Sociedad Andaluza para el Desarrollo de las Telecomunicaciones S.A.
CONSEJERÍA DE ECONOMÍA, INNOVACIÓN, CIENCIA Y EMPLEO

AndalucíaCERT
CENTRO DE SEGURIDAD TIC

Informe de divulgación
Bastionado de Sistemas (II)

Tipo de documento:	*Informe*
Autor del documento:	*AndalucíaCERT*
Código del Documento:	*CERT-IF-2718-260912*
Edición:	*0*
Categoría	*Público*
Fecha de elaboración:	*26/09/2012*
N° de Páginas	*1 de 43*

1 TABLA DE CONTENIDOS

2 OBJETO

El objeto de este documento es proporcionar al personal técnico de la Junta de Andalucía una serie de directrices complementarias a la primera parte del informe divulgativo que les sirva para poder implementar de una manera técnica las recomendaciones de fortificado, reduciendo de esta manera la exposición de los sistemas ante posibles ataques.

3 ALCANCE

Este documento va destinado al personal técnico de la Junta de Andalucía y al público en general. En esta segunda parte del informe divulgativo de bastionado se han expuesto de una manera técnica los puntos desarrollados en el informe previo, por lo que recomiendan conocimientos sobre administración de sistemas.

Los ejemplos y configuraciones se han desarrollado para entornos Linux, principalmente enfocado a sistemas Debian y Red Hat, aunque se considera citar en los casos que se han considerado oportunos otras platafomas.

4 INTRODUCCIÓN

El bastionado de sistemas es un proceso necesario en el marco de cualquier proyecto que contemple la aplicación de controles de seguridad sobre los sistemas de información. El objetivo de bastionar un sistema es reducir el área de superficie de ataque, eliminando tantos riesgos para la seguridad como sean posible del sistema objetivo.

Habitualmente, el bastionado de un sistema es un proceso que requiere una gran inversión de tiempo y en el que hay que tener en cuenta distintos puntos claves. Así mismo, es necesario recordar que como todos los distintos controles de seguridad y de calidad de una organización, asegurar un sistema, dentro de un programa de mejora o control de la seguridad de la información, se encuentra permanentemente en constante desarrollo y mejora.

Como en el primer informe divulgativo, se ha distribuido el documento teniendo en cuenta los principales puntos a los que debemos prestar especial atención a la hora de llevar a cabo esta labor:

- Aplicaciones y servicios
- Acceso remoto
- Acceso local
- Sistema operativo

5 APLICACIONES Y SERVICIOS

Actualmente, muchas distribuciones de distintos sistemas operativos están diseñadas para presentar al usuario el máximo número de opciones y aplicaciones activas posibles con el peligro que todo esto conlleva. Hay que tener presente que, potencialmente, cada programa instalado es un riesgo para la seguridad del sistema, ya que puede ser aprovechado de distintas maneras por un atacante para comprometer la seguridad del mismo. Asegurar un sistema no sólo requiere comprender cómo funciona internamente; requiere también aprender a deshacer lo que otros han hecho en interés de asegurar el trabajo interno.

En el nivel de aplicación, la máxima en el mundo de la seguridad: *"Lo que no está permitido explicitamente, está prohibido"* es extensible a este ámbito, de esta forma se diría: *"Ninguna aplicación o proceso debería tener más privilegios en un entorno de sistema operativo local que el necesario para funcionar"*. Las recomendaciones para el bastionado de un sistema con respecto a los programas instalados en el sistemas son las siguientes:

5.1 Instalar sólo el software necesario

Para determinar "*qué es necesario"* hay que basarse en tres pilares:

- Sentido común.
- Manuales e información.
- Gestores de paquetes.

La primera de ellas nos dicta que, por ejemplo, no hay necesidad de instalar un servidor web Apache en un cortafuegos o de tener entorno gráfico en un servidor que sólo será accesible mediante consola, hay que tener siempre presente que lo que no es necesario podría ser considerado necesariamente como un riesgo.

Antes de instalar o desinstalar, hay que preguntarse siempre **si realmente ese software es imprescindible y qué implicaciones puede tener para el sistema este cambio**. Si se desconoce qué hace un determinado paquete o comando existen varias maneras de precisar su funcionamiento. La manera más sencilla es acudiendo al manual, este nos mostrará una sinopsis del comando en cuestión:

$man <comando/programa>

Si no existe la página del manual o si no tiene información completa sobre qué comando está buscando, se puede intentar obtener información mediante palabras clave:

$man -k <palabra-clave>
$apropos <palabra-clave>

Si se requiere información avanzada sobre los cambios que realiza un determinado programa o paquete dentro del sistema, siempre puede acudir a la información facilitada a través del gestor de paquetes que tenga instalado en su sistema:

- Paquetería .deb (dpkg y apt):

```
$apt-cache show <paquete>      #Información sobre el paquete
$dpkg -L <paquete>             #Listado del contenido del paquete
$dpkg -S <archivo>             #Muestra a qué paquete en concreto pertenece <archivo>
```

- Paqueteria .rpm (yum):

```
$yum info <paquete>  #Muestra información sobre el paquete
$yum provides <paquete/archivo> # Muestra que funcionalidad realiza cierto paquete o a que paquete
                                pertenece cierto archivo
```

5.2 Mantener el software actualizado

Una vez determinado el software que permanece en el sistema, este deberá mantenerse actualizado. Mantener el sistema actualizado es una de las prácticas generales más recomendadas en el campo de la seguridad informática, las actualizaciones de los programas vienen justificadas principalmente por dos motivos:

- Corregir las vulnerabilidades detectadas.
- Proporcionar nuevas funcionalidades o mejoras respecto a las versiones anteriores.

La gran mayoría de sistemas operativos en la actualidad poseen mecanismos de actualización de software de manera automática, aunque estas tareas se pueden realizar manualmente, se intenta siempre descargar de este cometido al usuario o administrador intentando que estas sean lo más transparentes posibles.

La existencia e instalación de actualizaciones se pueden realizar de varias maneras, dependiendo del gestor de paquetes que se use:

- Paquetería .deb (apt):

```
$sudo apt-get update           #Actualiza la lista de paquetes de los repositorios
$sudo apt-get upgrade          #Actualiza los paquetes que se hayan marcado como actualizables
$sudo apt-get dist-upgrade     #Actualiza a la última versión del S.O
```

- Paqueteria RPM (yum):

```
$yum update <paquete>          #Actualizar un paquete determinado
$yum update                    #Actualiza el sistema y todas las aplicaciones actualizables
$yum update –exclude=<pkg>     #Actualiza excluyendo un paquete
```

En sistemas Windows las actualizaciones del sistema se realizan mediante *Windows Update* y se delega la responsabilidad de la comprobación y actualización de cada aplicación de terceras partes en manos del programa o del usuario. Aplicaciones como **Secunia PSI o CONAN (Inteco)**, comentadas en el anterior informe, permiten detectar programas vulnerables y desactualizados que requieran de parches de seguridad, así como configuraciones incorrectas.

En la administración de sistemas críticos hay que hacer ciertas consideraciones entre sistemas de producción estables y sistemas de desarrollo e investigación. Es importante ser especialmente cuidadosos a la hora de añadir, eliminar o actualizar software en los sistemas de producción. En líneas generales se recomienda probar antes todos los cambios que vayamos a realizar en estos sistemas en un sistema de preproducción de similares características y configuración.

5.3 Integridad del software y archivos

La comprobación de la integridad de un archivo o programa nos permite averiguar si algún dato del mismo ha variado desde su creación. En el anterior informe se comentaron varios comandos básicos para la comprobación de integridad, como son **md5sum** y **sha256sum**. Para la comprobación de la integridad en la gestión de la paquetería, se buscó una manera de solucionar el problema de una manera automática y a partir de la versión 0.6 del gestor de paquetes apt, se introdujo la comprobación de integridad mediante hash criptográficos, combinados con la criptografía de clave pública para validar los paquetes descargados, llamado **apt-secure**.

El funcionamiento de este sistema es el siguiente:

Un paquete *.deb* contiene un archivo llamado *Release*, el cual se actualiza cada vez que cualquiera de los paquetes en el archivo cambian. Entre otras cosas, el fichero *Release* contiene algunos md5sums de otros ficheros en el archivo. Un extracto de un ejemplo de fichero:

```
MD5Sum:
6b05b392f792ba5a436d590c129de21f        3453 Packages
1356479a23edda7a69f24eb8d6f4a14b        1131 Packages.gz
2a5167881adc9ad1a8864f281b1eb959        1715 Sources
88de3533bf6e054d1799f8e49b6aed8b         658 Sources.gz
```

Si observamos en el interior de un fichero *Packages*, encontraremos más md5sums, uno por cada paquete listado en él. Por ejemplo:

Package: uqm
Priority: optional
...
Filename: unstable/uqm_0.4.0-1_i386.deb
Size: 580558
MD5sum: 864ec6157c1eea88acfef44d0f34d219

Estos dos checksums permiten a apt verificar que se ha descargado una copia correcta del archivo *Packages*, con el checksum que concuerde con el que se encuentra en el archivo *Release*. Y cuando se descargue cada paquete individual, también podrá comparar el checksum con el contenido del archivo *Packages*. Si apt falla en cualquiera de estos pasos no continuará la instalación.

