An Introduction to Prometheus

Evgeny Shmarnev,
Prague Containers Meetup,
19.06.2018

$ whoami

- Software Engineer at GoodData
- Specialising in Containers, Immutable Infra, Cloud technologies, etc
- Open-Source contributor (Docker, Kubernetes …)
- Organiser of Docker Meetups in Prague, Co-organiser of Prague Containers Meetups

Inspired by Brian Brazil's talks and his book:
"Prometheus Up and Running"

Agenda

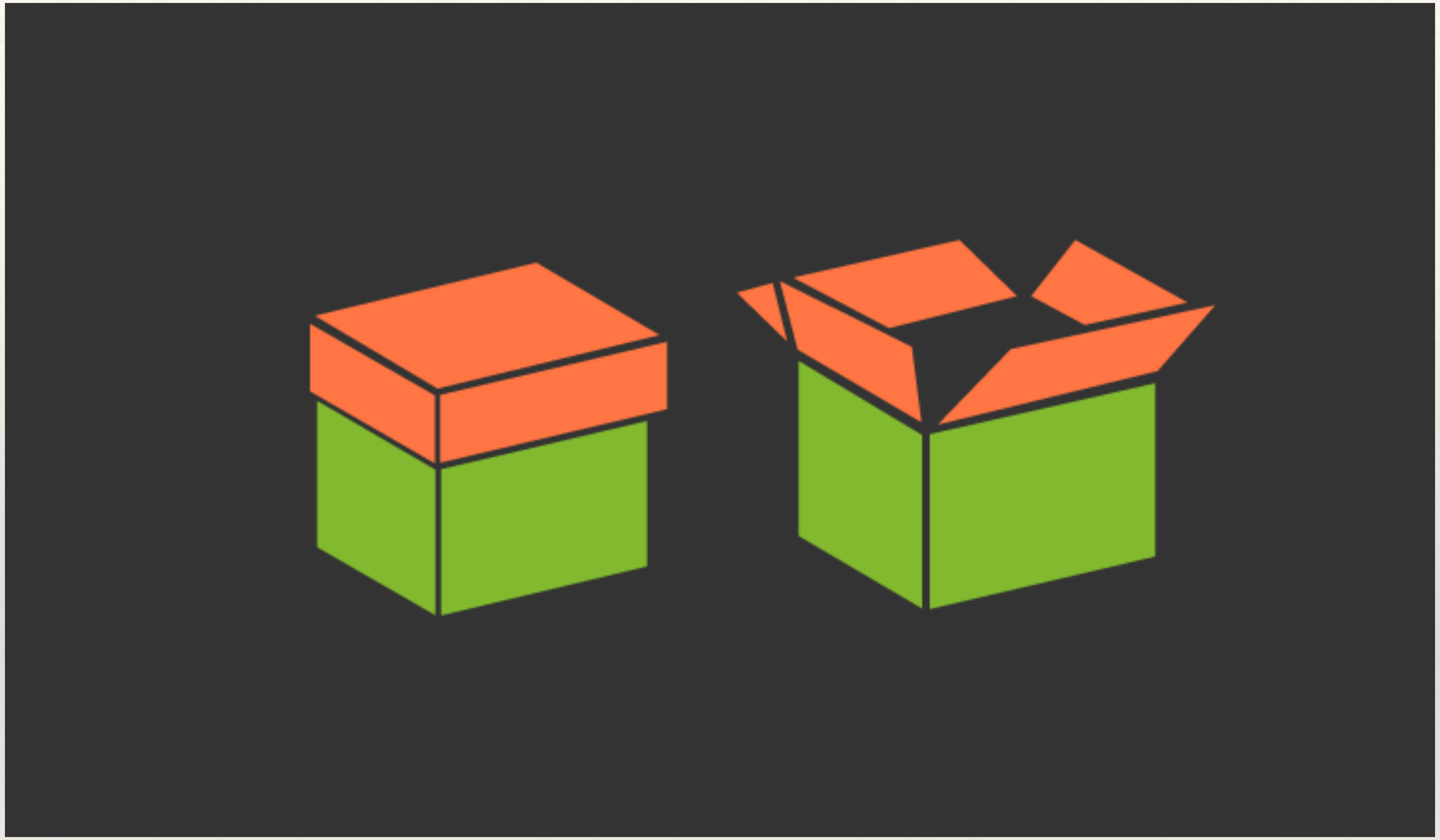
- What is “monitoring”?
- Monitoring with Prometheus
- Alerting with Prometheus

What is Monitoring?

