Engineer

Sikkerhetskopi

en kopi av data mot tap, og det å lage den

Hva det er

En sikkerhetskopi er det dataene hentes tilbake fra når originalen blir slettet, ødelagt eller forsvinner sammen med maskinen den var lagret på, og derfor må den ligge et sted disse hendelsene ikke når. NIST-ordlisten definerer den som en kopi av filer og programmer laget for å gjøre gjenoppretting enklere. På norsk heter selve kopien sikkerhetskopi og arbeidet med å lage den sikkerhetskopiering, mens mange i bransjen bare sier backup om begge deler: en nattlig jobb sikkerhetskopierer databasen og etterlater en sikkerhetskopi.

Beslektede løsninger blir forvekslet med sikkerhetskopier. En replika får hver endring i løpet av sekunder, noe som beskytter mot en død disk og kopierer en feilaktig `DELETE` like trofast. Et øyeblikksbilde fryser et volum i ett øyeblikk, og blir en sikkerhetskopi først når det lagres et sted der feilene som rammer originalen, ikke kan følge etter. En sikkerhetskopi er definert av historikk og atskillelse: den gir tilbake dataene slik de var på et tidligere tidspunkt, fra et sted som overlevde det som tok originalen.

Databaser gjør dette konkret. PostgreSQL-manualen beskriver tre grunnleggende forskjellige tilnærminger. En SQL-dump, laget med `pg_dump`, skriver ut setningene som bygger opp databasen igjen, som et konsistent bilde av øyeblikket dumpen startet. En sikkerhetskopi på filsystemnivå kopierer selve datakatalogen. Kontinuerlig arkivering kombinerer én slik basekopi (base backup) med write-ahead-loggen (WAL) serveren skriver etterpå, og ved å spille av den loggen kan databasen gjenopprettes til et hvilket som helst øyeblikk etter basekopien. Den evnen kalles gjenoppretting til et tidspunkt (point-in-time recovery).

Bruk begrepet i en beslutning

Bruk «Hva er en sikkerhetskopi?» i en konkret beslutning. Vi kan knytte begrepet til systemet dere bygger eller kjøper. Sikkerhetskopier som er testet i stedet for tatt for gitt, og en skriftlig plan for morgenen da en server ikke kommer opp igjen.

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. Automatiserte 100 kritiske sikkerhetskopier — 100 kritiske sikkerhetskopier automatisert på tvers av databasene

Automatiserte 100 kritiske sikkerhetskopier Ta kontakt →

Hvorfor det er viktig

To tall gjør et generelt ønske om å være trygg om til et design. NIST definerer gjenopprettingspunktmålet, RPO (recovery point objective), som tidspunktet dataene må gjenopprettes til etter et avbrudd: i praksis hvor mye nylig arbeid virksomheten har råd til å miste. Gjenopprettingstidsmålet, RTO (recovery time objective), er hvor lenge et system kan være under gjenoppretting før avbruddet skader det organisasjonen er til for.

De to trekker i ulike deler av designet. RPO bestemmes av hvor ofte en kopi tas: en nattlig dump kan miste et døgn med skrivinger, mens WAL-arkivering krymper tapet til de siste øyeblikkene med logg som fortsatt er på vei til arkivet. RTO bestemmes av hvor lang tid en gjenoppretting tar. PostgreSQL-manualen påpeker at serveren spiller av hvert WAL-segment siden siste basekopi, noe som kan ta tid når den kopien er gammel, så hvor ofte basekopier tas, er like mye en RTO-beslutning som et spørsmål om lagringsplass.

Begge tallene er påstander om en gjenoppretting, og derfor bedømmes en sikkerhetskopi ved å gjenopprette den. En jobb som avslutter med status null, viser at noen byte ble skrevet et sted. Om de bygger opp et fungerende system innenfor RTO, får du svar på ved å gjenopprette etter en fast plan, et annet sted enn i produksjon, og sammenligne det som kommer tilbake.

Atskillelse er den andre halvdelen. En kopi i samme konto, under samme påloggingsinformasjon, deler feilene den er ment å overleve: en stengt konto, en stjålet nøkkel, et skript kjørt mot feil mål. Det nyttige spørsmålet for hver kopi er hvilken enkelt hendelse som ville tatt den sammen med originalen.

Slik ser det ut i praksis

I den lille enden ble selskapets implementasjon av sikkerhetskopiering og gjenopprettingstest utformet for å gi gjenoppretting et grunnlag utover leverandørens øyeblikksbilde. En Ansible-rolle installerer restic-skript og systemd-enheter for databasedump der det er konfigurert, krypterte og dedupliserte øyeblikksbilder, sletting etter oppbevaringsregler og integritetskontroller. En egen månedlig arbeidsflyt gjenoppretter en kjent fil til et midlertidig sted og sammenligner bytene. Sikkerhetskopiering og gjenoppretting har hvert sitt suksessignal, og tidsurene krever et konfigurert lager. Passfrasen leses fra en beskyttet fil, ikke fra en miljøverdi som systemd tolker. Dette er implementerte kontroller og forutsetninger for aktivering: implementasjonsregistreringen datert 2026-10-27 sier at tidsurene var deaktivert uten et lager, og dokumenterer ingen fullført gjenoppretting under reelle driftsforhold. Det underbygger automatiseringsarbeidet, ikke en påstand om at gjenoppretting har lykkes eller at verten allerede kan gå tapt uten å sette data i fare.

