KHOA CÔNG NGHỆ THÔNG TIN
TRƯỜNG ĐH KHOA HỌC TỰ NHIÊN - TPHCM

PROMETHEUS & GRAFANA

NGUYỄN TRỌNG NGHĨA
NGUYỄN HỒNG QUÂN

CHALLENGES IN TRADITIONAL MONITORING SYSTEMS:

📈 SCALABILITY: HARD TO MONITOR LARGE, GROWING SYSTEMS.
🔧 MAINTENANCE: ADDING FEATURES BECOMES COMPLEX.
🔌 INTEGRATION: DIFFICULT TO CONNECT NEW SERVICES.

PROBLEM STATEMENT

- You have 100 application services running on cloud, Kubernetes, or VMs.
- Some services are down due to high traffic or causing latency.
- How will you identify the problematic server?

HERE’S WHERE PROMETHUS SHINE!

🚀 SCALABLE: HANDLES LARGE SYSTEMS EASILY.
⚙️ SIMPLE SETUP: EASY TO ADD NEW COMPONENTS.
🔄 FLEXIBLE: WORKS WITH VARIOUS TOOLS AND EXPORTERS.

WHAT IS PROMETHEUS?

Definition and Purpose

Time-Series Database

A specialized database optimized for storing and querying time-stamped data efficiently.

Monitoring Tool

A software application designed to observe systems and alert users of issues.

Open source

🚀 LICENSE: APACHE LICENSE 2.0
📅 INITIAL RELEASE: 24 NOVEMBER 2012
💻 WRITTEN IN: GO
🍴 FORK: 9.7K
⭐ STAR: 59.8K

Widely used in many applications to monitor the health & performance of servers, apps, cloud-native environments and kubernetes