“A process within a (distributed) system for collecting and storing state data ”

- *wikipedia.org*

“Blackbox” vs “Whitebox”



Blackbox monitoring

Blackbox Monitoring

- Monitoring from the outside
- We know nothing about what's going inside of the app
- Typical examples: ping the node
- Typical Monitoring systems: Nagios, Icinga

Pros and Cons

- Useful to know if your Node is broken
- and for more obvious cases

- Not so good if you actually know your app
- Don’t try to monitor every single feature with it

Whitebox monitoring

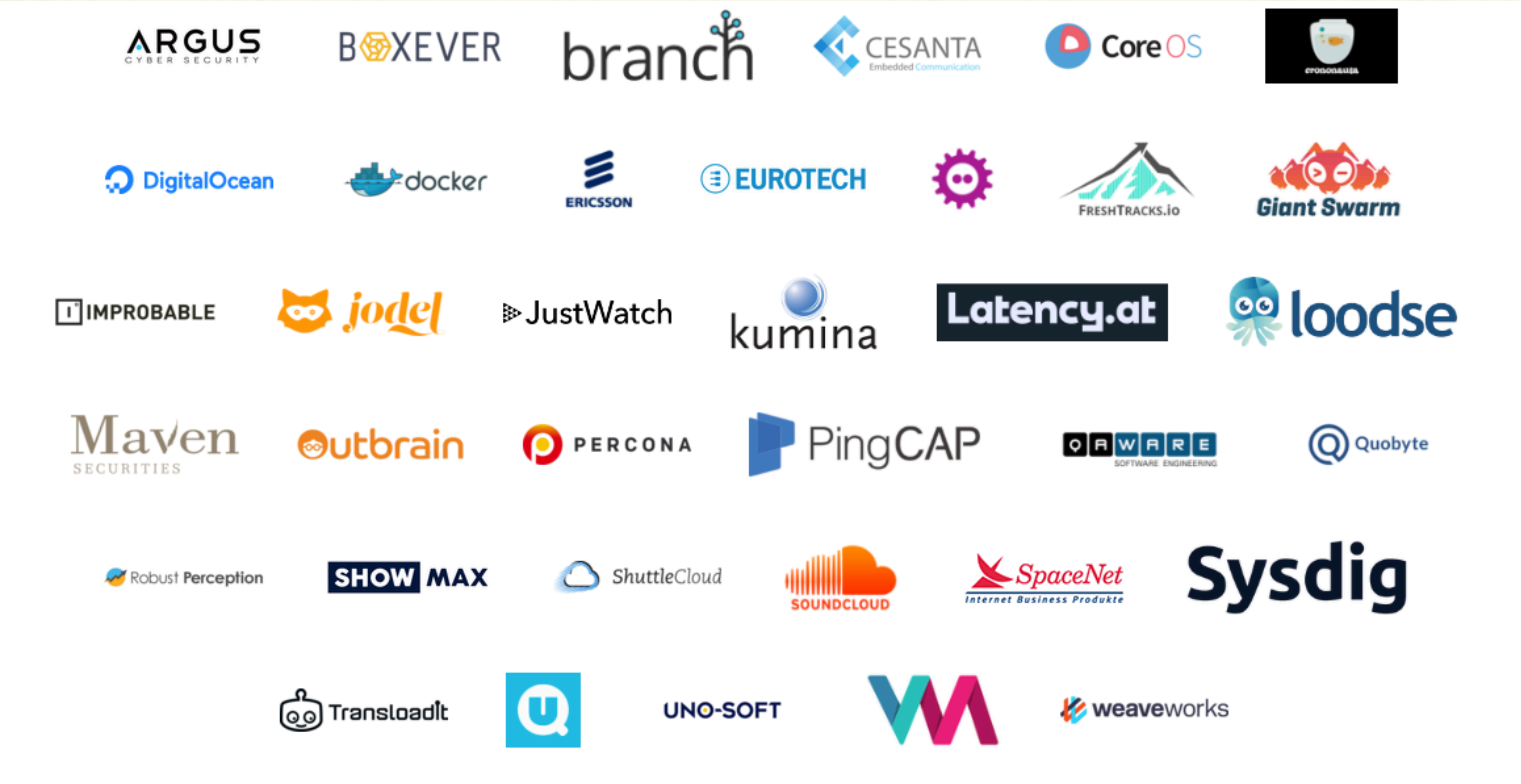
Whitebox Monitoring

- You know your application pretty well
- You can work with information from inside of your system
- Typical examples: from CPU usage to number of requests triggered by some code block inside of your app
- Monitoring Systems: Prometheus

Prometheus

- Inspired by Borgmon project from Google
- Started at Soundcloud
- Written in Go
- 100% Open-Source and community-driven
- CNCF member project

Users


ARGUS CYBER SECURITY
BOXEVER
branch
CESANTA Embedded Communication
CoreOS
DigitalOcean
docker
ERICSSON
EUROTECH
FRESHTRACKS.io
Giant Swarm
IMPROBABLE
jodel
JustWatch
kumina
Latency.at
loodse
Maven SECURITIES
outbrain
PERCONA
PingCAP
QAWARE SOFTWARE ENGINEERING
Quobyte
Robust Perception
SHOW MAX
ShuttleCloud
SOUNDCLOUD
SpaceNet Internet Business Produkte
Sysdig
Transloadit
UNO-SOFT
weaveworks

