Endurecimiento en el sistema operativo Linux

IBARRA, ANDREA
aaibarraf@gmail.com

Resumen— Es un proceso efectivo para prevención de ataques, protegerlo contra acceso no autorizado, mientras se toman medidas para hacer que el sistema sea confiable. En general, cualquier actividad que se lleve a cabo como parte del endurecimiento de un sistema asegura que el sistema sea seguro y confiable.
El endurecimiento de un sistema operativo es necesario porque algunos sistemas operacionales son diseñados para ser asequibles es decir son fáciles de utilizar, en vez de ser seguros cuando son instalados por primera vez. Se pueden habilitar algunas medidas de seguridad en la mayoría de los sistemas que les permite ser parte de ambientes altamente seguros y confiables.
Como cualquier otro sistema operativo, las fallas de seguridad a nivel de aplicación dejan a LINUX vulnerables, a una variedad de ataques malintencionados. Con los años, han desarrollado muchas herramientas y técnicas para "hardening", el hosts de Linux en un intento por mitigar el riesgo planteado por buggy software. Quitar o deshabilitar servicios innecesarios y demonios, configurando los servicios correctamente y cambiar proveedores impagos, aplicando la seguridad apropiada y los parches, creación de copias de seguridad y configuración de herramientas de monitoreo son todos los pasos esenciales en la construcción de un sistema seguro.
A través de este documento mostraremos algunas maneras de endurecer a Linux desde del Kernel

Índice de Términos—control de acceso, endurecimiento, hardening, kernel, Linux, núcleo, Parches, Seguridad.

Abstract-It is an effective process for preventing many ways to attack safely configuring an operating system, protect against unauthorized access, while steps are taken to make the system reliable. In general, any activity carried out as part of a system hardening ensures that the system is safe and reliable.
Hardening of an operating system is necessary because some operating systems are designed to be easy to use instead of being safe when they are first installed. You can enable some security measures in most systems allowing them to be part of highly secure and reliable environments.
Like any other operating system security flaws application layer Linux leave them vulnerable to a variety of malicious attacks. Over the years, they have developed many tools and techniques to "hardening" the Linux hosts in an attempt to mitigate the risk posed by buggy software. Remove or disable unnecessary services and daemons, services correctly configuring defaults and change suppliers, applying the appropriate security patches, creating backups and configuration monitoring tools are all essential steps in building a secure system.
Through this paper we will show some ways to harden Linux from Kernel.

Primary keys -access control, curing, hardening, kernel, Linux, kernel, Patches, Security.

I. INTRODUCCIÓN

Una vez que el sistema operativo ha sido creado y puesto en producción, el seguimiento diario de los foros de seguridad es necesario para asegurar que el software se mantiene actualizado. Desafortunadamente, este método de administración es totalmente reaccionario, mientras que muchos sistemas están en peligro debido a la falta de mantenimiento adecuado (es decir: no se ejecutan los parches disponibles cuando son lanzados si no se deja como una tarea de mantenimiento posterior y en el momento del incidente se aplican de manera repentina), hay una fuerte necesidad de reducir los riesgos asociados con el software defectuoso,

mantener los sistemas actualizados con parches de proveedor evitará al atacante casual, pero no mantendrá siempre a un atacante que se dirige a un sistema, objetivo definido.

En los entornos informáticos de hoy, hay un creciente número de aplicaciones que requieren conexiones de red, es raro encontrar una máquina que puede ser simplemente desconectado de una red en un intento de impedir que se vea comprometida, normalmente en el entorno que se ejecuta la maquina es imposible desconectarla ya que se afecta de manera radical la disponibilidad del servicio comprometiendo el core del negocio.

Mientras que el número de ordenadores conectados a internet puede correlacionarse directamente con esta tendencia, el número de ataques y la cantidad de daño incurrido sólo aumentará mientras más personas se vuelven dependientes de esta conectividad, es necesario reevaluar el equilibrio entre la seguridad a nivel de host y la comodidad ofrecida por diversos tipos de servicios y funciones si la intención es mantener estos hosts conectados a internet.

