Seguridad en Servidores Web

Antonio Miguel Pozo Cámara
apozoetsiit@correo.ugr.es
Jose Ignacio Recuerda Cambil
ignaciorecuerda@correo.ugr.es
Ignacio Romero Cabrerizo
nachorc@correo.ugr.es

Índice

1. *Breve nota introductoria:*

¿Por qué se siguen realizando ataques a servidores y webs?

Actualmente la mayor parte de población con acceso a la red, hace uso frecuentemente del navegador para realizar búsquedas de información, realizar pagos y transacciones online, comprar o compartir contenido por las redes sociales. Este hábito por tanto se resume, en un acceso a gran número de páginas y de servicios web.

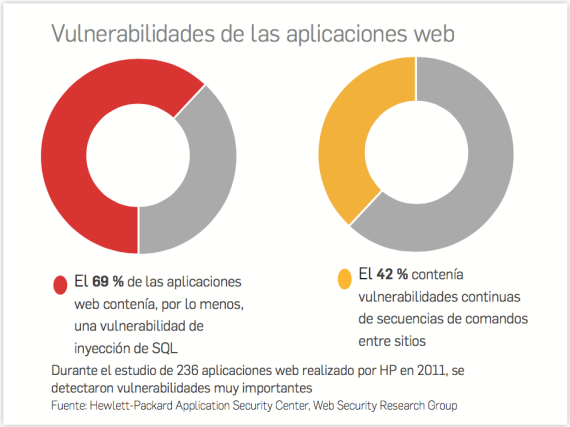
Los servidores web se han convertido en un "escaparate" para muchas empresas. Cualquier organización dispone de su página web para divulgar su negocio o imagen de empresa, páginas que en muchos casos, cuentan con contenido dinámico y scripts que permiten un uso avanzado de las mismas. Estos servidores a su vez cuentan con una serie de herramientas (servicios) que permiten servir y mantener sus páginas al igual que información de usuarios y empresa, pública y privada (datos que pueden verse comprometidos).

Con todo esto, los servidores web donde se alojan tales páginas junto con la información sensible, han pasado a ser un blanco fácil para atacantes de diversa índole. Hasta los sistemas mejor diseñados y caros, en cuanto a métodos de defensa e infraestructura de red, pueden verse totalmente comprometidos por un fallo de seguridad en la programación web.

Las nuevas técnicas de ataque y vulnerabilidades que afectan a los servidores web hacen que los cortafuegos tradicionales ya no sean suficientes para proteger las redes actuales. Al integrar la protección para servidores web en las infraestructuras de seguridad informática, se crean entornos más seguros [7].

Principales riesgos para la seguridad de aplicaciones web:

1. Inyección
2. Secuencias de comandos entre sitios (XSS)
3. Referencias directas no seguras a objetos
4. Pérdida de autenticación y gestión de sesiones
5. Falsificación de peticiones entre sitios (CSRF)
6. Errores en la configuración de la seguridad
7. Almacenamiento criptográfico poco seguro
8. Protección insuficiente de la capa de transporte
9. Redirecciones y desvíos no validados
10. Falta de restricción del acceso a direcciones web



¿Responsables?

Desarrolladores y administradores web que en algunos casos se mantienen ajenos a la realidad actual, en la que conviven hackers y lammers[1] que tienen gran libertad y se mueven a sus anchas en este entorno.

La mayor parte de estos ataques, en la actualidad, vienen como consecuencia de una mala configuración del servidor o un mal diseño del mismo, así como de fallos de programación derivados de los ajustados Service Level Agreement (SLA) al que se enfrentan los desarrolladores de los portales web.
La falta o escasez de un *Pentesting continuo* por parte de las empresas que evitaría muchos de los problemas de seguridad antes mencionados.

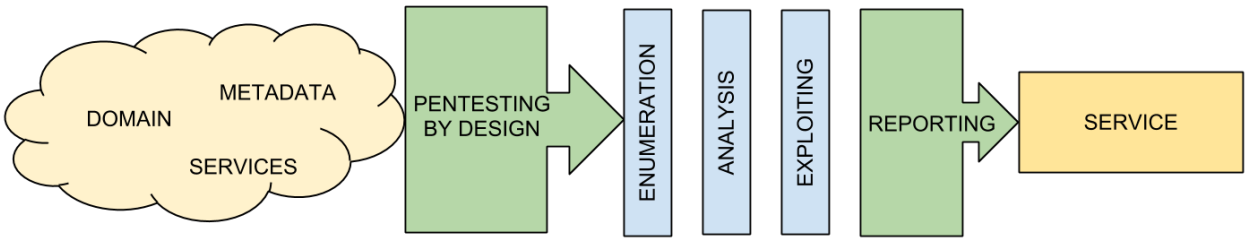
[1] Persona que presume de tener unos conocimientos que en realidad no tiene.

PENETRATION TEST (PENTESTING)

Una prueba de penetración (pentesting) es un ataque a un sistema informático con la intención de encontrar las debilidades de seguridad que podrían acabar con un acceso al sistema, su funcionalidad y sus datos. Un sistema capaz de enumerar, analizar y explotar los activos de una organización con el fin de encontrar agujeros de seguridad que podrían ser explotados por otros usuarios con fines maliciosos,

Con las auditorías de seguridad suelen aparecer vulnerabilidades menores que ayudan a preparar ataques más importantes al sistema.

