Criptografía y Seguridad

Explotación de Vulnerabilidades: Cross Site Scripting

Facultad de Ciencias
Universidad Nacional Autónoma de México

13 de marzo de 2023

---

1. Objetivos

Generales:

Aprovecharse de una vulnerabilidad es una reacción natural después de saber la existencia de esta. Cross Site Scripting, también conocida como XSS, es un tipo de vulnerabilidad que nació junto a la creciente popularidad de los sitios web. Esta vulnerabilidad se aprovecha de sitios con fallas de seguridad que permite a los atacantes, implantar scripts maliciosos en un sitio de web legítimo para ejecutar códigos en el navegador y así afectar a algún internauta desprevenido.

Este tipo de vulnerabilidad explota la confianza del internauta hacia un sitio web, con el objetivo de realizar phishing o infectar al usuario de algún malware. Los alcances de un ataque de Cross Site Scripting pueden ser fatales para una compañía, gobierno o internauta. Los ataques hacia internautas han ido en aumento en la última década, siendo el internauta promedio el blanco favorito de los cibercriminales.

El objetivo general de esta práctica es que el alumno conozca la vulnerabilidad de Cross Site Scripting, la explote y conozca su alcance para poder prevenirla posteriormente.

Particulares:

2. Requisitos

- **Conocimientos previos:** Conocimientos básicos de lenguajes de desarrollo web y base de datos Mysql
- **Tiempo de realización sugerido:**
  8 horas.
- **Número de colaboradores:**
  2
- **Software a utilizar:**
  - XAMPP, php, javascript, Mysql, html y css.

3. Planteamiento

La vulnerabilidad de Cross Site Scripting, también conocida como XSS es una vulnerabilidad de seguridad que permite al atacante comprometer la interacción que un usuario tiene con un sitio vulnerable. Permite a un atacante evitar las normas y reglas que el programador había establecido en un principio en su sitio web, provocando que esta no realice lo que en un principio se planteó.

Esta vulnerabilidad es explotada con secuencias de comandos en sitios vulnerables permitiendo que un atacante se haga pasar por un usuario y lleve a cabo cualquier acción que el usuario pueda hacer y acceda a cualquiera de los datos del usuario. Si el usuario tiene un acceso privilegiado dentro de la aplicación, entonces el atacante podría obtener control total sobre toda la funcionalidad y los datos de la aplicación.

Cross Site Scripting funciona principalmente manipulando tanto al usuario como al sitio web de manera que regrese un código malicioso de JavaScript hacia los usuarios de dicho sitio. Cuándo el usuario en su inocencia, accede al sitio y de alguna manera ejecuta el exploit dentro de su propio navegador, el atacante puede comprometer completamente la interacción del usuario con el sitio, robando datos como cookies o contraseñas.

Este tipo de vulnerabilidad puede ser explotada de dos maneras: reflejada y almacenada. Algunos autores hacen también mención de una tercera llamada DOM XSS.

Reflected XSS, surge cuando una aplicación recibe datos en una solicitud HTTP e incluye esos datos en la respuesta inmediata de forma no segura.

Si un atacante puede controlar un script que se ejecuta en el navegador de la víctima, normalmente puede comprometer completamente a ese usuario. Puede realizar diversos tipos de ataques basándose en la confianza que inspira la plataforma en el usuario. Desde redirigir a otro sitio para robar información mediante phishing, hasta hacer que se descargue alguna amenaza y se ejecute en el sistema.

Las secuencias de comandos entre sitios almacenados, también conocidas como XSS persistentes, de segundo orden o Stored XSS, surgen cuando una aplicación recibe datos de una fuente que no es de confianza e incluye esos datos en sus respuestas HTTP posteriores de una manera no segura.

En términos de explotabilidad, la diferencia clave entre el XSS reflejado y el almacenado es que una vulnerabilidad XSS almacenada permite ataques que están autocontenidos dentro de la propia aplicación. El atacante no necesita encontrar una forma externa de inducir a otros usuarios a realizar una solicitud particular que contenga su exploit. Más bien, el atacante coloca su exploit en la propia aplicación y simplemente espera a que los usuarios lo encuentren.

Finalmente, las vulnerabilidades XSS basadas en DOM generalmente surgen cuando JavaScript toma datos de una fuente controlable por un atacante, como la URL, y los pasa a un receptor que admite la ejecución dinámica de código, como *eval()* o *innerHTML*. Esto permite a los atacantes ejecutar JavaScript malicioso, lo que normalmente les permite secuestrar las cuentas de otros usuarios.

La fuente más común para DOM XSS es la URL, a la que normalmente se accede con el objeto window.location. Un atacante puede construir un enlace para enviar a una víctima a una página vulnerable con una carga útil en la cadena de consulta y fragmentar partes de la URL.

