Engineer

Forge

en fælles server til koderepositories og samarbejde

Hvad det er

En forge, også kaldet en softwareforge, en kodeforge eller en git-forge, er den server, et team aftaler at behandle som hjemsted for sine repositories, sammen med de tjenester, den kører omkring dem. Enhver fuld klon af et Git-repository bærer hele historikken, så i princippet står ingen kopi over en anden. I praksis har et team brug for ét sted at pushe til, ét sted, der afgør, hvem der må læse og skrive, og ét sted, hvor samtalen om koden foregår. Det sted er forgen.

Nedenunder opbevarer en forge bare repositories, den form, der kun rummer historikken og ingen udtjekkede filer, og den taler en eller flere af Gits overførselsprotokoller. Git-bogen nævner fire: den lokale protokol til en mappe på samme maskine, HTTP, SSH og Gits egen protokol, en dæmon på port 9418 helt uden autentificering. Alt andet bygges ovenpå, og mængden varierer betydeligt.

Forgejos dokumentation beskriver den fulde ende af skalaen: issue-sporing, pull requests, en indbygget wiki, et pakkeregister til mange sprogøkosystemer, Forgejo Actions til at køre jobs, spejling af repositories, migrering fra andre tjenester, beskyttelse af grene og tags samt webhooks. Soft Serve ligger i den anden ende. Projektet beskriver sig selv som en Git-server til kommandolinjen, som man kan hoste selv: en enkelt binærfil ved navn `soft`, der serverer repositories over SSH, HTTP eller Git-protokollen, styrer adgang med offentlige SSH-nøgler og understøtter samarbejdspartnere, private repositories og webhooks. Begge er forges; forskellen er, hvor meget af teamets arbejde de forsøger at rumme.

Forges er også begyndt at tale sammen. ForgeFed, en føderationsprotokol for platforme til hosting af og samarbejde om softwareprojekter, bygget som en udvidelse af ActivityPub, modellerer repositories, ticket-trackere, patch-trackere,

projekter og teams som aktører, der sender aktiviteter til hinanden. Målet er et fælles sprog på tværs af netværk af forges og værktøjer til projektstyring, så en person på én server kan oprette en ticket mod et repository på en anden, og så et repository kan flytte mellem servere, mens den gamle kopi arkiveres som skrivebeskyttet og peger på sit nye hjem.

Brug begrebet i en beslutning

Brug »Hvad er en forge?« i en konkret beslutning. Vi kan knytte begrebet til det system, I bygger eller køber. Testet, udgivet og til at rulle tilbage, så ændringer når kunderne en tirsdag eftermiddag i stedet for sent en fredag aften.

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 automatiseret GIS-SaaS-udrulning med CI/CD — udrulning, data og rapportering, alt sammen uden manuelt arbejde

Har automatiseret GIS-SaaS-udrulning med CI/CD Kontakt os →

Hvorfor det betyder noget

Forgen er stedet, hvor regler kan håndhæves. En regel, der bor på en udviklers maskine, gælder kun, så længe den maskine er sat korrekt op, men en hook på serveren kører ved hvert push. Gits `pre-receive` -hook læser hver pushet ref som et gammelt id, et nyt id og et navn på standard input, og afslutter den med en status forskellig fra nul, bliver ingen refs opdateret. Grenbeskyttelse, krævede reviews og krav om signerede commits på større forges forfiner samme idé: den fælles kopi afviser det, teamet har aftalt ikke at tage imod.

At vælge en forge er også at vælge, hvem der har koden. En hostet tjeneste påtager sig driftsomkostningerne og oppetiden, og til gengæld ligger koden, adgangslisten og hver eneste diskussion på en andens hardware og på en andens vilkår. Selvhosting vender byttehandlen om. Begge valg har en pris, og forskellen betyder mest for en virksomhed, der fortæller sine kunder, hvor deres data befinder sig.

En forge rummer også data, som Git ikke rummer. Commits og tags følger med hver klon, mens issues, review-tråde, adgangslister og hooks på serversiden alene tilhører forgen. Den opdeling afgør, hvad en backup skal dække, og hvad en flytning mellem forges skal have med.

Sådan ser det ud i praksis

Virksomheden kører sin egen forge, og tre projekter på dette site viser, hvad det krævede.

Virksomhedens kildekode lå tidligere hos en tredjeparts hostingtjeneste, og projektet, hvor virksomhedens egen git-forge blev sat i drift, tager udgangspunkt i, at det at køre en forge betyder at køre en forge: autentificering, adgangskontrol, lagring, backups og et offentligt ansigt udadtil. Forgen er Soft Serve, en enkelt binærfil overvåget af styresystemet og valgt, fordi den er SSH-først uden administrativ webgrænseflade. Den er privat som standard: anonym adgang afvises, adgang uden nøgle afvises, og hvert erklæret repository er markeret som privat. SSH-lytteren binder kun til loopback-grænsefladen på port 23231 og nås udefra gennem en jump host. En multiplexer, der ville lægge HTTPS og git over SSH på én offentlig port, er skrevet og står slået fra, fordi en lytter, som ingen bruger, er angrebsflade. Resultatet siges ligeud: virksomheden hoster sin egen kode på sin egen hardware.

