Engineer

Privilegio

🎟 derecho que un sistema concede a una persona o proceso

Qué es

Un privilegio es un permiso otorgado por decisión de una autoridad a una persona o a un programa en ejecución para hacer algo que el sistema prohíbe sin él: leer los archivos de otro usuario, asociarse a un puerto inferior a 1024, crear cuentas o ejecutar una función de base de datos. Se habla de privilegios en plural porque un sujeto rara vez tiene solo uno, y de cuentas o procesos «con privilegios» y «sin privilegios» para indicar de qué lado de esa frontera se encuentran. El glosario de NIST añade la parte que es fácil olvidar: los privilegios describen el comportamiento autorizado de un sujeto, y una autoridad los define y los recoge por escrito en políticas o reglas.

El término está cerca de otros dos. El control de acceso es el acto de decidir sobre cada solicitud; un privilegio es uno de los datos sobre el solicitante en los que se basa esa decisión. Un permiso suele ser la misma idea vista desde el recurso: el archivo dice quién puede leerlo, y el lector tiene el derecho de hacerlo.

En Linux, la palabra tiene un significado preciso. El manual capabilities(7) describe la división tradicional de Unix: los procesos con privilegios, que se ejecutan con un identificador efectivo de usuario de 0 y se conocen como root, eluden todas las comprobaciones de permisos del kernel, mientras que cualquier otro proceso se comprueba frente a sus credenciales. Desde Linux 2.2, el kernel divide el poder de root en unidades separadas llamadas capacidades, que pueden activarse y desactivarse de una en una, por lo que un programa que solo necesita asociarse a un puerto bajo puede recibir `CAP_NET_BIND_SERVICE` y nada más. La página credentials(7) explica de quién es la identidad comprobada: el identificador real de usuario indica a quién pertenece un proceso, el identificador efectivo es el que el kernel compara con los permisos, y el identificador de usuario guardado (saved set-user-ID) permite que un programa deje de usar una identidad con más privilegios y la recupere después.

Las bases de datos y las organizaciones usan la misma palabra para sus propias concesiones. Un rol de base de datos tiene privilegios sobre esquemas, tablas y funciones. Una cuenta de administración tiene privilegios de los que carece una cuenta de personal. La idea es idéntica a todos los niveles: un derecho, concedido de forma deliberada, para cruzar una frontera que de otro modo está cerrada.

Utilice el concepto en una decisión

Aplique «¿Qué es un privilegio?» a una decisión real. Podemos ayudarle a relacionar el concepto con el sistema que está construyendo o comprando. Cuentas, contraseñas y permisos organizados para que cada persona tenga exactamente el acceso que necesita y lo pierda el mismo día que deja la empresa.

Explore nuestros servicios Póngase en contacto →

Compruebe los supuestos detrás del término

Utilice la explicación y los trabajos enlazados para formular preguntas sobre su proyecto. Podemos examinar los supuestos con usted antes de elegir una herramienta. 1.000 reglas RBAC para servidores Linux — 1,000 reglas de acceso escritas para cerrar una brecha de seguridad

1.000 reglas RBAC para servidores Linux Póngase en contacto →

Por qué importa

Los privilegios de un programa determinan el alcance del daño que puede causar cuando algo falla en él. Ese es el objetivo declarado de las capacidades: un programa comprometido causa menos daño con un conjunto reducido de derechos que el mismo programa ejecutado como root. Un servidor web con un fallo de ejecución de código y privilegios de root entrega toda la máquina a quien descubre el fallo; el mismo fallo en un proceso que puede leer su propio directorio y asociarse a un puerto tiene un alcance muy reducido.

NIST recoge esto como el principio de mínimo privilegio. El control AC-6 de SP 800-53 pide a una organización permitir solo los accesos que los usuarios y los procesos que actúan en su nombre necesitan para realizar las tareas asignadas. Las mejoras de ese control convierten ese principio en trabajo rutinario: los administradores utilizan una cuenta sin privilegios para tareas cotidianas como el correo, las cuentas con privilegios se restringen a quienes las necesitan, se registra el uso de funciones privilegiadas y se revisan y retiran los privilegios cuando ya no puede justificarse su necesidad.

Los valores predeterminados rara vez parten de ahí. En systemd, un servicio del sistema sin una línea `User=` se ejecuta como root, y `NoNewPrivileges=`, el ajuste que impide que un servicio y sus hijos adquieran más privilegios mediante `execve()`, está desactivado salvo que alguien lo active. Las herramientas de base de datos suelen conectarse como propietarias de todas las tablas porque así se

creó el esquema. Las cuentas de personal nuevo suelen copiarse de las de alguien que ya trabaja allí, con todos sus derechos. Cada valor predeterminado resulta cómodo el primer día y caro el día en que hay una intrusión.

La contrapartida es real. Restringir privilegios exige tiempo porque alguien tiene que averiguar qué hace realmente un programa, y un conjunto demasiado limitado rompe el funcionamiento durante la ejecución en lugar de hacerlo durante la instalación. La decisión del equipo suele ser dónde invertir primero ese tiempo: en lo que está expuesto a internet, en lo que guarda datos que sus usuarios echarían en falta y en las cuentas que pueden modificar otras cuentas. Para un comprador o un auditor, la pregunta útil es sencilla y directa: quién y qué tiene aquí derechos de administración, y por qué.

