Mecanismos de aislamiento y privilegios en sistemas operativos

—

Gabriel Hernandez Yukioayax Canek - 321058526

Facultad de Ingeniería, UNAM

Mecanismos de aislamiento y privilegios en sistemas operativos

Los sistemas operativos modernos no solo ejecutan aplicaciones, también garantizan la integridad y la disponibilidad del entorno de ejecución aplicando dos principios complementarios, el aislamiento y los privilegios.

De manera formal, el aislamiento de procesos es un mecanismo de seguridad crítico que garantiza que un proceso esté aislado de otros procesos y limita su interacción con el sistema operativo, esta técnica evita que el comportamiento erróneo o malicioso de un proceso afecte a otros procesos, esto se logra limitando los recursos (memoria, sistemas de archivos, red, etc.) y controlando los permisos de acceso, mientras que el privilegio se refiere a un nivel de autorización concedido a un usuario, proceso, cuenta o aplicación, que le permite acceder a recursos del sistema (archivos, redes) o realizar acciones específicas (instalar software, cambiar configuraciones, gestionar cuentas).

En términos generales, el aislamiento delimita el contexto de ejecución de un proceso y establece las fronteras de su interacción con el resto del sistema, mientras que los privilegios definen las operaciones autorizadas dentro de ese contexto y las condiciones bajo las cuales pueden ejecutarse, ambos conceptos constituyen el fundamento sobre el que los sistemas operativos son diseñados y cómo se aplican sus controles de seguridad: el aislamiento limita la exposición y posible impacto de una carga y los privilegios regulan quién puede hacer qué, cuándo y bajo qué condiciones, cumplimiento que el sistema operativo hace efectivo sobre los objetos que controla mediante permisos tradicionales (usuario, grupo y otros), listas de control de acceso, etiquetas de control obligatorio y políticas basadas en roles, mientras que las aplicaciones especializadas pueden añadir reglas internas apoyándose en esas mismas primitivas.

Existen diversos sistemas operativos y UNIX es uno de los pioneros, pues fijó muchas de las ideas que hoy heredan Linux, FreeBSD, OpenBSD, Solaris, entre otros, en la práctica se nota una base compartida en las herramientas de línea de comandos que usamos a diario, por eso comandos como cd, mkdir y los listados de directorios con utilidades equivalentes se usan de forma parecida tanto en Linux como en Windows, sobre esa influencia se forma la base común de los Unix, el sistema expone objetos como archivos, cada proceso se ejecuta con un identificador de usuario y uno o más grupos, la autorización se resuelve con permisos de usuario, grupo y otros junto con la umask, y cuando aplica los bits setuid, setgid y sticky, si se requiere mayor precisión del control pueden emplearse ACL POSIX y, en Linux, además, las capabilities del kernel,

y sobre esta base cada plataforma añade mecanismos específicos de aislamiento y control namespaces y cgroups v2 junto con políticas de control obligatorio como SELinux en Linux, Jails y Capsicum en FreeBSD, pledge/unveil en OpenBSD, y Zones en Solaris con controles de recursos y RBAC para perseguir el mismo objetivo de seguridad con diferencias en alcance, precisión del control, sobrecarga y casos de uso.
A continuación se describen los mecanismos que cada plataforma emplea para materializar estas funciones en la práctica:
En Windows:

- ACL (Listas de Control de Acceso): Cada objeto securizable (archivo, clave de registro, etc.) tiene un descriptor de seguridad que incluye las ACL, la DACL específica permisos permitidos o denegados para identidades (usuarios/grupos) y la SACL define qué eventos deben auditarse, Windows evalúa estas listas al intentar un acceso para decidir si procede la operación.
- UAC (Control de Cuentas de Usuario): UAC separa el contexto de ejecución y exige consentimiento o credenciales para ejecutar acciones sensibles, en la práctica, reduce la probabilidad de que procesos o usuarios realicen cambios elevados sin la intervención explícita del usuario, usando tokens y flujo de elevación documentados por la propia plataforma.
- MIC (Control de Integridad Obligatorio): El sistema asigna niveles de integridad a procesos y objetos, verificando esa relación antes de la evaluación por ACL, como resultado, un proceso con integridad baja tiene restricciones adicionales (por ejemplo, no puede escribir en objetos de integridad superior), lo que añade una capa previa de separación basada en confianza.

