atacante pueda utilizar las vulnerabilidades encontradas en el sitio web, debido a la falta de validación o “sanitización” en los caracteres que se envían en búsquedas y en formularios, por lo que es recomendable añadir reglas de validación de caracteres, para evitar el robo de información sensible o confidencial.

4.2. Test Validados y no realizados

Las siguientes pruebas no se realizan, debido a diversos motivos, entre ellos la arquitectura de la aplicación web, el lenguaje que maneja el CMS, entre otros. Por lo que se muestra en la tabla su descripción respectiva.

Tabla 9.

Test no realizados de Data Validation.

Prueba	Motivo
Testing for Incubated vulnerabilities	Se puede llegar a tener vulnerabilidades incubadas por vulnerabilidades XSS, debido a la complejidad y duración de las pruebas no se realizan.
Testing for Oracle, MySQL, SQL Server, PostreSQL, MS Access, NoSQL	El sitio no es propenso a SQL Injection.
Testing for LDAP injection	El sitio no usa una estructura LDAP.
Testing for ORM injection	El sitio no usa una estructura ORM.
Testing for XML Injection	El sitio no guarda ningún tipo de dato con estructura XML.
Testing for SSI Injection	Las directivas solo pueden ser ejecutadas en archivos específicos los cuales tienen extensión .HTML, este sitio en particular maneja archivos .php.
Testing for XPath Injection	El sitio no guarda datos con estructura XML ni contiene una Base de datos XML.
IMAP/SMTP Injection	El sitio no usa las capacidades *IMAP/SMTP*, o se conecta a un servicio externo que maneja mail.
Testing for Command Injection	El CMS controla de manera correcta la ejecución de comandos PHP dentro de links.
Testing for Buffer Overflow	El aplicativo web está basado en *PHP* y estructuras como *GOT*, .dtors o TEB están basadas en sitios web realizados en C/C++, se necesita tener acceso de administrador para ejecutar código a nivel de Sistema Operativo.
Testing for HTTP Incoming Requests	Esta prueba se realiza cuando aún se encuentra en desarrollo la aplicación.

4.3. Resultados de *Data Test Validation*

A continuación, se presenta una tabla resumen del capítulo con las pruebas realizadas y validadas:

Tabla 10.
Resultados de Input Validation Testing.

OTG-INPVAL	Herramienta	Resultado
001	-Navegador Web -Barra de Búsqueda	El sitio es vulnerable a *Reflected XSS*, por lo que puede ser posible un ataque como robo de cookies o la sesión de usuarios e información de ingreso del CMS.
002	-Navegador Web -Beef	El sitio es teóricamente vulnerable a Stored Cross-Side Scripting, por lo que atacantes malintencionados pueden guardar archivos en el Payload del CMS, comprometiendo la seguridad de los usuarios y de la página web.
003	-Consola Linux -*Script* Bash con Netcat	El sitio web no es propenso a este tipo de ataques. El CMS de WordPress se maneja por un archivo index.php, este ataque es recurrente en sitios web que no son manejados por CMS y tienen una estructura con index.html.
004	-OWASP Zap	OWASP ZAP no se encuentran resultados de este tipo, debido a que es un CMS el que tiene control total sobre este tipo de ataques.
005	-SQLMap	El sitio no es vulnerable a inyección SQL, se encuentra un enlace que cumple con las características de inyección SQL, pero esto no es suficiente para realizar un ataque de

OTG-INPVAL	Herramienta	Resultado
		SQL Injection. Se usa las opciones adicionales del asistente para recabar más información sobre links vulnerables, sin encontrar ningún resultado relevante.
016	-Navegador Web	La aplicación web muestra la búsqueda y ejecuta el *script* por separado, se puede realizar este tipo de ataque para desencadenar varios scripts parciales a la vez que se adjuntan a la petición de la página HTML.

5. Client Side Testing

En el presente capítulo se realizan pruebas del lado del cliente, las cuales se refieren a la ejecución del código en sesión del cliente de forma nativa dentro de un navegador web y con complementos del navegador. La ejecución del código en el lado del cliente se realiza para analizar la facilidad de ejecución de código malicioso y su afectación en la aplicación web.

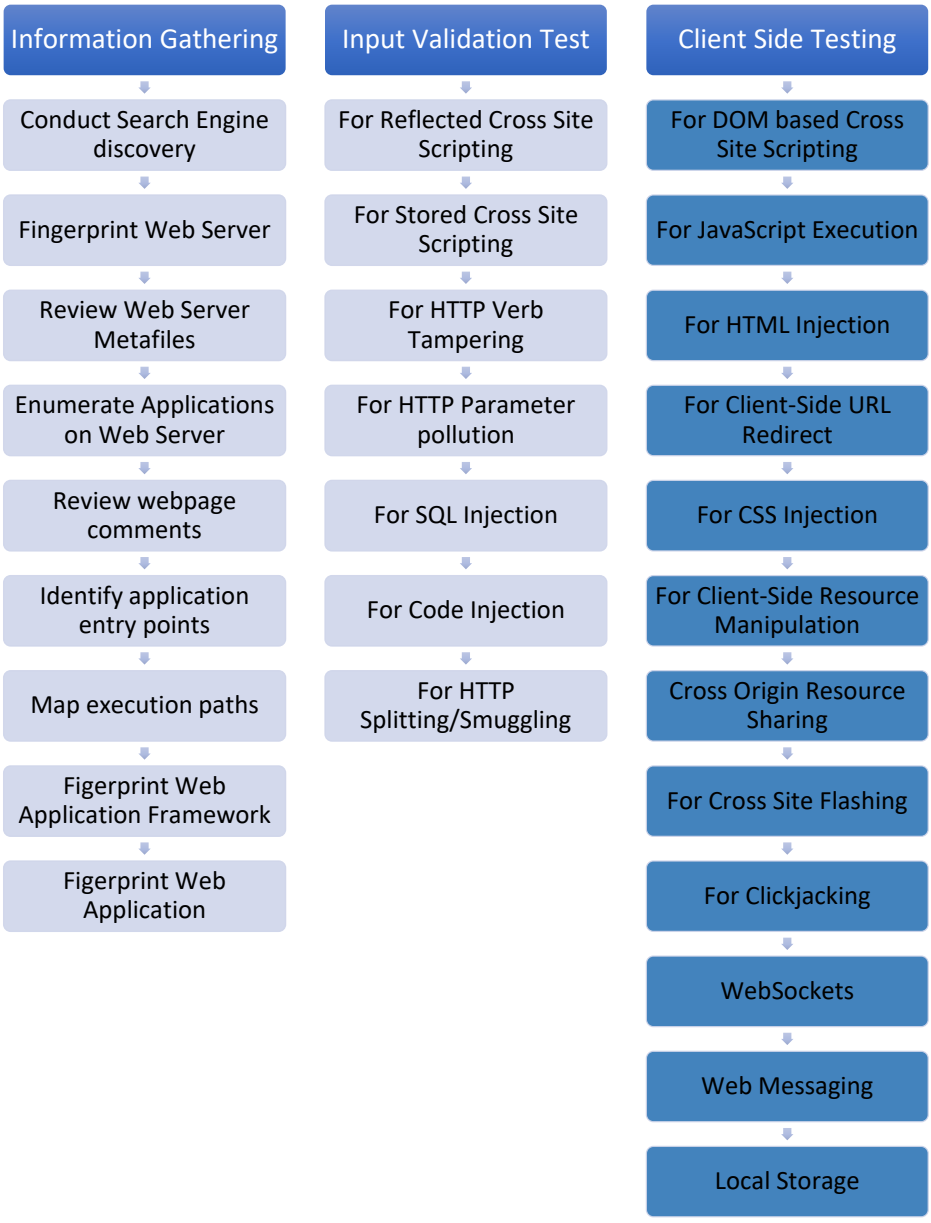

Figura 35. Diagrama de Pruebas de *Client Side Testing* de OWASP.

5.1. Testing for DOM based Cross Site Scripting (OTG-CLIENT-001)

¿Qué es?: Es una prueba basada en DOM en busca de contenido activo del lado del navegador en una página, típicamente JavaScript, obteniendo la entrada del usuario y luego haciendo algo inseguro con él.

Objetivo: Verificar el ataque al añadir #<script>alert('xss')</script> a la URL de la página afectada esperando que se muestre el cuadro de alerta.

Herramienta: Navegador Web.

Explicación de *Script*: En el *script* ejecuta una alerta el cual entrega la cookie de la sesión actual.

Evidencia:

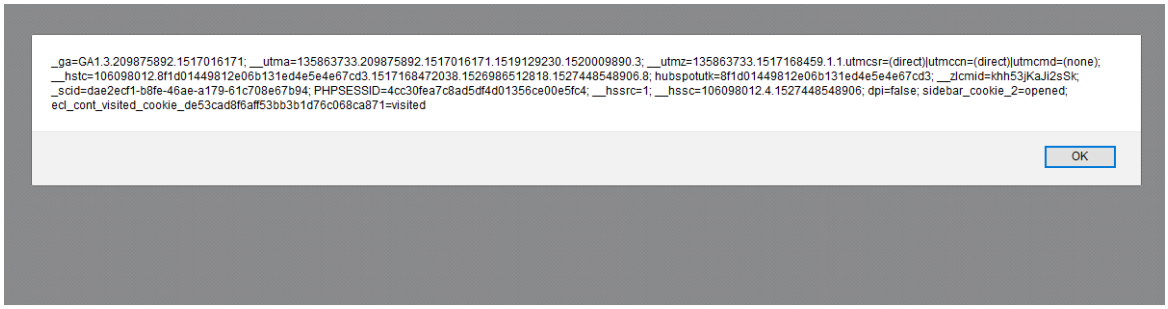

Figura 36. DOM Base Cross Site Scripting.

Resultado: Es posible realizar ejecuciones de *scripts* en el sitio debido a que no se hace un filtrado adecuado.

5.2. Testing for JavaScript Execution (OTG-CLIENT-002)

¿Qué es?: Es un subtipo de *Cross Site Scripting* (XSS) que implica la capacidad de inyectar código JavaScript arbitrario, ejecutarlo en la aplicación y dentro del navegador de la víctima. Esta vulnerabilidad puede tener muchas

consecuencias, como la revelación de cookies de sesión de usuario las cuales pueden utilizarse para hacerse pasar por la víctima, o, en general, puede permitir que el atacante modifique el contenido de la página y el comportamiento de la aplicación.

Objetivo: Vulnerar la aplicación cuando esta carece de una validación de entrada y salida adecuada suministrada por el usuario.

Herramienta: Navegador Web.

Explicación de Script:

En el *script* se ejecuta una alerta el cual nos entrega la *cookie* de la sesión actual.

Resultado: Es posible realizar ejecuciones de *scripts* en el sitio, con métodos de *deface* en la ejecución del código *JavaScript* debido a que no se realiza un saneamiento adecuado.

5.3. Testing for HTML Injection (OTG-CLIENT-003)

¿Qué es?: La inyección HTML es un tipo de problema de inyección que ocurre cuando un intruso es capaz de controlar un punto de entrada y es capaz de inyectar código HTML arbitrario en una página web vulnerable.

Objetivo: Vulnerar la aplicación cuando esta carece de una validación de entrada y salida adecuada suministrada por el usuario.

Herramienta: Navegador Web.

Ejemplo de Script por Usar:

```
1.  < img % 20 src = 'aaa' % 20 onerror = alert(1) >
```

Figura 37. Extracto de Script para HTML Injection.

Explicación de Script:

Este *script* añade una URL a la página con una etiqueta de imagen que ejecutará un código JavaScript arbitrario insertado por el intruso en el contexto HTML.

Resultado:

El CMS transforma correctamente los tags HTML lo cual hace imposible que se puedan correr esta clase de *scripts* en la URL del sitio.

5.4. Testing for Client-Side URL Redirect (OTG-CLIENT-004)

¿Qué es?: Falta de validación de entradas que existe cuando una aplicación acepta una entrada controlada por el usuario, que especifica un enlace que conduce a una URL externa que podría ser maliciosa. Este tipo de vulnerabilidad puede utilizarse para realizar un ataque de *phishing* o redirigir a una víctima a una página de infección.

Objetivo: Modificar la posibilidad de entrada de URL redireccionando a un sitio malicioso, con lo cual un atacante puede lanzar con éxito una estafa de *phishing* y robar credenciales de usuario. Dado que la redirección es originada por la aplicación real, los intentos de *phishing* pueden tener una apariencia más confiable.

Herramienta: Navegador Web.

Ejemplo de Script por Usar:

```
1.  http: //www.target.site?#redirect=www.fake-target.site
```

Figura 38. Extracto del *Script de Client-Side URL Redirect.* Fuente: Elaboración Propia.

Explicación de Script: El aplicativo no valida la URL haciendo que se haga una redirección hacia otro sitio.

Resultado: El manejo de URL por parte del CMS se realiza de manera correcta por lo cual esta vulnerabilidad no puede ser aplicada.

5.5. Testing for CSS Injection (OTG-CLIENT-005)

¿Qué es?: Una vulnerabilidad de inyección de CSS implica la capacidad de inyectar código CSS arbitrario en el contexto de un sitio web de confianza, y esto renderizarlo dentro del navegador de la víctima.

Objetivo: Vulnerar la aplicación cuando esta carece de una validación de entrada y salida adecuada suministrada por el usuario.

Herramienta: Navegador Web.

Ejemplo de Script por Usar:

```
1.  Target.site/#red;-o-link:'javascript: alert (1)';-o-link-source: current;
```

Figura 39. Extracto de *script* para *CSS Injection*.

Explicación de Script:

El *Script* Inyecta en la URL un estilo CSS que se desea ejecutar el cual dentro contiene un *script*.

Resultado:

El CMS transforma correctamente los tags CSS con lo cual no es posible inyectar este tipo de ataques en la URL del sitio.

5.6. Testing for Client-Side Resource Manipulation (OTG-CLIENT-006)

¿Qué es?: Prueba de lado del cliente que ocurre cuando una aplicación acepta una entrada controlada por el usuario esta especifica la ruta de un recurso (por ejemplo, la fuente de un *iframe*, *js*, *applet* o el manejador de una XMLHttpRequest).

Objetivo: Modificar la entrada de URL redireccionando a un recurso que se encuentre en una página web distinta.

Herramienta: Navegador Web.

Ejemplo de Script por Usar:

```
1.  www.victim.com/#http://something.com/js.js
```

Figura 40. Ejemplo de *Script* para *Resource Manipulation*.

Explicación de Script: El *script* esta redireccionando a un URL que contiene un *js* el cual puede ejecutar cualquier código.

Resultado: Para que esta vulnerabilidad funcione es necesario que el servidor victima sea propensa a la redirección por URL, en este caso no lo es.

5.7. Test Cross Origin Resource Sharing (OTG-CLIENT-007)

¿Qué es?: Permiso a un navegador web para realizar peticiones "entre dominios" utilizando la API XMLHttpRequest L2 de forma controlada.

Objetivo: Vulnerar las peticiones del encabezado Origen el cual siempre es enviado por el navegador en una solicitud CORS e indica el origen de la solicitud.

Herramienta: OWASP ZAP.

Resultado: El CMS no es vulnerable a este ataque, el mismo se evita mediante el uso del *.htaccess* específicamente haciendo uso de la política *Access-Control-Allow-Origin* en donde se especifica que solo el dominio donde se encuentra la instalación recibirá las peticiones.

5.8. Testing for Cross Site Flashing (OTG-CLIENT-008)

¿Qué es?: Vulnerabilidad en aplicaciones *Flash* defectuosas que están incrustadas en los navegadores, relacionado con la vulnerabilidad de *Cross-Site Scripting* (XSS) basado en DOM.

Objetivo: Vulnerar contenido *Flash* en el sitio para poder escalar la superficie de ataque.

Herramienta: Navegador Web.

Resultado: El sitio no tiene contenido creado en *Flash*.

