Prácticas

1. Control de los archivos

Objetivo: controlar los archivos y derechos asociados.

1. Se puede establecer para cada archivo una suma de control o checksum. El comando **sha256sum** calcula la suma de control de un archivo en formato sha256. Calcule el checksum del archivo /etc/passwd:

```
# sha256sum /etc/passwd
675a4789af0c7b4c03ac6f487b95e1019230cd500d5bc00f470aed612492fbd1 /etc/passwd
```

2. Modifique una información, aunque sea sencilla, por ejemplo un comentario, en /etc/passwd, y vuelva a calcular el sha256. Aquí se ha sustituido una «C» por una «c»:

```
$ md5sum /etc/passwd
a7450bcc205cbee80c429d9522c43f53 /etc/passwd
```

   La suma es totalmente diferente, lo que muestra que se ha modificado el archivo. Si conserva en un archivo MD5SUM las sumas de control de origen, podrá detectar las modificaciones de manera más fiable que con la fecha.

3. Durante la búsqueda siguiente:

```
# find ! -user root -perm -4000
```

   Encuentra un programa /usr/bin/lppasswd particular:

```
-rwsr-xr-x   1 lp       sys        14112 abr 18 00:55
./bin/lppasswd
```

   El derecho SUID está activado y el propietario no es root. ¿Por qué?

   El propietario de los archivos manejados por **lppasswd** es el usuario `lp`. El uso del SUID-bit no depende del propietario root. En este caso, se ejecutará el comando **lppasswd** como usuario `lp`.

2. Seguridad de los usuarios

Objetivo: gestionar la política de seguridad de las contraseñas de los usuarios.

1. Supongamos que actualmente, no existe ninguna política de seguridad de los usuarios, son libres de modificar su contraseña a su antojo. Ha utilizado el comando **chage** para modificar la política de seguridad para los usuarios existentes. Ahora va a modificar esta política para los futuros usuarios. Modifique el archivo **`/etc/login.defs`** para definir un cambio de contraseña cada 40 días, la prohibición de cambiar la contraseña antes de 7 días, y que se avise al usuario 10 días antes:

   PASS_MAX_DAYS 40

   PASS_MIN_DAYS 7

   PASS_WARN_AGE 10

2. Como root, quiere modificar sobre la marcha las contraseñas de todos los usuarios usando un script. Se genera la contraseña de 8 caracteres con pwck. Los UID de los usuarios empiezan en 1000. Construya primero un bucle que lea el archivo **`/etc/passwd`** :

   ```
   while read line
   do
   ...
   done </etc/passwd
   ```

3. Para cada línea, aísle el login y el UID:

   ```
   user=$(echo $linea| cut -d: -f1)
   uid=$(echo $linea| cut -d: -f3)
   ```

4. Genere una contraseña segura con 8 caracteres y colóquela en una variable llamada `pass`:

```
pass=$(pwgen -s -1)
```

5. Modifique la contraseña del usuario:

```
echo $pass | passwd --stdin $user
```

El script completo es:

```
while read line
do
user=$(echo $linea| cut -d: -f1)
uid=$(echo $linea| cut -d: -f3)
pass=$(pwgen -s -1)
echo $pass | passwd --stdin $user
done </etc/passwd
```

3. Seguridad general del sistema

Objetivo: evitar los rootkits, virus y controlar los límites.

1. El comando **chkrootkit** permite la detección de los rootkits más corrientes en un sistema. Usted decide colocar el comando en crontab. Se ejecutará cada día a la 1:00 h de la madrugada. Cree un archivo cron_rootkit en /etc/cron.d y coloque en él la línea siguiente:

```
0 1 * * * /sbin/chkrootkit >/tmp/rootkit
```

2. Los resultados se colocan en /tmp/rootkit, pero no es una buena idea. Lo mejor sería recibirlos por mail:

```
0 1 * * * /sbin/chkrootkit | mail user@server -s "Resultados rootkit $(date)"
```

3. Efectúe una actualización de la base de los antivirus de clamav cada dos días a las 2:00 h de la madrugada, según el mismo principio:

```
0 2 */2 * * freshclam >/dev/null 2>&1
```

4. Sus usuarios suelen consumir demasiados recursos. Limite a las personas del grupo users a 256 procesos. Modifique el archivo **`/etc/security/limits.conf`** añadiendo la línea siguiente:

```
@users   hard    nproc    256
```

5. Sea cual sea la distribución, intente actualizar siempre lo más a menudo posible sus paquetes por una cuestión de seguridad:

   Debian y Ubuntu: `apt-get upgrade`

   Red Hat, CentOS y Fedora: `yum update`

   OpenSuSE: `zypper update`

4. Seguridad de red

Objetivo: comprobar los puertos, la configuración del cortafuegos y de los TCP Wrappers.

1. Inicie nmap en su propia máquina. Compare los resultados con los de netstat -A inet -a. ¿Cuál es la principal diferencia?

   Nmap escanea los puertos abiertos en su máquina, no las conexiones salientes. netstat da la lista de los puertos locales y remotos abiertos, así como de los procesos asociados. Observe que con el argumento -A net solo se muestra el protocolo IPv4.

   Pero, sobre todo, nmap facilita, cuando los puede determinar, el nombre real y las versiones de los servicios y del sistema operativo probados. Nmap es tanto una herramienta de seguridad como de hacking.

2. Un producto como Wireshark (ex Ethereal) permite «esnifar» una red: está a la escucha de todo el tráfico, puede grabarlo, filtrarlo, etc. Dos servicios se ejecutan en un servidor: telnetd y sshd. Si escucha el tráfico de este servidor y hacia él, debería notar algo. ¿Qué?

   El tráfico a destino y desde el puerto 22 es ilegible: está encriptado. No hay una manera sencilla de recuperar o analizar el contenido. Por el contrario, el tráfico del puerto 23 es totalmente legible: no se ha añadido seguridad a la comunicación. Todo pasa sin codificar, incluyendo las contraseñas. ¿Qué tiene que hacer?

   Desactive el servicio asociado cuyo nombre encontrará en /etc/services: telnetd.

3. Puede elegir entre los TCP Wrappers para colocar una protección de tipo cortafuegos. ¿Qué principio debe guiar su solución?

   El cortafuegos netfilter trabaja a nivel de los protocolos: filtra las direcciones, los puertos, los protocolos. El control de acceso se hace al nivel del núcleo.

   Los TCP Wrappers están orientados, como indica su nombre, TCP; por lo tanto, actúan en la capa de transporte y los servicios binarios: el control de acceso se hace al nivel del servicio.

4. El archivo **`/etc/hosts.allow`** contiene:

```
sshd: ALL
```

El archivo **`/etc/hosts.deny`** contiene:

```
sshd: ALL EXCEPT 192.168.1.25
```

Puede comprobar que todo el mundo se conecta vía ssh. Es evidente que existe un error en la configuración: sólo 192.168.1.25 debería lograrlo. Corrija.

Tiene varias soluciones:

- Suprima la línea de hosts.allow y deje hosts.deny intacta.
- Modifique hosts.allow:

```
sshd: 192.1.68.1.25
```

Y modifique hosts.deny de la manera siguiente:

```
sshd: ALL
```

5. Cree reglas netfilter que prohíban a los usuarios cualquier conexión tcp en el puerto 23, salvo para las máquinas de la subred 192.168.1.0/24:

```
# iptables -A INPUT -p tcp --dport 23 -s ! 192.168.1.0/24 -j DROP
```