Auditoría y ataques Web

[10.1] ¿Cómo estudiar este tema?

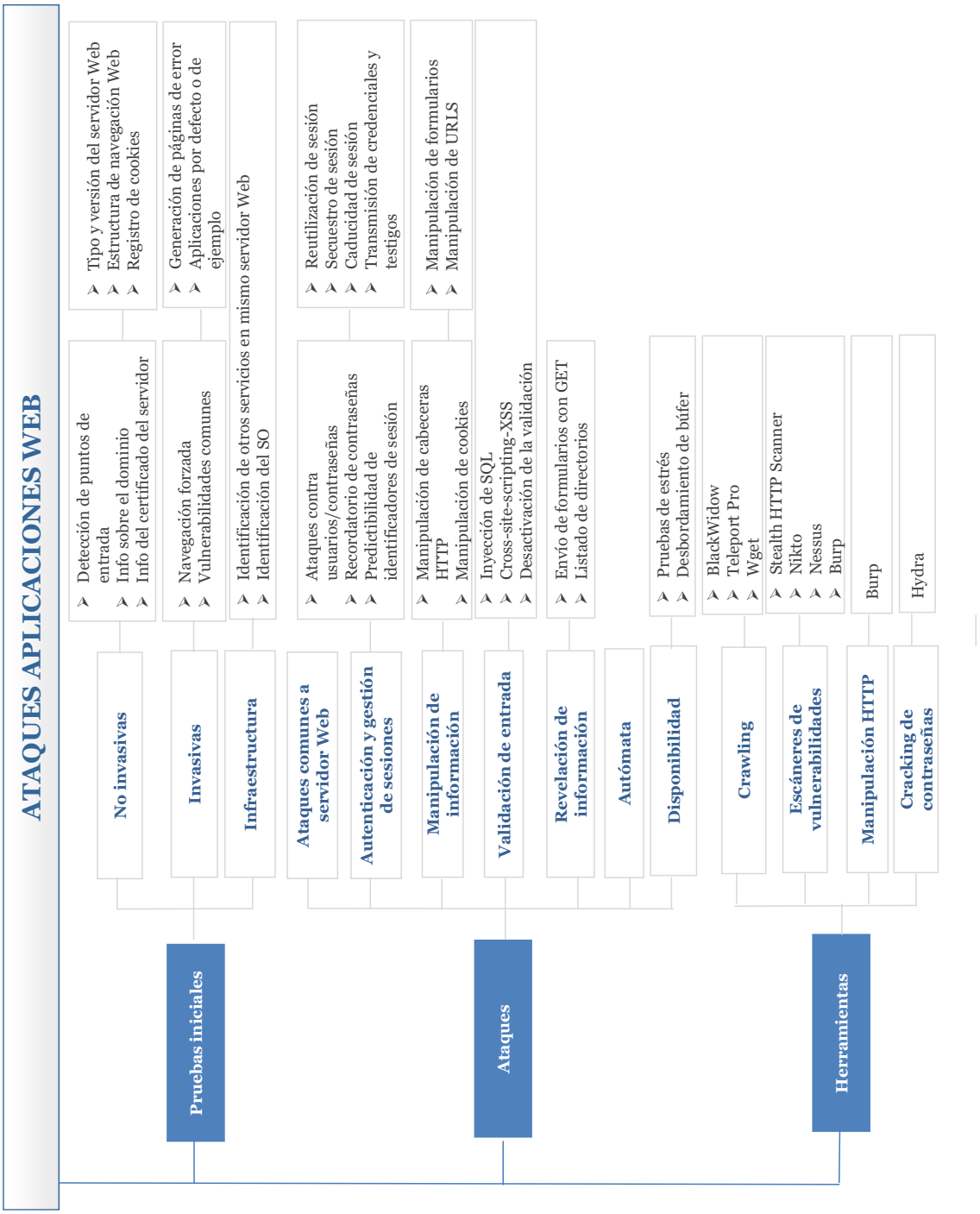
[10.2] Recopilación de información

[10.3] Técnicas de ataque

[10.4] Herramientas

TEMA 10

Esquema


ATAQUES APLICACIONES WEB
Pruebas iniciales
Ataques
Herramientas
No invasivas
Invasivas
Infraestructura
Ataques comunes a servidor Web
Autenticación y gestión de sesiones
Manipulación de información
Validación de entrada
Revelación de información
Autómata
Disponibilidad
Crawling
Escáneres de vulnerabilidades
Manipulación HTTP
Cracking de contraseñas
Detección de puntos de entrada
Info sobre el dominio
Info del certificado del servidor
Tipo y versión del servidor Web
Estructura de navegación Web
Registro de cookies
Navegación forzada
Vulnerabilidades comunes
Generación de páginas de error
Aplicaciones por defecto o de ejemplo
Identificación de otros servicios en mismo servidor Web
Identificación del SO
Ataques contra usuarios/contraseñas
Recordatorio de contraseñas
Predictibilidad de identificadores de sesión
Reutilización de sesión
Secuestro de sesión
Caducidad de sesión
Transmisión de credenciales y testigos
Manipulación de cabeceras HTTP
Manipulación de cookies
Manipulación de formularios
Manipulación de URLS
Inyección de SQL
Cross-site-scripting-XSS
Desactivación de la validación
Envío de formularios con GET
Listado de directorios
Pruebas de estrés
Desbordamiento de búfer
BlackWidow
Teleport Pro
Wget
Stealth HTTP Scanner
Nikto
Nessus
Burp
Burp
Hydra

Ideas clave

10.1. ¿Cómo estudiar este tema?

El estudio de este tema se realiza a través de los contenidos desarrollados en las **Ideas clave** expuestas a continuación.

