

Prácticas Básicas de Seguridad en Linux

El siguiente manual constituye una guía de mandatos básicos para instalar una estación de trabajo segura en Linux Debian. Las opciones pueden cambiar de acuerdo a las funciones para las que se prepare la estación así como por la distribución de linux escogida.

Por favor tenga en cuenta que estas son opciones básicas.

Antes de la Instalación

- Identificar los servicios que se quiere proveer, a quien se les debe proveer, y de que manera.
- Escoger las herramientas que servirá de plataforma para ofrecer el servicio, teniendo en cuenta como parámetros primordiales la estabilidad, mecanismos de seguridad y eficiencia, por ejemplo se prefiere apache2 sobre apache, porque tiene resuelto muchos bugs de seguridad que fueron encontrados en apache, además ofrece incluido un sistema para crear hosts virtuales seguros (ssl).
- Identificar las contraseñas que se vayan a usar, clasificarlas por su importancia y frecuencia de uso, identificar responsables de la mismas, para luego aplicar a cada una de ellas la mayor cantidad de cualidades de seguridad:
 - La longitud de la contraseña debe ser mayor o igual a 8 caracteres.
 - Contener gran variedad de caracteres alfanuméricos y especiales (A-Z, a-z, 0-9, “!@#%&*()_+=-?”).
 - No contener información relacionada con el trabajo o la vida personal del administrador.
 - Cambios periódicos para la contraseña y evitar repetirlas (3 meses en promedio).
- Aplicar algunas características de seguridad física sobre servidores, por ejemplo:
 - Restringir el acceso a personas que no estén autorizadas para manejar el servidor.
 - Deshabilitar el arranque a otros medios que no sea el disco duro del sistema operativo.
 - Colocar contraseña para acceder al sistema de Configuración del BIOS. (Tenga cuidado de no colocarla al sistema de arranque).
 - Activar el auto-encendido ante problemas de electricidad en la configuración del BIOS.

Durante la instalación

- Desconectarla de la red: Los Sistema Linux en su instalación dan la facilidad de conectarse a la red para descargar los paquetes que va a instalar desde un repositorio. Pero durante este tiempo el sistema operativo esta desprotegido ante usuarios maliciosos en la red, por lo tanto se recomienda hacer este proceso de manera aislada de la red.
- Escoger el sistema de particiones y sistema de archivos: En la mayor parte de los casos la información que se almacena en un servidor es tan importante que no se puede dar el lujo de perderla ni por problemas técnicos ni por ataques desde la red.
 - Primero escogeremos el sistema de archivo para lo cual debemos tomar en cuenta la estabilidad del mismo ya que nuestro objetivo es evitar perder la información, por esta razón ext3 es la ganadora.
 - Luego diseñaremos un sistema de particiones que permita contener la información separada, evitando que se desborde la información y con el propósito que los ataques de negación de servicios no saturen la particiones más importantes.
 - El primer requerimiento se satisface separando las carpetas "/", de "/var", "/usr".
 - Muchas herramientas se instalan como paquete no perteneciente al sistema operativo en "/usr/local/", y por eso debemos crear una partición para esta carpeta, con el fin de no perder la funcionalidad de estos programas si se reinstala el sistema operativo.
 - Muchos ataques de negación de servicios incrementan muy rápidamente los datos temporales y de usuario. Por lo tanto debemos crear particiones para "/tmp" y "/home", para el caso cuando existe dicho ataque saturen estas particiones y no la principal "/", que podría parar el funcionamiento del sistema operativo por no haber espacio disponible para escribir.
 - Y por último, crear una partición de respaldo periódicos, en caso de alguna partición pase de un porcentaje peligroso y pueda saturarse con el nombre de su preferencia, por ejemplo "/data". A continuación tenemos como ejemplo este sistemas de particiones para un disco de 40 GB:

/	ext3	5 GB
/var	ext3	10 GB
/data	ext3	10 GB
/usr	ext3	5 GB
/home	ext3	2 GB
/tmp	ext3	2 GB
/usr/local	ext3	5 GB

swap swap 1 GB

- Se observa que las particiones “/var” y “/data” son las más grandes porque en ellas se almacenan la información y las bitácoras del sistema y muchos programas.