HISTORY

2012

Originated at SoundCloud for monitoring

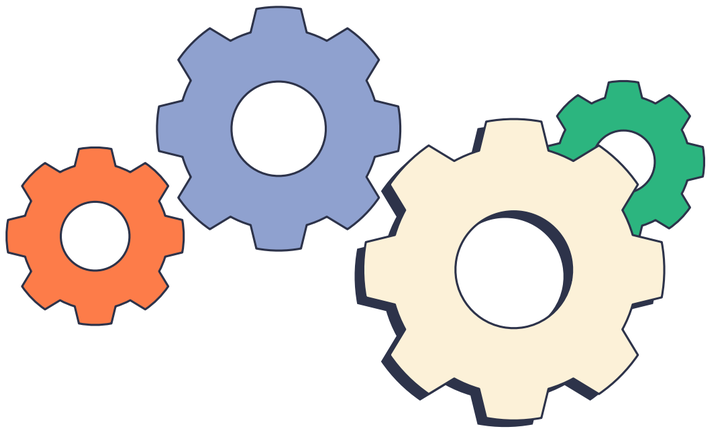
2016

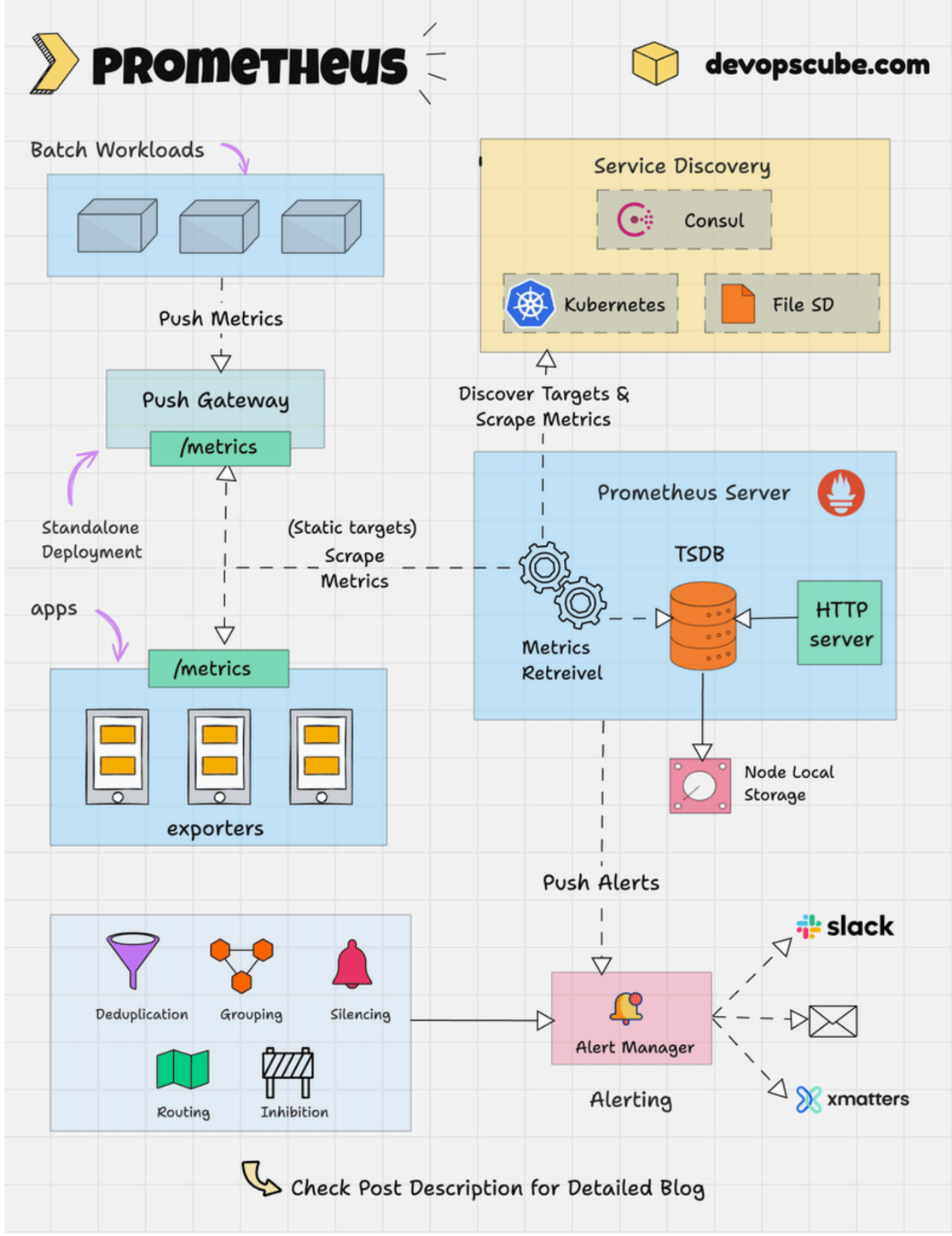
Adopted by CNCF, expanding its reach

2021

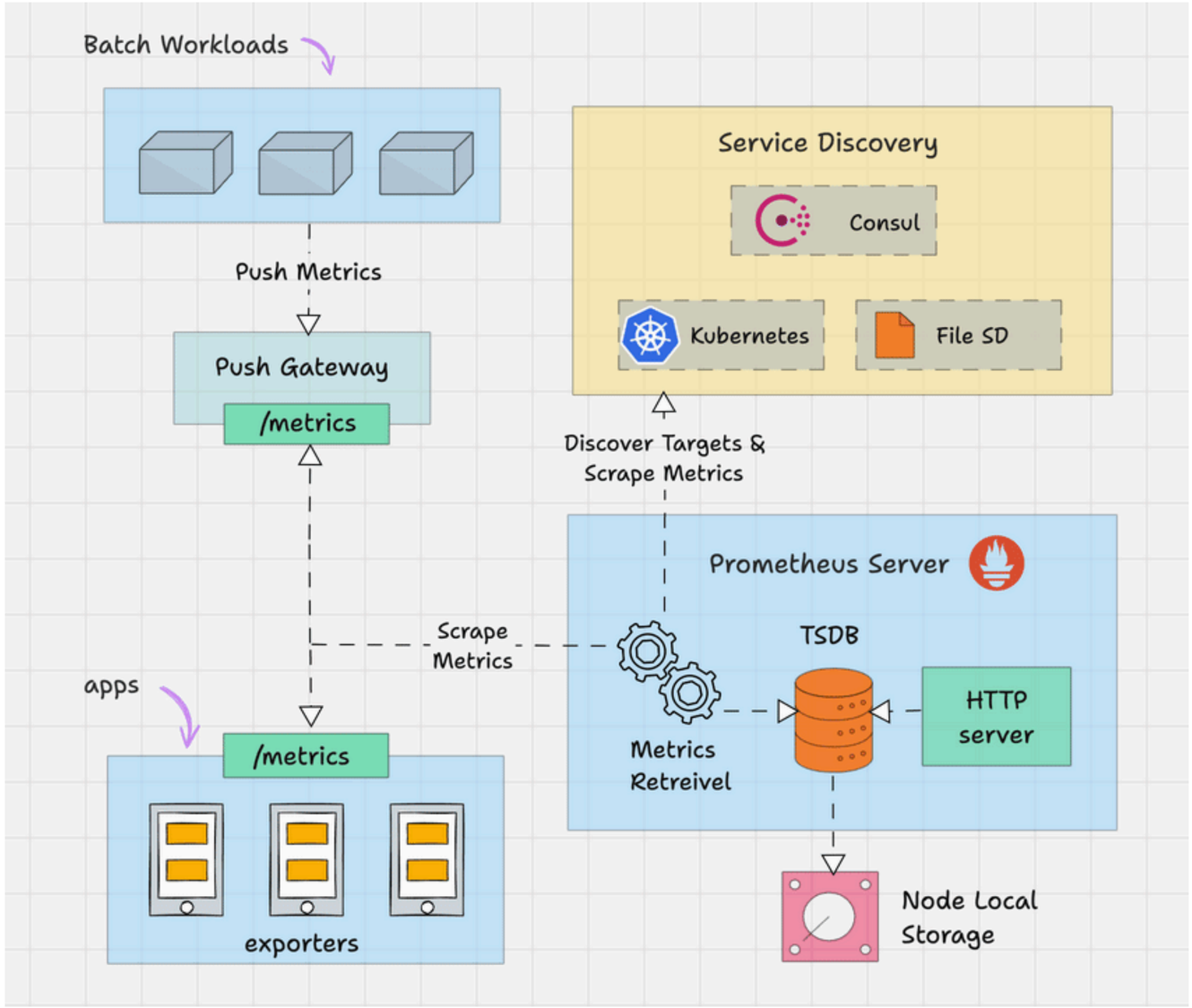
Strong growth in open-source community

ARCHITECTURE & COMPONENTS




PROMETHEUS
devopscube.com
Batch Workloads
Push Metrics
Push Gateway
/metrics
Standalone Deployment
apps
/metrics
exporters
Service Discovery
Consul
Kubernetes
File SD
Discover Targets & Scrape Metrics
Prometheus Server
(Static targets) Scrape Metrics
TSDB
HTTP server
Metrics Retreival
Node Local Storage
Push Alerts
Deduplication
Grouping
Silencing
Routing
Inhibition
Alert Manager
Alerting
slack
xmatters
Check Post Description for Detailed Blog

Prometheus server



THE PROMETHEUS SERVER IS THE BRAIN OF THE SYSTEM

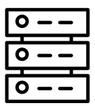
THE MAIN JOB OF THE SERVER IS TO COLLECT THE METRICS FROM VARIOUS TARGETS USING PULL MODE

SO WHAT ARE TARGETS & METRICS?