5.9. Testing for Clickjacking (OTG-CLIENT-009)

¿Qué es?: Un ataque *Clickjacking* utiliza características aparentemente inofensivas de HTML y *JavaScript* para forzar a la víctima a realizar acciones no deseadas, como hacer clic en un botón con funciones ocultas. Se trata de un problema de seguridad del "lado del cliente" que afecta a una gran variedad de navegadores y plataformas.

Objetivo: Renderizar la página web objetivo dentro de un *iframe* haciendo que el atacante pueda tomar control de la página externamente.

Herramienta: Navegador Web.

Diagrama de ataque:

1. El atacante renderiza la página web falsificada con un *iframe* en un servidor diferente. Esto quiere decir que la página web se parece a la original, pero en el fondo ejecuta código diferente.
2. Se envía el enlace del sitio a una víctima.
3. Dentro del *iframe* se ejecutan diferentes *scripts* maliciosos.

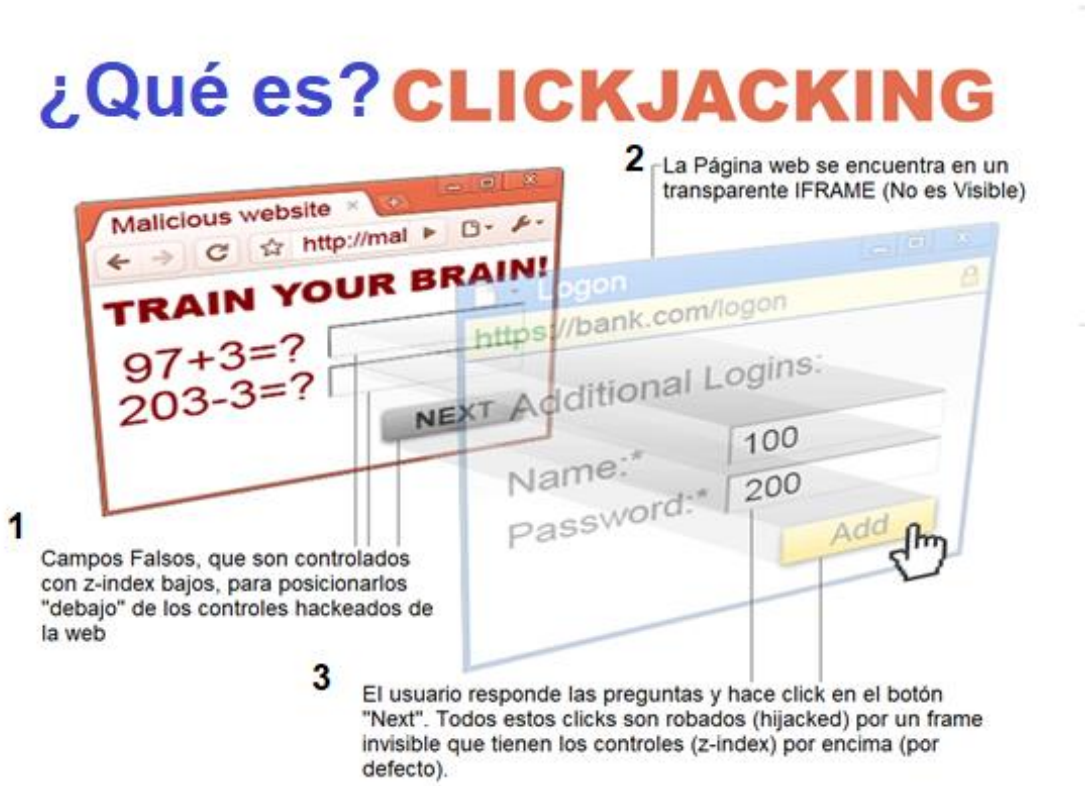

Figura 41. Diagrama de Ataque de ClickJacking.
Tomado de Tiwari, P.

Extracto de Script por Usar:

```
1.  < html >
2.  < head >
3.  <title > Clickjack test page < /title>
4.  < /head>
5.  < body >
6.  < p > Website is vulnerable to clickjacking! < /p>
7.  < iframe src = "https://[redacted]"  width = "500"  height = "500" >< /iframe>
8.  < /body>
9.  < /html>
```

Figura 42. Extracto de *Script* Por Usar.

Explicación de Script:

Se crea un archivo simple con HTML donde se intentará renderizar la página web de la víctima en un *iframe*.

Evidencia:

Figura 43. Clickjacking de aplicación web.

Resultado: El sitio no es vulnerable a este tipo de ataque ya que no permite su renderizado dentro del *iframe*.

5.10. Testing WebSockets (OTG-CLIENT-010)

¿Qué es?: Es responsabilidad del servidor verificar el encabezado Origen en el *handshake* inicial del HTTP *WebSocket*. Si el servidor no valida el encabezado de origen en el *handshake* inicial del *WebSocket*, el servidor *WebSocket* puede aceptar conexiones desde cualquier origen.

Objetivo: Buscar si el aplicativo usa *websockets* y encontrar vulnerabilidades del tipo CSFR.

Herramienta: OWASP ZAP.

Resultado: El aplicativo Web no tiene *websockets*.

5.11. Test Web Messaging (OTG-CLIENT-011)

¿Qué es?: Antes de la introducción de la mensajería web, la comunicación de diferentes orígenes (entre *iframes*, pestañas y ventanas) estaba restringida por la misma política de origen e impuesta por el navegador, sin embargo, los desarrolladores utilizaban múltiples hacks para llevar a cabo estas tareas, la mayoría de ellas eran inseguras.

Objetivo: Descubrir cómo se implementa la Mensajería Web en la aplicación, así como la restricción de mensajes de dominios no confiables y cómo se manejan datos incluso de dominios confiables.

Herramienta: OWASP ZAP.

Resultado: La aplicación web no utiliza *Web Messaging.*

5.12. Test Local Storage (OTG-CLIENT-012)

¿Qué es?: El Almacenamiento Local también conocido como Almacenamiento Web o Almacenamiento sin conexión es un mecanismo para almacenar datos

como pares clave/valor vinculados a un dominio y aplicados a la misma política de origen (SOP). Se guardan dos valores uno en el *localStorage* que es persistente y está pensado para sobrevivir al reinicio del navegador/sistema y el *sessionStorage* que es temporal y sólo existirá hasta que se cierre la ventana o pestaña.

Objetivo: Introducir código inseguro, de manera *client-side* en el *localStorage* del navegador.

Herramienta: OWASP ZAP.

Resultado: OWASP ZAP no arrojo ningún resultado sobre la posibilidad de este ataque por lo que el CMS está seguro sobre el efecto de estos ataques.

5.13. Resultados de Client Side Testing

A continuación, se presenta una tabla resumen del capítulo.

Tabla 11.
Resultados de Client Side Testing.

OTG-CLIENT	Herramienta	Resultado
001	Navegador Web	Es posible realizar ejecuciones de *scripts* en el sitio debido a que no se hace un filtrado adecuado.
002	Navegador Web	Es posible realizar ejecuciones de *scripts* en el sitio debido a que no se realiza un saneamiento adecuado.
003	Navegador Web	El CMS transforma correctamente los tags HTML lo cual hace imposible que se puedan correr esta clase de *scripts* en la URL del sitio.

OTG-CLIENT	Herramienta	Resultado
004	Navegador Web	El manejo de URL por parte del CMS se realiza de manera correcta por lo cual esta vulnerabilidad no puede ser aplicada.
005	Navegador Web	El CMS transforma correctamente los tags CSS con lo cual no es posible inyectar este tipo de ataques en la URL del sitio.
006	Navegador Web	Para que esta vulnerabilidad funcione es necesario que el servidor victima sea propensa a la redirección por URL, en este caso no lo es.
007	OWASP Zap	El CMS no es vulnerable a este ataque, el mismo se evita mediante el uso del .htaccess específicamente haciendo uso de la política *Access-Control-Allow-Origin* en donde se especifica que solo el dominio donde se encuentra la instalación recibirá las peticiones
008	Navegador Web	El sitio no tiene contenido creado en *Flash*.
009	Navegador Web	El sitio no es vulnerable a este tipo de ataque ya que no permite su renderizado dentro del *iframe*.
010	OWASP ZAP	El aplicativo Web no tiene *websockets*.
011	OWASP ZAP	La aplicación web no utiliza *Web Messaging*
012	OWASP ZAP	*OWASP ZAP no arrojo ningún resultado sobre la posibilidad de este ataque por lo que el CMS está seguro sobre el efecto de estos ataques.*

Una vez recabadas estas pruebas, se puede empezar el análisis de las amenazas y vulnerabilidades sobre el negocio, así como la posibilidad de que estos ocurran y el impacto en el mismo.

6. Análisis de Amenazas y Vulnerabilidades

En el presente capítulo se realizan formulaciones respectivas sobre las vulnerabilidades encontradas y se analizan si son un riesgo para la aplicación web y por lo tanto para los usuarios y el negocio. En la Figura 1 se muestra el curso de acciones que deben realizar por cada vulnerabilidad encontrada en los sitios web de las Instituciones de Educación Superior que puedan afectar al negocio.

6.1. Identificación de Riesgos

Estos son las vulnerabilidades encontradas en la aplicación web, los mismos serán puntuados con respecto a la metodología de *OWASP Risk Rating Methodology*, descrita en la Figura 2 y Figura 3.

- Vulnerabilidades Apache 2.2.15(*Shellshock*, *Brute Force Filenames*, *DDOS*).
- Visibilidad de información de PHP.
- *Reflected XSS.*
- *Stored XSS.*

Identificado las vulnerabilidades se realiza la descripción del riesgo en base a los factores y a su incidencia para el negocio.

6.1.1. Riesgo en base a Vulnerabilidades de Apache

Apache es un servidor web HTTP de código abierto utilizado por el 70% de los sitios web del mundo (The Apache Software Foundation, 2018). Muchas de sus vulnerabilidades solo pueden ser aprovechadas por usuarios locales y no remotamente. Sin embargo, se recomienda el uso de la última versión, debido a que los fallos de seguridad generalmente son documentados y publicados. Se presenta el motivo de la puntuación (Common Vulnerabilities and Exposures, 2018).

- Factores de Amenaza: Explotar vulnerabilidades de Apache, requieren de un usuario avanzado que sea capaz de usar *frameworks* especializados para su explotación, al igual que existe la posibilidad de beneficio. El *exploit* de *shellshock* permite el acceso al servidor. Se necesita de un *exploit* secundario para encontrar el script *CGI* necesario para su ejecución. No es necesario el acceso a ningún recurso en específico y lo puede realizar cualquier usuario del internet.
- Factores de Vulnerabilidad: Es fácil descubrir la versión de Apache y sus vulnerabilidades con herramientas automáticas, es difícil explotarlas debido a que generalmente se requiere de *exploits* sofisticados, además se necesita de una búsqueda extensiva de módulos activos en Apache. Estos ataques son rastreados activamente en el aplicativo apareciendo en auditorias.
- Impacto Técnico: La explotación de la vulnerabilidad no tiene mucho efecto sobre la confidencialidad además tiene un efecto mínimo sobre la corrupción de datos, principalmente se puede denegar servicios primarios y dar acceso a bash, pero estos *exploits* crean trazabilidad.
- Impacto de negocios: Si el exploit se ejecuta exitosamente se tiene un efecto menor en el beneficio anual, no afecta a la reputación de la empresa como tal y es una violación menor con un mínimo efecto en la privacidad.

En la figura 44 se muestra un resumen del riesgo sobre la vulnerabilidad de Apache.

Probabilidad								
factores agente de amenaza					Los factores de vulnerabilidad			
Nivel de habilidad	Motivo	Oportunidad	Tamaño		Facilidad de descubrimiento	La facilidad de explotar	Conciencia	La detección de intrusiones
5 - Usuario avanzado del ordenador	4 - Posible recompensa	9 - No requiere acceso o recursos	9 - Los usuarios de Internet anónimos		9 - Herramientas automatizadas disponibles	3 - Difícil	9 - Conocimiento público	1 - Detección activa en la aplicación
probabilidad general: 6.125 ALTO								
Impacto								
Impacto técnico					Impacto de negocios			
Pérdida de confidencialidad	Pérdida de integridad	Pérdida de disponibilidad	Pérdida de trazabilidad del Atacante		Daño financiero	daños a la reputación	Incumplimiento	violación de privacidad
2 - Mínima revelación de datos No Sensibles	3 - Mínimo de datos seriamente corruptos	7 - Servicios primarios extensos interrumpidos	1 - Totalmente trazable		3 - Efecto menor en el beneficio anual	1 - Daño mínimo	2 - Violación menor	3 - Un individuo
En general impacto técnico: 3.250 MEDIO					el impacto global de las empresas: 2.250 BAJO			
Impacto global: 2.750 BAJO								

Figura 44. Resumen de Probabilidad x Impacto en la aplicación web sobre Vulnerabilidades en Apache. Adaptado de *(The OWASP Fundation, 2015)*.

6.1.2. Riesgo en base a Vulnerabilidades de PHP

PHP (acrónimo recursivo de PHP: *Hypertext Preprocessor*) es un lenguaje de código abierto utilizado para el desarrollo web, que puede ser incrustado en HTML ejecutado desde el lado del Servidor (The PHP Group, 2018). Los factores de Impacto encontrados son los siguientes:

- Factores de Amenaza: El entregar información de PHP supone un riesgo a largo plazo especialmente si este no se actualiza continuamente, para ver esta información no es necesario tener conocimientos técnicos, esta información no da un beneficio inmediato a los atacantes. Además, para ver dicha información no es necesario accesos especiales.
- Factores de Vulnerabilidad: Es fácil descubrir la versión de PHP con el uso de herramientas automáticas se descubre el info.php del servidor, la facilidad de explotar vulnerabilidades con esta información es “teórica” debido a que depende de las vulnerabilidades de la versión de PHP, toda la información sobre PHP es de dominio público, se detecta esta intrusión mediante el aplicativo de Apache.
- Impacto Técnico: No tiene un impacto considerable en la confidencialidad, integridad o disponibilidad del sitio, el acceso a esta información es trazable.
- Impacto de negocios: No hay daño financiero con esta vulnerabilidad, o daños a la reputación de esta, aunque se tiene una violación menor por permitir que información confidencial del servidor sea accedida por terceros, esto causa una violación mínima de privacidad.

En la figura 45 se resume la Vulnerabilidad de PHP.

Probabilidad								
factores agente de amenaza					Los factores de vulnerabilidad			
Nivel de habilidad	Motivo	Oportunidad	Tamaño		Facilidad de descubrimiento	La facilidad de explotar	Conciencia	La detección de intrusiones
1 - Sin conocimientos técnicos	1 - Baja o ninguna recompensa	9 - No requiere acceso o recursos	9 - Los usuarios de Internet anónimos		9 - Herramientas automatizadas disponibles	1 - Teórico	9 - Conocimiento público	1 - Detección activa en la aplicación
probabilidad general: 5.000 MEDIO								
Impacto								
Impacto técnico					Impacto de negocios			
Pérdida de confidencialidad	Pérdida de integridad	Pérdida de disponibilidad	Pérdida de trazabilidad del Atacante		Daño financiero	daños a la reputación	Incumplimiento	violación de privacidad
2 - Mínima revelación de datos No Sensibles	1 - Mínimo de datos ligeramente corruptos	1 - Servicios secundarios mínimos interrumpidos	1 - Totalmente trazable		1 - Menos que el costo de arreglar la vulnerabilidad	1 - Daño mínimo	2 - Violación menor	3 - Un individuo
En general impacto técnico: 1.250 BAJO					el impacto global de las empresas: 1.750 BAJO			
Impacto global: 1.500 BAJO								

Figura 45. Resumen de Probabilidad x Impacto en la aplicación web sobre Vulnerabilidades en PHP. Adaptado de *(The OWASP Fundation, 2015)*.

