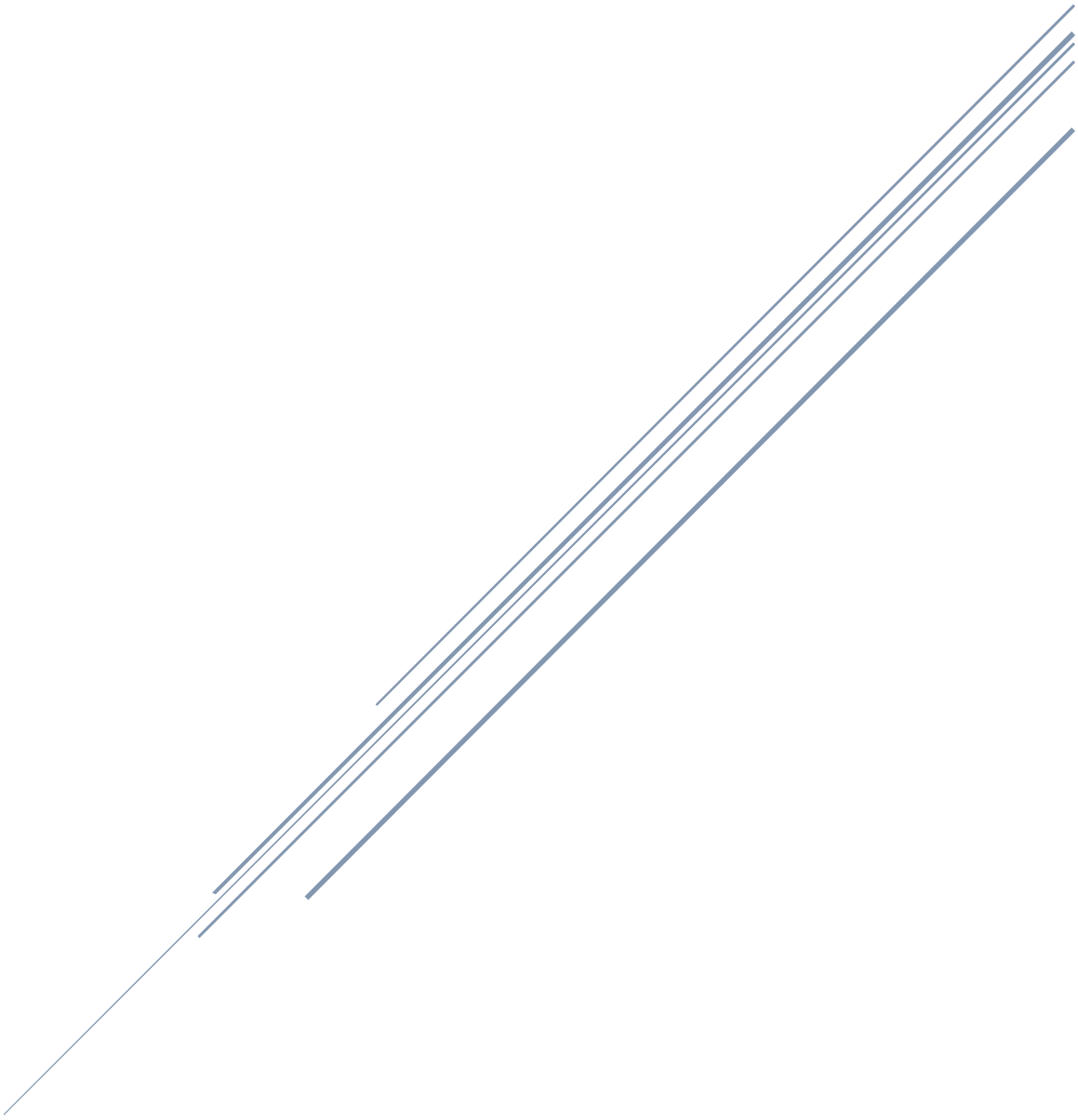
DISEÑO ORIENTADO A OBJETOS

Lic. Seguridad en Tecnologías de la información.



Carlos Abel Lugo González
1595054

Asignación de la Semana 3

Investigación a modo ensayo de los riesgos/aspectos de seguridad con html y/o javascript.

- Vulnerabilidad

Una vulnerabilidad es una debilidad del sistema informático que puede ser utilizada para causar un daño. Las debilidades pueden aparecer en cualquiera de los elementos de una computadora, tanto en el hardware, el sistema operativo, cómo en el software.

