Engineer

Systemkerne

kernen i et styresystem

Hvad det er

Systemkernen er det program i midten af et styresystem, der kører med processorens fulde rettigheder og styrer adgangen til hardwaren. Den afgør, hvilket program der får processoren som det næste, uddeler hukommelse og holder hvert programs hukommelse adskilt fra de andres, læser og skriver diske gennem filsystemer, sender pakker gennem netværksstakken og håndhæver reglerne for, hvem der må tilgå hvilken fil, port eller enhed. Alt andet kører i brugerrummet med færre rettigheder og beder kernen om at handle på dets vegne.

Den slags anmodninger hedder systemkald, og Linux' egen manual kalder systemkaldet den grundlæggende grænseflade mellem et program og kernen. Programmer laver dem sjældent selv: de kalder indpakningsfunktioner i et C-bibliotek som glibc, der sender argumenterne videre og oversætter kernens svar til en returværdi og en fejlkode. Når et program stopper med »Permission denied«, er ordlyden som regel kernens svar på et af disse kald, oversat af biblioteket.

Systemkerner adskiller sig i, hvor meget de rummer. Linux har drivere, filsystemer og netværksstakken inde i selve kernen og kan indlæse flere som moduler, mens den kører, en opbygning der som regel kaldes monolitisk. Hver driver kører dermed med kernens fulde rettigheder. En mikrokerne beholder kun det allermest nødvendige. seL4 stiller mekanismer til rådighed for adgang til det fysiske adresserum, afbrydelser og processortid og lader drivere, filsystemer og netværksstakke køre som almindelige programmer i brugertilstand; den verificerede 64-bit-udgave til RISC-V er på omkring 10.000 linjer kode, og et formelt bevis viser, at C-implementeringen gør det, specifikationen siger. macOS' kerne, XNU, er en hybrid, der kombinerer mikrokernen Mach med dele af FreeBSD.

En kerneopdatering træder normalt først i kraft, når maskinen starter op på den nye kerne, og kernens versionsnummer, for eksempel 6.12, er uafhængigt af udgivelsesnummeret på distributionen omkring den.

Brug begrebet i en beslutning

Brug »Hvad er en systemkerne?« i en konkret beslutning. Vi kan knytte begrebet til det system, I bygger eller køber. Daglig drift af servere, mail, konti og opdateringer, så det, alle er afhængige af, forbliver kedeligt.

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 drevet 10 PostgreSQL- og MS SQL-servere — 10 databaseservere udrullet og holdt i god drift

Har drevet 10 PostgreSQL- og MS SQL-servere Kontakt os →

Hvorfor det betyder noget

For en virksomhed afgør kernen tre spørgsmål: hvor adskilte arbejdsopgaverne reelt er, hvad en sikkerhedsrettelse koster i nedetid, og hvor sandheden om en maskine ligger.

Først isolation. En container er en gruppe almindelige processer, som kernen har placeret i navnerum, der giver hver gruppe sit eget billede af proces-id'er, netværksgrænseflader, monteringspunkter og brugere, og i kontrolgrupper, der fordeler processortid, hukommelse og diskens læsning og skrivning. Alle containere på en vært kører på værtens ene kerne, så en fejl i den kerne er en fejl under dem alle på én gang, og et image med sine egne biblioteker afhænger stadig af de kernefunktioner, værten har. En virtuel maskine starter sin egen kerne. Til ét teams tjenester er containere en rimelig grænse; til kunder, der ikke stoler på hinanden, bør hver kunde have sin egen kerne.

Dernæst rettelser. En opdateret kernepakke på disken ændrer intet, før maskinen starter op på den, og på en server er genstarten prisen. Live-patching findes, fordi brugere, som kernens dokumentation udtrykker det, nødig genstarter et system: den omdirigerer enkelte funktioner til rettede udgaver, mens systemet kører. Kun funktioner, som kernen kan spore, kan rettes på den måde, så live-patching køber tid til en hastende rettelse i stedet for at erstatte genstarter. En plan med en fast, indøvet genstart koster mindre end en, der går ud fra, at genstarten aldrig bliver nødvendig.

Til sidst ligger den aktuelle tilstand i kernen. En firewallfil beskriver regler; reglerne i kernens pakkefilter afgør, hvad der sker med en pakke. En fil i `/etc/sysctl.d` beskriver en indstilling; værdien under `/proc/sys` er den, der gælder. En tjeneste kan melde sig sund, uanset om dens handling nogensinde nåede kernen. Tjek, der læser kernens egen tilstand, fanger de fejl, som en gennemgang af konfigurationsfiler ville godkende.

Sådan ser det ud i praksis

En hærdningsgennemgang af en produktionsvært i august 2026 spurgte, om dens beskyttelser mod brute force på SSH virkede. Hver af dem var konfigureret og stod i hver rapport, så gennemgangen efterprøvede dem mod hvad kernen rent faktisk gør ved en pakke frem for mod de filer, der beskriver, hvad der burde ske. Alle tre havde været uvirksomme, siden værten blev bygget. Blokeringsenhedens (jail) regler gjaldt port 22, efter at SSH var flyttet til port 1986. En hastighedsbegrænsning i firewallen lå under en bredere regel, der matchede først. Blokeringshandlingen kaldte et pakkefilterprogram, som pakkesystemet kun anbefaler, så jailen startede, talte og besluttede og fejlede så i det øjeblik, det forsøgte at blokere. Der kom to regler ud af det: en sikkerhedskontrol får en assertion frem for en kommentar, og en firewall verificeres på reglernes placering frem for på deres tilstedeværelse.

