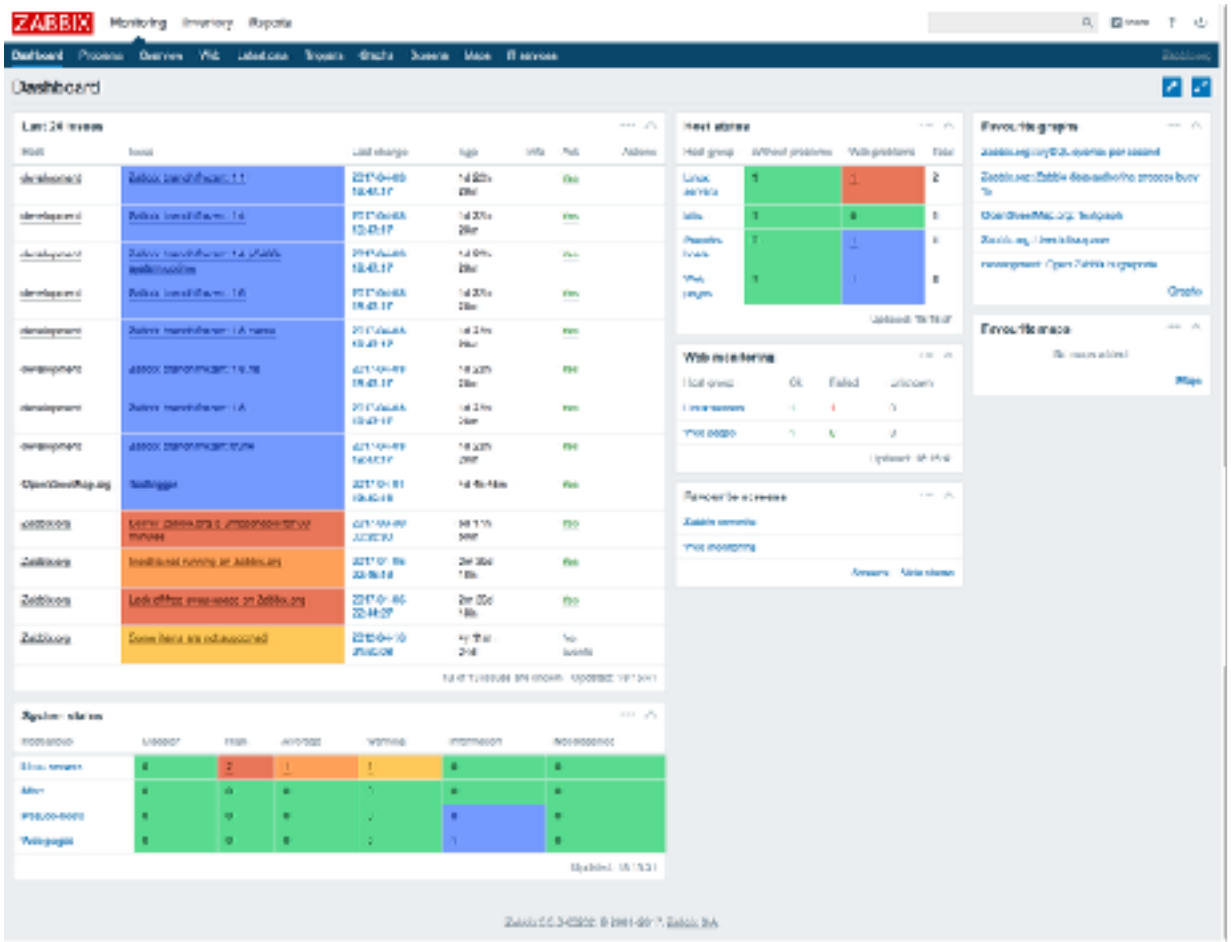
基于Prometheus的数据库监控

杭州沃趣科技股份有限公司
Hangzhou WOQU Technology Co., Ltd.

Let Data Drive!