En este tema haremos un recorrido exhaustivo sobre las **técnicas de ataque de las aplicaciones Web**. Las analizaremos, siempre que sea posible, en orden temporal, comenzando con las de **reconocimiento y extracción de información**, que suelen utilizarse en primer lugar.

Importante:
Siempre se ha sabido en el mundo de la seguridad informática que para poder protegerse adecuadamente de un atacante hay que **conocer bien las técnicas que éste utiliza**. Pero, «un gran poder conlleva una gran responsabilidad». Recuerda que **nunca debes probar estas técnicas en sistemas en producción o en aquellos donde no dispones de autorización expresa y por escrito**.

10.2. Recopilación de información

Como en el resto de los temas anteriores, hagamos una perspectiva real de la importancia del tema. Cuando el alumno haya visto este tema, se podrá preguntar, como «jugar» e iniciarse en el mundo de los ataques sin cometer delito. Para ello, se propone al alumno, que conozca «entrenadores», por ejemplo WebGoat, una aplicación web J2EE deliberadamente insegura, mantenida por OWASP y diseñada para enseñar lecciones de seguridad en aplicaciones Web. Desde Webgoat, el usuario debe demostrar su entendimiento de los problemas de seguridad al explotar la vulnerabilidad real en un entorno simulado. La aplicación es un ambiente realista de enseñanza, que provee a los usuarios con pistas y código para explicar mejor la lección. Sabido esto, comencemos por el principio…

En esta primera fase se **recopila información** acerca del blanco bajo análisis. Durante esta fase **no se realiza ningún ataque contra el servidor**, sino que solamente se extrae información sobre el mismo y sobre la aplicación web, siendo el objetivo la identificación de vulnerabilidades y posibles puertas de entrada.

Con el fin de reunir el mayor número posible de datos, se realizan los siguientes **tipos de pruebas**:

1. **Pruebas no invasivas:** se recopila información sobre el servidor web y la aplicación web albergada utilizando peticiones convencionales sobre el protocolo HTTP.
2. **Pruebas invasivas:** se recopila información sobre el servidor web y la aplicación web albergada utilizando peticiones que el servidor web no espera recibir. Muchas de estas peticiones generan errores en el servidor, que pueden revelar la información buscada.
3. **Pruebas de infraestructura:** se recopila información sobre la plataforma y servicios ofrecidos, además del web.

A continuación se describen cada una de las pruebas de esta fase preparatoria para el ataque, explicándose con más detalle **en qué consisten las técnicas agrupadas bajo cada categoría**.

1. Pruebas no invasivas

Las pruebas no invasivas **sirven para recopilar información sobre el blanco utilizando peticiones convencionales sobre el protocolo HTTP**. Se consideran no invasivas en tanto que no realizan ninguna petición anómala que el servidor no esté preparado para aceptar. Estas peticiones no generarán errores en el servidor y se caracterizan porque el código HTTP devuelto por el servidor es el 200.

Las pruebas no invasivas **recopilan la siguiente información**:

- Detección de puntos de entrada
- Información sobre el dominio
- Información del certificado de servidor
- Tipo y versión del servidor web
- Estructura de navegación de la aplicación web

- Registro de cookies

Detección de puntos de entrada

Se consulta el servidor DNS para detectar servidores implicados en ofrecer el servicio web. Mediante la transferencia de zona se puede descargar la tabla completa de nombres y direcciones IP almacenadas en el servidor DNS. Esta información permite elaborar un esquema de la red atacada. **Las direcciones IP indican qué equipos probablemente se encuentran activos**, mientras que los nombres a menudo indican la localización de servidores importantes, a la vez que pueden revelar nombres de usuario.

Información sobre el dominio

Se consulta la base de datos de whois con el fin de obtener nombres, teléfonos y direcciones personales. Esta información puede utilizarse posteriormente en ataques de ingeniería social.

Información del certificado de servidor

En el caso de que el servidor utilice el protocolo SSL para cifrar el canal de comunicaciones con los clientes, **se consulta su certificado con el fin de buscar filtraciones de información sobre la compañía y debilidades en el cifrado utilizado**. Esta información puede utilizarse posteriormente en ataques de ingeniería social, en ataques de fuerza bruta contra el cifrado si éste es débil (por ejemplo, claves simétricas de 40 bits) o en la falsificación de sitios web para engañar a usuarios, técnica conocida como ***web spoofing***.

Tipo y versión del servidor web

De forma rutinaria, **en cada respuesta a una petición HTTP, el servidor devuelve una cabecera informando de su tipo y versión**. Se puede utilizar cualquier herramienta que permita interceptar las respuestas del servidor para capturar esta cabecera, llamada Server. El tipo y versión de servidor determina el tipo de ataques comunes a utilizar contra él.

Estructura de navegación de la aplicación web

Utilizando técnicas de spidering o crawling se descarga el contenido completo de la aplicación web. Una vez almacenadas en local todas las páginas web y archivos, se puede proceder a su análisis, utilizando herramientas de búsqueda de cadenas y patrones en archivos.

Registro de cookies

Se registran todas las cookies enviadas por el servidor para su posterior análisis. Se presta especial atención a las cookies generadas durante la autenticación de usuarios en áreas de acceso restringido del servidor, conocidas como testigos o identificadores de sesión.

2. Pruebas invasivas

Las pruebas invasivas **sirven para recopilar información sobre el servidor web y la aplicación web albergada utilizando peticiones que el servidor web no espera recibir**.

Muchas de estas peticiones generan errores en el servidor, que pueden revelar la información buscada. Se caracterizan porque el código HTTP devuelto por el servidor puede ser distinto de 200. Aunque el código de retorno sea 200, se piden páginas que un usuario convencional nunca pediría, por lo que siempre son indicio de un ataque en preparación o en curso.

