Engineer

Konfigurasjon

innstillingene som avgjør hvordan et system oppfører seg

Hva det er

Konfigurasjon er alt et program leser utenfra sin egen kildekode for å bestemme hvordan det skal kjøre på ett bestemt sted: porten det lytter på, databasen det kobler seg til, kontoene det slipper inn, funksjonene det slår på. I daglig tale blir ordet nesten alltid forkortet til config, og filene som bærer den, kalles konfigurasjonsfiler eller config-filer. NIST definerer begrepet i SP 800-128, veiledningen for sikkerhetsfokusert konfigurasjonsstyring, som «the possible conditions, parameters, and specifications with which an information system or system component can be described or arranged», altså de mulige tilstandene, parameterne og spesifikasjonene et informasjonssystem eller en systemkomponent kan beskrives eller settes opp med. Ordet å holde fast ved er beskrives: en konfigurasjon er en beskrivelse, og systemet kan være enig i den, eller la være.

Et program henter sjelden konfigurasjonen sin fra ett sted. Standardverdier er kompilert inn i binærfilen, en hovedfil ligger på disk med en katalog av drop-in-fragmenter ved siden av, og miljøvariabler og flagg på kommandolinjen kommer til ved oppstart. Programmet slår disse sammen til ett sett verdier, med en regel for hva som skjer når to kilder setter den samme nøkkelen, og den regelen tilhører hvert enkelt program. OpenSSH-serveren beholder den første verdien den leser for de fleste nøkkelord, mens andre verktøy lar en senere fil overstyre en tidligere. De samme filene lest i en annen rekkefølge gir et annet system.

Formatene varierer: linjer i INI-stil, TOML, YAML, JSON og formater som bare ett program bruker. TOML 1.0.0 sikter mot å være «a minimal configuration file format», et minimalt format for konfigurasjonsfiler, mens YAML 1.2.2 er et generelt dataspråk som konfigurasjonsfiler låner, og hvert av dem har sine egne regler for hvordan tekst blir til verdier.

Bruk begrepet i en beslutning

Bruk «Hva er konfigurasjon?» i en konkret beslutning. Vi kan knytte begrepet til systemet dere bygger eller kjøper. Hele oppsettet skrevet ned som filer, slik at det kan gjenskapes nøyaktig og aldri bare finnes i hodet på én person.

Se tjenestene våre Ta kontakt →

Undersøk antakelsene bak ordet

Bruk forklaringen og det tilknyttede arbeidet til å formulere spørsmål for prosjektet. Vi kan undersøke antakelsene sammen før et verktøy velges. Skrev 1 000 RBAC-regler for å sikre Linux-servere — 1,000 tilgangsregler skrevet for å tette et sikkerhetshull

Skrev 1 000 RBAC-regler for å sikre Linux-servere Ta kontakt →

Hvorfor det er viktig

Konfigurasjon er måten ett bygg betjener mange steder på. Det samme programmet kjører på en laptop, en testvert og i produksjon, og det som skiller dem, er konfigurasjonen. The Twelve-Factor App, en metode for tjenester som skal rulles ut, trekker grensen der: config er «everything that is likely to vary between deploys», alt som sannsynligvis varierer mellom utrullinger, som henvisninger til tjenester appen støtter seg på, påloggingsdata til eksterne tjenester og vertsnavnet til hver utrulling. Metoden krever et strengt skille mellom konfigurasjon og kode, og den tilbyr en test som er verdt å bruke overalt: om «the codebase could be made open source at any moment, without compromising any credentials», altså om kodebasen kunne blitt gjort til åpen kildekode når som helst uten å avsløre noen påloggingsdata.

Den samme teksten markerer den motsatte kanten. Intern oppkobling som er lik i hver utrulling, som tabellen over ruter i et webrammeverk, faller utenfor definisjonen og er «best done in the code», best gjort i koden. En verdi som er den samme i hver utrulling, er kode utkledd som en innstilling, og en fil gir den bare enda et sted å være feil.

