ATAQUES A APLICACIONES WEB

TIPOS DE ATAQUES

Ataques de inyección de Scripts

Ataques de inyección de código

Ataques de inyección de ficheros

ATAQUES DE INYECCIÓN DE SCRIPTS

OBJETIVO: *lograr inyectar en el contexto de un dominio un código Javascript, VBScript o simplemente HTML.*

FINALIDAD: *engañar al usuario para realizar una acción no deseada por éste.*

AFECTADO DIRECTO: *Usuario*

RESULTADO: *mediante suplantación de personalidad se podrán ejecutar las acciones deseadas en el servidor afectado, al que podrá accederse desde una página web.*

Cross Site Scripting (XSS)

XSS puede permitir la obtención de privilegios sobre algún sistema lo suficientemente débil como para permitir insertar código Javascript.

Una página es vulnerable a XSS cuando lo que enviamos al servidor se ve posteriormente mostrado en la página de respuesta.
Ej. Realizar comentarios, Modificaciones en el perfil de usuario, Búsquedas.

Fallos XSS

Permanentes: cuando el código queda almacenado en algún lugar, habitualmente una base de datos.

No permanentes: no se almacenan en ningún lugar.

Cross Site Scripting (XSS)

ATAQUES

Toma de control del navegador

Realización de acciones maliciosas en la aplicación Web. El peligro aumenta cuando el usuario afectado es un administrador.

Phishing

Modificar el comportamiento y la apariencia de una página web.

Ataques de defacement

Modificación de la apariencia original de una página web para que muestre un mensaje, normalmente reivindicativo, en lugar de su apariencia normal.

Ataques de denegación de servicios distribuidos (DDoS)

Forzar a los navegadores a realizar un uso intensivo de recursos muy costosos en ancho de banda o en capacidad de procesamiento de un servidor de forma asíncrona.

Gusano XSS

Código Javascript que se propaga dentro de un sitio web o entre páginas de Internet.

Cross Site Scripting (XSS)

MÉTODOS PARA INTRODUCIR XSS

El código se copia entre dos etiquetas HTML

Introducir el código Javascript que deseemos ejecutar

El código se copia dentro de una etiqueta value de una etiqueta <input>

Cuando realizamos una búsqueda lo habitual es que el término introducido se copie dentro del campo del buscador.

El código se copia dentro de un comentario HTML

El código se copia dentro de un código Javascript

Es habitual cuando las páginas utilizan datos introducidos por el usuario para generar algún tipo de evento personalizado o almacenar los que se van a usar posteriormente en algún otro lugar de la aplicación web.

Cross Site Scripting (XSS)

FILTROS XSS

Cuando introducimos ' o " estos caracteres se cambian por \' o \“

El código introducido no puede ser superior a X caracteres

No podemos introducir la cadena script o no se nos permite introducir los caracteres de mayor que (>) y menor que (<)

Cross Site Request Forgery (CSRF)

El objetivo es lograr que el usuario realice acciones no deseadas en dominios remotos.

Se basa en la idea aprovechar la persistencia de sesiones entre las pestañas de un navegador.

Cada vez que nos loqueamos en un sitio web se nos asocia una ***cookie*** de sesión que nos autentica únicamente frente al servidor, y esta técnica hace uso de esto, ya que no podemos controlar cuándo se envían estas cookies.

La técnica de CSRF, le permite a un atacante "simular" clics verídicos sobre una aplicación usando las credenciales (cookie de sesión) de un usuario.

Precauciones:

- Cerrar la sesión inmediatamente después del uso de una aplicación.
- No permitir que el navegador almacene las credenciales de ninguna página, ni que ningún servidor mantenga nuestra sesión recordada más que durante el tiempo de uso.
- Utilizar navegadores distintos para las aplicaciones de ocio y las críticas. independencia de las cookies de sesión.

Clickjacking

Se basan en engañar a los usuarios para que hagan clic sobre elementos de un sitio web donde ellos nunca lo harían voluntariamente, esto se consigue superponiendo dos páginas.

Una página principal, donde queremos que los usuarios hagan clic en zonas específicas, y otra, que sirve de señuelo, superpuesta sobre la anterior y con contenidos que sirvan de aliciente para que el usuario realice los clics en las zonas deseadas.

ATAQUES DE INYECCIÓN DE CÓDIGO

Es una técnica de ataque a aplicaciones web cuyo objetivo principal es aprovechar conexiones a bases de datos desde aplicaciones web no securizadas para permitir a un atacante la ejecución de comandos directamente en la base de datos.

SQL Injection

Acciones que se pueden realizar contra el sistema

Descubrimiento de información. Modificar consultas para acceder a registros y/o objetos de la base de datos.

Elevación de privilegios. Permitir a un atacante acceder a los identificadores de usuarios más privilegiados y cambiarse las credenciales.

Denegación de servicio. Ejecución de acciones destructivas, como el borrado de datos o parada de servicios.

Suplantación de usuarios. Realizar acciones con la identidad robada o spoofeada a otro usuario.

