Practical monitoring with Prometheus and Grafana

Jess Portnoy jess.portnoy@kaltura.com, Kaltura, Inc

Abstract

Prometheus is an open source monitoring and alerting toolkit.

In this session we'll review the Prometheus architecture and various tools and walk you through erecting an end-to-end monitoring and alerting infrastructure with the Prometheus stack.

In addition to Prometheus, we'll use:

- Consul for automatic service discovery
- Grafana for data visualisation

Session Overview

The session will cover the following topics:

- The Prometheus daemon and metric exporters
- Common metric exporters (MySQL, memcached, Apache, and Nginx)
- Writing custom exporters to instrument your web app's metrics
- Leveraging Consul for auto detection and configuration of services
- Deploying and configuring AlertManager
- Deploying Grafana and generating different graphs and reports

Prometheus - main features

- A multi-dimensional data model with time series data identified by metric name and key/value pairs
- A flexible query language to leverage this dimensionality
- No reliance on distributed storage; single server nodes are autonomous
- Time series collection happens via a pull model over HTTP
- Targets are discovered via service discovery or static configuration
- Pushing time series is supported via an intermediary gateway
- Multiple modes of graphing and dashboarding support

Prometheus - main components

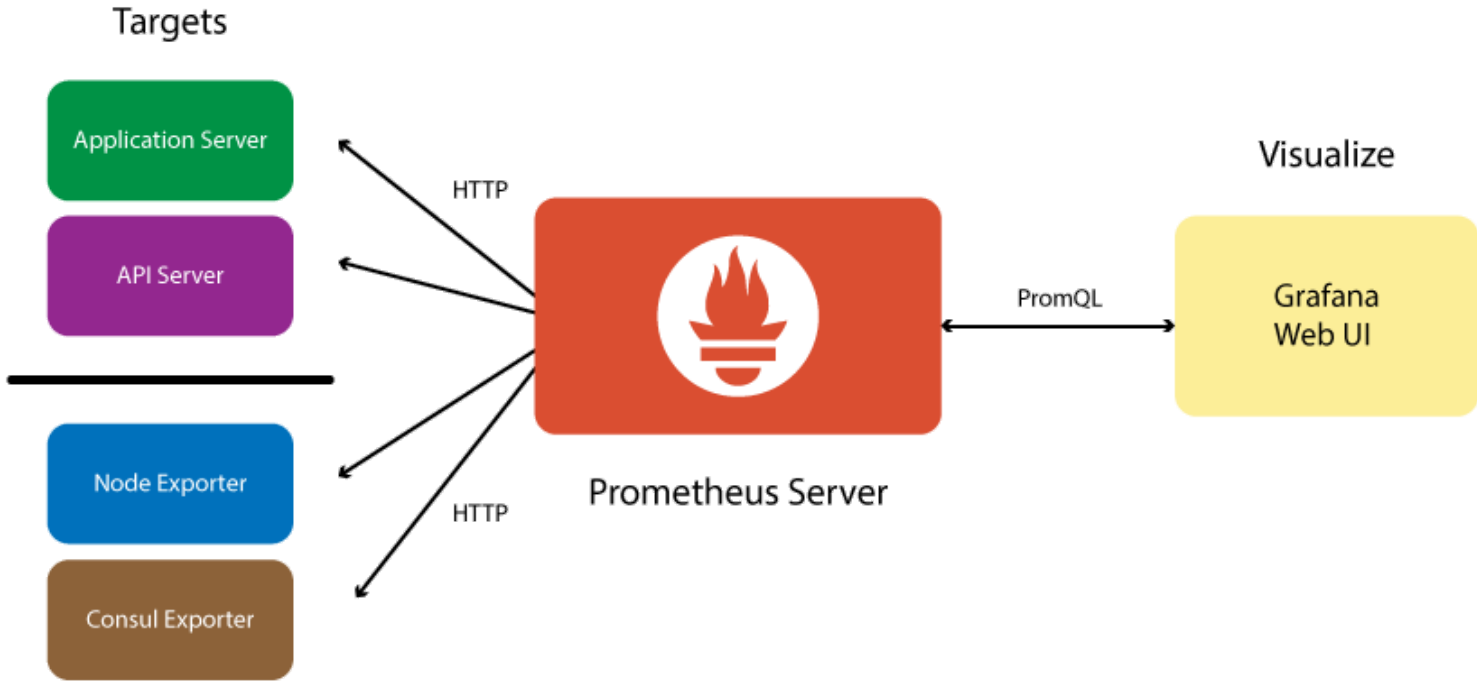
- The Prometheus server which scrapes and stores time series data
- Special-purpose exporters for services (HAProxy, MySQL, memcached, Apache, etc)
- Client libraries for instrumenting application code
- The Alertmanager
- A push gateway for supporting short-lived jobs (optional)

Prometheus - data model

Prometheus fundamentally stores all data as time series: streams of timestamped values belonging to the same metric and the same set of labeled dimensions.

Besides stored time series, Prometheus may generate temporary derived time series as the result of queries.

Architecture overview



Prometheus - monitoring

In Prometheus terms, the main monitoring service is referred to as the Prometheus Server and the services Prometheus monitors are called targets.
A target can be a host, a network equipment or a specific service.

Typically, the Prometheus server/daemon collects metrics from targets by making HTTP[s] requests to the Prometheus exporters.
In Prometheus terms, that process is called scarping.

Prometheus - exporters

An Exporter is a piece of software that fetches metrics from a given system and exports them in a format that the Prometheus server can understand.

There are a number of libraries that can be used to write custom exporters but there are also many existing FOSS exporters, written and maintained by the Prometheus team, third party vendors and community members.

If you need to monitor a popular FOSS system, chances are you'll find an exporter has already been written to get the job done.

Prometheus - monitoring your app

In order to monitor your application, you'll need to write code that retrieves the desired metrics and exports it in a format Prometheus can interpret.
Prometheus offers several official client libraries that can make the task easier:

- Go
- Java or Scala
- Python
- Ruby

Additional third party clients are also available. For a full list, see: https://prometheus.io/docs/instrumenting/clientlibs/

Consul - service auto discovery

Service discovery uses a registry to keep a real-time list of services, their location, and their health.
The registry can then be used to discover the location of upstream services and probe/connect to them directly.

This allows you to scale up/down and gracefully handle failure points.

In this demo, we'll see how Consul can be used in correlation with Prometheus so that targets are automatically discovered and scraped without having to modify Prometheus' configuration.

Prometheus - AlertManager

The Promtheus Alertmanager handles alerts sent by client applications such as the Prometheus server.

It takes care of de-duplicating, grouping, and routing them to the correct receiver integration such as email, PagerDuty, or OpsGenie.

AlertManager is also capable of silencing and inhibition of alerts. In this context, inhibition means suppressing notifications for certain alerts if certain other alerts are already firing.

Grafana - reporting and data visualisation

Grafana is an open source platform for analytics and monitoring.

It allows you to query, visualise, alert on and understand your metrics.
Grafana can connect to many different data sources including, though not limited to: MySQL, ElasticSearch and of course, Prometheus:)

In this session, we'll learn how to create, explore, and share dashboards.


Thank you && Questions

Appendix - Useful Resources

Prometheus

AlertManager

Consul

Grafana

Prometheus exporters

Prometheus client libs