Engineer

Overvåking

å følge med på et kjørende system så det varsler deg først

Hva det er

Overvåking er å lese kontinuerlig det et system i drift rapporterer om seg selv: noe holder rapportene opp mot grenser noen har trukket, og avgjør når en person må få vite det. På engelsk heter det monitoring, og i norske driftsmiljøer sier mange også monitorering. Googles bok om Site Reliability Engineering (SRE) definerer overvåking som å samle inn, behandle, aggregere og vise kvantitative sanntidsdata om et system: hvor mange forespørsler som kommer inn og av hvilke slag, hvor mange som feiler og hvordan, hvor lang tid de tar, og hvor lenge hver server har kjørt. Målet er en bestemt rekkefølge: systemet sier fra om at det har problemer, før en bruker gjør det.

Dataene kommer fra to retninger. Hvitboksovervåking (white-box) leser det et system selv viser om sitt indre, f.eks. et endepunkt for metrikker, lengden på en kø eller antall nye forsøk, og kan derfor se et problem bygge seg opp før noen merker det. Svartboksovervåking (black-box) står utenfor og oppfører seg som en bruker: den ber om en side, slår opp et navn eller fullfører en innlogging, og rapporterer om det virket og hvor lang tid det tok. Den ser bare det som allerede er i stykker, og nettopp det gjør den til en pålitelig grunn til å vekke noen.

Det meste et overvåkingssystem tar vare på, er tall målt over tid, fordi tall er billige å lagre og lette å holde opp mot en terskel. Det samme SRE-kapittelet peker ut fire av dem som stedet å begynne for alt som betjener brukere: latens, tiden en forespørsel tar; trafikk, hvor mye etterspørsel som kommer inn; feil, andelen forespørsler som mislykkes; og metning, hvor nær den travleste ressursen er å være full.

Rundt tallene sitter to typer utdata. Et dashbord er til å se på når noen allerede mistenker noe. Et driftsvarsel er til å avbryte når alle er opptatt med noe annet. I Prometheus-økosystemet er det serveren som evaluerer reglene, mens

Alertmanager grupperer, fjerner duplikater og ruter det som utløses.

To naboer er lette å forveksle med overvåking. Logging skriver ned enkelthendelser, én linje per ting som skjedde, og er det noen leser etter et varsel for å finne ut hvorfor. Observerbarhet (observability), slik OpenTelemetry-prosjektet beskriver det, er den bredere evnen til å stille et system nye spørsmål utenfra. Overvåking er den smalere og eldre praksisen: spørsmål bestemt på forhånd, stilt hele tiden, med et svar som kan tilkalle en person.

Bruk begrepet i en beslutning

Bruk «Hva er overvåking?» i en konkret beslutning. Vi kan knytte begrepet til systemet dere bygger eller kjøper. Overvåking og varsler som sier fra mens en feil fortsatt er liten, og tallene som viser hvor ofte den holder seg liten.

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. Bygde selskapets første observerbarhetsdashbord — selskapets første innsyn i egne systemer

Bygde selskapets første observerbarhetsdashbord Ta kontakt →

Hvorfor det er viktig

Alternativet er å høre om problemene fra dem som rammes, uansett når på døgnet de merker det. Overvåking fremskynder oppdagelsen og varsler noen som kan gjøre noe med problemet.

Den knappe ressursen er oppmerksomhet, ikke programvare. Et varsel som tilkaller noen, avbryter arbeid om dagen og søvn om natten, og SRE-boken sier rett ut hva som skjer når slike varsler kommer for ofte: folk skumleser dem, tviler på dem og overser dem til slutt, iblant også det ekte varselet som ligger gjemt i støyen. Prometheus-prosjektets råd får plass i én setning: hold varslingen enkel, varsle på symptomer, ha gode konsoller for å finne årsakene, og unngå å tilkalle noen når det ikke er noe å gjøre. Hver regel som kan vekke noen, er et løfte om at en person har noe som haster når den utløses.

Dermed blir den sentrale beslutningen et spørsmål om hva brukerne merker. En disk som er 85 % full, er en årsak; et kjøp i kassen som feiler, er et symptom. Å varsle på symptomer holder antallet regler nede, fordi det finnes langt færre måter brukere kan bli rammet på enn måter maskiner kan oppføre seg galt på. Årsakene hører likevel hjemme på dashbord og i saker, der de hjelper feilsøkingen uten å vekke noen. Kapasitet er det ærlige unntaket: en ressurs som snart er full, skader ingen i dag, men den kommer til å trenge en person snart.

Så kommer spørsmålet om å drifte selv eller leie. Å drifte selv betyr et lager for tidsserier, en ruter for varsler og noen som holder begge i god stand. En hostet tjeneste tar det arbeidet bort og fakturerer etter mengden data som sendes, så hver ny vert dukker opp på en faktura. SRE-boken nevner at selv hos Google har et team på 10–12 ingeniører typisk én eller to som har overvåking som hovedjobb. Uansett valg er det et system med sitt eget vedlikehold, og det skal dimensjoneres etter systemet det passer på.

Slik ser det ut i praksis

To av grunnleggerens tidligere roller og selskapets egen vertsmaskin viser den samme ideen i svært ulik skala.

