PREVENCIÓN DE VULNERABILIDADES EN EL DESARROLLO WEB

Juan Pablo Luna Felipez, M.Sc.
Docente Ingeniería Informatica
Universidad Nacional "Siglo XX"
jplunaf@gmail.com

RESUMEN

El constante avance de la ciencia informática hace posible el desarrollo e implementación de soluciones informáticas a diferentes necesidades y requerimientos de la sociedad.

Entre estas necesidades se encuentra el hecho de que las empresas e instituciones hayan llevado sus negocios a la WEB. Es impensable hoy en día encontrar empresa o instituciones sin presencia en Internet, ya esta actividad les brinda varios beneficios como: presencia corporativa, marketing, expansión de mercados, facilidad de llegar al cliente, etc.

Por tanto la existencia de sitios en la WEB en Internet es inmensa y estos sitios son desarrollados de diferente manera, buscando los desarrolladores brindar la mayor seguridad posible a estos sitios.

Sin embargo existen vulnerabilidades que pueden ser aprovechadas para vulnerar un sitio WEB con diversos propósitos, estando entre los más importantes el robo de información sensible.

Por ello se hace necesario conocer las vulnerabilidades existentes y la forma como se debe evitar estas vulnerabilidades a la hora de desarrollar un sitio Web.

El presente artículo aborda las vulnerabilidades conocidas a la hora de desarrollar para la web, así también como las medidas para prevenir ataques aprovechando estas vulnerabilidades.

PALABRAS CLAVE: *Vulnerabilidades, intromisiones, sitios Web, ataque Web, prevención, seguridad.*

ABSTRACT

The constant advances of the computer science makes possible the development and implementation of computer solutions to different necessities and requirements of the society.

Among these necessities is the fact that the companies and institutions have taken their business to the WEB. It is unthinkable today in day to find company or institutions without presence in Internet, this activity already offers them several benefits like: witnesses corporate, marketing, expansion of markets, easiness of arriving to the client, etc.

Therefore the existence of places in the WEB in Internet is immense and these places are developed in a different way, looking for the developers to offer the biggest security possible to these places.

However exists vulnerabilities that can be taken advantage of to harm a place WEB with diverse purposes, being among the most important the robbery of sensitive information.

For it becomes it necessary to know the existent vulnerabilities and the form like it should be avoided these vulnerabilities when developing a place Web.

The present article approaches the well-known vulnerabilities when developing places web, likewise as the measures to prevent attacks taking advantage of these vulnerabilities.

KEYWORDS: *Vulnerabilities, interferences, places Web, attack Web, prevention, security.*

1. INTRODUCCIÓN

El constante avance de la ciencia informática hace posible el desarrollo e implementación de soluciones informáticas a diferentes necesidades y requerimientos de la sociedad en diferentes niveles.

Uno de esos niveles es la presencia en Internet, ya que las empresas e instituciones han llevado su presencia y sus negocios a la WEB.

Es impensable hoy en día encontrar empresa o instituciones sin presencia sólida en Internet, ya que eso les da un sinfín de posibilidades como: presencia corporativa, marketing, expansión de mercados, facilidad de llegar al cliente, etc.

Por tanto la existencia de sitios en la WEB es inmensa.

Los sitios Web son desarrollados por empresas de desarrollo y programadores que emplean diferentes métodos y técnicas, buscando brindar la mayor calidad y seguridad posible a estos sitios.

Sin embargo existen vulnerabilidades que pueden ser aprovechadas para vulnerar un sitio WEB con diversos propósitos, estando entre los más importantes el robo de información sensible, el robo de contraseñas, manipulación de la información, etc.