El ***Pentesting by Design*** necesita un dimensionamiento de la memoria, el almacenamiento y el ancho de banda en las comunicaciones en los sistemas diseñados a priori para dar calidad de servicio a los usuarios, más un porcentaje destinado al ataque continuo de los atacantes, más un porcentaje necesario para que los auditores puedan realizar el proceso de pentesting continuo desde el primer día.



Visión global de Pentesting by Design

Fuente: ***http://blog.elevenpaths.com/2013/06/faas-vision-global-de-pentesting-by.html***

2. *Tipología de atacantes y de ataques:*

● *Tipos de atacantes*

Hay que diferenciar entre 2 tipos de atacantes, conocidos como *White Hack* y *Black Hack*.
Los White Hack, son aquellos profesionales que prueban la vulnerabilidad de un sistema, y realizan un reporte detallado que servirá para mejorar el sistema, son conocidos como auditores de seguridad informática.

Los Black Hack, vulneran un sistema, con la intención de extraer información sensible.

● *Tipología de ataques*

1. DDOS
2. Fuerza Bruta
3. Inyección SQL (SQL)
4. Phishing
5. XSS (Cross Site Scripting)

- ## Ataques DDOS:

Este tipo de ataque consigue que el sistema esté inaccesible para los usuarios que solicitan su servicio. Consiste en realizar un gran número de accesos al sistema servidor provocando la saturación del ancho de banda de la víctima o la sobrecarga de su sistema. El fin que persiguen este tipo de ataques es que el sistema no pueda seguir prestando sus servicios por la saturación que se le produce. Estos ataques se suelen producir desde un gran número de puntos de conexión y son muy usados por el colectivo Anonymous sobre webs gubernamentales.

Se distinguen 3 tipos de ataques DDOS:

1. Ancho de banda: Ataque que consiste en saturar la capacidad de la red del servidor, haciendo que sea imposible llegar a él.
2. Recursos: Ataque que consiste en agotar los recursos de la máquina servidora, impidiendo que esta pueda responder a las peticiones legítimas.
3. Explotación de fallos de software: Categoría de ataque que explota fallos en el software que inhabilitan el equipo o toman su control.

- ## Ataques de fuerza bruta:

Este tipo de ataques permiten ir comprobando todas las combinaciones posibles para poder acceder a un recurso.
Las posibles combinaciones de los parámetros a utilizar pueden proporcionarse mediante diccionarios (ficheros de texto) y su uso puede ser general, específico o especialmente creados para tal fin. También es posible proporcionar valores sistemáticos y secuenciales que cubran todas las combinaciones posibles.

Hay varias aplicaciones que realizan este tipo de ataque, las más populares son las siguientes:

- ➢ BrutusAE.
- ➢ thc-hydra.
- ➢ Medusa.

A continuación vamos a comentar thc-hydra, ya que viene instalada en todas las distribuciones específicas de seguridad y por soportar un gran número de protocolos.
La aplicación thc-hydra es muy sencilla. En la pestaña target (Imagen 1) se indica el objetivo mediante el nombre o dirección IP del servidor y el puerto dónde se va a realizar el ataque de fuerza bruta.

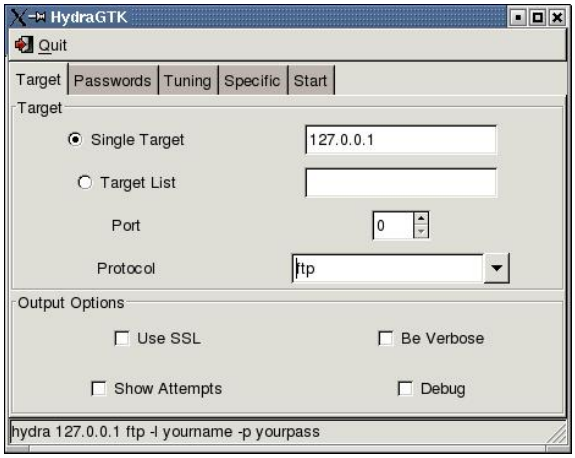


Imagen 1

Fuente: ***https://www.thc.org/thc-hydra/***

En la pestaña Password (Imagen 2) se indican los usuarios y contraseñas a utilizar. Para ello se puede indicar un usuario o lista de usuarios y para las contraseñas se puede utilizar un archivo de texto que usará a modo de diccionario para probar todas las contraseñas posibles. El diccionario podemos crearlo nosotros o bien bajarlo de Internet.

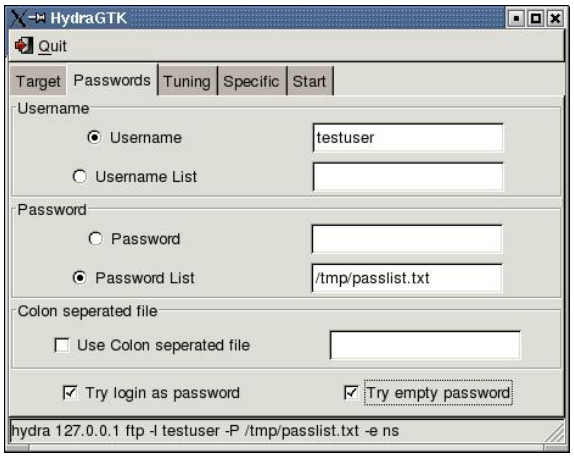


Imagen 2

Fuente: ***https://www.thc.org/thc-hydra/***

En la pestaña Start (Imagen 3) solo hay que pulsar el botón “Start” para iniciar el ataque de fuerza bruta.