Aun así, apt no tiene manera de comprobar la integridad del archivo *Release*. Para solucionar este problema, se añade una firma GPG para dicho archivo llamada *Release.gpg* y que se descarga junto a dicho archivo *Release*, validando de esta manera la transferencia mediante el uso sistemas de claves públicas.

Es por ello que muchas veces a la hora de instalar paquetes o actualizar el sistema nos encontramos con uno o varios mensajes de error de este tipo:

W: GPG error: http://ftp.us.debian.org testing Release: The following signatures couldn't be verified because the public key is not available: NO_PUBKEY 010908312D230C5F

Esto nos indica que no se ha podido descargar el archivo *Release.gpg* o que la firma no es válida. Si intentamos forzar la instalación nos mostrará el siguiente aviso:

WARNING: The following packages cannot be authenticated!
libglib-perl libgtk2-perl
Install these packages without verification [y/N]?

Es importante tener en cuenta estos avisos a la hora de instalar programas en nuestro sistema, ya que puede indicarnos que un paquete no es de quien realmente debería ser.

Por tanto, la seguridad del sistema entero depende de que haya un fichero *Release.gpg*, el cual firma un fichero *Release*, y de Apt comprobando que esa firma use gpg. Para comprobar la firma, debe conocer la llave pública de la persona que firmó el fichero. Estas llaves (keys) se almacenan en servidores públicos como *keyring.debian.org, wwwkeys.pgp.net o keyserver.ubuntu.com* y deben ser importadas antes de usarse:

```
$sudo apt-key list                                  #Muestra la lista de claves del sistema
$gpg –keyserver  <server> –recv-keys  <key>         #Descarga la llave y la añade al keyring de gpg
$gpg -a –export 55BE302B | sudo apt-key add - #Se exporta la llave del keyring gpg y se añade al sistema
```

Aunque *apt-secure* se puede considerar bastante seguro en cuanto a la integridad y validez en la gestión de paquetes, si se usa un sistema operativo basado en repositorios de paquetes, hay que tener siempre en consideración que dichos servidores podrían ser también vulnerables. De hecho, son continuamente atacados, ya que debido a su naturaleza son un objetivo tentador para cualquier atacante. Comprometer un sistema de este tipo o conseguir poder modificar ilícitamente el software ahí alojado, comprometería automáticamente a cientos de miles de máquinas.

Hay que tener en mente que **la seguridad de nuestro sistema dependerá siempre de la seguridad de su punto más débil**. Descargar e instalar paquetes de repositorios es confiar (y en cierta manera, delegar) la seguridad de nuestro sistema en un servidor que se encuentra administrado por terceros y del que no tenemos información de su grado de seguridad.

5.4 Servicios potencialmente inseguros. Configuración y deshabilitación

La mejor protección contra software vulnerable es ejecutar menos software. Desinstalar o deshabilitar servicios que se estén ejecutando en el sistema permiten que la superficie de ataque contra un sistema se reduzca.

Para deshabilitar o desinstalar servicios innecesarios es necesario conocer a fondo que hace cada uno de ellos y que potencial impacto puede tener en el sistema al que aplicamos los cambios, pues puede verse afectadas funcionalidades criticas de los sistemas.

Para la eliminación de programas usando sistemas de paqueteria:

- Paquetería .deb (apt):

$ apt-get remove <paquete> #Elimina el paquete objetivo

- Paquetería RPM (yum):

$ yum remove <paquete> #Elimina el paquete objetivo

Aunque lo ideal es eliminar los paquetes no necesarios, esta no siempre es la mejor opción, ya que puede ser que de ese paquete dependan otros que si sean necesarios o quizás desea conservar ese programa por si lo usa en el futuro, en ese caso, hay que proceder a deshabilitar el servicio del arranque.

$rpm -qf /etc/init.d/<servicio> #Comprueba qué paquete se encuentra asociado a <servicio> (RPM)

#chkconfig –list #Listado con las configuraciones de ejecución de los servicios

Ejemplo:

```
acpid          0:off  1:off  2:on   3:on   4:on   5:on   6:off
apache2        0:off  1:off  2:off  3:off  4:off  5:off  6:off
arpwatch       0:off  1:off  2:on   3:on   4:on   5:on   6:off
atd            0:off  1:off  2:on   3:on   4:on   5:on   6:off
bootlogd       0:off  1:off  2:off  3:off  4:off  5:off  6:off  S:on
bootmisc.sh    0:off  1:off  2:off  3:off  4:off  5:off  6:off  S:on
checkfs.sh     0:off  1:off  2:off  3:off  4:off  5:off  6:off  S:on
```

#chkconfig –level <n> <service> on/off # Activa/desactiva un servicio al nivel n

En distribuciones basadas en Debian, además podemos usar *update-rc.d:*

#update-rc.d -f <servicio> remove #Deshabilita un servicio

Aunque la máxima es dejar tan pocos servicios ejecutándose en el sistema como sea imprescindible, obviamente una máquina debe tener servicios disponibles para cumplir con sus funciones.

Estos servicios que deban estar disponibles, deben ser asegurados correctamente. De manera general, los servicios pueden ser asegurados de dos maneras distintas:

- **Haciéndolos accesibles únicamente en los puntos de acceso** (interfaces y redes) en los que necesiten estar.
- **Configurándolos apropiadamente** de manera que sólo pueda usarse por usuarios legítimos de la manera apropiada.

Como ejemplos de una correcta configuración de los servicios, vamos a estudiar dos servicios comúnmente instalados y habilitados en muchos sistemas por necesidad o por desconocimiento, y que son especialmente vulnerables: Avahi y DHCP.

5.4.1 Avahi Server:

Avahi es una implementación libre de zeroconf, el cual hace uso del protocol multicast DNS (mDNS/DNS-SD) para el descubrimiento de servicios y equipos en una red. Esto permite a un sistema identificar de manera automática qué recursos se encuentran disponibles en su red, como impresoras y servidores. Avahi se encuentra habilitado por defecto en multitud de distribuciones Linux, con el riesgo que esto conlleva.

En primer lugar, lo ideal sería deshabilitar el servicio si fuera posible:

 # chkconfig avahi-daemon off

Si se requiere que el servicio se encuentre activo, sería necesaria una configuración más restrictiva para aumentar la seguridad y evitar volcar a la red más información de la indispensable.

La configuración del demonio avahi se realiza en el fichero: */etc/avahi/avahi-daemon.conf*.

- **Disponibilidad exclusiva por el protocolo adecuado**

La configuración por defecto se establece para permitir el uso tanto de IPv4 como de IPv6, si sólo se usa uno de los dos protocolos, se puede modificar la linea:

use-ipv6=no (o *use-ipv4=no* si requiere lo contrario)

- **Comprobaciones del campo TTL**

Se puede configurar el servicio para ignorar paquetes IP a no ser que su campo TTL sea 255. En la sección *[server]:*

check-response-ttl=yes

- **Prevenir que otros programas usen el puerto de Avahi**

En la sección *[server]* del archivo de configuración, añadir:

disallow-other-stacks=yes

- **Deshabilitar la publicación si es posible**

La configuración por defecto permite al servicio Avahi enviar distinta información sobre el sistema, como pueden ser los servicios o su registro de direcciones, a toda la red local.
Obviamente, por motivos de seguridad, **estas publicaciones deben de evitarse si no son absolutamente necesarias.**

Para evitar la publicación de la información del equipo, pero seguir permitiendo realizar peticiones a la red para el descubrimiento de servicios, hay que establecer el siguiente valor en la sección [publish]:

disable-publishing=yes

Esta configuración es recomendada para sistemas que no deban revelar sus servicios o su existencia en la red.

- **Restringir la información de las publicaciones**

Por otra parte, es posible **controlar qué tipo de información se quiere publicar a la red**. Para ello, en la seccion *[publish]*:

disable-user-service-publishing=yes

y posteriormente, añadir tantas restricciones como se quieran:

publish-addresses=no
publish-hinfo=no
publish-workstation=no
publish-domain=no

Se recomienda usar estas opciones **incluso si se restringe completamente la publicación** vía *disable-publishing=yes ,* ya que previene cualquier tipo de intento de publicación. El uso de ambas opciones perfecciona una configuración más segura del servicio.

5.4.2 DHCP:

DHCP (Dynamic Host Configuration Protocol, en español "protocolo de configuración dinámica de host") es un protocolo de red que permite a los clientes de una red IP obtener sus parámetros de configuración automáticamente.

Existen multitud de ataques que permiten aprovecharse de una configuración incorrecta de este servicio y que permiten vulnerar la seguridad de la red de distintas maneras, es por ello que es necesario comprobar si es absolutamente necesario tener activado protocolo y en caso afirmativo, establecer la configuración más adecuada para nuestro entorno.

