

Asegurando un servidor Linux Fedora Core

**Apache 2.x / PHP 4.x / vsftpd / sshd / MySQL / SynAckFlood /
Iptables / yum**

Versión: 1.1
por Aarón Mizrachi
aaron@unmanarc.com

Atención:

No nos hacemos responsables por el mal uso de este documento, no ofrecemos ninguna garantía sobre el documento, todas las marcas expuestas en este documento están registradas a sus respectivos autores.

En la actualidad existe una configuración para los servidores apache que se ha vuelto muy común, es la configuración de la "NCSA server configuration", sin embargo, las configuraciones que traen por defecto dejan mucho que desear, si bien es cierto que originalmente y para los parametros "regulares" esa configuración sirve, en este articulo pretendemos reforzar al máximo la seguridad en las configuraciones del PHP y del Apache.

Nosotros intentaremos mejorar las configuraciones, mediante dos métodos que se complementan, el primero, seguridad preventiva en el servidor, por ejemplo, activar "modo seguro" en php, no permitir que se ejecute el comando TRACE o TRACK en el servidor, asegurar la estructura de directorios entre otras cosas. Y como complemento, utilizaremos métodos de "Seguridad por oscuridad", que despistarán a muchos scanners.

Cambios en la configuración httpd.conf:

httpd.conf, comunmente (en distribuciones basadas en redhat) se encuentra en el directorio "/etc/httpd/conf", para editar este archivo de configuración se necesita estar en modo root, y finalmente usted deberá ejecutar: "service httpd restart" para que los cambios surtan efecto.

Basados en la configuración original de la NCSA, les mostraremos alguno de los cambios que realizamos:

1. En la linea que dice: ServerTokens, asegurarse que diga:

```
ServerTokens min
```

Eso le agrega un poco de seguridad por oscuridad al servidor, en caso de que un hacker decida revisar la version del apache con algun script ordinario, probablemente no logre obtenerla, y si por mala suerte no actualizamos el servidor, y tiene una vulnerabilidad, probablemente el hacker no intente explotarla.

2. Cambie el nombre de ServerAdmin a un email especial para entregarlo al público, es decir, use algo como "httpadmin@yourdomain.com", de modo que no pueda ser usado (en caso de que el servidor lo deje aparecer en alguna página), para enviarle spam.
3. Verifique que los valores de User y Group son "apache", a mi juicio, para la configuración que haremos a continuación no es suficiente con usar "nobody". Necesitamos distinguir entre "nobody" y apache.
4. Por razones de seguridad es bueno desactivar el keep-alive, para ello en la opción KeepAlive, se debe poner: "KeepAlive Off"
5. En el directorio /var/www/html (y en otros) normalmente se activa la siguiente opción: "Options Indexes FollowSymLinks", lamentablemente el indexes puede causar que sean expuestos directorios que usted no desee exponer. Por ello recomiendo solo dejar: "Options FollowSymLinks"
6. Si usted desea, puede usar un directorio publico que si admita indexes:

```
<Directory "/var/www/html/public">
  Options FollowSymLinks Indexes
  AllowOverride None
  Order allow,deny
  Allow from all
</Directory>
```

7. Es importante que el “mod_userdir” este deshabilitado si realmente no lo requiere, es preferible manejar la situación con subdominios (virtual-hosts) para evitar dar el nombre de usuario de algún usuario al usar: ~user

```
<IfModule mod_userdir.c>
  UserDir disable
</IfModule>
```

8. El directory index podría ser algo como:

```
DirectoryIndex index.php index.html index.htm index.asp
```

9. Probablemente necesitamos, crear “una pantalla” para que la gente crea que su servidor es en efecto un servidor Microsoft-IIS ®, es por ello que usted puede crear archivos .asp con contenido de php y que se ejecuten de igual manera con tan solo agregar:

```
AddType application/x-httpd-php .asp
```

10. El CustomLog original de apache no guarda información que podría ser vital a la hora de rastrear un ataque, por ello, todas las ocurrencias de CustomLog deberían ser:

```
CustomLog logs/access_log combined
```

11. ServerSignature debe estar apagado para no revelar de forma sencilla varios datos del servidor, que podrían llevar al hacker a intentar encontrar una vulnerabilidad.

```
ServerSignature Off
```