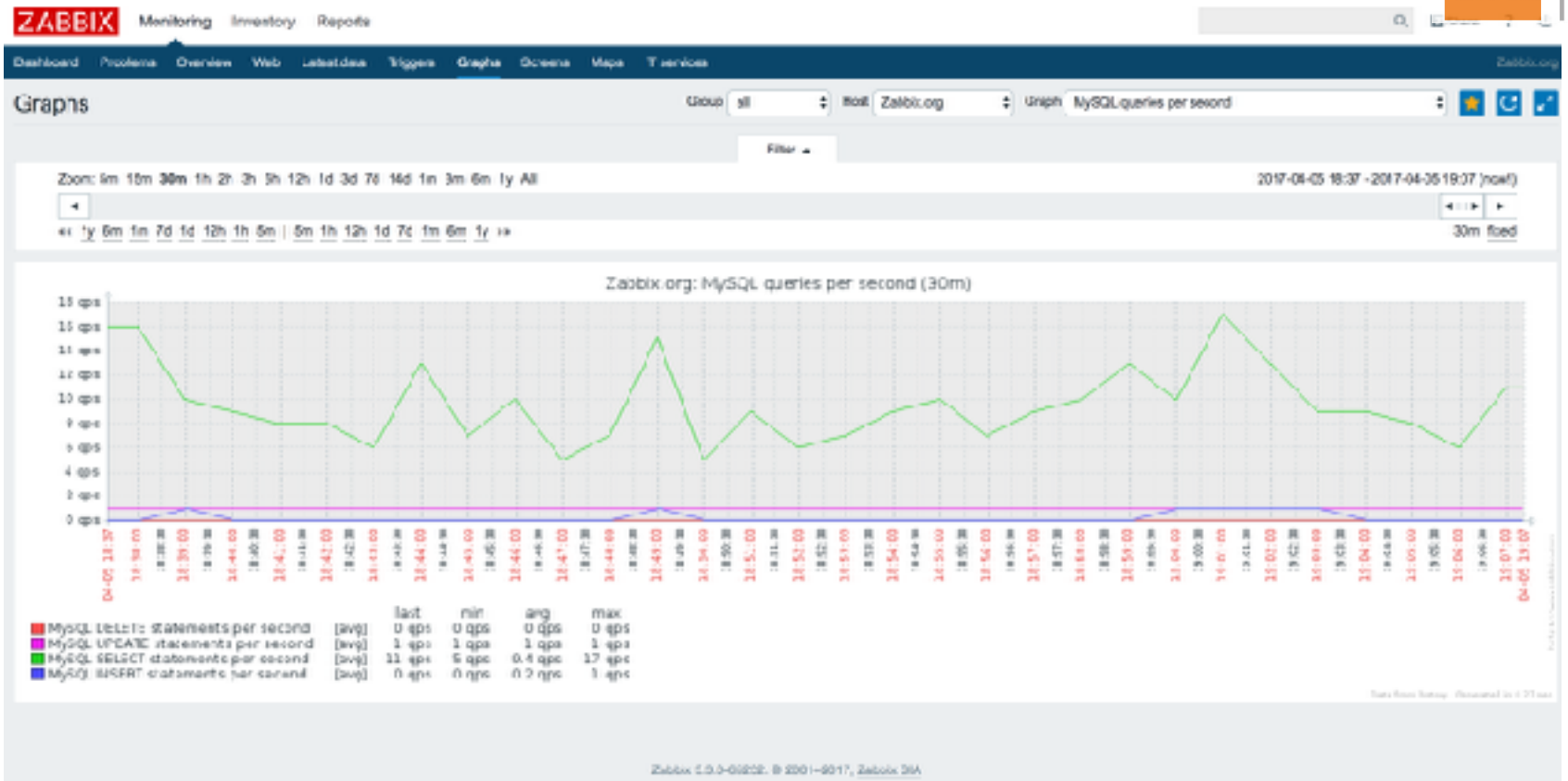
monitoring

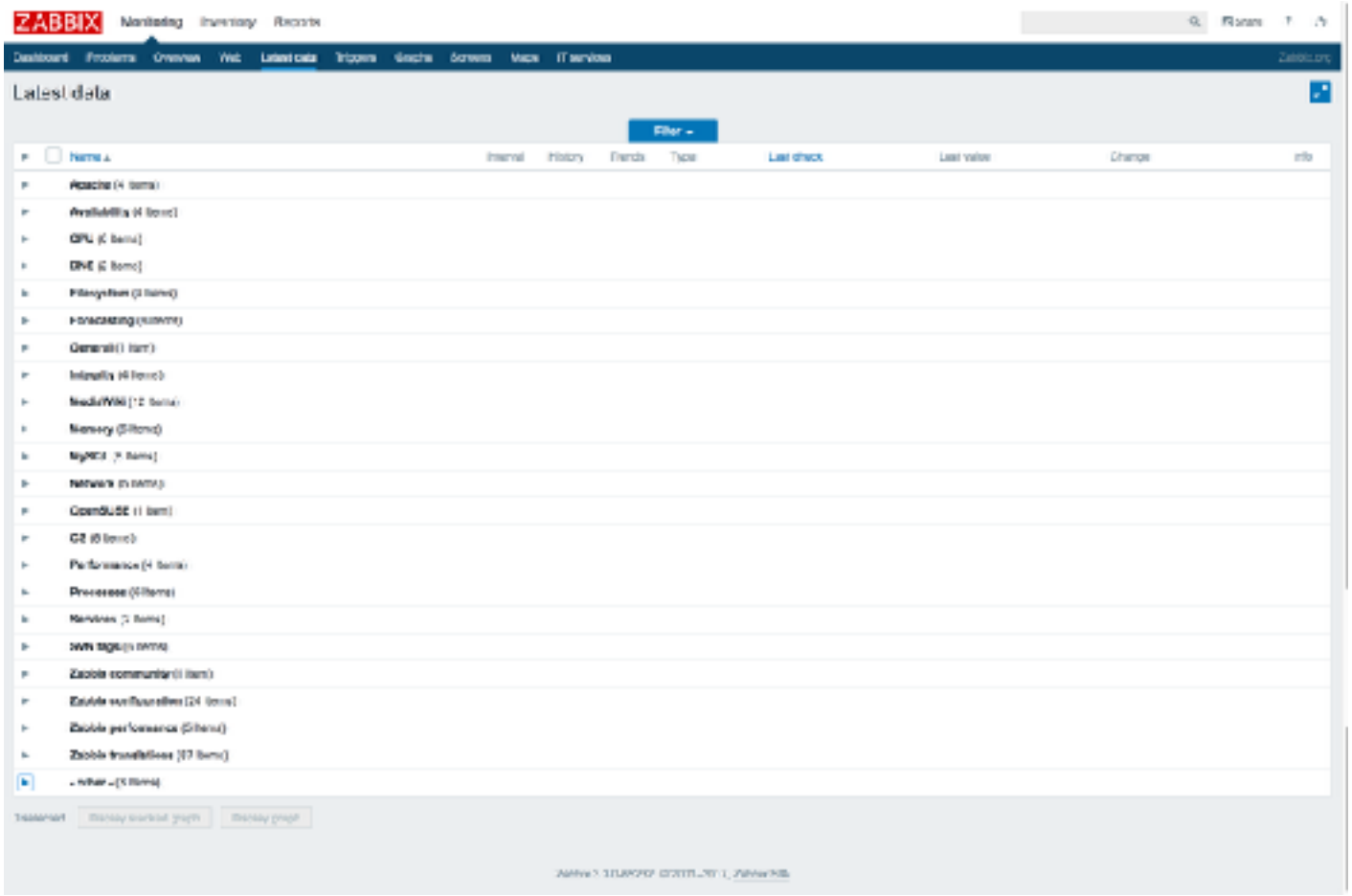


杭州沃趣科技股份有限公司

Hangzhou WOQU Technology Co., Ltd.

Let Data Drive!


ZABBIX
Monitoring
Inventory
Reports
Dashboard
Problems
Overview
Web
Latest data
Triggers
Graphs
Screens
Maps
IT services
Graphs
Group
all
Host
Zabbix.org
Graph
MySQL queries per second
Filter
Zoom: 6m 15m 30m 1h 2h 3h 5h 12h 1d 3d 7d 14d 1m 3m 6m 1y All
2017-04-05 18:37 - 2017-04-05 19:07 (now!)
30m fixed
Zabbix.org: MySQL queries per second (30m)
last
min
avg
max
MySQL SELECT statements per second [avg] 0 qps 0 qps 0 qps 0 qps
MySQL UPDATE statements per second [avg] 1 qps 1 qps 1 qps 1 qps
MySQL SELECT statements per second [avg] 11 qps 6 qps 8.4 qps 17 qps
MySQL INSERT statements per second [avg] 0 qps 0 qps 0.2 qps 1 qps



杭州沃趣科技股份有限公司

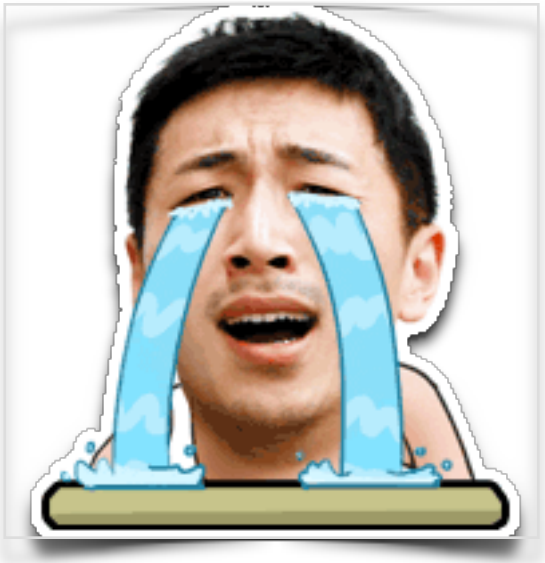
Hangzhou WOQU Technology Co., Ltd.

Let Data Drive!

现状

服务器数量和业务的增长...

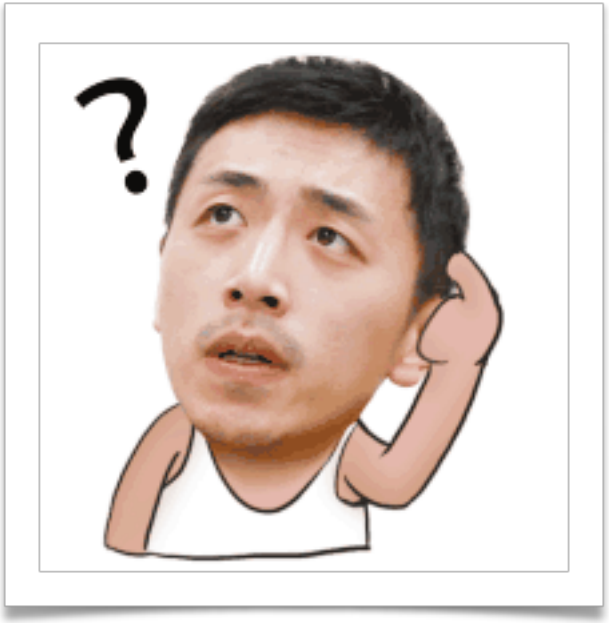
- DB性能瓶颈
- 多套部署，管理成本高
- 易用性差
- 邮件风暴



天天加班，女朋友闹分手

未来

- 容器的监控?
- 微服务的监控?
- 集群性能的聚合分析计算?
- 大量配置脚本的管理?



Large-scale cluster management at Google with Borg

Abhishek Verma[1] Luis Pedrosa[2] Madhukar Korupolu
David Oppenheimer Eric Tune John Wilkes
Google Inc.

Abstract

Google's Borg system is a cluster manager that runs hundreds of thousands of jobs, from many thousands of different applications, across a number of clusters each with up to tens of thousands of machines.

It achieves high utilization by combining admission control, efficient task-packing, over-commitment, and machine sharing with process-level performance isolation. It supports high-availability applications with runtime features that minimize fault-recovery time, and scheduling policies that reduce the probability of correlated failures. Borg simplifies life for its users by offering a declarative job specification language, name service integration, real-time job monitoring, and tools to analyze and simulate system behavior.