Por lo que se hace necesario conocer las vulnerabilidades existentes y la forma como se debe evitar estas vulnerabilidades a la hora de desarrollar un sitio Web ya que no existe ningún sistema exento de fallos y totalmente seguro por lo que queda minimizar los riesgos para garantizar en lo posible la integridad de la información y del sitio web y de

2. VULNERABILIDAD

En Informática la palabra vulnerabilidad hace referencia a una debilidad que permite a un atacante violar la confidencialidad, integridad, disponibilidad, control de acceso y consistencia del sistema o de sus datos. (1)

Entonces un buen desarrollador con el fin de garantizar la seguridad de su sistema debe buscar prevenir que se viole la confidencialidad, integridad, disponibilidad, acceso y consistencias del sistema o sus datos.

Las vulnerabilidades se producen por varios motivos entre los principales se tiene: “fallos en el diseño de los sistemas, limitaciones tecnológicas o, fruto del desconocimiento” (1)

Por lo que se hace necesario conocer las vulnerabilidades existentes para prevenir las mismas, mejorar la seguridad del sistema y mejorar el nivel de programación.

En los diferentes niveles de seguridad se pueden apreciar las diferencias entre código seguro y bien estructurado y código vulnerable siguiendo malas prácticas de programación

3. TIPOS DE VULNERABILIDADES PARA SITIOS WEB

En el caso de los sitios web existen varios tipos de vulnerabilidades entre los que se tiene (1):

- Brute Force
- Command execution
- CSFR
- File inclusion
- SQL Injection
- File Upload
- XSS Reflected
- XSS Stored

4. BRUTE FORCE

El método de fuerza bruta o Brute Force, es empleado en criptografía para recuperar claves probando todas la combinaciones posibles hasta encontrar aquella que permite el acceso (1)

Este método se emplea en sitios web con el fin de encontrar usuario/contraseña validos probando combinaciones de palabras y letras y una vez obtenido uno valido, ingresar al sitio con un usuario/contraseña.

Para realizar este ataque se deben tomar en cuenta dos aspectos:

- La longitud de palabra buscada.
- El alfabeto empleado para para obtener todas las combinaciones.

El número de combinaciones existentes es igual a la longitud de palabra elevado a la longitud del alfabeto, por lo que cuanto mayor es el número de elementos del alfabeto y mayor la longitud de ésta, entonces existen mayor es el número de combinaciones y se requiere mayor tiempo en obtener una posible respuesta.

Como el número de combinaciones para obtener una respuesta favorable es muy elevado y requiere mucho tiempo de cómputo, entonces este método se suele emplear conjuntamente con el método de ataque por diccionario, el cual consiste en almacenar cadenas o caracteres y probar todas las combinaciones con estas palabras o caracteres seleccionados.

Para elegir las palabras o caracteres que se probaran, principalmente se eligen tomando en cuenta el idioma y preferencias del usuario , lo que se conoce como ingeniería social, de forma que permita fácilmente lograr encontrar la clave buscada. En internet existen diversos sitios que ofrecen diccionarios como http://www.insidepro.com/dictionaries.php que ofrece diccionarios en base a un idioma determinado.
Sin embargo a mayor tamaño del diccionario, mayor el tiempo que se requerirá para calcular todas y cada una de las posibilidades, pero tiene el beneficio de que es mayor la probabilidad obtener una clave satisfactoria. Existen también herramientas que permiten crear diccionarios en base a contenidos de las páginas web y eligiendo palabras relacionadas como el sitio *http://www.darknet.org.uk/2009/01/cewl-custom-word-list-generator-tool-for-password-cracking/,* este sitio genera diccionarios en a las palabras del contenido Web y con palabras relacionadas a estas.

Para emplear el método de fuerza bruta combinada con el diccionario se aplica herramientas que lo implementan, entre las más destacadas se encuentra Hydra, herramienta que se encuentra disponible en: http://www.thc.org/thc-hydra/. Esta herramienta trabaja por consola y con comandos; también se tiene la herramienta Burp Suite Free disponible en *http://www.portswigger.net/burp/* que presenta una interfaz visual; ambas herramientas permiten obtener el usuario y contraseña de un sitio web aplicando los métodos de fuerza bruta y ataque por diccionario, aunque también ofrecen otras utilidades adicionales.

