Engineer

Grafana

ett dashboardverktyg för mätvärden och loggar

Vad det är

Grafana gör om siffror som förvaras i andra system till dashboards. Det skickar en fråga till en datakälla som Prometheus, Loki, PostgreSQL eller MySQL, ritar upp det som kommer tillbaka som en tidsseriegraf, en tabell, en mätare eller en värmekarta, och utlöser ett larm när ett värde passerar en gräns som någon har satt. Den egna dokumentationen beskriver det som mjukvara för att ställa frågor till, visualisera, larma på och utforska mätvärden, loggar och spår var de än lagras, och de fyra sista orden bär hela designen.

Telemetrin på skärmen stannar där den samlades in. Varje panel hämtar sin del från datakällan när dashboarden laddas eller uppdateras, genom ett av fler än 150 insticksprogram för datakällor, och Grafana behåller bara sin egen konfiguration: dashboards, mappar, användare, behörigheter och larmregler. Den ligger i en liten databas, som standard SQLite inbäddad i binärfilen, medan MySQL eller PostgreSQL rekommenderas för produktion. En dashboard är ett JSON-dokument: paneler ordnade i rader, där varje panel är en fråga plus ett val av hur den ska ritas, och variabler längst upp som låter en och samma dashboard växla mellan värdar, miljöer eller tjänster från en rullgardinsmeny.

Grafana utvecklas av Grafana Labs, vid sidan av Loki för loggar, Tempo för spår och Mimir för långtidslagring av data från Prometheus, en uppsättning som ofta kallas Grafana-stacken. Den 20 april 2021 gick Grafana, Loki och Tempo över från licensen Apache 2.0 till AGPLv3. Grafana finns som utgåvan med öppen källkod, som Grafana Enterprise under en kommersiell licens och som Grafana Cloud, som Grafana Labs själva är värd för och driver.

Det vägs oftast mot Kibana, som spelar samma roll framför Elasticsearch, mot uttrycksbläddraren (expression browser) som är inbyggd i Prometheus, och mot driftade sviter för observerbarhet som säljer lagring och skärmar som en enda

produkt. Verktyg för affärsanalys som Metabase eller Power BI kan läsa samma databaser, men de är byggda för rapporter som en människa studerar, medan Grafana är byggt för skärmar som uppdateras med några sekunders mellanrum.

Använd begreppet i ett beslut

Använd ”Vad är Grafana?” i ett konkret beslut. Vi kan knyta begreppet till systemet ni bygger eller köper. Övervakning och larm som säger till medan ett fel fortfarande är litet, och siffrorna som visar hur ofta det förblir så.

Se våra tjänster Hör av er →

Granska antagandena bakom ordet

Använd förklaringen och det länkade arbetet för att formulera frågor till projektet. Vi kan undersöka antagandena tillsammans innan ett verktyg väljs. Byggde företagets första observerbarhetsdashboard — företagets första överblick över de egna systemen

Byggde företagets första observerbarhetsdashboard Hör av er →

Varför det spelar roll

Att installera Grafana är den enkla delen. Det körs på en liten maskin och kan riktas mot en databas som teamet redan har. De beslut som avgör vad det är värt kommer efteråt: vem varje skärm är till för, hur dashboards förvaltas när de har blivit dussintals, och var det körs.

Eftersom Grafana bara visar data flyttar ett införande ingenting. Ett team kan behålla Prometheus, PostgreSQL och sitt logglager där de är, och skulle senare kunna byta ut visningslagret utan att migrera en enda serie, så både inlåsningen och fakturan ligger hos datakällorna. Sviter som säljer lagring och visning tillsammans gör den motsatta avvägningen: en faktura och en inloggning, och data som är svårare att ta med sig någon annanstans.

Publiken formar en dashboard mer än verktyget gör. En ingenjör i beredskap vill ha en sida som med en blick besvarar ”är det trasigt, och var”, med länkar in i detaljerna. Ett rum fullt av människor med olika jobb behöver färre och större paneler, enkla rubriker och en tröskellinje som säger när det är dags att oroa sig. Samma Grafana tjänar båda genom olika dashboards, och en sida som byggs för båda på en gång brukar inte tjäna någon av dem.

Hur dashboards förvaltas är det andra beslutet. Grafana kan läsa in dem från JSON-filer på disk och bevaka mappen efter ändringar, vilket dokumentationen kallar provisionering. I versionshantering får en dashboard granskning, historik och en väg tillbaka, och en ny miljö får samma skärmar utan att någon behöver klicka sig igenom dem. Priset är att redigering i webbläsaren slutar vara sättet som ändringar görs på, och det måste hela teamet acceptera.

Licens och drift kommer sist. Enligt AGPLv3 måste den som ändrar Grafana och låter andra använda den ändrade versionen över ett nätverk erbjuda dem dess källkod. Ett team som kör ett oförändrat Grafana för sin egen personal står på en helt annan plats än en leverantör som bygger in en patchad version i en produkt, och leverantören bör läsa licensen tillsammans med en jurist. Att driva det själv handlar om en enda binärfil tills en andra instans behövs. Från den punkten delar instanserna en databas i MySQL eller PostgreSQL, och Grafana Cloud är sättet att i stället lämna över det arbetet till Grafana Labs.

Så ser det ut i praktiken

Båda gångerna Grafana förekommer i företagets arbete är det den del som folk tittade på, med Prometheus bakom.