Dette sites eget repository bor der: dets instruktioner udpeger den selvhostede forge, Soft Serve over SSH, som det kanoniske remote, kalder den ældre hostede kopi et forældet spejl og siger, at der ikke findes hostet CI. Kvalitetsporten kører lokalt og i git-hooks.

En hook på serversiden skal testes i modtagermiljøet for et rigtigt Git-push, så hooken blev øvet mod et rigtigt bare repository og et rigtigt push, før den fik tillid. Øvelsen opretter et bare repository, installerer hooken, kloner, committer, pusher og kontrollerer både vejen til accept og vejen til afvisning. Den fandt to fejl, og begge var usynlige i filen. Den anden er den lærerige: Git eksporterer en variabel med repositoryets mappe, når den kører en hook, testsuiten, som hooken startede, arvede den, og nitten test gik røde på korrekt kode, indtil miljøet blev ryddet ved grænsen. Siden nævner også sin grænse: tvungne opdateringer og sletning af grene er push, som øvelsen ikke konstruerer.

Alt dette deler én værtsmaskine med én kerne og 512 MB hukommelse med alt det andet, virksomheden stiller til rådighed offentligt, fra de syv TLS-domæner til onion-tjenesten, en Gemini- og en Gopher-server og blokering af indtrængningsforsøg. Automatisering af sikkerhedskopiering er implementeret separat; det er noget andet end aktive sikkerhedskopier. En forge med et webpanel ville have været endnu en fast kørende applikation på det budget, og

derfor blev platforme med webpanel fravalgt allerede på designstadiet; på den vært nåede den tilgængelige hukommelse sit lavpunkt på omkring 310 MB af 464 over et registreret døgns stikprøver.

Hvor det går galt

Anonyme besøgende får læseadgang som standard på nogle forges. Soft Serves dokumentation siger, at den fra start giver anonyme forbindelser skrivebeskyttet adgang, styret af to indstillinger, `anon-access` og `allow-keyless`. En privat forge sætter den første til `no-access` og den anden til false, og resultatet er værd at teste fra en maskine uden indlæst nøgle. Gits egen dæmon er mere kontant: Git-bogen bemærker, at et repository, den serverer, enten kan klones af alle eller slet ikke serveres.

En hook, der kalder Git i en anden mappe, rammer det forkerte repository. Under et push kører Git de modtagende hooks inde i det bare repository og eksporterer `GIT_DIR` og beslægtede variabler, så Git-kommandoer i hooken finder det. Alt, hvad hooken starter, arver dem, også en testkørsel i en udtjekning et andet sted. Manualen til githooks siger, at variablerne skal ryddes, før man arbejder i et fremmed repository. En minimal `pre-receive`-hook, der tester hvert pushet commit i en midlertidig mappe, gør det i en subshell:

```
while read -r _ newrev _; do
  git cat-file -e "$newrev^{commit}" 2>/dev/null || continue
  tree=$(mktemp -d)
  git archive "$newrev" | tar -x -C "$tree"
  (
    unset $(git rev-parse --local-env-vars)
    cd "$tree" && ./run-tests
  ) || exit 1
done
```

En gendannet forge kommer tilbage med sin kode og uden sine regler. Enhver klon kan genbefolke en tabt servers repositories, men Git-bogen advarer om, at hooks på serversiden og lignende indstillinger bliver tilbage, og issues, reviews, nøgler og adgangslister var aldrig i klonerne. Backup af en forge betyder backup af dens datamappe og dens konfiguration, derefter en gendannelse af begge et andet sted og en kontrol af, at et push, som hooken skal afvise, faktisk bliver afvist.

Hver lytter, en forge åbner, skal forsvares. Soft Serves eksempelkonfiguration har SSH på port 23231, HTTP på 23232, Git-dæmonen på 9418 og en statistikserver på 23233. Et team, der kun pusher over SSH, kan slå resten fra eller binde dem til loopback, så de åbne porte svarer til det, der faktisk bruges.

En flytning til en anden forge tager koden med og efterlader samtalen. Et spejl-push kopierer hver ref under `refs/`, grene og tags medregnet, og det er alt, det kopierer. Issues, review-tråde, labels og adgangslister kræver den nye forges migreringsværktøj eller en protokol som ForgeFeds, der kopierer commits, tags, tickets, merge requests og kommentarer og efterlader det gamle repository arkiveret med en henvisning til det nye.

Læs mere

- ForgeFed-specifikationen — føderationsprotokollen for forges med dens aktører, dens aktiviteter og den flytning, der arkiverer et gammelt repository.
- Soft Serve — projektets egen dokumentation for en SSH-først Git-server i én binærfil, herunder indstillingerne `anon-access` og `allow-keyless`.
- Forgejo-dokumentationen — vejledninger til administratorer og brugere af en selvhostet forge med alle funktioner, fra installation og backups til Actions og spejle.
- githooks — alle de hooks, Git kører, herunder `pre-receive` og det miljø, en push-hook arver.
- The Protocols — de fire måder, Git flytter data mellem maskiner på, og hvad hver af dem koster i opsætning og sikkerhed.
- ActivityPub — W3C-anbefalingen, som ForgeFed udvider.

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 forge?« 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/forge/