Las **pruebas invasivas** se agrupan en las siguientes categorías:

- Navegación forzada
- Vulnerabilidades comunes
- Generación de páginas de error
- Aplicaciones de ejemplo

Navegación forzada

Los servidores web devuelven diferentes códigos de error en función de la existencia o no de una página, y de si se poseen permisos o no para ejecutarla/visualizarla.

Enviando peticiones sucesivas de distintos nombres de páginas para distintas rutas de acceso, se puede llegar a determinar con precisión el mapa de archivos del servidor.

Vulnerabilidades comunes

A lo largo del tiempo se van descubriendo agujeros de seguridad en los servidores web. Existen bases de datos con todas las vulnerabilidades descubiertas y herramientas que se sirven de la información contenida en esas bases de datos para probarlas todas contra un servidor elegido como blanco.

Si un servidor no se mantiene al día en la aplicación de los parches del fabricante y no se configura de forma segura, entonces es seguro que poseerá alguna de las vulnerabilidades comunes ya publicadas. **Estas vulnerabilidades pueden explotarse para atacar al servidor**, con un impacto variable.

En esta fase simplemente se enumeran las vulnerabilidades descubiertas utilizando herramientas automatizadas de exploración. En la siguiente fase se intentan explotar, lo cual no siempre resulta posible.

Generación de páginas de error

Cuando debido a una entrada inesperada se produce un error durante la ejecución de una página activa, puede revelarse una cantidad variable de información, que normalmente incluye como mínimo la ruta de acceso física del archivo y a menudo revela la localización de archivos de inclusión o sentencias completas de SQL.

Aplicaciones de ejemplo

Muchos servidores web y herramientas de comercio electrónico se distribuyen con aplicaciones de ejemplo en las que se terminan encontrando agujeros que permiten desde revelar el código fuente de otras páginas activas o la estructura de directorios hasta la ejecución de comandos arbitrarios del sistema operativo.

Un servidor en explotación que no haya eliminado estas aplicaciones, se encuentra expuesto a este tipo de vulnerabilidades.

3. Pruebas de infraestructura

Las pruebas de infraestructura **sirven para recopilar información sobre la infraestructura sobre la que se ofrece el servicio web**. Se busca obtener información acerca de los siguientes aspectos:

- Identificación del sistema operativo
- Identificación de servicios ofrecidos en el mismo servidor

Identificación del sistema operativo

Mediante el **uso de técnicas de identificación de sistemas operativos, normalmente invasivas, se averigua cuál es el del servidor web**. Esta información resultará de utilidad más adelante, con el fin de dirigir ataques contra el servidor, pero no a nivel web.

Identificación de servicios

Mediante **la exploración de puertos y la captura de banners, se averigua qué servicios, además del web, se están ejecutando en el servidor bajo análisis**. Posteriormente, se buscará la presencia de vulnerabilidades en el resto de servicios ofrecidos, que permitan comprometer el servidor.

10.3. Técnicas de ataque

Una vez que ya se ha recopilado toda la información posible sobre el servidor, y se conoce el sistema operativo de la plataforma, el tipo y versión de servidor web, los lenguajes utilizados, la estructura de directorios, el uso de cookies, etc., se pasa a la **fase propiamente de ataque**, en la que **se trata de explotar las vulnerabilidades identificadas previamente**.

Durante esta fase de ataque se presta atención a los s**iguientes puntos de entrada posibles**:

1. **Ataques comunes al servidor web**
2. **Autenticación y gestión de sesiones**

3. **Manipulación de información**
4. **Validación de entrada**
5. **Revelación de información**
6. **Autómatas**
7. **Disponibilidad**

Analicemos cada punto con mayor detalle.

1. Ataques comunes al servidor web

Los agujeros de seguridad pueden localizarse en la aplicación web o bien en la plataforma sobre la que ésta se explota: el servidor web, el servidor de aplicaciones, el servidor de bases de datos, el cortafuegos, el sistema operativo, etc.

En el primer caso, el agujero o la aparición de vulnerabilidades es responsabilidad del equipo de desarrollo y programación. Mientras que en el segundo caso, **los agujeros aparecen como consecuencia de errores en la plataforma desarrollada por el fabricante o en la mala configuración de la misma por parte del administrador**. En esta sección se describen los **agujeros correspondientes al segundo caso**:

- Revisión de vulnerabilidades conocidas
- Revisión de configuraciones erróneas

Revisión de vulnerabilidades conocidas

Las vulnerabilidades comunes incluyen todos los bugs y agujeros de seguridad publicados y documentados (y por lo tanto bien conocidos) que pueden encontrarse en el sistema operativo, en el servidor web, en el servidor de aplicaciones, en el servidor de bases de datos, así como en componentes de terceras partes (como carritos de la compra, módulos criptográficos, pasarelas de pago, etc.).

La mayoría de estas vulnerabilidades pueden solucionarse mediante la aplicación de parches publicados por el fabricante. Los atacantes buscan explotar estas vulnerabilidades en sistemas que no han sido parcheados convenientemente.

Existen herramientas tanto comerciales como de libre distribución que incorporan bases de datos actualizadas con todas las vulnerabilidades conocidas de este tipo y comprueban de forma automatizada la presencia de la vulnerabilidad y si es explotable o no.

Revisión de configuraciones erróneas

Otra fuente común de vulnerabilidades procede de una **mala configuración de la plataforma**, en cualquiera de sus niveles (servidor web, servidor de aplicaciones, servidor de bases de datos, sistema operativo, etc.): permisos de acceso inadecuados a archivos y procesos, rutas de acceso por defecto a programas y aplicaciones, usuarios predeterminados con contraseñas conocidas, servicios innecesarios activos, aplicaciones de ejemplo, etc.