)Riesgo en base a Vulnerabilidades Reflected XSS

Un ataque exitoso permite ejecutar comandos arbitrarios (HTML y *JavaScript*) en el navegador de la víctima:

- Factores de Amenaza: El usuario necesita algunos conocimientos técnicos. La inyección se hace dentro de una URL enviada a la víctima y no requiere de mucha experiencia, esta vulnerabilidad ofrece una posible recompensa y uno de los ataques más comunes de *reflected XSS* como es el robo de *cookies*. No es necesario un acceso especial a la información y dicha vulnerabilidad, puede ser usada por cualquier usuario con acceso a internet.
- Factores de Vulnerabilidad: Existen herramientas automatizadas para explotar estas vulnerabilidades, el *exploit* pasa desapercibido por la aplicación web, a su vez el conocimiento sobre este tipo de vulnerabilidades es difundido por desarrolladores y empresas de seguridad.
- Impacto Técnico: Genera divulgación de datos críticos mínima y de datos no sensibles extensa, este ataque se basa en los usuarios que acceden a la web, no genera pérdidas de integridad o perdidas en la disponibilidad del servicio, es totalmente trazable y se puede determinar el *script* que corre en la URL del navegador.
- Impacto de negocios: Si el *exploit* se llega a ejecutar exitosamente se puede tener un efecto financiero mínimo, se puede tener pérdidas importantes de las cuentas del servicio. Causa una violación de alto perfil que afecta a una gran porción de usuarios.

En la figura 46 se resume las Vulnerabilidades de *Reflect XSS* en la aplicación web.

Probabilidad								
factores agente de amenaza					Los factores de vulnerabilidad			
Nivel de habilidad	Motivo	Oportunidad	Tamaño		Facilidad de descubrimiento	La facilidad de explotar	Conciencia	La detección de intrusiones
3 - Algunos conocimientos técnicos	4 - posible recompensa	9 - No requiere acceso o recursos	9 - Los usuarios de Internet anónimos		9 - Herramientas automatizadas disponibles	9 - Herramientas automatizadas disponibles	4 - Oculto	9 - No registado
probabilidad general: 7.000 ALTO								
Impacto								
Impacto técnico					Impacto de negocios			
Pérdida de confidencialidad	Pérdida de integridad	Pérdida de disponibilidad	Pérdida de trazabilidad del Atacante		Daño financiero	daños a la reputación	Incumplimiento	violación de privacidad
4 - Divulgación Mínima de datos Críticos, divulgación extensa de datos no sensibles	0 -	0 -	1 - Totalmente trazable		1 - Menos que el costo de arreglar la vulnerabilidad	4 - Pérdida de las principales cuentas	7 - Violación de alto perfil	7 - Miles de personas
En general impacto técnico: 1.250 BAJO					el impacto global de las empresas: 4.750 MEDIO			
Impacto global: 3.000 MEDIO								

Figura 46. Resumen de Probabilidad x Impacto en la aplicación web sobre Vulnerabilidades en Reflected XSS. Adaptado de *(The OWASP Fundation, 2015)*.

6.1.3. Riesgo en base a Vulnerabilidades Stored XSS

La aplicación web almacena datos proporcionados por el usuario sin validar o sin sanitización, pueden llegar a ser visualizados o utilizados por otros usuarios ocasionando comportamientos imprevistos cuando son usados por otro usuario o un administrador. Usualmente se considera como riesgo de nivel alto o crítico:

- Factores de Amenaza: El atacante debe tener experiencia avanzada con el computador, la inyección se lo hace dentro de un formulario, requiere de un uso avanzado de herramientas de desarrollo, esta vulnerabilidad puede ofrecer grandes beneficios a los atacantes debido al robo de *cookies* y cuentas. El alcance de la vulnerabilidad se agrava dependiendo de la habilidad del atacante, no es necesario acceso especial y es aprovechado por cualquier usuario con acceso a internet.
- Factores de Vulnerabilidad: Existen herramientas de automatización para encontrar estas vulnerabilidades, se requiere de un conocimiento técnico mayor para aprovecharlo. El *exploit* puede pasar desapercibido y la concienciación sobre este tipo de vulnerabilidades y su afectación es poco extendida. La actividad de dicho *exploit* es registrada y es necesario realizar una revisión de la base de datos.
- Impacto Técnico: Genera pérdida de confidencialidad si se filtran datos críticos. La integridad de datos no es afectada debido a que el atacante necesita que la Base de Datos se mantenga funcional el mayor tiempo posible para ejecutarse completamente. Hay afectación en el sitio *on-live*, existe la posibilidad de trazar el ataque siempre y cuando no exista ofuscación a fondo del *script* del atacante.
- Impacto de negocios: Tiene un efecto significativo en el beneficio anual del negocio desprestigiando a la Institución de Educación Superior por no controlar dicha vulnerabilidad. Se puede generar una violación de alto perfil que afectando a una gran parte de la población activa de la Institución de Educación Superior en la figura 47 se resumen el riesgo.

Probabilidad								
factores agente de amenaza					Los factores de vulnerabilidad			
Nivel de habilidad	Motivo	Oportunidad	Tamaño		Facilidad de descubrimiento	Facilidad de explotación	Concienciación	Detección de intrusión
5 - Usuario avanzado del ordenador	9 - Recompensa alta	9 - No requiere acceso o recursos	9 - Los usuarios de Internet anónimos		3 - Difícil	1 - Teórico	4 - Oculto	8 - Registrado sin revisión
probabilidad general: 6.000 ALTO								
Impacto								
Impacto técnico					Impacto de negocios			
Pérdida de confidencialidad	Pérdida de integridad	Pérdida de disponibilidad	Pérdida de trazabilidad del Atacante		Daño financiero	daños a la reputación	Incumplimiento	violación de privacidad
5 - Amplia divulgación de datos críticos	1 - Mínimo de datos ligeramente corruptos	7 - Servicios primarios extensos interrumpidos	7 - Posiblemente rastreable		7 - Efecto significativo en el beneficio anual	4 - Pérdida de las principales cuentas	7 - Violación de alto perfil	7 - Miles de personas
En general impacto técnico: 5.000 MEDIO					el impacto global de las empresas: 6.250 ALTO			
Impacto global: 5.625 MEDIO								

Figura 47. Resumen de Probabilidad x Impacto en la aplicación web sobre Vulnerabilidades en *Stored XSS*. Adaptado de *(The OWASP Fundation, 2015)*.

6.2. Matriz de Riesgos sobre Vulnerabilidades

La figura 48 muestra la matriz de Riesgos de las vulnerabilidades encontradas.

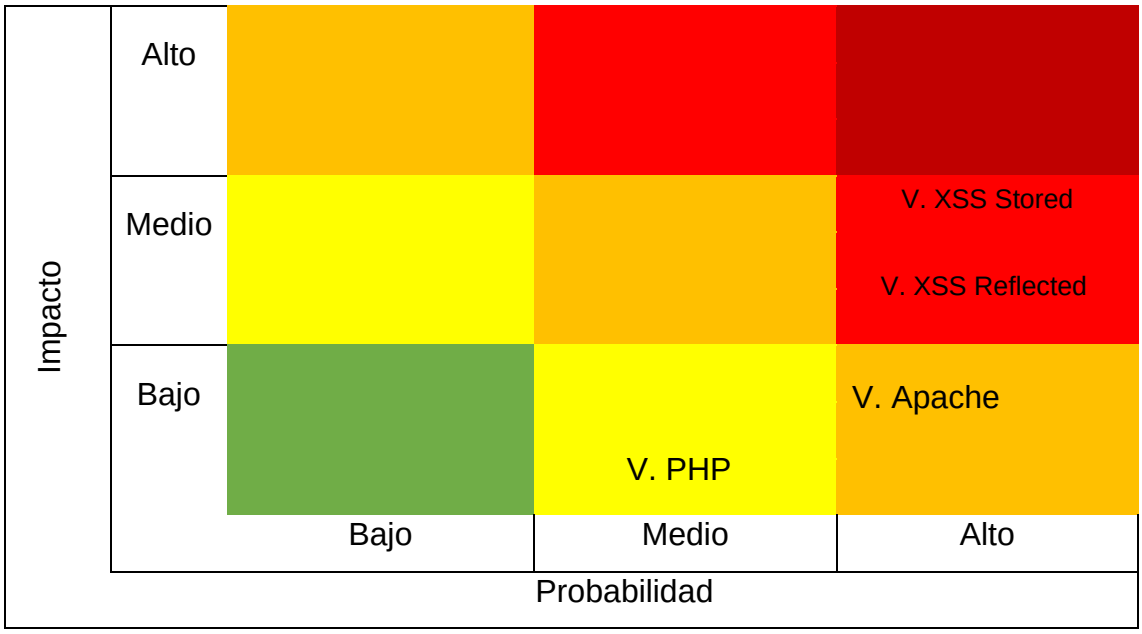

Figura 48. Matriz de Riesgos en base a la Probabilidad e Impacto de las Vulnerabilidades encontradas, basado en OWASP.

Con los riesgos descritos, se procede a la creación de una Propuesta de Mitigación, Prevención y Remediación de ataques informáticos, basado en las recomendaciones de NIST *Computer Security Incident Handling Guide*.

7. Propuesta de Mitigación de ataques Informáticos

Se presenta una propuesta de Mitigación de ataques informáticos, basado en las recomendaciones para Respuesta ante Incidentes Computacionales (Mandia, et. al, 2014) y el modelo de *Computer Security Incident Handling Guide* (NIST, 2012) el modelo presenta acciones para el grupo de Respuesta a Incidentes de Seguridad Informática en conjunto con una Institución de Educación Superior, su respuesta, acción y posible remediación para ataques informáticos.

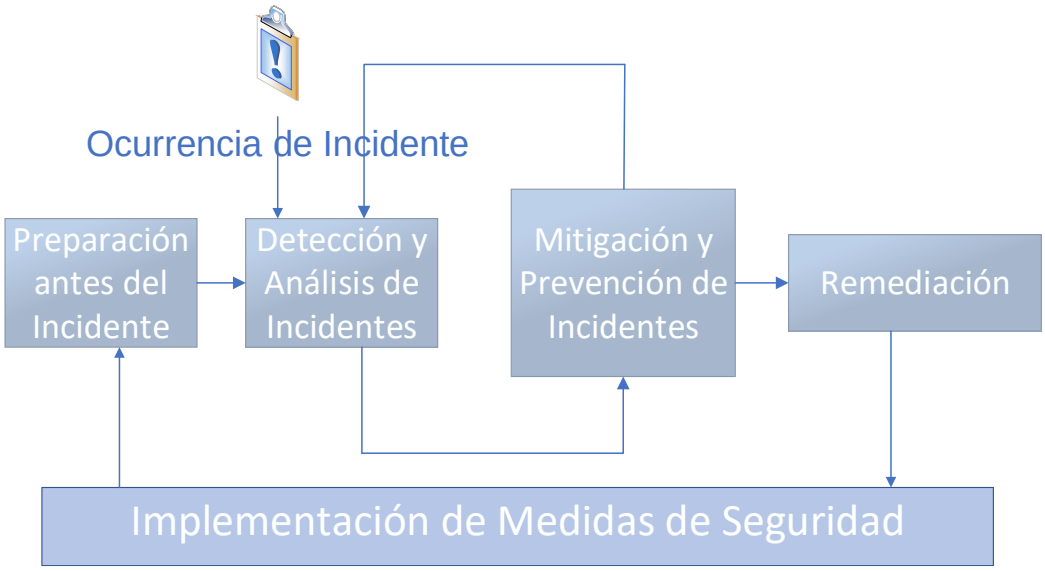

Figura 49. Modelo de respuesta ante incidentes.
Adaptado de (NIST,2012).

En el diagrama de flujo (Figura 50) se presenta las acciones recomendadas a seguir por una Institución de Educación Superior ante ataques informáticos. Se asume que el *host* tiene *Firewall* o Antivirus como medidas de seguridad esenciales para el funcionamiento mínimo del servicio.

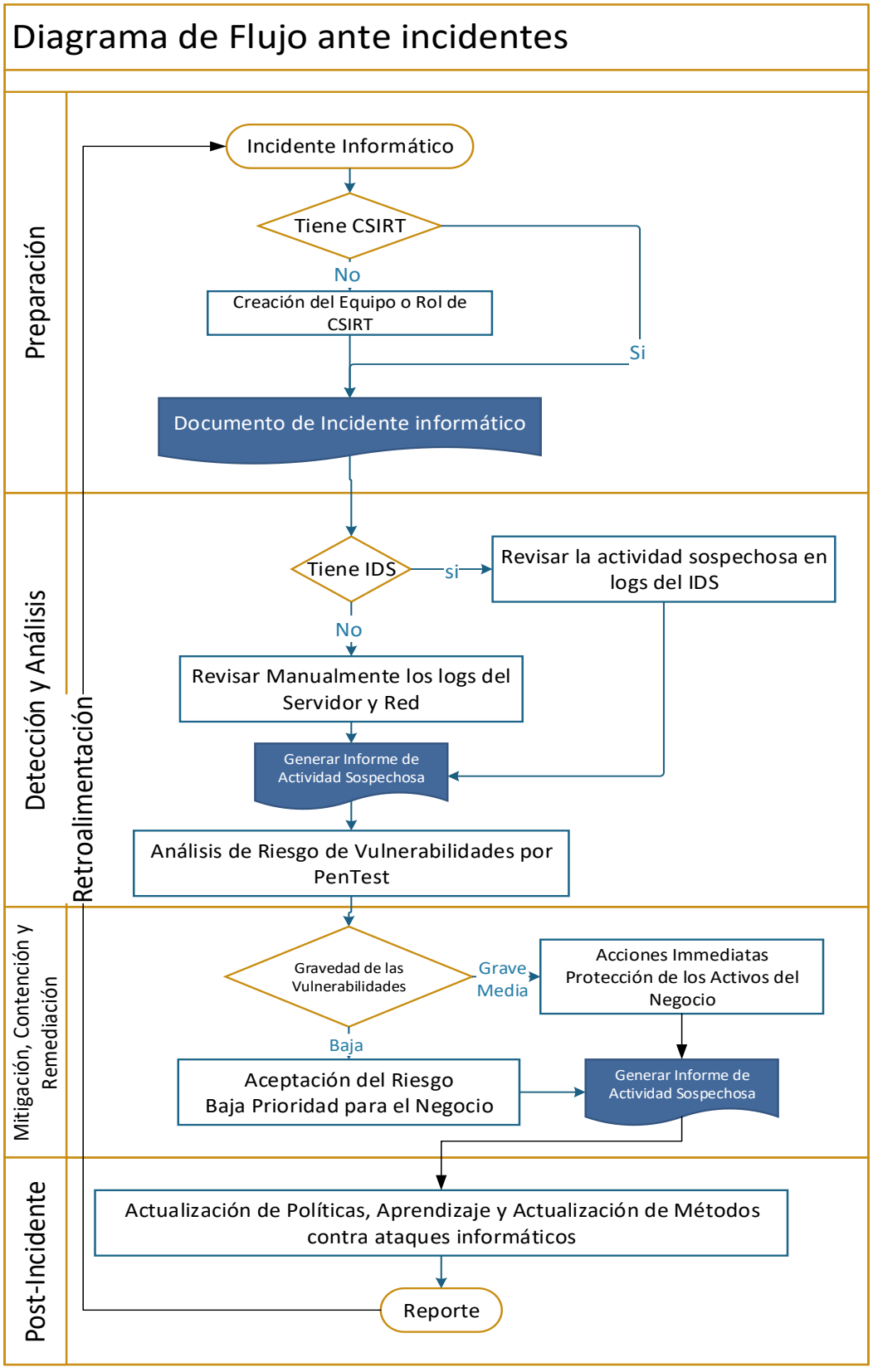

Figura 50. Diagrama de Flujo ante Incidentes Informáticos.

7.1. Preparación antes del Incidentes de ataque informático

En esta fase se propone todos los preparativos para evitar ataques informáticos antes de su ocurrencia en los sistemas web de una Institución de Educación Superior.

