Engineer

Konfiguration

de indstillinger, der afgør, hvordan et system opfører sig

Hvad det er

Konfiguration er alt det, et program læser uden for sin egen kildekode for at vide, hvordan det skal køre på et bestemt sted: hvilken port det lytter på, hvilken database det forbinder til, hvem der må logge ind, og hvilke funktioner der er slået til. I daglig tale siger de fleste udviklere bare config, og filerne kaldes config-filer eller konfigurationsfiler. NIST definerer i SP 800-128, sin vejledning i sikkerhedsfokuseret konfigurationsstyring, begrebet som de mulige tilstande, parametre og specifikationer, som et informationssystem eller en del af det kan beskrives eller indrettes med. Det ord, man skal hæfte sig ved, er beskrives: en konfiguration er en beskrivelse, og det kørende system kan sagtens afvige fra den.

Konfigurationen kommer sjældent fra ét sted. Standardværdier er kompileret ind i binærfilen, en hovedfil ligger på disken med en mappe af drop-in-fragmenter ved siden af, og miljøvariabler og kommandolinjeflag kommer til ved opstart. Programmet fletter det hele sammen til ét sæt værdier efter en regel for, hvad der sker, når to kilder sætter samme nøgle, og den regel er programmets egen. OpenSSH-serveren beholder for de fleste nøgleord den første værdi, den møder, mens andre værktøjer lader en senere fil overskrive en tidligere. De samme filer læst i en anden rækkefølge giver et andet system.

Formaterne skifter: INI-lignende linjer, TOML, YAML, JSON og formater, som kun ét program bruger. TOML 1.0.0 sigter efter at være et minimalt format til konfigurationsfiler, mens YAML 1.2.2 er et generelt datasprog, som konfigurationsfiler låner, og hvert format har sine egne regler for, hvordan tekst bliver til værdier.

Brug begrebet i en beslutning

Brug »Hvad er konfiguration?« i en konkret beslutning. Vi kan knytte begrebet til det system, I bygger eller køber. Hele opsætningen skrevet ned som filer, så den kan genskabes præcist og aldrig kun findes i én persons hoved.

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

Det er konfigurationen, der lader ét build tjene mange steder. Samme program kører på en bærbar, på en staging-vært og i produktion, og forskellen mellem de tre ligger i konfigurationen. The Twelve-Factor App, en metode til tjenester, der skal deployes, trækker grænsen netop her: konfiguration er alt, hvad der sandsynligvis varierer mellem deploys, for eksempel adresser på databaser og andre tjenester, adgangsoplysninger til eksterne tjenester og hver deploys værtsnavn. Metoden kræver en streng adskillelse af konfiguration og kode og giver en prøve, der kan bruges overalt: kan kodebasen gøres open source når som helst, uden at en eneste adgangsoplysning kompromitteres?

Samme tekst sætter også den modsatte grænse. Intern opsætning, der er ens ved hver deploy, for eksempel et webframeworks tabel over ruter, falder uden for definitionen og hører hjemme i koden. En værdi, der er den samme overalt, er kode forklædt som en indstilling, og en fil giver den kun et ekstra sted at være forkert.

Konfiguration er også der, hvor et system erklærer sin sikkerhed: hvem der må logge ind, hvilke porte der svarer, og hvad en firewall lukker igennem. En vært er præcis så hærdet som de værdier, den faktisk bruger, og mellem filen og de værdier står parseren, prioritetsreglerne, de indkompilerede standardværdier, programversionen og miljøet. Hver af dem kan ændre udfaldet, mens filen står uændret, så den pålidelige vane er at spørge programmet, hvad det bruger, og regne svaret for den egentlige konfiguration.

Sådan ser det ud i praksis

Dette site konfigureres i én TOML-fil, hugo.toml, der blandt meget andet fastlægger sitets sprog, taksonomier og mounts. Én tabel i filen viser, hvor meget få linjer kan bestemme:

```
[frontmatter]
  date = ["date", ":git", ":fileModTime"]
  lastmod = ["lastmod", ":git", ":fileModTime"]
```

Hver liste er en opslagsrækkefølge: først en dato skrevet i selve siden, så datoen for den seneste commit, der rørte filen, og til sidst filens ændringstidspunkt. Et flag øverst i filen, `enableGitInfo = true`, giver Hugo adgang til git-historikken overhovedet. Kommentaren over tabellen forklarer det første element: står feltet fra front matter ikke selv i listen, springer Hugo en `lastmod` skrevet i en side over i stilhed, og sitet bygger som normalt.

Den tabel er rettelsen bag et sitemap, hvor næsten alle sider bar samme tidsstempel. Sitemappet meldte 172 af sitets 176 sider ændret på nøjagtig samme tidspunkt, fordi datoen blev taget fra filens ændringstid, og indholdseksporten skriver alle filer på ny ved hver kørsel. Fallback-kæden flyttede datokilden til versionshistorikken. På det dokumenterede implementeringstrin berørte en ny synkronisering efter en måned med indholdsændringer otte af 186 filer.