En Linux:

- Namespaces: El kernel encapsula recursos que tradicionalmente son globales (por ejemplo, PID, red y puntos de montaje) dentro de espacios de nombres, los procesos que pertenecen a un mismo namespace ven su propia instancia de esos recursos, lo que posibilita ejecutar servicios aislados (por ejemplo contenedores) sin interferir con el sistema anfitrión.
- Cgroups v2: Los grupos de control organizan procesos en una jerarquía y permiten asignar o limitar recursos (CPU, memoria, E/S) mediante controladores y parámetros (p. ej. cpu.max), evitando que una carga consuma todos los recursos y manteniendo un comportamiento más predecible del sistema, cgroups v2 unifica el modelo de control frente a versiones anteriores.
- SELinux: Sobre esa base de aislamiento y control de recursos, SELinux aplica un modelo de control de acceso obligatorio (MAC) mediante etiquetas/contextos y políticas, las decisiones de acceso se basan en la política y los modos de

operación (por ejemplo enforcing o permissive), de modo que si una aplicación intenta una acción no permitida por la política, ésta se deniega para contener el impacto.

En FreeBSD:

- Jails: FreeBSD extiende chroot creando entornos aislados a nivel de sistema que restringen sistema de archivos, usuarios y red, permiten ejecutar servicios con su propia vista del sistema y controles de recursos para no afectar al host.
- Capsicum: Framework de capacidades, los procesos entran en capability mode, transforman descriptores en capacidades y quedan limitados a operaciones permitidas, facilitando sandboxes sin privilegios globales.

En OpenBSD:

- pledge: Llamada al sistema por la que un proceso declara categorías de operaciones POSIX que usará (ej. stdio, rpath, inet), si intenta algo fuera de lo prometido el kernel termina el proceso.
- unveil: Complementa a pledge restringiendo las rutas del sistema de archivos que un proceso puede abrir, limitando el acceso a ficheros concretos.

En Solaris / Illumos:

- Zones (Containers): Entornos privados dentro del mismo kernel, con identidad y visibilidad limitada para ejecutar aplicaciones aisladas.
- Resource controls y RBAC: Controles para fijar cuotas de CPU/memoria/E/S por zona y un modelo RBAC/privilegios para delegar capacidades administrativas por roles.

En macOS:

- App Sandbox (entitlements): Las apps declaran permisos (entitlements) que limitan acceso a archivos, red y APIs sensibles, el sistema aplica esas restricciones en tiempo de ejecución.
- System Integrity Protection (SIP): Protege ubicaciones y procesos críticos contra modificaciones incluso por root, evitando la alteración de componentes esenciales.

En NetBSD:

- Veriexec: Mecanismo de integridad para definir/verificar firmas o huellas de ejecutables y evitar ejecutar binarios no autorizados.
- Kauth / frameworks de autorización: Subsistemas del kernel para centralizar decisiones de autorización y aplicar políticas de control de acceso.

En MINIX:

- Microkernel / multiserver: Arquitectura que minimiza lo que corre en kernel y mueve servicios a espacio de usuario, cada servicio tiene privilegios mínimos, lo que reduce impacto de fallos y facilita recuperación.

En QNX:

- Microkernel y resource managers: Microkernel y gestores de recursos en espacio de usuario y mensajería IPC, la separación permite aislar componentes y aplicar políticas por servidor/gestor.

En VxWorks:

- Aislamiento en embebidos: Configuraciones de seguridad para entornos críticos, secure boot, soporte TrustZone/TEEs, módulos criptográficos y políticas de particionado que protegen imágenes y restringen ejecución privilegiada.

Es importante mencionar que los listados anteriores no abarcan todos los sistemas operativos, existen otras familias y variantes no incluidas por brevedad y porque pueden analizarse con la misma base de aislamiento y privilegios, asimismo, aunque la mayoría de los aquí presentados son derivados de Unix y comparten la base de objetos como archivos y permisos de usuario, grupo y otros, en arquitecturas de microkernel como MINIX, QNX o VxWorks varios servicios se trasladan a espacio de usuario y el modelo de seguridad puede diferir de forma notable, desde la separación de planos hasta los mecanismos de autorización, por lo que las comparaciones deben leerse con esa consideración.