Muchas de estas configuraciones defectuosas están perfectamente documentadas y son generalmente conocidas, por lo que las herramientas automatizadas de exploración de vulnerabilidades las incorporan, permitiendo su rápida identificación.

2. Autenticación y gestión de sesiones

La autenticación y autorización resultan fundamentales en toda aplicación web en la que se desee restringir lo que los usuarios pueden hacer. La autenticación permite conocer la identidad de quienes se conectan al servidor. La autorización permite verificar qué privilegios tiene asignados cada usuario y saber así qué acciones le están permitidas.

Una vez detectada la necesidad de autenticar y autorizara los usuarios, surge la importante cuestión de decidir cómo se llevan a cabo. Normalmente, en aplicaciones para Internet con un número elevado de usuarios, se recurre a una solución basada en formularios. El inconveniente que plantea este tipo de solución reside en la necesidad de que el propio diseñador de la aplicación web gestiones todos los pasos de la autenticación y del mantenimiento de la información de sesión: al usuario se le piden las credenciales una sola vez y en caso de ser válidas se le entrega un testigo, cuya presentación servirá para autenticarle en las sucesivas páginas y recursos protegidos.

Se llevan a cabo las siguientes **pruebas contra la autenticación y gestión de sesiones**:

- Ataques contra nombres de usuario y contraseñas.
- Recordatorio de contraseñas.
- Predictibilidad de identificadores de sesión.
- Reutilización de sesión.
- Secuestro de sesión.
- Caducidad de sesión
- Transmisión de credenciales y testigos.

Ataques contra nombres de usuario y contraseñas

Si no se crea una directiva de creación de contraseñas que obligue a la utilización de contraseñas fuertes, los usuarios tienden a inventar contraseñas que son fáciles de recordar y por tanto, fáciles de adivinar por un atacante.

Se realizan los siguientes **ataques:**

- **Nombres de usuario/Contraseñas comunes:** muchas aplicaciones web tienen una cuenta de usuario activada por defecto. Esta cuenta, que suele el administrador del sistema, viene preconfigurada con el sistema, a menudo con una contraseña estándar. El sistema podría verse comprometido si el atacante accede a una de estas cuentas utilizando valores predeterminados.

  Por otro lado, muchos usuarios utilizan con**traseñas relacionadas directamente con objetos, acciones, fechas o personas próximas a ellos**.

- **Ataques de diccionario: e**n los ataques de diccionario se utiliza como contraseña una extensa lista de palabras, que puede incluir algunas pequeñas variaciones entre las letras o sustituciones comunes (‘e’ por ‘3’, ‘l’ por ‘1’ o ‘!’, ‘o’ por ‘0’, ‘z’ por ‘2’, etc.). Se van probando todas secuencialmente, hasta dar con la correcta.

- **Ataques de fuerza bruta:** en los ataques de fuerza bruta se van probando como contraseñas todas las posibles combinaciones de dígitos, letras y otros caracteres del espacio de claves. Cuanto mayor sea el espacio de claves, es decir, el número de caracteres de la contraseña y el número total de caracteres utilizado, mayor será la entropía de la contraseña y, por consiguiente, mayor el esfuerzo computacional para adivinarla.

Recordatorio de contraseñas

El recordatorio de contraseñas resulta necesario cuando un usuario legítimo de un sistema no puede autenticarse porque ha olvidado su nombre de usuario y/o contraseña. Todos los sistemas que requieren autenticación necesitan implantar alguna directiva o mecanismo de recuperación de contraseña.

Los **dos procedimientos más utilizados para recuperar la contraseña olvidada** son:

- **Envío de la contraseña por correo electrónico:** se envía la contraseña a la dirección de correo electrónico del usuario. Es posiblemente la forma más utilizada, debido a su sencillez, automatización y seguridad relativa, ya que se supone que el canal entre la aplicación web y el servidor de correo del usuario es seguro. Sin embargo, existen varias vías de ataque para hacerse con esta contraseña de forma ilegítima:
  - Espiar el tráfico entre el usuario y el servidor de correo utilizando un *sniffer*. Este ataque sólo es posible cuando se tiene acceso a la red de la víctima.
  - Tomar control del servidor de correo de la víctima.
  - Predicción del enlace donde se puede leer la contraseña: A veces no se envía la contraseña en el correo electrónico, sino un enlace a una página web donde se puede leer la contraseña o que donde se puede crear una nueva. Si este enlace es predecible, el atacante puede generarlo él mismo, sin necesidad de interceptar ningún correo.

- **Preguntas y respuestas:** cuando el usuario se registra en el servidor web, proporciona la respuesta a una serie de preguntas. Si en el futuro olvida la contraseña, se le presentan las mismas preguntas, a las que deberá responder correctamente para que la contraseña le sea entregada. Desgraciadamente, resulta difícil crear un conjunto adecuado de preguntas, ya que normalmente sus respuestas son fáciles de adivinar o de obtener mediante ingeniería inversa.

Predictibilidad de identificadores de sesión

Cuando el usuario se autentica presentando credenciales válidas, l**a aplicación web genera un testigo de sesión que deberá ser presentado por el usuario junto con el resto de peticiones de páginas y recursos**. Si **el testigo de sesión es predecible, el atacante podría generarlo de manera que pueda acceder al área protegida con la identidad de un usuario**, pero sin conocer su nombre de usuario ni contraseña.

Si este ataque puede automatizarse, entonces podría extraerse la información personal de todos los usuarios de la aplicación.