We present a summary of the Borg system architecture and features, important design decisions, a quantitative analysis of some of its policy decisions, and a qualitative examination of lessons learned from a decade of operational experience with it.

1. Introduction

The cluster management system we internally call Borg admits, schedules, starts, restarts, and monitors the full range of applications that Google runs. This paper explains how.

Borg provides three main benefits: it (1) hides the details of resource management and failure handling so its users can focus on application development instead; (2) operates with very high reliability and availability, and supports applications that do the same; and (3) lets us run workloads across tens of thousands of machines effectively. Borg is not the first system to address these issues, but it's one of the few operating at this scale, with this degree of resiliency and completeness. This paper is organized around these topics, concluding with a set of qualitative observations we have made from operating Borg in production for more than a decade.

[1] Work done while author was at Google.
[2] Currently at University of Southern California

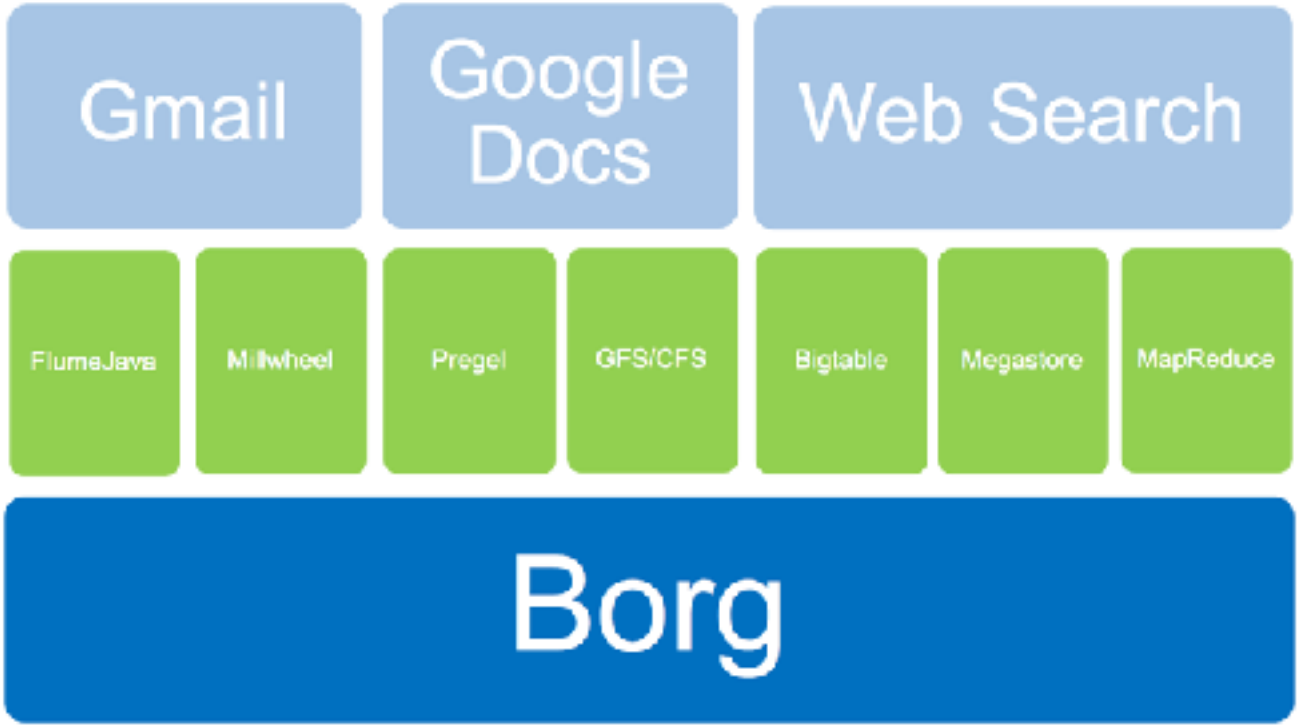
Figure 1: The high-level architecture of Borg. *Only a tiny fraction of the thousands of worker nodes are shown.*

2. The user perspective

Borg's users are Google developers and system administrators (site reliability engineers or SREs) that run Google's applications and services. Users submit their work to Borg in the form of *jobs*, each of which consists of one or more *tasks* that all run the same program (binary). Each job runs in one Borg *cell*, a set of machines that are managed as a unit. The remainder of this section describes the main features exposed in the user view of Borg.

