[FORT] Tema 1: Fortificando el sistema operativo

Un sistema operativo recién instalado es siembre inseguro. Normalmente tiene cierto número de vulnerabilidades que se originan de:

- La edad del sistema: Un sistema viejo es potencialmente más inseguro
- Los servicios que provee
- Inclusión de aplicaciones no parcheadas
- Políticas por defecto donde la seguridad no es la meta principal.

Que es endurecer el sistema operativo (Hardening)

Proceso de configurar el sistema operativo para que sea lo más seguro posible. Implica:

- Desinstalar cosas
- Deshabilitar servicios
- Restringir privilegios de aplicaciones y usuarios
- Cambiar políticas por defecto del sistema

Principios de la securización del sistema

- Se busca que el sistema sea lo más seguro posible
- Se busca minimizar el riesgo de que el sistema y la información que contiene sean compormetidos

Para lograr esto se debe:

- Identificar posibles amenazas y vulnerabilidades
- Tener muchas líneas de defensa
- Aplicar siempre el principio del mínimo privilegio

Posibles amenazas y vulnerabilidades

Para un sistema de información las amenazas pueden venir de:

- Mal funcionamiento de una aplicación
- Usuarios mal preparados
- Usuarios hostiles

Si se explota una vulnerabilidad de nuestro sistema es posible que: Nuestro sistema deje de rendir como debe La información de nuestro sistema es destruida o filtrada

Aplicaciones seguras

El software en un sistema fortificado debe ser fiable y seguro que hace lo que tiene que hacer y lo hace bien:

- Un software fiable es aquel que hace correctamente lo que tiene que hacer
- Un software seguro es el que hace SOLO lo que tiene que hacer y nada más.

Principios de seguridad

Tenemos 2 principales principios de seguridad:

Principio de varias lineas de defensa

Tenemos que asumir que todas las medidas de seguridad implementadas van a fallar:

- Se añaden medidas de seguridad en caso de que la anterior falle y así recursivamente
- También se debe saber que medida de seguridad ha fallado.

Restricción de privilegios

Cada usuario debe tener los permisos mínimos para hacer la tarea que está haciendo. Generalmente en los sistemas se tienen más privilegios de los necesarios, lo que puede ser un problema de seguridad.

- Se debe suponer que toda medida de seguridad va a fallar
- Cuando una aplicación o usuario es comprometido, la cantidad de daño que puede provocar está limitada por sus privilegios

Se recomienda que las aplicaciones se ejecuten virtualizadas.

Fases de la securización

- Instalación
- Post instalación: Se hace quitando cosas, deshabilitando servicios, modificando cuentas de usuarios y modificando configuraciones por defecto.
- Mantenimiento: Vigilancia de la máquina, aplicación de parches...

Securizando el Arranque

Es una de las partes que más dependen del hardware. En intel coexisten 2 métodos de arranque: BIOS o Legacy y UEFI:

- BIOS:
- UEFI:

ARM utilizaun sistema de Arranque similar a UEFI, mientras que SBCs como raspberry usan BIOS.

Proceso de arranque

1. Se ejecuta el código de la BIOS (Firmware) guardado en una ROM.
    - Se realiza un POST (Power On Self Test)
    - Se hace una prueba de la memoria
2. Tras eso viene el cargador, un programa lo suficientemente simple para ser ejecutado por el firmware y lo suficientemente sofisticado como para cargar el Sistema Operativo. En linux se usa GRUB (Grand Unified Boot Loader).
    - El cargador debe saber donde está el kernel para cargar el sistema operativo.
    - El cargador debe esta en el primer bloque de memoria
    - Es un archivo en un formato especial llamado .EFI en el caso de UEFI (localizados en /boot/efi/EFI).
    - Linux es un sistema operativo modular, por lo que se carga lo que es necesario en el arranque, haciendo que sea muy rápido.
    - En el caso de Linux el cargador debe saber donde están los diferentes módulos del sistema (Directorio /boot para kernel y /lib/modules para los módulos).
    - Los módulos se almacenan en el initrd agrupados (Initial Ram Disk).
    - Con “efibootmgr” se puede modificar el orden de arranque UEFI.
    - Con “efibootmgr -v” se pueden ver los archivos de arranque UEFI.
    - El cargador tiene un archivo de configuración que le dice que sistema operativo tiene que cargar.
3. Arranca el sistema operativo

Configurar Consola del Grub

```
ls #Para ver las particiones
set root=hd0,msdos3 #Seleccionamos partición
ls / #Vemos que hay en la partición en cuestión
linux /boot/vmlinux-amd64 #Le decimos donde está el kernel a GRUB
initrd /boot/initrd-nombre root=/dev/sda1 #Le decimos donde está Initrd y donde están los ficheros al GRUB
boot
```

Vulnerabilidades en el proceso de arranque

Se le debe poner una contraseña al Firmware para cambiar la configuración (También se puede hacer para cada vez que se arranca, pero esto hace que sea tedioso arrancarlo.) Introduciendo parámetros en GRUB es posible acceder al Root del equipo, lo que es una vulnerabilidad muy grave.