En general, el objetivo inicial de que un atacante intenta acceder a un host LINUX (o alguna otra variante del UNIX) es controlar la cuenta súper-usuario (aka. raíz). [1] En la mayoría de los casos, hay poca o ninguna supervisión o auditoría de acciones emprendidas por el super-usuario, incluso cuando se configura, un atacante con privilegios de super-usuario tiene la capacidad de desactivar estos servicios, o cubrir sus pistas mediante la modificación de los archivos de registro, una vez que un atacante tiene el control de la cuenta de super-usuario, instalará módulos rootkits y kernel que le permitirá continuar utilizando el sistema para fines nefastos sin ser detectado fácilmente.

Generalmente se aplican métodos para prevenir o mitigar los riesgos de un ataque exitoso en el nivel de aplicación o en el nivel de sistema operativo, una forma estándar para mitigar el riesgo en el nivel de aplicación está concediendo la menor cantidad de privilegios necesarios para realizar sus funciones, mientras que la mayoría de las aplicaciones no es necesario ejecutar como usuario con privilegios, hay algunas aplicaciones de nivel de sistema que requieren privilegios de super-usuario para realizar ciertas tareas.

Otra táctica adoptada para asegurar cierto sistema aplicaciones de nivelación es mediante el programa chroot. chroot hace una aplicación ejecutar dentro de una carcasa especial con un nuevo directorio raíz, efectivamente aislándolo del resto del sistema.

Los millones de líneas de código que conforman las aplicaciones utilizadas en los sistemas LINUX de hoy hacen casi imposible auditar completamente cada aplicación y eliminar todas las vulnerabilidades potenciales, mientras que aún un componente crítico de una estrategia de defensa en profundidad, técnicas de endurecimiento en el nivel de aplicación son generalmente inadecuadas como un medio para proteger al resto del sistema de fallas en las aplicaciones, independientemente de las aplicaciones de hardening de tiempo pasado, una vulnerabilidad explotada en una sola aplicación a menudo conduce a un compromiso completo del sistema, a medida que aumenta la dependencia de conectividad de red, la necesidad de mirar a métodos de nivel inferiores para prevenir los ataques exitosos llega a ser evidente.

II. EL KERNEL DE LINUX

El software de más bajo nivel de componentes del "sistema de ach incluye un conjunto básico de programas llamados el sistema operativo.

El programa más importante en el conjunto se llama el núcleo, está cargada en la RAM cuando el sistema arranca y contiene muchos procedimientos críticos que son necesarios para que operar el sistema. , mientras el sistema está funcionando, el kernel actúa como mediador entre los componentes de hardware y los procesos que se ejecutan en el sistema – ninguno de los procesos de acceder directamente a cualquier componente de hardware por su propia cuenta.

El kernel de LINUX proporciona un sistema operativo multiusuario. Esto significa que cualquier usuario puede ejecutar cualquier programa en cualquier momento sin preocuparse de los demás usuarios, debido a la naturaleza multiusuario del sistema, el núcleo también es responsable de proporcionar mecanismos para control de acceso y autenticación de usuario evitar que los usuarios o aplicaciones de interferir con la actividad de otros usuarios o programas.

Desbordamientos de búfer son una de las principales vulnerabilidades explotadas para hacerse

con el control de un sistema, al mismo tiempo de procesamiento de datos, el problema surge cuando un usuario es capaz de introducir más datos en este buffer que el programador originalmente asignado, este desbordamiento de datos en los búferes adyacentes podría dañar la aplicación o incluso todo el sistema.
Tres proyectos que han investigado e implementado para proporcionar cierta protección contra desbordamientos de búfer incluyen el proyecto Openwall, Exec escudo y PaX.[2]

A. *El proyecto Openwall :* Creó uno de los anteriores parches de seguridad para el kernel de LINUX que aborda los problemas creados por un área de la pila de usuario ejecutable, esta función evita ataques donde un desbordamiento de búfer sobrescribe punteros retorno permitiendo a un atacante ejecutar código arbitrario insertado en la pila.
   Otras mejoras de seguridad (independiente de la arquitectura) proporcionados por el openwall parche para el núcleo incluyen:
   - Acceso restringido FIFOs y enlaces en/tmp;
   - Acceso restringido a/proc;
   - Mejorada la aplicación del número de los procesos del usuario;
   - Destrucción de segmentos de memoria compartida no está en uso;
   - Otras mejoras para los núcleos pre-2.4.