Reutilización de sesión

Reutilización de un testigo válido una vez que ha caducado una sesión de usuario. El testigo puede proporcionarse al servidor como un parámetro adicional en el URL o bien como una cookie. Existen **dos formas básicas de realización de este ataque**:

- **Intercepción de las peticiones entre el navegador de la víctima y el servidor: permite** robar el testigo transmitido en el URL o en la cookie, siempre que no se utilice SSL para proteger el canal de comunicaciones.
- **Robo de la cookie:** se explota alguna de las vulnerabilidades de los navegadores o bien algún agujero de XSS en el servidor para hacerse con la cookie que almacena el testigo válido. Este ataque funciona incluso con SSL activado. Cuando se realiza el ataque, el usuario legítimo no necesita haber iniciado sesión para que el ataque tenga éxito.

Secuestro de sesión

Utilización de un testigo capturado, creado por fuerza bruta o ingeniería inversa, para tomar control de una sesión web de un usuario legítimo mientras su sesión permanece iniciada. El usuario legítimo podría perder la sesión con el servidor o ver reducida su funcionalidad. Este ataque normalmente depende de otros ataques, como la predictibilidad de los identificadores de sesión.

Caducidad de sesión

Tiempo transcurrido desde que el servidor genera un testigo de sesión hasta que deja de ser válido. Cuanto mayor sea este tiempo, mayor será la ventana de oportunidad para ataques de reutilización de sesión y de secuestro de sesión. En esta prueba siempre se verifica la presencia de un método para que el usuario pueda hacer expirar inmediatamente una sesión.

Transmisión de credenciales y testigos

Mecanismos de protección utilizados para el envío del nombre de usuario y la contraseña. La protección se realiza mediante el uso de SSL. Un error común consiste en proteger con SSL el envío de las credenciales de usuario, pero no del testigo de sesión una vez que el usuario ha sido autenticado. Si este testigo es interceptado, el atacante podría suplantar al usuario y cambiar su contraseña.

3. Manipulación de información

En muchas aplicaciones Web se asume que la información procedente de otras páginas Web, o de las cookies o de las cabeceras HTTP no ha sido modificada. Sin embargo, **existen muchos ataques que permiten manipularlas**:

- Manipulación de cabeceras HTTP
- Manipulación de cookies
- Manipulación de formularios
- Manipulación de URLs

Manipulación de cabeceras HTTP

Con las herramientas adecuadas se puede modificar fácilmente cualquier valor de las cabeceras HTTP: Server, Referer, Cookies, etc.

Manipulación de cookies

Al igual que el resto de las cabeceras, las cookies se pueden manipular con facilidad incluso en el disco del usuario, por lo que no se puede confiar sin más en los valores almacenados en ellas.

Manipulación de formularios

Resulta igualmente sencillo manipular cualquier campo contenido dentro de un formulario, como los campos ocultos, botones de radio y verificación o las opciones de listas de selección.

Manipulación de URLs

A menudo se utilizan los propios URLs para transmitir información de unas páginas a otras. Evidentemente, sin una validación adecuada de sus valores no se puede confiar en esta información que puede modificarse de manera trivial

4. Validación de entrada

En cualquier aplicación que reciba datos del usuario para manipularlos y generar una respuesta a partir de ellos, resulta fundamental validar toda entrada introducida por el usuario. Asumir que la entrada de usuario se ciñe al formato esperado puede acarrear consecuencias desastrosas. **La pobre validación de entrada da pie a las siguientes vías de ataque:**

- Desactivación de la validación
- Cross Site Scripting (XSS)
- Inyección de comandos SQL
- Inyección de comandos del sistema operativo
- Comandos para depuración
- Server Side Includes
- Comprobación de límites
- Rutas de acceso de entrada

Desactivación de la validación

Un **error común consiste en validar los datos de entrada únicamente en el extremo cliente y no en el servidor**, asumiendo que todo lo que llega al servidor ha superado este filtro. Resulta trivial desactivar la validación en el extremo cliente, eliminando por tanto toda validación de los datos.

Cross Site Scripting (XSS)

Se trata del **proceso de inserción de código dentro de páginas enviadas por otra fuente**. El XSS permite a un atacante introducir código ejecutable arbitrario dentro de la sesión web de otro usuario. Una vez que el código se ejecuta, lo hace dentro del contexto de seguridad de la página web visitada, pudiendo realizar una gran variedad de acciones, desde monitorizar la sesión web del usuario hasta robarle sus cookies. Sólo afecta a otros usuarios de la aplicación, no al servidor.

Inyección de comandos SQL

Una pobre validación de la entrada a una página activa puede conducir a la ejecución de sentencias arbitrarias de SQL en la base de datos del back-end. Dependiendo de la configuración de la base de datos, se puede llegar incluso a la ejecución de procedimientos almacenados extendidos que permiten ejecutar comandos arbitrarios del SO con privilegios de administrador.

Dependiendo de los privilegios del usuario de base de datos con los que se ejecutan las sentencias SQL, puede verse el contenido de tablas, modificarlo, borrarlo, crear nuevos objetos y usuarios, e incluso ejecutar comandos arbitrarios del sistema operativo.

Inyección de comandos del sistema operativo

Una pobre validación de la entrada a una página activa puede conducir a la ejecución de comandos arbitrarios del SO, especialmente en plataformas UNIX. Los comandos típicamente se ejecutan con los mismos privilegios que el componente de aplicación o el servidor web.

Se puede realizar una escalada de privilegios que conduzca a la usurpación de la cuenta administrador/root.

Comandos para depuración

Resulta una práctica común i**ntroducir estructuras de depuración durante el desarrollo de software con el fin de localizar errores de programación**. La depuración puede adoptar múltiples formas: puntos de interrupción, volcados de memoria, lectura de registros, presentación de datos intermedios, etc.

