Seguridad básica en servidores Linux

Jul 5 2010 in Artículos *por RevistaLiNUX.net*

Por Adolfo Lozano Tello, Marcos Blanco Galán

Por su naturaleza, GNU/Linux es considerado un sistema operativo robusto al incorporar características comunes de UNIX relacionadas con la seguridad. A pesar de ello, el administrador de sistemas sigue jugando un papel fundamental en este aspecto, especialmente, cuando hay involucrados servidores que ofrecen servicios a miles de usuarios.

Dependiendo de las fuentes de amenaza, el concepto de seguridad informática puede dividirse, principalmente, en seguridad física y seguridad lógica. Esta guía se centra exclusivamente en la seguridad lógica, esto es, en cómo asegurar el acceso a los recursos de la máquina y garantizar la integridad del sistema ante posibles ataques. Los puntos que se van a tratar en este documento se centran en aspectos técnicos muy concretos. Al final de la guía se proporcionan una serie de referencias para que el lector pueda profundizar en los temas que estime conveniente.

Permisos de ficheros y atributos

Establecer correctamente los permisos y atributos de los ficheros del sistema es crucial para mantener su integridad. Regularmente, es conveniente llevar a cabo auditorías en busca de permisos que no deberían estar autorizados o atributos inapropiados. Algunos de los aspectos que deberían analizarse para tomar las medidas oportunas, son los siguientes:

- Los bits *SETUID* o *SETGID* en ficheros ejecutables permiten que un usuario sin privilegios de administración pueda llevar a cabo acciones que únicamente podría realizar el usuario *root*. En concreto, la activación de estos bits hace que el *eUID* sea igual al *ownerUID* y que el *eGID* sea igual al *ownerGID*, respectivamente. Para localizar estos ficheros ejecutaremos: "*find / -perm /6000 -type f -ls*". Para suprimir los permisos *SETUID* y *SETGID* ejecutaremos "*chmod -s*" sobre los ficheros deseados. Los códigos de los permisos, en notación octal, del *SETUID* y el *SETGID* son 4000 y 2000, respectivamente.

- El llamado sticky bit (también conocido como bit de permanencia, bit pegajoso o modo t) impide que un fichero se elimine del área de swapping. Esto suele resultar útil en programas que son ejecutados frecuentemente por varios usuarios. Aplicado sobre un directorio, permite que únicamente el propietario del fichero, el propietario del directorio o el administrador puedan renombrar o eliminar los ficheros contenidos en él. El código de permiso en notación octal del sticky bit es 1000. Si el archivo posee permiso de ejecución, en la terna correspondiente al resto de usuarios (other) aparecerá una t en lugar de x; si no tiene permiso de ejecución se mostrará una T

- Los archivos *world-writable* (modificables por cualquier usuario) también pueden suponer un importante agujero de seguridad. Algunos de los ficheros propios de los sistemas basados en UNIX que nunca deberán ser *world-writable* son: /etc/passwd, /etc/shadow y /etc/group. Para listar todos los ficheros y directorios modificables por cualquier usuario ejecutaremos: "*find / -perm -o=w ! -type l -ls*". La salida habitual del comando anterior muestra entradas pertenecientes a los directorios /dev y /tmp.

- Un posible indicio de intrusiones en el sistema son los archivos que carecen de dueño o no pertenecen a ningún grupo. Para localizarlos realizaremos la siguiente búsqueda: "find / -nouser -o -nogroup".

- Los comandos lsattr y chattr permiten al administrador de sistemas cambiar

características de archivos y directorios, incluyendo la posibilidad de controlar el borrado y modificación por encima de las funciones que provee chmod. Los atributos “append-only” y “immutable” son particularmente efectivos a la hora de prevenir el borrado de archivos de log, o cuando algún tipo de software malicioso intenta sobrescribir o suplantar ficheros legítimos del sistema. El comando lsattr se emplea para listar estas propiedades, mientras que el comando chattr permite añadirlas (+) o eliminarlas (-) con los parámetros i (immutable) y a (append-only).

▪ Una estrategia adicional para mejorar la seguridad del sistema de ficheros es utilizar adecuadamente las opciones de mount, bien a través del propio comando mount desde la línea de comandos, o desde el fichero de configuración /etc/fstab. Algunas de las opciones más interesantes a la hora de montar una partición o sistema de ficheros son: nosuid (evita la asignación de los bits SETUID y SETGID), noexec (evita la ejecución de archivos binarios), nodev (evita la interpretación de los dispositivos especiales de bloque o de carácter) y ro (acceso de sólo lectura).