12. Debe quedar comentada la sección del manual de apache, para evitar reconocimiento:

```
#AliasMatch ^/manual(?:/(?:de|en|fr|ja|ko|ru))?(/.*)?$ "/var/www/manual$1"
#<Directory "/var/www/manual">
#  Options Indexes
#  AllowOverride None
#  Order allow,deny
#  Allow from all
#
#  <Files *.html>
#    SetHandler type-map
#  </Files>
#
#  SetEnvIf Request_URI ^/manual/de/ prefer-language=de
#  SetEnvIf Request_URI ^/manual/en/ prefer-language=en
#  SetEnvIf Request_URI ^/manual/fr/ prefer-language=fr
#  SetEnvIf Request_URI ^/manual/ja/ prefer-language=ja
#  SetEnvIf Request_URI ^/manual/ko/ prefer-language=ko
#  SetEnvIf Request_URI ^/manual/ru/ prefer-language=ru
```

```
# RedirectMatch 301 ^/manual(?:/(de|en|fr|ja|ko|ru)){2,}(/.*)?$ /manual/$1$2
#</Directory>
```

13. Personalize su /error, de modo que emita usted sus propios 404, es importante destacar que en sus HTML/PHP de error no debe poner información que el usuario ingrese, simplemente límitese a decir: “La página no puede ser accesada”, sin explicar el número de error que ocurrió:

```
ErrorDocument 400 /error/error.htm
ErrorDocument 401 /error/error.htm
ErrorDocument 403 /error/error.htm

ErrorDocument 405 /error/error.htm
ErrorDocument 406 /error/error.htm
ErrorDocument 407 /error/error.htm
ErrorDocument 410 /error/error.htm
ErrorDocument 412 /error/error.htm
ErrorDocument 414 /error/error.htm
ErrorDocument 500 /error/error.htm
ErrorDocument 501 /error/error.htm
ErrorDocument 502 /error/error.htm

ErrorDocument 404 /error/error.htm
```

En tal caso, sabemos por unas líneas más arriba que:

```
Alias /error/ "/var/www/error/"
```

por lo tanto “/var/www/error/error.htm” debe contener algo como “Error al cargar la página”. Es importante no darle al hacker detalles de porque falló el acceso a cierto sitio, de modo que no pueda hacer suposiciones (tal directorio existe, etc)

14. Probablemente necesitemos usar “virtual hosts”, así que vamos a mostrarles como hacerlo, en primera instancia, debe estar activada la línea:

```
NameVirtualHost *:80
```

Luego de eso, podemos usar el siguiente modelo para cada uno de los virtual-hosts:

```
<VirtualHost *:80>
    DocumentRoot /var/www/html
    ServerName www.unmanarc.com
    CustomLog logs/www.unmanarc.com-access.log combined
    RewriteEngine on
    RewriteCond %{REQUEST_METHOD} ^(TRACE|TRACK)
    RewriteRule .* - [F]
    RewriteRule ^(.*)\.asp$ /$1.php [L,NC]
</VirtualHost>

<VirtualHost *:80>
    DocumentRoot /home/juan/html
    ServerName juan.unmanarc.com
    CustomLog logs/juan.unmanarc.com-access.log combined
    RewriteEngine on
    RewriteCond %{REQUEST_METHOD} ^(TRACE|TRACK)
    RewriteRule .* - [F]
    RewriteRule ^(.*)\.asp$ /$1.php [L,NC]
</VirtualHost>
```

El primer virtual-host sería el virtualhost por defecto del servidor, el segundo sería un virtualhost de un usuario llamado "juan", para hacer eso, debemos ver en el siguiente punto como configurar el directorio home de juan para que sea visible por httpd.

15. Debemos configurar los permisos de los usuarios para que puedan entrar en su ftp, y que a la vez el servidor web tenga acceso a los archivos, y que otros usuarios no:

```
#cd /home
#chmod 750 juan
#chgrp -R apache juan
#cd juan
#mkdir html
#chown -R juan html
#chgrp -R apache html
#chmod 750 html
```

El usuario juan esta listo para hospedar su página en este servidor, igualmente lo mismo que se hizo con /home/juan/html. se debe hacer con /var/www/html:

```
#cd /var/www
#chmod 750 html
#chgrp -R apache html
```

Eso sería básicamente lo que es una configuración http de un apache medianamente seguro, es importante que ningún usuario deje ver su directorio html a otros debido a que muchas veces el password de la base de datos se guarda en forma plana en los archivos .php dentro de sus directorios.

Existe una opción un poco mas interesante para complementar esto, y es modificar hexadecimalmente el servidor httpd para que en vez que diga: Apache/2.0.49... diga Microsoft-IIS ®.