- Amenaza

Una amenaza a un sistema informático es una circunstancia que tiene el potencial de causar un daño o una pérdida. Es decir, las amenazas pueden materializarse dando lugar a un ataque en el equipo.

Como ejemplos de amenaza están los ataques por parte de personas, al igual que los desastres naturales que puedan afectar a su computadora. También se pueden considerar amenazas los fallos cometidos por los usuarios al utilizar el sistema, o los fallos internos tanto del hardware o cómo del software.

- Riesgo

El riesgo es la posibilidad de que una amenaza se produzca, dando lugar a un ataque al equipo. Esto no es otra cosa que la probabilidad de que ocurra el ataque por parte de la amenaza.

El riesgo se utiliza sobre todo el análisis de riesgos de un sistema informático. Esté riesgo permite tomar decisiones para proteger mejor al sistema. Se puede comparar con el riesgo límite que acepte para su equipo, de tal forma que si el riesgo calculado es inferior al de referencia, éste se convierte en un riesgo residual que podemos considerar cómo riesgo aceptable.

Javascript es una de las tecnologías que más se habla en la actualidad y se está utilizando en todas partes desde el navegador web de escritorio para teléfonos móviles. El lenguaje de programación se ha contenido dinámico a un nuevo nivel. Sin embargo, existen algunos riesgos asociados con el uso de javascript para cualquier trabajo de desarrollo graves. Velocidad

- Velocidad

Y las cuestiones relacionadas con la velocidad han estado plagando javascript desde sus primeros días. La situación ha mejorado significativamente, pero la velocidad sigue siendo un grave problema para ciertos dominios y plataformas. Esto es particularmente cierto para los juegos. Su nuevo y estremecedor juego puede funcionar de maravilla en su pc de escritorio de doble núcleo, pero trate de cargarlo en su iphone o dispositivo android. Es probable que las animaciones que has trabajado tan duro están muy por debajo de los 30 fotogramas por segundo que necesita para tener una buena experiencia para los usuarios.

- Diferencias motor

< p > no hay un motor de javascript . Google , apple y otras organizaciones tienen sus motores preferidos. Son similares, pero no idénticas, y no puede haber diferencias de rendimiento. Esto se nota especialmente en los dispositivos móviles que apple y google están encerrados en una lucha para producir el motor más rápido y menos intensivo de la batería.

Usuarios puede acceder a su código fuente de la mayoría de navegadores web comunes, simplemente haciendo clic en el botón " ver código fuente". Los visitantes del sitio pueden, sin su conocimiento, copie su código y hacerlo pasar como propio. Es poco lo que se puede hacer para combatir esto con excepción de ofuscar el código, o intencionalmente

escribir el código de una manera que es difícil de leer y entender. Por supuesto, eso no impide que cualquier persona de mayor robo de su código, pero puede disuadir a alguien que quiera modificar su código. Debe tenerse en cuenta que este problema no existe cuando se trabaja con javascript embebido en dispositivos móviles.

- Seguridad

Seguridad sigue siendo un problema con JAVASCRIPT, aunque la situación ha mejorado mucho desde los primeros días del idioma. Algunos de los problemas de seguridad más comunes relacionados con la caída lenguaje bajo la amplia categoría de "vulnerabilidades de cross-site ". Esto es cuando un atacante es capaz de conseguir una página web de confianza, como un sitio de banca en línea, para incluir un script malicioso con sus propios guiones benignos, el script malicioso normalmente registrará su log-in credencial y enviarla al atacante utilizarse en un momento posterior .

JavaScript y el DOM permite que existan programadores que hagan un uso inapropiado para introducir scripts que ejecuten código con contenido malicioso sin el consentimiento del usuario y que pueda así comprometer su seguridad.

Los desarrolladores de los navegadores tienen en cuenta este riesgo utilizando dos restricciones. En primer lugar, los scripts se ejecutan en un sandbox en el que sólo se pueden llevar a cabo acciones relacionadas con la web, no con tareas de programación de propósito general, como la creación de archivos. En segundo lugar, está limitada por la política del mismo origen: los scripts de un sitio web no tienen acceso a la información enviada a otro sitio web (de otro dominio) como pudiera ser nombres de usuario, contraseñas o cookies. La mayoría de los fallos de seguridad de JavaScript están relacionados con violaciones de cualquiera de estas dos restricciones.

Existen proyectos como AdSafe o Secure ECMA script (SES) que proporcionan mayores niveles de seguridad, en especial en el código creado por terceros (tales como los anuncios).

La Política de Contenido Seguro (CSP) es el método principal previsto para garantizar que sólo código de confianza pueda ser ejecutado en una página web.