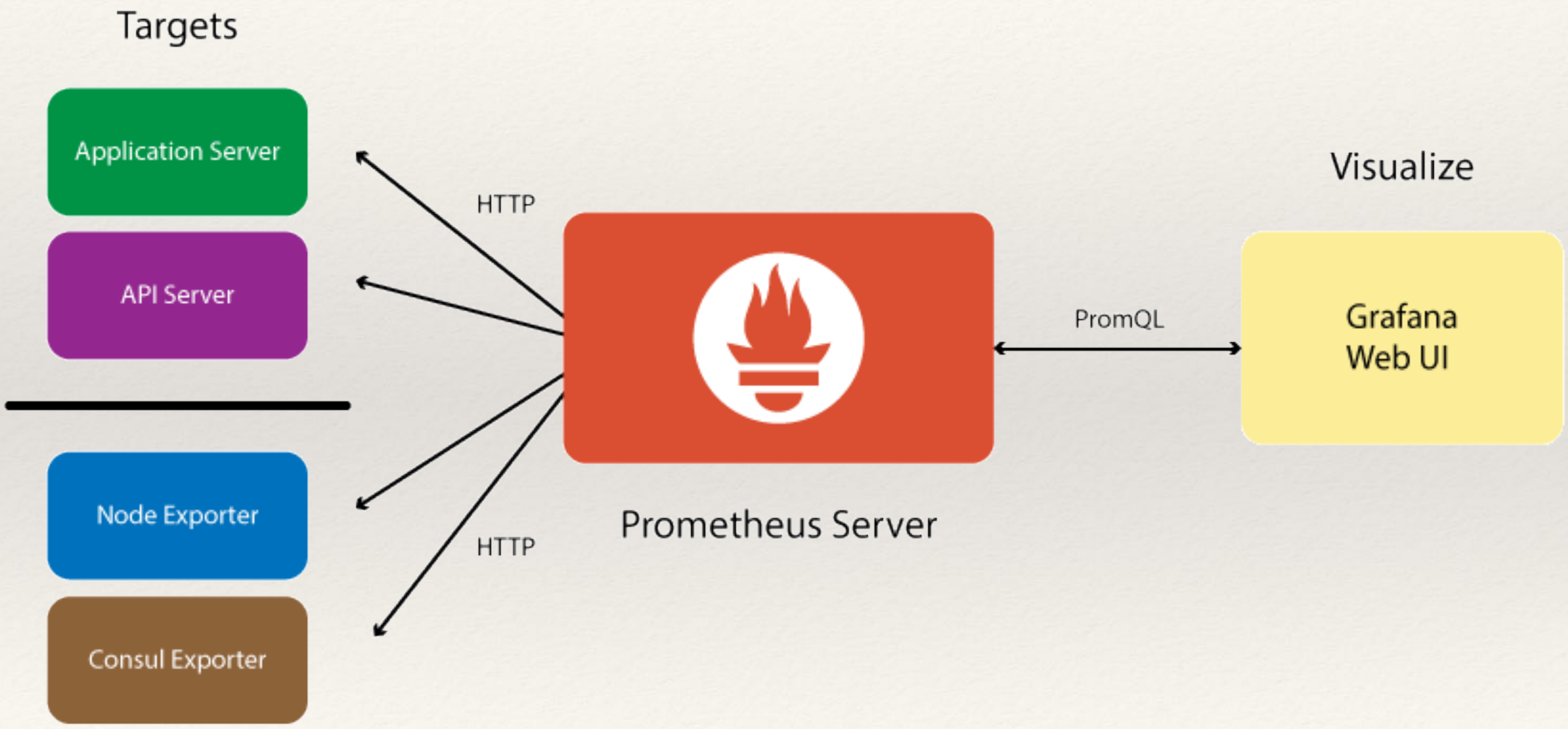
Highlights

- Powerful Queries
- Simple Operation
- Precise Alerting
- Many Client Libraries
- Monitor as the services not as machines
- …

Overview

- **Server** - the one who monitors
- **Targets** - the things that Server Monitors
- **Metrics** - Unit of targets (CPU status, Memory Usage, some specific unit)
- **Time Series Database** - scraped data
- **PromQL** - language that lets the user select and aggregate time series data in real time

Overview