2.1 The workload

Borg cells run a heterogenous workload with two main parts. The first is long-running services that should "never" go down, and handle short-lived latency-sensitive requests (a



The **cluster management** system we internally call Borg

admits, schedules, starts, restarts, and monitors

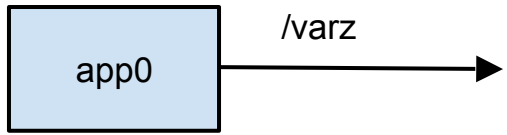
the full range of applications that Google runs.

10 years

Borgmon监控

- 单集群上万服务器
- 几千个不同的应用
- 几十万个以上的jobs
- 多个集群

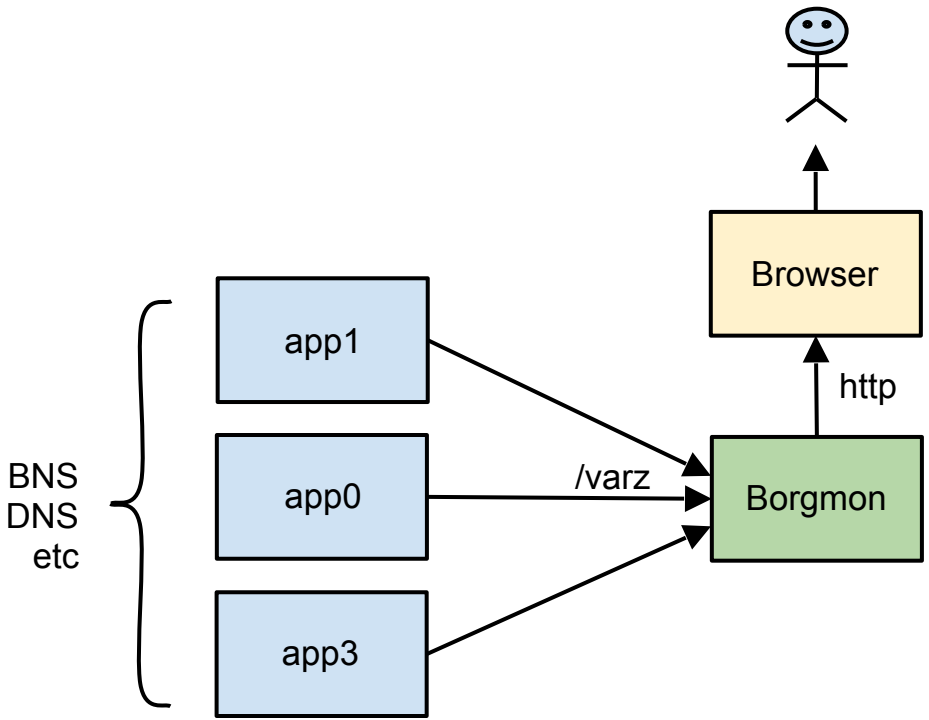
Borgmon监控



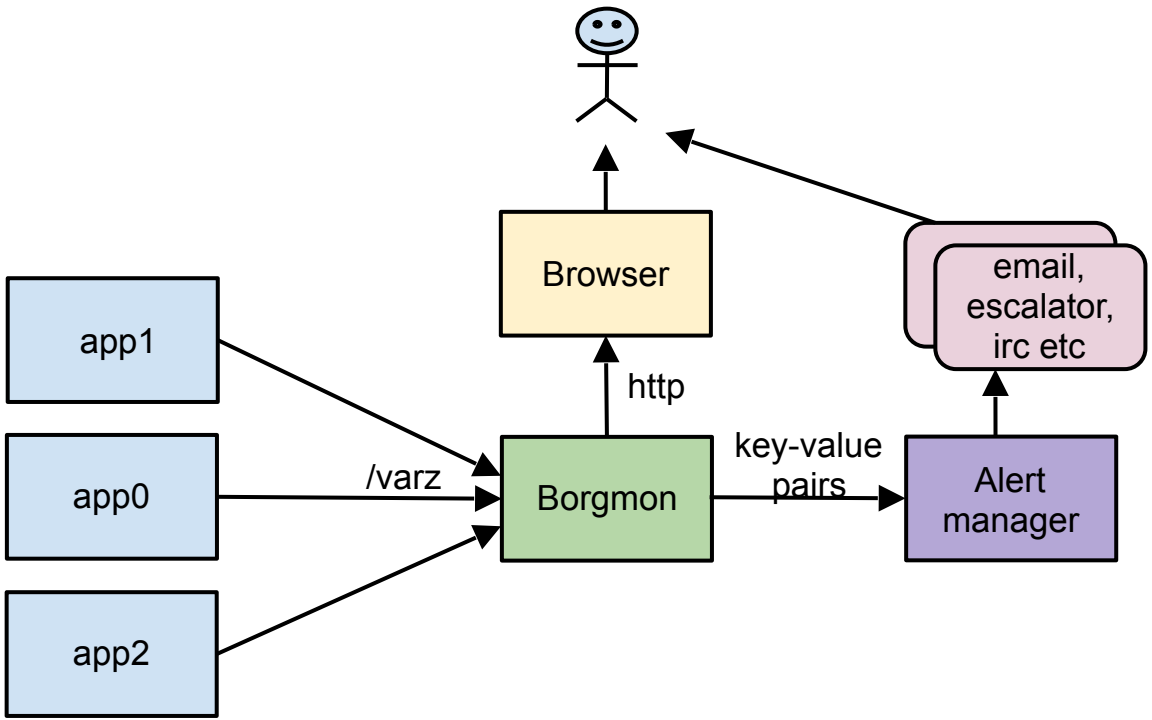
curl http://app0:80/varz

http_requests 37
errors_total 12

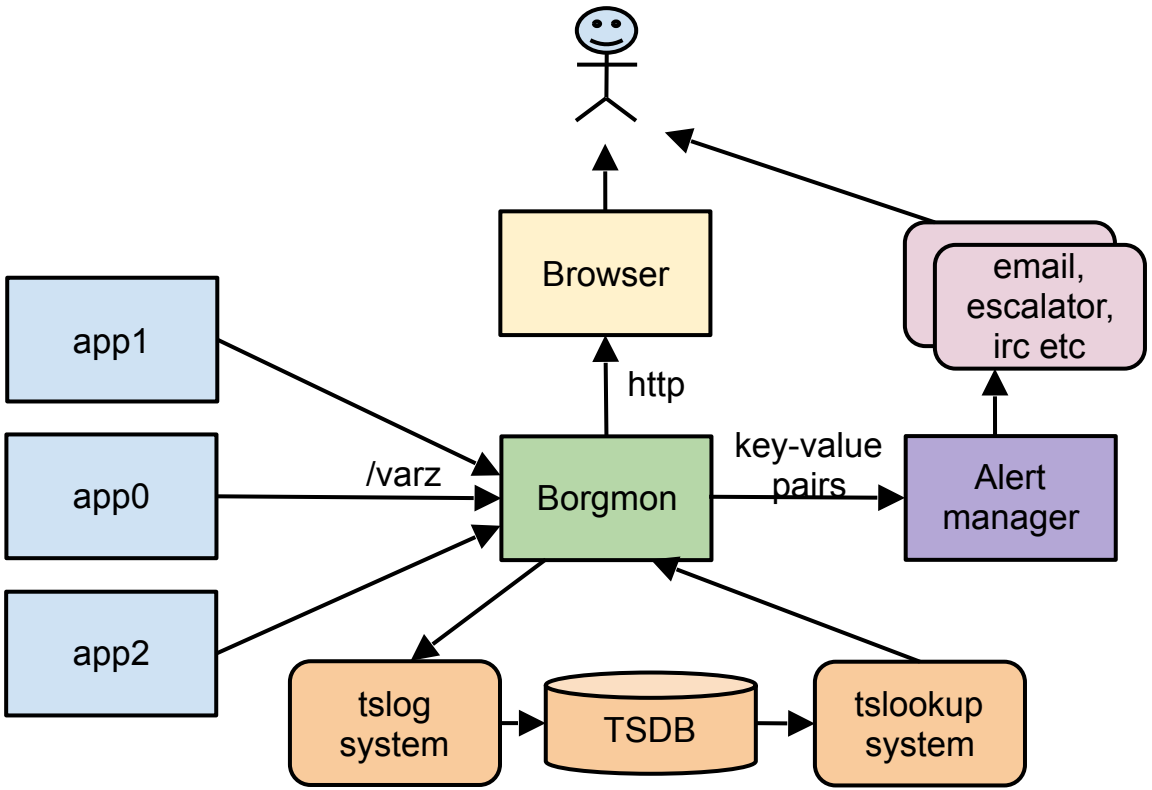
Borgmon监控



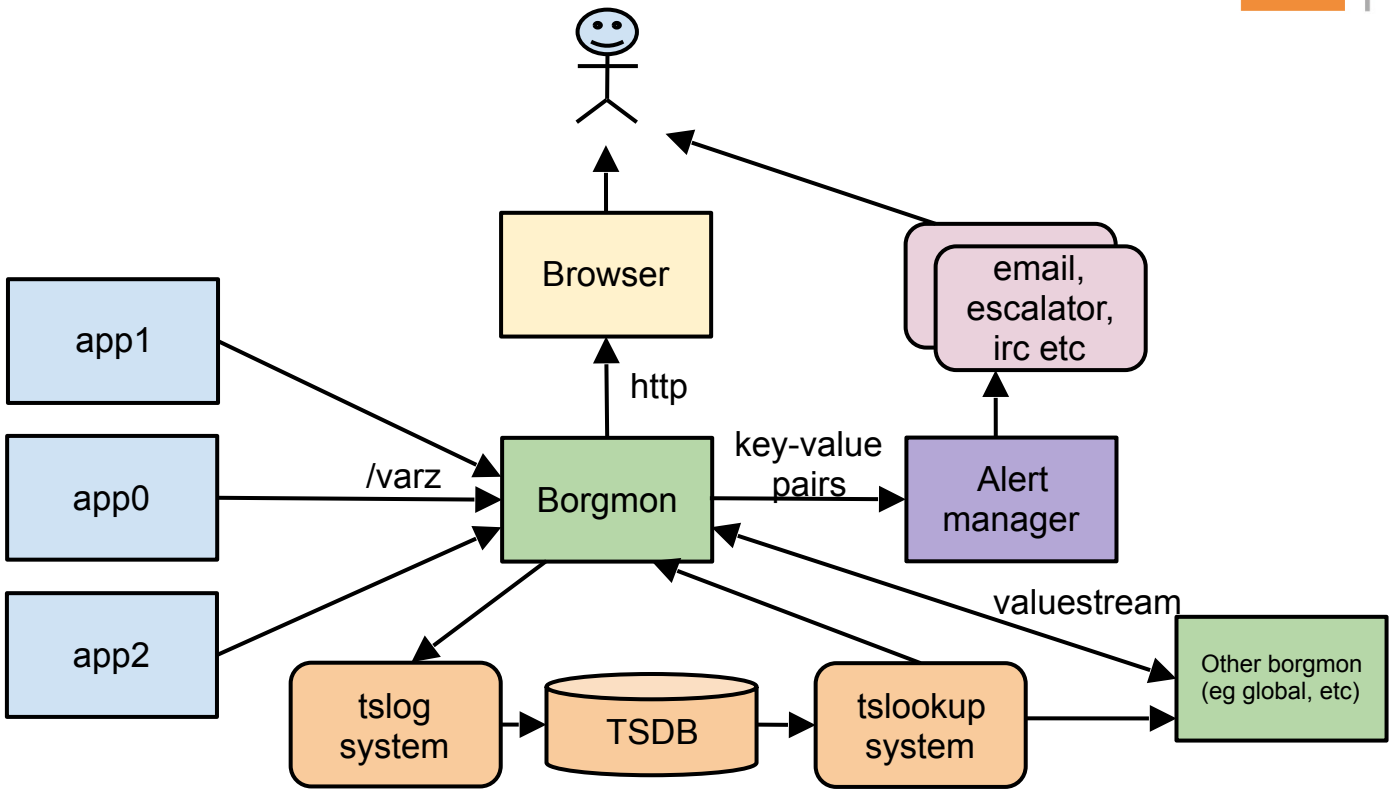
Borgmon监控


Browser
email,
escalator,
irc etc
app1
http
app0
/varz
Borgmon
key-value
pairs
Alert
manager
app2

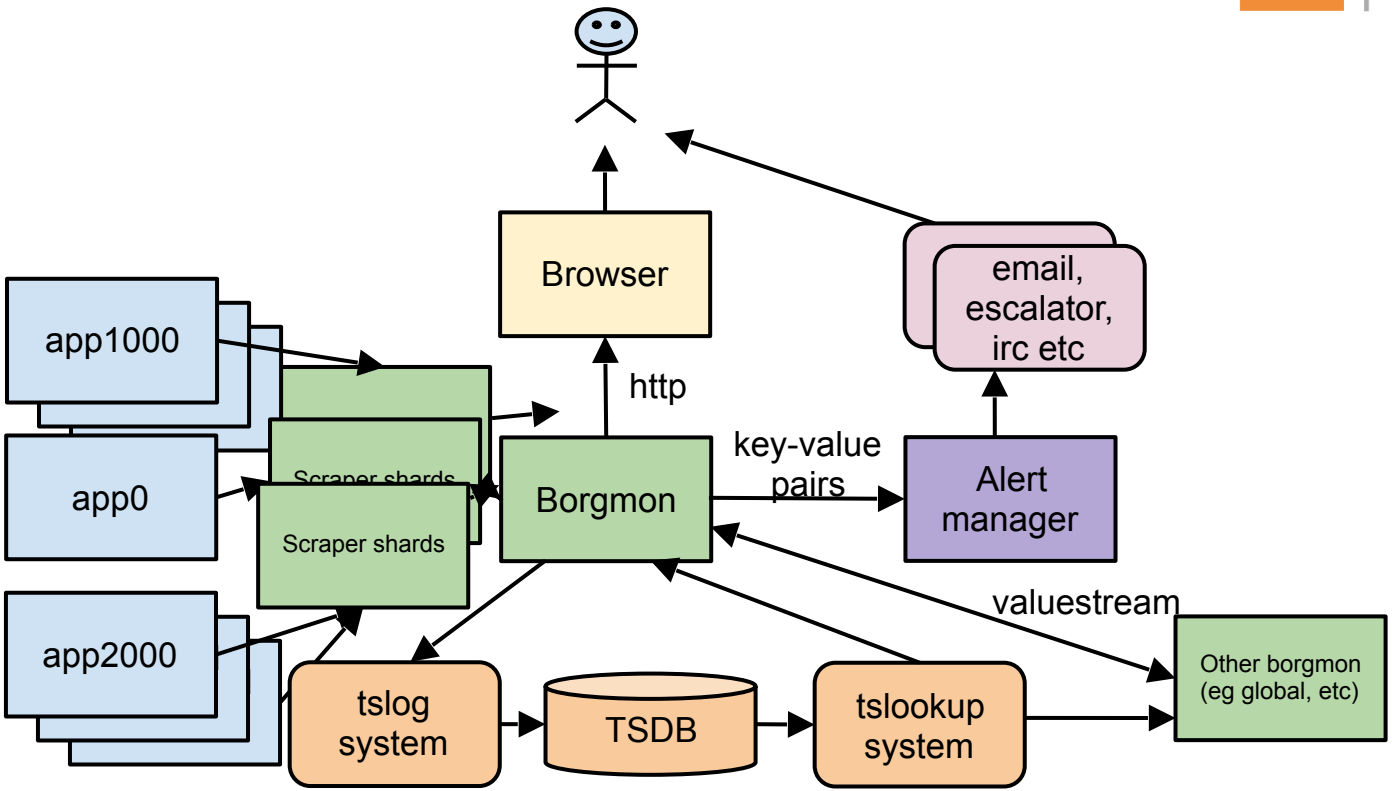
Borgmon监控


Browser
email,
escalator,
irc etc
app1
http
app0
/varz
Borgmon
key-value
pairs
Alert
manager
app2
tslog
system
TSDB
tslookup
system