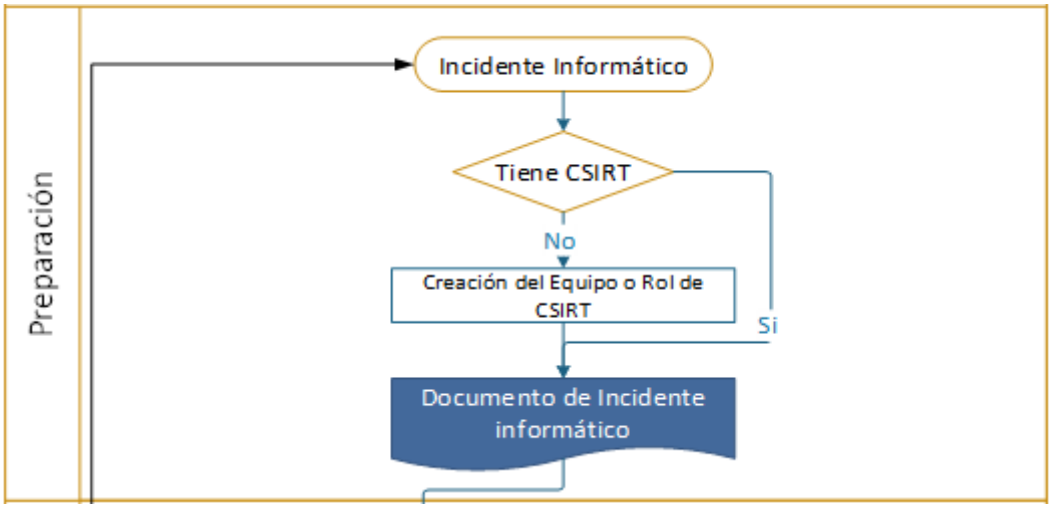

Figura 51. Fase de Preparación para Mitigación de Ataques Informáticos.

7.1.1. Creación de un Equipo de Respuesta de Incidentes Computacionales (CSIRT)

Puede ser una persona o un equipo, encargados de analizar datos del incidente, determinar su impacto, actuar apropiadamente y cooperar con otras organizaciones si fuera necesario. Este equipo debe tener una estructura Centralizada (Un solo equipo para todos los sistemas), Distribuida (Múltiples equipos por cada área de negocio, pero interconectadas) o Coordinada (Apoyo entre diferentes CSIRTs). En la tabla 12 se presenta el detalle de los procedimientos que el equipo de CSIRT debe realizar (NIST, 2012).

Tabla 12.
Características del Equipo de Respuesta de Incidentes Informáticos.

Característica	Aplicado a la Institución de Educación Superior
Establecer formalmente un CSIRT	Al ser un área pequeña de la empresa se recomienda que se asigne a una persona lo suficientemente calificada en seguridad informática para que pueda dar un informe detallado a las autoridades competentes, en caso de sospecha de Incidentes Informáticos.
Crear una política de respuesta a Incidentes informáticos	Crear una política que considere: • ¿Qué eventos son incidentes? • ¿A quién (de la institución o no) le corresponde seguir el incidente? • Definir roles y responsabilidades, tales como; ¿Quién debe realizar el monitoreo?, ¿Quién debe realizar la mitigación del incidente?, ¿Quién recolecta la información física o digital? • Equipo, materiales y software necesario como un *IDS*, para el reporte.
Desarrollar un plan de mitigación de ataques informáticos basado en políticas de respuesta a Incidentes informáticos	Crear una hoja de ruta (Figura 50), que implemente métricas para el procedimiento a largo y corto plazo, de la resolución de incidentes, así como la frecuencia de actualización de estos conocimientos y la habilidad requerida por el técnico a cargo.
Desarrollar procedimientos para la mitigación de ataques informáticos	Describir los pasos detallados para la respuesta a incidentes informáticos.
Establecer políticas y procedimiento para la divulgación de información	El CSIRT debe llegar a un acuerdo con el departamento legal, encargados de asuntos públicos y la gerencia, para cumplir con las políticas existentes en la organización.

Característica	Aplicado a la Institución de Educación Superior
Comunicar la información pertinente a la organización adecuada	En Ecuador tiene organismos como el Ecucert (Centro de respuestas a incidentes informáticos del Ecuador), Arcotel (Agencia de regulación y control de las telecomunicaciones), AEPROVI (Asociación de empresas proveedores de servicios de Internet, valor agregado, portadores y tecnologías de la información) y FIRST (*Forum of Incident Response and Security Teams*). Cabe mencionar el esfuerzo de las Universidades de la Escuela Politécnica Nacional y de La Universidad Particular de Loja al crear equipos de CSIRT.
Identificar otros grupos que necesiten participar para la gestión de Incidentes	Considerar la experiencia, el juicio y habilidades de otros equipos, incluyendo su administración, aseguramiento de la información, soporte de TI, asuntos legales y públicos.
Determinar la posibilidad de ofrecer otros servicios	En caso de ser posible, el mismo equipo de CSIRT puede monitorizar el sistema, detectar intrusiones, distribuir alertas de seguridad y educación a usuarios sobre seguridad.

Las pruebas de hacking ético realizadas en la Institución de Educación Superior, se asume que no cuenta con una metodología de respuesta a Incidentes computacionales de manera formal. El equipo encargado de seguridad de los servicios web debe crear un esquema de preparación (Figura 50), para futuras brechas de seguridad.

7.1.2. Pasos generales para implementación de un IDS

- Invertir en personal tanto para la instalación inicial como para el cuidado y la administración continua.
- Obtener parches para el Sistema Operativo ofrecidos por el proveedor del OS especialmente actualizaciones de seguridad.

- Corregir vulnerabilidades antes de la implementación del IDS.
- Escoger el lugar de instalación del IDS para determinar datos recolectados sobre ataques desde afuera de la red o desde adentro.
- Obtener control sobre la red administrativa, de tal manera que la red de producción no controle mensajes entre componentes de infraestructura.
- Conseguir almacenamiento suficiente (mínimo 1 TB) para guardar el tiempo necesario registros del IDS.
- El IDS debe registrar ambos lados de las transacciones de red (tráfico entrante/saliente).
- Deducir y registrar los *logs* y alertas que sean significativas del servidor ya sea del OS o de la BDD o del Web Server, de esta manera se puede ajustar el tiempo de las alertas recibidas.
- Mantener actualizado las definiciones del IDS.

7.1.3. Configurar un ambiente de PenTest

- Seleccionar la metodología a seguir para pruebas de penetración descritas en el capítulo 2.2.
- Seleccionar el tipo de ambiente que se ajuste a las necesidades del negocio, descritas en el capítulo 2.2.1.
- La metodología debe abarcar instrucciones y herramientas recomendadas para pruebas de *Pen Test,* así como priorizar ataques frecuentes en aplicaciones web ver Tabla 1.
- Detectar posibles vectores de ataques y puntuar riesgos y su posibilidad de ocurrencia, en base a la importancia para el negocio descritos en el capítulo 2.1.5.
- Crear una plantilla para la detección de Incidentes (Anexo 7).

7.2. Detección y Análisis de Incidentes

A continuación, se presentan pasos generales que debe tener un sistema para detectar Incidentes.

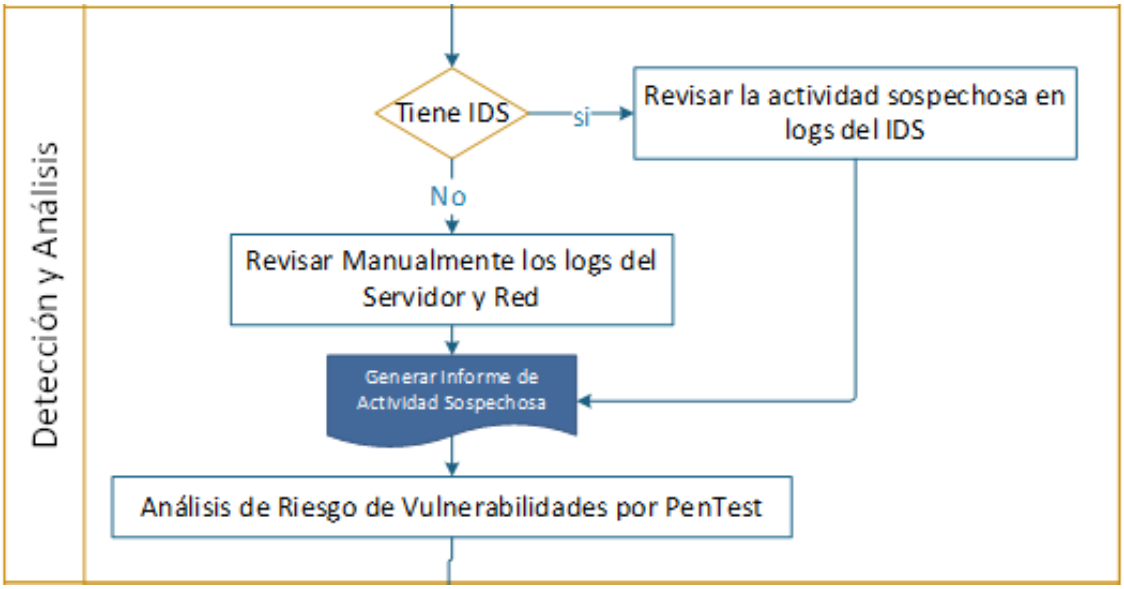

Figura 52. Fase de Detección y Análisis para Mitigación de Ataques Informáticos.

Al analizar las vulnerabilidades detalladas en los Capítulo 3, 4 y 5, se encuentran los siguientes problemas generales en las Instituciones de Educación Superior:

- Vulnerabilidad *XSS Reflected.*
- Vulnerabilidad *XSS Stored.*
- Vulnerabilidades de Apache.
- Vulnerabilidades de PHP.

7.3. Mitigación, Contención y Recuperación de Ataques Informáticos

Una vez encontradas las vulnerabilidades, se procede al análisis de Riesgo, para puntuar su gravedad e importancia Figura 3. En la fase de mitigación, contención y remediación de la Figura 53, se describen las acciones recomendadas de acuerdo con los activos del negocio afectados.

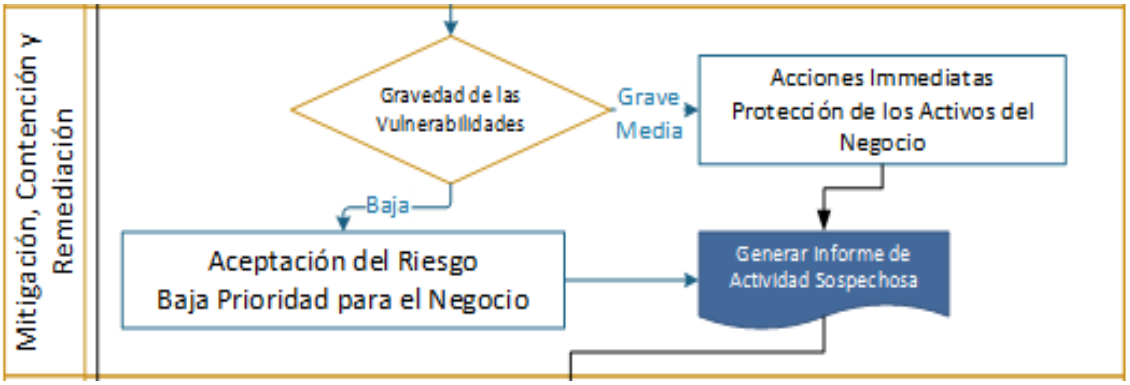

Figura 53. Fase de Mitigación, Contención y Remediación de Ataque Informático.

Valorar la gravedad de las vulnerabilidades en base a los activos del negocio, capítulo 2.1.5. Guardar la información concerniente, ver Capitulo 2.3.1 c, para el reporte.

De acuerdo con el capítulo 6, los riesgos que son más probables de ocurrencia dependiendo de sus Factores de probabilidad e Impacto fueron cuatro; Riesgo de *XSS Reflected* (Alto), *XSS Stored* (Alto), Riesgo basados en Apache (Medio) y Riesgos basados en PHP (Bajo).

7.3.1. Mitigación, Contención y Recuperación de vulnerabilidades XSS Reflected y XSS Stored

a. Implementar Header X-XSS-Protection en Apache:

- La cabecera X-XSS-Protection previene algunos niveles de ataques XSS (*cross-site-scripting*), además de ser compatible los navegadores: IE 8+, Chrome, Opera, Safari & Android.
- Hay cuatro maneras posibles de configurar esta cabecera.

Tabla 13.
Configuración de la cabecera de Apache.

Valor del Parámetro	**Significado**
0	Filtro XSS desactivado

Valor del Parámetro	Significado
1	Filtro XSS activo y la página será sanitizada si detecta un ataque.
1;mode=block	Filtro XSS activo y no renderizara la página si detecta un ataque.
1;report=http://example.com/report_URI	Filtro XSS activo y reportara la incidencia si detecta un ataque.

Adaptado de (The Apache Software Foundation, 2018).

b. Implementar en el Servidor HTTP Apache

- Añadir la siguiente entrada en httpd.conf del servidor web Apache:

```
1.  Header set X - XSS - Protection "1; mode=block"
```

Figura 54. Extracto de código en la configuración de Apache.
Adaptado de (The Apache Software Foundation, 2018).

- Reiniciar Apache para completar los cambios.

c. Activar módulo de WordPress para sanitización de campos:

La barra de búsqueda debe ser sanitizada para su uso en la página web de la Institución de Educación Superior, lo cual se realiza con el siguiente módulo:

- **Prevent XSS Vulnerability:** Se encuentra en el siguiente enlace: https://wordpress.org/plugins/prevent-xss-vulnerability/ o puede ser instalado directamente con el gestor de módulos de WordPress.

7.3.2. Mitigación, Contención y Recuperación sobre Vulnerabilidades de Apache

a. Actualizar el Servidor Apache:

Seguir los pasos recomendados para actualizar la versión de Apache, dependiendo de la versión exacta de sistema operativo, en este caso CentOS 6.

Apache puede ser actualizado de dos formas usando RPMs, método directo para realizar la actualización, o utilizando Colecciones de Software o SCL lo cuales instalarán Apache 2.4 como una entidad aparte dentro de la carpeta /opt, ideal para hacer pruebas (The Apache Software Foundation, 2018).

b. Activar el header HTTP Strict Transport Security (HSTS)

La cabecera HSTS (*HTTP Strict Transport Security*) asegura que toda la comunicación desde un navegador se envíe a través de HTTPS (*HTTP Secure*) y redirige las solicitudes HTTP a HTTPS. Antes de implementar esta cabecera, es necesario asegurarse de que toda la página web sea accesible a través de HTTPS, de lo contrario se bloquearán ciertos elementos web.

La cabecera HSTS es compatible con todas las versiones más recientes de navegadores como IE, Firefox, Opera, Safari y Chrome. Existen tres parámetros de configuración.

Tabla 14.
Parámetros HSTS recomendados.

Valor del Parámetro	Significado
max-age	Duración (en segundos) indica a un navegador que las peticiones sólo están disponibles a través de HTTPS.
includeSubDomains	La configuración es válida para el subdominio.
Preload	En caso de que se desee que el dominio se incluya en el directorio de Lista de precarga HSTS

Adaptado de (Apache,2017).

c. **Lista de precarga HSTS**

Es una lista de sitios que están codificados en Chrome como HTTPS solamente, este formulario se utiliza para enviar dominios para su inclusión en la lista de precarga *HTTP Strict Transport Security* (HSTS) de Chrome. La mayoría de los navegadores principales (Chrome, Firefox, Opera, Safari, IE 11 y Edge) también tienen listas de precarga HSTS basadas en la lista Chrome.

- **Implementación en el Servidor HTTP Apache:** Puede implementar HSTS en Apache añadiendo la siguiente entrada en el archivo httpd.conf:

```
1.  Header set Strict-Transport-Security "max-age=31536000; includeSubDomains; preload"
```

Figura 55. Extracto de código para Lista de precarga HSTS.
Adaptado de (Apache, 2017).

- Reiniciar Apache para completar los cambios.

d. Activar el header X-Content-Type-Options

Previene el riesgo de seguridad del tipo MIME, añadiendo esta cabecera a la respuesta HTTP de la página web. Este encabezado instruye al navegador para que considere tipos de archivos como definidos y no permita el *sniffing* de contenido. Sólo hay un parámetro que se debe añadir "nosniff".