Cómo se aplica en la práctica

En el host de la empresa, trece archivos de unidad se ejecutaban con los privilegios que el supervisor concede de forma predeterminada, es decir, con la mayoría de los privilegios. Cada plantilla de unidad recibió un límite de privilegios fijado a los propios de la unidad, un filtro de llamadas al sistema, familias de direcciones restringidas y memoria que puede escribirse o ejecutarse, pero nunca ambas cosas. La puntuación de endurecimiento del mecanismo de alerta por ausencia de señales pasó de 9,6 a 1,5; la de la forja de Git, de 8,3 a 1,5; y las de las dos unidades de copia de seguridad, de 9,6 a 2,3 y 2,5. Una directiva resultó estar mal escrita en tres plantillas, y el supervisor había estado registrando una clave desconocida y arrancando la unidad de todos modos, por lo que ahora todas las unidades pasan por el verificador del propio supervisor en cada ejecución de convergencia. La página también recoge lo que la puntuación no muestra: una directiva que hace fallar una unidad lo hace durante la ejecución.

En una plataforma de aplicaciones, la API inicia sesión en la base de datos con un rol de mínimo privilegio que solo tiene el permiso EXECUTE sobre las funciones de la aplicación, que se ejecutan como SECURITY DEFINER con consultas parametrizadas. Los derechos de acceso, como el nivel del plan, el rol y los ámbitos de organización y buque, se vuelven a comprobar en el servidor en cada solicitud, y las restricciones de la interfaz se tratan únicamente como experiencia de usuario.

En una organización, se consolidaron las identidades y el acceso en M365, 1Password, Red Hat SSO y OKTA SSO, se impuso el mínimo privilegio y se normalizaron las altas y bajas para revocar el acceso con prontitud en lugar de

dejarlo activo después de que alguien se marchara.

Dónde surgen los problemas

Un servicio se ejecuta como root porque su unidad no dice nada. El valor predeterminado del supervisor para un servicio del sistema es root, por lo que una unidad sin una línea `User=` tiene todos los privilegios de la máquina. Asigne a cada servicio su propio usuario o deje que systemd asigne uno con `DynamicUser=`, que crea un usuario al arrancar la unidad y lo libera cuando se detiene.

Una capacidad concede en realidad la mayor parte del poder de root. `CAP_SYS_ADMIN` cubre tantas operaciones que su propio manual la califica de sobrecargada y de «el nuevo root». Un equipo que la concede para habilitar una función ha concedido la mayor parte del poder de root. Busque la capacidad concreta que necesita el programa y limite el conjunto máximo de capacidades a esa lista:

```
[Service]
User=web
NoNewPrivileges=yes
CapabilityBoundingSet=CAP_NET_BIND_SERVICE
AmbientCapabilities=CAP_NET_BIND_SERVICE
```

Esos ajustes permiten que un servicio ejecutado como usuario normal se asocie al puerto 80, y eliminan cualquier otra capacidad de su conjunto máximo.

Un proceso adquiere más privilegios de los que tenía al empezar. Un programa ejecutado como usuario normal todavía puede ejecutar un binario con set-user-ID o un archivo con capacidades asociadas, y el kernel eleva sus privilegios para esa ejecución. Quienes defienden el sistema lo reconocen como un proceso cuyo identificador efectivo de usuario difiere del real. `NoNewPrivileges=yes` cierra esa vía para un servicio y todos los hijos que inicia.

Los derechos se acumulan cuando las personas cambian de puesto. Alguien cambia de equipo y conserva el acceso anterior junto con el nuevo, y al cabo de unos años su cuenta reúne los derechos de todos los roles que ha tenido. AC-6

responde con una revisión periódica, y AC-2 con una gestión de cuentas vinculada a los traslados y las salidas, de modo que un cambio de puesto también suponga un cambio de acceso.

La propia cuenta de conexión de la aplicación puede hacerlo todo. Cuando una API se conecta como propietaria del esquema, un solo fallo de inyección alcanza todas las tablas. Un rol de conexión que solo puede invocar las funciones utilizadas por la aplicación convierte el mismo fallo en un problema mucho menor.

Lecturas adicionales

- Privilegio, en el glosario de NIST CSRC — La definición de CNSSI 4009, con la nota de que una autoridad define los privilegios.
- capabilities(7) — Cómo Linux divide root en capacidades separadas y cómo las poseen los hilos.
- credentials(7) — Identificadores de usuario real, efectivo, guardado y del sistema de archivos, y qué decide cada uno.
- systemd.exec(5) — `User=`, `DynamicUser=`, `NoNewPrivileges=` y los ajustes de capacidades.
- NIST SP 800-53 Rev. 5 — El control AC-6, el mínimo privilegio y las mejoras que lo convierten en práctica rutinaria.

Glosario

Elija según sus restricciones reales

Describa la ruta de despliegue, los riesgos operativos y quiénes se encargan de las guardias. Podemos definir el alcance de un cambio con comprobaciones, una vía de recuperación y responsabilidades claras.

Vea cuánto cuesta Póngase en contacto →

Haga práctica la terminología

Pregunte cómo se aplica «¿Qué es un privilegio?» a su organización. Basta un ejemplo concreto de su sistema actual para iniciar la conversación.

Explore nuestros servicios

welcome@engineer.company +45 50 10 61 84 Copenhague, Dinamarca

Llévese esta página

Tarjeta de contacto

Creado con esmero y cariño por D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Derechos de autor © 2025 – actualidad · Engineer ApS · Todos los derechos reservados.

Impreso desde https://engineer.company/es/glossary/privilege/