Targets
Application Server
API Server
Node Exporter
Consul Exporter
HTTP
HTTP
Prometheus Server
PromQL
Visualize
Grafana
Web UI

Scraping Targets

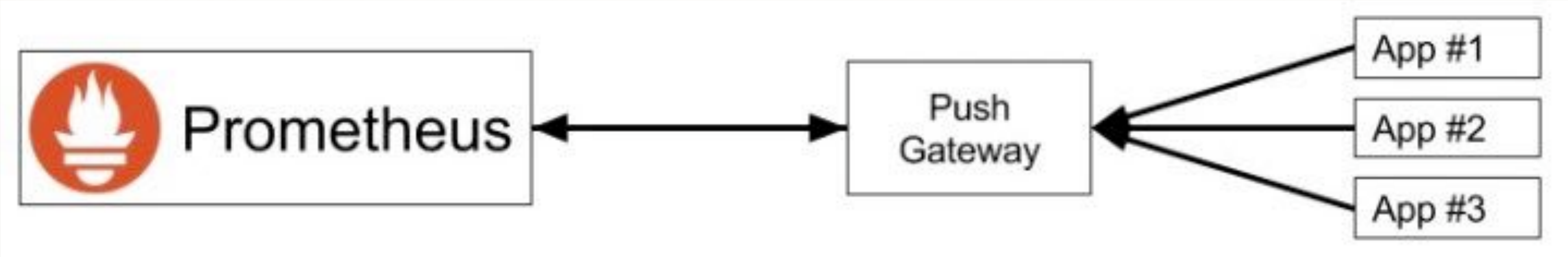
Pull approach

Prometheus uses **Pull Approach,** when metrics sources don’t try to be smart and just provide their readings on demand.

How to scrape?

- For example, **etcd** is “Prometheus -ready”
- There are exporters for many applications
- **Pushgateway** tool

Pushgateway tool



How do the metrics look like?