Después de la instalación

- Colocar una contraseña a Lilo o Grub: los gestores de arranque de linux permiten no solo entrar al sistema operativo deseado, si no también habilitan una consola para montar particiones con distintos modo de acceso que podrían ofrecer la información a personas no permitidas.
- Para añadir la contraseña al lilo simplemente modifique y/o agregue las siguientes líneas al archivo de configuración de lilo /etc/lilo.conf

```
image=/boot/2.6.16-vmlinuz
label=Linux
read-only
password=contraseña
restricted
```

Para que surta efecto el cambio, debe ejecutar en la consola como usuario root:

```
# lilo
```

- Para colocar la contraseña al gestor de arranque Grub, se generará un hash a partir de una palabra clave:

```
# grub-md5-crypt
# Password: *****
# Retype Password: *****
# $1$q19Yf1$fV0t8Dr1rjMtxNytKUXy5/
```

- Luego edite el archivo de configuración de grub /boot/grub/menu.lst y luego del segmento “timeout” anexe la línea

```
password --md5 $1$q19Yf1$fV0t8Dr1rjMtxNytKUXy5/
```

- Por último, efectuamos los cambios con el comando siendo hda disco duro donde se instalará el grub.

```
# grub-install /dev/hda
```

- Detener y desactivar los servicios innecesarios y complementos: Linux trae por defecto servicios activos por omisión que en la mayoría

de los casos no se usan al menos que se tengan conocimientos de ellos y sea un objetivo para el servidor. Para que estos servicios no dejen puertos abiertos tentadores para los atacantes, debemos pararlos.

```
# /etc/init.d/exim4 stop
# /etc/init.d/portmap stop
# /etc/init.d/inetd stop
# /etc/init.d/fam stop
# /etc/init.d/alsa stop
# /etc/init.d/lpd stop
# /etc/init.d/nfs-common stop
```

Aplicar lo que vimos anteriormente mata los procesos y cierra los puertos asociados a él, pero no impide que al reiniciar la computadora, los servicios no se levanten de nuevo. Por esta razón debemos quitar estos scripts del arranque de demonios del sistema operativo "rc" haciendo

```
# update-rc.d -f exim4 remove
# update-rc.d -f portmap remove
# update-rc.d -f inetd remove
# update-rc.d -f fam remove
# update-rc.d -f alsa remove
# update-rc.d -f lpd remove
# update-rc.d -f nfs-common remove
```

Este proceso tiene también otro beneficio, liberar al procesador de aplicaciones que no se usan, y por lo tanto hacer mas eficiente y rápido el trabajo del sistema operativo y las demás aplicaciones.

Por ultimo, se deben desinstalar los programas de los servicios innecesarios, de la siguiente manera:

```
# aptitude purge exim4 alsa fam .....
```

No olvidar actualizar el sistema de repositorios que maneja el sistema en el archivos correspondiente para la distribución (para debian /etc/apt/source.list) y agregar repositorios de seguridad para la distribución además de los oficiales y más cercanos:

```
deb http://security.debian.org stable/updates main contrib non-free
deb http://security.debian.org/debian-security stable/updates main
contrib non-free
```

Y actualizamos los repositorios y el sistema

```
# aptitude update
# aptitude upgrade
```

- Instalar un paquete para el envío de correo a un servidor en caso de eventos extraños:

```
# aptitude install ssmtp
```

- Y editamos el archivo de configuración /etc/ssmtp/ssmtp.conf, cambiamos las líneas

```
root=ums@ula.ve  
mailhub=smtp-mail.ula.ve  
hostname=servidor.ing.ula.ve
```

- FIREWALL (IPTABLES): esta es una herramienta para el filtrado de paquetes en capa 2, capa 3, capa 4. El buen uso de esta herramienta incrementan significativamente la seguridad en el sistema operativo y nos permite hacer las políticas de acceso para restringir la conectividad a solo hosts conocidos sobre servicios requeridos.

Iptables maneja los paquetes que entran y salen de una computadora, aceptándolos, eliminándolos, o dejando un registro del paquete. La herramienta maneja tres tipos de políticas, INPUT (paquetes que llegan a la computadora), OUTPUT (paquetes que salen de la computadora), FORWARD (paquetes que entran pero que su destino es otra computadora). Estas políticas contiene ordenadamente reglas, que permiten filtrar los paquetes según sus características. Para las políticas y las reglas existen acciones, que son ACCEPT (aceptar el paquete), DROP (denegar el paquete) y REJECT (denegar y responder con otro paquete avisando de lo denegó). Cuando un paquete va a pasar por el firewall, primero se identifica la política a quien pertenece, luego busca en las reglas la primera que corresponda todos los parámetros del paquete, si la encuentra aplica la acción, pero si no la encuentra, aplica la acción de la política.