En los sistemas operativos puros existen muchos mecanismos integrados para aislar procesos y controlar privilegios (por ejemplo: jails, pledge, zones, microkernels, SIP, TrustZone, etc.), he de aclarar que no me refiero a las distribuciones que solo empaquetan y añaden herramientas sobre un SO, sino al núcleo mismo, osea, la base de las diferentes distribuciones, estas bases pueden ser aprovechadas, configuradas o ampliadas por las distribuciones y productos, por lo que el comportamiento real en la implementación depende tanto del sistema operativo base como de las decisiones de integración y operación, a continuación se presentan ejemplos representativos que muestran cómo se aplican estas funciones:

- FreeBSD Jails: FreeBSD extiende el concepto chroot creando contenedores de nivel de sistema que virtualiza el sistema de archivos, usuarios y red, las jaulas gruesas replican todo el sistema base proporcionando bibliotecas y ejecutables propios, las jaulas delgadas comparten la base del sistema para ahorrar recursos, y las "service jails" comparten todo el sistema de archivos pero confinan servicios específicos.
- OpenBSD pledge: La llamada del sistema pledge restringe un proceso a un subconjunto de operaciones POSIX (como stdio, rpath, wpath, cpath, fattr, inet,

dns), el proceso declara qué categorías de operaciones utilizará y cualquier violación provoca la terminación del proceso, reduciendo la superficie de ataque y promoviendo la separación de privilegios.

- Cisco IOS: Es un sistema operativo para equipos de red con una CLI organizada por modos (usuario > y privilegiado #, más modo de configuración), su modelo de seguridad gira en niveles de privilegio 0–15: el 1 ofrece comandos básicos de monitoreo, el 15 permite administración total, y los niveles intermedios pueden personalizarse asignando comandos concretos a cada nivel, la autenticación puede ser local o mediante AAA (TACACS+/RADIUS) para autenticar, autorizar y registrar qué comandos ejecuta cada usuario, el objetivo es mantener disponibilidad y trazabilidad en routers y switches, más que un esquema de archivos y ACL al estilo Unix.

En conclusión, el aislamiento y el control de privilegios determinan cómo se ejecutan los procesos y quién puede realizar acciones sensibles en un sistema, si se diseñan y mantienen correctamente dando solo los permisos necesarios, manteniendo el software y parches al día, definiendo políticas claras de seguridad, registrando y previniendo eventos indeseables, y probando cambios en entornos controlados, estos mecanismos reducen el área de ataque, limitan el daño que podría producirse y facilitan la detección y recuperación, además hacen que el sistema sea menos atractivo para un atacante; es importante mencionar que esto no depende solo de las capacidades del sistema operativo base (namespaces/cgroups/SELinux, jails/pledge, zones, SIP, microkernels, etc.), sino también de cómo se configuran, integran y mantienen esas capacidades en la operación diaria. En la práctica, combinar buenas herramientas en el SO con prácticas como gestión disciplinada, automatización de actualizaciones, monitoreo continuo y revisión periódica de políticas convierte esas medidas en una defensa efectiva y manejable.

Bibliografía

Achacoso, K. K. (diciembre 6, 2017). *VxWorks Secure Boot.* WNDRVR. Recuperado de https://www.windriver.com/blog/vxworks-secure-boot

Apesteguía, F. (febrero 18, 2025). *Chapter 4. The Jail Subsystem*. FreeBSD, Recuperado de https://docs.freebsd.org/en/books/arch-handbook/jail/

Bobiles, K. (2015). *Cisco IOS - Privilege Levels*. CISCO. Recuperado de https://learningnetwork.cisco.com/s/blogs/a0D3i000002eeWTEAY/cisco-ios-privilege-levels

Global Limited, H. (marzo 9, 2025). *Técnicas de aislamiento de procesos y sandboxing en sistemas operativos.* HOSTRAGONS. Recuperado de

https://www.hostragons.com/es/blog/tecnicas-de-aislamiento-de-procesos-y-sandboxing-en-sistemas-operativos/

Heo, T. (octubre, 2015). *Control Group v2.* The kernel development community. Recuperado de https://docs.kernel.org/admin-guide/cgroup-v2.html

Polarian. (septiembre 26, 2025). *Chapter 17. Jails and Containers*. FreeBSD. Recuperado de https://docs.freebsd.org/en/books/handbook/jails/

Tsagaankhuu, G. (diciembre 31, 2024). *FreeBSD Handbook.* FreeBSD. Recuperado de https://docs.freebsd.org/en/books/handbook/book/

S.A. (marzo 18, 2011). *veriexec(5) - NetBSD Manual Pages*. NetBSD. Recuperado de https://man.netbsd.org/veriexec.5

S.A. (octubre, 2017). *Zone Administration Overview.* ORACLE. Recuperado de https://docs.oracle.com/cd/E53394_01/html/E54762/gqhar.html