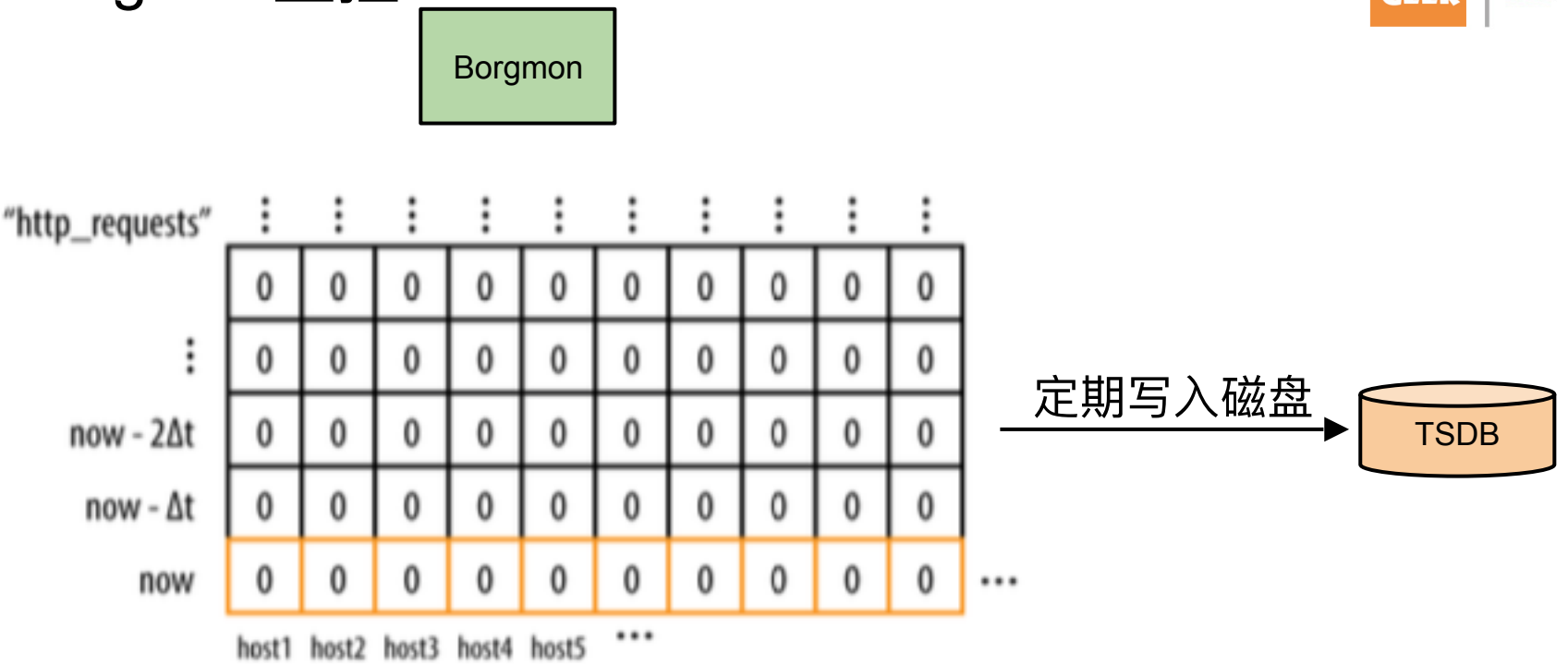
Borgmon监控


Browser
email,
escalator,
irc etc
app1
http
app0
/varz
Borgmon
key-value
pairs
Alert
manager
app2
valuestream
tslog
system
TSDB
tslookup
system
Other borgmon
(eg global, etc)

Borgmon监控


Browser
email,
escalator,
irc etc
app1000
http
app0
Scraper shards
Scraper shards
Borgmon
key-value
pairs
Alert
manager
app2000
valuestream
tslog
system
TSDB
tslookup
system
Other borgmon
(eg global, etc)

Borgmon监控


Borgmon
"http_requests"
now - 2Δt
now - Δt
now
host1 host2 host3 host4 host5
定期写入磁盘
TSDB

查询某个性能指标

```
{var=http_requests,job=webserver,instance=host0:80,service=web,zone=us-west}
```

查询一个集群的性能指标

```
{var=http_requests,job=webserver,service=web,zone=us-west}
```

结果

```
{var=http_requests,job=webserver,instance=host0:80,service=web,zone=us-west} 10
{var=http_requests,job=webserver,instance=host1:80,service=web,zone=us-west} 9
{var=http_requests,job=webserver,instance=host2:80,service=web,zone=us-west} 11
{var=http_requests,job=webserver,instance=host3:80,service=web,zone=us-west} 0
{var=http_requests,job=webserver,instance=host4:80,service=web,zone=us-west} 10
```

Rule Evaluation

```
rules <<<
      # Compute the rate of requests for each task from the count of requests
      {var=task:http_requests:rate10m,job=webserver} =
        rate({var=http_requests,job=webserver}[10m]);
      # Sum the rates to get the aggregate rate of queries for the cluster;
      # 'without instance' instructs Borgmon to remove the instance label
      # from the right hand side.
      {var=dc:http_requests:rate10m,job=webserver} =
        sum without instance({var=task:http_requests:rate10m,job=webserver})
    >>>
```

Alerting

```
rules <<<
      {var=dc:http_errors:ratio_rate10m,job=webserver} > 0.01
        and by job, error
      {var=dc:http_errors:rate10m,job=webserver} > 1
        for 2m
        => ErrorRatioTooHigh
          details "webserver error ratio at [[trigger_value]]"
          labels {severity=page};
    >>>
```

