Uso de Instaladores de Paquetes

Generalmente se suele usar **AutoRPM** (Red Hat, Mandrake, etc) y

apt-get (Debian) para instalar o actualizar paquetes de software en un sistema. Debes usarlos con "cuidado" y no utilizarlos para actualizar automáticamente Servidores en producción. La recomendación es que bajes los paquetes y verifiques la firma antes de instalarlos.

Nmap

Instalá **nmap** para determinar potenciales canales de comunicación o conexiones "no autorizadas". **Nmap** es un potente port-scanner que puede determinar remotamente el SO de un equipo (mediante finger-print), hacer *"stealth"* scans (escaneos heurísticos) manipulando paquetes **ICMP**, **TCP** y **UDP**, y puede llegar a determinar el nombre del servicio y versión que esta corriendo del mismo.

Password Lilo

En servidores instalados en ambientes *"no seguros"* o con acceso de personal no especializado protege con password a **LILO,** para requerir autentificación cuando se pasen parámetros al Kernel en el momento de Boot. Agrega un password en el **/etc/lilo.conf** de la siguiente manera:

image = /boot/vmlinuz-2.6.10
label = Linux
read-only
restricted
password = pone-aca-tu-password

ahora ejecuta **lilo** para fijar los cambios.

Openwall kernel patch

Instala el OpenWall kernel patch que agrega al Kernel opciones de seguridad que permiten, entre otras cosas, prevenir **buffer overflows**, restingir información del **/proc** que en general está disponible para cualquier usuario, etc.

NOTA: Para su instalación se requiere re-compilar el Kernel

Fecha y Hora

Siempre aseguraré que la fecha y hora de los equipos estén sincronizadas. Esto es importante ya que en caso de un ataque es mas fácil trazar una "línea de tiempo y eventos" que pueden llegar a determinar el origen de los mismos mediante el análisis de los archivos de **log** (En seguridad a esto se lo llama **Análisis Forense**)

Esto lo puedes hacer agregando una línea como ésta al **crontab**:

0-59/30 * * * * root /usr/sbin/ntpdate -su servidor_en_hora

Bastille Linux Hardening Tool

En lo posible instala y ejecuta una herramienta como Bastille Linux Hardening Tool. **Bastille** está compuesto por una serie de *shell scripts* que eliminan muchas de las vulnerabilidades producto de las instalaciones Linux por default.

Sudo

Configura **sudo** (superuser do) para ejecutar comandos privilegiados como un usuario normal en lugar de **su**. De esta manera el usuario usará su propio password para ejecutar comando específicos que de otra manera requerirían acceso root.
En el archivo **/etc/sudoers** se determina que usuarios y cuáles programas pueden ejecutar. Por ejemplo: usando el programa **/usr/sbin/visudo** (editor **vi** para configurar **/etc/sudoers**) podemos permitir a pepito usar la impresora instalada en magneto:

Cmnd_Alias LPCMDS = /usr/sbin/lpc, /usr/bin/lprm
pepito magneto = LPCMDS

Pepito debe ejecutar **sudo** con el/los comando/s autorizados con su propio password:

pepito$ sudo /usr/sbin/lpc
Password: <contraseña_de_pepito>
lpc>

Contraseñas Seguras

Debes elegir contraseñas con cierta dificultad y con combinaciones alfanuméricas y acostumbra a tus usuarios para que hagan lo mismo. Obtener acceso a una cuenta de usuario es relativamente fácil (suelen dejar sus contraseñas pegadas en el monitor, debajo del pad del mouse, etc). Conseguir acceso como root está solo a un paso!. Ejecuta programas como John The Ripper regularmente en aquellos sistemas donde la seguridad sea crítica y deshabilita las cuentas no utilizadas usando **/usr/bin/passwd -l**. En lo posible siempre usa MD5 para tus passwords

Iptables

El filtrado de paquetes no es solo para los *firewalls*. Usando **IPTABLES** puedes aumentar la protección de un equipo significativamente de ataques externos. Puedes bloquear el acceso a uno o varios servicios desde fuera de la red (Internet) o dentro de la misma LAN. Lee el HOWTO de **IPTABLES** para implementar un filtrado efectivo.