S.A. (2018). *chmod.* The Open GROUP. Recuperado de https://pubs.opengroup.org/onlinepubs/9699919799/utilities/chmod.html

S.A. (2018). *V3_chap01.* The Open GROUP. Recuperado de https://pubs.opengroup.org/onlinepubs/9699919799/utilities/V3_chap01.html

S.A. (2018). *V3_chap01 2018 edition.* The Open GROUP. Recuperado de https://pubs.opengroup.org/onlinepubs/9699919799.2018edition/

S.A. (noviembre, 2020). *Zones Concepts Overview*. ORACLE. Recuperado de https://docs.oracle.com/cd/E37838_01/html/E61038/zones.intro-2.html

S.A. (septiembre 6, 2021). *UNVEIL(2)*. OpenBSD manual page server. Recuperado de https://man.openbsd.org/unveil.2

S.A. (enero 3, 2024). *FreeBSD Manual Pages.* FreeBSD. Recuperado de https://man.freebsd.org/cgi/man.cgi?capsicum

S.A. (abril 20, 2024). *About cgroup v2.* kubernetes. Recuperado de https://kubernetes.io/docs/concepts/architecture/cgroups/

S.A. (marzo 20, 2025). *About System Integrity Protection on your Mac.* Apple. Recuperado de https://support.apple.com/en-us/102149

S.A. (mayo 16, 2025). *Cómo funciona el Control de cuentas de usuario.* Microsoft. Recuperado de https://learn.microsoft.com/es-es/windows/security/application-security/application-control/user-account-control/how-it-works

S.A. (mayo 17, 2025). *namespaces (7) - Linux manual page.* ma77.org. Recuperado de https://man7.org/linux/man-pages/man7/namespaces.7.html

S.A. (julio 1, 2025). *kauth(9) - NetBSD Manual Pages*. NetBSD. Recuperado de https://man.netbsd.org/kauth.9

S.A. (julio 2, 2025). *PLEDGE(2)*. OpenBSD manual page server. Recuperado de https://man.openbsd.org/pledge.2

S.A. (julio 2, 2025). *System Calls Manual.* OpenBSD manual page server. Recuperado de https://man.openbsd.org/pledge.2

S.A. (julio 8, 2025). *Mecanismo de control de integridad obligatorio.* Microsoft. Recuperado de https://learn.microsoft.com/es-es/windows/win32/secauthz/mandatory-integrity-control

S.A. (julio 10, 2025). *Listas de control de acceso.* Microsoft. Recuperado de https://learn.microsoft.com/es-es/windows/win32/secauthz/access-control-lists

S.A. (julio 27, 2025). *Mejoras de seguridad.* source. Recuperado de https://source.android.com/docs/security/enhancements?hl=es-419

S.A. (julio 27, 2025). *Zona de pruebas para aplicaciones.* source. Recuperado de https://source.android.com/docs/security/app-sandbox?hl=es-419

S.A. (agosto 11, 2025). *Resource managers.* QNX. Recuperado de https://www.qnx.com/developers/docs/8.0/com.qnx.doc.neutrino.user_guide/topic/os_intro_RESMGR.html

S.A. (s.f.). *Controlling Switch Access with Passwords and Privilege Levels*. CISCO. Recuperado el 4 de octubre de 2025 de https://www.cisco.com/en/US/docs/switches/lan/catalyst3850/software/release/3.2_0_se/multibook/configuration_guide/b_consolidated_config_guide_3850_chapter_0101101.html

S.A. (s.f.). *Documentation.* Red Hat Documentation. Recuperado el 1 de octubre de 2025 de https://docs.redhat.com/en

S.A. (s.f.). *Overview of Minix 3 architecture.* MINIX 3. Recuperado el 4 de octubre de 2025 de https://wiki.minix3.org/doku.php?id=developersguide%3Aoverviewofminixarchitecture

S.A. (s.f.). *Privilegio.* orsys le mag', Recuperado el 29 de septiembre de 2025 de https://www.orsys.fr/orsys-lemag/es/glosario/privilegio/

S.A. (s.f.). *Resource Controls (Overview).* ORACLE. Recuperado el 4 de octubre de 2025 de https://docs.oracle.com/cd/E23824_01/html/821-1460/rmctrls-1.html

S.A. (s.f.). *Role-Based Access Control (Overview).* ORACLE. Recuperado el 4 de octubre de 2025 de https://docs.oracle.com/cd/E23824_01/html/821-1456/rbac-1.html