A continuación se mostrará una plantilla, para modificar políticas y reglas

Políticas:

```
# iptables -P <política> <acción>
```

Reglas:

```
# iptables {-A | -I } <política> [{-i | -o} <interfaz>] [-s <origen>]  
[-d <destino>] [-p <protocolo>] [--sport <puerto>] [--dport  
<puerto>] -j <acción>
```

{-A | -I} <política>: -A para añadir la regla al final o -I al principio (la opción es obligatoria).

[{-i | -o} <interfaz>]: -i para interfaz entrante o -o para saliente (opcional).

[-s <origen>]: hosts o red origen (opcional).

[-d <destino>]: hosts o red destino (opcional).

[-p <protocolo>]: tcp o udp (opcional, por defecto tcp).

[--sport <puerto>]: puerto de origen (opcional).

[--dport <puerto>]: puerto de destino (opcional).

-j <acción>: acción a aplicar.

Para ilustrar de una mejor manera como se hacen las reglas y se definen las políticas con iptables, mostraremos un ejemplo que servirá como script para añadirlas y eliminarlas.

Existen dos maneras o filosofías para crear un cortafuego (firewall), la primera tiene como base suponer que lo inseguro es conocido, y la segunda supone lo contrario, que no se conoce lo inseguro. Debido a tantos eventos que hace vulnerable al sistema operativo desde la red que aveces no da tiempo de enterarse de todas ellas, por lo tanto preferimos escoger la segunda manera de restringir los paquetes de la red, definir las políticas negando todo, para luego agregar las reglas que acepten lo que nosotros sabemos que es seguro.

El script esta hecho en sh, y se colocara en el directorio /etc/init.d/ con el nombre "iptables".

```
#!/bin/sh
```

```
#####Cortafuego para la seguridad de una máquina linux como servidor#####
```

```
# almacenar en una variable la ip local
```

```
EIP=`ifconfig eth0 | grep "inet addr:" | cut -d ":" -f2 |cut -d " " -f1`
```

```
case "$1" in  
start)
```

```
    echo -n " * Starting Iptables Rules ..... "
```

```
    # Limpiar todas las reglas de iptables
```

```
    iptables -X  
    iptables -F  
    iptables -Z
```

```
# Definir las políticas por omisión
```

```
iptables -P INPUT DROP  
iptables -P OUTPUT DROP  
iptables -P FORWARD DROP
```

```
# reglas para recibir y enviar paquetes de/a localhost
```

```
iptables -A INPUT -i lo -s localhost -j ACCEPT  
iptables -A OUTPUT -o lo -d localhost -j ACCEPT
```

```
# reglas para recibir y enviar paquetes de/a la misma ip eth0
```

```
if [ ! -z $EIP ]; then  
    iptables -A INPUT -s $EIP -j ACCEPT  
    iptables -A OUTPUT -d $EIP -j ACCEPT  
fi
```

```
# Crear reglas para el uso de servicios en la red
```

```
# reglas para resolver con dns
```

```
iptables -A INPUT -p udp --sport 53 -j ACCEPT  
iptables -A OUTPUT -p udp --dport 53 -j ACCEPT
```

```
# reglas para enviar paquetes a servidor de correos
```

```
iptables -A INPUT -p tcp --sport 25 -j ACCEPT  
iptables -A OUTPUT -p tcp --dport 25 -j ACCEPT
```

```
# reglas para conectarse a servidores ssh
```

```
iptables -A INPUT -p tcp --sport 22 -j ACCEPT  
iptables -A OUTPUT -p tcp --dport 22 -j ACCEPT
```

```
# reglas para recibir y enviar archivos a servidores ftp
```

```
iptables -A INPUT -p tcp --sport 20:21 -j ACCEPT  
iptables -A OUTPUT -p tcp --dport 20:21 -j ACCEPT
```

```
# reglas para permitir hacer ping a otros host
```

```
iptables -A OUTPUT -p icmp --icmp-type echo-request -j ACCEPT  
iptables -A INPUT -p icmp --icmp-type echo-reply -j ACCEPT
```

```
# Reglas para permitir pasar a los paquetes que entran y salen de un  
servicio de localhost
```

```
# reglas para permitir acceso al servidor local del servicio web
```

```
iptables -A INPUT -p tcp --dport 443 -j ACCEPT
iptables -A OUTPUT -p tcp --sport 443 -j ACCEPT
```

```
# reglas para permitir acceso al servidor local del servicio ssh
```

```
iptables -A INPUT -p tcp --dport 22 -j ACCEPT
iptables -A OUTPUT -p tcp --sport 22 -j ACCEPT
```

```
# reglas para permitir acceso al servidor local del servicio ftp
```

```
iptables -A INPUT -s 150.185.181.0/24 -p tcp --dport 21 -j ACCEPT
iptables -A OUTPUT -d 150.185.181.0/24 -p tcp --sport 21 -j ACCEPT
```

```
echo "done."
```

```
::
```

```
stop)
```

```
echo -n " * Stopping Iptables Rules ..... "
```

```
##### script para detener y limpiar todas las reglas de iptables
```

```
# acepta todo lo que venga, se envía a otra y salga de la maquina
```

```
iptables -P INPUT ACCEPT
iptables -P OUTPUT ACCEPT
iptables -P FORWARD ACCEPT
```

```
# limpia todas las reglas de filter y nat
```

```
iptables -F
iptables -X
iptables -Z
```

```
echo "done."
```

```
::
```

```
restart)
```

```
echo -n " * Restarting Iptables Rules ..... "
$0 stop > /dev/null
$0 start > /dev/null
echo "done."
```

```
::
```

```
*)
```

```
echo "Use: $0 {start|stop|restart}"
```

```
::
```