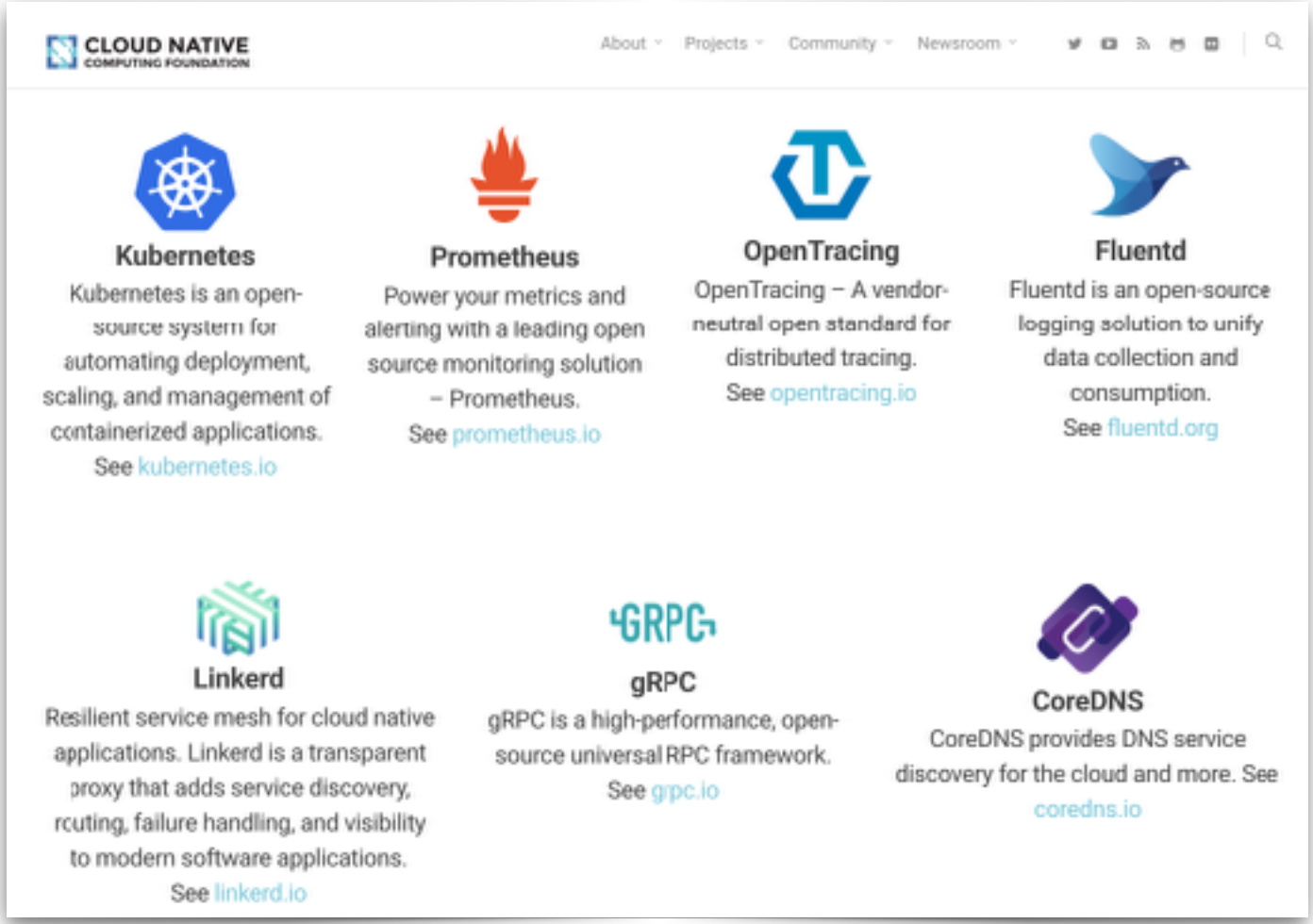
Prometheus build passing

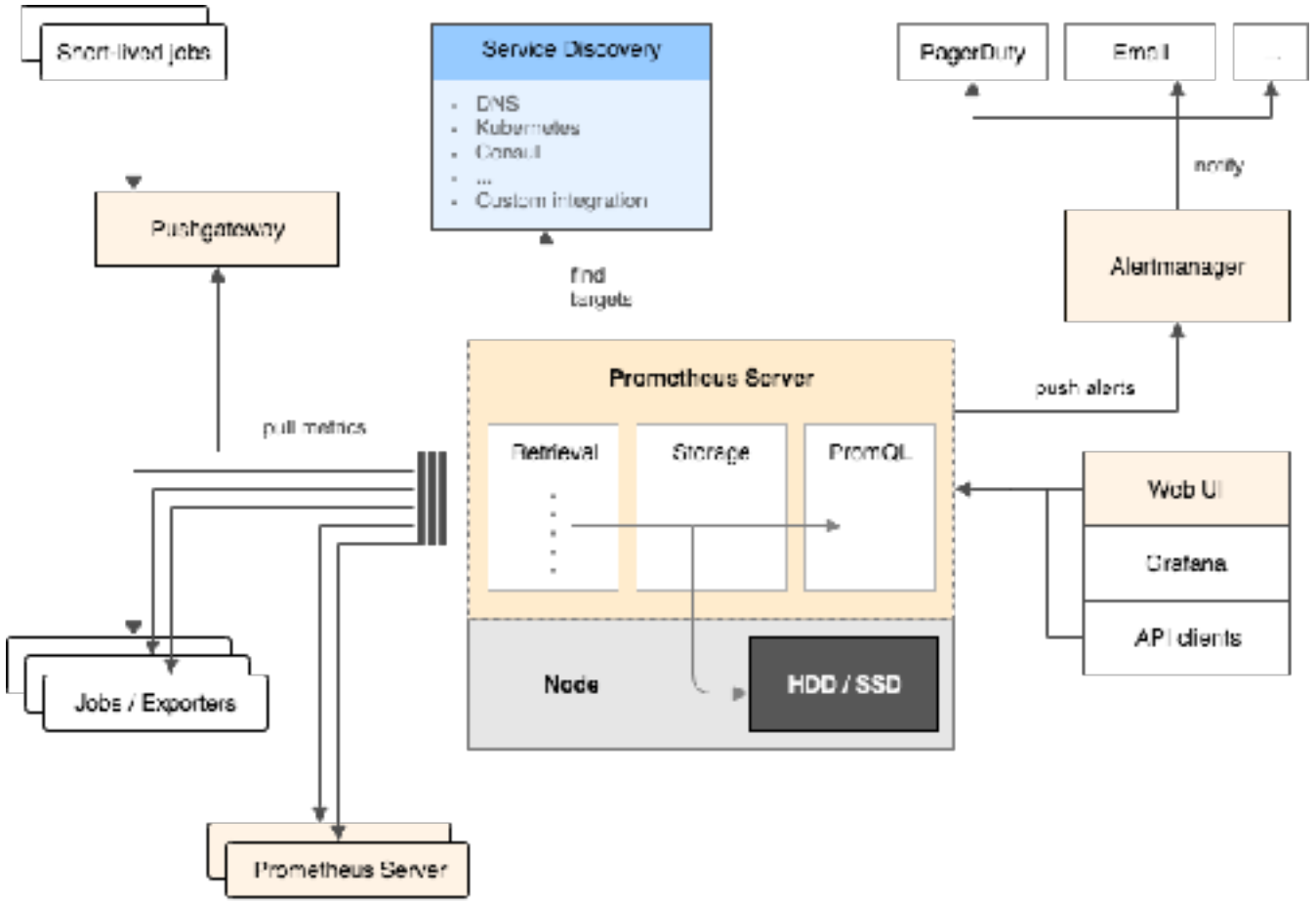
Visit prometheus.io for the full documentation, examples and guides.

Prometheus, a Cloud Native Computing Foundation project, is a systems and service monitoring system. It collects metrics from configured targets at given intervals, evaluates rule expressions, displays the results, and can trigger alerts if some condition is observed to be true.

«Even though Borgmon remains internal to Google, the idea of treating time-series data as a data source for generating alerts is now accessible to everyone through those open source tools like Prometheus [...]»

— **Site Reliability Engineering:** How Google Runs Production Systems (O'Reilly Media)

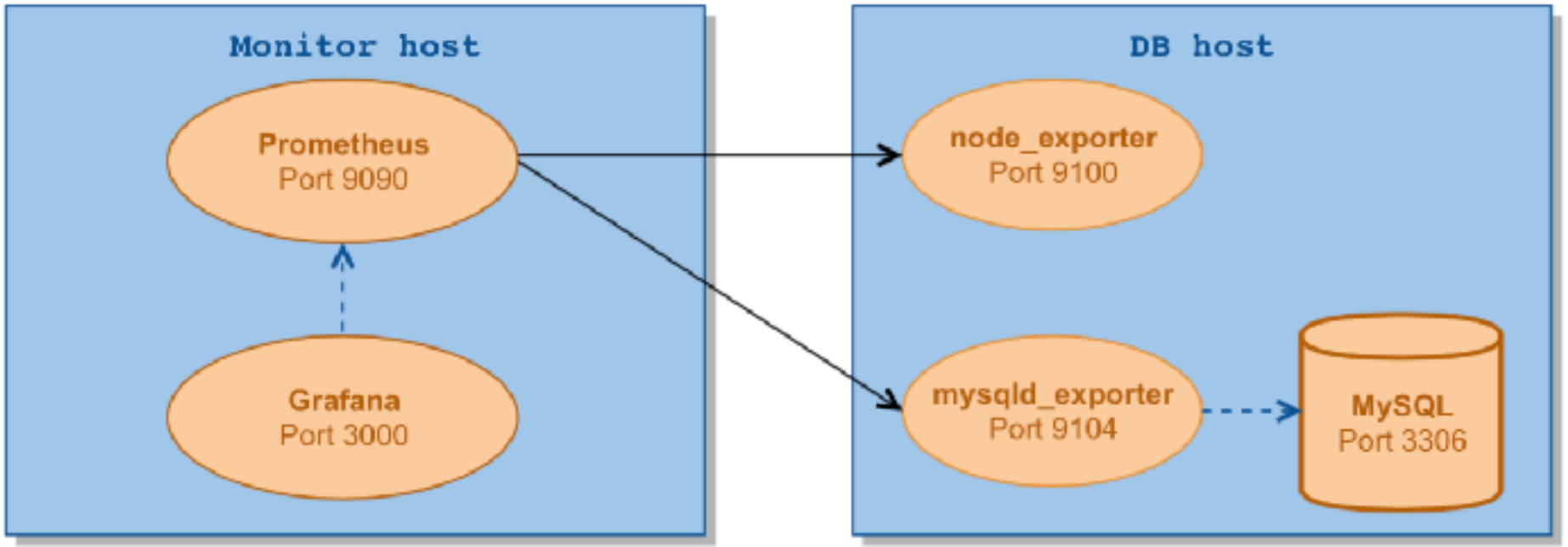

CLOUD NATIVE
COMPUTING FOUNDATION
About
Projects
Community
Newsroom
Kubernetes
Kubernetes is an open-source system for automating deployment, scaling, and management of containerized applications.
See kubernetes.io
Prometheus
Power your metrics and alerting with a leading open source monitoring solution – Prometheus.
See prometheus.io
OpenTracing
OpenTracing – A vendor-neutral open standard for distributed tracing.
See opentracing.io
Fluentd
Fluentd is an open-source logging solution to unify data collection and consumption.
See fluentd.org
Linkerd
Resilient service mesh for cloud native applications. Linkerd is a transparent proxy that adds service discovery, routing, failure handling, and visibility to modern software applications.
See linkerd.io
gRPC
gRPC is a high-performance, open-source universal RPC framework.
See grpc.io
CoreDNS
CoreDNS provides DNS service discovery for the cloud and more. See coredns.io


Short-lived jobs
Service Discovery
DNS
Kubernetes
Consul
...
Custom integration
PagerDuty
Email
...
notify
Pushgateway
find targets
Alertmanager
Prometheus Server
push alerts
pull metrics
Retrieval
Storage
PromQL
Web UI
Grafana
API clients
Node
HDD / SSD
Jobs / Exporters
Prometheus Server

Database Monitoring

杭州沃趣科技股份有限公司
Hangzhou WOQU Technology Co., Ltd.

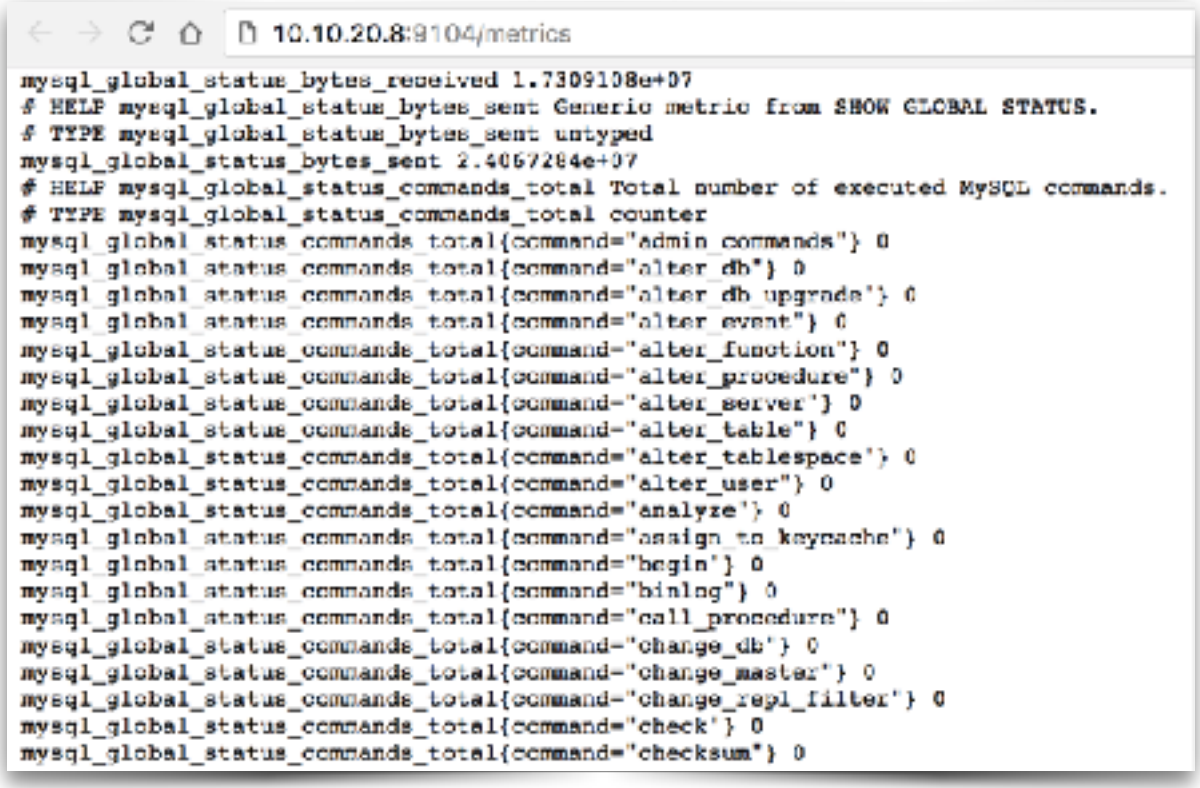
Let Data Drive!