I en roll som infrastrukturingenjör 2022–2024 var utgångsläget ett team som fick höra om problem efter användarna, eftersom det saknades en samlad bild av hur systemen mådde. En övervakningsdashboard i Grafana och Prometheus gav teamet den bilden. De viktigaste tjänsterna instrumenterades, mätvärdena valdes för att de betydde något och inte för att de såg livliga ut, och larmtrösklar sattes för dem och visades där teamet skulle se dem medan det fortfarande fanns tid att agera. Problem började fångas upp innan de eskalerade, tillförlitligheten förbättrades, och teamet gick från att släcka bränder till att ligga steget före.

Det andra fallet byggdes för en blandad publik. En dashboard på en stor tv-skärm på kontoret var avsedd för både tekniska och icke-tekniska kollegor. Den andra versionen, byggd på Grafana och Prometheus, visade mätvärden som hade valts ut tillsammans med DevOps- och utvecklingsteamen och som matades av datapipelines som läste in prestandasiffror från flera system, med larm när ett tröskelvärde överskreds. Teamen upptäckte och löste incidenter 40 % snabbare, och med siffrorna framför hela kontoret blev systemens hälsa något som alla ägde tillsammans.

En sådan väggskärm läses från andra sidan rummet av människor som aldrig kommer att redigera en fråga. Den belönar några få stora paneler, enheter i rubrikerna och färger som bara ändras när någon borde agera, och Grafana lämnar vartenda ett av de valen till den som bygger dashboarden.

Var det går fel

En dashboard växer tills ingen läser den. Varje incident lägger till en panel och ingen tar bort någon panel, så ett år senare rymmer sidan fyrtio små grafer och den som spelar roll ligger nedanför skärmkanten. Ge varje dashboard en publik och en fråga, lägg svaret på ”är det friskt” i den översta raden och flytta detaljerna till dashboards som länkas därifrån. En panel som ingen i teamet kan förklara är en panel att radera.

En rättning som görs i webbläsaren försvinner. Dashboarden provisionerades från en fil. Med `allowUiUpdates` på sitt standardvärde false vägrar Grafana att spara och visar dialogrutan ”Cannot save provisioned dashboard”. Är inställningen satt till true sparar Grafana ändringen i sin databas och skriver sedan över den med filen nästa gång filen ändras. Redigera filen, eller exportera JSON-koden från dialogrutan för att spara och committa den, och skriv ut inställningen uttryckligen i providern:

```
apiVersion: 1
providers:
    - name: operations
      type: file
      allowUiUpdates: false
      options:
          path: /var/lib/grafana/dashboards
```

Väggskärmen saktar ned systemet den bevakar. Varje öppen webbläsare, tv:n inräknad, kör om varje panels fråga vid varje uppdatering, och fem sekunder är det kortaste uppdateringsintervall som Grafana tillåter som standard. Tjugo paneler som uppdateras var femte sekund blir fyra frågor i sekunden från en enda skärm. Uppdatera i den takt som data ändras, vilket för Prometheus betyder insamlingsintervallet eller långsammare, höj golvet för alla och förberäkna tunga uttryck med inspelningsregler (recording rules) i datakällan:

```
[dashboards]
min_refresh_interval = 30s
```

Samma incident larmar samma personer två gånger. Regeln finns i Prometheus, dirigerad via Alertmanager, och en gång till i Grafana Alerting. Eller så utvärderar två Grafana-instanser varje regel var för sig, eftersom

larmhanteringen lämnades utanför när den andra instansen lades till. Ge varje regel ett enda hem, och följ Grafanas steg för hög tillgänglighet i larmhanteringen innan en andra instans tas i drift.

Dashboards och användare skiftar beroende på vilken server som svarar. Två Grafana-instanser bakom en lastbalanserare har var sin SQLite-fil, så en dashboard som sparas på den ena saknas på den andra och en användare finns bara på en av dem. Grafanas egen guide gör en delad databas i MySQL eller PostgreSQL till ett krav för hög tillgänglighet.

Läs mer

- About Grafana — projektets översikt över vad Grafana gör, dess utgåvor och projekten med öppen källkod runt det.
- Data sources — de system som Grafana kan ställa frågor till, och hur vart och ett läggs till och konfigureras.
- Dashboards — paneler, rader, variabler och den JSON-modell som en dashboard lagras som.
- Provision Grafana — datakällor och dashboards från filer, och vad som händer med ändringar som görs i webbläsaren.
- Set up Grafana for high availability — den delade databas som en andra instans behöver, och de extra stegen för larmhantering.
- Licensing — Grafana Labs sida om bytet från Apache 2.0 till AGPLv3 år 2021.

Ordlista

Välj utifrån era faktiska förutsättningar

Ta med driftsättningsvägen, driftsriskerna och dem som har jour. Vi kan avgränsa en förändring med kontroller, en väg tillbaka och tydligt ansvar.

Se vad det kostar Hör av er →

Gör begreppen praktiska

Fråga hur ”Vad är Grafana?” gäller i er organisation. Ett konkret exempel från ert nuvarande system räcker för att börja samtalet.

Se våra tjänster

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

Ta med dig den här sidan

Visitkort

Omsorgsfullt skapad med kärlek av D.

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

Upphovsrätt © 2025 – nu · Engineer ApS · Alla rättigheter förbehållna.

Utskrivet från https://engineer.company/sv/glossary/grafana/