Aplicar estas medidas de seguridad no bastan, debemos aplicar estrictas medidas de seguridad sobre la configuración php, es decir, el archivo:

/etc/php.ini

A la configuración original que trae redhat, podemos agregar lo siguiente:

1. Activar el safemode:

```
;
; Safe Mode
;
safe_mode = On

; By default, Safe Mode does a UID compare check when
; opening files. If you want to relax this to a GID compare,
; then turn on safe_mode_gid.
safe_mode_gid = Off

; When safe_mode is on, UID/GID checks are bypassed when
```

```
; including files from this directory and its subdirectories.
; (directory must also be in include_path or full path must
; be used when including)
safe_mode_include_dir =

; When safe_mode is on, only executables located in the safe_mode_exec_dir
; will be allowed to be executed via the exec family of functions.
safe_mode_exec_dir =
```

2. Deshabilitar que se muestre la versión de php en los headers del servidor:

```
;
; Misc
;
; Decides whether PHP may expose the fact that it is installed on the server
; (e.g. by adding its signature to the Web server header). It is no security
; threat in any way, but it makes it possible to determine whether you use PHP
; on your server or not.
expose_php = Off
```

3. Habilitar solamente que se muestre los errores graves de php, de modo que si ocurre alguna falla en un script de php no se exponga cierta información que podría ser vital para un hacker:

```
; Examples:
;
; - Show all errors, except for notices
;
;error_reporting = E_ALL & ~E_NOTICE
;
; - Show only errors
;
error_reporting = E_COMPILE_ERROR|E_ERROR|E_CORE_ERROR
;
; - Show all errors
;
;error_reporting = E_ALL
```

4. importante tener esto para evitar exponer información vital de páginas web:

```
display_errors = Off
```

5. Verificar que register_globals este en modo off, normalmente viene configurado en modo off.

6. Otro detalle muy importante es tener apagado esto:

```
allow_url_fopen = Off
```

7. Probablemente el haber activado el safemode haya dañado el squirrelmail, la razón es que solo los archivos con usuario "apache" tienen derecho a ser abiertos por el engine php, así que quizá le recomendamos hacer lo siguiente:

```
#cd /usr/share
#chown -R apache squirrelmail
```

Esta quizá es la primera línea de acción a tomar por un administrador de sistemas para evitar que se de con vulnerabilidades que esten del lado del usuario.

Como vemos, tenemos ya la mitad del trabajo hecho, el servidor web será

entonces medianamente seguro. Vamos a otras partes de configuración, en especial, debemos configurar el servidor vsftpd para que restrinja a los usuarios de forma que no puedan salirse de su directorio \$HOME.

Para ello, nuestro primer paso es hacer una lista de aquellos usuarios que van a tener privilegios mas altos (cuentas shell, acceso a la estructura del disco), y una lista de quienes simplemente tendrán hospedaje web.

Imagine que el usuario unmanarc es un usuario privilegiado, y el usuario juan sea un usuario web, entonces:

El usuario juan debe tener `/sbin/nologin` como shell:

```
#chsh -s /sbin/nologin juan
```

Una vez escrita la configuración y todos los parametros, usted para cargarla simplemente debera ejecutar: `“service httpd restart”`

Servidor Vsftpd:

En `/etc/vsftpd`, se encuentra el archivo `vsftpd.conf`, `vsftpd.conf` debe ser editado y se le deben agregar las siguientes directivas:

1. Restringir el acceso anonimo al servidor:

```
anonymous_enable=NO
```

2. Cambiar el FTP banner.

```
ftpd_banner>Welcome to Yourhost.com FTP service.
```

3. Activar el chroot al \$HOME de los usuarios no privilegiados para evitar que accedan a otros directorios importantes (configuraciones de sistema, etc):

```
chroot_local_user=YES  
chroot_list_enable=YES  
chroot_list_file=/etc/vsftpd.chroot_list
```

Eso asegurara un poquito lo que un usuario pueda hacer por FTP, sin embargo, debemos editar el archivo `/etc/vsftpd.chroot_list` añadiendo ahi (linea a linea) los usuarios que si tengan acceso a salirse de su estructura de su directorio home. Por ejemplo una linea sería:

```
unmanarc
```

Una vez escrita la configuración, usted para cargarla simplemente debera ejecutar: `“service vsftpd restart”`

Servidor sshd:

Es importante configurar bien el servicio sshd, de modo que si usted va a conectarse desde el exterior no lo haga de forma insegura. El archivo a editar esta vez es el: `/etc/ssh/sshd_config`

1. Se debe especificar que solo se debe usar el protocolo 2, el protocolo 1 puede ser inseguro:

```
#Port 22  
#Protocol 2,1  
#ListenAddress 0.0.0.0
```

```
#ListenAddress ::  
Protocol 2
```

El resto de la configuración se deja como la trae Redhat.