- **Deshabilitar el servicio si fuera posible**

Es recomendable **estudiar si es realmente imprescindible que la configuración de red se realice mediante DHCP.** En servidores y equipos estáticos que siempre se encuentren en la misma red habitualmente es posible configurar los parámetros de red estáticamente introduciendo los valores que correspondan en la configuración de la interfaz.

En Debian, editar el archivo */etc/network/interfaces:*

auto <iface>
iface <iface> inet static
IP : <IP>
Netmask : <Netmask>
Puerta de enlace : <IP>

En RedHat, editar */etc/sysconfig/network-scripts/ifcfg-IFACE* y realizar los siguientes cambios:

BOOTPROTO=static
NETMASK=<Netmask>
IPADDR=<IP>
GATEWAY=<IP>

Para deshabilitar los servicios:

 # chkconfig dhcpd off

Y si es posible, desinstalar el paquete:

 # yum erase dhcp #En RedHat
 # apt-get remove dhcp #En Debian

- **Minimizar las configuraciones permitidas para DHCP**

Si se debe usar DHCP obligatoriamente, existen ciertos cambios en la configuración que reducen la cantidad de información que se recibe y expone a la red. Por ejemplo, existe la posibilidad de solicitar tan sólo ciertos parámetros al servidor DHCP, teniendo configurados otros por defecto. Todas las configuraciones se modifican en el archivo */etc/dhclient.conf* .

Si no se desea obtener algún parámetro de red (IP, máscara, dns...) específico desde el servidor, se introducirá:

supersede <parámetro> <valor>;

Si se quiere obtener un parámetro específico desde el servidor, se añadirá:

request <parámetro> ;

Así, por ejemplo, si sólo nos interesa obtener la dirección IP y la mascara de red:

```
supersede       domain-name "example.com ";
supersede       domain-name-servers 192.168.1.2 ;
supersede       nis-domain "";
supersede       nis-servers "";
supersede       ntp-servers "ntp.example.com ";
supersede       routers 192.168.1.1 ;
supersede       time-offset -18000 ;

request subnet-mask;
require ip-address;
```

- **No permitir el uso de DNS dinámicos**

Para evitar que el servidor DHCP reciba información DNS de los clientes, se debe añadir el siguiente parámetro de configuración en /etc/dhcpd.conf:

ddns-update-style none;

- **Denegar mensajes DECLINE**

Un mensaje DECLINE es enviado por el cliente para indicar que no considera válida la IP enviada por el servidor, un atacante podría acabar con toda la reserva de IP's del servidor causando una denegación de servicio. Modificar en /etc/dhcpd.conf:

deny declines;

- **Denegar peticiones BOOTP**

En el caso de que no se necesite soporte a peticiones BOOTP, se recomienda deshabilitar esta opción:

deny bootp;

- **Minimizar la información expuesta**

Finalmente, se recomienda editar el archivo */etc/dhcpd.conf* y examinar cada sección dentro del mismo. Es necesario asegurarse que las siguientes opciones *no se encuentran definidas* a no ser que sea operacionalmente indispensable permitir que esta información se exponga a través de DHCP.

option domain-name
option domain-name-servers
option nis-domain
option nis-servers
option ntp-servers
option routers
option time-offset

Se han estudiado aquí dos servicios que, debido a su posible impacto o su amplia distribución, se han considerado que deben tratarse en profundidad. Sin embargo existen otros servicios de igual o más importancia según el entorno, que deben ser considerados, analizada su necesidad y configurados correctamente. Son los casos por ejemplo de los distintos servicios de impresión (CUPS, HPLIP...) o los servicios NTP (Network Time Protocol).

6 ACCESO REMOTO

6.1 Gestión de conexiones de red

Es importante saber gestionar las conexiones de red de una manera adecuada. Sin duda, un perímetro de red correctamente configurado puede mitigar gran parte de los ataques al servidor y puede prevenir que sistemas comprometidos dentro de una red puedan ser utilizados para atacar a otros sistemas. Sin embargo, el diseño e implementación de perímetros de red se sale del alcance de este documento, tanto por temática como por envergadura, por lo que vamos a centrarnos en una serie de consideraciones relativas a la la gestión de conexiones de red en las que se encuentre involucrado exclusivamente el equipo que queramos bastionar.

6.1.1 Aislar la máquina del resto de equipos de la misma red

Esta primera medida es de carácter topológico. Es importante, estudiar las necesidades de nuestra red para ubicar el sistema bastionado: qué servicios provee al exterior, quién debe permitirse acceder a dichos servicios, etc.

En redes de un determinado tamaño, se cuenta con varias subredes con diferentes políticas de seguridad, siendo necesario configurar arquitecturas de firewall que aíslen las diferentes redes de una compañía y que permitan circular entre las redes solo cierto tráfico definido. Como norma general, si el equipo provee de un servicio al exterior, el sistema se ubicará en la denominada zona desmilitarizada o DMZ.

El objetivo de una DMZ es que las conexiones desde la red interna y la externa a la DMZ estén permitidas, mientras que las conexiones desde la DMZ solo se permitan a la red externa, los equipos en la DMZ no pueden conectar con la red interna.

Esto permite que los equipos de la DMZ puedan dar servicios a la red externa a la vez que protegen la red interna en el caso de que intrusos comprometan la seguridad de los equipos situados en la zona desmilitarizada. Para cualquiera de la red externa que quiera conectarse ilegalmente a la red interna, la zona desmilitarizada se convierte en un callejón sin salida.

6.1.2 Escuchas de los servicios en la interfaces adecuadas

En muchas ocasiones, los programas se configuran por defecto para escuchar por todas las interfaces disponibles en los sistemas en los que se instalan. Esto puede suponer un problema, ya que podemos encontrarnos con servicios sirviendo en redes donde no necesitamos o no queremos que estos servicios se encuentren disponibles, aumentando la superficie de ataque de dicho sistema.

Como norma general, se recomienda:

- **Comprobar que servicios y en qué puertos se están siendo ejecutados los programas** mediante el comando "***netstat -paunt***", así como hacia donde permite las conexiones.
- **Repasar los archivos de configuración** de los programas afectados y los que se instalen en el sistemas para asegurarse de que no se instalan en interfaces inadecuadas.

6.1.3 Servicios con accesos locales exclusivos

Existen una serie de servicios, que si bien necesitan encontrarse presente en el sistema, es posibles configurarlos para que **sólo atiendan a peticiones locales** (esto es, peticiones realizadas y recibidas desde el propio equipo) y que no necesitan atender a peticiones externas para su correcto funcionamiento. En muchas ocasiones, debido a una mala configuración por defecto o a modificaciones que se haga al servicio, este servicio se encuentra disponible a la red sin que esto sea realmente necesario. Es el caso por ejemplo de mysql, si no es necesario que se acepten peticiones externas al equipo local, porque la aplicación que la necesite ya se encuentre en el mismo equipo o por cualquier otro motivo, es necesario comprobar que en el archivo de configuración *my.cnf* se encuentra la línea:

bind-address=127.0.0.1

6.1.4 Uso de hosts.allow y hosts.deny

En multitud de ocasiones, no es necesario configurar complejas reglas de firewall para evitar el acceso a ciertos servicios o redes en el equipo. Además de la configuración del firewall es interesante tener en mente que *hosts.allow* y *hosts.deny* , permiten agregar una capa extra de seguridad al sistema. Sin embargo, es necesario indicar que hay algunos programas que no tienen soporte para este tipo de filtrado. Por ejemplo servicios como ssh lo permiten, mientras que otros como mysql no.

Modificando *host.allow* indicaremos que direcciones o servicios queremos permitir el acceso, mientras que con *host.deny* indicaremos cuales queremos denegar. La sintaxis de la configuración es la siguiente:

daemon : dirección : acción

ó

daemon : dirección

Por ejemplo, si queremos permitir *únicamente* el acceso a las IP's para el servicio SSH, en el archivo host.allow indicamos:

sshd: 192.168.0.111 192.168.0.112 192.168.0.113

Mientras que en el archivo /etc/hosts.deny denegamos al resto:

sshd: ALL

6.1.5 Desactivación de IPV6

Como cualquier otro protocolo de red, **IPv6 debería ser deshabilitado si no va a ser necesario.** En la mayoría de distribuciones de Linux y en muchos otros sistemas operativos, IPv6 viene configurado por defecto, sin embargo, no son muchos los usuarios ni servicios que hagan uso de este protocolo, por lo que en la mayoría de las casos, y para evitar posibles ataques usando este protocolo de red, es recomendable desactivarlo.