- **Implementación en el Servidor HTTP Apache:** Puede implementar HSTS en Apache añadiendo la siguiente entrada en el archivo httpd.conf:

```
1.  Header set X-Content-Type-Options nosniff
```

Figura 56. Extracto del *script* del encabezado de X-Content.
Adaptado de (Apache, 2017).

- Reiniciar Apache para completar los cambios.

e. Ocultar la Huella digital del servidor

Para ocultar el número de versión del servidor web, los detalles del sistema operativo del servidor, los módulos Apache instalados y demás, editar el archivo de configuración del servidor web Apache httpd.conf.

- Es necesario añadir/modificar las siguientes líneas:

```
1. ServerTokens Prod

2. ServerSignature Off
```

Figura 57. Extracto de configuración en el archivo httpd.conf.
Adaptado de (Apache, 2017).

- Reiniciar Apache para completar los cambios.

7.3.3. Sobre Vulnerabilidades de PHP

Esta vulnerabilidad es generada por la accesibilidad que ofrece el servidor a los archivos con información sobre PHP. Es necesario eliminar los archivos info.php a los cuales el atacante puede tener acceso, esencialmente los que se encuentran en la ruta /php de la carpeta raíz del sitio.

- Ocultar la información de la versión de PHP que recogen los escáneres editando el php.ini del servidor, se debe añadir o modificar la siguiente línea:

```
1. expose_php = off
```

Figura 58. Extracto de Script de php.ini.
Adaptado de *(The PHP Group, 2018)*.

- Reiniciar Apache para completar los cambios.

7.4. Actividades Post Incidentes

El objetivo del proceso de post-incidente es establecer o proponer mejoras en el actual proceso de contención de incidentes informáticos, restablecer el sistema a la normalidad eliminando vulnerabilidades conocidas evitando así, que vuelvan a ocurrir incidentes similares.

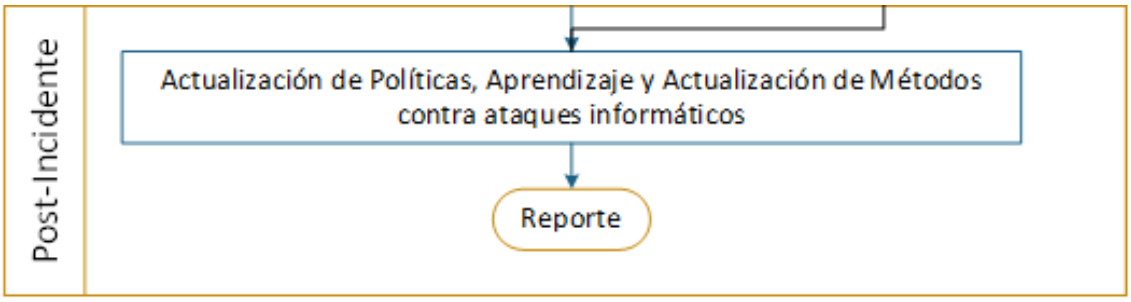

Figura 59. Actividades Post-Incidentes.

7.4.1. Manejo de un IDS en WordPress

WordPress ofrece un plugin llamado WP Security Audit Log**.** El módulo se puede encontrar y descargar en el siguiente enlace: https://en-gb.wordpress.org/plugins/wp-security-audit-log/. Su instalación se la puede realizar desde el mismo WordPress con su manejador de módulos.

a. Configuración y Familiarización WP Security Audit Log

- **Detección y Alerta de un *Hackeo* en WordPress**

  - Revisar si se crearon nuevos usuarios y si crearon *backdoors*.
  - Revisar si se instalaron módulos de WordPress extraños
  - Revisar el contenido de widgets, páginas web, foros, etc.

- **Detección por medio de escaneos automáticos de Seguridad de WordPress**

El escaneo automatizado envía miles de peticiones HTTP, tratando de explotar vulnerabilidades conocidas en plugins o temas de WordPress. Esta actividad

genera muchos errores HTTP 404, el plugin WP Security Audit Log hace un seguimiento de todas las peticiones a páginas inexistentes con dos alertas:

- Alerta 6007 para 404 errores generados por usuarios conectados,
- Alerta 6023 para 404 errores generados por usuarios anónimos.

El plugin registra los errores HTTP 404 en un archivo de registro. Analizando estos archivos de registro se puede aprender sobre el tipo de ataques o vulnerabilidades que los atacantes están tratando de encontrar y explotar en el sitio web de WordPress, protegiendo de mejor manera él sitio.

- **Detección de ataques de fuerza bruta en la página de inicio de sesión de WordPress**

El plugin de registro de auditoría de seguridad de WordPress mantiene un registro mediante el uso de estos dos tipos de alertas:

- Alerta 1003 para cuando alguien intenta acceder a WordPress usando un usuario no existente.
- Alerta 1002 para cuando alguien intenta iniciar sesión usando un usuario existente.

- **Recibir Notificación de Otras Actividades Sospechosas de Usuarios de WordPress**

Utilizar notificaciones de correo electrónico para avisar sobre cualquier posible cambio que se produzca en el ambiente de WordPress, estas notificaciones pueden ser por cambios en:

- Un usuario de WordPress cambia la contraseña de otro usuario.
- Un usuario de WordPress se conecta desde una dirección IP diferente.
- Un usuario de WordPress inicia sesión durante ciertas horas.
- Un usuario de WordPress instaló o activó un plugin.

La Institución de Educación Superior debe configurar el tipo de notificaciones que el Plugin de WordPress puede ofrecer de acuerdo con las necesidades y prioridades de su negocio

8. Conclusiones y Recomendaciones

En el presente capítulo se darán las conclusiones y recomendaciones que se tiene al realizar este trabajo de titulación:

8.1. Conclusiones

Todas las aplicaciones son vulnerables a diferentes tipos de ataque, realizar pruebas personalizadas de PenTesting mantiene la criticidad de vulnerabilidades a tratar desde el punto de vista del negocio, esto quiere decir enfrentar el costo de arreglar los problemas contra el costo de no hacerlo.

La importancia de un CIRT en las empresas de Educación Superior del Ecuador ha ido en aumento, debido a los constantes ataques realizados por hackers, la comunicación, y el apoyo mutuo son cruciales para mantener la seguridad en un futuro, para lo cual se debe seguir todos los años conferencias y exposiciones presentadas por expertos en seguridad informática que recomiendan parámetros básicos ante incidentes de seguridad informática entre todos los Equipos de Seguridad a cargo.

La implementación de un sistema de Detección de Intrusos es crucial para la empresa y sus activos, esto es, debido a que la mejor forma de protegerse ante ataques informáticos es previniendo su ocurrencia, el antivirus y el firewall por si solos no son suficientes para una defensa activa ante ataques informáticos.

El estado actual de las páginas web de las Instituciones de Educación Superior y del Ecuador, ha mejorado con respecto a años anteriores, esto lo demuestra el análisis general de situación actual en comparación de hace 5 años, en donde las instituciones han cambiado su forma de gestión de la información a servicios extranjeros relegando la función de mantenimiento de equipos y de adquisición de personal a terceros. Por consecuente, se ve la pérdida de personas internas en la organización cualificadas para la respuesta pronta ante incidentes informáticos en el Ecuador.

8.2. Recomendaciones

Incluso las grandes Instituciones de Educación Superior con gran talento y recursos significativos dedicados a la ciberseguridad han sufrido importantes compromisos en esta área, y organizaciones que no tienen tales niveles de talento o recursos se enfrentan a retos aún mayores. Un mayor número de trabajadores altamente cualificados en funciones de ciberseguridad ayudaría a responder más enérgicamente a los problemas de ciberseguridad que se presentan. Todas las organizaciones deben comprender su entorno de amenazas y los riesgos a los que se enfrentan, abordar sus problemas de ciberseguridad y contratar a las personas más adecuadas para realizar ese trabajo.

Para empezar a realizar pruebas de PenTesting, se recomienda Kali Linux. Para otras pruebas más especializadas se recomienda Ambientes de PenTesting pagados y auditados para cada necesidad de la empresa.

Se recomienda hacer pruebas del mismo tipo a los demás servicios que ofrece la Institución de Educación Superior para poder comprobar el estado de su seguridad y confiabilidad de los usuarios finales.

Cuando se realiza pruebas de Pen Testing es necesario mantener un nivel de ética con la entidad objetivo debido a que toda la información que se encuentre o pruebe no debe afectarla bajo ningún sentido o ser ocultada de la misma para luego ser aprovechada por otros atacantes, en caso de que la vulnerabilidad sea grave es necesario reportarla lo más pronto posible.

REFERENCIAS

© Identity Theft Resource Center. (2017). *ANNUAL DATA BREACH YEAR-END REVIEW.* Recuperado el 16 de marzo del 2018, de https://www.idtheftcenter.org/images/breach/2017Breaches/2017AnnualDataBreachYearEndReview.pdf

© 2018 AO Kaspersky Lab. (2017). Kaspersky Lab: Brasil, México y Colombia lideran incidentes de secuestros digitales en América Latina. Recuperado el Comunicados de prensaq: https://latam.kaspersky.com/about/press-releases/2017_kaspersky-lab-incidents-of-digital-kidnappings-in-latin-america

Alexa Internet Inc. (2018). *Website Traffic, Statistics and Analytics.* Recuperado el 21 de marzo del 2018 de https://www.alexa.com/siteinfo

Babulak, E. (2015). *Web Security: SQL Injection.* Recuperado el 23 de marzo del 2018 de https://www.slideshare.net/sayvortana/powerpoint-presentation-1-47302656

Borghello, C. (2001). Presupuestos para la Punibilidad del Hacking. Recuperado el 20 de noviembre del 2017 de: http://www.segu-info.com.ar/delitos/tiposdelito.htm

Carettoni, L. (2009). *HTTP Parameter Pollution (HPP) - SEaCURE.it edition.* Recuperado el 15 de abril del 2018 de https://es.slideshare.net/ikkisoft/http-parameter-pollution-hpp-seacureit-presentation-by-luca-carettoni-and-stefano-di-paola

Codecamp Romania. (2012). *Developing Secure Web Application - Cross-Site Scripting (XSS) .* Recuperado el 18 de marzo del 2018 de https://www.slideshare.net/codecampiasi/developing-secure-web-application-crosssite-scripting-xss

Common Vulnerabilities and Exposures. (2018). *Common Vulnerabilities and Exposures (CVE).* Recuperado el 13 de marzo del 2018 de https://cve.mitre.org/

Escrivá Gascó, G., Romero Serrano, R. M., Ramada, D. J., & Onrubia Pérez, R. (2017). Seguridad informática. Madrid: MACMILLAN IBERIA, S.A.U.

Freire, A., & Lennin, A. (2017). La ciberseguridad, y su incidencia en las Políticas de la Seguridad de la Información de la Armada del Ecuador. *Yura: Relaciones internacionales*, 306-323. Recuperado el 3 de mayo del 2018 de http://world_business.espe.edu.ec/wp-content/uploads/2017/06/11.15-La-ciberseguridad-y-su-incidencia-en-las-Políticas-de-la-Seguridad-de-la-Información-de-la-Armada-del-Ecuador-1.pdf

Gómez Vieites, Á. (2011). Enciclopedia de la seguridad informática. México: Alfaomega.

Imperva. (2018). *Remote File Inclusion (RFI).* Recuperado el 21 de diciembre del 2017 de https://www.incapsula.com/web-application-security/rfi-remote-file-inclusion.html

ISACA (2015). Gestión de Riesgos y Continuidad Operativa en IT: Recomendaciones Prácticas. Recuperado el 27 de noviembre del 2017 de https://www.isaca.org/chapters8/Montevideo/cigras/Documents/CIGRAS2015/CIGRAS-2015.09.09-09-Gestion%20de%20Riesgos%20y%20Continuidad%20Operativa%20de%20IT%20Recomendaciones%20Practicas-Rodrigo%20Guirado.pdf

ISECOM. (2010). *The Open Source Security Testing Methodology Manual*. Cardedeu, Barcelona, España.

Mandia, K., Prosise, C., & Pepe, M. (2014). *Incident Response & Computer Forensics.* McGraw Hill Professional, 2014.

Nadji, Y., Saxena, P., & Dawn, S. (2009). *Document Structure Integrity: A Robust Basis for Cross-site Scripting Defense*. Recuperado el 2 de enero del 2018 de http://webblaze.cs.berkeley.edu/papers/nadji-saxena-song.pdf

Neo Kobo. (2014). *Client-Side Attacks.* Recuperado el 8 de marzo del 2018 de http://neokobo.blogspot.com/2014/01/3218-client-side-attacks.html

NIST. (2002). *Archived NIST Technical Series Publication.* Recuperado el 20 de febrero del 2018 de https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-30.pdf

NIST. (2007). *Guide to Intrusion Detection and Prevention Systems (IDPS).* doi:SP 800-31

NIST. (2012). Computer Security Incident Handling Guide. *National Institute of Standards and Technology Special Publication 800-61*, 79. doi:SP 800-61 Rev. 2

Offensive Security. (2018). *Offensive Security Certifications, Training, Courses and Services.* Recuperado el 14 de mayo del 2018 de https://www.offensive-security.com/

Open Source. (2011). *Securing Apache, Part 5: HTTP Message Architecture.* Recuperado el 5 de junio del 2018 de https://opensourceforu.com/2011/01/securing-apache-part-5-http-message-architecture/

Pérez Agudín, J., Pérez Míguez, C., Matas Garcia, A. M., Picouto Ramos, F., & Ramos Varón, A. Á. (2006). La Biblia del Hacker. Madrid: Grupo Anaya, S.A.

SANS Institute (2011). *Incident Handler's Handbook.* Recuperado el 16 de marzo del 2018 de https://www.sans.org/reading-room/whitepapers/incident/incident-handlers-handbook-33901

Shakeel, I. (2017). *InfoSec Institute*. Recuperado el 5 de abril del 2018 de https://resources.infosecinstitute.com/

The Apache Software Foundation. (2018). *The Apache HTTP Server Project.* Recuperado el 15 de diciembre del 2017 de https://httpd.apache.org/

The Apache Software Foundation. (2018). *Upgrading to 2.4 from 2.2.* Recuperado el 13 de junio del 2018 de https://httpd.apache.org/docs/trunk/upgrading.html

The OWASP Foundation. (2014). *Web Application Penetration Testing.* Recuperado el 4 de abril del 2018 de https://www.owasp.org/index.php/Web_Application_Penetration_Testing

The OWASP Foundation. (2017). OWASP Top 10 - 2017 Los diez riesgos más críticos en Aplicaciones Web. San Jose, California, Estados Unidos.

The OWASP Fundation. (2015). *OWASP Risk Rating Template Example.* Recuperado el 20 de febrero del 2018 de https://www.owasp.org/index.php/File:OWASP_Risk_Rating_Template_Example.xlsx

The OWASP Fundation. (2018). *OWASP Testing Guide 4.0*. San Jose, California, Estados Unidos.

The PHP Group. (2018). *PHP: Hypertext Preprocessor.* Recuperado el 14 de mayo del 2018 de: http://php.net/

Viegas, E., Santin, A. O., França, A., Jasinski, R., Pedroni, V. A., & Oliveira, L. S. (2017). *Towards an Energy-Efficient Anomaly-Based Intrusion Detection Engine for Embedded Systems*. *IEEE TRANSACTIONS ON COMPUTERS*, 167-177.

ANEXOS

Anexo 1.

Acuerdo con AWS

AWS Vulnerability / Penetration Testing Request Form

In order to request permission to conduct vulnerability and penetration testing originating from any resources in the AWS Cloud, the following information is required. The requesting party must also accept the Terms and Conditions and AWS's policy regarding the Use of Security Assessment Tools and Services. Upon receipt and validation of the information, an authorization email approving the test plan will be sent to the addresses provided below. Testing is not authorized until the requesting party receives that authorization. An asterisk (*) indicates required information:

Contact Information

Please provide the email address and the associated name of the AWS account owner with which you have used to log into this form. The AWS Account ID number of the account used to log into this form will be sent along with your submission. If you would like to request testing for a different account, please log out and log back in with the account for which you want to test.