Una vez escrita la configuración, usted para cargarla simplemente deberá ejecutar: “service sshd restart”

Servidor MySQL:

El servidor MySQL quizá es uno de los servicios mas propensos a que sea hackeado con facilidad, el servidor mysql viene sin password de root por defecto, eso quiere decir, que cualquier persona con acceso al puerto de MySQL (3306) podría acceder de forma total a toda la base de datos.

Ese problema se soluciona facilmente, y realmente es casi el único problema de seguridad real que trae el MySQL, usted como root solo deberá ejecutar:

```
#mysqladmin -u root password 'nuevo-password'
```

```
#mysqladmin -u root -h localhost password 'nuevo-password'
```

A continuación usted tendrá definido el root password del servidor MySQL impidiendo así la posibilidad de que alguien ajeno a la base de datos la altere u obtenga acceso total.

SynAckFlood:

SynAckFlood es una herramienta personal que yo diseñe/escribí, les voy a explicar un poco como trabaja esta herramienta... Normalmente cuando un hacker quiere revisar un sistema, este corre un sistema de análisis de vulnerabilidades, para ejecutar un sistema de análisis de vulnerabilidades, este primero deberá conocer cuales puertos estan abiertos y cuales cerrados en la computadora a atacar, e incluso debe saber que esta corriendo en cada puerto de su computador. Algunos administradores de sistemas que simplemente les encanta entregar toda información posible acerca de su sistema, porque confían en que lo que pusieron atrás es tan seguro, que no necesitan ocultar nada. Mi opinión es un poco mas paranóica y por ello he desarrollado una herramienta que se encarga de darle información falsa a los scanners.

SynackFlood es una herramienta (OJO: solo sirve para ethernet) que engaña a los scanners con 2 métodos que se complementan, el primer método es responder a cualquier petición de conexión, haciendole creer al scanner que todos los puertos estan abiertos. El segundo método y mas corrosivo, consiste en detectar y emular conexiones a puertos que realmente usted no use, y realmente represente alguien haciendo un “discovery”. En ese caso, SynAckFlood agregará su ip a una lista negra haciendo NAT de todos sus puertos al puerto 1337 donde se estará corriendo un servidor que por cada conexión emite 1Kb de /dev/urandom.

La instalación de synAckFlood es sencilla, lo primero que debe hacer es bajar la versión 2 de la siguiente dirección web:

http://sourceforge.net/project/showfiles.php?group_id=101035&package_id=108635&release_id=216966

Una vez bajado el archivo un TarGz, simplemente haga lo siguiente (como root):

```
tar xzvf synackflood2.tgz
cd synackflood2
./compile
cp synackflood /usr/sbin/
```

En este momento, synAckFlood esta listo para ser ejecutado en el servidor, ahora solo necesitamos conocer cuales puertos estan abiertos (que servicios tenemos), cuales no lo estan, y establecer un patrón de que puertos son considerados hostiles.

Resumiendo un poco, tenemos un servidor web con http y https, ftp, mysql (no queremos acceso externo al mysql), SMTP, pop y sshd. entonces los puertos "seguros" son: 21,22,25,80,110,443

Por otra parte tenemos una serie de puertos catalogados hostiles, por ejemplo, el 139, el 135, el 445, el 111 y el 23.

si queremos ejecutar synAckFlood en un sistema así tan solo debemos ejecutar como root:

```
synackflood -s 21,22,25,80,110,443 -h 23,111,135,139,445 -r -v -k &
```

Si deseamos que se ejecute cada vez que iniciemos linux, agregemos esa linea al final del archivo: /etc/rc.local

Iptables:

iptables es una de las herramientas de firewalling mas interesantes de linux, desde los kernel 2.4, pasando por los 2.6 usan este software para tener un firewall local en el servidor. Siempre es bueno restringir el acceso a los puertos que no estamos usando y solo permitir los servidores. No es muy eficiente a la hora de un backdoor, sin embargo, es lo suficientemente eficiente como primera linea de acción para proteger servicios inseguros/desconfigurados que estemos corriendo en nuestro servidor. Básicamente necesitamos los siguientes puntos:

1. Permitir cualquier tráfico que venga hacia los puertos:
21,22,25,80,110,443,1337
2. Bloquear cualquier tráfico que vaya hacia cualquier otro puerto
3. Deshacernos de ciertos paquetes emitidos por virus/ataques comunes (atacar por ejemplo al 139 con un exploit de RPC de windows a una máquina linux...)
4. Logear lo demas

Para ello hemos diseñado una configuración iptables adaptada a ese modelo, la

configuración de iptables en redhat se encuentra en /etc/sysconfig/iptables:

```
# Completed on Sun Jan 18 19:28:23 2004
# Generated by iptables-save v1.2.8 on Sun Jan 18 19:28:23 2004
*filter
:INPUT DROP [0:0]
:FORWARD DROP [0:0]
:OUTPUT DROP [3:360]
:LOGDROP - [0:0]
-A INPUT -i eth0 -p icmp --icmp-type any -j ACCEPT
-A INPUT -i lo -j ACCEPT
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -i eth0 -p udp -m udp --sport 67:68 --dport 67:68 -j ACCEPT
-A INPUT -i eth0 -p tcp -m state --state NEW -m tcp --dport 22 -j ACCEPT
-A INPUT -i eth0 -p tcp -m state --state NEW -m tcp --dport 80 -j ACCEPT
-A INPUT -i eth0 -p tcp -m state --state NEW -m tcp --dport 21 -j ACCEPT
-A INPUT -i eth0 -p tcp -m state --state NEW -m tcp --dport 110 -j ACCEPT
-A INPUT -i eth0 -p tcp -m state --state NEW -m tcp --dport 25 -j ACCEPT
-A INPUT -i eth0 -p tcp -m state --state NEW -m tcp --dport 443 -j ACCEPT
-A INPUT -i eth0 -p tcp -m state --state NEW -m tcp --dport 1337 -j ACCEPT
-A INPUT -p udp -m udp -j REJECT
-A INPUT -p tcp -m state --state NEW -m tcp --dport 113 -j DROP
-A INPUT -p tcp -m state --state NEW -m tcp --dport 139 -j DROP
-A INPUT -p tcp -m state --state NEW -m tcp --dport 445 -j DROP
-A INPUT -p tcp -m state --state NEW -m tcp --dport 1080 -j DROP
-A INPUT -p tcp -m state --state NEW -m tcp --dport 8080 -j DROP
-A INPUT -p tcp -m state --state NEW -m tcp --dport 4662 -j DROP
-A INPUT -p tcp -m state --state NEW -m tcp --dport 3128 -j DROP
-A INPUT -p tcp -m state --state NEW -m tcp --dport 5490 -j DROP
-A INPUT -p tcp -m state --state NEW -m tcp --dport 6588 -j DROP
-A INPUT -p tcp -m state --state NEW -m tcp --dport 554 -j DROP
-A INPUT -p tcp -m state --state NEW -m tcp --dport 901 -j DROP
-A INPUT -p tcp -m state --state NEW -m tcp --dport 17300 -j DROP
-A INPUT -j LOGDROP
-A OUTPUT -o lo -j ACCEPT
-A OUTPUT -m state --state NEW,RELATED,ESTABLISHED -j ACCEPT
-A LOGDROP -j LOG --log-tcp-options
-A LOGDROP -j DROP
COMMIT
# Completed on Sun Jan 18 19:28:23 2004
```

Una vez escrita la configuración, usted para cargarla simplemente debera ejecutar: “iptables-restore < /etc/sysconfig/iptables”

yum:

yum es una herramienta de actualización de software via consola, sin embargo, Fedora Core 1, trae un servicio “yum” que sirve para automáticamente bajar e instalar los upgrades del sistema, virtualmente estarías creando un sistema que no habría que mantener, el mismo se bajaría los updates.

Por defecto no viene instalado como servicio, sin embargo, para activarlo como servicio es sumamente sencillo, en una consola de administrador (root), usted deberá ejecutar el programa “setup”, luego vaya a “Servicios del sistema”, y luego vaya casi hasta el final y seleccione la opción “yum”, luego de eso, presione tab, y luego presione enter para seleccionar “ok”. salga de esa pantalla con el boton salir, y estará ya listo para que al iniciar la computadora automáticamente se baje los updates de linux.

Eso nos ahorra mucho tiempo de administración, y nos limita a leer logs diariamente sin tener que buscar vulnerabilidades de software.

Debemos destacar que un servidor medianamente seguro no implica que un website no pueda ser atacado, el webmaster si debe tener otras políticas de seguridad aparte de las que nosotros tengamos que configurar como "root".