



UNIVERSIDAD AUTÓNOMA DE NUEVO LEÓN

FACULTAD DE CIENCIAS FÍSICO MATEMÁTICAS

DISEÑO ORIENTADO A OBJETOS

TAREA 2

MIGUEL ANGEL SALAZAR SANTILLAN

Manuel Ricardo Lara Amaya

1623929

Aspectos Básicos de la Seguridad en Aplicaciones Web

Introducción a la seguridad en sistemas web.

seguridad de la información manejada cotidianamente. Sitios de comercio electrónico, servicios, bancos e incluso redes sociales contienen información sensible que en la mayoría de los casos resulta ser muy importante.

Se puede decir que uno de los puntos más críticos de la seguridad en Internet son las herramientas que interactúan de forma directa con los usuarios, en este caso los servidores web. Es común escuchar sobre fallas en los sistemas de protección de los servidores más frecuentemente utilizados, por ejemplo Apache, NGINX, IIS, etc. (Build With, 2016) O en los lenguajes de programación en que son escritas las aplicaciones.

Debemos entender que programar aplicaciones web seguras no es una tarea fácil, ya que requiere por parte del programador, no sólo cumplir con el objetivo funcional básico de la aplicación, sino una concepción general de los riesgos que puede correr la información procesada por el sistema.

1. Problemas principales en la programación de sistemas web

Gran parte de los problemas de seguridad en las aplicaciones web son causados por la falta de seguimiento en dos rubros muy importantes de los que depende cualquier aplicación, las entradas y salidas del sistema.

2. *Prácticas básicas de seguridad web*

2.1 Balancear riesgo y usabilidad

Si bien la usabilidad y la seguridad en una aplicación web no son excluyentes una de la otra, alguna medida tomada para incrementar la seguridad con frecuencia afecta la usabilidad. Normalmente siempre se debe pensar en las maneras en que usuarios ilegítimos nos pueden atacar y la facilidad de uso para los usuarios legítimos.

Es conveniente emplear medidas de seguridad que sean transparentes a los usuarios y que no resulten engorrosas en su empleo. Por ejemplo, el uso de un *login* que solicita el nombre de usuario y contraseña, permite controlar el acceso de los usuarios hacia secciones restringidas de la aplicación. Este paso adicional, es una característica que impacta en la rapidez de acceso a la información por parte del usuario, pero que proporciona un elemento adicional de protección.

2.2 Rastrear el paso de los datos

Es muy importante mantener conocimiento de los pasos que ha recorrido la información en todo momento. Conocer de dónde vienen los datos y hacia dónde van. En muchas ocasiones lograr esto puede ser complicado, especialmente sin un conocimiento profundo de cómo funcionan las aplicaciones web.

En las aplicaciones web, existen maneras de distinguir los orígenes de los datos y poder así reconocer cuando los datos pueden ser dignos de confianza y cuando no.

Normalmente existen arreglos globales en la aplicación (por ejemplo en PHP los arreglos `$_GET`, `$_POST`, `$_COOKIE` y `$_SESSION` entre otros) que sirven para identificar de forma clara las entradas proporcionadas por el usuario. Si esto lo combinamos con una convención estricta para el nombrado de las variables podemos tener un control sobre el origen de los datos usados en el código.

2.3 Filtrar entradas

El filtrado es una de las piedras angulares de la seguridad en aplicaciones web. Es el proceso por el cual se prueba la validez de los datos. Si nos aseguramos que los datos son filtrados apropiadamente al entrar, podemos eliminar el riesgo de que datos contaminados sean usados para provocar funcionamientos no deseados en la aplicación.

Existen muchos puntos de vista diferentes sobre cómo realizar el filtrado o proceso de limpieza. Lo que usualmente se recomienda es ver al filtrado como un proceso de inspección, no debemos tratar de corregir los datos, es mejor forzar a los usuarios a jugar con las reglas válidas.

2.4 Escapado de salidas

Otro punto importante de la seguridad es el proceso de escapado y su contraparte para codificar o decodificar caracteres especiales de tal forma que su significado original sea preservado. Si llegamos a utilizar una codificación en particular es necesario conocer los caracteres reservados los cuales serán necesarios escapar.

El proceso de escapado debe estar compuesto a su vez por los siguientes pasos:

- Identificar las salidas.
- Escapar las salidas.
- Distinguir entre datos escapados y no escapados.

El proceso de escapado debe adecuarse al tipo de salida de que se trate (si es al cliente, a la base de datos, etcétera). Para la mayoría de los destinatarios, existen funciones nativas en los lenguajes de programación para esta finalidad. En alguno

de los casos como podría ser el de base de datos es importante incluso observar la codificación en la que son enviados.

3. Clasificación de ataques web.

3.1 Ataques URL de tipo semántico

Este tipo de ataques involucran a un usuario modificando la URL a modo de descubrir acciones a realizar que originalmente no están planeadas para ser manejadas correctamente por el servidor. La implementación de cualquier formulario debe contemplar validaciones necesarias para evitar esas acciones y se deben realizar adecuaciones de acuerdo a nuestras entradas, en la ilustración 5 se muestra un formulario de *login* con campos de usuario y contraseña.