Prevenir estas secuencias de comandos malicioso es necesario para mantener a los usuarios libres de peligro a un posible ataque que pueda comprometer su integridad. Muchas veces no es difícil prevenir estos ataques, e incluso los navegadores actualizados tienen a prevenir estos molestos intentos de ataques, pero aun así, es necesario que el programador tome medidas preventivas antes de liberar un sito a internet.

4. Desarrollo

La práctica consiste en aprovechar la vulnerabilidad de dos páginas web programadas en php, de manera que el alumno practique el cómo realizar estos ataques y analice el porqué es llevan a cabo para finalmente prevenirlos.

Para la realización de esta práctica se deberá contar con Mysql y XAMPP instalado en tu computadora. Para la instalación de XAMPP primero es necesario contar con php instalado. Para verificar si cuentas con php, puedes correr el siguiente comando:

```
~php -v

```

Si no cuentas con php, puedes instalarlo con el siguiente comando:

```
~sudo apt install php8.1-cli

```

Una vez instalado, puedes instalar XAMPP. XAMPP es un paquete de software libre, que consiste principalmente en el sistema de gestión de bases de datos MySQL, el servidor web Apache y los intérpretes para lenguajes de script PHP y Perl.

Para instalarlo en Ubuntu, debes descargar el archivo de la pagina oficial y asegurarte que se encuentre en tu carpeta personal. Una vez hecho esto basta con correr los siguientes comandos en tu terminal.

```
~sudo chmod a+x xampp-linux-x64-8.0.19-0-installer.run
~sudo ./xampp-linux-x64-8.0.19-0-installer.run

```

Los nombres de los comandos anteriores pueden variar dependiendo de la versión de XAMPP descargada.

Una vez descargado, hay que habilitar los servicios MySQL y FTP en la pestaña de Manage servers. Para comprobar que los servicios se estén ejecutando, abre un navegador y busca la siguiente url http://localhost/phpmyadmin, la cual es la página predeterminada de XAMPP.

Con esto solo falta crear la base de datos que usará el proyecto. Para esto deberás ya tener instalado Mysql. En caso de no tenerlo instalado, basta con correr los siguientes comandos.

```
~sudo apt update
~sudo apt install mysql-server

```

Una vez instalado, deberás configurarlos con el siguiente comando, el cual te llevará de la mano con una serie de instrucciones para realizar cambios en las opciones de seguridad.

```
~sudo mysql_secure_installation

```

Finalmente, deberás ajustar la autentificación y los privilegios de usuarios, para posteriormente conectar la base de datos a la página web.

```
~sudo mysql
mysql>ALTER USER 'nombre que quieras'@'localhost' IDENTIFIED WITH password BY 'contreseña';
mysql> FLUSH PRIVILEGES;
mysql> GRANT ALL PRIVILEGES ON * . * TO 'nombre que quieras'@'localhost';

```

La tabla que requiere la aplicación basta con que tenga una columna llamada comentarios, puedes usar mysqlworkbench o DBeaver para realizarla, o realizándola con los siguientes comandos.

```
mysql> CREATE DATABASE storexss;
mysql> CREATE TABLE comentarios (comentario VARCHAR(50) NOT NULL);

```

Finalmente, para comenzar con la práctica, deberás descomprimir la práctica en la carpeta de hdocs de XAMPP y modificar los primeros campos de storeXss.php, para terminar de conectar la aplicación a tu base de datos recién creada.

La práctica consiste en crear una serie de expliots que ataquen a 2 programas en php, los cuales fueron programados de tal manera que son vulnerables. Se recomienda usar una versión despreciada de navegador o uno que no cuente con tantas políticas de seguridad como Edge u Opera.

Los programas que se van a explotar se encuentran dentro de la carpeta *Practica03*. Lo que se pretende es que se realice un pentesting de caja blanca, en el cual tienen acceso total al código fuente de los siguientes programas:

- reflectedxss.php
- storexss.php

Para crear tu exploit, lo primero que deberás hacer es seguir los pasos para la realización de pentesting. La fase de recopilación de información ya la tienes en su mayoría, ya que cuentas con el código fuente a vulnerar, ahora solo falta el análisis de la vulnerabilidad, él ¿cómo?, y ¿dónde?.

Para esta etapa tendrás que analizar en que parte se puede aprovechar las series de vulnerabilidades explicadas a detalle en los sitios web. Deberás seguir cada uno de los ejercicios explotando y modificando los sitios con uso de esta vulnerabilidad.