A continuación se presenta un ejemplo de comando de Hydra para un ataque de fuerza bruta combinado con diccionario:

```
$ hydra -l admin -P diccionario.dic www.misitio.com
http-get-form
"/vulnerabilities/brute/
index.php:username=^USER^&password=^PASS^&Lo
gin=Login:Usern
ame and/or error en contraseña.:H=Cookie:
security=low;
PHPSESSID=216k4v6mcli2ptscohj3de9ld4"
```

Una vez que se han hecho los ataque por fuerza bruta, debe esperarse un tiempo considerable en función del tamaño del diccionario para obtener una respuesta favorable, no se logra en tiempos cortos.

Para prevenir esta vulnerabilidad, primeramente se recomienda ampliamente emplear el método sleep() en la programación de php, de forma que se logre un retardo luego del ingreso del usuario/ contraseña, esto causa que el programe se retarde más y por tanto el ataque para recuperación de usuarios/contraseñas se vuelve extremadamente lento y a veces ineficaz.

Como segunda medida adicional se recomienda altamente emplear los Captchas, estos elementos hacen necesario la intervención de un usuario y permiten diferenciar al usuario humano de un sistema automático que pretende obtener el usuario/contraseña.

Como tercera medida también se recomienda bloquear aquellas IPs que han ingresado varios usuarios/contraseña y tienen varios intentos fallidos en un determinado de tiempo, por lo que se hace necesario crear una tabla de intentos de acceso/origen para este propósito.

Finalmente como otra medida adicional se recomienda emplear tokens aleatorios en los formularios que se generen en cada sesión.

5. COMMAND EXECUTION

Principalmente esta vulnerabilidad se emplea para ejecutar comandos del sistema operativo principalmente en sistemas *nix a partir de un sitio WEB vulnerable, la infinidad de usos a la hora de ejecutar comandos es grande, pero se busca principalmente obtener usuarios/contraseñas con acceso al sistema, la modificación de archivos y otros.

La única limitante de este tipo de ataque es que se emplea el usuario apache, y por tanto las posibilidades de ejecución de comandos están ligadas a los permisos que tenga esta cuenta.

La ejecución de comandos se aplica en sitios que a través de sus formularios y de elementos inputs permiten la ejecución de comandos, es decir son sitios que ofrecen capacidad de ejecutar algún comando pero con solo un determinado fin como realizar un ping.

Entonces esta vulnerabilidad se aprovecha cuando en a la hora de introducir el parámetro requerido en el sitio además se le concatena otro comando, ya que Linux permite la concatenación de comandos con los siguientes operadores (1).

- Ampersand (“&”):dos o más comandos se ejecutan de manera simultánea.

```
$ cd.. & rmdir directorio
```

- Barra (“|”): la salida del primer comando se convierte en la entrada del segundo comando.

```
$ find . | xargs grep cadena_a_buscar
```

- Doble ampersand (“&&”) o AND: el segundo comando sólo se ejecutará si el primero termina y se ejecuta con éxito.

$ make && make install

- Doble barra (“||”) u operador OR: el segundo comando se ejecutará si el primero NO termina con
Éxito.

$ cp /home/pepe/*.doc /backup/usuarios/pepe || echo "probando"

- Punto y coma (“;”): el segundo comando se ejecutará sin importar el resultado del primer comando

$ cd carpeta ; mkdir carpeta

Entonces como se puede ver existe múltiples posibilidades de inserción de comandos en este tipo de sitios, si estos no están adecuadamente protegidos.

Para evitar esta vulnerabilidad primeramente se debe emplear la función str_replace() de PHP que permite realizar un filtrado de los caracteres pasadas por URL, eliminando los operadores doble ampersand y el punto y coma.

```
<?php
$substituir= array(
'&&' => '',
';' => '',
);
$target = str_replace( array_keys( $substitutir ), $substituir, $target );
?>
```

Y seguidamente se debe emplear la función **stripslashes()** con el fin de eliminar las barras, con estas dos medidas se evita que introduzcan comandos adicionales concatenados.

