Engineer

Privilegium

🎟 en ret, et system giver en person eller en proces

Hvad det er

Et privilegium er lov til at gøre noget, som et system ellers forbyder, og det tilhører en person eller et kørende program, fordi en ansvarlig instans har besluttet det: at læse andre brugeres filer, at lytte på en port under 1024, at oprette konti eller at kalde en databasefunktion. I praksis taler man oftest om privilegier i flertal, fordi en bruger sjældent har kun ét, og om »privilegerede« og »uprivilegerede« konti og processer for at sige, hvilken side af grænsen de står på. NIST's ordliste tilføjer den del, der er nem at glemme: privilegier beskriver, hvad et subjekt har tilladelse til, og de fastlægges af en ansvarlig instans og skrives ned i en politik eller i regler.

Ordet står tæt på to andre. Adgangskontrol er selve afgørelsen af hver anmodning, og et privilegium er en af de oplysninger om den, der spørger, som afgørelsen slår op i. En tilladelse er som regel det samme set fra ressourcens side: filen siger, hvem der må læse den, og læseren har retten til det.

På Linux har ordet en præcis betydning. Manualsiden capabilities(7) beskriver den klassiske Unix-opdeling: privilegerede processer, hvis effektive bruger-id er 0, og som kaldes root, går uden om alle kernens tjek af tilladelser, mens alle andre processer bliver tjekket mod deres identitetsoplysninger. Siden Linux 2.2 har kernen delt roots magt op i separate enheder, kaldet capabilities, som kan slås til og fra hver for sig. Et program, der kun skal lytte på en lav port, kan derfor få `CAP_NET_BIND_SERVICE` og intet andet. Siden credentials(7) forklarer, hvis identitet der bliver tjekket: det reelle bruger-id siger, hvem der ejer processen, det effektive bruger-id er det, kernen sammenligner med tilladelserne, og det gemte set-user-ID lader et program lægge en hævet identitet fra sig og tage den op igen senere.

Databaser og organisationer bruger samme ord om deres egne tildelinger. En databaserolle har privilegier på skemaer, tabeller og funktioner, og en administratorkonto har privilegier, som en almindelig medarbejderkonto ikke har. Tanken er den samme på alle niveauer: en ret, givet med vilje, til at krydse en grænse, der ellers er lukket.

Brug begrebet i en beslutning

Brug »Hvad er et privilegium?« i en konkret beslutning. Vi kan knytte begrebet til det system, I bygger eller køber. Konti, adgangskoder og rettigheder sat op, så folk har præcis den adgang, jobbet kræver, og fratrådte mister den samme dag.

Se vores ydelser Kontakt os →

Undersøg antagelserne bag ordet

Brug forklaringen og det tilknyttede arbejde til at formulere spørgsmål til jeres projekt. Vi kan undersøge antagelserne sammen, før et værktøj vælges. Har skrevet 1.000 RBAC-regler til Linux-servere — 1,000 adgangsregler skrevet for at lukke et sikkerhedshul

Har skrevet 1.000 RBAC-regler til Linux-servere Kontakt os →

Hvorfor det betyder noget

De privilegier, et program har, bestemmer, hvor stor skaden bliver, den dag noget går i stykker. Det er det erklærede formål med capabilities: et kompromitteret program gør mindre skade med et smalt sæt rettigheder end det samme program, der kører som root. En webserver med en fejl, der tillader kodekørsel, og med root-rettigheder giver hele maskinen til den, der fandt fejlen. Den samme fejl i en proces, der kun kan læse sin egen mappe og lytte på én port, har en meget begrænset rækkevidde.

NIST kalder princippet mindste privilegium. Kontrol AC-6 i SP 800-53 beder en organisation om kun at tillade de adgange, som brugere og de processer, der handler på deres vegne, har brug for til deres opgaver. Udvidelserne gør det til daglig rutine: administratorer bruger en almindelig konto til hverdagsopgaver som mail, privilegerede konti forbeholdes dem, der har brug for dem, brugen af privilegerede funktioner logges, og privilegier gennemgås og fjernes, når behovet ikke længere kan dokumenteres.

Standardindstillingerne begynder sjældent der. I systemd kører en systemtjeneste uden en `User=` -linje som root, og `NoNewPrivileges=` , indstillingen, der forhindrer en tjeneste og dens underprocesser i at få flere rettigheder via `execve()` , er slået fra, indtil nogen slår den til. Databaseværktøjer forbinder ofte som ejer af alle tabeller, fordi det var sådan,

skemaet blev oprettet. Nye medarbejdere bliver ofte kopieret fra en kollegas konto med alle rettigheder. Hver standard er bekvem den første dag og dyr den dag, nogen bryder ind.

Afvejningen er reel. Det tager tid at indsnævre privilegier, fordi nogen skal finde ud af, hvad et program faktisk gør, og et sæt, der er for smalt, går i stykker under kørslen og ikke ved installationen. Beslutningen handler som regel om, hvor tiden skal bruges først: på alt, der vender ud mod internettet, på alt, der rummer data, folk vil savne, og på de konti, der kan ændre andre konti. For en køber eller en revisor er det nyttige spørgsmål enkelt: hvem og hvad har administratorrettigheder her, og hvorfor?