Archivo .bashrc

Crear comandos alias para utilizar en la consola Para crear estos alias editaremos el archivo **.bashrc** que esta dentro de nuestro home con procesador de texto preferido, aquí dejo un ejemplo:

```
# .bashrc
# User specific aliases and functions

alias ll='ls -lh'
alias rm='rm -i'
alias cp='cp -i'
alias mv='mv -i'
```

En la primera línea descomentada aparece el comando **ls -lh** y su alias será **ll**, entonces cuando estemos en la consola o terminan podremos ocupar **ll** que será el equivalente a escribir **ls -lh**. Las demás líneas idem.

```
[user@yoda ~]$ ll
total 13M
-rw-r--r--   1 user user  91K jun 12 14:29 access
-rw-r--r--   1 user user  34K jun 12 14:29 access_LPT.txt
-rw-r--r--   1 user user 1,4K mar 22 20:22 anaconda-ks.cfg
-rw-r--r--   1 user user  430 abr 18 00:30 busca_hosts.sh
drwxr-xr-x   2 user user 4,0K mar 22 20:48 firewall
-rw-r--r--   1 user user    0 mar 24 13:40 help.txt
-rw-r--r--   1 user user  46K mar 22 20:22 install.log
-rw-r--r--   1 user user 3,3K mar 22 20:22 install.log.syslog
```

Creando vínculos o links virtuales

```
$ ln -s /home/user/.vimrc /home/user/vim.conf
```

Luego verificamos el enlace virtual

```
$ ls -l

lrwxrwxrwx   1 user user        6 ago 18 21:57 vim.conf -> .vimrc
```

Eso quiere decir que creamos un vínculo llamado **vim.conf** que llama a **.vimrc**, ahora si editamos el archivo **vim.conf** veremos y editaremos el contenido de **.vimrc**

Mandar Mail desde Consola

Primero que todo el equipo debe estar configurado para poder enviar mail o tener un smtp que envíe el correo por el.

Usando el comando **mail** podemos enviar mail de la siguiente forma:

```
[usuario@mail ~]$ mail -v usuario@dondesea.com -s "test de mail"
hola
este es el mensaje del mail
.
Cc:
usuario@dondesea.com... Connecting to [127.0.0.1] via relay...
220 mail.tuhostmail.com ESMTP Sendmail 8.13.1/8.13.1; Mon, 5 Sep 2005
20:28:26 -0400
>>> EHLO mail.tuhostmail.com
250-mail.tuhostmail.com Hello localhost.localdomain [127.0.0.1], pleased to meet
you
250-ENHANCEDSTATUSCODES
250-PIPELINING
250-8BITMIME
250-SIZE
250-DSN
250-AUTH DIGEST-MD5 CRAM-MD5
250-DELIVERBY
250 HELP
>>> MAIL From:<usuario@mail.tuhostmail.com> SIZE=79
AUTH=usuario@mail.tuhostmail.com
250 2.1.0 <usuario@mail.tuhostmail.com>... Sender ok
>>> RCPT To:<usuario@dondesea.com>
>>> DATA
250 2.1.5 <usuario@dondesea.com>... Recipient ok
354 Enter mail, end with "." on a line by itself
>>> .
250 2.0.0 j860SQmB003258 Message accepted for delivery
usuario@dondesea.com... Sent (j860SQmB003258 Message accepted for delivery)
Closing connection to [127.0.0.1]
>>> QUIT
221 2.0.0 mail.tuhostmail.com closing connection
```

Ahí nos preguntara otros detalles como CC y BCC. Para terminar el mensaje poner un **.** en la ultima línea, con eso enviará el correo al destinatario.

Ocupación en disco de los usuarios

Utilizamos el comando **du** y una ordenación con **sort**

```
[root@localhost ~]# du --kilobytes -s /home/* | sort -rn
32072104        /home/pedro
1548756         /home/luis
143632          /home/jorge
```

Notar que el comando debe ser lanzado por un usuario con privilegios.

Ver actividad de los logs

Estas son algunas herramientas para la monitorización de algunos servicios (los que registran logs), especiales para los administradores de servidores linux.