Monitor host
Prometheus
Port 9090
Grafana
Port 3000
DB host
node_exporter
Port 9100
mysqld_exporter
Port 9104
MySQL
Port 3306

```
[root@server prometheus]# tree -L 1
.
├── alertmanager
├── alertmanager.yml
├── host.yml
├── LICENSE
├── mysql.yml
├── NOTICE
├── prometheus
└── prometheus.yml

0 directories, 8 files
```

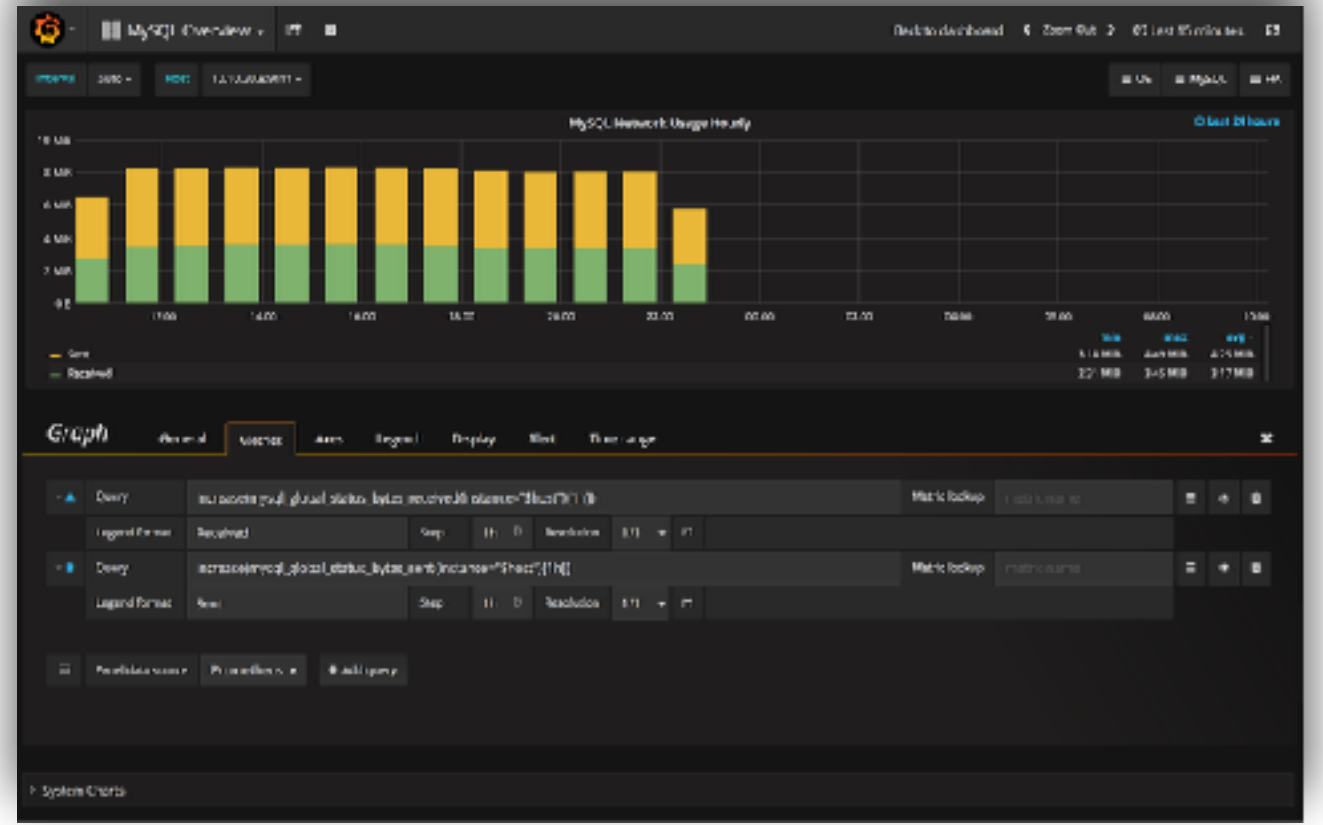
```
[root@server prometheus]# cat prometheus.yml
global:
  evaluation_interval: 15s
  external_labels:
    monitor: QMonitorPlus
  scrape_interval: 15s
rule_files: □
scrape_configs:
- file_sd_configs:
  - files:
    - host.yml
  job_name: Host
  scrape_interval: 1m
- file_sd_configs:
  - files:
    - mysql.yml
  job_name: MySQL
  scrape_interval: 1m
- job_name: prometheus
  static_configs:
  - targets:
    - localhost:9090
[root@server prometheus]# cat mysql.yml
- targets:
  - 10.10.20.7:9104
  - 10.10.20.7:5555
```

```
[root@server prometheus]# export DATA_SOURCE_NAME='test:test@(10.10.20.8:9306)/'
[root@server prometheus]# /usr/local/bin/mysqld_exporter
INFO[0000] Starting mysqld_exporter (version=0.9.0, branch=master, revision=8400af20ccdbf6b5e0faa2c925c56c48cdf8d70b)  source=mysqld_exporter.go:432
INFO[0000] Build context (go=go1.6.3, user=root@2c131c66ca20, date=20160926-18:28:09)  source=mysqld_exporter.go:433
INFO[0000] Listening on :9104                            source=mysqld_exporter.go:451
```

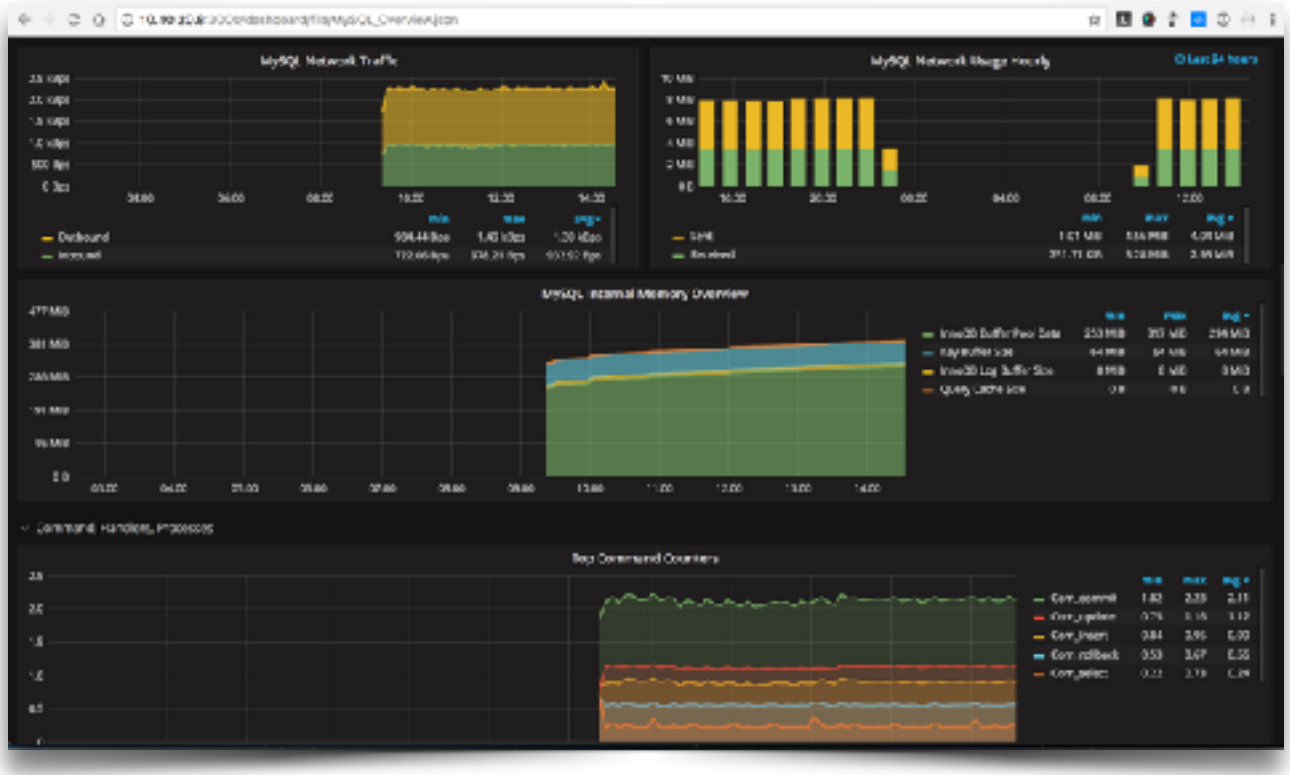

10.10.20.8:9104/metrics
mysql_global_status_bytes_received 1.7309108e+07
HELP mysql_global_status_bytes_sent Generic metric from SHOW GLOBAL STATUS.
TYPE mysql_global_status_bytes_sent untyped
mysql_global_status_bytes_sent 2.4067284e+07
HELP mysql_global_status_commands_total Total number of executed MySQL commands.
TYPE mysql_global_status_commands_total counter
mysql_global_status_commands_total{command="admin_commands"} 0
mysql_global_status_commands_total{command="alter_db"} 0
mysql_global_status_commands_total{command="alter_db_upgrade"} 0
mysql_global_status_commands_total{command="alter_event"} 0
mysql_global_status_commands_total{command="alter_function"} 0
mysql_global_status_commands_total{command="alter_procedure"} 0
mysql_global_status_commands_total{command="alter_server"} 0
mysql_global_status_commands_total{command="alter_table"} 0
mysql_global_status_commands_total{command="alter_tablespace"} 0
mysql_global_status_commands_total{command="alter_user"} 0
mysql_global_status_commands_total{command="analyze"} 0
mysql_global_status_commands_total{command="assign_to_keycache"} 0
mysql_global_status_commands_total{command="begin"} 0
mysql_global_status_commands_total{command="binlog"} 0
mysql_global_status_commands_total{command="call_procedure"} 0
mysql_global_status_commands_total{command="change_db"} 0
mysql_global_status_commands_total{command="change_master"} 0
mysql_global_status_commands_total{command="change_repl_filter"} 0
mysql_global_status_commands_total{command="check"} 0
mysql_global_status_commands_total{command="checksum"} 0

irate(mysql_global_status_bytes_received(instance="10.10.20.8:9104"[5m])

increase(mysql_global_status_bytes_received{instance="$host"}[1h])