Targets & Metrics

Targets

- Target is what to monitor.
- It could be:
  - a server
  - a service
  - a kubernetes pod
  - an application endpoint
  - a database



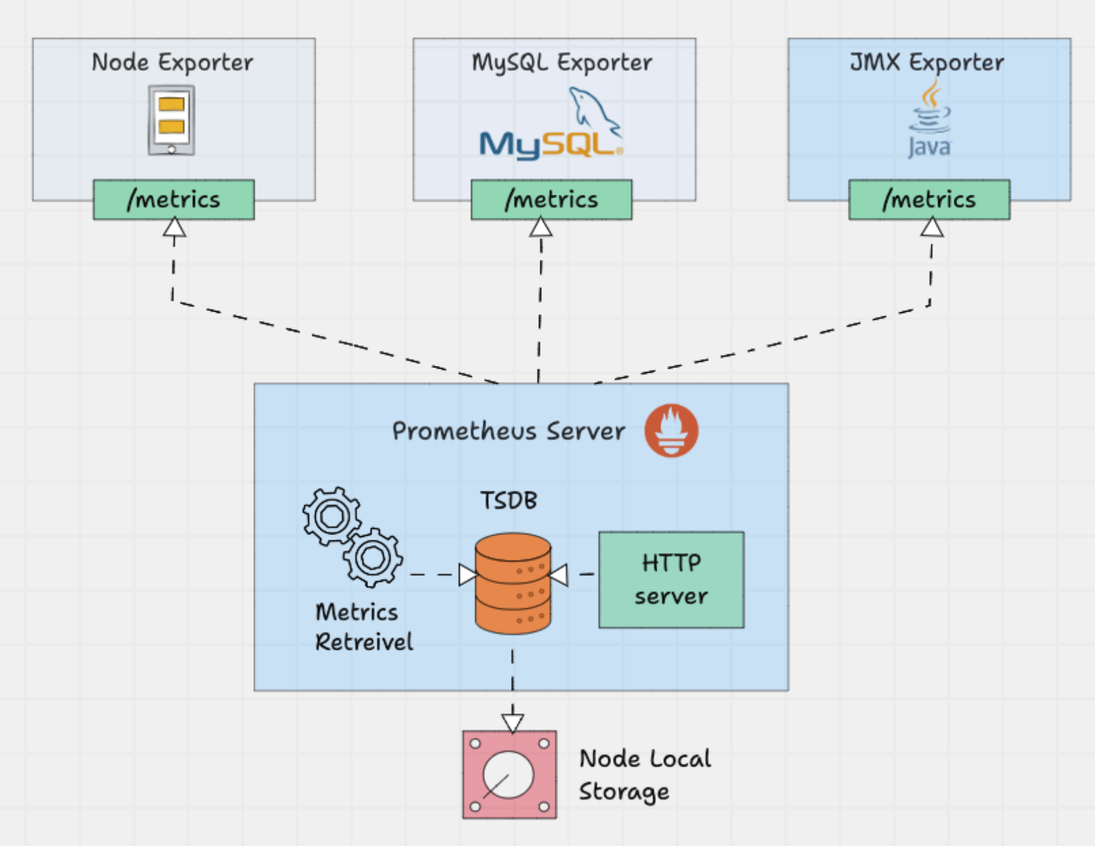



Metrics

- Metrics are measurements and records in numbers
- In timestamp format
- Example:
  - http_requests_total{method="GET", status="200"} 1027

BUT HOW CAN WE MAKE SURE
THAT THE TARGET DATA IS
COMPATIBLE WITH PROMETHEUS
EXPECTATION?

EXPORTER



- By default, Prometheus pulls metrics from /metrics endpoint
- Exporters help convert metrics to the format expected by Prometheus
- Some types of exporter:
  - Node exporter
  - Application exporter
  - Custom exporter
- Data is saved in Time-series database