6. CROSS SITE REQUEST FORGERY O CSRF

CSRF (del inglés *Cross-site request forgery* o falsificación de petición en sitios cruzados) es un tipo de *exploit* en el que comandos no autorizados son transmitidos por un usuario en el cual el sitio web confía (1).

También es conocido por otros nombres como ataque de un click, ataque automático, XSRF, enlace hostil y cabalgamiento de sesión.

Este tipo de vulnerabilidad hace que un usuario ya autenticado en un sitio Web, realice una acción confiando en el sitio web en el que se encuentra, normalmente un cambio de contraseña, sin embargo esto es atendido por otro sitio o página web sin que el usuario sea consciente de ello.

Por tanto con esta vulnerabilidad el usuario activa alguna acción en el sitio web original, se captura la petición y se desvía la petición a otro sitio web logrando inclusive cambiar el usuario y contraseña del usuario sin su intervención, ya que solo basta conocer el usuario y como este ya está autenticado, solo basta asignarle una nueva contraseña .

El usuario piensa que como el sitio es de confianza todo es normal, sin embargo sin su conocimiento suele enviarse solicitudes a un segundo sitio web.

Esta vulnerabilidad se suele aprovechar con algunos programas para capturar la petición como foxiproxy *http://getfoxyproxy.org/* el cual capturara todo el tráfico web desde un puerto definido y también se emplea programas que muestren información de las peticiones de formularios como **WASP CSRFTester** que se encuentra disponible en: *https://www.owasp.org/index.php/Category:OWASP_CSRFTester_Project/es*

El programa OWASP CSRFTester permite generar una petición explicita con un fichero HTML o de forma implícita pidiendo el cambio de contraseña con una imagen u otro elemento, el usuario confiando en el sitio puede realizar esta acción sin notarlo y cuando desee ingresar al sitio no podrá ya que su contraseña habrá sido cambiada.

Para prevenir este ataque primeramente se debe confirmar que el cambio de contraseña viene del propio servidor donde está alojada la página web y en caso contrario no atenderla, sin embargo esto también puede vulnerase con un script que al ser ejecutado por la víctima en un servidor externo, ésta se haría pasar como si fuese ejecutada desde el propio servidor.
Otra segunda medida es la solicitud de escritura y rescritura de una contraseña nueva.

Y finalmente la tercera medida de seguridad adicional es el uso de tokens de autorización para los usuarios en cada sesión, el cual se generaría a partir de un script con los datos actuales de cada sesión.

7. FILE INCLUSION

Se trata de incluir archivos locales o externos y ejecutarlos en el servidor web (2)

Este tipo de vulnerabilidad aprovecha que la aplicación requiere incluir algún archivo interno o externo que el sistema solicita y busca y esto se ve en la URL, por tanto

en vez de cargar el archivo solicitado, se aprovecha esta vulnerabilidad cargando otro archivo que tenga fines diferentes.

Por los archivos que se pueden ejecutar en este ataque se diferencian dos tipos diferentes de ataques: La inclusión Local de archivos (LFI *Local File Inclusion)* y la Inclusión remota de archivos (RFI *Remote File Inclusion)*.

Esta vulnerabilidad se debe a una mala programación a la hora de utilizar las funciones *include*, *include_once*, *require*, *require_once* y *fopen,* las cuales hacen referencia a que el sistema requieres ciertos archivos durante su funcionamiento.

Esta vulnerabilidad permitiría internamente ejecutar archivos del servidor como el archivo etc/passwd, también se puede crear directorios, cargar scritps maliciosos, etc y cuyos resultados se podrían visualizar inmediatamente en el sitio web, pudiéndose recuperar usuarios y contraseñas aunque estas últimas se encontrarían normalmente cifradas con MD5, pero esto es fácil de descifrar ya que existen varias herramientas online que permiten esta tarea.

Para prevenir este ataque se recomienda emplear la función str_replace() para eliminar las cadenas de texto que tengan las cadenas "http" y "https" y filtrar la información de entrada.