Konfigurasjon er også der et system erklærer sikkerheten sin: hvem som får logge inn, hvilke porter som svarer, hva en brannmur slipper gjennom. En vert er nøyaktig så herdet som verdiene den faktisk bruker, og mellom en fil og de verdiene står parseren, reglene for forrang, de kompilerte standardverdiene, programversjonen og miljøet. Hver av dem kan endre utfallet mens filen står uendret, så den pålitelige vanen er å spørre programmet hva det bruker, og regne det svaret som konfigurasjonen.

Slik ser det ut i praksis

Dette nettstedet er konfigurert i én TOML-fil, hugo.toml, som blant mye annet fastsetter nettstedets språk, taksonomier og monteringer. Én tabell i hugo.toml viser hvor mye noen få linjer kan avgjøre:

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

Hver liste er en oppslagsrekkefølge: en dato skrevet i selve siden, så datoen for den siste commiten som rørte filen, så filens endringstidspunkt. Et flagg nær toppen av filen, `enableGitInfo = true`, lar Hugo lese git-historikken i det hele tatt. Kommentaren over tabellen forklarer den første oppføringen: det bokstavelige feltet i front matter må navngis, ellers blir en `lastmod` skrevet i en side ignorert i det stille mens nettstedet bygges som vanlig.

Den tabellen er rettelsen som beskrives i reparasjonen av et sitemap der de fleste sidene delte ett tidsstempel. Sitemapen hadde fortalt søkemotorene at 172 av nettstedets 176 sider var sist endret i det samme øyeblikket. Datoen kom fra filens endringstidspunkt, den siste oppføringen i listene over, og den oppføringen sier lite om innholdet når innholdseksporten lager hver fil på nytt hver gang den kjører. Reservekjeden flyttet kilden til versjonshistorikken. På det dokumenterte implementeringstrinnet berørte en ny synkronisering etter en måned med innholdsendringer åtte filer av 186.

På en server går den samme lærdommen gjennom herdingen av SSH med validering i tre trinn. En automatiseringsrolle hadde skrevet konfigurasjonen til SSH-daemonen i månedsvis da en tilgangsrapport i september 2026 leste daemonens effektive innstillinger og fant at to av dem var uenige med rollens fil. Operativsystemets egne standardverdier lå over drop-in-filen, og for de to innstillingene vinner den første forekomsten, så rettelsen var et filnavnprefiks som sorteres foran leverandørens. Rundt den validerer rollen nå i tre trinn: kandidatfilen syntakssjekkes før installasjon, den sammensatte konfigurasjonen sjekkes etterpå, og de effektive verdiene leses tilbake med 24 direktiver kontrollert. To direktiver ble utelatt med vilje, fordi daemonen ikke lenger implementerer dem.

Tre SSH-vern mot brute force som aldri virket viser feilen fra den andre siden. Da en gjennomgang i august 2026 testet dem, var alle tre konfigurert, alle tre dukket opp i hver rapport, og alle tre hadde vært virkningsløse siden verten ble bygget. Jailen for utestenging overvåket port 22, mens daemonen var flyttet til 1986. En ratebegrensning i brannmuren lå under en bredere regel som tok trafikken først. Utestengingshandlingen kalte et pakkefilterprogram som pakkesystemet anbefaler uten å kreve det, og på denne verten kunne programmet ikke finnes. Gjennomgangen sjekket det kjørende systemet i stedet for filene, og to regler kom ut av det: en sikkerhetskontroll får en automatisk sjekk i stedet for en kommentar, og en brannmur verifiseres ut fra hvor en regel står, og ikke ut fra om den finnes.

Hvor det går galt

Hver feil nedenfor er en fil som er riktig som tekst og feil som oppførsel, og hver av dem har en sjekk som leser oppførselen.