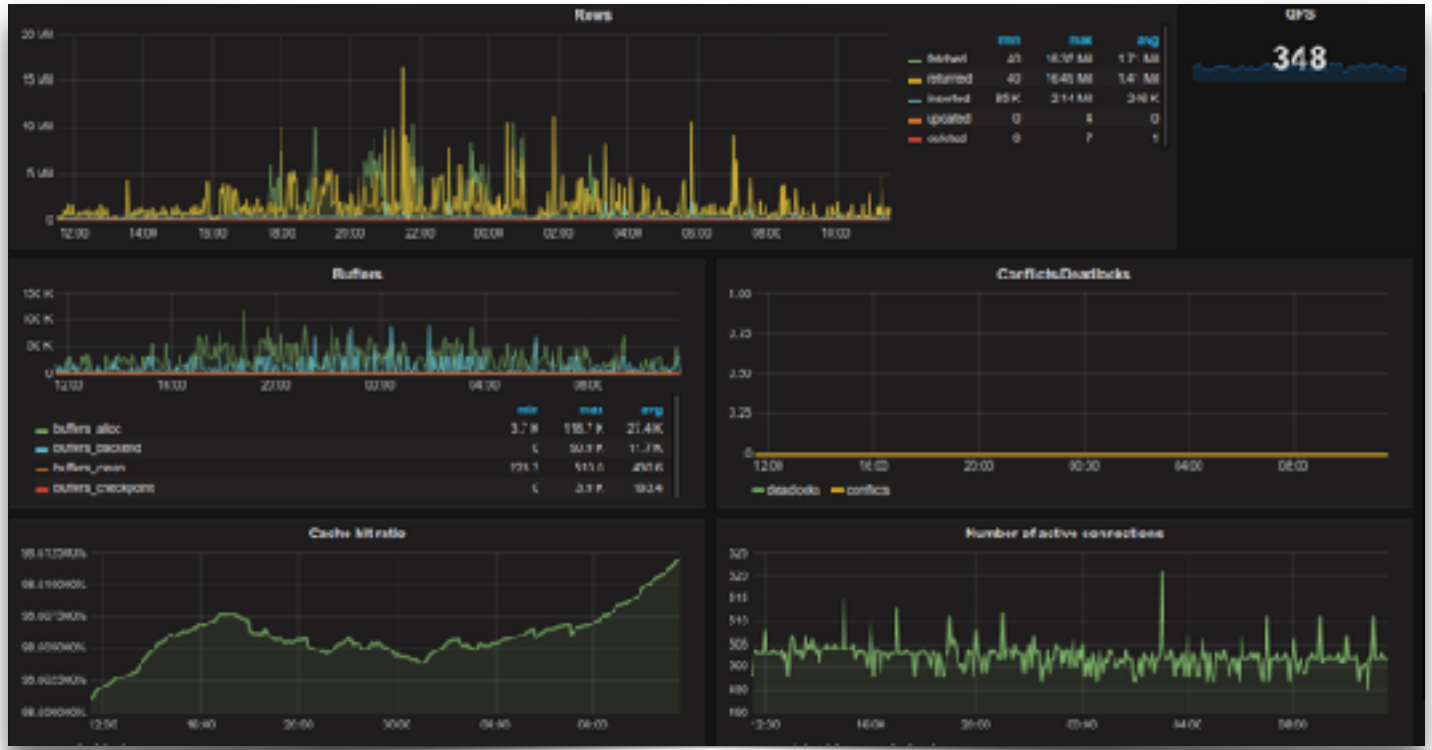


```
[root@server prometheus]# cat mysql.rules
# Alert: The replication lag is non-zero and it predicted to not recover within
#        2 minutes.  This allows for a small amount of replication lag.
# NOTE: This alert depends on the recording rule at the top of the file.
ALERT MySQLReplicationLag
  IF
      (mysql_heartbeat_lag_seconds > 30)
    AND on (instance)
      (predict_linear(mysql_heartbeat_lag_seconds[5m], 60*2) > 0)
  FOR 1m
  LABELS {
    severity = "critical"
  }
  ANNOTATIONS {
    summary = "MySQL slave replication is lagging",
    description = "The mysql slave replication has fallen behind and is not recovering",
  }

###
# Performance Alerts

# Alert: InnoDB log writes are stalling.
ALERT MySQLInnoDBLogWaits
  IF rate(mysql_global_status_innodb_log_waits[15m]) > 10
  LABELS {
    severity = "warning"
  }
  ANNOTATIONS {
    summary = "MySQL innodb log writes stalling",
    description = "The innodb logs are waiting for disk at a rate of {{$value}} / second",
  }
```

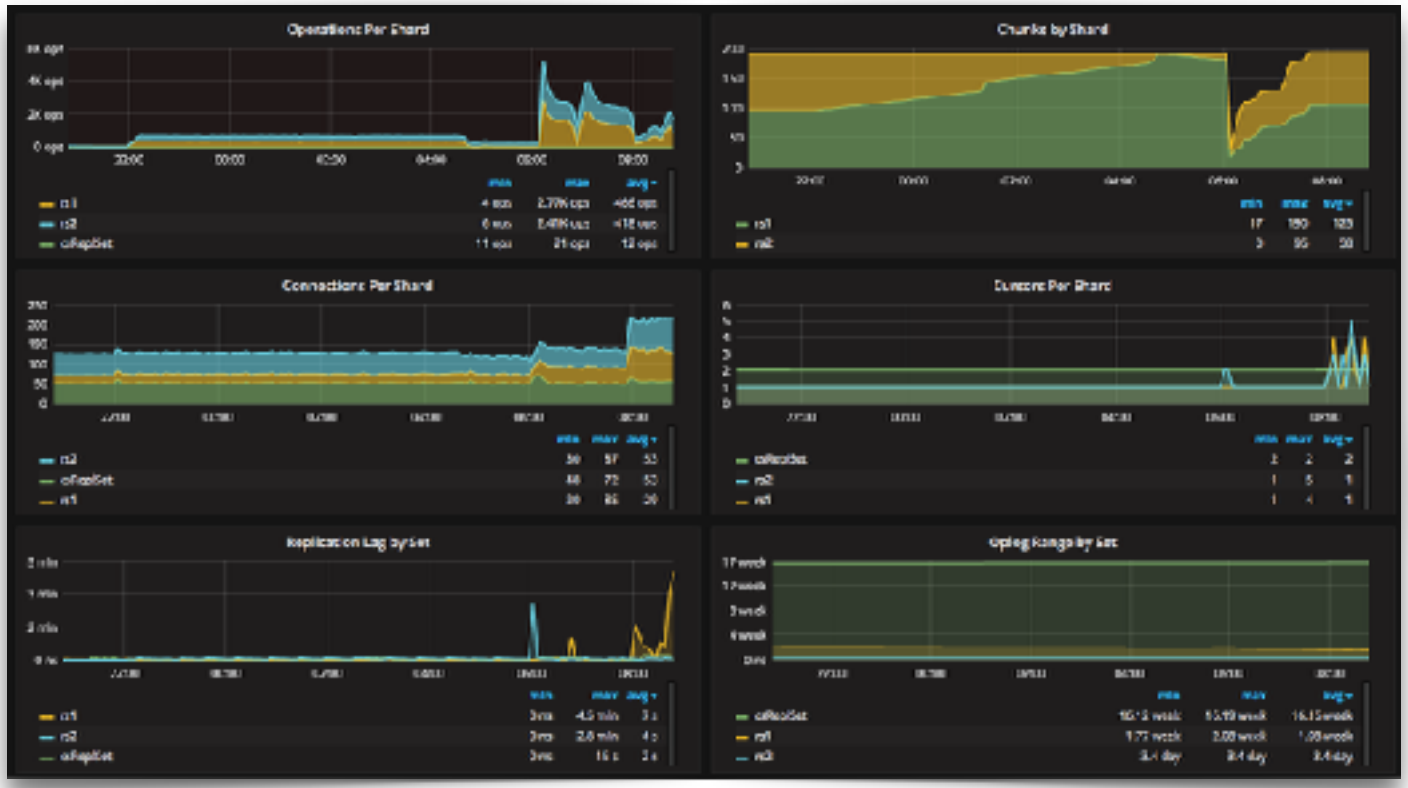


https://github.com/prometheus/mysqld_exporter

https://github.com/oliver006/redis_exporter



https://github.com/wrouesnel/postgres_exporter



https://github.com/percona/mongodb_exporter

- 高性能时序数据库
- 灵活的查询语言
- 活跃的社区支持
- 避免告警风暴
- 部署配置简单

Thanks & QA

沃趣 WOQUTECH

Let Data Drive!

微信二维码

微博二维码

E-mail：ge.jin@woqutech.com