Prometheus
Alerts
Graph
Status
Help
○ Enable query history
Execute
- insert metric at cursor -
Graph
Console

```
kube_pod_container_status_restarts
```

Load time: 251ms
Resolution: 14s
Total time series: 27

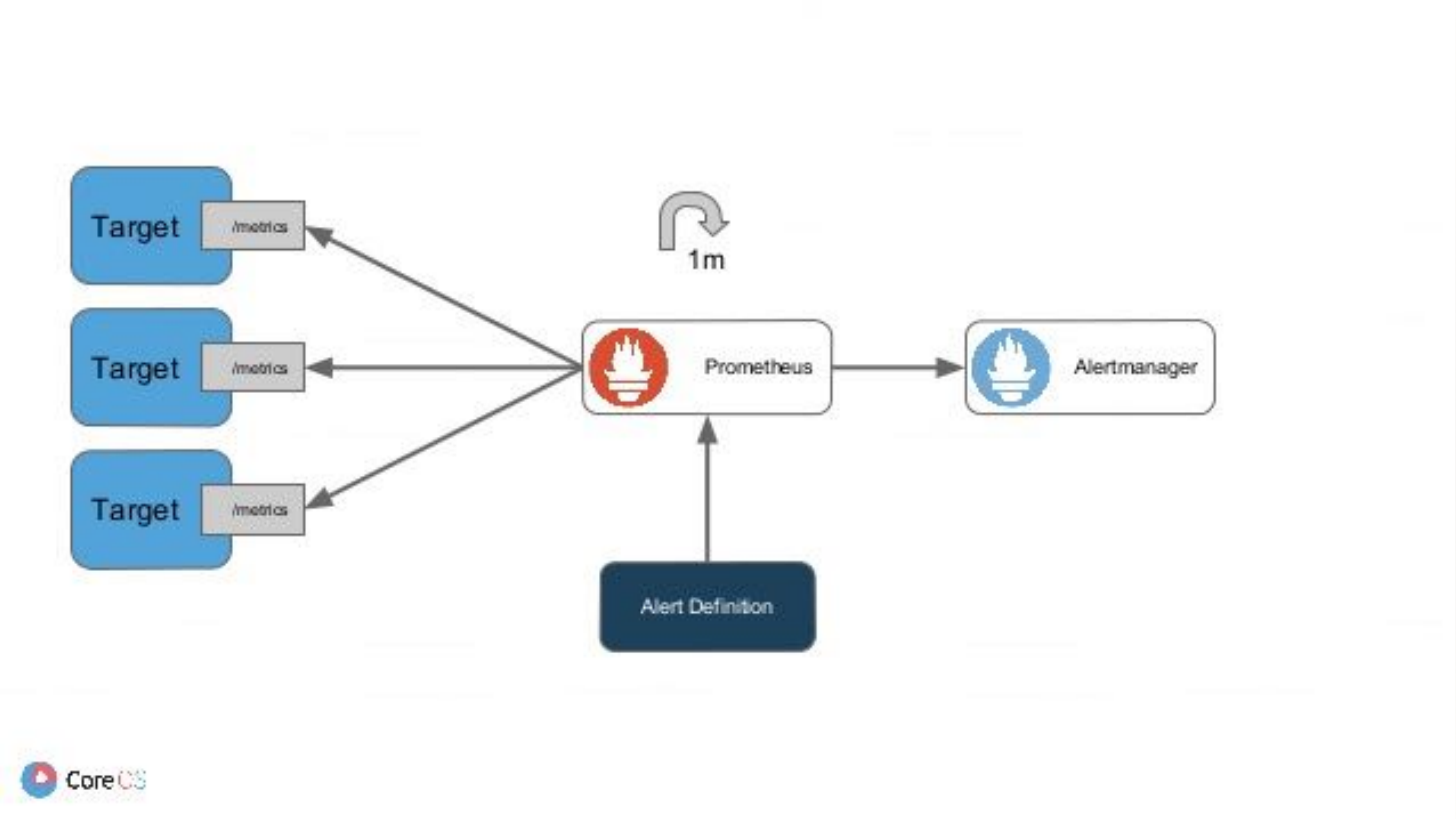
Element	Value
kube_pod_container_status_restarts{container="agent",instance="10.32.0.11:8080",job="kubernetes-service-endpoints",k8s_app="kube-state-metrics",kubernetes_name="kube-state-metrics",kubernetes_namespace="kube-system",namespace="weave-scope",pod="weave-scope-agent-2tz2q"}	4
kube_pod_container_status_restarts{container="alertmanager",instance="10.32.0.11:8080",job="kubernetes-service-endpoints",k8s_app="kube-state-metrics",kubernetes_name="kube-state-metrics",kubernetes_namespace="kube-system",namespace="monitoring",pod="alertmanager-5c6b6cc4b8-cnrp8"}	1
kube_pod_container_status_restarts{container="app",instance="10.32.0.11:8080",job="kubernetes-service-endpoints",k8s_app="kube-state-metrics",kubernetes_name="kube-state-metrics",kubernetes_namespace="kube-system",namespace="weave-scope",pod="weave-scope-app-564cdb7f86-svkbh"}	4
kube_pod_container_status_restarts{container="createcm-volume-test",instance="10.32.0.11:8080",job="kubernetes-service-endpoints",k8s_app="kube-state-metrics",kubernetes_name="kube-state-metrics",kubernetes_namespace="kube-system",namespace="e2e-tests-projected-d7f87",pod="pod-projected-configmaps-c46efa02-3d74-11e8-8b35-666f2e72fa49"}	0
