Engineer

Privilegium

🎟 en rättighet som ett system ger en person eller process

Vad det är

Ett privilegium är rätten att göra något som ett system annars förbjuder. Det innehas av en person eller av ett körande program, eftersom en auktoritet har bestämt att det ska ha den rätten: läsa en annan användares filer, binda en nätverksport under 1024, skapa konton eller köra en funktion i en databas. Oftast talar man om privilegier i plural, eftersom ett subjekt sällan har bara ett, och om privilegierade och oprivilegierade konton eller processer för att säga på vilken sida av gränsen något står. NIST:s ordlista lägger till den del som är lätt att glömma: privilegier beskriver ett subjekts auktoriserade beteende, och de definieras av en auktoritet och skrivs ned i en policy eller i regler.

Ordet står bredvid två andra. Åtkomstkontroll är själva beslutet om varje begäran, och ett privilegium är ett av de fakta som beslutet slår upp om den som ber om något. En behörighet är oftast samma idé sedd från resursens sida: filen säger vem som får läsa den, och läsaren har rätten att göra det. I svensk text glider orden lätt ihop, och principen som på engelska heter least privilege kallas både minsta privilegium och minsta behörighet.

I Linux har ordet en exakt betydelse. Manualsidan capabilities(7) beskriver den traditionella uppdelningen i Unix: privilegierade processer, vars effektiva användar-ID är 0 och som kallas root, går förbi alla behörighetskontroller i systemkärnan, medan alla andra processer prövas mot sina identitetsuppgifter (credentials). Sedan Linux 2.2 delar systemkärnan upp roots makt i separata enheter, förmågor (capabilities), som kan slås på och av en i taget. Ett program som bara behöver binda en låg port kan alltså få `CAP_NET_BIND_SERVICE` och ingenting annat. Sidan credentials(7) förklarar vems identitet som prövas. Det verkliga användar-ID:t säger vem som äger en process, det effektiva användar-ID:t är det som systemkärnan jämför mot behörigheterna, och det sparade set-user-ID:t låter ett program lägga undan en upphöjd identitet och ta upp den igen senare.

Databaser och organisationer använder samma ord för sina egna tilldelningar. En databasroll har privilegier på scheman, tabeller och funktioner. Ett administratörskonto har privilegier som ett vanligt personalkonto saknar. Idén är densamma på varje nivå: en rättighet, given med avsikt, att korsa en gräns som annars är stängd.

Använd begreppet i ett beslut

Använd ”Vad är ett privilegium?” i ett konkret beslut. Vi kan knyta begreppet till systemet ni bygger eller köper. Konton, lösenord och behörigheter ordnade så att alla har exakt den åtkomst deras arbete kräver, och de som slutar förlorar den samma dag.

Se våra tjänster Hör av er →

Granska antagandena bakom ordet

Använd förklaringen och det länkade arbetet för att formulera frågor till projektet. Vi kan undersöka antagandena tillsammans innan ett verktyg väljs. Skrev 1 000 RBAC-regler som säkrar Linux-servrar — 1,000 åtkomstregler, skrivna för att täppa till en säkerhetslucka

Skrev 1 000 RBAC-regler som säkrar Linux-servrar Hör av er →

Varför det spelar roll

De privilegier ett program har sätter storleken på den skada det kan göra när något går fel med det. Det är det uttalade syftet med förmågorna: ett program som har kompromitterats gör mindre skada med en smal uppsättning rättigheter än samma program gör när det körs som root. En webbserver med en bugg som tillåter kodkörning och som har root-privilegier lämnar över hela maskinen till den som hittade buggen. Samma bugg i en process som kan läsa sin egen katalog och binda en enda port når väldigt lite.

NIST skriver ned det här som minsta privilegium. Kontrollen AC-6 i SP 800-53 ber en organisation att bara tillåta den åtkomst som användarna, och de processer som agerar för dem, behöver för sina tilldelade uppgifter. Förstärkningarna av kontrollen gör det till rutinarbete: administratörer använder ett oprivilegierat konto för vardagliga uppgifter som e-post, privilegierade konton begränsas till de personer som behöver dem, användningen av privilegierade funktioner loggas, och privilegier ses över och tas bort när behovet av dem inte längre kan visas.

Standardvärdena börjar sällan där. I systemd körs en systemtjänst utan raden `User=` som root, och `NoNewPrivileges=`, inställningen som hindrar en tjänst och dess barnprocesser från att någonsin få mer genom `execve()`, är avstängd tills någon slår på den. Databasverktyg ansluter ofta som ägare till varenda tabell,

eftersom det var så databasschemat skapades. Nyanställda kopieras ofta från en befintlig kollegas konto, med rättigheter och allt. Varje sådant standardval är bekvämt den första dagen och dyrt den dag ett intrång sker.

Avvägningen är verklig. Att smalna av privilegier tar tid, eftersom någon måste ta reda på vad ett program faktiskt gör, och en uppsättning som är för smal får saker att gå sönder vid körning snarare än vid installation. Beslutet ett team står inför handlar oftast om var den tiden ska läggas först: på allt som är exponerat mot internet, på allt som rymmer data som folk skulle sakna och på de konton som kan ändra andra konton. För en köpare eller en revisor är den användbara frågan enkel och direkt: vem och vad har administratörsrättigheter här, och varför?