Your Name:*

Company Name*

Email Address

Additional Email Address

Additional Email Address

Additional Email Address

Third Party Contact Information

Anexo 2.

Acuerdo con Institución de Educación Superior

CONTRATO DE CONFIDENCIALIDAD

Al objeto de garantizar la confidencialidad del presente [**Proyecto, colaboración entre las partes implicadas**], se hace necesario la firma de un acuerdo que garantice unos niveles de confianza entre las partes. El documento se firmará una vez aceptado y firmado el (**tipo: contrato, acuerdo,...]** por ambas partes.

El contenido del acuerdo es el que figura a continuación. Contenido

DE UNA PARTE: **[nombre de la organización]**y en su nombre y representación (con poder suficiente para ello) D/Dña. **[nombre completo],** en calidad de **[cargo, administrador, apoderado,...]**

DE OTRA PARTE: **[nombre de la organización].** y en su nombre y representación **(con poder suficiente para ello)** D/Dña. [nombre completo], en calidad de **[cargo, administrador, apoderado,...]**

Reunidos en **[lugar de la firma del contrato]**, a **[día]** de **[Mes]** de **[Año]**

EXPONEN

I – Que las partes, anteriormente citadas, están interesadas en el desarrollo del presente contrato, para lo cual, aceptaron celebrar el presente Acuerdo de Confidencialidad con el fin de establecer el procedimiento que regirá la custodia y no transmisión a terceros de la información distribuida entre las partes, así como los derechos, responsabilidades y obligaciones inherentes en calidad de remitente, Propietario y «Destinatario» de la referida información.

II – Que las partes, en virtud de lo anteriormente expuesto, convinieron que el presente Acuerdo de Confidencialidad se rija por la normativa aplicable al efecto y, en especial por las siguientes.

CLÁUSULAS

PRIMERA - Definiciones

A los efectos del presente Acuerdo, los siguientes términos serán interpretados de acuerdo con las definiciones anexas a los mismos. Entendiéndose por:

- «Información propia»: tendrá tal consideración y a título meramente enunciativo y no limitativo, lo siguiente: descubrimientos, conceptos, ideas, conocimientos, técnicas, diseños, dibujos, borradores, diagramas, textos, modelos, muestras, bases de datos de cualquier tipo, aplicaciones, programas, marcas, logotipos, así como cualquier información de tipo técnico, industrial, financiero, publicitario, de carácter personal o comercial de cualquiera de las partes, esté o no incluida en la solicitud de oferta
presentada, independientemente de su formato de presentación o distribución, y aceptada por los «Destinatarios».

- «Fuente»: tendrá la consideración de tal, cualquiera de las partes cuando, dentro de los términos del presente Acuerdo, sea ella la que suministre la Información Propia y/o cualquiera de los implicados (accionistas, directores, empleados, ...) de la empresa o la organización.

- «Destinatarios»: tendrán la consideración de tales cualquiera de las partes cuando, dentro de los términos del presente Acuerdo, sea ellos quienes reciban la Información Propia de la otra parte.

SEGUNDA.- Información Propia.

Las partes acuerdan que cualquier información relativa a sus aspectos financieros, comerciales, técnicos, y/o industriales suministrada a la otra parte como consecuencia de la solicitud de Oferta para el desarrollo del presente proyecto objeto del contrato, o en su caso, de los acuerdos a los que se lleguen (con independencia de que tal transmisión sea oral, escrita, en soporte magnético o en cualquier otro mecanismo informático, gráfico, o de la

Anexo 3.

```
| _clock - skew: mean: -2 m25s, deviation: 0 s, median:
2 m25s TRACEROUTE(using port 443 / tcp)
HOP RTT ADDRESS
1 7.57 ms 192.168.0.1
2...
3 21.01 ms 172.21.19.78
4 22.72 ms 172.21.8.81
5 22.79 ms 172.21.8.82
6 79.47 ms ae3 - 205. cr0 - mia1.ip4.gtt.net(173.241.130.49)
7 73.33 ms et - 0 - 0 - 4 - 0. cr4 - mia1.ip4.gtt.net(89.149.140.106)
8 84.64 ms microsoft - corporation - gw.ip4.gtt.net(199.229.231.142)
9 89.53 ms ae9 - 0. mia - 96 cbe - 1 b.ntwk.msn.net(104.44.225.167)
10 99.78 ms be - 75 - 0. ibr02.atb.ntwk.msn.net(104.44.224.230)
11 109.99 ms be - 2 - 0. ibr03.atb.ntwk.msn.net(104.44.4.41)
12 100.21 ms be - 4 - 0. ibr01.was05.ntwk.msn.net(104.44.4.22)
13 110.78 ms be - 6 - 0. ibr02.bl20.ntwk.msn.net(104.44.5.83)
14 104.78 ms ae123 - 0. icr02.bl7.ntwk.msn.net(104.44.10.147)
15...28
29 109.51 ms
```

Anexo 4.

```
if (typeof method === 'object') {
    $.extend(true, this.settings, method);
}
if (typeof method != 'string') {
    if (!this.settings.init) {
        this.events();
    }
    this.assemble();
    return this.settings.init;
} else {
    return this[method].call(this, options);
}
}, assemble: function() {
        $('form.custom input[type="radio"]', $(this.scope)).not('[data-customforms="disabled"]').each(this.append_custom_markup);
        $('form.custom input[type="checkbox"]', $(this.scope)).not('[data-
customforms="disabled"]').each(this.append_custom_markup);
        $('form.custom select', $(this.scope)).not('[data-
customforms="disabled"]').not('[multiple=multiple]').each(this.append_custom_select);
    }, events: function() {
        var self = this;
        $(this.scope).on('click.fndtn.forms', 'form.custom span.custom.checkbox', function(e) {
            e.preventDefault();
            e.stopPropagation();
            self.toggle_checkbox($(this));
        }).on('click.fndtn.forms', 'form.custom span.custom.radio', function(e) {
            e.preventDefault();
            e.stopPropagation();
            self.toggle_radio($(this));
        }).on('change.fndtn.forms', 'form.custom select:not([data-customforms="disabled"])', function(e, force_refresh) {
            self.refresh_custom_select($(this), force_refresh);
        }).on('click.fndtn.forms', 'form.custom label', function(e) {
            if ($(e.target).is('label')) {
                var $associatedElement = $('#' + self.escape($(this).attr('for')) + ':not([data-customforms="disabled"])'),
                    $customCheckbox, $customRadio;
                if ($associatedElement.length !== 0) {
                    if ($associatedElement.attr('type') === 'checkbox') {
                        e.preventDefault();
                        $customCheckbox = $(this).find('span.custom.checkbox'); //the checkbox might be outside after the label or inside of a
nother element
                        if ($customCheckbox.length == 0) {
                            $customCheckbox = $associatedElement.add(this).siblings('span.custom.checkbox').first();
                        }
                        self.toggle_checkbox($customCheckbox);
                    } else if ($associatedElement.attr('type') === 'radio') {
                        e.preventDefault();
                        $customRadio = $(this).find('span.custom.radio'); //the radio might be outside after the label or inside of another eleme
nt
                        if ($customRadio.length == 0) {
                            $customRadio = $associatedElement.add(this).siblings('span.custom.radio').first();
                        }
                        self.toggle_radio($customRadio);
                    }
                }
            }
        }).on('mousedown.fndtn.forms', 'form.custom div.custom.dropdown', function() {
            return false;
        }).on('click.fndtn.forms', 'form.custom div.custom.dropdown a.current, form.custom div.custom.dropdown a.selector', function(e)
{
            var $this = $(this),
                $dropdown = $this.closest('div.custom.dropdown'),
```

```
            $select = getFirstPrevSibling($dropdown, 'select'); // make sure other dropdowns close
        if (!$dropdown.hasClass('open')) $(self.scope).trigger('click');
        e.preventDefault();
        if (false === $select.is(':disabled')) {
            $dropdown.toggleClass('open');
            if ($dropdown.hasClass('open')) {
                $(self.scope).on('click.fndtn.forms.customdropdown', function() {
                    $dropdown.removeClass('open');
                    $(self.scope).off('.fndtn.forms.customdropdown');
                });
            } else {
                $(self.scope).on('.fndtn.forms.customdropdown');
            }
            return false;
        }
    }).on('click.fndtn.forms touchend.fndtn.forms', 'form.custom div.custom.dropdown li', function(e) {
        var $this = $(this),
            $customDropdown = $this.closest('div.custom.dropdown'),
            $select = getFirstPrevSibling($customDropdown, 'select'),
            selectedIndex = 0;
        e.preventDefault();
        e.stopPropagation();
        if (!$(this).hasClass('disabled')) {
            $('div.dropdown').not($customDropdown).removeClass('open');
            var $oldThis = $this.closest('ul').find('li.selected');
            $oldThis.removeClass('selected');
            $this.addClass('selected');
            $customDropdown.removeClass('open').find('a.current').text($this.text());
            $this.closest('ul').find('li').each(function(index) {
                if ($this[0] == this) {
                    selectedIndex = index;
                }
            });
            $select[0].selectedIndex = selectedIndex; //store the old value in data
            $select.data('prevalue', $oldThis.html());
            $select.trigger('change');
        }
    });
    $(window).on('keydown', function(e) {
        var focus = document.activeElement,
            self = Foundation.libs.forms,
            dropdown = $('.custom.dropdown.open');
        if (dropdown.length > 0) {
            e.preventDefault();
            if (e.which === 13) {
                dropdown.find('li.selected').trigger('click');
            }
            if (e.which === 27) {
                dropdown.removeClass('open');
            }
            if (e.which >= 65 && e.which <= 90) {
                var next = self.go_to(dropdown, e.which),
                    current = dropdown.find('li.selected');
                if (next) {
                    current.removeClass('selected');
                    self.scrollTo(next.addClass('selected'), 300);
                }
            }
            if (e.which === 38) {
                var current = dropdown.find('li.selected'),
                    prev = current.prev(':not(.disabled)');
                if (prev.length > 0) {
                    prev.parent()[0].scrollTop = prev.parent().scrollTop() - self.outerHeight(prev);
```

```
                    current.removeClass('selected');
                    prev.addClass('selected');
                }
            } else if (e.which === 40) {
                var current = dropdown.find('li.selected'),
                    next = current.next(':not(.disabled)');
                if (next.length > 0) {
                    next.parent()[0].scrollTop = next.parent().scrollTop() + self.outerHeight(next);
                    current.removeClass('selected');
                    next.addClass('selected');
                }
            }
        }
    });
    this.settings.init = true;
}, go_to: function(dropdown, character) {
    var lis = dropdown.find('li'),
        count = lis.length;
    if (count > 0) {
        for (var i = 0; i < count; i++) {
            var first_letter = lis.eq(i).text().charAt(0).toLowerCase();
            if (first_letter === String.fromCharCode(character).toLowerCase()) return lis.eq(i);
        }
    }
}, scrollTo: function(el, duration) {
    if (duration < 0) return;
    var parent = el.parent();
    var li_height = this.outerHeight(el);
    var difference = (li_height * (el.index())) - parent.scrollTop();
    var perTick = difference / duration * 10;
    this.scrollToTimerCache = setTimeout(function() {
        if (!isNaN(parseInt(perTick, 10))) {
            parent[0].scrollTop = parent.scrollTop() + perTick;
            this.scrollTo(el, duration - 10);
        }
    }.bind(this), 10);
}, append_custom_markup: function(idx, sel) {
    var $this = $(sel),
        type = $this.attr('type'),
        $span = $this.next('span.custom.' + type);
    if ($span.length === 0) {
        $span = $('<span class="custom ' + type + '"></span>').insertAfter($this);
    }
    $span.toggleClass('checked', $this.is(':checked'));
    $span.toggleClass('disabled', $this.is(':disabled'));
}, append_custom_select: function(idx, sel) {
    var self = Foundation.libs.forms,
        $this = $(sel),
        $customSelect = $this.next('div.custom.dropdown'),
        $customList = $customSelect.find('ul'),
        $selectCurrent = $customSelect.find(".current"),
        $selector = $customSelect.find(".selector"),
        $options = $this.find('option'),
        $selectedOption = $options.filter(':selected'),
        copyClasses = $this.attr('class') ? $this.attr('class').split(' ') : [],
        maxWidth = 0,
        liHtml = '',
        $listItems, $currentSelect = false;
    if ($this.hasClass(self.settings.disable_class)) return;
    if ($customSelect.length === 0) {
        var customSelectSize = $this.hasClass('small') ? 'small' : $this.hasClass('medium') ? 'medium' : $this.hasClass('large') ? 'larg
e' : $this.hasClass('expand') ? 'expand' : '';
```

```
            $customSelect = $('<div class="' + ['custom', 'dropdown', customSelectSize].concat(copyClasses).filter(function(item, idx, arr)
 {
                if (item == '') return false;
                return arr.indexOf(item) == idx;
            }).join(' ') + '"><a href="#" class="selector"></a><ul /></div>');
            $selector = $customSelect.find(".selector");
            $customList = $customSelect.find("ul");
            liHtml = $options.map(function() {
                var       = $(this).attr('class') ? $(this).attr('class') : '';
                return "<li class='" + copyClasses + "'>" + $(this).html() + "</li>";
            }).get().join('');
            $customList.append(liHtml);
            $currentSelect = $customSelect.prepend('<a href="#" class="current">' + $selectedOption.html() + '</a>').find(".current");
            $this.after($customSelect).addClass('hidden-field');
        } else {
            liHtml = $options.map(function() {
                return "<li>" + $(this).html() + "</li>";
            }).get().join('');
            $customList.html('').append(liHtml);
 } // endif
```