kube_pod_container_status_restarts{container="default-http-backend",instance="10.32.0.11:8080",job="kubernetes-service-endpoints",k8s_app="kube-state-metrics",kubernetes_name="kube-state-metrics",kubernetes_namespace="kube-system",namespace="default",pod="default-http-backend-684b46d757-x5fb4"}	4
kube_pod_container_status_restarts{container="delcm-volume-test",instance="10.32.0.11:8080",job="kubernetes-service-endpoints",k8s_app="kube-state-metrics",kubernetes_name="kube-state-metrics",kubernetes_namespace="kube-system",namespace="e2e-tests-projected-d7f87",pod="pod-projected-configmaps-c46efa02-3d74-11e8-8b35-666f2e72fa49"}	0
kube_pod_container_status_restarts{container="dnsmasq",instance="10.32.0.11:8080",job="kubernetes-service-endpoints",k8s_app="kube-state-metrics",kubernetes_name="kube-state-metrics",kubernetes_namespace="kube-system",namespace="kube-system",pod="kube-dns-f95559775-xpn7q"}	4
kube_pod_container_status_restarts{container="etcd",instance="10.32.0.11:8080",job="kubernetes-service-endpoints",k8s_app="kube-state-metrics",kubernetes_name="kube-state-metrics",kubernetes_namespace="kube-system",namespace="kube-system",pod="etcd-esh-k8s-master01.int.na.intgdc.com"}	5
kube_pod_container_status_restarts{container="grafana",instance="10.32.0.11:8080",job="kubernetes-service-endpoints",k8s_app="kube-state-metrics",kubernetes_name="kube-state-metrics",kubernetes_namespace="kube-system",namespace="kube-system",pod="monitoring-grafana-bfdf6b8c-tbwhd"}	4