I større skala ble hundre kritiske sikkerhetskopier av data automatisert med Barman og Google Cloud for et produkt som lever av GIS-dataene sine. Barman tok seg av PostgreSQL, Google Cloud holdt kopiene utenfor huset, og Bash og Python orkestrerte og validerte kjøringene. Integritetskontroller bekreftet at hver sikkerhetskopi var god, og ikke bare til stede.

På et tredje produkt gikk arbeidet ut på å definere gjenopprettingsevnen i stedet for å anta den. Oppbevaringstiden for sikkerhetskopier ligger i Bicep-malen ved siden av databasen, så gjenoppretting til et tidspunkt er en egenskap ved malen og ikke en innstilling noen en gang klikket seg fram til i en portal, og miljøet rundt er også kode. Revisjonen skrev ned hullene: georedundans er slått av, og RTO er foreslått og ikke målt, med øvelsen som skulle måle den, beskrevet og klar til å kjøres.

Hvor det går galt

Problemer med sikkerhetskopier holder seg gjerne skjult til det trengs en gjenoppretting, så hvert av dem er verdt å lete etter en rolig dag.

En filkopi av en kjørende database blir gjenopprettet som en ødelagt klynge. Å kopiere datakatalogen til PostgreSQL med `tar` mens serveren kjører, samler filer fra ulike øyeblikk, og manualen sier at serveren må stoppes for å få en

brukbar sikkerhetskopi på denne måten. Et øyeblikksbilde av filsystemet tatt i ett atomært øyeblikk virker hvis det tar med WAL, som serveren spiller av ved oppstart slik den ville gjort etter et krasj; med data og WAL på ulike filsystemer kan et samtidig øyeblikksbilde være umulig. `pg_dump`, eller en basekopi for kontinuerlig arkivering, gir en konsistent kopi mens serveren fortsetter å kjøre.

Et hull i WAL-arkivet avslutter gjenopprettingen til et tidspunkt ved hullet. Gjenoppretting krever en ubrutt rekke arkiverte WAL-filer helt tilbake til starten av basekopien. Når arkiveringskommandoen feiler gang på gang, fortsetter PostgreSQL å prøve på nytt, og WAL hoper seg opp i `pg_wal/`, så det første symptomet er ofte en disk som fylles. Manualen råder til å sette opp og teste arkiveringen før den første basekopien, og å overvåke den for forsinkelser.

Jobben stopper, og tausheten ser ut som suksess. En timerjobb som slutter å virke, skriver feilen sin til en logg ingen leser, og hver stille dag går for å være en god dag. Overvåkingen må behandle taushet som alarmen: en innsjekking som sendes bare etter en vellykket kjøring, til en tjeneste som varsler vakthavende når den uteblir, fanger opp en jobb som har stoppet, like godt som en som feiler høylytt.

Nøkkelen blir borte, eller kommer fram endret. Kryptering gjør tilgangen avhengig av nøklene. restic støtter flere passord for samme repo; advarselen om ugjenkallelig datatap gjelder når det ikke lenger finnes noen brukbar nøkkel til repoet. Nøkkelen må ligge et sted som overlever maskinen. I restic-oppsettet ovenfor leses passfrasen fra en fil og ikke fra miljøet, fordi tjenestestyreren tolker escape-sekvenser i miljøverdier, og en passfrase med en omvendt skråstrek ville ha nådd restic som andre byte enn dem sikkerhetskopirepoet ble opprettet med, uten noen advarsel.

Oppbevaringsregelen fjerner kopien som trengtes. Korrupsjon blir ofte oppdaget uker etter at den skjedde, og en regel som beholder sju daglige kopier, har da erstattet hver rene kopi med en skadet en. En regel i flere trinn beholder en lang hale til lav kostnad, og `--dry-run` i restic viser hva en regel ville fjernet før noe forsvinner:

```
restic forget --keep-daily 7 --keep-weekly 5 --keep-monthly 12 \
  --prune --dry-run
```

`forget` fjerner øyeblikksbilder, og `prune` fjerner dataene bare de viste til, så etter en prune er de gamle kopiene borte for godt.

Integritetssjekken går gjennom, og gjenopprettingen feiler likevel. Som standard verifiserer `restic check` strukturen i sikkerhetskopirepoet uten å lese hver pakkefil, siden det ville bety å laste ned alle sammen; `--read-data` leser alt, og `--read-data-subset=1/5` leser en femtedel, så fem kjøringer fra `1/5` til `5/5` dekker hele mengden. En full lesing beviser bare at de lagrede bytene er intakte. Om de bygger opp en fungerende database, med de riktige rollene og innstillingene, er noe bare en gjenoppretting viser.

Les mer

- NIST glossary: backup — definisjonen som brukes her, og publikasjonene den bygger på.
- NIST SP 800-34 Rev. 1 — veiledningen i beredskapsplanlegging som definerer RPO og RTO.
- PostgreSQL: Backup and Restore — de tre tilnærmingene og hva hver av dem koster.
- PostgreSQL: Continuous Archiving and Point-in-Time Recovery — basekopier, WAL-arkivering og gjenoppretting til et valgt øyeblikk.
- restic: Preparing a new repository — passord, passordfiler og advarselen om å miste dem.
- restic: Removing backup snapshots — regler for oppbevaring, `forget`, `prune` og `--dry-run`.



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 en sikkerhetskopi?» 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/backup/