B. *Exec shield:* El parche para el núcleo exec-shield toma el enfoque utilizado por la implementación del proyecto Openwall de una pila no ejecutable y desplegarla en un intento de proteger a todos los segmentos de memoria virtual, "la función exec-shield proporciona protección contra la pila, tampón o desbordamientos de puntero de función y otros tipos de ataques que se basan en estructuras de datos de sobre escritura o poner el código en esas estructuras, en este momento, el parche para el núcleo exec-shield se limita a la prevención de ataques de espacio de dirección y no proporciona ninguna otras características, como con el parche para el núcleo Openwall, este parche está diseñado para arquitecturas i386 basado.

C. *PaX:* El equipo de PaX ha creado un parche para el núcleo para evitar varias clases de vulnerabilidades de software separando las propiedades de escritura y ejecución en las páginas de memoria, la función NOEXEC se compone de las siguientes tres opciones:
   - Segmentación basada en páginas no ejecutable (SEGMEXEC): Esta opción utiliza las características de segmentación únicas para i386 CPU para la implementación de páginas no ejecutables en arquitecturas i386.
   - Restricciones mmap() y mprotect() (MPROTECT):estas características son para evitar que "la introducción del nuevo código ejecutable en dirección space"(10) de la tarea mediante la adición de restricciones a las funciones mmap() y mprotect(), estas características son en su mayoría arquitectura independiente.

Cada uno de los parches mencionados anteriormente hace que sea más difícil para un atacante ejecutar ataques contra el espacio de direcciones de una aplicación, aunque existen diferentes grados de protección proporcionados por cada uno, el uso de cualquiera de los parches puede aumentar enormemente la seguridad general de un sistema Linux.

III. CONTROL DE ACCESO

Plataformas de computación modernas proporcionan sistemas de control de acceso para asegurar que sólo los usuarios autorizados tengan acceso a objetos que tienen permiso para acceder, en las distribuciones de LINUX estándar, existen dos niveles de acceso por defecto los usuarios regulares y el superusuario.
Control de acceso discrecional (DAC) permite a los usuarios regulares modificar cualquier objeto que les pertenece. DAC se basa únicamente en la identidad de un usuario determinado, la propiedad de un objeto y los permisos que se han otorgado a ese usuario, el super-usuario, el dueño de todos los objetos en el sistema, no está sujeto a ninguna de estas restricciones y puede invalidar cualquier restricción creada por los usuarios regulares.[3]

Con la implementación de una política de seguridad fuerte y algunas restricciones de capa de aplicación, acceso a la cuenta de root puede ser restringido por:

- Restringir acceso root a la consola del sistema.
- Restringir uso del comando su para un grupo de administradores.
- Requerir a los administradores a usar herramientas tales como sudo para ejecutar comandos como root.

Estas técnicas junto con una política de contraseñas fuertes son excelentes métodos de protección contra la divulgación accidental contraseña, errores cometidos por los administradores logueados como root, y ataques de fuerza bruta contraseña contra la cuenta de root, sin embargo, estas técnicas hacen poco para proteger el sistema de aplicaciones que se ejecutan como root.

Se han propuesto diversos enfoques diferentes sobre el control de acceso, en un intento de aumentar la seguridad general de los ejércitos de LINUX, el equilibrio entre seguridad y conveniencia es algo que necesita reexaminar.

Algunas alternativas a DAC incluyen:

- Control de acceso obligatorio (MAC): Un sistema que requiere un administrador de seguridad definir los atributos de control de acceso asignados a los objetos, los usuarios y los procesos, estas políticas no pueden ser modificadas por los usuarios regulares, al comparar los atributos de control de acceso de un objeto con los atributos de control de acceso del usuario o proceso que intenta acceder a ese objeto, el sistema determinará si permitir o impedir el acceso.
- Role Based Access Control (RBAC): Especifica un sistema que asigna a los usuarios a los grupos y luego asigna a los grupos la capacidad de realizar tareas especificadas en pedazos de datos.
- Basado en el Control de acceso (RSBAC): Una combinación modular de modelos de seguridad que divide el acceso de control.
- Dominio y tipo de aplicación (DTE): Un sistema que separa los procesos en los dominios y tipos de objetos, entonces, el sistema puede configurarse para permitir o impedir el acceso a tipos especificados dominios específicos. controles de acceso pueden también configurarse para permitir o impedir que los dominios interactuando unos con otros.