Alerting with Prometheus

Alertmanager

- Takes care of routing alerts from Prometheus to a receiver
- Supports: e-mail, HipChat, Slack, PagerDuty etc

Alertmanager



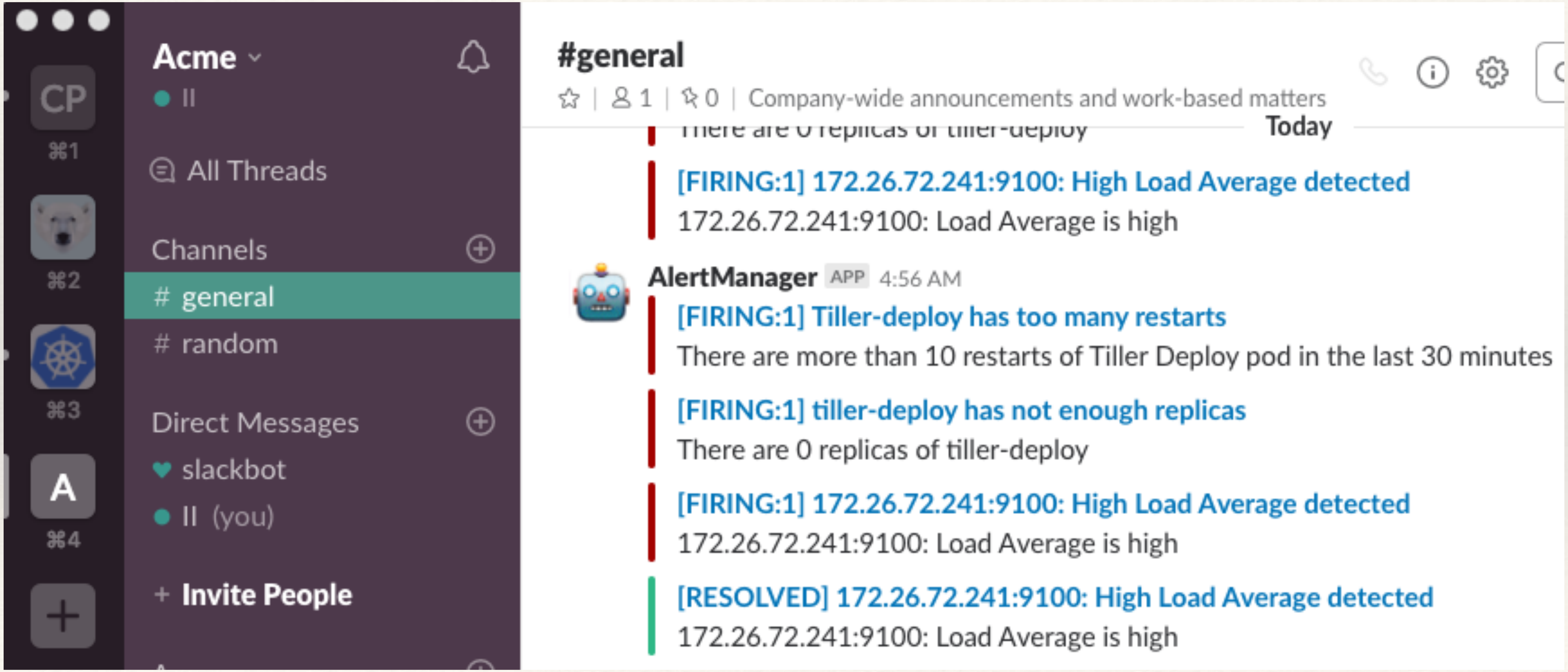
Functions

- Grouping of alerts
- Inhibitions
- Silences

Alert

```
kubernetes.rules.yaml: |+
  groups:
  - name: kubernetes.rules
    rules:
    - alert: "K8S Tiller-Deploy has too many restarts"
      expr: kube_pod_container_status_restarts{namespace='kube-system',pod=~'^tiller-deploy.*'} > 10
      for: 30m
      labels:
        severity: critical
      annotations:
        description: There are more than 10 restarts of Tiller Deploy pod in the last 30 minutes
        summary: Tiller-deploy has too many restarts
```

Example: Slack



There are 0 replicas of tiller-deploy

[FIRING:1] 172.26.72.241:9100: High Load Average detected
172.26.72.241:9100: Load Average is high

AlertManager APP 4:56 AM

[FIRING:1] Tiller-deploy has too many restarts
There are more than 10 restarts of Tiller Deploy pod in the last 30 minutes

[FIRING:1] tiller-deploy has not enough replicas
There are 0 replicas of tiller-deploy

[FIRING:1] 172.26.72.241:9100: High Load Average detected
172.26.72.241:9100: Load Average is high

[RESOLVED] 172.26.72.241:9100: High Load Average detected
172.26.72.241:9100: Load Average is high

Side notes

- PromQL - really powerful tool to make use of the data
- Graphing is the weakest part of Prometheus
- Alertmanager's web-interface is simple with all it’s Pros and Cons