Como segunda medida se debe mantener desactivada la función *allow_url_include* que controla si se permiten incluir scripts utilizando direcciones web.

Como tercera medida se recomienda a la hora de programar no emplear includes utilizando el paso por URL.

8. SQL INJECTION

Consiste en inyectar código SQL invasor dentro del código SQL que ataca a la base de datos y que tiene por objetivo cambiar el funcionamiento normal del programa con carácter malicioso (1).

Esta vulnerabilidad se da cuando se solicita al usuario datos para realizar búsquedas o completar registros para almacenarlos en una base de datos a través de formularios, por lo que se aprovecha esta situación para concatenar código SQL.

Este ataque permite obtener toda la información que se desee de una base de datos y otros daños.

Para aprovechar esta vulnerabilidad, las consultas SQL se pueden concatenar para realizar una inyección de código SQL con:

1" OR 1=1----"
1" OR "1" = "1
"OR"="
" OR 1=1----"
" OR 0=0 ----"
" OR "x"="x

Para prevenir esta situación inicialmente se recomienda emplear la función mysql_real_scape_string(), que restringe los caracteres especiales como comillas simples.

En segundo lugar se recomienda emplear las funciones addcslashes() para restringir caracteres como \n, \r etc.

En tercer lugar se recomienda emplear la función mysql_real_escape_string() que restringirá los caracteres especiales y también stripslashes() que quita las barras de un string con comillas.

Finalmente se recomienda no generar sentencias SQL de forma dinámica, no utilizar cuentas con privilegios administrativos y emplear la librería mysqli que ofrece mucha más seguridad frente a la vulnerabilidad SQL Injection.

9. FILE UPLOAD

Este tipo de vulnerabilidad aprovecha que se realiza una petición al usuario de carga de un archivo (1).

Existen en internet varios sitios que por diversos motivos solicitan la carga de un archivo al servidor, esto generalmente se solicita por medio de un formulario que solicita la carga de archivos de algún tipo de archivos como documentos, imágenes, audio o video, entonces en vez de subir un archivo del tipo deseado por el sitio web, se aprovecha de subir código en otro archivo como un script php.

Esta situación conlleva mucho riesgo ya que puede aprovecharse esta carga de archivos para ingresar al sistema donde esté alojada la página web.

Esta vulnerabilidad aprovecha inicialmente que la página web permite subir cualquier tipo de archivo, entonces se suele subir un archivo que ejecute comandos sobre el sistema.

Para prevenir este tipo de ataques primeramente se realiza una validación del tipo de fichero que se está subiendo.

Pero la validación del tipo de fichero puede ser vulnerada haciendo pasar un fichero ejecutable como si fuera una documentos, imagen, audio, etc., para ello se emplea herramientas como Tamper Data que se encuentra disponible en *https://addons.mozilla.org/es/firefox/addon/tamper-data/* y que es una extensión del explorador web Firefox para modificar el tipo de fichero y que sea aceptado por el servidor.

Es decir se hace pasar el archivo de comandos o ejecutable como si fuese un archivo de imagen, documento, etc. Modificando sus cabeceras.

Por lo tanto además se debe tener en cuenta otras medidas adicionales como:

- Limitar el tamaño del archivo.
- El archivo recientemente subido por ninguna razón debe ser accesible inmediatamente al usuario.
- Una vez subido el archivo se debe cambiar el nombre por uno aleatorio al almacenarlo.
- El archivo debe almacenarse fuera de la carpeta de publicación.

Con estas medidas se evitara este tipo de vulnerabilidad.

10. XSS REFLECTED

Es una vulnerabilidad que consiste en inyectar código HTML o Javascript en una web y que el navegador ejecute este código inyectado cuando se accede a la página. (1)

Este tipo de ataque se realiza en sitios que reciben la información sobretodo vía GET, es decir a la hora de pasar un atributo, cuando el sistema realiza la búsqueda, se cambiaría el parámetro de búsqueda por un script u otro código en los campos de búsqueda de formularios.