IV. SISTEMAS DE DETECCIÓN DE INTRUSOS EN LINUX

El Proyecto LINUX "Intrusion Detection System "en sus siglas (LIDS), ha lanzado un parche para el kernel y la herramienta administrativa que añade control de acceso obligatorio para el kernel, estos sirven para proteger tanto a los objetos como las conexiones de red, mientras que este sistema podría ser considerado más de un sistema de prevención de intrusiones, un detector de exploración integrada puerto también ayuda a detectar intento de acceso no autorizado. [3]

El control de acceso basado en LINUX es un parche para el núcleo y un conjunto de herramientas administrativas incluyendo la implementación de varios módulos para el control de acceso, entre otras opciones de seguridad avanzada:

- Un sistema de control de acceso avanzado basado en MAC;
- Módulos de RBAC que permiten a los administradores de seguridad y sistemas de acceso a la información necesaria para desempeñar sus funciones;
- Un módulo de aislamiento para el almacenamiento de datos personales;
- Un módulo que analiza los archivos en busca de malware;
- Un módulo que añade nuevas banderas de archivos a la lectura tradicional, escritura y ejecución banderas utilizadas en los objetos (es decir, añadir-solamente, seguro-borrar y no_delete_or_rename.);
- Listas de control de acceso para los objetos;
- Aplicación de autorización para controlar aún más la capacidad de un usuario para cambiar a otro usuario;
- Un módulo de capacidades para restringir los privilegios tradicionalmente otorgados a la raíz cuenta y conceder estos privilegios a otros usuarios, es necesario limitar los recursos avanzados para los usuarios y las aplicaciones.

V. LINUX CON SEGURIDAD MEJORADA

Security-Enhanced LINUX (SELinux) es un proyecto que ha integrado basado en roles de control de acceso y tipo de aplicación en el kernel estándar de LINUX. Algunas de las características distintivas del sistema Linux con seguridad mejorada incluyen: [2]

- La separación limpia de la política de cumplimiento. Interfaces política bien definida.
- Independiente de Políticas Específicas y Política de Idiomas. Independiente de Formatos y contenido de la etiqueta de seguridad específicos.
- Las etiquetas individuales y controles de objetos y servicios del núcleo.
- El almacenamiento en caché de las Decisiones de acceso para la Eficiencia.
- Apoyo a la Política de Cambios. Los controles sobre inicialización de procesos y sobre Sucesiones y ejecución del programa.
- Los controles sobre los sistemas de archivos, directorios, archivos y descripción de archivos abiertos.
- Control de los mensajes y las interfaces de red.

VI. MÓDULOS DE SEGURIDAD DE LINUX

Las funciones de seguridad mejoradas y los mecanismos de control de acceso han impedido que muchas de las soluciones propuestas deban estar integradas en el kernel LINUX, aunque LINUX ha apoyado durante mucho tiempo, "creación de los módulos de seguridad eficaces es problemático ya que el núcleo no proporciona la infraestructura para permitir que los módulos del núcleo para mediar el acceso a los objetos del kernel." [4]

En la serie del kernel de desarrollo 2.5, los privilegios de la cuenta de super-usuario tradicional se ha separado a cabo en un sistema de capacidades. Con la integración del marco módulos de seguridad LINUX (LSM), es posible reemplazar la cuenta de super-usuario tradicional con un sistema alternativo de control de acceso en la corriente principal núcleo de LINUX.

Algunos de los sistemas actuales que han sido portados a la aplicación a través de la

Marco LSM incluyen: Capacidades POSIX.1e, SE LINUX, dominio y tipo

Las distribuciones que ofrecen los núcleos endurecidos, además de los proyectos se centran en el núcleo, se formalizaron otros proyectos para empaquetar varias mejoras de seguridad en la totalidad de las distribuciones de LINUX, si bien muchas de las distribuciones principales empaquetar algunos de estos componentes, por lo general, no se aplican de forma predeterminada durante la instalación del sistema.