Integridad del sistema de archivos

Antes de habilitar por primera vez el acceso a redes de un servidor, es recomendable crear una base de datos que contenga un listado donde se identifiquen de forma inequívoca todos los ficheros existentes actualmente en el sistema. Un mecanismo de estas características garantizará el control ante posibles modificaciones no deseadas. En GNU/Linux existen herramientas que facilitan esta tarea, capaces de monitorizar y alertar al usuario ante cambios en el sistema de archivos basándose en la comprobación de hashes. Algunos ejemplos son Tripwire, AIDE o Afick, considerados sistemas IDS (Intrusion Detection System).

Contraseñas vulnerables

Según algunos estudios (Infosec Europe 2006) , más del 40% de las contraseñas elegidas por los usuarios no son lo suficientemente fuertes. Se suelen caracterizar por ser de longitud corta, contener palabras simples, el nombre de la máquina, nombres de usuario o combinaciones que pueden ser averiguadas rápidamente con diccionarios.

Todo administrador debería llevar a cabo periódicamente pruebas de vulnerabilidad sobre las contraseñas almacenadas en el sistema de ficheros. Para facilitar esta tarea, es posible recurrir a herramientas como John the Ripper o checkpasswd, capaces de realizar ataques por diccionario a ficheros de claves como /etc/shadow o /etc/gshadow.

La utilidad passwd, característica de los sistemas operativos tipo UNIX, proporciona mecanismos para evitar que los usuarios utilicen contraseñas fácilmente averiguables. En GNU/Linux, passwd está configurado para trabajar con la API Linux-PAM (Pluggable Authentication Modules). El módulo pam_cracklib.so, disponible a través del paquete libpam-cracklib, puede ayudar a prevenir estas contraseñas débiles añadiendo controles a la hora de cambiar la contraseña, realizando chequeos contra diccionarios o estableciendo periodos de expiración. La configuración de PAM se define en el directorio /etc/pam.d/. .

Módulos del kernel

Una de las reglas básicas a la hora de configurar el kernel es suprimir todo aquello que no se vaya a utilizar. Además de obtener un binario más reducido, se eliminan riesgos innecesarios ante posibles vulnerabilidades en los módulos o características del kernel. Llevado al extremo, ante una máquina con las funciones perfectamente definidas y hardware inalterable en el tiempo, probablemente, lo más recomendable sea utilizar un kernel monolítico. Los kernels monolíticos aportan mayor seguridad en servidores, ya que se evita la carga de módulos con funcionalidades que podrían comprometer la integridad del

sistema. La desactivación de módulos cargables, característica intrínseca de los kernels monolíticos, dificulta la instalación de algunos rootkits. La decisión de utilizar un kernel monolítico o modular dependerá, por lo tanto, del propósito y escenario concretos.

El pseudo-sistema de archivos /proc

A través del pseudosistema de archivos /proc también es posible modificar ciertos parámetros del kernel relacionados con la seguridad en tiempo de ejecución. El comando sysctl es usado para visualizar y modificar las configuraciones en /proc/sys/. Ejecutando "sysctl -a" se obtiene la lista con todas las variables del kernel configurables. La sintaxis para establecer el valor de un parámetro con sysctl es "sysctl -w kernel.PARÁMETRO=VALOR". Además de sysctl, también puede usarse el comando echo sobre el parámetro cuyo valor deseamos obtener o modificar. A continuación, se enumeran algunas de las principales opciones del kernel que el administrador debería tener presente.

- /proc/sys/proc/sys/net/ipv4/ip_forward: La activación de esta opción hace que un paquete con dirección IP de destino diferente de la local sea reenviado utilizando la tabla de enrutamiento. Esta opción debería desactivarse si tan sólo existe una conexión de red. Es recomendable activar o desactivar ip_forward antes que cualquier otra opción, ya que el evento también activa o desactiva otras opciones del kernel de forma automática.

- /proc/sys/net/ipv4/icmp_echo_ignore_all: Ignora las peticiones de ECHO ICMP. Activando esta opción se evita que el servidor responda a las peticiones de ping.

- /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts: Ignora las peticiones de ECHO ICMP con direcciones de destino broadcast (difusión) y multicast (multidifusión). Se trata de un mecanismo de prevención ante posibles intentos de utilizar la red local como arma en posibles ataques de denegación de servicio (Dennial of Service, DoS) mediante packet flooding a otros hosts.