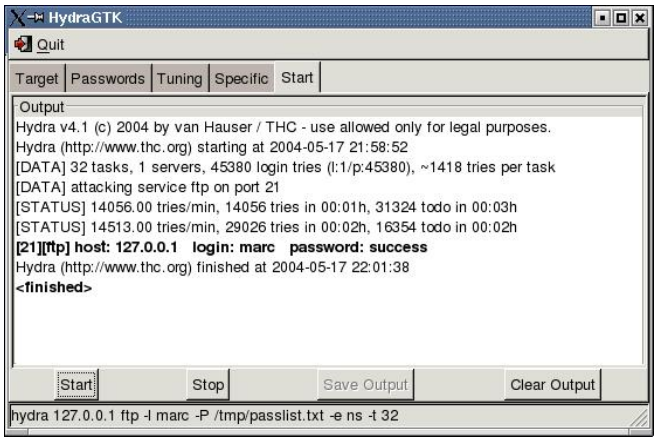


Imagen 3

Fuente: ***https://www.thc.org/thc-hydra/***

Esta técnica no suele ser recomendable por varios motivos:

- Puede causar mucho estrés en el sistema analizado.
- Puede causar bloqueos de cuentas.
- Necesita mucho tiempo.
- Es poco elegante.

Aunque también hay que decir que esta técnica en determinadas ocasiones es útil:

- Las políticas de contraseñas incorrectas son habituales, es relativamente fácil encontrar credenciales débiles o predecibles.
- Cuando no exista otra vía más rápida, eficiente y elegante.

- **Ataques de inyección SQL**:

Se entiende por inyección SQL el hecho de insertar una serie de sentencias SQL en una consulta mediante la manipulación de la entrada de datos de una aplicación.
Estos ataques se realizan sobre webs que interactuan con bases de datos, sino se realiza un filtro de la información enviada por los usuarios, estos pueden insertar código SQL a través de los datos de entrada que se envían al servidor y que, a través de este código insertado, se podría acceder a información almacenada en el servidor. Esta inclusión de código consiste en poner caracteres especiales en los campos que se rellenan en formularios de la web, ejemplo de ello sería añadir en un campo contraseña: 1234 ' OR '1' = '1. Esto enviará al servidor el dato '1234' OR '1' = '1' lo cual sería una condición que siempre es verdadera. Un tipo de ataque es el llamado "Ataque a ciegas por inyeción de SQL" este tipo de web se delata de ser vulnerables al no mostrar un mensaje de error al ejecutar una sentencia SQL errónea, por ejemplo introduciendo una comilla (') en el campo de texto, pudiendo realizarse pruebas hasta encontrar el nombre o tablas sobre los que se pueden actuar.
A continuación puede ver una serie de inyecciones SQL para base de datos MySQL:

- Eliminar tablas Usuarios → **'; drop table Usuarios**
- Muestra un mensaje de error en el cual muestra el nombre de la tabla y la primera columna → **'having 1=1**
- Se obtiene un usuario que empiece por 'a' → **' union select min(Usuario),'1','1','1','1','1','1','1' from Usuario > 'a'**

Entre las bases de datos susceptibles a estos ataques se encuentran: MySQL,Oracle,PostgresoMSSQL.
(NO PROBAR ESTA TÉCNICA, SÓLO SE NOMBRA PARA EXPLICAR ESTE TIPO DE ATAQUE)

- **XSS (Cross Site Scripting)**

También llamado *HTML injection*. Esta vulnerabilidad compromete la seguridad del usuario, no la del servidor. Esta técnica consiste en inyectar código HTML o JavaScript en una web con el fin de que sean ejecutados por el navegador web de los usuarios finales en el momento de cargar la página alterada. Por esta misma razón a este tipo de ataques se le denomina "ataque del lado del cliente", ya que no se ejecutaría en el servidor. Con este ataque se suele acceder a la información contenida en las cookies o token de sesión que se han usado en este sitio web para después usarlo sin el permiso de dicho usuario.
Existen dos tipos de ataques XSS:

- por un lado están los XSS permanentes, este tipo se suele dar en foros y formularios. La información proporcionada se almacena en la base de datos y se mostrará a los usuarios que visiten la página, por eso se dice que la vulnerabilidad "persiste".
- y por otro lado los XSS no permanentes, esta vulnerabilidades muy fácil de encontrar en motores de búsqueda, formularios, URL, cookies, programas flash o incluso videos. Se suele utilizar en páginas no estáticas. En cambio es más difícil de utilizar ya que hay que conseguir que alguien entre en el enlace malicioso.