Su propósito consiste en permitir al programador ver cómo está funcionando el programa y deducir dónde se encuentran los fallos. El atacante puede probar algunas construcciones de depuración obvias para ver si se han eliminado todas las construcciones de depuración antes del despliegue en explotación.

Cualquier olvido puede resultar en una grave vulnerabilidad debido a la potencia de la mayoría de construcciones de depuración.

Server Side Includes

Los Server-Side Includes (SSI) son directivas que se pueden incluir dentro de una página HTML, de manera que se presente como parte de la página el resultado de la ejecución de la directiva. Normalmente, se trata de la inclusión del contenido de un archivo o de la salida de la ejecución de un comando.

Aunque los SSI pueden ser muy útiles y dotan de gran flexibilidad a una página web, también pueden resultar computacionalmente costosos, pueden impedir la portabilidad de las páginas web y, más importante, pueden llegar a abrir agujeros de seguridad, ya que el autor de la página HTML decide qué programas se ejecutarán y con qué argumentos.

En el caso peor, se podría llegar a ejecutar cualquier comando bajo la identidad del servicio web y así producir daños irreparables o revelar información confidencial.

Comprobación de límites

Si la página no valida correctamente la longitud de los parámetros de entrada, se pueden realizar todo tipo de ataques: inyección de scripts, inyección de SQL, inyección de comandos del sistema operativo, desbordamientos de búfer, etc.

Rutas de acceso de entrada

Debido a vulnerabilidades o a fallos en la configuración del servidor Web, o bien a una pobre validación de entrada, puede engañarse a la aplicación Web para que sirva páginas o ejecute scripts fuera del área de trabajo predeterminada para el usuario.

Normalmente, estos ataques se materializan utilizando rutas de acceso padre de la forma '../' o variaciones de las mismas codificadas en URL, Unicode, Hexadecimal, etc.

5. Revelación de información

Las aplicaciones web pueden revelar inadvertidamente información confidencial, como nombres y direcciones de correo de desarrolladores, estructuras de directorios, rutas de acceso de archivos, nombres de usuario y contraseñas para acceso a otros servidores, etc. Normalmente, la información se filtra a través de las siguientes **prácticas de programación deficientes**:

- Caché del navegador
- Envío de formularios con GET
- Comentarios reveladores dentro del código HTML
- Rutas de acceso en <title>
- Listado de directorios
- Archivos de respaldo
- Lenguajes utilizados

Caché del navegador

Los navegadores almacenan en su caché una copia de todas las páginas que visitan. Algunas de estas copias pueden contener información confidencial que ha sido enviada previamente, por lo que quedaría a merced de un atacante que robase su caché. El peligro de ataque es mayor en quioscos Internet y en ordenadores compartidos en cibercafés, bibliotecas, etc.

Envío de formularios con GET

El uso del método **GET** para el envío de formularios presenta el inconveniente de almacenar todos los datos enviados en el historial del navegador del usuario, en el registro de actividad del servidor web y en las cabeceras *Referer* de páginas a las que se visite siguiendo enlaces desde el servidor en cuestión.

Comentarios reveladores dentro del código HTML

Inclusión de comentarios dentro de las páginas HTML con indicaciones sobre la estructura del sistema de archivos, personal de desarrollo, claves de acceso, etc.

Rutas de acceso en <title>

Cuando se crean páginas HTML con ciertas herramientas de desarrollo, aquellas a las que no se les especifica un título pueden adoptar la ruta de acceso del archivo.

Listado de directorios

Si no se desactiva el listado de directorios, se puede revelar información como nombres de archivos secretos o que no conviene exponer. Este es un fallo muy común, conocido en inglés como ***directory traversal***.

Archivos de respaldo

Cuando se editan los archivos web directamente en el servidor de explotación, dependiendo de la herramienta de edición de páginas web utilizada es posible que queden en el servidor archivos con nombres idénticos, pero con extensión distinta, como *.bak* o *.base*. Si se solicitan estos archivos con un navegador, se obtendrá el código fuente del mismo, en lugar de ser procesados por el servidor.

Lenguajes utilizados

Examinando las extensiones de archivos y los códigos de error se identifican los lenguajes de programación utilizados. Esta información resulta de utilidad para el atacante para probar ataques utilizando sintaxis y comandos específicos del lenguaje empleado en cada caso.

7. Disponibilidad

La disponibilidad de servicio c**onstituye un objetivo básico de todo servidor web que oferta sus servicios en Internet**. Se pretende que los usuarios legítimos no vean denegado su acceso a la información y recursos ofertados.

Los ***ataques de denegación de servicio (DoS)*** constituyen la forma más frecuente de ataque contra la disponibilidad. Los ataques DoS se pueden dirigir contra el equipo que alberga el servicio web o contra la propia aplicación web.

En el primer caso, el ataque utiliza técnicas igualmente válidas para denegar el servicio a cualquier equipo conectado a Internet. En el segundo caso, el ataque explota vulnerabilidades en el servidor web o en la aplicación web para interrumpir la prestación normal del servicio.

Con el fin de evaluar la disponibilidad del servicio web del blanco analizado, se realizan las siguientes **pruebas de ataque**:

- Pruebas de estrés
- Desbordamientos de búfer
- Inyección de errores

Pruebas de estrés

Las pruebas de estrés **simulan miles e incluso decenas de miles de transacciones web concurrentes contra un mismo sitio web**. Se utilizan clientes virtuales cada uno de los cuales genera múltiples peticiones HTTP/HTTPS. Cada cliente despliega múltiples tareas multihebra, que actúan como generadores de carga, representando a docenas o cientos de clientes. De esta forma se puede medir objetivamente, con parámetros tales como transacciones por segundo (TPS) el tiempo de respuesta y el rendimiento global.