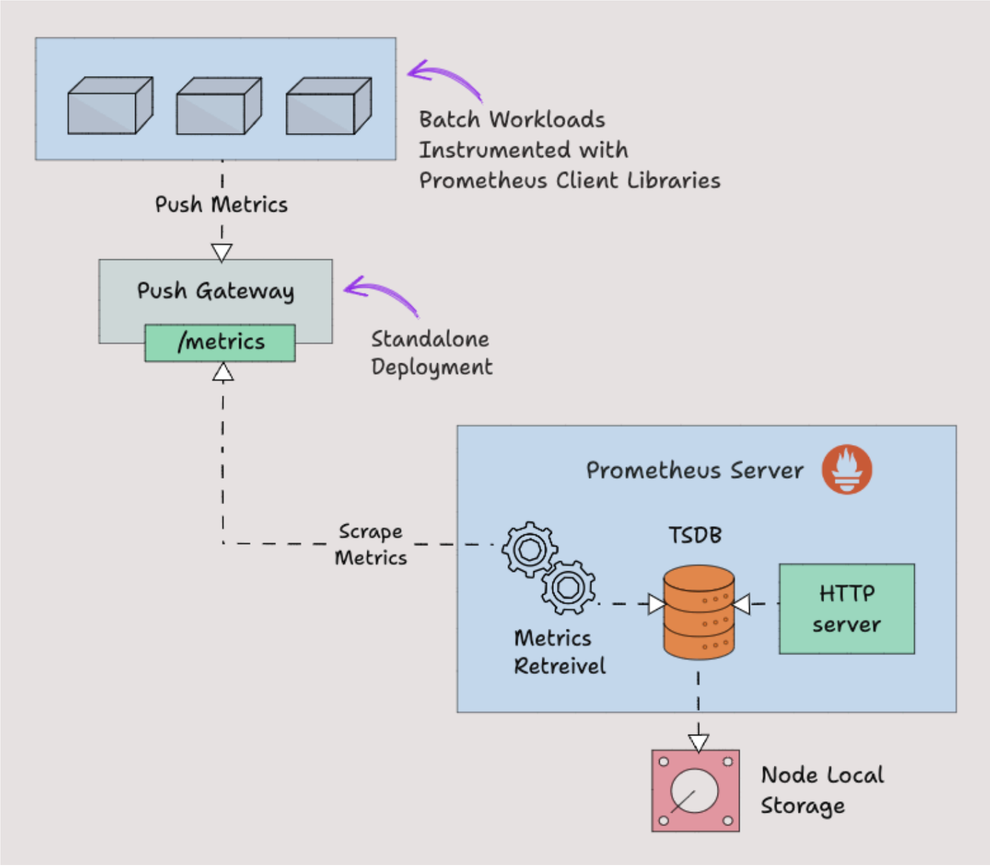
So if Prometheus can automatically pull metrics

WHY DO WE STILL NEED PUSH GATEWAY?

Cron Job

- Imagine we have a batch job running on a Kubernetes cronjob that runs daily for 1 min
- Prometheus will not be able to scrape the service level metrics
- How to solve this problem?

The solution is ... Push gateway



- The batch jobs can push the metrics to the pushgateway endpoint
- Pushgateway exposes those metrics
- Prometheus scrapes those metrics from the Pushgateway

HOW CAN PROMETHEUS KNOW ABOUT THE EXISTENCE OF SERVICES?

Static configs

- Define a fixed list of targets in the config file.
- Suitable for environments with few changes.
- Requires manual updates when adding/removing services.