Så ser det ut i praktiken

Privilegier skars här ned i tre lager: tjänsterna på en värddator, en applikations inloggning i databasen och människorna i en organisation.

På företagets egen värd kördes tretton enhetsfiler med de privilegier som tjänstehanteraren ger som standard, och det är de flesta. Varje enhetsmall fick ett privilegietak låst vid enhetens egen nivå, ett filter för systemanrop, begränsade adressfamiljer och minne som är skrivbart eller körbart men aldrig båda. Härdningspoängen för dödmansgreppet gick från 9,6 till 1,5, för git-forgen från 8,3 till 1,5 och för de två enheterna för säkerhetskopiering från 9,6 till 2,3 respektive 2,5. Ett direktiv visade sig vara felstavat i tre mallar. Tjänstehanteraren hade loggat en okänd nyckel och startat enheten ändå, så nu körs varje enhet genom tjänstehanterarens egen verifierare vid varje konvergenskörning. Sidan skriver också ned vad poängen lämnar utanför: ett direktiv som får en enhet att gå sönder gör det vid körning.

På en applikationsplattform loggar API:t in i databasen som en roll med minsta möjliga behörighet som bara får anropa appens funktioner med EXECUTE, och funktionerna körs som SECURITY DEFINER med parametriserade databasfrågor. Rättigheter som nivå, roll, organisation och avgränsning per fartyg kontrolleras på nytt på servern vid varje anrop, och spärrarna i gränssnittet behandlas enbart som UX.

I en organisation samlades identiteter och åtkomst i M365, 1Password, Red Hat SSO och OKTA SSO, principen om minsta behörighet upprätthölls, och onboarding och offboarding standardiserades. Då återkallades åtkomst snabbt i

stället för att ligga kvar efter att någon hade gått vidare.

Var det går fel

En tjänst körs som root eftersom enhetsfilen tiger. Tjänstehanterarens standard för en systemtjänst är root, så en enhet utan raden `User=` har varje privilegium som maskinen har. Ge varje tjänst en egen användare, eller låt systemd tilldela en med `DynamicUser=`, som skapar en användare när enheten startar och släpper den när enheten stoppas.

En enda förmåga visar sig vara det mesta av root. `CAP_SYS_ADMIN` täcker så många operationer att dess egen manualsida kallar den överlastad och ”the new root”, den nya root. Ett team som ger den för att låsa upp en funktion har gett bort det mesta av root. Leta upp den specifika förmåga som programmet behöver och håll begränsningsmängden (bounding set) till den listan:

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

Med de inställningarna kan en tjänst som körs som vanlig användare binda port 80, och alla andra förmågor tas bort ur dess begränsningsmängd.

En process klättrar högre än där den startade. Ett program som körs som vanlig användare kan fortfarande köra en binär med set-user-ID eller en fil med förmågor kopplade till sig, och systemkärnan höjer då dess privilegier för den körningen. Försvarare känner igen det som en process vars effektiva användar-ID skiljer sig från det verkliga. `NoNewPrivileges=yes` stänger den vägen för en tjänst och för varje barnprocess den startar.

Rättigheter samlas på hög när folk byter jobb. Någon byter team och behåller den gamla åtkomsten bredvid den nya, och efter några år har kontot unionen av varje roll personen någonsin haft. AC-6 svarar på det med regelbunden översyn och AC-2 med kontohantering kopplad till omplaceringar och avgångar, så att ett byte av jobb också blir ett byte av åtkomst.

Applikationens egen inloggning kan göra allt. När ett API ansluter som ägare till databasschemat når en enda injektionsbrist varje tabell. En inloggningsroll som bara kan anropa de funktioner applikationen använder gör samma brist till ett mycket mindre problem.

Läs mer

- Privilege, in the NIST CSRC glossary — definitionen från CNSSI 4009, med anmärkningen att en auktoritet definierar privilegierna.
- capabilities(7) — hur Linux delar upp root i separata förmågor och hur trådar bär dem.
- credentials(7) — verkliga, effektiva, sparade och filsystemets användar-ID, och vad vart och ett avgör.
- systemd.exec(5) — `User=`, `DynamicUser=`, `NoNewPrivileges=` och inställningarna för förmågor.
- NIST SP 800-53 Rev. 5 — kontrollen AC-6, minsta privilegium, och förstärkningarna som gör den till rutin.

Ordlista

Välj utifrån era faktiska förutsättningar

Ta med driftsättningsvägen, driftsriskerna och dem som har jour. Vi kan avgränsa en förändring med kontroller, en väg tillbaka och tydligt ansvar.

Se vad det kostar Hör av er →

Gör begreppen praktiska

Fråga hur ”Vad är ett privilegium?” gäller i er organisation. Ett konkret exempel från ert nuvarande system räcker för att börja samtalet.

Se våra tjänster

welcome@engineer.company +45 50 10 61 84 Köpenhamn, Danmark

Ta med dig den här sidan

Visitkort

Omsorgsfullt skapad med kärlek av D.

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

Upphovsrätt © 2025 – nu · Engineer ApS · Alla rättigheter förbehållna.

Utskrivet från https://engineer.company/sv/glossary/privilege/