- **Deshabilitar la carga del módulo del kernel**

En RedHat añadir al archivo */etc/modprobe.conf* , la siguiente linea:

install ipv6 /bin/true

En Debian editamos el archivo /etc/modprobe.d/aliases o si usamos un kernel 2.6 o /etc/modprobe.conf si usamos un kernel 2.4 y cambiamos la linea que dice:

alias net-pf-10 ipv6

por

alias net-pf-10 off

- **Deshabilitar el uso de la interfaz de red**

Para deshabilitar el uso de la interfaz de red, editar */etc/sysconfig/network* y añadir o modificar las siguientes lineas:

NETWORKING_IPV6=no
IPV6INIT=no

6.1.6 IPTables

Es posible configurar un firewall para gestionar las conexiones entrantes y salientes de un equipo bastionado para los distintos servicios. Es importante antes de crear las reglas, determinar correctamente los diferentes parámetros (puertos, direcciones de red, mascara...) así como la acción que queramos implementar.

Sin pretender extendernos, si por ejemplo, quisiéramos permitir acceso al servicio SSH exclusivamente a los equipos de la red local:

En el archivo /etc/sysconfig/iptables introducir la siguiente linea:

iptables -A RH-Firewall-1-INPUT -s <red/mascara> /mask -m state –state NEW -p tcp –dport 22 -j ACCEPT

6.2 Control de acceso. Administración remota

6.2.1 Configuración de terminales

Habitualmente, el acceso a las distintas cuentas shell son accesibles mediante un par valido de nombre de usuario y contraseña, Este tipo de autenticación es muy vulnerable a cierto tipo de ataques como el sniffing de la red o la adivinación de contraseñas débiles. Por lo tanto, los mecanismos de acceso a cuentas introduciendo nombres de usuario y las contraseñas deben limitarse a lo operativamente necesario.

En los accesos locales es necesario establecer y delimitar a través de qué canales se va a permitir el acceso como superusuario, siendo necesario restringir los terminales donde no sea necesario acceder como root al sistema. Para ello, se debe editar el archivo */etc/securetty* y se debe comprobar que la lista de terminales permitidos sea la estrictamente establecida.

Por otra parte, como medida adicional de seguridad, tanto para accesos locales como remotos, es recomendable establecer un tiempo de expiración de la sesión por inactividad.

La variable de entorno TMOUT nos permite definir el tiempo que queremos permitir a un usuario permanecer dentro de la shell o sesión SSH sin hacer nada (estado idle o inactivo). Para configurar esta variable, simplemente la añadimos dentro de nuestro perfil de variables/configuraciones de bash *.bash_profile o .bashrc.* Conviene ser configurada como read only para evitar que el propio usuario pueda modificarla. Para establecer un timeout de 2 minutos:

TMOUT=120
readonly TMOUT

6.2.2 Configuración de acceso SSH

Como se comentó en la primera parte del informe, la forma más habitual de administrar sistemas Linux de manera remota es mediante el uso de Secure Shell (SSH), este programa, aunque es notablemente más seguro que sus predecesores, no es invulnerable, por ello es necesaria una correcta configuración para hacer el sistema más seguro:

- **Habilitar explícitamente la versión 2 del protocolo**

Se deben permitir exclusivamente las conexiones que usen la versión 2 del protocolo. La versión 1 del protocolo tiene varios fallos de seguridad, por lo tanto es necesario comprobar que en el archivo */etc/ssh/sshd_config* se encuentra la línea:

Protocol 2

- **Limitar los usuarios que tienen acceso a SSH**

Por defecto, SSH permite el acceso al sistema a cualquier usuario, es posible establecer una lista de usuarios que no tienen acceso al sistema mediante el parámetro:

DenyUsers USER1 USER2

De la misma manera también se puede establecer una lista con tan sólo los usuarios que estén permitidos:

AllowUsers USER1 USER2

- **Establecer tiempo de expiración de la sesión**

Se permite establecer un tiempo máximo de inactividad para la sesión.

ClientAliveInterval <tiempo>
ClientAliveCountMax 0

- **Deshabilitar los archivos .rhosts**

Es necesario comprobar que existe la siguiente línea:

IgnoreRhosts yes

- **Deshabilitar el acceso mediante root**

El usuario root no debería permitirse autenticarse directamente de manera remota. Para deshabilitar dicho acceso directo, añadir la siguiente línea:

PermitRootLogin no

- **Deshabilitar el acceso a contraseñas vacias**

PermitEmptyPasswords no

- **Establecer un banner de aviso**

Es posible crear un aviso para que aparezca como aviso al acceder al sistema, para habilitarlo, añadir la siguiente línea con la ruta donde se introducirá dicho aviso:

Banner /etc/issue

- **Prohibir que los usuarios establezcan variables de entorno**

PermitUserEnvironment no

- **Configurar el acceso por certificado**

Es posible configurar el servicio SSH para añadir una capa de seguridad extra, configurando el acceso exclusivo por certificados. Para realizar esta tarea en primer lugar es necesario generar un certificado en el equipo cliente:

 # ssh-keygen <-b bits> -t <type> <-N passphrase> <-f fichero>

Una vez completado tendremos dos ficheros *id_rsa* e *id_rsa.pub*, que es el que contiene la clave privada y pública respectivamente.

Después de haber generado la clave, es necesario copiar al servidor la clave pública:

 # ssh-copy-id <-i fichero_clave> usuario@servidor

Con la opción *-i* indicaremos el fichero que contiene la clave pública generada en el paso anterior. Solicitará la "passphrase" anteriormente indicada y se instalará en *.ssh/autorized_keys.*

Una vez finalizado este paso ya es posible acceder al servidor mediante el uso de los certificados, que solicitará el passphrase en lugar de la contraseña habitual.

Es posible hacer que el servidor no responda ninguna petición si no viene acompañada de certificado, para ello editamos el fichero */etc/ssh/sshd_config* y modificamos la línea:

PasswordAuthentication no

7 ACCESO LOCAL

Es habitual que durante el bastionado de un servidor se preste gran atención a protegerlo contra atacantes externos o remotos. Pero, si un atacante es capaz de burlar a los sistemas de protección perimetral, y consigue conexión directa con la máquina víctima, todavía no tiene que estar todo perdido si hemos reforzado la seguridad local del sistema adecuadamente.

Es necesario proteger el sistema localmente para posteriormente poder protegerlo cuando se exponga al exterior.

7.1 Cuentas de usuario

En sistemas Linux, es habitual que tras la instalación del sistema o de algún programa se creen cuentas de usuario en el fichero */etc/passwd* para aplicaciones especificas. Aunque parezca no tener importancia, la existencia de esas cuentas pueden ser potencialmente utilizadas por atacantes.

Es recomendable verificar */etc/passwd* y comentar las entradas que se crean innecesarias. Es importante que las cuentas que no sean de usuario no tengan una cuenta shell disponible (el campo del interprete de comandos debe ser */sbin/nologin* o */bin/false*) de lo contrario esa cuenta podría estar siendo usada para acceder al sistema.

Nota: Una forma de determinar la validez de la cuenta es verificar si es propietaria de archivos y si es así, ver cual fue la ultima fecha de modificación.

Habitualmente, algunas de las cuentas necesarias suelen ser:

- root
- bin
- daemon
- halt
- shutdown
- man
- at

Por el contrario, algunas cuentas que son a menudo innecesarias es un sistema bastionado:

- uucp
- games
- gdm
- xfs
- rpcuser
- rpc

Por otra parte, se debe tener en cuenta que si se comparte el acceso con varios usuarios, es necesario determinar que estos usuarios son realmente necesarios y que poseen los permisos adecuados.

7.2 Política de contraseña

Como norma general, se establece la necesidad de establecer una **correcta política de contraseñas que obligue a los usuarios a cumplir con una serie de requisitos mínimos** a la hora de seleccionar una contraseña adecuada para el acceso al sistema.

Herramientas de sistema como *secpwgen* permiten la generación de contraseñas seguras bajo las condiciones que le pidamos, sin embargo, es necesario comprobar y forzar a que las contraseñas del sistema se encuentren bajo unos criterios comunes de seguridad.

7.2.1 Comprobaciones generales

De manera general existen una serie de comprobaciones que son interesantes realizar en cualquier sistema:

- **Verificar que no existen cuentas con contraseñas vacías:** Si alguna cuenta tiene el campo de contraseñas vacío, cualquiera podría acceder en el sistema y ejecutar comandos con los permisos de dicha cuenta.

  Para comprobar que no hay cuentas sin contraseñas la salida de este comando debe estar vacía:

  #awk -F: '($2 == "") {print}' /etc/shadow

- **Verificar que todos los hashes de todas las contraseñas están "Shadowed":** Para comprobar que ninguna hash de ninguna contraseña se guarda en */etc/passwd*, el siguiente comando no debería devolver ninguna salida:

  # awk -F: '($2 != "x") {print}' /etc/passwd

- **Verificar que no hay ninguna cuenta que no sea root que tenga UID 0:** La salida de este comando debería ser root, la existencia de cualquier otra cuenta, debe estar justificada.