Securizando el sistema de Ficheros

Tenemos dos aproximaciones al uso de discos y particiones en linux:

- **Sistema de archivos en particiones**: Considerada la aproximación tradicional, el sistema de archivos se crea en cada dispositivo físico. Cada dispositivo físico debe ser montado para ser accesible. Las particiones no pueden ser cambiadas de tamaño fácilmente. Este tipo de sistema puede ser cifrado
- **LVM**: Sistema de volúmenes lógicos. Es más flexime que la aproximación tradicional ya que se puede añadir espacio dinámicamente. Es más fácil de administrar y aca volúmen lógico puede ser cifrado. No es recomendable para la partición /boot. Para usar esta aproximación en sistemas debian se debe instalar el paquete lvm2
    - Volúmenes Físicos: Son discos duros o particiones configuradas como tales.

Posibles amenazas para el sistema de archivos

- Acceso No Autorizado: Para prevenirlo se puede hacer lo siguiente:
    - Todos los directorios home deben tener el permiso 700 y podemos establecer el umask para que sea 077
    - Los archivos de configuración de diferentes daemons no deben ser legibles.
    - Algunas distros tienen programas que revisan y establecen periódicamente los permisos de los archivos en el sistema de archivos
- Que el sistema de ficheros sea llenado a tope y nadie pueda escribir en el
- Corrupción del sistema de archivos haciéndolo inutilizable
    - Usar un sistema de archivos fiable y probado
    - Mantener la máquina en un ambiente estale.
    - Tener cuidado con los permisos de archivos
- Ganar acceso a archivos maliciosos con privilegios altos.
    - Usar ACL (Listas de Control de Acceso)

Securización de Aplicaciones

Identificar y eliminar aplicaciones no utilizadas

Hay 2 tipos de aplicaciones:

- Las que tienen agujeros de seguridad
- Las que las tienen pero aún no se han descubierto

Cuando se instala una distribución de linux hay muchas aplicaciones que no son necesarias de las cuales se puede prescindir. Se pueden poner límites a las configuraciones en /etc/security/limits.conf. Estos límites son por sesión, estos límites tienen uno hard y uno soft. El límite soft se puede superar, siempre y cuando no sobrepase el límite hard. Para limitar una aplicación podemos usar:

- cpulimit: Limita el consumo de CPU de una aplicación, pero es demasiado rudimentario
- prlimit: Permite poner límites a los gastos de recursos de un proceso.

Limitar recursos de aplicaciones con cgroups

Los cgroups son jerárquicos. Para crear un cgroup hacemos lo siguiente

```
cd /sys/fs/cgroup/ #Ruta de los cgroups
mkdir prueba #Creamos nuevo cgroup
```

Los ficheros del cgroup son como los ficheros de proc. Ahora el directorio /prueba debería estar poblado por numerosos archivos. Para añadir un proceso al cgroup hacemos lo siguiente:

```
echo 2007 > cgroup.procs #Metemos el proceso 2007 en cgroup.procs
```

Para limitar el consumo de cpu hacemos:

```
echo 10000 1000000 > cpu.max #Por cada 100000 de CPU se asigna 10000 de CPU
al programa.
```

Para limitar el consumo de memoria:

```
echo 5000000 > memory.high #Se limita a 500000 de memoria el uso del
programa
```

se pueden parar todos los procesos del cegroup con:

```
echo 1 > csgroup.freeze
```

Para liberar los programas se usa:

```
 echo 0 > csgroup.freeze
```

Ejecución en jaulas chroot

El programa que se ejecuta en estas jaulas no puede subir del directorio en el que se ejecuta. Antiguamente se usaba para testear software y para servidores FTP. Se suele usar cuando se arranca un medio de instalación. Para tener un entorno ChRoot funcional se hace lo siguiente:

Entorno de virtualización

La creación de containers es muy simple, utilizamos lxc: <Code> lxc-create </code>