Anexo 5.

```
root@ kali: ~#nikto - host https: //[redacted] --output  betazeroiris.txt
-Nikto v2.1.6-- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -
+Target IP: [redacted]get Hostname: www.ba[redacted]
+ Target Port: 443-- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -
+SSL Info: Subject: /businessCategory=Government Entity/jurisdictionC = EC / serialNumber = Government Entity / C = EC / L = Quito / O = U[redacted] Ciphers: ECDHE - RSA - AES256 - GCM - SHA384 Issuer: /C=US/O = DigiCert Inc / OU = www.digicert.com / CN = DigiCert SHA2 Extended Validation Server CA + Start Time: 2018 - 05 - 09 22: 54: 54(GMT - 4) -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -
+Server: Apache / 2.2.15(CentOS) + Cookie PHPSESSID created without the secure flag
+ Cookie PHPSESSID created without the httponly flag
+ Retrieved x - powered - by header: PHP / 7.1.16
+ X - Frame - Options header is set to allow framing from https: //forms.hubspot.com/. This is only supported in Internet Explorer and may lead to the header being ignored.
+The X - XSS - Protection header is not defined.This header can hint to the user agent to protect against some forms of XSS
+ Uncommon header 'link' found, with contents: < https: /[redacted]c/>; rel=shortlink
+The site uses SSL and the Strict - Transport - Security HTTP header is not defined.
+ The X - Content - Type - Options header is not set.This could allow the user agent to render the content of the site in a different fashion to the MIME type
+ Uncommon header 'x-robots-tag'  found, with contents: noindex, follow
+ Entry '/wp-admin/' in robots.txt returned a non - forbidden or redirect HTTP code(302)
+ "robots.txt" contains 2 entries which should be manually viewed.
+ Apache / 2.2.15 appears to be outdated(current is at least Apache / 2.4.12).Apache 2.0.65(final release) and 2.2.29 are also current.
+ Uncommon header 'tcn' found, with contents: list
+ Apache mod_negotiation is enabled with MultiViews, which allows attackers to easily brute force file names.See http: //www.wisec.it/sectou.php?id=4698ebdc59d15. The following alternatives for 'index' were found: index.php
+The Content - Encoding header is set to "deflate"this may mean that the server is vulnerable to the BREACH attack.
+ Server leaks inodes via ETags, header found with file / favicon.ico, fields: 0x57e 0x5468738bd1b80
+ Web Server returns a valid response with junk HTTP methods, this may cause false positives.
+ DEBUG HTTP verb may show server debugging information. See http: //msdn.microsoft.com/en-us/library/e8z01xdh%28VS.80%29.aspx for details.
+/php: Output from the phpinfo() function was found.
+ OSVDB - 112004: /php: Site appears vulnerable to the 'shellshock' vulnerability (http:/ / cve.mitre.org / cgi - bin / cvename.cgi ? name = CVE - 2014 - 6271).
+ OSVDB - 112004: /php: Site appears vulnerable to the 'shellshock' vulnerability (http:/ / cve.mitre.org / cgi - bin / cvename.cgi ? name = CVE - 2014 - 6278).
+ /php/php.exe ? c : \winnt\ boot.ini: Output from the phpinfo() function was found.
+ /php/php.exe ? c : \boot.ini: Output from the phpinfo() function was found.
+ OSVDB - 41361: /templates/form_header.php ? noticemsg = < script > javascript : alert(document.cookie) < /script>: MyMarket 1.71 is vulnerable to Cross Site Scripting (XSS). http:/ / www.cert.org / advisories / CA - 2000 - 02. html.
+ /php/: Output from the phpinfo()function was found.
+ OSVDB - 3092: /php/: This might be interesting...
+ OSVDB - 3092: /readme: This might be interesting...
+ OSVDB - 3092: /test/: This might be interesting...
```

```
+ /php/gaestebuch / admin / index.php: Output from the phpinfo()function was found.
+ OSVDB - 3093: /php/gaestebuch / admin / index.php: This might be interesting...has been seen in web logs from an unknown scanner.
+ /php/php 4 ts.dll: Output from the phpinfo()function was found.
+ OSVDB - 3093: /php/php4 ts.dll: This might be interesting...has been seen in web logs from an unknown scanner.
+ /php/index.php: Output from the phpinfo() function was found.
+ OSVDB - 3233: /php/index.php: Monkey Http Daemon default PHP file found.
+ OSVDB - 3268: /icons/: Directory indexing found.
+ /php/mlog.html: Output from the phpinfo() function was found.
+ OSVDB - 3396: /php/mlog.html: Remote file read vulnerability 1999 - 0346
+ /php/mlog.phtml: Output from the phpinfo() function was found.
+ OSVDB - 3396: /php/mlog.phtml: Remote file read vulnerability 1999 - 0346
+ /php/mylog.html ? screen = /etc/passwd : Output from the phpinfo()function was found.
+ /php/mylog.phtml ? screen = /etc/passwd : Output from the phpinfo() function was found.
+ OSVDB - 3233: /icons/README: Apache default file found.
+ /php/init.gallery.php ? include_class = http : //cirt.net/rfiinc.txt?/something: Output from the phpinfo() function was found.
+OSVDB - 5292: /php/init.gallery.php ? include_class = http : //cirt.net/rfiinc.txt?/something: RFI from RSnake's list (http://ha.ckers.org/weird/rfi-locations.dat) or from http://osvdb.org/
+/wp-links opml.php: This WordPress script reveals the installed version.
+ OSVDB - 3092: /license.txt: License file found may identify site software.
+ /wp-app.log: Wordpress' wp-app.log may leak application/system
details.
+ Cookie wordpress_test_cookie created without the httponly flag
+ Uncommon header 'referrer-policy' found, with contents:strict-origin-when-cross-origin
+ /wp-login/: Admin login page / section found.
+ OSVDB - 3092: /test.php: This might be interesting...
+ /wordpress/: A Wordpress installation was found.
+ 9200 requests: 0 error(s) and 52 item(s) reported on remote host
+ End Time: 2018 - 05 - 10 03: 43: 57(GMT - 4)(17343 seconds) -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- - -- -
+1 host(s) tested
```

Anexo 6.

```
root@ kali: ~#sqlmap - u "https://[redacted]c/?s=a"

http://sqlmap.org

[!] legal disclaimer: Usage of sqlmap  for attacking targets without prior mutual con
sent is illegal.It is the end user 's responsibility to obey all applicable local, st
ate and federal laws. Developers assume no liability and are not responsible for any
misuse or damage caused by this program

[ * ] starting at 18: 15: 49

[18: 15: 50][INFO] testing connection to the target URL

[18: 15: 53][INFO] checking if the target is protected by some kind of WAF/IPS/IDS

[18: 15: 54][WARNING] reflective value(s) found and filtering out

[18: 15: 55][INFO] testing if the target URL content is stable

[18: 15: 56][WARNING] target URL content is not stable.sqlmap will base the page comp
arison on a sequence matcher.If no dynamic nor injectable parameters are detected, or
 in case of junk results, refer to user 's manual paragraph '  Page comparison '

how do you want to proceed ?

[(C) ontinue / (s) tring / (r) egex / (q) uit] c

[18: 16: 10][INFO] testing  if  GET parameter 's'  is dynamic

[18: 16: 12][INFO] confirming that GET parameter 's'  is dynamic

[18: 16: 14][INFO] GET parameter 's'  is dynamic

[18: 16: 16][WARNING] heuristic(basic) test shows that GET parameter 's' might not be
 injectable

[18: 16: 18][INFO] testing  for SQL injection on GET parameter 's'

[18: 16: 18][INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'

[18: 16: 42][INFO] testing 'MySQL >= 5.0 boolean-based blind - Parameter replace'

[18: 16: 46][INFO] testing 'MySQL >= 5.0 AND error-
based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)'

[18: 16: 57][INFO] testing 'PostgreSQL AND error-based - WHERE or HAVING clause'

[18: 17: 09][INFO] testing 'Microsoft SQL Server/Sybase AND error-
based - WHERE or HAVING clause (IN)'

[18: 17: 21][INFO] testing 'Oracle AND error-
based - WHERE or HAVING clause (XMLType)'

[18: 17: 32][INFO] testing 'MySQL >= 5.0 error-based - Parameter replace (FLOOR)'
```

```
[18: 17: 34][INFO] testing 'MySQL inline queries'
[18: 17: 36][INFO] testing 'PostgreSQL inline queries'
[18: 17: 38][INFO] testing 'Microsoft SQL Server/Sybase inline queries'
[18: 17: 40][INFO] testing 'PostgreSQL > 8.1 stacked queries (comment)'
[18: 17: 49][INFO] testing 'Microsoft SQL Server/Sybase stacked queries (comment)'
[18: 17: 59][INFO] testing 'Oracle stacked queries (DBMS_PIPE.RECEIVE_MESSAGE - comment)'
[18: 18: 08][INFO] testing 'MySQL >= 5.0.12 AND time-based blind'
[18: 18: 19][INFO] testing 'PostgreSQL > 8.1 AND time-based blind'
[18: 18: 31][INFO] testing 'Microsoft SQL Server/Sybase time-based blind (IF)'
[18: 18: 46][INFO] testing 'Oracle AND time-based blind'
[18: 19: 05][INFO] testing 'Generic UNION query (NULL) - 1 to 10 columns'
[18: 21: 17][WARNING] GET parameter 's'  does not seem to be injectable
[18: 21: 17][CRITICAL] all tested parameters do not appear to be injectable.
Try to increase values for '--level'/'-risk' options if
you wish to perform more tests.
If you suspect that there is some kind of protection mechanism involved(e.g.WAF) maybe you could  try to use option '--tamper' (e.g.  '--tamper=space2comment')
[ * ] shutting down at 18: 21: 17
```

Anexo 7.

Modelo de Reporte de Incidente Computacional

Hora y Fecha actual	25/05/2020	Cuando Ocurrió el Incidente	En horas Pico de actividad de las 13:00 a 15:00
Quien reporta	Encargado de sistemas de Turno Alfonso Gonzáles		
Naturaleza del Ataque	El ataque parece ser proveniente de una fuente externa, se denegó servicios institucionales. Se comprueba el estado e integridad de la base de datos y se espera análisis de logs del servidor		
Hardware/software involucrados	El hardware involucrado son los servidores institucionales, encargados de la página principal de la Institución y el software que corre es el CMS Sharepoint		
Puntos de contacto para el personal involucrado	Se espera contactar con el encargado de soporte en *cloud* de *AWS*		
Necesita asistencia Legal (Motivo)	Si	Debido a que se sospecha que información sensible pudo ser robada o que el servidor fue comprometido.	
Detalle de su respuesta			
Afirmo que mi respuesta fue la de terminar todas las sesiones del servidor y tratar de resetear la instancia del servidor ocasionando que el servicio vuelva a la normalidad al cabo de 5 minutos, se realizó la comprobación manual del servicio y de su estado sin encontrar afectaciones, pero por falta de permisos administrativos de Linux, no se puede comprobar el estado completo del servidor. De mi consideración se realizó lo posible en mis facultades para evitar problemas con el negocio.			
Es necesario que alguna organización externa sea comunicada	Si, la Ecucert y los CERTS de otras instituciones para encontrar similitud de ataques y si tuvieron ataques similares en el rango de 1 a 3 meses		

Anexo 8.

Glosario de Términos

Ciberseguridad: “Protección de activos de información, mediante el tratamiento de las amenazas que ponen en riesgo la información que se procesa, se almacena y se transporta mediante los sistemas de información que se encuentran interconectados” (ISACA)

Ciberdelitos: Es toda aquella acción antijurídica y culpable a través de vías informáticas o que tiene como objetivo destruir y dañar ordenadores, medios electrónicos y redes de Internet.

Crackers: Se utiliza para referirse a las personas que rompen o vulneran algún sistema de seguridad. Los crackers pueden estar motivados por una multitud de razones, incluyendo fines de lucro, protesta, o por el desafío.

Sniffers: Programa informático que registra la información que envían los periféricos, así como la actividad realizada en un determinado ordenador

Spammers: Referencia a cualquier tipo de publicidad no deseada a través de internet, ya sea en correo electrónico, blogs, foros, redes sociales

Virus: Un virus informático, al igual que un virus de la gripe, está diseñado para propagarse de huésped a huésped y tiene la capacidad de replicarse a sí mismo. Del mismo modo, del mismo modo que los virus no pueden reproducirse sin una célula huésped, los virus informáticos no pueden reproducirse y propagarse sin programación, como un archivo o documento.

En términos más técnicos, un virus informático es un tipo de código o programa malicioso escrito para alterar el funcionamiento de un ordenador y que está diseñado para propagarse de un ordenador a otro.

Troyanos: Un troyano es un tipo de malware que a menudo se disfraza de software legítimo. Los troyanos pueden ser utilizados por ciberdelincuentes y hackers que intentan acceder a los sistemas de los usuarios. Los usuarios son típicamente engañados por alguna forma de ingeniería social para cargar y ejecutar troyanos en sus sistemas. Una vez activados, los troyanos pueden permitir a los ciberdelincuentes espiarle, robarle sus datos confidenciales y acceder por la puerta trasera a su sistema.

(Worms): Un gusano informático es un malware autorreplicante que se duplica a sí mismo para propagarse a equipos no infectados. Los gusanos suelen utilizar partes de un sistema operativo que son automáticas e invisibles para el usuario. Es común que los gusanos sólo se noten cuando su replicación incontrolada consume recursos del sistema, ralentizando o deteniendo otras tareas.

(“Spyware”): Spyware es software que se instala en un dispositivo informático sin el conocimiento del usuario final. Este tipo de software es polémico porque, aunque a veces se instala por razones relativamente inocuas, puede violar la privacidad del usuario final y puede ser objeto de abuso.

Creepware: Creepware es un troyano de acceso remoto (RAT) que permite a la gente hackear tu ordenador o dispositivo móvil y controlarlo a distancia. Los ejemplos más famosos de esto son el uso de la cámara y el micrófono de un dispositivo para observar y escuchar a las víctimas. Este tipo de spyware permite a los hackers, ciberdelincuentes o delincuentes en línea espiar a su familia o utilizar material grabado con fines ilegales.

Shoulder Surfing: El Shoulder Surfing utiliza técnicas de observación directa, como mirar por encima del hombro de alguien, para obtener información. El "Shoulder Surfing" es una manera efectiva de obtener información en lugares concurridos porque es relativamente fácil pararse al lado de alguien y ver cómo llena un formulario, ingresa un número de PIN en un cajero automático o usa una tarjeta telefónica en un teléfono público. El surf de hombro también se puede

hacer a larga distancia con la ayuda de binoculares u otros dispositivos para mejorar la visión. Para prevenir la navegación por el hombro, los expertos recomiendan que usted proteja el papeleo o su teclado de la vista usando su cuerpo o ahuecando su mano.

Decoy: Por lo general, un decoy es una persona, un dispositivo o un evento concebido como una distracción, para ocultar lo que un individuo o un grupo podría estar buscando. Los señuelos se han utilizado durante siglos sobre todo en la caza, pero también en tiempos de guerra y en la perpetración o resolución de delitos.

Escaneo TCP/SYN: El análisis TCP/SYN es una táctica que un hacker malintencionado (o cracker) puede utilizar para determinar el estado de un puerto de comunicaciones sin establecer una conexión completa. Este enfoque, uno de los más antiguos en el repertorio de los crackers, se utiliza a veces para realizar ataques de denegación de servicio (DoS). La exploración TCP/SYN también se conoce como exploración semiabierta.

Escaneo FIN: Se trata de un ataque más antiguo que originalmente pretendía ser un "bypass de cortafuegos furtivo" que dependía de unos pocos factores que hoy en día son poco comunes: sistemas operativos Unix antiguos, falta de cortafuegos de estado, falta de NIDS/NIPS, etc. Todavía puede ser útil cuando se prueban (es decir, como una técnica de huellas dactilares y no como un ataque per se) pilas TCP/IP completamente nuevas o novedosas.

Escaneo fragmentado: Escanea enviando fragmentos de paquetes que pueden pasar a través de simples filtros de paquetes en un cortafuego.

Eavesdropping: Eavesdropping es la interceptación no autorizada en tiempo real de una comunicación privada, como una llamada telefónica, un mensaje instantáneo, una videoconferencia o una transmisión por fax.

Suplantación de IP: El secuestro del Protocolo de Internet (IP hijacking) es una forma específica de piratería informática que utiliza las direcciones IP para mover datos a través de Internet. El hacking IP explota algunas vulnerabilidades de las redes IP en general y del Border Gateway Protocol, un sistema utilizado para designar rutas para los paquetes de datos enrutados.

Suplantación de DNS: El secuestro de DNS es un exploit malicioso en el que un hacker u otra parte redirige a los usuarios a través del uso de un servidor DNS malicioso u otra estrategia que cambia la dirección IP a la que se redirige a un usuario de Internet. El secuestro de DNS puede dejar a los usuarios sin saber a dónde van en términos de uso de servidores específicos durante una sesión de Internet.

Puertas Traseras: Una puerta trasera es una técnica en la que un mecanismo de seguridad del sistema se evita de forma indetectable para acceder a un ordenador o a sus datos. El método de acceso por la puerta trasera a veces es escrito por el programador que desarrolla el programa.

SYN Flood: Un ataque SYN es un tipo de ataque de denegación de servicio (DoS) en el que un atacante utiliza el protocolo de comunicación de Internet, TCP/IP, para bombardear un sistema objetivo con peticiones SYN en un intento de abrumar las colas de conexión y forzar a un sistema a no responder a peticiones legítimas.

Ataque LAND: En un ataque LAND (Local Area Network Denial), el atacante envía un paquete TCP SYN falso donde las IPs y puertos de origen y destino están configurados para ser idénticos. Cuando el equipo de destino intenta responder, entra en un bucle, enviándose repetidamente respuestas a sí mismo, lo que finalmente provoca que el equipo víctima se bloquee.

Tormenta BROADCAST: Una tormenta BROADCAST ocurre cuando un sistema de red es abrumado por el tráfico continuo de multidifusión o transmisión.

Cuando diferentes nodos están enviando/transmitiendo datos a través de un enlace de red, y los otros dispositivos de red están retransmitiendo los datos de vuelta al enlace de red en respuesta, esto eventualmente causa que toda la red se derrita y lleve a la falla de la comunicación de red.