Se ha realizado una prueba de esta vulnerabilidad en un formulario web propio del que adjunto las imágenes para que se pueda entender mejor el concepto de esta vulnerabilidad:

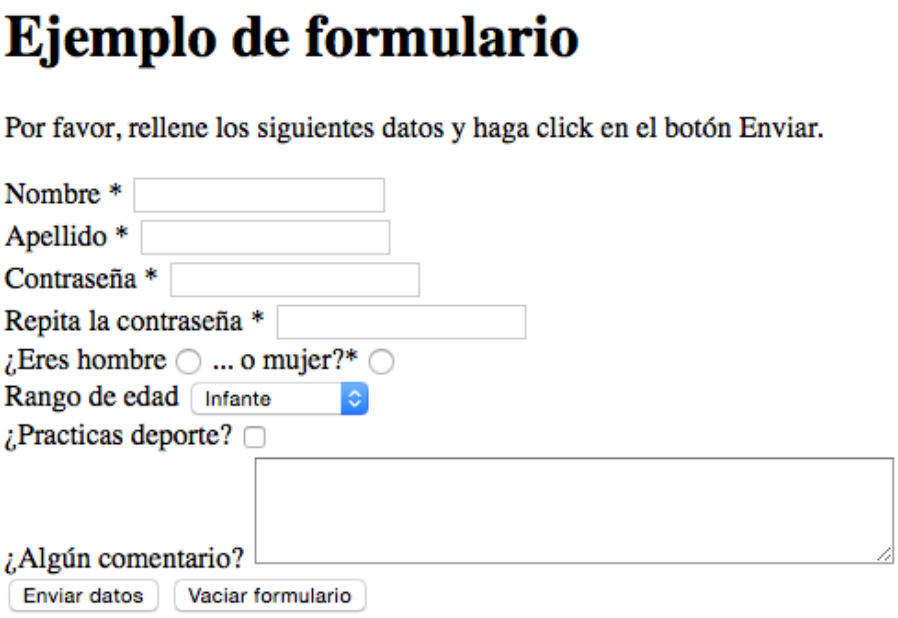


Imagen 4 (formulario inicial)

Imagen 5 (escribimos código en un campo de texto)

Escribo en el campo exactamente lo siguiente:

```
"/><p> Esto es una prueba de vulnerabilidad XSS no permanente</p><BR
```

Y el resultado obtenido es el mostrado en la siguiente imagen, como podemos ver hemos conseguido modificar la web.

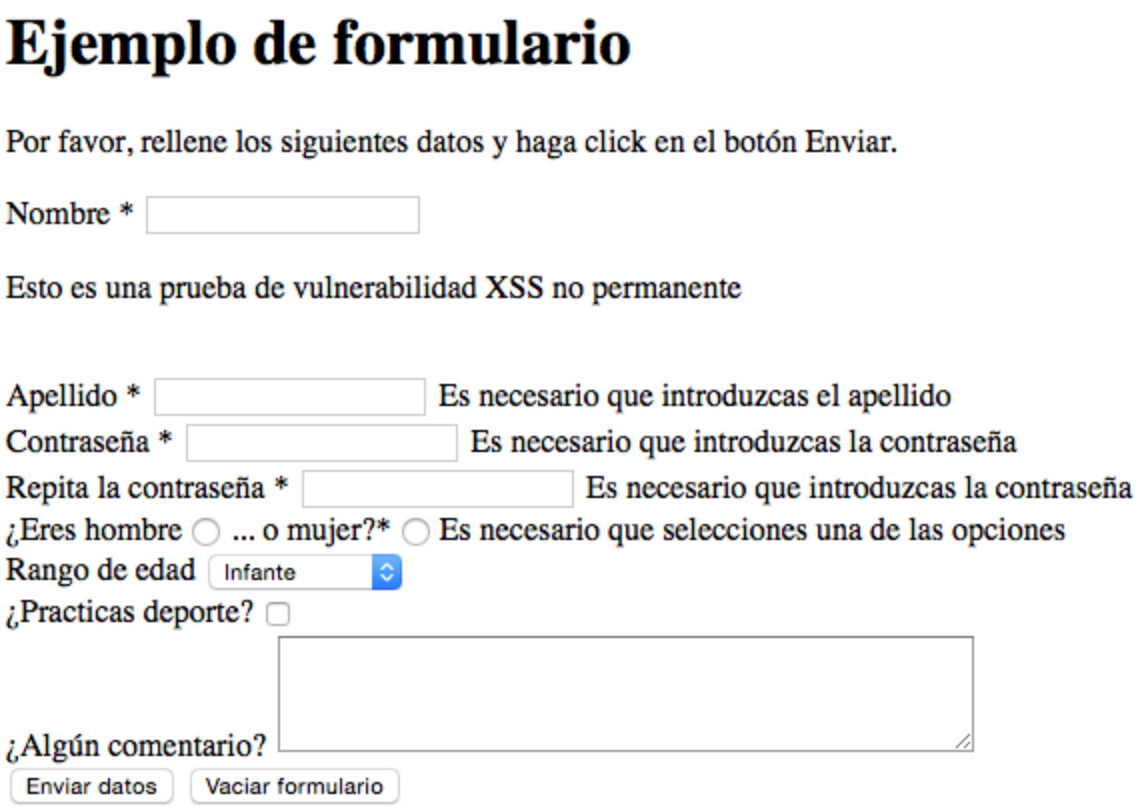


Imagen 6 (escribimos código en un campo de texto)