- /proc/sys/net/ipv4/tcp_syncookies: Protección contra SYN packet flooding, uno de los ataques DoS más conocidos. SYN flooding aprovecha la mala implementación del protocolo TCP basándose en un "saludo" incompleto entre el equipo atacante y la víctima.

- /proc/sys/net/ipv4/conf/all/rp_filter: Prevención ante posibles ataques IP spoofing dentro de la red interna. Básicamente, consiste en ignorar los paquetes entrantes cuya dirección de origen no aparece en la tabla de routing de la interfaz de red receptora, es decir, tiene lugar una verificación de la dirección del origen.

- /proc/sys/net/ipv4/conf/all/secure_redirects: Únicamente se aceptarán mensajes de redirección ICMP para las puertas de enlace (gateways) de la lista de puertas de enlace por defecto.

- /proc/sys/net/ipv4/conf/all/accept_source_route: Deshabilita la opción de enviar paquetes enrutados desde el origen. El enrutamiento desde el origen es raramente utilizado con fines legítimos, por lo que resulta más seguro desactivarlo.

- /proc/sys/net/ipv4/conf/all/log_martians: La activación de esta opción permite registrar en el log del kernel los paquetes falsos (spoofed), enrutados en el origen y paquetes de redirección.

Todos los ajustes del kernel establecidos con el comando echo (ejemplo: /bin/echo "1" > /proc/sys/net/ipv4/conf/all/rp_filter) se perderán al reiniciar la máquina. Para hacerlos permanentes se recomienda añadirlos al archivo /etc/sysctl.conf, el cual se inicializa automáticamente a través del script de arranque /etc/init.d/bootmisc.

Control de Acceso Obligatorio (MAC)

Tradicionalmente, los sistemas de tipo UNIX han venido utilizando el llamado modelo de Control de Acceso Discrecional (Discretionary Access Control) o DAC, donde el usuario tiene completo control sobre los objetos que le pertenecen y los programas que ejecuta. En este modelo, el programa ejecutado también dispone de los mismos permisos que el usuario que lo está ejecutando. DAC implica que la seguridad del sistema depende de las aplicaciones que se están ejecutando y, por tanto, cuando se produce una vulnerabilidad en una aplicación, ésta afecta a todos los objetos a los que el usuario tiene acceso.
Una alternativa a DAC es el denominado Control de Acceso Obligatorio (Mandatory Access Control) o MAC, donde el administrador define una política de seguridad que los usuarios no pueden modificar. Esta política va más allá que el establecimiento de propietarios de archivos. DAC fija contextos donde se indica cuándo un objeto puede acceder a otro objeto. Este modelo de control de acceso puede mejorar el nivel de seguridad.
Security-Enhanced Linux o SELinux es un proyecto de la Agencia de Seguridad Nacional de los Estados Unidos (NSA) que puede ser considerado como una implementación práctica del modelo de seguridad de Control de Acceso Obligatorio en el kernel Linux. Su objetivo es forzar la ejecución de los procesos dentro de un entorno con el mínimo de privilegios necesarios. SELinux añade una serie de modificaciones en el kernel Linux a través de los denominados Linux Security Modules (LSM). Está inspirado originalmente en el Trusted Computer System Evaluation Criteria o TCSEC (1985) del Departamento de Defensa de los Estados Unidos (DoD). TCSEC también es ampliamente conocido como El Libro Naranja, una de las publicaciones Rainbow Series del DoD. En este documento se define la norma que establece los requisitos básicos para la evaluación de la eficacia de los controles de seguridad incorporados en un sistema informático. A día de hoy, TCSEC ya ha sido reemplazado por el estándar internacional Common Criteria for Information Technology Security Evaluation o simplemente Common Criteria (ISO/IEC 15408), publicado en 2005.

Hardening con ‘Bastille’

El proyecto Bastille, iniciado por Jay Beale y en el que también han contribuido grandes empresas como IBM y HP, consiste en una suite de seguridad creada originalmente para distribuciones basadas en Red Hat. Bastille agrupa un conjunto de herramientas cuya finalidad es reforzar la seguridad de los principales servicios del sistema, de ahí su denominación “hardening tool”. Actualmente, también se encuentra disponible en Ubuntu a través de los repositorios oficiales. Esta suite se compone de módulos, cada uno de ellos dirigidos a cubrir un aspecto concreto: permisos, cuentas de usuario, arranque del sistema, demonios y firewalling son sólo algunos ejemplos.