  # awk -F: '($3 == "0") {print}' /etc/passwd

- **Comprobar que existe una políticas de contraseñas:** Y que esta se alinea con unas recomendaciones mínimas de seguridad en cuanto a longitud, caducidad y complejidad de la contraseña.
  En sistemas Linux, la caducidad se puede establecer en el archivo */etc/login.defs:*

  PASS_MAX_DAYS 60
  PASS_MIN_DAYS 7
  PASS_MIN_LEN 10
  PASS_WARN_AGE 7

  Así mismo, si necesitamos cambiar las políticas de seguridad para algún usuario en particular:

  #chage -W <DÍAS_MAX> -m <DÍAS_MIN> -W <DÍAS_AVISO> <usuario>

  En sistemas Windows, se recomienda activar o modificar la directiva de contraseñas para el sistema o para el dominio.

- **Comprobar la existencia de contraseñas débiles**: Se recomienda que se realicen comprobaciones periódicas de la fortaleza de las contraseñas del sistema, especialmente en ataques por diccionario. Para esta tarea se puede usar la conocida herramienta *Jhon the ripper* o, para sistemas Windows, la herramienta *Microsoft Baseline Security Analyzer (MBSA)*.

7.2.2 Protección de cuentas mediante PAM

Para establecer criterios comunes se seguridad a todas las contraseñas, se usa comúnmente PAM.

PAM (Pluggable Authentication Module) no es un modelo de autenticación en sí, sino que se trata de un mecanismo que proporciona una interfaz entre las aplicaciones de usuario y operaciones de manejo de la autenticación de los mismos.

PAM proporciona un marco en el cual se permite, entre otras cosas, configurar ciertas políticas de cuentas de usuario, entre ellas:

- **Establecer requerimientos de seguridad de las contraseñas**

Por defecto, el módulo *pam_cracklib* proporciona comprobaciones de fortaleza de contraseñas. Éste, realiza una serie de verificaciones, como pueden ser, asegurarse de que las contraseñas no son parecidas a palabras del diccionarios, que tienen al menos cierta longitud, que no es ninguna modificación de la contraseña anterior e incluso comprobar que tiene cierta clase de caracteres determinados.

El módulo *pam_passwdqc* ofrece la posibilidad de imponer incluso requerimientos más exigentes a las comprobaciones de contraseña.

Un ejemplo de configuración usando *pam_cracklib* (archivo de configuración /etc/pam.d/system-auth)*:*

password required pam_cracklib.so try_first_pass retry=3 minlen=14 dcredit=-1 ucredit=-1 \ocredit=-1 lcredit=-1

 # Exige al menos una mayúscula, una minúscula, un dígito, un carácter especial y que tenga, al menos una longitud minima de 14 caracteres.

- **Establecer el bloqueo de la cuenta después de un cierto número de intentos fallidos**

El modulo *pam_tally2* ofrece la posibilidad de bloquear una cuenta de usuario tras un número determinado de intentos fallidos.

Por ejemplo, para bloquear durante 900 segundos tras 5 fallos, editar el archivo */etc/pam.d/system-auth* , y en la sección *[auth]* introducir:

auth required pam_tally2.so deny=5 onerr=fail unlock_time=900

y el la sección [accounts]:

account required pam_tally2.so

- **Aumentar el Algoritmo de Hash a SHA-512**

Para usar SHA-512 es necesario realizar varios cambios. En el archivo */etc/pam.d/system-auth*, introducir o corregir la siguiente línea:

password sufficient pam_unix.so sha512 shadow nullok try_first_pass use_authtok

En /etc/login.defs:

MD5_CRYPT_ENAB no
ENCRYPT_METHOD SHA512

Y finalmente, en /etc/libuser.conf:

crypt_style = sha512

- **Limitar la reutilización de contraseñas**

Para evitar que los usuarios usen contraseñas que hayan utilizado recientemente (las 5 últimas contraseñas) es necesario introducir la siguiente línea en el archivo */etc/pam.d/system-auth*

password sufficient pam_unix.so existing_options remember=5

7.3 Límite de recursos

Los usuarios de cualquier sistema requieren recursos como pueden ser CPU, memoria o disco para realizar su trabajo, sin embargo, si se trata de servidores compartidos o públicos, **es necesario limitar los diferentes recursos que disponemos para que ningún usuario los acapare** y lo pueda dejar inutilizado. Habitualmente se utilizan tres métodos para realizar esta tarea:

7.3.1 PAM

Como hemos comentado anteriormente, PAM provee una interfaz estándar para realizar distintas operaciones de autenticación en un sistema, entre ellas, también permite la asignación de límite de recursos a los usuarios. PAM permite aplicar, de una manera centralizada, limites globales, de grupo o a usuarios individuales.

Para usar el límite de recursos con PAM es necesario añadir la siguiente línea al archivo de configuración de PAM adecuado (Por ejemplo: /etc/pam.d/sshd):

session required /lib/security/pam_limits.so

La configuración de los limites se define editando el archivo */etc/security/limits.conf.*
Las distintas opciones de límites que se nos permite configurar son:

- core – Limita el tamaño de los archivos "core" (KB). Se suele establecer a 0 para evitar volcados de núcleo.
- data – Limita el tamaño máximo de los datos (KB).
- fsize – Limita el tamaño máximo de los archivos (KB).
- nofile – Limita el numero máximo de archivos abiertos.
- stack – Limita el tamaño máximo del tamaño de la pila (KB).
- cpu – Limite de CPU.
- nproc – Limita el numero máximo de procesos.
- as – Límite del espacio de direcciones.
- maxlogins – Limita el numero de intentos de acceso para el usuario o grupo.
- priority – Establece la prioridad con la que el usuario ejecuta procesos.

Los límites pueden aplicarse a usuario por su nombre, por grupos usando la sintaxis "@group" o de forma global usando asterisco "*":

```
*           soft    core          0
*           hard    rss           10000
@student      hard    nproc         20
@faculty      soft    nproc         20
@faculty      hard    nproc         50
ftp           hard    nproc         0
@student      -       maxlogins     4
```

7.3.2 Ulimit

La misma línea de comando de Linux nos provee de una solución para establecer los límites de los recursos del sistema, el comando *ulimit* proporciona controles sobre los recursos disponibles para los terminales de consola y los procesos comenzados por ellos. Estos límites son necesarios establecerlos bien de manera global en */etc/profile* o de manera individualizada en el perfil de usuario *.bash_profile* (teniendo en cuenta que no pueda eliminar o editar dicho archivo). Algunas opciones del comando son:

-m : Tamaño máximo de memoria RAM

-v : Tamaño máximo del fichero de intercambio (Swap)
-s : Tamaño máximo de las pilas
-c : Tamaño máximo de los fichero core
-a :Mostrar límites establecidos
-S Usa el limite de recursos "suave"
-H Usa el limite de recursos "duro

Por ejemplo, si se quisiera limitar el tamaño de los archivos "core":

ulimit -Hc 0

7.3.3 QUOTA

Quota permite establecer limites en el uso de disco para usuarios y grupos. Es el método usado comúnmente para evitar que los usuarios acaparen el espacio en disco que es compartido con otros usuarios. Para su configuración es necesario editar el archivo */etc/fstab* y añadir *"grpquota"* or *"usrquota"* a los parámetros, por ejemplo:

/dev/hd5 /home ext3 defaults,errors=remount-ro,usrquota,grpquota 0 1

Para cada sistema de fichero en el que se haya instalado quota, se deben crear dos archivos que deben ser accesibles exclusivamente por el usuario administrador:

touch /home/quota.user
touch /home/quota.group

Tras el reinicio del sistemas, se habrán habilitado las cuotas en usuarios y grupos. Habitualmente, sólo será necesario establecer limites de espacio en aquellos sistemas de archivos que permitan la escritura a los usuarios (/home, /tmp, /var/tmp...).

Para el establecimiento y manejo de las distintas cuotas de usuario, se usará el comando edquota:

edquota -u usuario

Con ello se nos mostrará un archivo con diferentes campos:

- Filesystem: El sistema de archivos en el que se aplica la cuota.
- blocks: El número de bloques ocupados por el usuario.
- soft: El número de KB máximo a ocupar para cuota flexible. 0 = ilimitado
- hard: El número de KB máximo a ocupar para quota rígida. 0 = ilimitado
- inodes: El número de inodos máximo (ficheros y directorios). 0 = ilimitado

8 SISTEMA OPERATIVO

8.1 Procesos de arranque

La protección con contraseñas para el BIOS (o equivalentes al BIOS) y el gestor de arranque, pueden ayudar a prevenir que usuarios no autorizados que tengan acceso físico a sus sistemas, arranquen desde medios removibles u obtengan acceso como root a través del modo monousuario. Pero las medidas de seguridad que se deberían tomar para protegerse contra tales ataques dependen tanto de la confidencialidad de la información que las estaciones tengan como de la ubicación de la máquina.

Sin entrar en como establecer una contraseña en la BIOS del sistema, vamos a centrarnos en proteger el gestor de arranque de los sistemas Linux (Grub). Las principales razones por las que conviene asegurar dicho gestor de arranque son:

- **Previene el acceso en modo monousuario**: Si un atacante puede arrancar en modo monousuario, se convierte en el superusuario de forma automática sin que se le solicite la contraseña de acceso.