Sådan ser det ud i praksis

Tretten unit-filer på virksomhedens egen server kørte med de privilegier, som supervisoren giver som standard, og det er de fleste. Hver unit-skabelon fik et privilegieloft fastsat til unittens eget, et filter for systemkald, begrænsede adressefamilier og hukommelse, der kan være skrivbar eller eksekverbar, men aldrig begge dele. Hardening-scoren for dødmandsknappen faldt fra 9,6 til 1,5, git-forgen fra 8,3 til 1,5 og de to backup-units fra 9,6 til 2,3 og 2,5. Et direktiv viste sig at være stavet forkert i tre skabeloner, og supervisoren havde logget en ukendt nøgle og startet unitten alligevel. Derfor kører hver unit nu gennem supervisorens egen verifikator ved hver konvergering. Siden skriver også, hvad scoren lader ude: et direktiv, der ødelægger en unit, gør det under kørslen.

På en applikationsplatform logger API'et ind i databasen som en rolle med mindste privilegium, der kun kan køre EXECUTE på applikationens funktioner, som kører som SECURITY DEFINER med parametriserede databaseforespørgsler. Rettigheder som abonnementsniveau, rolle, organisation og skibstilknytning tjekkes igen på serveren ved hver anmodning, og spærringerne i brugerfladen regnes kun for UX.

I en hel organisation blev identiteter og adgange samlet på tværs af M365, 1Password, Red Hat SSO og OKTA SSO, mindste privilegium blev håndhævet, og on- og offboarding blev standardiseret, så adgang blev trukket tilbage hurtigt i stedet for at blive hængende, når nogen skiftede job.

Hvor det går galt

En tjeneste kører som root, fordi dens unit er tavs om brugeren. Supervisorens standard for en systemtjeneste er root, så en unit uden en `User=`-linje har alle maskinens privilegier. Giv hver tjeneste sin egen bruger, eller lad systemd tildele en med `DynamicUser=`, som opretter en bruger, når unitten starter, og frigiver den, når den stopper.

Én capability viser sig at være det meste af root. `CAP_SYS_ADMIN` dækker så mange operationer, at dens egen manual kalder den overlæsset og »the new root«. Et team, der giver den for at låse én funktion op, har givet det meste af root. Find den specifikke capability, programmet har brug for, og hold bounding-sættet til den liste:

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

De indstillinger lader en tjeneste, der kører som almindelig bruger, lytte på port 80, og alle andre capabilities fjernes fra dens bounding-sæt.

En proces kravler op over sit udgangspunkt. Et program, der kører som almindelig bruger, kan stadig køre en set-user-ID-fil eller en fil med capabilities tilknyttet, og så hæver kernen dens privilegier under kørslen. Forsvarere genkender det som en proces, hvis effektive bruger-id er et andet end det reelle. `NoNewPrivileges=yes` lukker den vej for en tjeneste og alle de processer, den starter.

Rettigheder hober sig op, når folk skifter job. En medarbejder skifter team og beholder den gamle adgang ved siden af den nye, og efter nogle år rummer kontoen summen af alle de roller, personen nogensinde har haft. AC-6 svarer med periodisk gennemgang og AC-2 med kontostyring, der er knyttet til flytninger og fratrædelser, så et nyt job også betyder ny adgang.

Applikationens eget login kan alt. Når et API forbinder som ejer af skemaet, når en enkelt injektionsfejl alle tabeller. En login-rolle, der kun kan kalde de funktioner, applikationen bruger, gør den samme fejl til et langt mindre problem.

Læs mere

- Privilege i NIST CSRC's ordliste — definitionen fra CNSSI 4009, med noten om, at en ansvarlig instans fastlægger privilegier.
- capabilities(7) — hvordan Linux deler root op i capabilities, og hvordan tråde bærer dem.
- credentials(7) — reelle, effektive, gemte og filsystem-bruger-id'er, og hvad hvert af dem afgør.
- systemd.exec(5) — `User=`, `DynamicUser=`, `NoNewPrivileges=` og indstillingerne for capabilities.
- NIST SP 800-53 Rev. 5 — kontrol AC-6, mindste privilegium, og udvidelserne, der gør det til rutine.

Ordliste

Vælg ud fra jeres faktiske vilkår

Medbring udrulningsforløbet, driftsrisiciene og dem, der har vagten. Vi kan afgrænse en ændring med kontroller, en vej tilbage og tydeligt ansvar.

Se, hvad det koster Kontakt os →

Gør begreberne praktiske

Spørg, hvordan »Hvad er et privilegium?« gælder i jeres organisation. Et konkret eksempel fra jeres nuværende system er nok til at begynde samtalen.

Se vores ydelser

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

Tag denne side med dig

Visitkort

Lavet med kærlighed af **D.**

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

Copyright © 2025 – nu · Engineer ApS · Alle rettigheder forbeholdes.

Udskrevet fra https://engineer.company/da/glossary/privilege/