- Kilden som vinner, er en annen enn du trodde. Manualen for OpenSSH-serverens konfigurasjon sier: «Unless noted otherwise, for each keyword, the first obtained value will be used.» For hvert nøkkelord brukes altså den første verdien som blir lest, med mindre noe annet er oppgitt. En standardverdi fra leverandøren som leses før drop-in-filen din, slår den derfor, og et verktøy der den siste verdien vinner, snur den samme oppbygningen. Les regelen for forrang i programmets egen manual, og spør så programmet: `sshd -t` sjekker at en konfigurasjon er gyldig, og `sshd -T` skriver ut den effektive.
- En TOML-nøkkel havner i feil tabell. I TOML 1.0.0 hører parene under en `[table]`-header til den tabellen «until the next header or EOF», fram til neste header eller slutten av filen, og innrykk blir oversett. En nøkkel som legges til nederst i en fil, blir med i den siste tabellen selv når den står helt til venstre. Hold tabellene til slutt og de vanlige nøklene over dem. Å definere en nøkkel eller en tabell to ganger er ugyldig, så et duplikat er en feil som skal rettes, aldri en måte å overstyre på.
- En vanlig YAML-verdi skifter type. Under kjerneskjemaet, som YAML 1.2.2 anbefaler som standard, blir en verdi uten anførselstegn som passer ett av mønstrene, et tall, en boolsk verdi eller null: `1.10` er flyttallet 1,1, `0o17` er heltallet 15, `0123` kommer tilbake som 123, `FALSE` er en boolsk verdi og `~` er null. Sett anførselstegn rundt hver verdi som må forbli tekst, versjonsstrenger og identifikatorer med innledende nuller først. Spesifikasjonen regner også gjentatte nøkler i en mapping blant måtene en innlasting ender ufullstendig på, så en nøkkel skrevet to ganger er en defekt.

- En innstilling ser aktiv ut og gjør ingenting. En regel som skygges av en tidligere regel, et direktiv en nyere utgave har fjernet, en port tjenesten for lengst har forlatt: hver av dem blir stående i filen og ser tilsiktet ut. Jailen på port 22 og ratebegrensningen under en bredere regel ble begge vist som friske. En sjekk som leser hver fil for seg, slipper dem gjennom, så kontroller rekkefølge og tilstedeværelse mot det kjørende systemet.
- En hemmelighet reiser med filen. Et passord i en konfigurasjonsfil går dit filen går: til repoet, til sikkerhetskopiene, til feilrapporten. The Twelve-Factor App holder konfigurasjon i miljøvariabler delvis fordi det, i motsetning til konfigurasjonsfiler, er «little chance of them being checked into the code repo accidentally», liten sjanse for at de ved et uhell blir sjekket inn i repoet.

Les mer

- NIST CSRC Glossary: configuration — hver definisjon NIST-publikasjonene gir begrepet, med dokumentet den kommer fra.
- TOML v1.0.0 — hele spesifikasjonen, kort nok til å leses i én økt.
- YAML Ain't Markup Language, version 1.2.2 — spesifikasjonen, der kapittel 10 rommer kjerneskjemaet som avgjør hva en vanlig verdi blir.
- The Twelve-Factor App: Config — argumentet for å holde verdier som er knyttet til én utrulling, utenfor koden, med testen med åpen kildekode.
- sshd_config(5) — manualen for konfigurasjonen av OpenSSH-serveren, med regelen om første verdi, Include og Match.
- sshd(8) — manualen for daemonen, der `-t` sjekker en konfigurasjon og `-T` skriver ut den effektive.

Ordliste

Velg ut fra deres faktiske rammer

Ta med utrullingsløpet, driftsrisikoene og dem som har vakt. Vi kan avgrense en endring med kontroller, en vei tilbake og tydelig ansvar.

Se hva det koster Ta kontakt →

Gjør begrepene praktiske

Spør hvordan «Hva er konfigurasjon?» gjelder i organisasjonen deres. Et konkret eksempel fra dagens system er nok til å starte samtalen.

Se tjenestene våre

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

Ta med deg denne siden

Visittkort

Laget med kjærlighet av D.

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

Opphavsrett © 2025 – nå · Engineer ApS · Alle rettigheter forbeholdt.

Skrevet ut fra https://engineer.company/nb/glossary/configuration/