På en server går samme pointe igen i hærdningen af SSH med validering i tre trin. En automatiseringsrolle havde i måneder skrevet SSH-dæmonens konfiguration, da en adgangsrapport i september 2026 spurgte dæmonen selv, hvilke værdier den brugte, og to af svarene afveg fra rollens fil. Styresystemets egne standardværdier stod aktive før drop-in-filen, og for netop de to indstillinger gælder den værdi, der læses først. Løsningen blev et præfiks i filnavnet, så drop-in-filen sorteres før leverandørens. Rundt om den validerer rollen nu i tre trin: kandidatfilen syntakstjekkes før installation, den samlede konfiguration tjekkes bagefter, og dæmonens effektive værdier læses tilbage med 24 direktiver kontrolleret. To direktiver er bevidst udeladt, fordi dæmonen ikke længere implementerer dem.

Den anden side af samme fejl ses i en gennemgang af tre SSH-forsvar mod brute force fra august 2026. De tre stod i konfigurationen og i hver rapport, men ingen af dem havde grebet ind en eneste gang, siden værten blev sat op. Blokeringsjailen var sat op til port 22, men SSH-dæmonen kørte på port 1986. En hastighedsbegrænsning i firewallen stod under en bredere regel, der fangede trafikken først. Blokeringshandlingen kaldte en binærfil til pakkefiltrering, som pakkesystemet anbefaler uden at kræve den, og på denne vært kunne programmet ikke findes. Gennemgangen læste det kørende system frem for

filerne, og den gav to nye regler: en sikkerhedskontrol får en assertion i stedet for en kommentar, og en firewall verificeres på, hvor en regel står, frem for på, om den findes.

Hvor det går galt

Hver fejl herunder er en fil, der er rigtig som tekst og forkert som adfærd, og til hver hører et tjek, der læser adfærden.

- Den forkerte kilde vinder. OpenSSH's manual for serverkonfigurationen siger, at for hvert nøgleord bruges den først indlæste værdi, medmindre andet er angivet. En leverandørstandard, der læses før din drop-in-fil, slår den derfor, og i et værktøj, hvor den sidste værdi vinder, er det omvendt. Læs prioritetsreglen i programmets egen manual, og spørg så programmet: `sshd -t` tjekker, at en konfiguration er gyldig, og `sshd -T` udskriver den effektive.
- En TOML-nøgle havner i den forkerte tabel. I TOML 1.0.0 hører nøgle-værdi-parrene under en `[table]` -overskrift til den tabel indtil næste overskrift eller filens slutning, og indrykning ignoreres. En nøgle, der tilføjes nederst i filen, ender i den sidste tabel, også selv om den står helt ude til venstre. Placér tabellerne til sidst og de løse nøgler over dem. Det er ugyldigt at definere en nøgle eller en tabel to gange, så en dublet er en fejl, der skal rettes, og aldrig en måde at overskrive på.
- En YAML-værdi uden anførselstegn skifter type. Med kerneskemaet, som YAML 1.2.2 anbefaler som standard, bliver en værdi uden anførselstegn, der passer på et af skemaets mønstre, til et tal, en boolsk værdi eller null: `1.10` bliver kommatallet 1,1, `0o17` heltallet 15, `0123` kommer tilbage som 123, `FALSE` er en boolsk værdi, og `~` er null. Sæt anførselstegn om alle værdier, der skal forblive tekst, først og fremmest versionsnumre og id'er med foranstillede nuller. Specifikationen regner også gentagne nøgler i en mapping blandt de forhold, der efterlader en indlæsning ufuldstændig, så en nøgle skrevet to gange er en fejl.
- En indstilling ser aktiv ud og gør intet. En regel, som en tidligere regel overskygger, et direktiv, en nyere udgave har droppet, en port, tjenesten for længst har forladt: alle står de i filen og ser tilsigtede ud. Jailen på port 22 og hastighedsbegrænsningen under den bredere regel så begge sunde ud i hver status. Et tjek, der læser hver fil for sig, godkender dem, så kontrollér rækkefølge og tilstedeværelse mod det kørende system.
- En hemmelighed rejser med filen. En adgangskode i en konfigurationsfil følger filen, hvorhen den går: ind i repositoryet, ind i backupperne, ind i fejlrapporten. Twelve-Factor lægger konfigurationen i miljøvariabler blandt andet, fordi de i modsætning til konfigurationsfiler kun har ringe risiko for ved et uheld at blive tjekket ind i kodens repository.

Læs mere

- NIST CSRC Glossary: configuration — alle de definitioner, NIST's publikationer giver begrebet, hver med sit kildedokument.
- TOML v1.0.0 — hele specifikationen, kort nok til at læse i ét stræk.
- YAML Ain't Markup Language, version 1.2.2 — specifikationen; kapitel 10 rummer kerneskemaet, der afgør, hvad en værdi uden anførselstegn bliver til.
- The Twelve-Factor App: Config — argumentet for at holde deploy-specifikke værdier ude af koden, med open source-prøven.
- sshd_config(5) — OpenSSH-serverens konfigurationsmanual med reglen om første værdi, Include og Match.
- sshd(8) — dæmonens manual, hvor `-t` tjekker en konfiguration, og `-T` udskriver den effektive.

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 konfiguration?« 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/configuration/