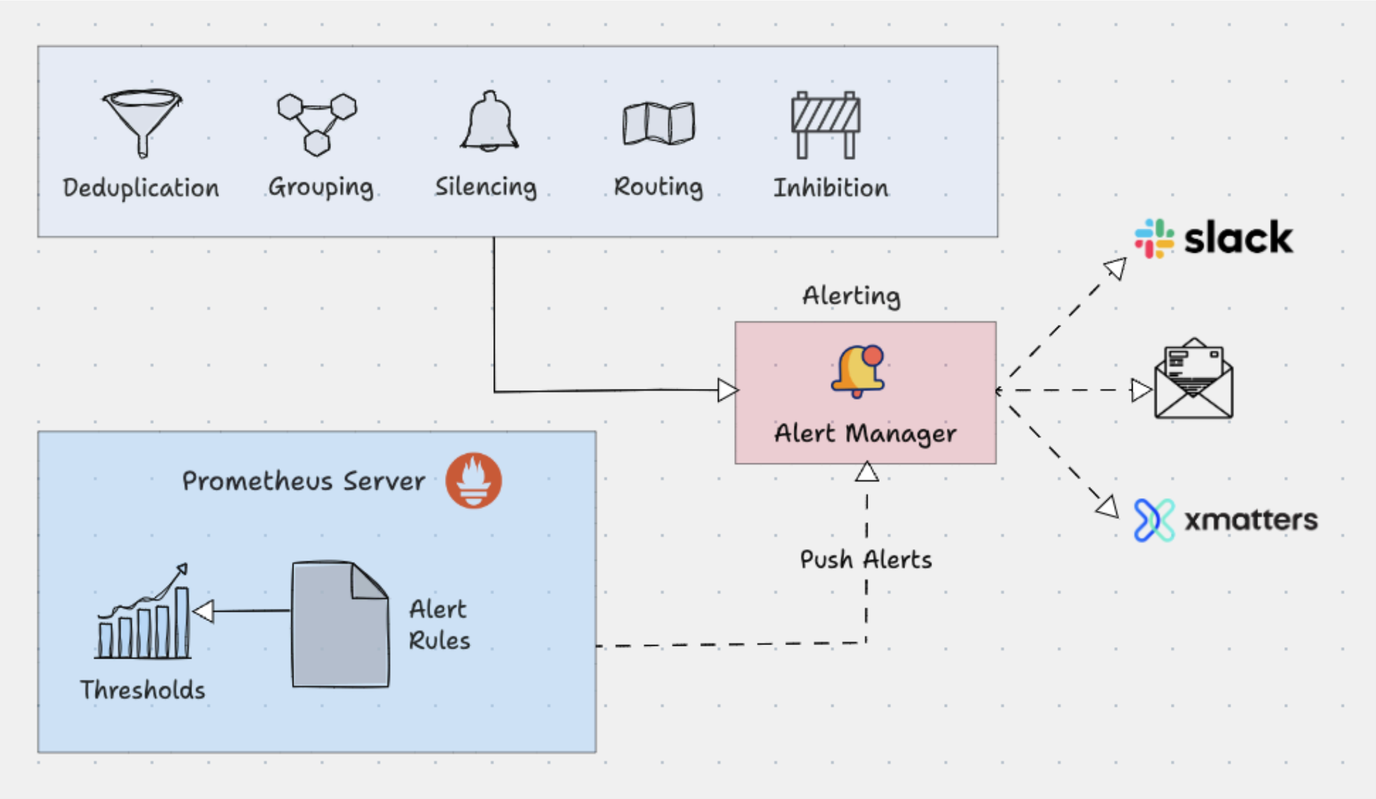
Sevice Discovery

- Automatically detects targets from the environment (Kubernetes, Consul, EC2...).
- Ideal for dynamic systems with frequently changing services.
- Reduces configuration effort and avoids missing targets.

Our system seems good

But what happens if we don’t have enough resources to manually monitor it 24/7?

Alert Manager



- Prometheus server pushes alert to alert manager
- The alert manager in turns forward to the respective notification systems/receivers
- Also, alert manager takes care of the following.
  - Alert Deduplicating
  - Grouping
  - Silencing
  - Routing
  - Inhibition