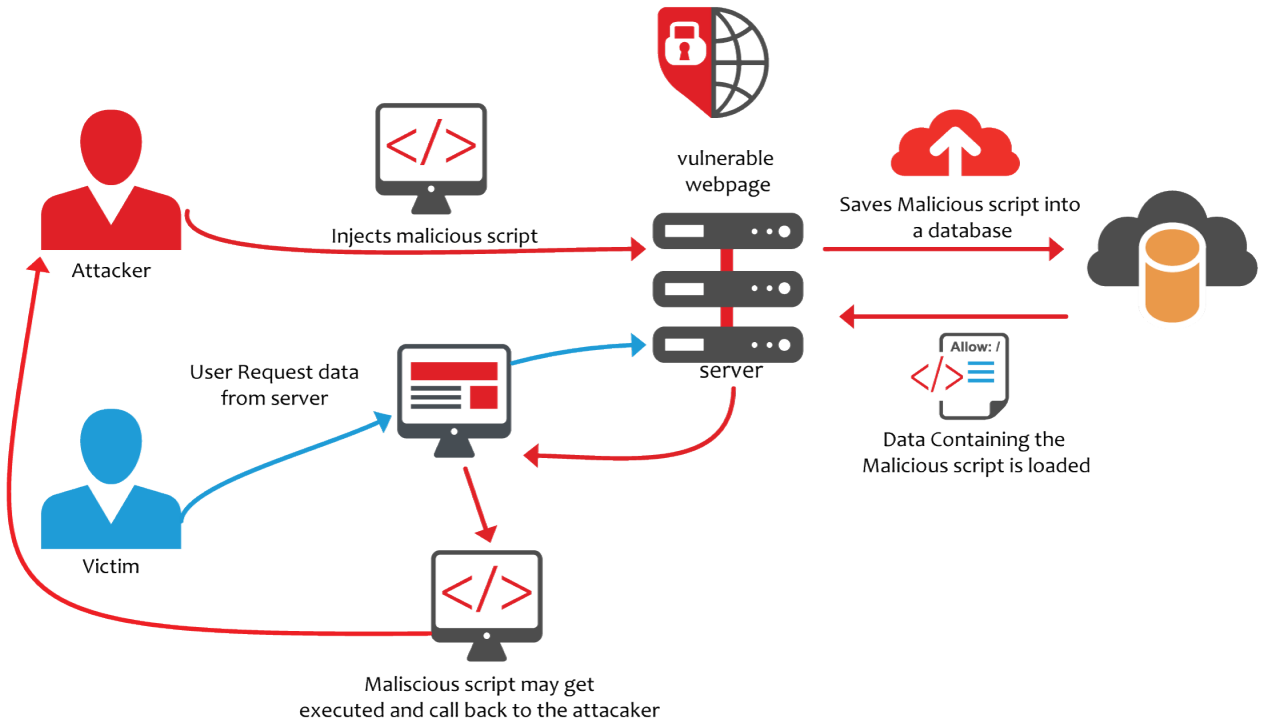
Vulnerabilidades cross-site

Un problema común de seguridad en JavaScript es el cross-site scripting o XSS, una violación de la política de mismo origen. Las vulnerabilidades XSS permiten a un atacante inyectar código JavaScript en páginas web visitadas por el usuario. Una de esas webs podría ser la de un banco, pudiendo el atacante acceder a la aplicación de banca con los privilegios de la víctima, lo que podría revelar información secreta o transferir dinero sin la autorización de la víctima. Una solución para las vulnerabilidades XSS es utilizar HTML escaping cuando se muestre información de fuentes no confiables

Algunos navegadores incluyen una protección parcial contra los ataques XSS reflejados (el atacante está en la misma petición web). El atacante

proporciona una URL incluyendo código malicioso. Sin embargo, incluso los usuarios de los navegadores son vulnerables a otros ataques XSS, tales como aquellos en los que el código malicioso se almacena en una base de datos. Sólo el correcto diseño de las aplicaciones Web en la parte servidora puede prevenir totalmente XSS. Las vulnerabilidades XSS también pueden ocurrir debido a errores de ejecución por los desarrolladores del navegador.

Otra vulnerabilidad es la falsificación de petición de sitio cruzado o CSRF. En CSRF, el código del sitio web atacante engaña al navegador de la víctima, permitiendo al atacante realizar peticiones en nombre de la víctima, haciendo imposible saber a la aplicación de destino (por ejemplo, la de un banco haciendo una transferencia de dinero) saber si la petición ha sido realizada voluntariamente por el usuario o por un ataque CSRF.