Si una aplicación web no está dimensionada adecuadamente, este tipo de pruebas pueden detectar ataques potenciales de DoS mediante la inundación de peticiones.

Desbordamientos de búfer

Los desbordamientos de búfer **constituyen una de las fuentes de agujeros de seguridad más importantes de las últimas décadas**. La causa principal de los problemas de desbordamiento de búfer se encuentra en la falta de comprobación de la longitud de los argumentos de entrada cuando se pasan a rutinas escritas en ciertos lenguajes como C/C++.

Cuando un programa intenta escribir más allá de los límites de un búfer de memoria, se produce un desbordamiento. Leer o escribir más allá de los límites del búfer reservado puede causar una serie de comportamientos diversos: los programas pueden actuar de formas extrañas o fallar por completo.

En el caso mejor, un desbordamiento de búfer puede interrumpir el servicio e incluso detener el servidor. En el caso peor, permite la ejecución de código arbitrario con los mismos privilegios que el programa donde está presente.

Inyección de errores

Otra forma de causar la denegación de servicio de un servidor consiste en **explotar errores en el software de servidor web o en la aplicación web**, que ante entradas erróneas entren en estado inestable o bucles infinitos, incapaces de servir más peticiones.

10.4. Herramientas

Para poner en práctica esta metodología y analizar la seguridad de las aplicaciones web, **se utilizan las siguientes categorías de herramientas**:

1. **Herramientas de spidering o crawling**
2. **Herramientas de exploración de vulnerabilidades web**
3. **Herramientas de identificación y exploración de puertos**
4. **Herramientas de manipulación de HTTP**
5. **Herramientas de cracking de contraseñas**

1. Herramientas de spidering o crawling

Estas herramientas **sirven para realizar el escaneo de todo un sitio Web**. Este suele ser el primer paso en una auditoría, pues sirve para hacerse una idea global de los sitios donde podrían surgir los problemas. Este tipo de herramientas comienzan con una URL base y a partir de ella, van visitando cada uno de los enlaces de la página, formando un árbol.

Windows	Linux
BlackWidow	Wget
Teleport Pro	

2. Herramientas de exploración de vulnerabilidades web

En este caso, **el objetivo de estas herramientas es automatizar en lo posible el descubrimiento de posibles vulnerabilidades**. Para ello cuentan con una gran base de datos, que comparan con el sitio Web en cuestión.

La mayoría de ellas son comerciales, y muy caras, pero cuentan con versiones gratuitas de prueba, habitualmente. La versión de Linux, libre, por supuesto, no es de mala calidad, aunque se queda algo corta para un entorno profesional.

Windows	Linux
Stealth HTTP Scanner de N-Stealth	Nikto
WebInspect de SPI Dynamics	Nessus
Burp	

3. Herramientas de manipulación de HTTP

Este tipo de herramientas constituyen la «navaja suiza» de un auditor de aplicaciones Web, pues **le permite parar y manipular las peticiones HTTP antes de que éstas lleguen al servidor**. De esta forma se pueden utilizar muchas de las técnicas que hemos analizado en las secciones anteriores.

Una de las mejores herramientas en este caso es, sin duda, ***Burp***, disponible tanto para plataformas *Windows* como *Linux*, pues está escrita en Java.

4. Herramientas de cracking de contraseñas

En este último tipo de herramientas veremos aquellas **destinadas a realizar ataques por fuerza bruta o diccionario a un formulario de autenticación**, por ejemplo. De esta forma, cuando todos los demás métodos fallan, aún se puede conseguir el acceso, pues habitualmente los usuarios utilizan claves débiles en muchos servicios.

También en este caso existe una única herramienta de referencia, conocida como ***Hydra***, nativa de *Linux*. Además no sólo sirve para la Web, sino para prácticamente cualquier otro protocolo de red incluyendo, entre otros muchos, FTP, ICQ, IMAP, POP3, RDP, SMB, SSH, Telnet, etc.

Lo + recomendado

No dejes de leer…

Informe del Web Application Security Consortium

Este informe contiene un listado exhaustivo de prácticamente todos los ataques posibles a una aplicación Web, junto con ejemplos de cada uno de ellos.

Accede al documento a través del aula virtual o desde la siguiente dirección web:
http://projects.webappsec.org/f/WASC_TC-1.0.spa.doc

A Survey of SQL Injection Attack Detection and Prevention

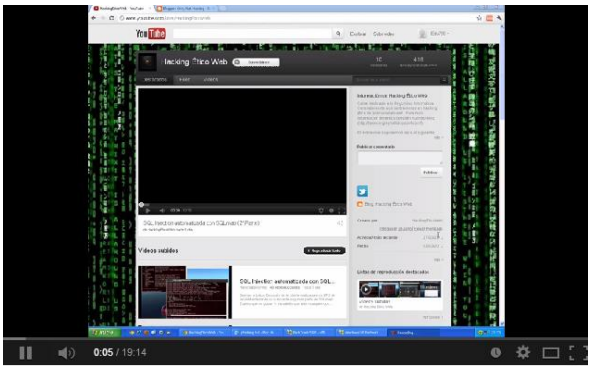
Ataque de inyección de lenguaje de consulta estructurado (SQLIA) como el más expuesto al ataque en Internet. A partir de este ataque, el atacante puede tomar el control de la base de datos, por lo tanto, ser capaz de interpolar los datos del servidor de base de datos para el sitio web.

Accede al documento a través del aula virtual o desde la siguiente dirección web:
https://pdfs.semanticscholar.org/1bdb/819d5ebaf67141f90f4fd03525c571b94e77.pdf

No dejes de ver...