SQL Injection

Entorno de explotación del ataque

Condicionantes necesarios para que en una aplicación web se pueda dar la vulnerabilidad de inyección de comandos SQL:

- **Fallo en la comprobación de parámetros de entrada.**
- **Utilización de parámetros en la construcción de llamadas a bases de datos**
  - El problema esta en la utilización de parámetros que no han sido comprobados correctamente.
- **Construcción no fiable de sentencias**
  - El problema se genera con la utilización de una estructura de construcción de sentencias SQL basada en la concatenación de cadenas de caracteres.
  - Se toma la sentencia como una cadena alfanumérica a la que se le va concatenando el valor de los parámetros.

SQL Injection

Blind SQL Injection

Es una forma de realizar los ataques a ciegas

Se consigue que los comandos se ejecuten sin la posibilidad de ver ninguno de los resultados, sin obtener ningún tipo de información de la base de datos.

A pesar de que no se pueda ver los datos extraídos directamente de la base de datos, al cambiar los datos que se están enviando como parámetros, se pueden realizar inferencias sobre ellos en función de los cambios que se obtienen.

El parámetro vulnerable

En primer lugar se debe encontrar una parte del código de la aplicación que no esté realizando una comprobación correcta de los parámetros de entrada a la aplicación que se están utilizando para componer las consultas a la base de datos.

SQL Injection

Blind SQL Injection

Automatización

El objetivo es introducir ISQL0 e ISQL+ y comprobar si los resultados obtenidos se pueden diferenciar de forma automática o no y cómo hacerlo.

Tipos de automatización

- **Búsqueda de palabras clave.**
- **Basados en firmas MD5.**
- **Motor de diferencia textual.**
- **Basados en árboles HTML.**
- **Representación lineal de sumas ASCII**

"Inyección SQL de cambio de comportamiento cero" es una cadena que se inyecta en una consulta SQL y **no realiza ningún cambio** en los resultados.

"Inyección SQL de cambio de comportamiento positivo" es una cadena que **sí provoca cambios**.

Existen distintas herramientas que permiten la automatización de dichos ataques. Entre ellas destacan Absinthe, SQLInjector, SQLBfTools, Bfsql y SQL PowerInjector.

SQL Injection

Blind SQL Injection

Protección contra Blind SQL Injection

1. En toda consulta que se vaya a lanzar contra la base de datos y cuyos parámetros vengan desde el usuario, estos **parámetros** deben ser **comprobados** y se deben realizar funciones de **tratamiento** para todos los casos posibles.

2. Utilizar códigos que ejecuten **consultas** ya **precompiladas** para evitar que interactúe con los parámetros de los usuarios.

3. Controlen todas las posibilidades que puedan generar **error** en cualquier procedimiento por parte del programador. Para cada acción de error se debe realizar un tratamiento seguro del mismo y **evitar dar información útil** a un posible atacante.

4. **Auditar** los **errores,** tanto los de aplicación como los de servidor, pues pueden representar un fallo en el funcionamiento normal del programa o un intento de ataque.

ATAQUES DE INYECCIÓN DE FICHEROS

Remote File Inclusion

- Ejecutar **código remoto** dentro de la aplicación vulnerable.
- Se basa en la idea de que al igual que es posible cargar un fichero local para su inclusión dentro de la página, podríamos cargar uno remoto que contuviese **código malicioso**.
- Esto es posible en **lenguajes interpretados**, donde podemos incluir un fichero con código y añadirlo a la ejecución.
- Una **mala configuración del servidor** de la aplicación permite la inclusión de ficheros externos.
- La **configuración correcta del servidor** debería ser la prohibición total de realizar estas acciones.
- Este fallo de seguridad puede dar lugar a que un atacante logre **ejecutar cualquier código** deseado en el servidor web con los riesgos que esto conlleva.

Local File Inclusion

- Afecta tanto a **lenguajes compilados** como **interpretados**.
- Se basa en la posibilidad de incluir dentro de la página un fichero local del usuario con el que se ejecuta el servidor de aplicaciones web que tenga **permisos de lectura**.
- Para lograr explotar esta vulnerabilidad de una manera satisfactoria deberíamos poder hacer **llamadas a ficheros fuera de la ruta original**. Esto lo haríamos intentando incluir ficheros de un directorio superior.
- Para localizar sin equivocaciones los ficheros podemos usar un **fallo de Path Disclosure** para que nos dé información sobre la ruta local donde se ubican los ficheros.
- Para evitar que esta técnica tenga mayor impacto es importante pensar en **montar el servidor con el mínimo privilegio posible**, limitando la posibilidad de acceso a ficheros del servidor dentro de su propia carpeta, con lo que lograremos evitar el acceso a ficheros del sistema.

Webtrojans

- Es una **shell remota** que podemos subir a un servidor aprovechando un fallo de seguridad.
- Cuando un usuario nos envía un fichero debemos **comprobar que sea un fichero legítimo**. Si estamos esperando la llegada de un fichero con una imagen es necesario verificar que lo que recibimos es realmente una imagen y no otro tipo de fichero.
- Si no comprobamos los ficheros que se nos envían podríamos estar copiando en nuestro servidor un **fichero malintencionado** conocido como **webtrojan**.
- Estos fichero **se pueden camuflar** incluyéndolos tras una cabecera de una imagen.
- Para evitar este problema, debemos **comprobar** siempre la **extensión** y también la **cabecera del fichero** para corroborar que es del tipo que nosotros hemos permitido.
- Una buena practica es **establecer** mediante programación **la extensión de los ficheros** que se nos suben, lo que evita que alguien pueda subir un fichero y establecerlo con extensiones como ASP o PHP.