Tipo	Nombre	Descripción
Tipo 0	Ataque basado en el DOM o local	Si un código de JavaScript accede a una URL como un parámetro de una petición al servidor y utiliza esta información para escribir HTML en la misma página sin ser codificada empleando entidades HTML
Tipo 1	Ataque no persistente o reflejado	Si los datos no validados por el usuario son incluidos en la página resultante sin codificación HTML, se le permite al cliente inyectar código en la página dinámica.
Tipo 2	Ataque persistente o almacenado	La información proporcionada por el usuario es almacenada en la base de datos, en el sistema de archivos o algún otro lugar; después es mostrada a otros usuarios que visiten la página

3.3 Ataques de Cross-Site Request Forgery

Este tipo de ataque permite al atacante enviar peticiones HTTP a voluntad desde la máquina de la víctima. Es difícil determinar cuándo una petición HTML se ha originado por un ataque de este tipo.

Cuando un atacante conoce el formato que debe tener una URL para lograr la ejecución de una acción en el sistema, ha logrado encontrar la posibilidad de explotar este tipo de ataques. Ahora lo que necesita el atacante es simplemente hacer que una víctima visite la URL.

Un recurso que se utiliza comúnmente para realizar este tipo de ataques suele tener embebida la petición en una imagen. En este caso el atacante sólo necesita crear alguna etiqueta HTML del siguiente tipo:

```

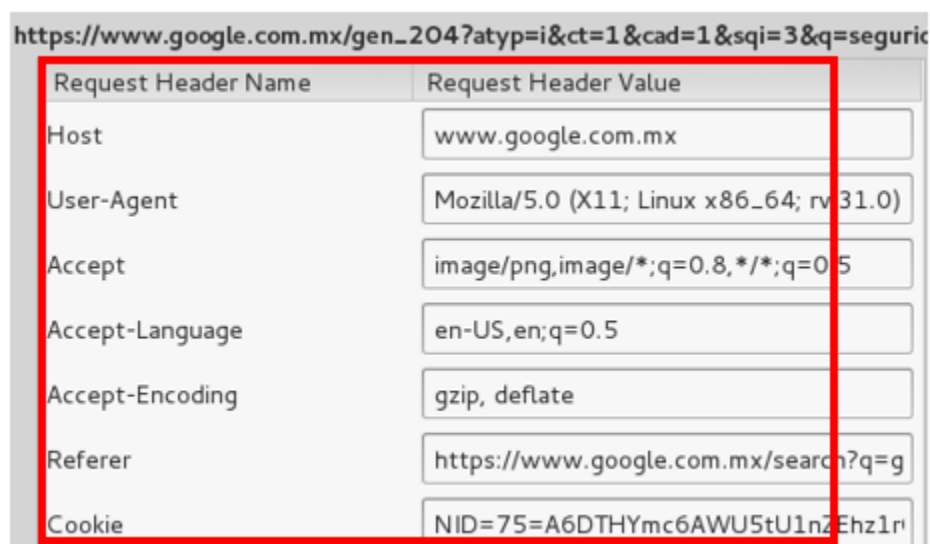
```

Existen acciones que podemos tomar para contrarrestar este tipo de ataques, una de estas es preferir el método POST para el procesamiento de formas en lugar del GET, otra posibilidad es solicitar confirmación por parte del solicitante antes de realizar los procesos (a costa de reducir la usabilidad en la aplicación).

3.4 Peticiones HTTP falsificadas

Un ataque más sofisticado que el anterior es enviar peticiones falsas empleando herramientas especiales para este propósito.

Para ello, se emplean herramientas de línea de comandos o plugins agregados a los navegadores, con estos se pone a la escucha de los servicios web que típicamente se conectan a través del puerto 80. En la ilustración 7 podemos observar los campos que pueden modificarse con Tamper Data.



Request Header Name	Request Header Value
Host	www.google.com.mx
User-Agent	Mozilla/5.0 (X11; Linux x86_64; rv:31.0)
Accept	image/png,image/*;q=0.8,*/*;q=0.5
Accept-Language	en-US,en;q=0.5
Accept-Encoding	gzip, deflate
Referer	https://www.google.com.mx/search?q=g
Cookie	NID=75=A6DTHYmc6AWU5tU1nZ_Ehz1r

Ilustración 7. Headers interceptados.

En realidad un atacante puede confeccionar a gusto sus peticiones HTTP, la fortaleza de nuestro sistema será medible por su capacidad de detectar que peticiones recibidas deben ser escuchadas y procesadas de acuerdo a los parámetros y valores que se vayan a recibir.

4. Seguridad de las aplicaciones y su relación con las bases de datos

La mayoría de las aplicaciones web son usadas como un conducto entre fuentes de información y el usuario, esto es, las aplicaciones web son usadas para interactuar con una base de datos.

Muchos programadores no dan importancia al filtrado de datos provenientes de una consulta a la base de datos, debido a que consideran a esta fuente como confiable. Aunque el riesgo a primera vista parecería menor, es una práctica recomendable no confiar en la seguridad de la base de datos e implementar la seguridad a fondo y con redundancia. De esta manera, si algún dato malicioso fue inyectado a la base de datos, nuestra lógica de filtrado puede percatarse de ello.

En la actualidad el crecimiento de internet ha impactado directamente.