Blokeringsvejen fik derefter en test, der kører ved hver konvergering, altså hver gang konfigurationen anvendes. Hver kørsel blokerer en adresse fra en blok, der er reserveret til dokumentation, læser den resulterende regel tilbage fra kernens pakkefilter og ophæver derefter blokeringen i et oprydningstrin, der kører uanset udfaldet. Det er tilbagelæsningen, der gør det til en test: tjenestens egen status ville melde succes for en blokering, der ikke gav nogen regel. En reserveret adresse tilhører ikke noget rigtigt netværk, så en blokering, som en fejlet kørsel efterlader, lukker ikke nogen ude. Prisen er nogle få sekunder pr. kørsel og en firewallregel, der skrives til og trækkes tilbage fra produktion hver gang.

En vært på 464 MB uden plads til en metrikstak fik syv skrivebeskyttede roller, der gengiver maskinens aktuelle tilstand som Markdown; dens faktabillede beregner processorudnyttelsen ud fra to stikprøver af kernens egen tæller. Sikkerhedsrapporten udskriver firewallreglerne i matchrækkefølge ved siden af de aktive blokeringsregler. For alle syv roller gælder, at et tal kun bliver udskrevet, hvis kørslen har målt det. Rapporterne har afsløret to døde sikkerhedskontroller og en uerklæret lyttende port.

Hvor det går galt

En kerneindstilling, der er ændret i hånden, springer tilbage ved næste genstart. `sysctl -w` skriver værdien ind i den kørende kerne og kun dér, så den holder til næste genstart. Den omvendte fejl er lige så almindelig: en fil under `/etc/sysctl.d` læses først ved opstart, så ændringen venter ubemærket på en senere genstart. Skriv indstillingen i en fil, indlæs den nu, og læs den gældende værdi tilbage:

```
echo 'vm.swappiness = 10' | sudo tee /etc/sysctl.d/90-local.conf
sudo sysctl --system
sysctl vm.swappiness
```

En CPU-graf ligger flad, mens belastningen topper. Linjen `cpu` i `/proc/stat` tæller tid siden opstart i enheder af USER_HZ (hundrededele sekund på de fleste arkitekturer), så én aflæsning giver gennemsnittet siden opstart, og det rokker sig knap på en maskine, der har kørt i måneder. Læs linjen to gange, og del forskellene:

```
head -1 /proc/stat; sleep 5; head -1 /proc/stat
```

Travl tid er ændringen i user, nice, system, irq og softirq, hviletid er ændringen i idle og iowait, og udnyttelsen er travl tid delt med travl tid plus hviletid. Manualen kalder iowait upålidelig, så brug den som et fingerpeg og ikke som grundlag for en alarm.

En nedbrudsrapport kommer retur med en anmodning om at genskabe fejlen på en kerne, der ikke er plettet. Kernen markerer sig selv som plettet (»tainted«), når der sker noget, som kan få betydning for en senere undersøgelse, for eksempel at et proprietært, eksternt bygget eller usigneret modul bliver indlæst, og dokumentationen advarer om, at udviklere ofte ignorerer fejlrapporter fra plettede kerner. Mærket bliver siddende til næste genstart, også efter at modulet er fjernet igen. Læs det, før du indsender rapporten:

```
cat /proc/sys/kernel/tainted
```

Nul betyder, at kernen er uplettet; ethvert andet tal er en sum af flag, blandt andre `1` for et proprietært modul, `4096` for et modul bygget uden for kernens kildetræ og `8192` for et usigneret modul. Genskab fejlen efter en opstart uden det modul, der satte flaget.

Et nyt netværkskort dukker op i hardwarelisten, men systemet tilbyder ikke en grænseflade til det. Drivere er en del af kernen, og en kerne, der er ældre end hardwaren, kan mangle en driver til den. `lspci -k` viser for hver enhed, hvilken kernedriver der er i brug, og hvilke moduler der kan håndtere enheden; en enhed uden linjen »Kernel driver in use« har ikke fået en driver bundet. Installer en nyere kerneserie fra distributionen frem for at bygge leverandørens modul i hånden: et eksternt bygget modul skal bygges om til hver ny kerne, og når det indlæses, bliver kernen plettet.

Læs mere

- The Linux kernel user's and administrator's guide — kerneprojektets egen håndbog i at drive Linux.
- syscalls(2) — alle Linux' systemkald med den kerneversion, der indførte hvert af dem.
- namespaces(7) — de typer navnerum, kernen tilbyder, og hvordan containere bygges af dem.
- proc_stat(5) — hvert felt i `/proc/stat`, inklusive advarslen om iowait.
- Tainted kernels — hvert taint-flag, og hvordan tallet afkodes.
- seL4 FAQ — hvad en mikrokerne lader ligge uden for kernen, og hvilke antagelser seL4's bevis hviler på.

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 en systemkerne?« 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/kernel/