Ejemplo:

busqueda.php?d=cadena

se podría reemplazar por busqueda.php?d=<script type="text/javascript">codigo();</script>

El peligro de este ataque es que de esta forma se puede incluso obtener la contraseña del usuario.

Primeramente para prevenir este tipo de ataque se debe emplear la función str_replace() para cambiar los símbolos “<” y ”>” y filtrar las tags <*script*> <object>, <applet>, <embed> y <form>.tanto en minúscula como en mayúscula.

Seguidamente se debe verificar que el tipo de dato introducido y la longitud de cada campo sean lo esperado.

Como tercera medida se debe emplear la función htmlspecialchars() para convertir caracteres especiales en entidades HTML.

Y por último se debe emplear la función strip_tags() para retirar *HTML y PHP* de un string.

11. XSS STORED

Consiste en embeber código HTML Javascript en una aplicación web de forma permanente, pudiendo llegar incluso a modificar la propia interfaz de un sitio web (*defacement*) (1).

Esta vulnerabilidad compromete la seguridad del usuario y no la del servidor y es parecida a la vulnerabilidad XSS reflected.

La diferencia frente a XSS reflected es que este tipo de vulnerabilidad se almacena en la base de datos y por tanto se mantiene almacenado en el tiempo.

Las amenazas de este ataque puedes ser varias como: inyectar código que robe cookies, inyectar código que redireccione la página web a una página externa, realizar un cambio en la interfaz(*deface)*, etc.

Para prevenir este tipo de amenaza primeramente se debe emplear las mismas medidas preventivas que XSS reflected.

En segundo lugar se debe emplear las funciones stripslashes() que quita las barras de una cadena y mysql_real_escape_string() que escapa caracteres especiales en un string para su uso en una sentencia SQL.

12. CONCLUSIONES

Con el crecimiento exponencial de los sitios web y con la cada vez mayor información disponible en estos sitios, se debe priorizar la seguridad de estos sitios para evitar que sean vulnerados.

Todo sitio web que no adopta medidas de seguridad en su programación se encuentra expuesto y es blanco de sufrir ataques.

Existen varios tipos de ataques que aprovechan distintas vulnerabilidades de un sitio web ya que no existe un sistema totalmente libre de fallos, por lo que los programadores deben buscar la forma de minimizar estos riesgos con el fin de garantizar y mantener la

integridad de la del sitio y de la información que contiene.

Por tanto deben conocerse las vulnerabilidades existentes y tomarse en cuenta todas las medidas para evitar esta vulnerabilidades y para evitar nuevos tipos de vulnerabilidades, tomar en cuenta medidas como validar toda información que ingrese, configurar adecuadamente el servidor con las directivas de seguridad correspondientes, manejar adecuadamente el sistema de permisos para el administrador apache, no emplear directivas que ponen podrían colocar en riesgo el sitio, tener cuidado con los permisos de subir los archivos y mantener actualizado constantemente los navegadores.

13. BIBLIOGRAFÍA

1. **Waysen Restoy, Javier y Francisco, Perez Sanchez.** *Web Vulnerable DVWA.* España : Universidad de Almeria, 2012.

2. **Inteco.** *Seguridad en Sitios Web.* Madrid : Inteco, 2011.

3. **Julio, Gomez Lopez.** *Hackers Aaprende a Atacar y Defenderte.* s.l. : Ra-Ma, 2010. ISBN 978-84-7897-955-4.

4. **Gomez, Julio, Villar, Eugenio y Alfredo, Alcayde.** *Seguridad en Sistemas Operativos Windows y GNU/Linux.* s.l. : Ra-Ma, 2011. ISBN: 978-84-9964-116-4.

5. **DWMA.** Damn Vulnerable Web Application. *sitio web de Damn Vulnerable Web Application.* [En línea] dvwa. [Citado el: 07 de 05 de 2016.] http://www.dvwa.co.uk/..

6. Wikipedia. Wikipedia. [En línea] [Citado el: 07 de 05 de 2016.] www.wikipedia.org.