Se pueden ver los contenedores que se pueden crear en /usr/share/

```
lxc-create -t alpine -n NOMBRE_CONTENEDOR #En este caso alpine sería el tipo
de contenedor
```

Para arrancar el container usamos el siguiente comando:

```
lxc-start -F -n NOMBRE_CONTENEDOR #el -F indica que es en primer plano y el -n el nombre del contenedor.
```

Para parar el contanier se usa:

```
lxc-stop -n NOMBRE_CONTENEDOR
```

Los containers tienen usuarios predefinidos que se suelen indicar al crear el container. Podemos ejecutar algo en el container con:

```
lxc-attach -n NOMBRE_CONTENEDOR /bin/sh #Por ejemplo, ejecutamos un shell en el container
```

Una vez conectados así al contanier le podemos poner una contraseña con el comando “passwd root” y podemos crear un usuario nuevo en esta con “useradd -m NOMBRE”. Podemos ver los container arrancados y sus ips con:

```
lxc-ls -f
```

Para entrar en un container que no se ha iniciado con -F podemos usar SSH contra su IP. lxc crea una interface llamada bridge 0 para conectar los containers mediante NAT como si fuera virtualbox. para configurar la red de un container vamos a la dirección:

```
cd /var/lib/lxc
```

Aquí hay una carpeta por container, para configurar uno vamos al que queramos y modificamos el archivo config. Dentro de este se pueden ajustar más parámetros, como el autoarranque o la memoria entre otros. Mediante NFTables podemos comunicar el container con el exterior para que pueda prestar servicios. Esto nos permite arrancar aplicaciones de forma aislada. Dentro de /var/lib/lxc/NOMBRE_CONTAINER/rootfs podemos encontrar los archivos usandos dentro del container. Estos aparecen como pertenecientes al usuario que creó el container, a pesar de ello la máquina los puede usar, esto se debe a que comparte los identificadores con la máquina HOST

Mandatory Access Control

Hay 2 tipos:

SELinux

Usado por Fedora. Permisos por archivos, todo tiene una etiqueta (Archivos, procesos, etc…) que dice que puede acceder a que. Solo se admiten los acceso permitidos por las etiquetas. Para convertir una máquina en SELinux primero hay que instalar los siguientes paquetes:

```
apt install selinux-basics selinux-utils selinux-policy-defualt auditd
```

Tras es activamos selinux con el siguiente comando:

```
selinux activate
```

esto crea un archivo /.autorelabel para etiquetar los ficheros no etiquetados. Selinux necesita EXT4. Si se usa Selinux no se puede compartir el directorio home entre dos distros que no lo usen. Selinux tiene 2 modos:

- Modo permisivo: selinux simplemente manda warnings al log, pero no bloquea el acceso.
- Modo enforce: Bloquea accesos no autorizados, lo malo es que solo avisa la primera vez que se intenta realizar un acceso no autorizado.

También añade al grub en /boot/grub/grub.cfg el modo security = selinux. En /etc/selinux/config podemos cambiar el modo de selinux de permisive a enforce.

APParmor

Usado por debuan. Permisos por aplicación indicando donde puede acceder y donde no. Para habilitar un programa con apparmor usamos el comando:

```
apparmor_parser /etc/apparmor.d/usr.bin.programa
```

Por defecto el programa se pondrá en modo enforce. Para saber el estado de las aplicaciones en apparmor usamos

```
aa-status
```

para que al usar un programa simplemente mande un warning usamos complain:

```
aa-complain /usr/bin/programa
```

Securización de las cuentas de usuarios

Se trata de endurecer la autenticación y evitar que algunos usuarios tengan más permisos de los que deberían.

Comandos

Obtener ID de una partición

```
blkid /dev/sda4
```

Crear un grupo de volúmenes

Ponemos el nómbre que le queremos dar al grupo y los volúmenes que queremos usar para crear este.

```
vgcreate GRUPOVOLS /dev/sda4
```

Estender un grupo de volúmenes

Primero ponemos el grupo a modificar y después los volúmenes que queramos añadir.

```
vgextend GRUPOVOLS /dev/sdb1 /devsdb2 /dev/sdb3
```

Mostrar grupo de volúmenes

```
vgdisplay
```

Crear un volúmen lógico

Con -L indicamos el tamaño, después el grupo de volúmenes que vamos a utilizar y finalmente el nombre del volúmen lógico.

```
lvcreate -L 15G GRUPOVOLS VOLUMILLO
```

La ubicación resultante está en: /dev/GRUPOVOLS/VOLUMILLO