- **Phishing**

Esta técnica consiste en captar información de los usuarios haciendo creer que están en una página de confianza para ellos (porque tiene un aspecto visual idéntico) no estando en ella, entonces el usuario facilita los datos de acceso y una vez hecho esto, el Hacker ya puede acceder con la identidad del usuario al lugar legítimo. En el mundo de la informática esto es más conocido como suplantación de la identidad.
El mecanismo más usado para este tipo de ataques es la generación de un correo electrónico falso que simula proceder de una empresa de confianza. Este correo electrónico contendrá enlaces que replican varias páginas web con el aspecto y funcionalidad de esa supuesta empresa. Si el receptor de este correo confía en que el mensaje procede de esta fuente probablemente introducirá información sensible en un formulario falso. [5]

3. Búsqueda de vulnerabilidades y su análisis (Software)

Actualmente existen herramientas que permiten analizar y buscar fallos de seguridad en los servidores Web. Las vulnerabilidades que se encuentran con más frecuencia generalmente brindan a los atacantes el control total del sistema o la denegación del acceso a sus servicios.
Algunas de estas herramientas más conocidas son las siguientes:

- **Nikto**
  - Es un escáner de vulnerabilidades de servidores web bajo la licencia GPL. Permite obtener un informe detallado sobre un sitio web para poder evitar posibles ataques.
  - Es capaz de detectar fallos de:

- Problemas de configuración.
- Archivos y scripts por defecto.
- Archivos y scripts inseguros.
- Versiones desactualizadas de software.

Ejemplo de uso de Nikto:

- El uso esencial de Nikto se realiza mediante el siguiente comando:

```
nikto.pl -h hostAEscanear*
```

** El host puede ser una dirección IP o un dominio (Ej.: 127.0.0.1 ó http://www.dominio.xx)*

- El puerto por defecto que usa Nikto es el 80, si quisiéramos cambiarlo bastaría hacer:

```
nikto.pl -h host -p puerto
```

- Nikto realiza un escaneo común HTTP, si falla, intenta realizar uno mediante SSL (HTTPS)
- Algunos de los parámetros destacados de nikto son:
  - *-evasion:* Habilita la detección de intrusiones por medio de técnicas de evasión.
  - *-findonly:* Solo escanea puertos HTTP(S) sin realizar comprobación.
  - *-format:* Guarda el log con un formato específico. Estos pueden ser csv, html, xml y txt.

- **Httpanalizer (Escaneo)**
  - IEInspector HTTP Analyzer es una herramienta práctica que permite monitorizar, rastrear, depurar y analizar tráfico HTTP / HTTPS en tiempo real. Es utilizado por las empresas líderes en la industria, incluyendo Microsoft, Cisco o Google.
  - Permite actuar como *proxy* entre el cliente y el servidor para analizar el tiempo real las peticiones y respuestas que se realizan sobre una web.
  - Construye paquetes que proporcionan multitud de métodos de intrusión en la web.

- **Archilles**
  - Archilles es un software diseñado para ser una herramienta de pruebas para aplicaciones basadas en web. Es un servidor proxy que se interpone entre un navegador cliente y un buscador de servidores durante una sesión HTTP. Intercepta y descifra los datos para el usuario en un log.
  - Es capaz de hacer esto mediante el filtrado de datos deteniéndolo y permitiendo al usuario editarlo antes de que se reenvíe.

- **ZAP (Zed Attack Proxy)**
  - Es un proyecto opensource que permite actuar como *proxy* intermedio entre el navegador y el servidor Web.
  - Está diseñado para ser utilizado por personas con una amplia gama de experiencia en seguridad y, como tal, es ideal para desarrolladores y probadores funcionales que son nuevos en la introducción de pruebas.
  - ZAP ofrece escáneres automatizados, así como un conjunto de herramientas que le permiten encontrar vulnerabilidades de seguridad de forma manual.
  - OWASP (Open Web Application Security Project), comunidad desarrollada por ZAP que realiza aportes importantes en la seguridad.

- **Web Crawlers**
  - Se dedican a indexar y recoger información sobre todas las webs del mundo y la información que contienen de forma metódica y automatizada.
  - En ocasiones, en aplicaciones web que tienen una base de datos muy grande o que habitualmente tienen una alta carga en el servidor (bien por afluencia de usuarios, por procesos internos, etc) las indexaciones de este tipo de programas pueden generar

sobrecarga en el sistema, volviendo más lenta la respuesta de los servicios, o incluso derivar en una caída del servicio.

- **Nmap + NSE Vulscan (Escaneo)**
  - Utilidad de código abierto para la detección de redes y auditoría de seguridad.
  - Su principal uso es el escaneo de puertos y descubrir servicios o servidores de una red informática.
  - Nmap utiliza paquetes IP para determinar qué servicios están disponibles y sirve para asegurarse de que sólo los puertos permitidos están disponibles para el público.
  - **NSE Vulscan (Nmap Script Engine):** es un escaneador que hace uso de las bases de datos de forma offline junto con NMAP.