Logs de mailserver (sendmail)

```
# tail -f /var/log/maillog
```

Otros parámetros útiles son **-n noLineas** donde **noLineas** es el número de líneas que queremos desplegar en pantalla. Ej:

```
# tail -f -n 30 /var/log/maillog
```

Otra buena ayuda es aplicar filtros para obtener lo que necesitemos ver y eliminar logs innecesarios, como en este ejemplo, mostraremos los logs de **maillog** excepto los login al servicio pop:

```
# tail -f /var/log/maillog | grep -v pop
```

Logs de seguridad

Seguridad del servidor

```
# tail -f /var/log/secure
```

Mensajes de los servicios

```
# tail -f /var/log/messages
```

Logs de apache (httpd)

Ver la actividad del servidor web apache desde el archivo **access.log**

```
# tail -f /var/log/httpd/access.log
```

Verificar que no hagan uso malicioso de apache y se aprovechen de algunas páginas php mal escritas (file injection)

```
# tail -f /var/log/httpd/access.log |grep -i wget
```

Editando el Crontab

NOTA: Las tareas programadas quedarán registradas al usuario que edite el crontab, es decir, si lo haces como un usuario normal, debes tener la precaución de tener los permisos adecuados para ejecutar los comandos que utilices.

Para editar, borrar o agregar una tarea programada ponemos en la consola de comandos:

```
$ crontab -e
```

Y se abrirá el editor de texto por defecto en nuestro entorno. Ahí debemos poner lo que queramos dejar programado con el siguiente formato:

```
*    *    *    *    *     /ruta/comando/ejecutar
-    -    -    -    -
|    |    |    |    |
|    |    |    |    +----- día de la semana (0 - 6) (dom = 0)
|    |    |    +---------- mes (1 - 12)
|    |    +--------------- día del mes (1 - 31)
|    +-------------------- hora (0 - 23)
+------------------------- minuto (0 - 59)
```

Todas estas actividades se detallan enviando un mail para confirmar su ejecución. Ahora bien si tenemos un proceso que registra un log muy extenso y que además no queremos que nos notifique con un mail, agregamos al final de la línea **> /dev/null 2>&1**, por ejemplo:

```
*    *    *    *    *     /ruta/comando/ejecutar > /dev/null 2>&1
```

IPtables

- Capturar todas las peticiones que se hagan a la IP_1 y reenviarlas a IP_2

```
iptables -t nat -A PREROUTING -d IP_1 -j DNAT --to-destination IP_2
```

- Capturar todas las peticiones que se hagan al puerto X de la IP_1 y reenviarlas al puerto Y de la IP_2

```
iptables -t nat -A PREROUTING -p tcp -d IP_1 --dport X -j DNAT --to-destination IP_2:Y
```

- Bloquear totalmente los paquetes que vengan desde IP

```
iptables -A INPUT -s IP -j [REJECT ] [DROP]
```

- Bloquear totalmente los paquetes que tengan como destino IP

```
iptables -A INPUT -d IP -j [REJECT ] [DROP]
```

- Bloquear acceso a WinMX

```
iptables -A FORWARD -d 64.124.41.0/24 -j [REJECT] [DROP]
```

- Bloquear acceso a eDonkey

```
iptables -A FORWARD -p tcp --dport 4661:4662 -j [REJECT] [DROP]
iptables -A FORWARD -p udp --dport 4665 -j [REJECT] [DROP]
```

Notas:

Con la opción REJECT la persona que envíe el paquete recibirá una respuesta del tipo "icmp-port-unreachable".

Con la opción DROP la persona que envíe el paquete nunca va a recibir una respuesta ya que son eliminados directamente.

Para eliminar una regla basta con reemplazar la opción "-A" por "-D"

Comando: Screen(1)

Screen es un administrador de ventanas que permite multiplexar un terminal físico en varios procesos (por ejemplo, varias shells interactivas).

Es decir, con este comando puedes tener varias shells (o procesos) corriendo en una misma ventana física.

Se ejecuta:

```
$ screen
```

Algunas combinaciones de teclas útiles (obs: C-a = Ctrl+a):

- C-a c: crea una nueva ventana con una shell e ingresa a esa ventana.
- C-a n: cambia a la ventana siguiente.
- C-a p: cambia a la ventana anterior.
- C-a ls: muestra las sessiones screen que quedaron abiertas en estado "detach".
- C-a d: Deja la session de screen en estado detach.
- C-a x: Bloquea un terminal.

Verificación de Aceleración Gráfica

- fglrxconfig

Comandos Mapeo de funciones

xev: Permite probar el mapeo de botones y teclas.

xmodmap -e "pointer = 1 2 3 8 9 4 5 6 7 10 11" : Modifica el mapeo de botones del mouse al referido.

Tomado de: http://linux.lcampino.cl/wiki/index.php/Tips