Fra 2015 til 2022, i rollen som infrastrukturingeniør, var jobben støtte døgnet rundt, 24/7, for rundt 1 000 servere på en IPTV/OTT-strømmeplattform, pluss en god del kundeeide systemer, for kunder i Kina, USA og Tyskland. Med kunder i tre tidssoner fantes det ingen rolig time, og et problem kl. 3 om natten lokal tid var beste sendetid for noen andre. Mye av arbeidet gikk ut på å merke at noe var i ferd med å skli ut før det ble et driftsavbrudd, og problemene ble tatt hånd om før de nådde skjermen til en seer. På en slik tjeneste ser suksess ut som at ingenting skjer, og overvåking er det som lar et team vite det i stedet for å håpe det.

I rollen som databaseingeniør fra 2022 til 2024 reduserte automatisert overvåking nedetiden med 50 % på tvers av 100 databaser i PostgreSQL, MS SQL og BigQuery, som for det meste holdt GIS- og tidsseriedata.

Å overvåke selskapets egen vert for offentlig tilgjengelige tjenester betyr å arbeide i den andre enden av skalaen. Én maskin med 512 MB kjører disse offentlig tilgjengelige tjenestene, med omtrent 464 MB brukbart minne, og minnet er argumentet som avgjør hver designbeslutning på den. Rapporteringen hentes på forespørsel i stedet for å gå gjennom en agent, en pipeline og et dashbord: sju kommandoer som leser verten og gjengir Markdown, ikke endrer noe og bare kjører når noen ber om det. Avgjørelsen om onion-tjenesten viser metoden: da spørsmålet kom opp, ble det tatt stikkprøvemålinger på verten gjennom et døgn, og målingene avgjorde det. I disse dokumenterte stikkprøvemålingene var lavpunktet for tilgjengelig minne omtrent 310 MB av 464, swap-bruken lå på 2,6 %, og prosessoren var ledig 99,7 % av tiden. Med så lite minne er overvåking en måling tatt i det øyeblikket en beslutning trenger den.

Hvor det går galt

Varslene kommer så ofte at folk slutter å lese dem. Hver driftshendelse etterlater seg en regel, mange av reglene utløses av årsaker som går over av seg selv, og i løpet av noen måneder blir det ekte varselet skumlest sammen med resten. Tilkall noen bare for det brukerne ville merket, på ett sted i stakken, og send alt annet til en sakskø eller et dashbord. La deretter tilstanden vare en stund før regelen utløses, slik at et blaff som går over på et minutt, ikke vekker noen:

```
groups:
    - name: checkout
      rules:
          - alert: CheckoutErrorsHigh
            expr: job:checkout_errors:ratio5m > 0.02
            for: 10m
            labels:
                severity: page
```

Overvåkingen stopper, og stillheten blir lest som gode nyheter. En regel som aldri utløses, ser nøyaktig ut som et system som aldri går i stykker. Går Prometheus-serveren, Alertmanager eller e-postreléet ned, stopper alle varsler på én gang. Prometheus-prosjektets eget råd er å overvåke selve overvåkingen, og enda bedre å teste hele veien fra ende til ende, slik at én kontroll beviser at et varsel kan komme fra regelen og helt frem til innboksen. Planlagte jobber går i den samme fella: en sikkerhetskopiering som slutter å kjøre, gir ingen feilmelding, den slutter bare å etterlate spor. Tilkall noen når den siste vellykkede kjøringen er for gammel, med rom for minst to hele kjøringer. For en jobb som kjører hver fjerde time og tar en time, foreslår prosjektet 10 timer:

```
time() - backup_last_success_timestamp_seconds > 10 * 3600
```

Gjennomsnittet sier at alt er i orden mens noen brukere venter. Gjennomsnitt er lette å tegne grafer av, og de skjuler halen. Eksempelet i SRE-boken er en tjeneste med 100 ms i snitt ved 1 000 forespørsler i sekundet, der 1 % av forespørslene fort kan ta 5 sekunder. Når en side er avhengig av flere slike tjenester, kan 99-persentilen til én backend bli medianen til frontenden. Raske feil gjør det verre, fordi en rask feilmelding drar snittet ned. Registrer latens i bøtter, varsle på en høy persentil, og følg mislykkede forespørsler for seg, atskilt fra de vellykkede.

Alle interne tall er grønne mens brukerne ikke når tjenesten.
Hvitboksmetrikker kommer fra innsiden av systemet og deler derfor blindsonene dets: et utløpt sertifikat, en ødelagt DNS-post eller en endring i brannmuren kan la hver eneste av dem se frisk ut. En svartbokskontroll fra en maskin utenfor systemets nettverk fanger det som ellers er usynlig, og fortsetter å virke når den interne overvåkingen går ned sammen med alt annet.

Les mer

- Prometheus: Overview — hva et metrikkbasert overvåkingssystem samler inn, hvilke deler det består av, og når presisjon per hendelse krever noe annet.
- Alerting — Prometheus-prosjektets råd om hva som skal tilkalle noen, fra nettjenester og batchjobber til overvåking av selve overvåkingen.
- Alertmanager — hvordan varsler fjernes som duplikater, grupperes, dempes, holdes tilbake og rutes til en mottaker.
- Monitoring Distributed Systems — kapittel 6 i Googles SRE-bok: symptomer og årsaker, de fire gylne signalene og halen.
- Observability primer — OpenTelemetrys gjennomgang av signaler, SLI-er og SLO-er, og hvor observerbarhet strekker seg forbi overvåking.

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 overvåking?» 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/monitoring/