Ejemplos de uso de NMAP:

- Obtener información de un host remoto y detección del SO (escaneo sigiloso sin ping):

```
nmap -sS -P0 -sV -O [dirección]
```

- Listar servidores con un puerto específico abierto:

```
nmap -sT -p 80 -oG – 192.168.1.* | grep open
```

- Detectar IP's activas en una red:

```
nmap -sP 192.168.0.*
```

- Escanear en busca de AP falsos:

```
nmap -A -p1-85,113,443,8080-8100 -T4 –min-hostgroup 50 –max-rtt-timeout 2000 –initial-rtt-timeout 300 –max-retries 3 –host-timeout 20m –max-scan-delay 1000 -oA wapscan 10.0.0.0/8
```

- Señuelo durante escaneo de puertos para evitar detección:

```
(como root) nmap -sS 192.168.0.10 -D 192.168.0.2
```

- **OpenVAS (Escaneo)**
  - Su propuesta inicial fue para pruebas de penetración en sistemas pero su marco de servicios y herramientas comprenden una completa suite software para el análisis, escaneo y gestión de vulnerabilidades.
  - Características principales:
    - Escaneo concurrente de múltiples nodos y temporizado.
    - Soporte SSl y WMI.
    - Servidor web integrado.

- **Nessus (Escaneo)**
  - Es un programa de escaneo de vulnerabilidades.
  - Escanea puertos abiertos con nmap o su escaneador de puertos propio para intentar atacarlos.
  - Nessus permite escanear:
    - Vulnerabilidades que permiten a un atacante remoto controlar o acceder a datos confidenciales en el sistema.
    - Mala configuración (parches faltantes...).
    - Contraseñas por defecto. Algunas contraseñas comunes, (en blanco o vacías) en algunas cuentas del sistema.
    - Denegaciones de servicio contra la pila TCP/IP mediante paquetes malformados.

4. *Técnicas de prevención:*

- Control de los archivos publicados
    - Denegar acceso por defecto: Uno de los problemas más frecuentes es que no se aplican las directivas adecuadas para controlar los archivos que se ubican en el directorio en el que se publican las páginas web. Este directorio está especificado mediante la directiva DocumentRoot. Si no se configura adecuadamente la directiva, es posible que existan errores de configuración que puedan posibilitar el acceso a otros directorios o subdirectorios.
    - Revisar las directivas Alias: Apache permite mediante la directiva <alias> asignar cualquier tipo de ruta en el sistema sea accesible desde la web. Se deben revisar estas reglas para verificar que no existan directorios asociados al servicio web que puedan ser mostrados.
    - Evitar la resolución de enlaces simbólicos[2]: Apache puede recibir peticiones a un archivo que es, a nivel de sistema operativo del servidor, un enlace simbólico y devolver el archivo apuntado por él aunque se encuentre fuera del DocumentRoot (directorio donde se almacenan los documentos web). Esto implica que un posible atacante, que tenga permisos de escritura sobre el DocumentRoot, cree un enlace simbólico a archivos contenidos fuera del DocumentRoot para luego acceder a ellos a través del navegador.
- Ocultar información
    - Deshabilitar el listado de ficheros: Si se le envía a Apache una URL que solicita un directorio y no un archivo concreto, Apache permite mostrar los contenidos de este directorio. Esta funcionalidad puede ser aprovechada por un atacante para descubrir archivos que el administrador del sitio no tenía intención de publicar.
    - Limitar el acceso a archivos por extensión: Hay archivos en el DocumentRoot como los .htaccess y .htpasswd que son delicados; para restringir su acceso se utilizan las directivas Files y FilesMatch.
    - Información en la cabecera “Server”: Las cabeceras http pueden contener información del software que ejecuta el servidor web. Esta información es fácilmente accesible mediante un analizador de protocolos. Para restringir esta información hace uso de la directiva ServerTokens.
    - Información en páginas generadas por el servidor: Cuando se intenta acceder a un recurso del servidor que no existe, el propio servidor genera y devuelve su propia información de error para comunicárselo al cliente. Dicha información puede comprometer por ejemplo el correo del administrador o listados del directorio FTP. Para no mostrar esa información se utilizan las directivas ServerAdmin y ServerSignature.
- Interfaz de entrada común (CGI) y módulos
    - La mayor parte de portales en la actualidad ofrecen contenido dinámico, es decir, hay scripts que el servidor tiene que ejecutar. La inclusión de errores en dichos scripts puede ser críticos. Típicos errores pueden ser no validar la entrada, errores en la gestión de las sesiones o condiciones de carrera etc.
- Autenticación y autorización
    - Por contraseña: El sistema ha de estar protegido por contraseña para todos los roles. Para identificar a administradores se suelen utilizar archivos sin extensión, archivos DBM o DLAP. Es imprescindible que estos archivos se encuentren fuera del DocumentRoot. El motivo por el cual estos archivos se utilizan para identificar administradores es que administradores de sistemas no suele haber muchos. Identificar usuario y contraseña de muchos usuarios en estos archivos es una tarea costosa y puede llegar a saturar el servidor. Para hacer esta tarea se suelen utilizar métodos de autenticación a nivel de aplicación tales como las cookies.
    - Autorización a grupos: Con la directiva Require valid-user podemos permitir el acceso a cualquier usuario válido. Usarla con cautela. Con la directiva Require group se permite el acceso a determinados grupos de usuarios. De igual manera ha de extremar la precaución.