- **Previene el acceso a la consola de GRUB:** Un atacante puede usar la interfaz del editor para cambiar su configuración o para reunir información del sistema.

- **Previene el acceso a sistemas operativos inseguros**: Si es un sistema de arranque dual, un atacante puede seleccionar un sistema operativo diferente en el momento de arranque, el cual ignora los controles de acceso y los permisos de archivos del sistema que queramos proteger

Se describen algunas medidas de protección del gestor de arranque:

8.1.1 Protección con contraseña

Se recomienda proteger el acceso a GRUB con contraseña, para ello es necesario crear una contraseña en formato MD5 mediante el comando:

 # grub-md5-crypt

Que nos solicitaría una contraseña y nos la mostraría en formato MD5:

Password:<ENTER-YOUR-PASSWORD>
Retype password:<ENTER-YOUR-PASSWORD>
1NYoR71$Sgv6pxQ6LG4GXpfihIJyL0

Es necesario copiar la salida en formato MD5 al archivo de configuracion de GRUB: */boot/grub/grub.conf* (o */boot/grub/menu.lst*, dependiendo de la distribución):

```
default       0
timeout       5
password  -md5 $1$NYoR71$Sgv6pxQ6LG4GXpfihIJyL0
title         Debian GNU/Linux, kernel 2.6.13.4-cust-en-smp
root          hd0,0)
kernel    /boot/vmlinuz root=/dev/hda3 ro
savedefault
boot
```

8.1.2 Ocultar el gestor de arranque

Para ocultar GRUB es necesario añadir o descomentar la linea en el archivo de configuración:

hiddenmenu

8.1.3 Bloquear el acceso a otros Sistemas Operativos o modos de arranque

Para evitar que se pueda acceder a otros sistemas operativos o modos desde el menú de arranque tan sólo es necesario añadir el parámetro "*lock*" tras el título correspondiente que queramos bloquear:

```
 title   Windows NT/2000/XP
lock
root   (hd0,1)
```

Esto es especialmente interesante para bloquear modos como "*recovery*" que suelen permitir el acceso como root sin contraseña.

8.2 Sistemas de archivos

Tradicionalmente, la seguridad en sistemas Unix/Linux se ha basado en los permisos de archivos y directorios para evitar la lectura o modificación de archivos por parte de usuarios que no deberían tener permiso a ellos. Rigiéndonos por el principio de menor privilegio, es necesario configurar cada archivo, directorio y sistema de fichero para permitir sólo el acceso mínimo necesario para que pueda cumplir su propósito.

Sin embargo, debido a la gran cantidad de archivos que puede llegar a tener un sistema, es casi imposible asegurarse de que cada archivo en la máquina tiene sólo los permisos que necesita. Vamos a presentar una serie de restricciones y acciones que son casi siempre apropiadas para mejorar la seguridad del sistema:

8.2.1 Restringir las opciones de montaje de particiones

En sistemas Linux, las particiones pueden montarse con ciertas opciones que limitan las acciones que pueden realizar los archivos de las mismas, estas opciones se establecen en */etc/fstab/* y pueden ser usadas para poner las cosas difíciles a quien intente hacer acciones maliciosas en el sistema, alguna de estas buenas prácticas son:

- Añadir la opción "*nodev*" a las particiones locales no raíces (que no sean "/").
- Añadir la opción "*nodev*", "*nosuid*" y "*noexec*" a las particiones de almacenamiento extraibles.
- Añadir la opción "*nodev*", "*nosuid*" y "*noexec*" a las particiones de almacenamiento temporales.
- Añadir la opción "*nodev*", "*nosuid*" y "*noexec*" a */tmp/.*
- Añadir la opción "*nodev*", "*nosuid*" y "*noexec*" a */dev/shm.*

8.2.2 Restringir el montaje y desmontaje automático de unidades

Por defecto, los sistemas Linux incluyen una gran cantidad de módulos que permiten añadir y descartar sistemas de ficheros de una manera cómoda, sin embargo, esto tiene un riesgo: Podría permitir a usuarios sin privilegios introducir y montar cualquier tipo de sistema de ficheros en el sistema o que un atacante pudiera comprometer un sistema a través de algún defecto en el montaje automático en la instalación de un programa. Por todo ello se recomienda:

- **Deshabilitar el soporte para dispositivos USB**: Las memorias Flash o los discos duros externos pueden permitir a un atacante con acceso físico al sistema copiar una gran cantidad de datos.

- **Deshabilitar la carga de los módulos del driver de almacenamiento USB**: En sistemas Debian esto se realiza añadiendo al archivo */etc/modprobe.d/blacklist.conf* el módulo de almacenamiento USB (habitualmente *usb_storage*). En otros sistemas como Red Hat hay que añadir la línea *install usb-storage /bin/true* al archivo */etc/modprobe.conf* para evitar que se cargue dicho módulo.

- **Eliminar el driver de almacenamiento USB:** Si el sistema nunca va a hacer uso de este tipo de dispositivos es siempre más efectivo eliminar el archivo del módulo, evitando que esté disponible.

- **Deshabilitar el soporte USB del Kernel mediante la configuración del gestor de arranque**: Otra manera de deshabilitar los dispositivos USB es impidiendo que se carguen los módulos al inicio introduciendo la palabra "nousb" en la linea correspondiente a la carga del sistema en el gestor de arranque (habitualmente /etc/grub.conf), por ejemplo:

  kernel /vmlinuz-version ro vga=ext root=/dev/VolGroup00/LogVol00 rhgb quiet nousb

- **Deshabilitar la posibilidad de arrancar desde dispositivos USB**: Modificando la configuración de la BIOS correspondiente.

- **Deshabilitar el montaje automático de unidades si es posible**:

  # chkconfig autofs off

- **En sistemas Windows, es importante deshabilitar la ejecución automática (autorun)** en archivos extraibles.

8.2.3 Comprobaciones de permisos en archivos y directorios

Es necesario comprobar que los archivos deben tener establecidos los permisos que les corresponden siguiendo la política de seguridad definida. Se va a estudiar los archivos más importantes y en los que se deben comprobar periódicamente que no existen discrepancias en cuanto a sus permisos.

- **Verificar los permisos en los archivos:** passwd, shadow, group and gshadow.

- **Verificar que todos directorios con permisos globales de escrituras tienen establecido el bit "sticky"**: Cuando el denominado bit "sticky" se encuentra activado en un directorio, tan sólo el propietario de dicho directorio puede eliminar archivos de él, por lo que es necesario evitar que para directorios visibles para todos los usuarios, se permita eliminar los archivos de manera arbitraria. Para comprobar esto ejecute:

  # find <partición> -xdev -type d \(-perm -0002 -a ! -perm -1000 \) -print

  Si este comando produce algún tipo de salida, para cada directorio listado hay que ejecutar:

  # chmod +t /dir

- **Encontrar archivos con permisos globales de escritura no autorizados:** Los datos en los archivos con permisos globales de escritura pueden ser modificados por cualquier usuario del sistema. En la gran mayoría de las circunstancias, los archivos pueden ser configurados usando combinaciones de permisos de grupo y usuarios que permitan el acceso legitimo cuando sea

necesario sin necesidad de correr el riesgo de activar la escritura global. Para encontrar todos los archivos que sean modificables globalmente:

 # find <partición> -xdev -type f -perm -0002 -print

Si este comando produce algún tipo de salida, para cada archivo listado es necesario ejecutar:

#chmod o-w file

- **Encontrar archivos sin dueño**: A pesar de que los archivos sin dueño no son directamente explotables, puede ser consecuencia de:
  - Problema con procesos del sistema.
  - Una creación realizada por un intruso.
  - Problemas en la instalación/desinstalación de algún programa.
  - Fallos a la hora de eliminar permisos de una cuenta de usuario borrada.

  Para solucionar este problema, no basta con borrar o asignar el permiso correctamente, es necesario un estudio pormenorizado sobre el motivo que ha llevado a que dicho archivo no tenga propietario. Para encontrar en el sistema archivos que no posean un dueño, se podría ejecutar el siguiente comando:

  # find <partición> -xdev \(-nouser -o -nogroup \) -print:

- **Verificar que todos los directorios con permisos globales de escritura tienen su propietario correcto:** Encontrar directorios en cualquier partición local y asegurarse de que solo root u otras cuentas del sistema tienen permisos globales de escritura. Para localizarlos se podría ejecutar:

  # find <partición> -xdev -type d -perm -0002 -uid +500 -print

8.2.4 Minimizar el uso de SUID root

Los bits de *SUID* y *SGID* son usados normalmente por comandos y demonios que los usuarios normales deberían ejecutar pero que también necesitan permisos para acceder a otras partes del sistema donde normalmente no tendrían permiso. Para ello los permisos de un archivo incluyen el bit de ID de usuario o de grupo indicado por una "s" donde normalmente debería haber una "x":

-rwsr-xr-x 1 root root 15000 feb 13 2012 <file>

En la práctica, esto hace que el comando no se ejecute con el nivel de privilegios del usuario que lo inicia, si no con el nivel del propio comando. Si el ID de usuario o grupo es cero (root) el comando se ejecutará con privilegios de superusuario sin importar quién lo esté ejecutando.