esac

exit 0

- Usando Tcprappers: Las envolturas de TCP se desarrollaron cuando no había ningún filtro del paquete real disponible y el control de acceso fue necesitado. Las envolturas de TCP facilitan permitir o negar un servicio para un organizador o un dominio y defina un valor permitido o niegue la regla.

El uso de esta herramienta existen dos archivo a modificar, /etc/hosts.allow y /etc/hosts.deny, para permitir y denegar el uso de algunos servicios para algunas ip's permitidas.

Agregar lineas en el archivo /etc/hosts.allow

```
# localhost
sshd:127.0.0.1

# Administrador remoto
sshd:150.185.181.50
```

Agregar las lineas en el archivo /etc/hosts.deny

```
sshd: ALL: SPAWN ( \
    echo -e "\n\
    TCP Wrappers\: Conexion rechazada en $(uname -n)\n\

    Proceso\: %d (pid %p)\n\
    Usuario\: %u\n\
    Host\: %c\n\
    Fecha\: $(date)\n\
    " | /usr/bin/mail -s "Conexion de %c bloqueada"
hmarquez@ula.ve) &
```

Las modificaciones que se han hecho a los archivos, significa que solo permitirá acceso desde la misma máquina y desde la ip 150.185.181.50, y en caso contrario de que se intenten conectar al servicio ssh desde otra ip, no lo permitirá y además enviará un correo a hmarquez@ula.ve, indicando cual ip intentó conectarse.

- Asegurando ssh: el uso de una computadora remotamente para muchas organizaciones y personas es muy útil, y para asegurar el manejo y control de los comandos ejecutados e información confidencial transferida, recomendamos usar el protocolo ssh sobre telnet, debido a que toda la información enviada desde el cliente al

servidor es cifrada, a diferencia de telnet (se sugiere desinstalar telnet y telnetd). Para empezar a manejar ssh y configurarlo, se instala desde un repositorio:

```
# aptitude purge telnet telnetd
# aptitude install ssh
```

Se colocan todos los valores por omisión en el momento de la instalación y luego procedemos a configurar el servicio ssh. Editamos el archivo `/etc/ssh/sshd_config` y agregamos algunas opciones a nuestra conveniencia.

- ListenAddress 192.168.0.1: Haga que ssh escuche solo la interfaz dada, sólo en un caso de que haya más de uno (y no necesite un ssh disponible sobre éste) o que en un futuro agregue una nueva tarjeta de red (y no necesite una conexión desde ssh en ésta).
 - PermitRootLogin No: Intente no permitir al Root entrar tanto como sea posible. Si alguien quiere volverse root por vía ssh, dos logins serán necesarios y la contraseña root no puede ser obtenida a fuerza bruta por vía SSH.
 - Port 666: Cambie el puerto de escucha de tal manera que el intruso no pueda estar completamente seguro de si está corriendo un demonio de sshd. (Note que esto es seguridad por oscuridad).
 - PermitEmptyPasswords no: Las contraseñas en blanco convierten en broma la seguridad del sistema.
 - AllowUsers hmarquez alex: Permita que solamente ciertos usuarios tengan acceso vía ssh a esta máquina.
 - AllowGroups wheel admin: Permita que solamente los miembros de ciertos grupos tengan acceso vía a ssh a esta máquina. AllowGroups y AllowUsers tienen directivas equivalentes para denegar el acceso a una máquina. Predeciblemente se llaman "DenyUsers" y "DenyGroups".
- Apache2 y ssl: el protocolo http usado para permitir el servicio web en la internet envía la información completamente plano, es decir, la información enviada viaja por la red exactamente igual a lo que se quiere transferir. Por esta razón, debemos asegurar que esta información viaje cifrada para mayor confidencialidad para el usuario y para las aplicaciones que corren sobre el servidor.