Juega y experimenta con los sitios, crea exploits que logren hacer las tareas asignadas de manera que puedas anexarlas dentro de la web de la siguiente manera:

```
<script>window.location='http://localhost/Proyecto03/exploit.php</script>

```

Todo exploit hecho deberá estar bien documentado, con nombres intuitivos y dentro de la carpeta Practica03. En caso de ser solo un script simple que no requiera de la creación de un archivo exclusivo para él o que sea para agregarlo a la url para la sección de DOM XSS, bastará con crear un archivo llamado comandos-exploits.txt, donde agregarás dichos comandos. Cada comando deberá estar comentado, de manera que explique como se utiliza y que es lo que hace. También deberá explicar que ejercicio soluciona.

Varios comandos podrán solucionar un solo ejercicio por lo que es necesario especificar bien todo a base de comentarios. Imagina que estás creando una guía para que cualquiera pueda realizar los ejercicios.

Una vez realizado los ejercicios, no olvides el pdf donde documentarás todo lo realizado, siguiendo el mismo formato que usaste en la práctica pasada. Recuerda que el secreto para prevenir una vulnerabilidad viene desde como esta se documenta. La información debe ser intuitiva para que ayude a cualquier persona que sea víctima de ella, sea programador o no.

5. Entrada

Deberás crear como mínimo un exploit.php, comandos-exploits.txt y Readme.txt, que resuelva los ejercicios planteados en el sitio web. En caso de crear más, deberás documentarlos en el pdf.

En el Readme, deberás dar una guía general del proyecto y de tus archivos, de manera que guie a cualquier usuario por la práctica. Mencionarás las versiones utilizadas de tus softwares y el navegador.

Se dará un punto extra a los equipos que desarrollen un exploit original. Podrás realizar lo que desees y de la manera que quieran, siempre y cuando este bien documentado. En caso de que creas que tu exploit es peligroso, deberás advertirlo al principio del código. Diviértete.

6. Salida

Tus exploits deberán realizar las actividades planteadas en las páginas web y estos mismos ejercicios deberán estar documentados en el pdf, junto con imágenes como evidencia. Deberás explicar ampliamente cada paso que das, de manera que sea intuitivo la fase de análisis y explotación.

También deberás corregir los archivos php de manera que estos dejen de ser vulnerables a algún ataque XSS. No olvides explicarlo de igual manera en el pdf.

7. Procedimento

Se entregará un pdf debidamente documentado con tu análisis de los archivos vulnerables. Deberás crear el tu reporte con los siguientes apartados:

- **Recopilación**: En esta fase analizarás tus herramientas, el código y todo lo que cuentas para llevar a cabo la practica.
- **Análisis**: Identificarás la vulnerabilidad y como aprovecharla. Documentarás todo lo que llegues a encontrar.
- **Explotación**: En esta etapa documentarás la creación de tu exploit, y como llevas a cabo la explotación. Adjunta, imágenes como evidencia.
- **Post-explotación**: En esta fase explicarás tu exploit resultante, también las conclusiones a las cuales llegaste. Explicarás como corregir el código de la práctica y anexarás la corrección en su respectiva carpeta.
- **Imagina**: En esta fase intenta imaginar todo lo que podrías hacer utilizando esta vulnerabilidad. Explícalo detalladamente y reflexiona el alcance de esta vulnerabilidad.

Recuerda agregar la bibliografía correspondiente o la practica se evaluará en cero y tus datos completos y debidamente explicados. La tarea deberá ser entregada en tiempo y forma. En el Classroom únicamente uno del equipo subirá un .zip con todos los archivos y el pdf con los datos completos del equipo (nombre y número de cuenta).

8. Preguntas

1. ¿Qué tan frecuentes son las vulnerabilidades XSS?
2. ¿Cuál es la diferencia entre XSS y CSRF?
3. ¿Cuál es la diferencia entre XSS y SQL Injection?

4. ¿Cómo funciona un ataque CSRF?
5. ¿Cuál es el impacto de un ataque CSRF?
6. ¿Qué es CORS (cross-origin resource sharing)?
7. ¿Cómo funciona un XXE (XML external entity injection)?
8. ¿Qué es SSRF (Server-side request forgery)?

9. Bibliografía

Comprendiendo la vulnerabilidad XSS (Cross-site Scripting) en sitios web. (2015, abril 29). WeLiveSecurity. https://www.welivesecurity.com/la-es/2015/04/29/vulnerabilidad-xss-cross-site-scripting-sitios-web/

Fox, D. (2012). Cross Site Scripting (XSS). Datenschutz und Datensicherheit - DuD, 36(11), 840–840. https://doi.org/10.1007/s11623-012-0284-2

Rager, A., Hansen, R., Grossman, J., & Fogie, S. (2007). XSS attacks: Cross site scripting exploits and defense. Syngress Media.

Drake, M., & Virdó, H. (2020, mayo 19). Cómo instalar MySQL en Ubuntu 20.04. Digitalocean.com; DigitalOcean. https://www.digitalocean.com/community/tutorials/how-to-install-mysql-on-ubuntu-20-04-es