Habitualmente, cuando se ejecuta un comando, esto se hace con los privilegios del usuario que lo ejecuta. Por ejemplo, si un usuario <user> intenta ejecutar el comando ls /root/ al que no tiene permisos, el sistema comprobará que el usuario lo esta intentando. No hay necesidad de resaltar lo peligroso que puede llegar a ser esto, por lo que se hace necesario saber qué aplicaciones ejecutan SUID/SGID y determinar si realmente es necesario que tengan este tipo de permisos. Para encontrarlo, ejecutaremos:

find <partición> -perm +4000 -user root -type f -print
find <partición> -perm +2000 -group root -type f -print

Si determina que un archivo localizado no necesita tener activado el bit de SUID/SGID, se puede desactivar de la siguiente manera:

chmod u-s <path> # Para usuario
chmod g-s <path> # Para grupo

8.2.5 Utilizar SUDO y SU

Muchos usuarios caen en el hábito de identificarse como root frecuentemente. Como norma general, **es una mala práctica acceder al sistema y mantenerse como root**, siendo incluso práctica habitual desactivar dicho usuario.

El programa *sudo* permite a los usuarios ejecutar programas con los privilegios de seguridad de otro usuario de manera segura, de acuerdo a como se especifique en el archivo */etc/sudoers*, donde se determina quien está autorizado a hacer qué.

De la misma manera, aunque menos cómoda, el comando SU nos permite ejecutar tareas de root sin necesidad de tener que cambiar de usuario. Mediante la opcion *"-c <comando>*" nos permite escalar privilegios cuando sea necesario.

$su -c <comando_privilegiado> #Nos permite lanzar el comando con permiso de root

8.2.6 Programas de comprobación de Integridad (Tripwire, aide, etc)

Los programas de comprobación de integridad se encargan de monitorizar y alertar de cambios específicos de ficheros en un rango de sistemas. Funciona cotejando los archivos y directorios con una base de datos de la ubicación y las fechas en que fueron modificados, además de otra serie de datos. La base de datos se genera tomando una instantánea en el momento de su instalación y se accede a ella mediante contraseña cifrada, por lo que su instalación en un sistema posiblemente infectado, carecería de

efectividad y se recomienda que su instalación y configuración sea hecha antes de haber conectado el computador por primera vez a internet.

Los programas de comprobación de integridad más usados son: *Tripwire, AIDE y samhain.*

8.2.7 Gestión de copias de seguridad

La realización y gestión de las copias de seguridad es un **elemento básico en cualquier política de seguridad**. Los soportes en los que se realicen (cartuchos, cintas, disquetes, discos, CDS, DVD´s,...) deben identificar el tipo de información que contienen, así mismo, los soportes tienen que ser inventariados, y almacenados con acceso restringido.

Las copias de respaldo, de seguridad, o backups tienen que garantizar la reconstrucción de los datos en el estado en que se encontraban en el momento de producirse la pérdida o destrucción. Parte de esta reconstrucción se puede basar en la nueva introducción de los datos perdidos, a partir de los mismos datos en otros soportes.

8.3 Asegurando el Kernel del sistema

El kernel de Linux permite establecer opciones avanzadas de la pila TCP/IP y de la memoria virtual para mejorar la seguridad, el rendimiento del sistema y prevenir ciertos tipos de ataque.

8.3.1 Sysctl

Sysctl es una interfaz que te permite realizar cambios a un kernel de Linux en ejecución. Modificando la configuración en el archivo /etc/sysctl.conf es posible configurar distintos parámetros, tanto de red como de sistema, como por ejemplo:

- Limitar la configuración de red transmitida para IPv4 y Ipv6.
- Activar la protección execshield.
- Prevenir ataques de denegación de servicio.
- Activar la verificación de IP de origen.
- Prevenir el uso de IP spoofing.
- Registrar varios tipos de paquetes de red sospechosos.

Algunas configuraciones de los parámetros más importantes a tener en cuenta:

- **Activar la protección contra IP spoofing**

sysctl -w net.ipv4.conf.default.rp_filter=1
sysctl -w net.ipv4.conf.all.rp_filter=1

- **Desactivar el reenvío de paquetes IP**

sysctl -w net.ipv4.ip_forward=0
sysctl -w net.ipv6.ip_forward=0

- **Ignorar Broadcasts Request:** Cuando un equipo envía un paquete a una dirección de Broadcast, ese paquete es enviado a todas las máquinas de la red, las cuales a su vez, responden a esta petición pudiendo llegar a colapsar la red. Desactivando este tipo de peticiones, mitigaremos ciertos tipos de ataques por denegación de servicio.

sysctl -w net.ipv4.icmp_echo_ignore_broadcasts=1

- **Activar la protección contra mensajes de error mal construidos**

sysctl -w net.ipv4.icmp_ignore_bogus_error_responses=1

- **Desactivar ICMP Redirect Acceptance:** Este tipo de paquetes son usados para indicar al sistema que existe una ruta mejor para llegar a otras redes, un atacante podría falsificar este tipo de envíos para redireccionar tráfico a su antojo o para realizar una ataque por denegación de servicio.

sysctl -w net/ipv4/conf/all/accept_redirects=0
sysctl -w net/ipv4/conf/all/send_redirects=0
sysctl -w net/ipv4/conf/all/secure_redirects=1

- **Desactivar IP Sourcing routing:** Es una técnica mediante la cual quien envía un paquete puede especificar la ruta que un paquete debería seguir a través de la red. Un atacante podría usar este método para encaminar paquetes a partes de la red donde no debiera.

sysctl -w net.ipv4.conf.all.accept_source_route=0

- **Registrar paquetes:** Spoofed Packets, Source Routed Packets, Redirect Packets

sysctl -w net.ipv4.conf.all.log_martians=1

- **Ignorar ICMP Requests PINGs:** Si queremos desactivar completamente que el sistema responda a pings, aunque también es recomendable denegar esta acción a través de reglas de IPTables.

sysctl -w net.ipv4.icmp_echo_ignore_all=1

Otra opción a tener en cuenta es la instalación de conjuntos de modificaciones adicionales y utilidades de seguridad para el Kernel como pueden ser *SELinux*, *AppArmor* o *GRSecurity*.

8.3.2 SELinux

SELinux es una implementación del *MAC* (Mandatory Acces Control). Se basa en la existencia de un conjunto de reglas de autorización o políticas las cuales determinan si una operación sobre un objeto realizada por un sujeto esta o no permitida basándose en los atributos de ambos, funciona a nivel de kernel, lo que implica que las aplicaciones no necesitan ser modificadas para aprovechar toda su potencia, ya que para ellas SELinux es transparente, algo muy importante a la hora de que el sistema sea desplegado en un entorno real.

La razón de usar SELinux es **limitar el acceso que tienen las aplicaciones a otras aplicaciones y a los ficheros**, impidiendo que un proceso pueda modificar cualquier fichero del usuario con el que se lanzó. Para realizar esta labor, extiende los atributos del sistema de fichero indicando el tipo de usuario, el rol y el tipo de objeto. Estos atributos son los que en función de las políticas definidas en SELinux, indican las interacciones entre ellos y los diferentes objetos del sistema.

8.3.3 APPArmor

APPArmor fue creado como alternativa a SELinux que era criticado por los administradores por ser demasiado difícil de instalar y mantener. Al contrario que SELinux, que se basa en añadir etiquetas a los archivos, APPArmor trabaja con las rutas de los ficheros.

Al igual que SELinux, provee MAC y tiene la característica de incluir un modo de aprendizaje, en el cual las violaciones a los perfiles son registradas en un principio, pero no prevenidas con el objetivo de estudiarlas y convertirlas en un perfil basado en el comportamiento típico del programa. AppArmor está diseñado para ayudar a los administradores a montar una trampa. El sistema monitoriza la forma en que los procesos acceden a los ficheros, distinguiendo entre accesos de lectura y escritura, así como el uso del privilegio de root.

Hasta la fecha no hay ninguna evidencia definitiva que haga a uno de los dos sistemas preferible al otro. La discusión sobre las ventajas y desventajas de cada método suele girar en torno a cual de los dos se alinea más con los mecanismos de control de UNIX/Linux y con la facilidad de implementación y administración.

8.3.4 GrSecurity

Grsecurity es una solución de seguridad a modo de parche de kernel que permite establecer múltiples comprobaciones, verificaciones y controles de una forma activa en nuestro sistema.