SUPERNUKE: Un Nuke es un tipo de ataque anticuado de denegación de servicio (DoS) que se lleva a cabo enviando paquetes fragmentados o corruptos (normalmente ICMP) a un equipo de destino. Para cualquier máquina que ejecute un sistema operativo más antiguo y vulnerable, el envío de estos paquetes se ralentizará y, finalmente, se detendrá, provocando un fallo o una pantalla azul de muerte (Blue Screen of Death, BSoD) en el caso de Windows.

TERADROP ONE Y TWO: Un ataque TEARDROP es un ataque de denegación de servicio (DoS) que implica el envío de paquetes fragmentados a un equipo objetivo. Dado que la máquina que recibe dichos paquetes no puede volver a ensamblarlos debido a un error en el re ensamblaje de la fragmentación TCP/IP, los paquetes se superponen entre sí, lo que produce un fallo en el dispositivo de red de destino. Esto ocurre generalmente en sistemas operativos más antiguos como Windows 3.1x, Windows 95, Windows NT y versiones del núcleo de Linux anteriores a la 2.1.63.

Bombardeo de email: Mail bombing es el envío de una cantidad masiva de correo electrónico a una persona o sistema específico. Una gran cantidad de correo puede simplemente llenar el espacio de disco del destinatario en el servidor o, en algunos casos, puede ser demasiado para que un servidor lo maneje y puede causar que el servidor deje de funcionar. En el pasado, las bombas de correo se han utilizado para "castigar" a los usuarios de Internet que han violado gravemente la netiqueta (por ejemplo, personas que utilizan el correo electrónico para publicidad no deseada o spam).

Tampering: Es el acto de modificar deliberadamente (destruir, manipular o editar) datos a través de canales no autorizados. Los datos existen en dos estados; en

tránsito o en reposo. En ambos casos, los datos pueden ser interceptados y manipulados.

Ataque Java Applets: El Java Applet Attack es considerado como uno de los métodos más exitosos y populares para comprometer un sistema. Es muy popular porque podemos crear el applet de Java infectado muy fácilmente, podemos clonar cualquier sitio que queramos que cargue el applet muy rápido y con éxito porque afecta a todas las plataformas.

ActiveX: ActiveX es un conjunto de tecnologías y herramientas de programación orientadas a objetos que Microsoft desarrolló para Internet Explorer para facilitar la reproducción de medios con tecnología avanzada.

Distribuciones Linux: Linux tiene un número de versiones diferentes para adaptarse a casi cualquier tipo de usuario. Desde los nuevos usuarios hasta los usuarios más exigentes, encontrará un "sabor" de Linux que se ajusta a sus necesidades. Estas versiones se denominan distribuciones (o, en forma abreviada, "distros.") Casi todas las distribuciones de Linux pueden descargarse gratuitamente, grabarse en un disco (o unidad USB) e instalarse (en tantos equipos como se desee).

Computación Forense: El ordenador forensics es la práctica de identificar, extrayendo y considerando evidencia de medios digitales como discos duros de ordenador. Las pruebas digitales son frágiles y volátiles y requieren la atención de un especialista certificado para asegurar que los materiales de valor probatorio puedan ser aislados y extraídos de manera efectiva de una manera científica que lleve el escrutinio de un tribunal de justicia.

Criptografía: La criptografía implica la creación de códigos escritos o generados que permiten mantener la información en secreto. La criptografía convierte los datos a un formato que es ilegible para un usuario no autorizado, lo que permite

que sean transmitidos sin que entidades no autorizadas los decodifiquen a un formato legible, comprometiendo así los datos.

Amazon Web Services: Amazon Web Services (AWS) es una plataforma segura de servicios en la nube que ofrece potencia computacional, almacenamiento de bases de datos, entrega de contenido y otras funcionalidades para ayudar a las empresas a escalar y crecer. Explore cómo millones de clientes aprovechan actualmente los productos y soluciones de AWS en la nube para crear aplicaciones sofisticadas con mayor flexibilidad, escalabilidad y fiabilidad.

Kali Linux: Kali Linux es una distribución Linux basada en Debian destinada a las Pruebas de Penetración y Auditorías de Seguridad avanzadas. Kali contiene varios cientos de herramientas que se orientan hacia diversas tareas de seguridad de la información, como pruebas de penetración, investigación de seguridad, informática forense e ingeniería inversa. Kali Linux está desarrollado, financiado y mantenido por Offensive Security, una empresa líder en formación en seguridad de la información.

Parrot Security OS: Parrot es una distribución de GNU/Linux basada en Debian y centrada en pruebas de penetración, análisis forense digital, programación y protección de la privacidad.

BackBox: BackBox, también conocido como BackBox Linux, es una variante del sistema operativo Linux basado en Ubuntu. Viene con muchas herramientas para realizar pruebas de penetración de red, pruebas de seguridad y hacking ético. Se puede utilizar para rastrear paquetes en una red, aplicar ingeniería inversa a programas compilados y otras tareas que podría requerir un experto en seguridad.

Samurai Web Testing Framework: El Samurai Web Testing Framework es una máquina virtual, soportada en VirtualBox y VMWare, que ha sido preconfigurada para funcionar como un entorno de pruebas de lápiz web. La VM contiene lo

mejor de las herramientas gratuitas y de código abierto que se centran en probar y atacar sitios web.

Pentoo Linux: Pentoo es un Live CD y Live USB diseñado para pruebas de penetración y evaluación de seguridad. Basado en Gentoo Linux, Pentoo se proporciona como un Live CD instalable de 32 y 64 bits. Pentoo también está disponible como superposición para una instalación existente de Gentoo. Incluye controladores wifi con parches de inyección de paquetes, software de descifrado de GPGPU y muchas herramientas para pruebas de penetración y evaluación de seguridad.

DEFT Linux: DEFT (acrónimo de Digital Evidence & Forensics Toolkit) es una distribución hecha para Computer Forensics, con el propósito de correr en vivo en equipos sin alterar o corromper dispositivos (discos duros, pendrives, etc..) conectados al PC donde se lleva a cabo el proceso de arranque.

El sistema DEFT está basado en GNU Linux, puede ejecutarse en vivo (vía DVDROM o USB pendrive), instalado o ejecutado como un dispositivo virtual en VMware o VirtualBox.

C.A.I.N.E.: CAINE (Computer Aided INvestigative Environment) es una distribución de Linux forense digital de grado profesional. Utiliza un entorno de escritorio antiguo con herramientas especiales de primera clase. CAINE proporciona una seguridad estricta y herramientas de investigación digital incorporadas, pero es menos atractivo para los especialistas no forenses utilizarlo como un escritorio Linux de uso diario. Sin embargo, podría servir a los usuarios que están dispuestos a manejar varios inconvenientes de la interfaz.

Network Security Toolkit(NST): Es una unidad de arranque ISO Live DVD/USB Flash Drive (NST Live) está basada en Fedora. El kit de herramientas fue diseñado para proporcionar fácil acceso a las mejores aplicaciones de seguridad de red de código abierto y debería ejecutarse en la mayoría de los sistemas

x86_64. El objetivo principal del desarrollo de este kit de herramientas era proporcionar al profesional de la seguridad y al administrador de redes un conjunto completo de herramientas de seguridad de red de código abierto.

BlackArch Linux: BlackArch Linux es una distribución de pruebas de penetración basada en Arch Linux para evaluadores de penetración e investigadores de seguridad. El repositorio contiene 1981 herramientas. Se pueden instalar herramientas individualmente o en grupos. BlackArch Linux es compatible con las instalaciones existentes de Arch.

BugTraq: Bugtraq es una distribución Linux de código abierto basada en Ubuntu y Debian con núcleos PAE 3.2 y 3.4, esta distribución cuenta con todo tipo de herramientas para los mejores sistemas de auditoría. Adaptada para principiantes en Seguridad Informática de Hacking Ético, y para expertos en este campo.

Kernel: El núcleo es un programa que constituye el núcleo central de un sistema operativo informático. Tiene control total sobre todo lo que ocurre en el sistema.

ARMEL: La arquitectura armel soporta el conjunto de instrucciones ARMv4. Esta arquitectura gestiona la computación en coma flotante en un modo de compatibilidad que ralentiza el rendimiento, pero permite la compatibilidad con código escrito para procesadores sin unidades de coma flotante.

ARMHF: La arquitectura armhf soporta la plataforma ARMv7, y más, añade soporte directo de hardware en coma flotante. Esto significa que la arquitectura armhf es más rápida que la armel, pero carece de compatibilidad con las arquitecturas antiguas.

Etiquetas html: Una etiqueta HTML se define comúnmente como un conjunto de caracteres que constituyen un comando formateado para una página Web. En el

centro del HTML, las etiquetas proporcionan las direcciones o recetas para el contenido visual que uno ve en la Web.

Metaetiquetas HTML: En HTML, las metaetiquetas o metaelementos son etiquetas colocadas dentro de la sección principal del código que ayudan a definir el contenido de una página web. Por ejemplo, los motores de búsqueda de Internet utilizan una metaetiqueta de descripción para mostrar una descripción de su página en sus resultados de búsqueda. Cada metaelemento debe contener una etiqueta <meta> de apertura y cierre.

Servidor Web: Un servidor web es un sistema que entrega contenido o servicios a los usuarios finales a través de Internet. Un servidor web consiste en un servidor físico, sistema operativo del servidor (SO) y software utilizado para facilitar la comunicación HTTP. Un servidor web también se conoce como servidor de Internet.

Versión del Software: El versionado de software es el proceso de numeración de diferentes versiones de un programa de software en particular, tanto para uso interno como para la designación de la versión. Permite a los programadores saber cuándo se han realizado cambios y realizar un seguimiento de los cambios realizados en el software. Al mismo tiempo, permite a los clientes potenciales familiarizarse con las nuevas versiones y reconocer las versiones actualizadas.

SiteDigger: SiteDigger es una herramienta de código abierto para examinar la caché de Google en busca de errores, vulnerabilidades, problemas de configuración, información propietaria y Nuggets de seguridad interesantes en los sitios web. SiteDigger se ejecuta en una máquina Windows y requiere Microsoft .NET Framework v3.5. SiteDigger no requiere la clave de licencia de la API de Google. Sus características incluyen soporte de proxy, resultados en tiempo real, conjuntos de resultados configurables, etc.

CMS: Significa "Content Management System". Un CMS es una herramienta de software que le permite crear, editar y publicar contenido. Mientras que los primeros programas de CMS se utilizaban para administrar documentos y archivos informáticos locales, la mayoría de los sistemas de CMS están ahora diseñados exclusivamente para administrar el contenido de la Web.

.htaccess: El archivo .htaccess es un archivo de configuración para el servidor HTTP Apache que permite a los administradores especificar opciones para directorios individuales. La sintaxis es exactamente la misma que la de los otros ficheros de configuración de Apache. El archivo se coloca en los directorios en los que se sirven las páginas web, para ofrecer controles más precisos que el archivo de configuración de todo el sistema, httpd.conf.

Raíz del sitio: Es la carpeta/directorio en un servidor Web que contiene las páginas Web visibles para el público. También llamado "docroot", los nombres de las carpetas son a menudo /www/public o /public_html.

NetCraft: Netcraft proporciona servicios de seguridad en Internet, incluyendo servicios antifraude y antiphishing, pruebas de aplicaciones y escaneo PCI.

Apache: Apache Web Server es un software de código abierto de creación, despliegue y gestión de servidores web. Inicialmente desarrollado por un grupo de programadores de software, ahora es mantenido por la Apache Software Foundation.

Linux CentOS: La distribución CentOS Linux es una plataforma estable, predecible, manejable y reproducible derivada de las fuentes de Red Hat Enterprise Linux (RHEL).

CMS Wordpress: Software web gratuito diseñado para crear sitios web o blogs basados en plantillas.

Azure: Microsoft Azure es una plataforma como solución de servicio (PaaS) para construir y alojar soluciones utilizando los productos de Microsoft y en sus centros de datos. Se trata de un conjunto completo de productos en la nube que permite a los usuarios crear aplicaciones de clase empresarial sin tener que construir su propia infraestructura.

DomainTools: DomainTools ayuda a los analistas de seguridad a convertir los datos de amenazas en información sobre amenazas. Se toman indicadores de su red, incluyendo dominios e IPs, y se conectan con casi todos los dominios activos en Internet. Estas conexiones informan las evaluaciones de riesgo, ayudan a perfilar a los atacantes, guían las investigaciones de fraude en línea y asignan la actividad cibernética a la infraestructura del atacante.

NMap: Nmap ("Network Mapper") es una utilidad gratuita y de código abierto (licencia) para la detección de redes y la auditoría de seguridad. Muchos sistemas y administradores de red también lo encuentran útil para tareas como el inventario de la red, la gestión de los programas de actualización de servicios y la supervisión del tiempo de actividad del host o del servicio. Nmap utiliza paquetes IP sin procesar de formas novedosas para determinar qué hosts están disponibles en la red, qué servicios (nombre y versión de la aplicación) ofrecen esos hosts, qué sistemas operativos (y versiones de SO) están ejecutando, qué tipo de filtros de paquetes/ cortafuegos están en uso, etc.

Topología de Red: La topología de red es el patrón interconectado de los elementos de red. Una topología de red puede ser física, configuración de hardware de mapeo, o lógica, mapeando el camino que los datos deben tomar para viajar por la red.

Código Abierto: El código abierto es una filosofía que promueve el libre acceso y distribución de un producto final, generalmente software o un programa, aunque puede extenderse a la implementación y diseño de otros objetos.

Metadato: Los metadatos son datos sobre datos. En otras palabras, son los datos que se utilizan para describir el contenido de otro artículo. El término metadatos se utiliza a menudo en el contexto de las páginas web, donde describe el contenido de una página para un motor de búsqueda.

ISECOM: ISECOM es una comunidad abierta y una organización sin ánimo de lucro registrada oficialmente en Cataluña, España. ISECOM tiene oficinas en Barcelona, España y en Nueva York, Estados Unidos.

Payload: Un payload se refiere al componente de un virus informático que ejecuta una actividad maliciosa. Además de la velocidad de propagación de un virus, el nivel de amenaza de un virus se calcula en función de los daños que causa. Los virus con payloads más potentes tienden a ser más dañinos.

Búfer: Un búfer es un área de retención temporal de datos mientras espera a ser transferido a otra ubicación. Normalmente se encuentra en la memoria RAM. El concepto de búfer se desarrolló para evitar la congestión de datos de un puerto de transferencia entrante a uno saliente.

C/C++: Lenguaje de Programación.

"Watering Hole": Un ataque Watering Hole es una vulnerabilidad de seguridad en la que el atacante intenta comprometer a un grupo específico de usuarios finales infectando sitios web que se sabe visitan los miembros del grupo. El objetivo es infectar la computadora de un usuario objetivo y obtener acceso a la red en el lugar de trabajo del usuario objetivo.

ActionScript: ActionScript es un lenguaje de programación y secuencias de comandos orientado a objetos diseñado para proporcionar capacidades interactivas sofisticadas a la plataforma Adobe Flash Player. La sintaxis de ActionScript es similar a la de JavaScript (ambas están basadas en el mismo estándar ECMAScript).

XPath: XPath es un lenguaje que gira alrededor del uso del lenguaje de programación Extensible Markup Language (XML). Usando XPath, los desarrolladores u otros pueden obtener cierta información de documentos XML, incluyendo diferentes valores y variables, y la ubicación de elementos específicos dentro de un documento XML.

Defacement: Defacement es una forma de vandalismo en la que un sitio web está marcado por hackers o crackers que están tratando de dejar su marca. Por lo general, la desfiguración del sitio web se utiliza para enmascarar un delito mayor que se comete entre bastidores

Mensajería Web: La Mensajería Web (también conocida como Mensajería de Documentos Cruzados) permite que las aplicaciones que ejecutan diferentes dominios se comuniquen de manera segura.

CGI: CGI (Common Gateway Interface) es una norma que permite a Apache ejecutar algunos programas, que pueden ser escritos en cualquier lenguaje de programación (Bash, C, Java, Perl, PHP, Python, etc), desde el momento en que es ejecutable y respeta ciertas restricciones de entrada/salida.