El ataque funciona porque, si el sitio de destino hace uso únicamente de las cookies para autenticar las solicitudes de la víctima, las peticiones iniciadas por el código del atacante tendrán las mismas credenciales de acceso legítimo que las solicitudes iniciadas por el propio usuario.

En general, la solución a CSRF consiste en introducir un campo de formulario oculto cuyo valor se utilice para realizar la autenticación, y no sólo por medio de las cookies, en solicitudes que puedan tener efectos duraderos. La comprobación de la cabecera HTTP referer también puede servir de ayuda.

"Hijacking JavaScript" es un tipo de ataque CSRF en el que una etiqueta <script> en el sitio web del atacante explota una vulnerabilidad en la página del sitio de la víctima que le hace devolver información privada, en forma de JSON o código JavaScript. Las posibles soluciones son: Que se requiera un token de autenticación en los parámetros de las peticiones POST y GET para aquellas peticiones que requieran devolver información privada del usuario.

Usar POST y nunca GET para solicitudes que devuelven información privada.

- Cómo funciona un script malicioso

Son porciones de código que pueden estar ocultas en webs completamente legítimas, cuya seguridad fue comprometida convirtiéndolas en cebos perfectos para las víctimas, que no van a sospechar nada ya que están visitando un sitio de confianza. De esta forma, los delincuentes pueden ejecutar código malicioso en los sistemas de los usuarios aprovechando alguna de las múltiples vulnerabilidades que pueden tener, tanto en los navegadores como en el propio sistema o en aplicaciones de terceros.

Las campañas de malvertising son un claro ejemplo: publicidades insertadas en sitios web que tienen código malicioso insertado. En caso de que un usuario haga clic, el cibercriminal podría tomar el control del dispositivo y ejecutar diversos ataques.

El motivo por el que se consigue la ejecución de este tipo de códigos de forma automática y sin la intervención del usuario tiene mucho que ver con los permisos que se otorgan cuando se configuran los sistemas. Aun a día de hoy, el número de cuentas de usuarios con permisos de administrador en sistemas Windows sigue siendo abrumador, y esto es totalmente innecesario en la mayoría de situaciones con las que alguien va a lidiar en su día a día.

Esto, unido a una mala configuración de alguna de las medidas de seguridad que incorpora el propio sistema Windows, como el UAC, hace que la gran mayoría de estos códigos maliciosos puedan campar a sus anchas en cientos de miles de ordenadores cada día. Tan solo con que se configurase esta capa de seguridad a un nivel medio/alto, se evitarían bastantes infecciones de este tipo, siempre que los usuarios sean conscientes de la importancia de leer las ventanas de alerta que muestra el sistema y no cometer el error cerrarlas o, peor aún, pulsar sobre el botón “Aceptar”.

- Cómo protegerte de scripts maliciosos

Para evitar este tipo de ataques, los usuarios han de tener en cuenta de que no existe la Web 100% segura en Internet y que deben tomar medidas para protegerse. La actualización del sistema operativo y de aquellas aplicaciones más vulnerables ante estos ataques (navegadores, Flash Player y Java, principalmente) resulta crucial para mitigar estos ataques. Pero a veces esto no es suficiente y es necesario contar con una solución de seguridad que sea capaz de detectar este tipo de scripts maliciosos, no solo los que utilicen JavaScript sino también los que usen PowerShell.

Conclusión

Sabemos que los scripts maliciosos hace años que están siendo utilizados por los ciberdelincuentes para propagar todo tipo de amenazas como troyanos, ransomware o bots. No obstante, ahora cuentas con medidas de seguridad adecuadas para, como mínimo, mitigar el alcance de este tipo de ataques. Tan solo debes tomarte la molestia de configurar aquellas medidas de seguridad que te protegen frente a un ataque de este tipo.

Bibliografía

http://www.ordenador.online/Programacion/JavaScript-Programaci%C3%B3n/Los-riesgos-con-JavaScript-.html

https://www.welivesecurity.com/la-es/2016/06/27/peligro-scripts-maliciosos-como-protegerte/

https://www.codejobs.biz/es/blog/2012/09/07/seguridad-informatica-que-es-una-vulnerabilidad-una-amenaza-y-un-riesgo

https://blogs.technet.microsoft.com/seguridad/2012/09/18/consejos-para-mitigar-el-riesgo-asociado-a-la-vulnerabilidad-de-ie-security-advisory-2757760/

https://es.wikipedia.org/wiki/JavaScript#Seguridad