Entre sus controles de seguridad implementados en el sistema destacan:

- La protección a nivel de funcionamiento del kernel.
- Prevención de la ejecución del código arbitrario.
- Control de ejecución de las tareas en el stack.
- Restricción que permite que un usuario visión solamente su procesos.
- Control de las actividades de los usuarios.
- Permisos de ejecución en determinadas áreas del sistema.
- Implementación de controles adicionales a la seguridad impuesta por chroot.
- Alarmas e intervenciones de seguridad que contienen el IP del que causa la alarma.
- Implementación de un control de acceso basado en roles (RBAC).

GRSecurity permite configurar distintos niveles de seguridad dependiendo de nuestras necesidades, hay que indicar que a mayor nivel de protección es posible que se incremente también el número de incompatibilidades con software poco habitual que se encuentre en el sistema.

El nivel de protección más elevado activa el sistema PaX que implementa protecciones del mínimo privilegio para las páginas de memoria. La aproximación del mínimo privilegio permite a los programas realizar lo que tienen que hacer de modo que sean capaces de ejecutarlo adecuadamente, y no poder realizar nada más, lo cual previene eficazmente muchos de los exploits, como algunos tipos de desbordamientos de buffers.

8.4 Trazabilidad, monitorización y gestión de logs (Logrotate, syslog, rsyslog) Correlación de logs

Como en muchos sistemas de producción, la trazabilidad de los sucesos que ocurren en los distintos sistemas es importante para poder desarrollar metodologías que permitan a las organizaciones estudiar qué ha sucedido, cómo ha ocurrido, establecer un patrón temporal de posibles anomalias y enfrentarse de una manera rápida y eficiente a posibles problemas que puedan ocurrir en los sistemas.

Cualquier sistema Linux protegido debe ir acompañado de detallados, completos y cuidadosos logs. Los logs tienen varios propósitos:

- Ayudan a solucionar problemas de sistemas y aplicaciones.
- Proporcionan señales de aviso ante posibles abusos sobre el sistema.
- Tras producirse un compromiso o caída del sistema, permiten obtener datos forenses cruciales.

8.4.1 Syslog

Es la herramienta principal de registro de eventos dentro de los sistemas Unix. Acepta datos de logs del núcleo (por medio de klogd), desde todos los procesos locales e incluso de sistemas remotos. Es flexible, permitiendo determinar desde donde obtener los logs y donde dejarlos. Sin embargo, es necesario configurarlo y personalizarlo correctamente para registrar lo que para nosotros sea importante y descartar el resto.

Cuando Syslogd recibe un mensaje de log, según en el tipo de mensaje o servicio y su prioridad y lo especificado en el archivo de configuración */etc/syslog.conf* realizará un tipo de acción u otra. Cada línea de este archivo especifica uno o más selectores de servicios/prioridades seguidos por una acción.

Por ejemplo:

#facility.priority acction
mail.info /var/log/mail.info

Indicaría que los mensajes de prioridad info o superior del sistema de mail se guarden en el fichero */var/log/mail.info.*

Existen distintas **fuentes de eventos o servicios** que vuelcan los datos:

- **auth**: Sistema de autenticación, incluyendo login, su y getty.
- **authpriv**: Programas de autorización privados.
- **cron**: cron y at.
- **daemon**: Demonios del sistema.
- **ftp**: Demonio de ftp.
- **kern**: Mensajes generados por el kernel.
- **local0-7**: Reservado para uso local.
- **lpr**: Sistema de impresión.

- **mail**: Sistema de mail.
- **mark**: Mensajes internos de syslogd (no usable por aplicaciones).
- **news**: Sistema de news en la red.
- **syslog**: Mensajes del demonio de syslog.
- **user**: Aplicaciones de usuario.
- **uucp**: Sistema UUCP.
- ***:** Todas menos mark.

Mientras que las distintas **prioridades** (en orden ascendente) son:

- **debug:** Mensajes generados en la depuración de un programa.
- **info:** Mensajes informativos.
- **notice:** Mensajes no críticos que requieren investigación adicional.
- **warning:** Mensajes de aviso.
- **err:** Otros errores.
- **crit:** Condiciones críticas como errores hardware.
- **alert:** Condiciones que se deberían resolver rápidamente.
- **emerg:** Condiciones de pánico muy importantes.

Finalmente, las **acciones** que permite definir por defecto syslog:

- **n_fichero:** Ruta absoluta al fichero donde se guardan los mensajes.
- **n_terminal:** Escribe los mensajes en el terminal.
- **@host o @ip:** Host remoto donde su syslogd recoge los mensajes.
- **user1, user2:** Escribe los mensajes a los usuarios si están conectados.
- ***:** Todos los usuarios conectados reciben los mensajes.

Por defecto, syslogd no acepta mensajes de otros hosts, teniendo que iniciarlo con la opción -r. Sin embargo para el envío de mensajes de syslog se recomienda el uso de *Remote Syslog* (*rsyslog*).

8.4.2 Rsyslog

Es un programa de registro de mensajes, implementa el protocolo básico de syslog y lo extiende agregando filtros, con una configuración flexible. Tiene la capacidad de reenviar via UDP o TCP los mensajes del log a otra maquina y una gestión más eficiente en las colas de recepción y envío. En varias distribuciones sustituye a syslog como programa de registro de eventos por defecto en el sistema.

El envío de los registros del sistema a un servidor centralizado de eventos es recomendable por distintos motivos:

- Se evita la posible manipulación de los eventos registrados en caso de que un atacante consiga acceso al sistema.
- Se gana en comodidad en el análisis de eventos.
- Permite correlación de eventos de manera centralizada.

La configuración de rsyslog es similar a la de syslog, sólo que permite incluir otro tipo de parámetros relacionados con el envío y recepción de mensajes:

```
$WorkDirectory /var/spool/rsyslog/work     # default location for work (spool) files
$ActionQueueType LinkedList              # use asynchronous processing
$ActionQueueFileName srvrfwd              # set file name, also enables disk mode
$ActionResumeRetryCount -1               # infinite retries on insert failure
$ActionQueueSaveOnShutdown on            # save in-memory data if rsyslog shuts down
*.*       @@SERVER_REMOTO
```

8.4.3 Logrotate

Configurar la rotación de archivos de mensajes del sistema es necesario para evitar que los ficheros de log de los distintos servicios (web,email,ftp,etc) ocupen una gran cantidad de espacio en disco o generen ficheros de gran tamaño.

La configuración de logrotate se encuentra en el fichero */etc/logrotate.conf,* en el cual podemos especificar directivas generales para la rotación de logs.

Un archivo de configuración de logrotate, consiste en una serie de especificaciones para los grupos de archivos de log que vamos a administrar. Las opciones especificadas fuera de cada contexto de un log concreto, (errors, rotate, weekly...) se aplican a todos ellos, pero pueden ser reemplazadas con una especificación concreta para un log en particular. Se recomienda no modificar la configuración por defecto, ya que esto puede afectar a los logs del sistema completo. En lugar de modificar el fichero *logrotate.conf*, existe un directorio donde existen configuraciones específicas para cada servicio: /etc/logrotate.d/

8.4.4 Monitorización y correlación de eventos

El tener configurado el sistema de eventos de manera centralizada tiene varias ventajas, entre ellas, se nos permite tener ordenados cronológicamente los eventos producidos por un mismo ataque al atravesar los diversos equipos.

La principal ventaja de este tipo de solución de centralización está bien clara, el operador de red que monitorice diferentes consolas de logs, ahora tendrá en un único punto toda la información relevante en orden cronológico. La eficiencia aumenta, por tanto el tiempo de reacción ante ataques también.

Una vuelta de tuerca más nos lleva a las soluciones que además de consolidar esos eventos y ponerlos a disposición de un operador, son capaces de elaborar alertas más elaboradas, basadas en los eventos recibidos de diversas fuentes de información como pueden ser eventos generados por firewalls, IDS o IPS's y correlarlas según ciertos criterios y parámetros, con lo que es posible identificar patrones de ataques complejos de una manera simple y automática.

9 CONCLUSIONES

Con la publicación de esta segunda parte del informe de bastionado se ha querido representar la parte más técnica de los distintos puntos comentados en el comunicado anterior. Se han expuesto una selección de comandos y configuraciones generales en todos los ámbitos tratados, con el objetivo de poder tener unas directrices básicas a la hora de querer enfrentarnos al aseguramiento de un sistema.

Hay que tener presente que bastionar un sistema va más allá de seguir una serie de guiones con comandos y configuraciones; cada equipo es distinto, por lo que el bastionado debe adaptarse, integrarse y evolucionar en un proceso de mejora continua dentro de la misma administración de los sistemas, que permita mitigar los posibles incidentes de seguridad relativos a los fallos de configuración de los mismos.

10 REFERENCIAS

- CERT-IF-2045-220812-Bastionado de Sistemas I.
- Guide to the Secure Configuration of Red Hat Enterprise Linux 5
- AppArmor
- Linux Kernel /etc/sysctl.conf Security Hardening
- PaX
- GrSecurity
- SecureApt
- Kernel Hardening
- GrSecurity and PaX configuration options