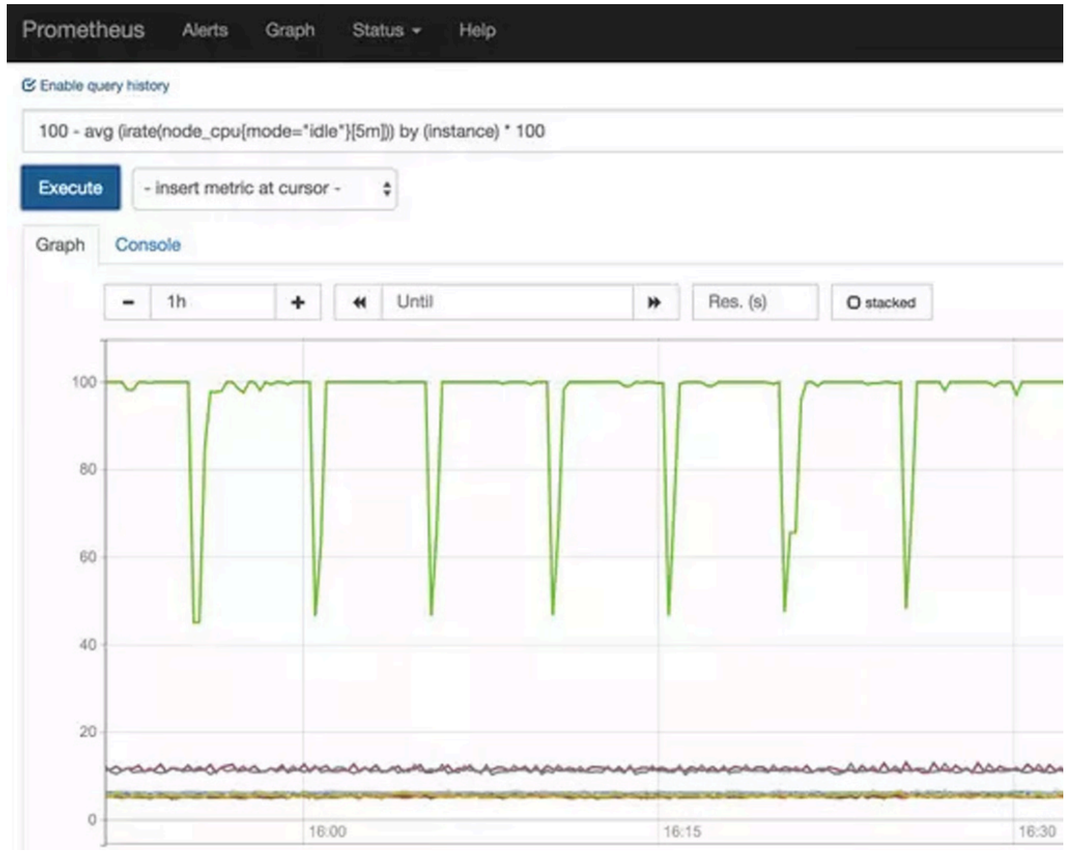
Ok, that’s cool

But how may we see the data we collected?

PromQL

A flexible query language that can be used to query time series metrics from the prometheus

Browser extension

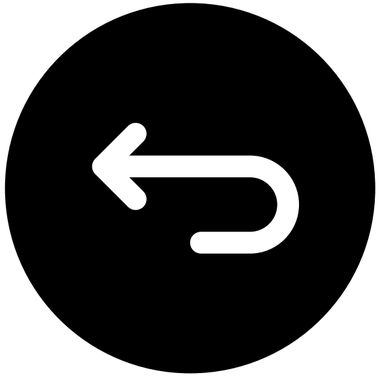


cURL

```
curl -g
'http://localhost:9090/api/v1/query?
query=rate(http_requests_total{status="2
00"}[5m])'
```