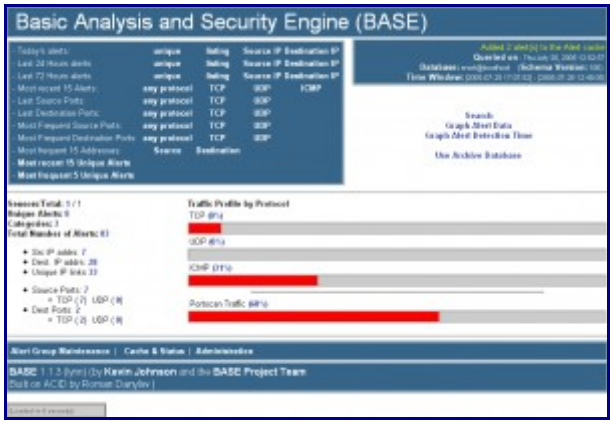
Bastille

Sistemas de Detección de Intrusos (IDS)

Los sistemas de detección de intrusos (IDS) facilitan al administrador de sistemas la detección de accesos no autorizados a un servidor o una red. En

muchas ocasiones, los intentos de acceso no autorizados son llevados a cabo por atacantes que utilizan herramientas automatizadas con distintos fines: inyección SQL, buffer overflows, port scanning, ataques CGI, backdoors, explotar vulnerabilidades conocidas, etc. En esencia, el funcionamiento de cualquier IDS se basa en el análisis minucioso del tráfico de red, el cual es comparado con firmas de ataques conocidos.

Dejando a un lado los IDSs citados en el apartado sobre Integridad del sistema de archivos, uno de los IDS libres más extendidos es, sin lugar a dudas, Snort. Desarrollado en 1998 originalmente por Martin Roesh, es capaz de capturar y analizar tráfico en tiempo real en redes IP. Algunas de las interfaces de usuario más utilizadas para Snort son BASE (Basic Analysis and Security Engine), NST (Network Security Toolkit) y Sguil.



Snort con BASE

En busca de rootkits

Periódicamente, tampoco está de más buscar posibles rootkits instalados en el sistema. Un rootkit es una herramienta o conjunto de herramientas cuya finalidad es permanecer ocultas y esconder a su vez otros programas, procesos, archivos, puertos de escucha, etc. Generalmente, los rootkits son utilizados por intrusos para mantener el acceso remoto a un sistema con fines maliciosos. Algunos de los rootkits más conocidos son IntoXonia, Adore-ng y Phalanx2. Aunque es complicado que un atacante instale un rootkit habiendo tomado el administrador las medidas de seguridad adecuadas, tampoco es imposible dado los fallos de seguridad que se descubren día a día. Algunos IDS como Snort, son capaces de detectar la presencia de rootkits, sin embargo, también existen herramientas específicas para realizar esta tarea, como Rootkit Hunter (rkhunter) o chrootkit

Poniendo a prueba nuestro sistema

A continuación, se describen brevemente algunas herramientas complementarias con las que podremos realizar nosotros mismos auditorías en busca de vulnerabilidades y configuraciones inapropiadas con el propósito de incrementar la seguridad de nuestro servidor. Todas estas herramientas están disponibles para las principales distribuciones GNU/Linux.

Lynis: Permite detectar en nuestra máquina errores de configuración y posibles vulnerabilidades mediante un profundo análisis del sistema. Entre las auditorías que lleva a cabo Lynis se encuentran: métodos de autentificación, expiración de certificados, versiones antiguas de aplicaciones, contraseñas de las cuentas de usuario y permisos de archivos.

Nikto: Scanner diseñado para detectar remotamente vulnerabilidades en servidores web. Entre sus principales características destacan el soporte para SSL, proxy y evasión ante software de detección de intrusos.

Nmap (Network Mapper): Scanner de puertos realmente útil a la hora de descubrir los servicios que se están ejecutando en un servidor, o incluso, la familia y

versión del sistema operativo (fingerprinting).

Nessus: Scanner de vulnerabilidades basado en un demonio que lleva a acabo el escaneo (nessusd) y un cliente que reporta los resultados obtenidos. Básicamente, el modo de operación de Nessus consiste en detectar inicialmente los puertos abiertos de la máquina objetivo para después lanzar exploits en busca de vulnerabilidades que puedan ser aprovechadas.

---

http://revistalinux.net/articulos/seguridad-basica-en-servidores-linux/