[2] Acceso a un directorio o fichero que se encuentra en un lugar distinto dentro de la estructura de directorios

    - Otros factores de autenticación: a partir de la versión 2.4 de Apache, la funcionalidad de la directiva Require se ha ampliado y ahora, entre otras cosas, permite especificar rangos de IPs autorizadas que antes se implementaban mediante la directiva Allow. Con ello permitiremos añadir host autorizados para conectarse con ese servidor. Esto es útil si se dispone de una IP fija.
- Comunicación HTTPS y autenticación con certificado de cliente
    - Ventajas de utilizar HTTPs con respecto a HTTP: https proporciona comunicación cifrada que por tanto imposibilita que se capture la información durante todo el proceso de envío y recepción. Se autentifica el servidor, lo que implica que si su certificado ha sido emitido por una CA de confianza, se evitarán los ataques de tipo man-in-the-middle.
    - Por contra, el trabajo de cifrado y descifrado lo realiza el servidor, lo que supone una sobrecarga del mismo con respecto a http. Este problema se debe tener en cuenta en sitios donde concurren un volumen amplio de visitas.
    - Configuración segura de HTTPs: Al crearse el archivo de clave privada, puede especificarse una contraseña con la que se cifrará el archivo. Al iniciarse Apache, se solicitará la contraseña al administrador para que Apache pueda descifrar y utilizar la clave privada. Pero, ¿y qué pasa si el archivo no se encuentra cifrado?, el servicio de Apache debe tener permisos de usuario “root” para verlo. Para hacer efectiva la configuración se hace uso de las directivas SSLCipherSuite y SSLHonorCipherOrder. Lo más común es que dentro de un sistema, tengamos partes http para información no confidencial y https para información confidencial. Esto podría permitir que partes accesibles por HTTPS lo fueran también por HTTP. Hay que tener en cuenta este problema y cambiar la configuración de apache con al directiva SSLRequireSSL para solucionarlo.
      Para comprobar la seguridad de la configuración HTTPS se puede hacer uso del portal www.trustworthyinternet.org/ssl-pulse o de la herramienta sslyze.
    - Autenticación del usuario por certificado: Además de la autenticación por contraseña, Apache permite la autenticación por certificado: que contiene piezas de información X.509 sobre su poseedor y sobre la autoridad certificadora que lo firma a parte de la clave pública del propietario y la firma de la CA.
    - Criterios compuestos de autorización: Son combinaciones de medidas de seguridad ya mencionadas para aumentar la seguridad. Por ejemplo hacer uso de la directiva Require y la nueva forma de definir expresiones lógicas.
- Monitorización
    - Monitorización del servicio: Existen módulos de Apache para realizar esta función. Un ejemplo de ellos es el módulo mod_status, que se utiliza para supervisar la carga de los procesos e hilos, consumo de recursos, caídas de servicio o demonio… Una herramienta que se puede utilizar para detectar este tipo de eventos es Monit [1].
    - Configuración y revisión de “logs”: Existen dos tipos de logs, uno almacena la información sobre las URLs a las que se han accedido y otro recoge los errores de Apache. Es obvio que los del tipo uno almacenarán la dirección del sitio web en cuestión. Los del tipo dos, a su vez, pueden ser de error o de acceso.
    - Detección de ataques: Es la última fase del proceso de monitorización. Existe multitud de herramientas para llevar a cabo esta acción. Por mencionar alguna, se podría hacer uso de un Web Application Firewall (WAF) como mod_security, que detecta por defecto los siguientes ataques:
        - Violaciones de HTTP.
        - IPs de listas negras.
        - Malware utilizando el API de Google Safe Browsing.
        - Prevención de ataques de denegación de servicio.
        - Ataques web comunes.

5. SOLUCIÓN A ATAQUES:

- Recogida de evidencias**:**

El primer paso para dar solución a un ataque es recoger pruebas de que efectivamente hemos sido atacados. Para ello debemos analizar los siguientes ámbitos.

  - Análisis de aplicaciones
  - Análisis de bases de datos
  - Análisis de red
  - Análisis de memoria RAM
  - Análisis de SWAP, paginación
  - Análisis de discos físicos
  - Análisis de dispositivos móviles
  - Análisis de impresoras

- Respuesta a incidentes**:**

Ya que sabemos que nuestro sistema ha sido atacado y tenemos evidencias de ello, hay que facilitar la recogida de información por parte del personal autorizado para hacer dicha labor. La evidencia digital es bastante frágil y puede ser alterada, dañada o incluso destruida con cierta facilidad, lo que puede conducir a un análisis que lleve a conclusiones incorrectas. Las evidencias deben ser recogidas de tal forma que sean protegidas y adecuadamente manipuladas manteniendo en todo momento su fiabilidad. De todos los equipos implicados se debe recoger la siguiente información:

  - Titular del equipo afectado y organización a la que pertenece el equipo
  - Tipo de BIOS e información hardware del modelo
  - Sistema operativo y versión del mismo
  - Tiempo que lleva el equipo sin apagarse
  - Estructura completa de directorios del sistema
  - Ubicación del archivo de paginación
  - Tipo y número de procesadores del equipo, tamaño de la memoria RAM

Una vez recogidas las evidencias y la información del equipo o los equipos implicados en el ataque debemos dar solución al mismo. La diversidad y complejidad de los mismos hace inviable la exposición de la solución en éste documento. Sin embargo, se publicarán herramientas, técnicas y pautas que ayudarán a que el sistema vuelva a funcionar con normalidad.