Tres proyectos que se centran en la construcción de un sistema operativo de alta seguridad incluyen Adamantix, búho GNU / * / Linux, y el proyecto Gentoo Hardened. [4]

A. *Adamantix: El* proyecto Adamantix ha creado una distribución *LINUX* completa sobre la base de la actual versión estable de debian GNU / Linux, mediante la integración y la habilitación de funciones de seguridad no se utilizan actualmente en las distribuciones de corriente principal, el proyecto Adamantix ha creado una distribución que es mucho más seguro fuera de la caja.

   El kernel Adamantix está parcheado con los parches del kernel personas y RSBAC para proporcionar una protección adicional contra los ataques de espacio de direcciones y un sistema de control de acceso de grano fino, hay tres tipos diferentes de configuraciones de núcleos disponibles para el uso en estos sistemas, el núcleo de base tiene RSBAC discapacitados; la imagen con el sufijo "-sec" tiene RSBAC habilitado y hace cumplir todas las políticas de seguridad; y Algunas de las características adicionales que se proporcionan en la distribución Adamantix incluyen:

   - El firewall de proxy a nivel de aplicación Zorp;
   - IPSec integrada;
   - Integrado dispositivo de bucle de retorno cifrado (es decir, para el cifrado de particiones de origen.);
   - Controladores en modo yWLAN Host-AP que permiten a Linux para ser utilizado como una WLAN estación base.

B. *Owl GNU / * / LINUX:* Además de proporcionar los parches del kernel de LINUX 2.0, 2.2, y 2.4 kernels, el proyecto openwall ofrece una total seguridad mejorada distribución de LINUX para servidores. Owl utiliza una combinación de métodos para proporcionar un sistema operativo seguro, la distribución Owl también cuenta con un entorno de construcción que permite a un administrador para reconstruir todo el sistema (excepto el kernel) de código fuente mediante la ejecución del comando "make buildworld".

C. *Proyecto Gentoo Hardened :* La distribución Gentoo LINUX se esfuerza por ofrecer un sistema que es altamente configurable y optimizada para cada aplicación específica, una de las mayores características de Gentoo es su distribución de software Portage y sistema de envasado, portage utiliza un conjunto de scripts proporcionados por los desarrolladores de Gentoo para construir paquetes personalizados a partir de código fuente para cada sistema. Como es posible construir todos los binarios del sistema de la fuente, la integración de las funciones avanzadas de seguridad se convierte en una tarea más fácil en algunos casos.

Proyecto Gentoo, si bien es cierto que muchas otras distribuciones principales ofrecen versiones de parches de seguridad del kernel empaquetados, el edificio y la configuración personalizada de imágenes del núcleo en general, no es de serie un procedimiento en otras distribuciones como lo es con Gentoo.

VII. CONCLUSIONES

- En el endurecimiento de un sistema operativo como lo es Linux se debe hacer mediante el uso de uno o más parches de seguridad, que mejora en gran medida la seguridad global del sistema.
- La finalidad de los parches es reducir la eficacia de muchos tipos de ataques previniendo por ejemplo problemas de desbordamiento de búfer
- En el caso de que un atacante sea capaz de explotar una vulnerabilidad del sistema, la implementación y configuración adecuada de control de acceso permitirá evitar que un atacante consiga pleno control del sistema.
- La seguridad física del host debe ser examinada, evaluada, y mejorada dependiendo del tipo de máquina que se tenga, para establecer la estrategia de defensa en profundidad en un host Linux.

REFERENCIAS

[1] SANS, "Linux Checklist," [Online]. Available: http://www.sans.org/score/checklists/Linuxchecklist.pdf

[2] SANS, "Score Sans," [Online]. Available: http://www.sans.org/score/Linuxchecklist.php

[3] NIST, "Hardening Linux," [Online]. Available: http://csrc.nist.gov/publications/nistpubs/800-123/SP800-123.pdf

[4] IMMUNIX, “Linux Security Modules: General Security Support for the Linux Kernel.” Available http://lsm.immunix.org/docs/lsm-usenix-2002/html/ (9 Nov 2003).

Autor
Andrea Alexandra Ibarra Fonseca
Ingeniera de Sistemas
Universidad Católica de Colombia2009
Auditor interno en la norma ISO 27001/2005/SGS 2011