Troyanizar un servidor Web

En este vídeo se muestra una técnica avanzada de ataque Web, haciendo uso de la plataforma *Metasploit*. En él se muestra cómo troyanizar un servidor, es decir, instalar un troyano en la máquina que corre el servidor Web, utilizando inyección de SQL.



Accede al vídeo a través del aula virtual o desde la siguiente dirección web:
https://www.youtube.com/watch?v=Ml1AmHFqruo

Ethical Hacking - Google Dorking

Tutorial sobre Google Dorks, combinaciones de operaciones de búsqueda que nos dará como resultado tanto la información que normalmente se muestra con cualquier búsqueda corriente como datos internos de la web que estén en el enramado de dicho sitio, pero no visible desde la navegación corriente.

Accede al vídeo a través del aula virtual o desde la siguiente dirección web:
https://www.youtube.com/watch?v=ELkuv1EzvVY

+ Información

A fondo

Hacking y seguridad de páginas Web

Ramos Varón, A.A. & Barbero Muñoz, C.A. (2014). *Hacking y seguridad de páginas Web*. RA-MA.

En este libro se describen las técnicas de ataque descritas en este tema, de una manera guiada, así como las contramedidas más habituales, de una forma eminentemente práctica. En el primer bloque se describen los entornos de prueba, para pasar después a encargarse de las técnicas Avanzadas, mostrando paso a paso una gran variedad de ataques funcionales diferentes que pueden vulnerar la seguridad de una aplicación Web. Por último, se trata un área interesante, denominada WAF (Web Applications Firewall), dando a conocer cómo funcionan estos dispositivos de protección.

Tutorial sobre inyección de SQL

Este interesante documento del grupo *HighSec* contiene ejemplos detallados de cómo atacar un servidor Web utilizando la técnica de inyección de código SQL. Comienza con una pequeña introducción a este lenguaje, que siempre viene bien, para después mostrar ejemplos reales sobre un entorno de pruebas

Accede al documento a través del aula virtual o desde la siguiente dirección web:
http://highsec.es/wp-content/uploads/2013/05/Seguridad-Web.pdf

Test

1. En la fase de recopilación de información de una auditoria Web, las pruebas invasivas:

A. Pueden provocar errores en el servidor, revelando de esta forma información relevante

B. No pueden provocar errores en ningún caso, pues se limitan a recolectar información como la versión del servidor, etc.

C. Recopilan información sobre la plataforma

D. Recopilan información sobre el servidor Web y la aplicación que corre en él

2. Estás haciendo una petición a un servidor Web que no ha producido un error, es decir, se ejecuta correctamente, puedes asegurar que el código que éste devuelve es el:

A. 200

B. 404

C. 500

D. 201

3. Un ataque por diccionario es un tipo de ataque contra:

A. Los sistemas de autenticación de doble factor.

B. Los sistemas de autenticación por usuario/contraseña.

C. Los servidores Web.

D. Las aplicaciones que corren en los servidores Web.

4. La predictibilidad de los identificadores de sesión es un problema de las aplicaciones Web que hace referencia a:

A. La inseguridad del esquema de recordatorio de contraseñas

B. El envío de la contraseña por correo electrónico

C. Las preguntas y respuestas habituales para poder recuperar la contraseña

D. El proceso de generación del testigo de sesión de cada usuario no es robusto, y puede predecirse

5. Las cookies son porciones de información que los servidores Web piden almacenar en el navegador del usuario. ¿Cuál de las siguientes afirmaciones es cierta?

A. Las cookies no pueden ser manipuladas, puesto que su ubicación en disco está protegida

B. Las cookies pueden ser manipuladas, pero sólo por el usuario que las creó

C. Las cookies pueden ser manipuladas fácilmente, por lo que no se debe confiar en los valores almacenados en ellas

D. Las cookies no pueden ser manipuladas, como medida de seguridad

6. Cross-site scripting (XSS) e inyección de código SQL son dos tipos de problemas relacionados con la:

A. Validación de entrada

B. Manipulación de la información

C. Revelación de la información

D. Autenticación y gestión de usuarios

7. Para enviar información sensible, como la contraseña de un formulario de autenticación, ¿es preferible hacerlo a través de peticiones GET o POST?

A. Es indiferente, ambas ofrecen el mismo nivel de seguridad

B. Es preferible hacerlo a través de peticiones GET, pues éstas no se reflejan en el historial de navegación ni en los logs del servidor Web

C. Es preferible hacerlo a través de peticiones POST, pues éstas no se reflejan en el historial de navegación ni en los logs del servidor Web

D. Es indiferente, aunque normalmente se utilizarán las peticiones GET, porque son más fáciles de depurar

8. Las pruebas de estrés a una aplicación Web pretenden determinar:

A. La disponibilidad de la misma

B. Su nivel de seguridad ante ataques de robo de información

C. Su nivel de seguridad ante ataques de manipulación de información

D. Su nivel de seguridad ante ataques de robo de cookies

9. Existe una herramienta considerada la «navaja suiza» para la manipulación de cabeceras HTTP, y que debe estar en el maletín de herramientas de todo auditor Web. Además, funciona en Windows y Linux. ¿Podrías citar su nombre?

A. Nessus
B. Burp
C. Nikto
D. Nmap

10. A la hora de atacar un formulario de autenticación, siempre quedar una última técnica o recurso, cuando todas las demás fallan. ¿A qué técnica nos referimos?

A. Manipulación de cabeceras HTTP
B. Ataques de manipulación de parámetros
C. Ataques de robo de sesión
D. Cracking de contraseñas, a través de ataques de fuerza bruta o diccionario