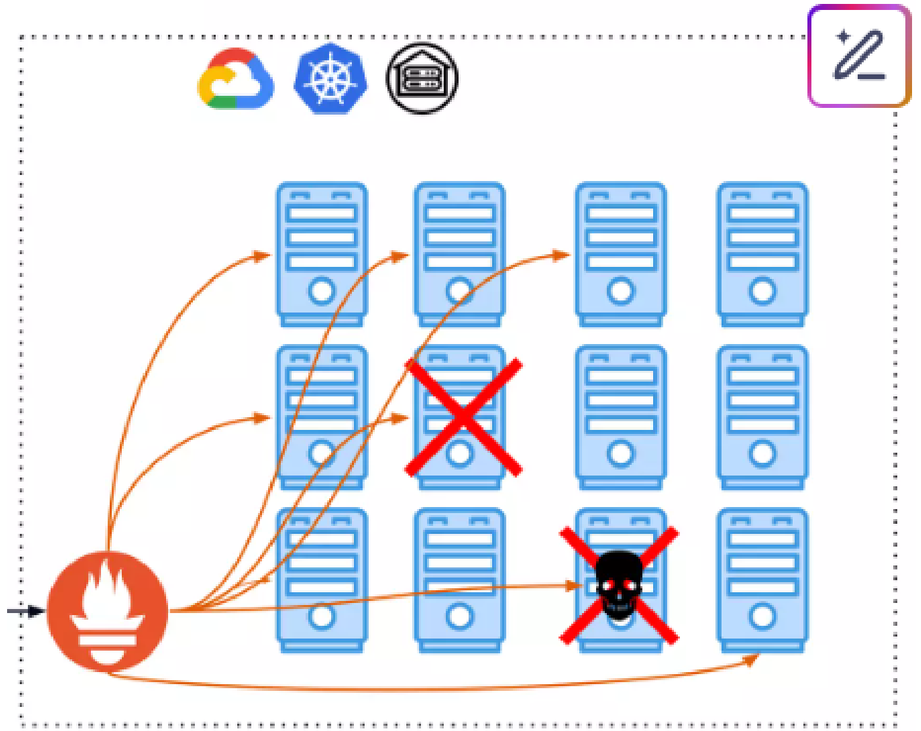

```
{
  "status": "success",
  "data": {
    "resultType": "vector",
    "result": [
      {
        "metric": {
          "status": "200"
        },
        "value": [1621294897.338, "13.5"]
      }
    ]
  }
}
```

So let’s get back to the problem raised at the beginning



Prometheus in game

- Set up Prometheus & Exporter to generate server and application metrics on port 8200.
- Configure Prometheus to scrape metrics from all exposed endpoints and store them with timestamps.
- Monitor metrics on each server and use PromQL queries to identify outages or issues.



- In fact, we can do almost everything with Prometheus
- However, Prometheus UI only provides a basic interface for querying and viewing data

What if we need a more powerful visualizing tool?

That's where Grafana comes in!

GRAFANA

GRAFANA IS AN OPEN-SOURCE PLATFORM DEVELOPED BY GRAFANA LABS FOR VISUALIZATION AND MONITORING.

KEY FEATURES

- QUERY DATA FROM MULTIPLE SOURCES
- VISUALIZE DATA THROUGH CUSTOMIZABLE DASHBOARDS
- SET ALERTS BASED ON DEFINED CONDITIONS
- GAIN INSIGHTS INTO SYSTEM PERFORMANCE AND METRICS

DATASOURCES

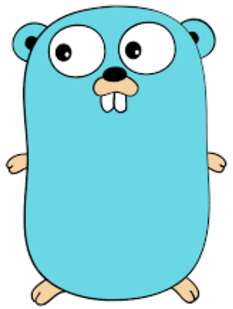
HOW GRAFANA WORKS

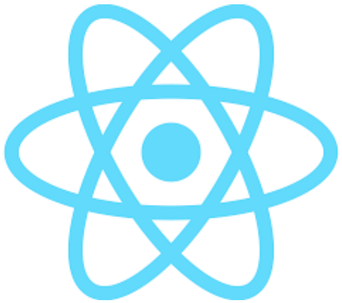
- BACKEND:

  WRITTEN IN GO, IT HANDLES DATA QUERIES FROM VARIOUS DATA SOURCES.

- FRONTEND:

  BUILT WITH TYPESCRIPT AND REACT, IT PROVIDES A USER-FRIENDLY INTERFACE FOR CREATING DASHBOARDS AND VISUALIZING DATA.





WORKFLOW

USE CASE

- INFRASTRUCTURE MONITORING
- SECURITY & COMPLIANCE
- APPLICATION PERFORMANCE MONITORING
- IOT DATA VISUALIZATION

DEMO

THANK YOU