Siguiendo las instrucciones podremos crear un servidor web y activando el modulo ssl de cifrado para que solo escuche por el puerto http seguro (https) 443. Además aplicaremos un sistemas de restricciones con usuario y contraseña para acceder al sitio virtual.

```
# aptitude install apache2
# a2enmod ssl
```

```
# echo "443" > /etc/apache2/ports.conf
# apache2-ssl-certificate -days 9999
# a2ensite default
```

Editamos el archivo de configuración de apache para el sitio default
/etc/apache2/sites-enabled/000-default

```
NameVirtualHost *:443
<VirtualHost *:443>
```

dentro del segmento <Directory /var/www/> y <Directory
"/usr/lib/cgi-bin"> añadiremos las líneas para el sistema de
autenticación de los usuarios.

```
AuthType Basic
AuthName "Acceso Restringido"
AuthUserFile /etc/apache2/.users.dat
Require valid-user
```

comentamos el redireccionamiento hacia la página principal

```
#RedirectMatch ^/$ /apache2-default/
```

y luego activamos el motor de cifrado con el certificado
anteriormente creado

```
ServerSignature Off
SSLEngine On
SSLCertificateFile /etc/apache2/ssl/apache.pem
```

Luego de esto, debemos crear el archivo que contendrá los
usuarios y contraseña para el acceso hacia el sitio virtual.

```
# touch /etc/apache2/.usert.dat
# htpasswd /etc/apache2/.usert.dat usuario
```

reiniciamos y recargamos los modulos del servicio de apache2

```
# /etc/init.d/apache2 force-reload
```

- Aplicación WEB: Finalizando instalaremos una base de datos y generaremos una aplicación web en php que solo muestre si se conectó a la base de datos o un mensaje de error, para observar el resultado del cortafuego instalado en el servidor.

Para esto requerimos de instalar el servidor de bases de datos mysql, algunos módulos para apache2 que ejecuten código php, librerías que conecten la aplicación con la base de datos, y editar algunos archivo para que todo esto funcione.

```
# aptitude install mysql-server libapache2-mod-php4, php4-
```

mysql

En este momento se ha instalado el servidor web, pero no tiene ningún indicio de seguridad en mysql. Primero debemos cambiar la contraseña de root de mysql, con el siguiente comando:

```
# mysqladmin -u root password "contrasena"
```

Luego de eso, ingresamos al sistema de control de mysql

```
# mysql -p  
Password:
```

luego crearemos una base de datos prueba

```
mysql >> create database prueba;
```

le asignamos un usuario porque root nunca debe ser usado como usuario de aplicacion.

```
mysql >> grant insert, delete, update, select on prueba.* to  
usuario@localhost identified by 'contrasena_usuario';
```

reflejamos los cambios

```
mysql >> flush privileges;
```

Para hacer que la aplicación se conecte a la base de datos, debemos editar el siguiente archivo

```
# vim /etc/php4/apache2/php.ini
```

descomentamos la linea que habilita el uso de mysql para script de php.

```
extension=mysql.so
```

y del archivo /etc/apache2/apache2.conf

```
AddType application/x-httpd-php .php  
AddType application/x-httpd-php-source .phps
```

Luego incluimos el archivo index.php que se mostrará a continuación en la carpeta "/var/www/":

```
<?php  
$link = mysql_connect('localhost:3306', 'prueba',  
'123456','prueba');  
if (!$link) {  
    die('Could not connect: ' . mysql_error());
```

```
}  
phpinfo();  
mysql_close($link);  
?>
```

Reiniciamos el servidor web

```
# /etc/init.d/apache2 restart
```

Y desde un navegador probar la aplicación montada en el servidor web colocando la siguiente dirección <http://150.185.181.50/index.php>