Solución a ataques del tipo:

- **Ataques DDOS:** Estos ataques no se pueden corregir de lado del servidor o al menos corregir eficientemente. Los firewall y los sistemas operativos ofrecen mecanismos capaces de detectar amenazas de éste tipo. Sin embargo, estas soluciones se ha demostrado que no son eficientes para cubrir estas amenazas, si no que a veces, acaban siendo los cuellos de botella en caso de ataques. Hay que poner un servicio entre el atacante/atacantes y el servidor que mitigue el problema. Existe la herramienta Kona Site Defender, que pertenece a la corporación Akamai. Kona Site Defender absorbe el tráfico DDoS dirigido a la capa de aplicación y a la capa de red, desviándolo y autenticando el tráfico válido en el borde desde la red. Esta protección sólo permite el tráfico de los puertos 80 o 443 (los de http y https respectivamente). Lo que hace esta corporación es duplicar el contenido de tu servidor web en sus propios servidores. Cuando un cliente quiere acceder al contenido del servidor web en cuestión, todo o gran parte del contenido multimedia es descargado desde los servidores de Akamai, liberando al servidor atacado de gran parte de la carga. Otras empresas que dan soporte a este tipo de ataques son Interoute [2] o Corero [3].

- **Ataques de fuerza bruta:** Para mitigar este tipo de ataques existe la aplicación Fail2Ban [4], que penaliza o bloquea las conexiones que intentan accesos por fuerza bruta. Un ejemplo de uso de la aplicación es el

siguiente: Si estamos recibiendo continuas peticiones de servicio de la ip X.X.X.X podemos configurar Fail2Ban para que a partir de un número de peticiones N bloques dicha IP.

- **Inyección SQL:** Una vez que hemos sido víctimas de una Inyección SQL y tenemos consciencia de ello, hemos dado un paso de gigante, ya que el proceso es costoso porque no deja huella en el sistema. La solución al problema no es trivial. Si sabemos qué datos han sido saqueados, (normalmente serán de usuarios) pediremos a esos usuarios que cambien sus datos. A partir de ese momento, lo que procede es adoptar las técnicas de prevención para este tipo de problemas.

- **XSS:** Como se ha comentado en secciones anteriores, éste tipo de ataques es parecido al de inyección SQL. No existe solución mecánica. Únicamente se deben tomar las medidas de prevención pertinentes.

- **Phishing:** Existen empresas dedicadas a la detección y mitigación de ataques phishing, una de ellas es BrandProtect™ [6]. Una vez más, la solución a este tipo de ataques pasa por su temprana detección. Ya que hemos facilitado nuestros datos de un sitio en cuestión al ente malicioso, lo primero y si se tiene suerte de ingresar en el sitio antes que el atacante, es modificar nuestras credenciales de usuario para así evitar que el atacante ingrese en el sitio.

Referencias:

- Libro “Hackers, Aprende a atacar y a defenderte” Capítulo 5 *Hacking de Servidores Web.*
- Libro “Análisis Forense Digital en Entornos Windows” Capítulo 2: Respuesta a incidentes.
- Libro “Análisis Forense Digital en Entornos Windows” Capítulo 4: Análisis de evidencias.
- [1,https://mmonit.com/monit/, 12-mayo-2015]
- [2,http://www.interoute.es/sites/default/files/files/datasheet_0ddos_mitigation.pdf, 12-mayo-2015]
- [3,http://www.corero.com/es/, 12-mayo-2015]
- [4,http://fail2ban.org/, 12-mayo-2015]
- [5,http://www.seguridadpc.net/phishing.htm, https://www.osi.es/es/actualidad/blog/2014/04/11/aprendiendo-identificar-los-10-phishing-mas-utilizados-por-ciberdelincuen 12-mayo-2015]
- [6, http://www.brandprotect.com, 12-mayo-2015]
- [7,https://www.sophos.com/es-es/medialibrary/Gated%20Assets/white%20papers/sophosclosingbackdoornetworkapplicationvulnerabilitieswpna.pdf?la=es-ES.pdf , 12-mayo-2015]

Otras referencias:

- [http://www.linux-party.com/8966-25-trucos-de-seguridad-para-servidores-linux-1-de-2#, 12-mayo-2015]
- [http://www.escueladeinternet.com/seguridad-en-servidores-dedicados, 12-mayo-2015]
- [http://www.tecmint.com/linux-server-hardening-security-tips/, 12-mayo-2015]
- [http://www.yolinux.com/TUTORIALS/LinuxTutorialInternetSecurity.html, 12-mayo-2015]
- [http://blog.elevenpaths.com/2013/06/faas-vision-global-de-pentesting-by.html 12-mayo-2015]
- [http://www.elladodelmal.com/2013/05/pentesting-by-desing.html, 12-mayo-2015]
- [http://calebbucker.blogspot.com.es/2012/10/penetration-testing-hacking-etico.html, 12-mayo-2015]