PromCon

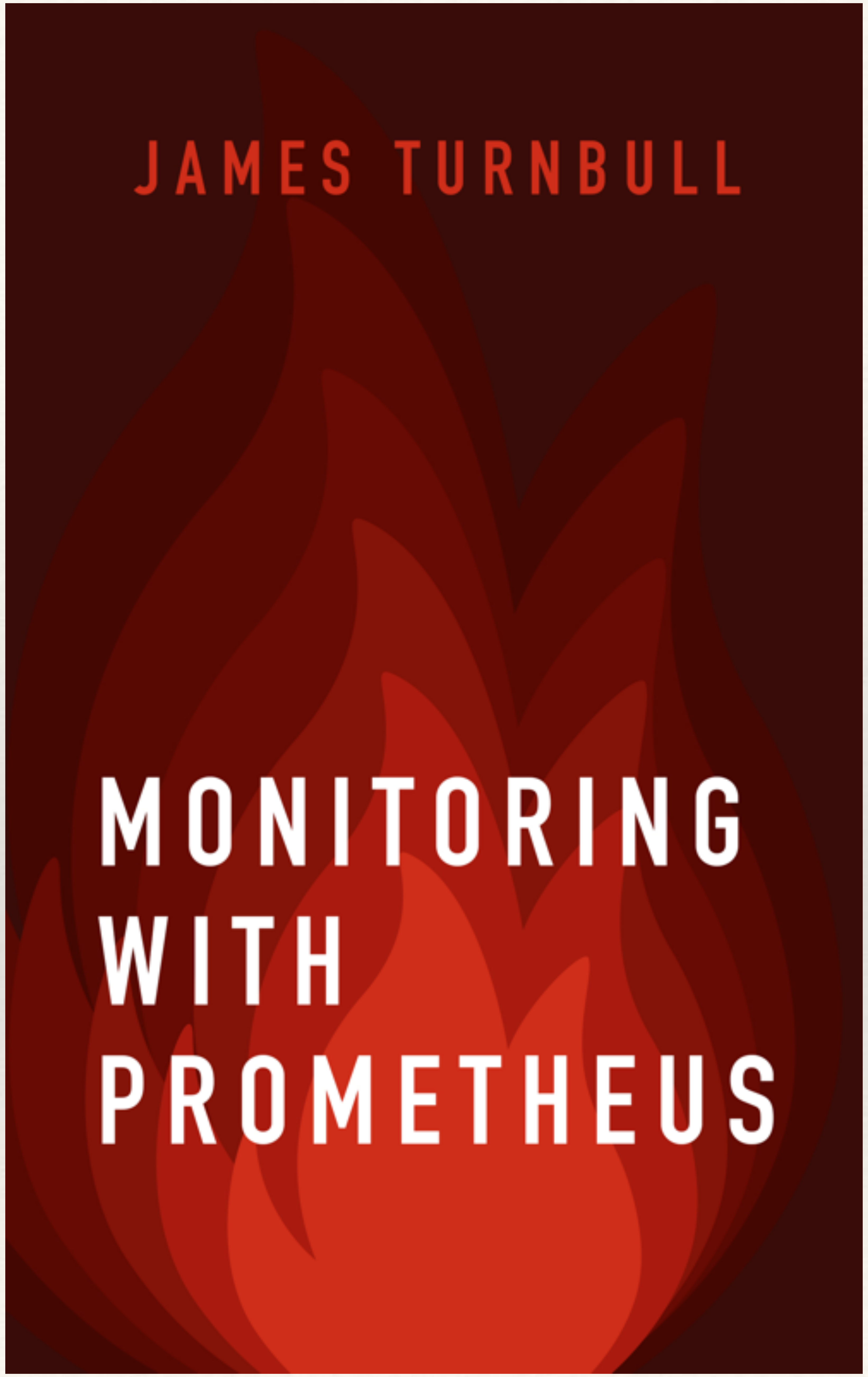
PromCon in Munich on August 9 & 10, 2018

“Prometheus Up and Running”

by Brian Brazil

“Monitoring with Prometheus”

By James Turnbull



Contacts

- @eshmarnev on Twitter
- Evalle on GitHub/GtiLab
- https://evalle.xyz