



Prometheus Introduction

Julius Volz

Co-founder,
Prometheus

@juliusvolz

What is Prometheus

Metrics-based monitoring & alerting stack.

- Instrumentation
- Metrics collection and storage
- Querying, alerting, dashboarding
- For all levels of the stack!

Made for dynamic cloud environments.

What is it not?

We don't do:

- Logging or tracing
- Automatic anomaly detection
- Scalable or durable storage

History

- Started 2012 at SoundCloud
- Motivation: Lack of suitable OSS tools
- Fully publicised in 2015
- Part of CNCF since 2016

Architecture

Architecture



Targets

web app

API
server

Architecture


Targets

web app	 clientlib
API server	 clientlib

Architecture

Targets

web app	 clientlib
---------	--

API server	 clientlib
---------------	--

Linux VM


mysqld

cgroups


Architecture


Targets

web app	 clientlib
---------	--

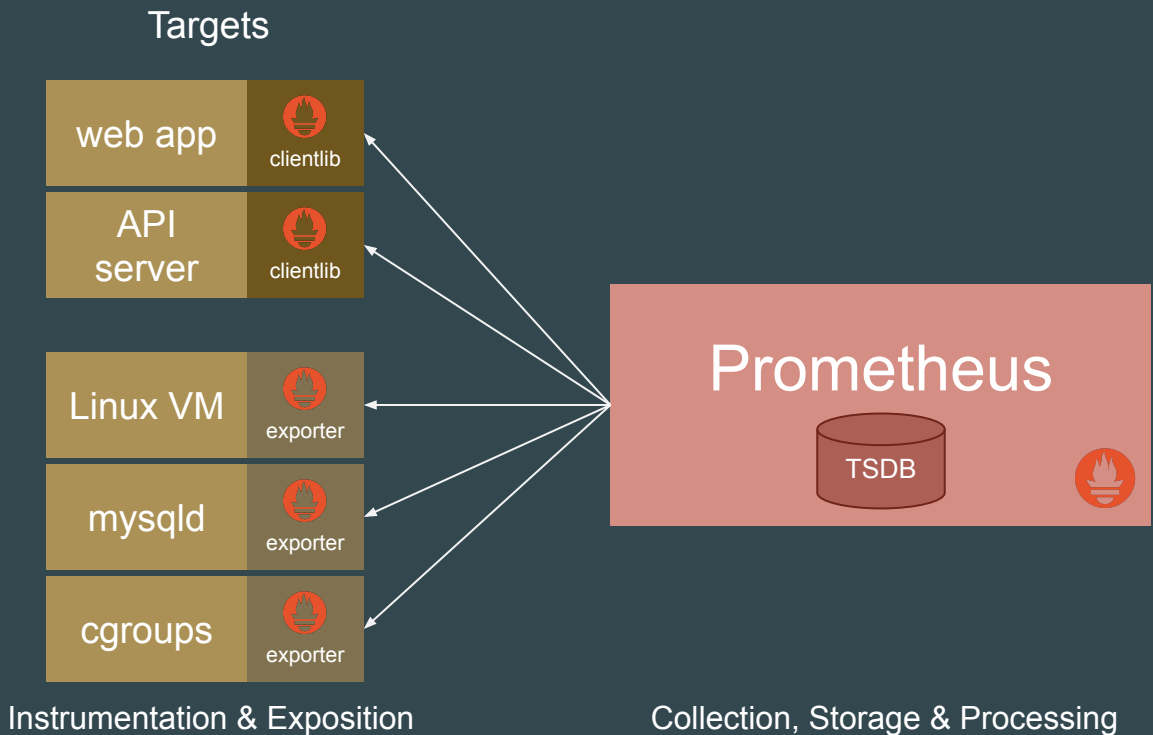
API server	 clientlib
------------	--

Linux VM	 exporter
----------	---

mysqld	 exporter
--------	---

cgroups	 exporter
---------	---

Architecture



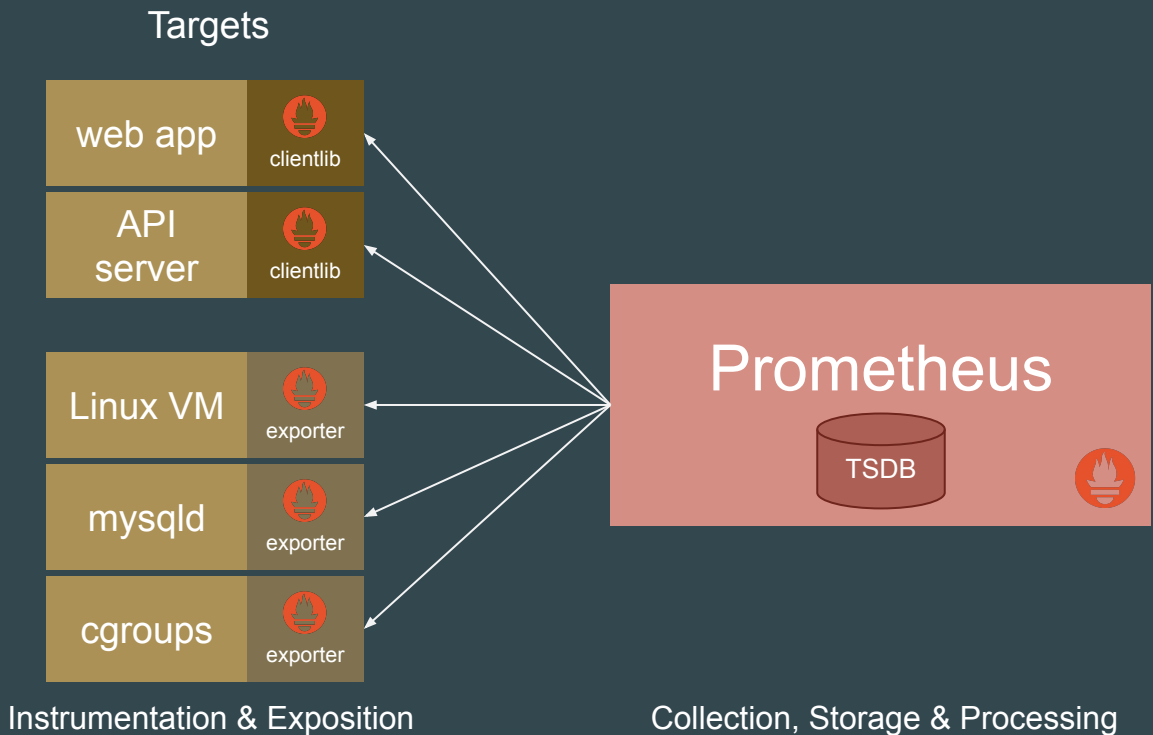
Interlude: Exposition Format

```
# HELP http_requests_total The total number of HTTP requests.
# TYPE http_requests_total counter
http_requests_total{method="post",code="200"} 1027
http_requests_total{method="post",code="400"} 3
```

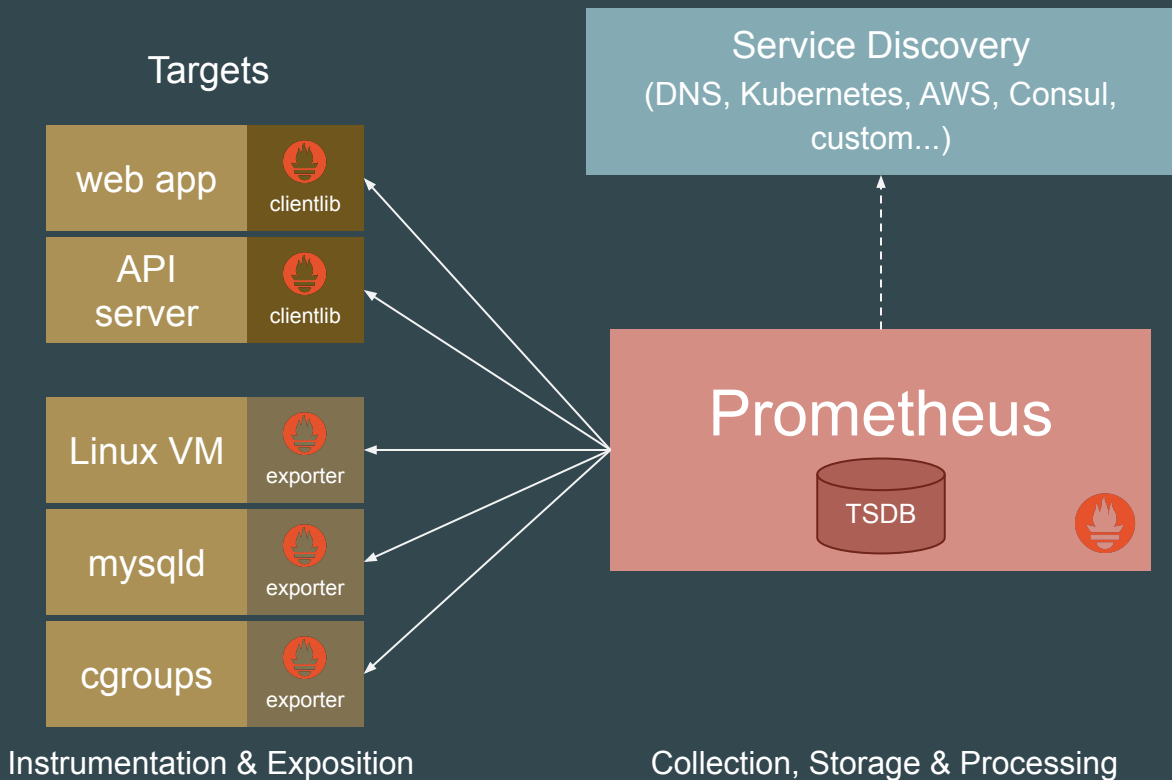
```
# HELP http_request_duration_seconds A histogram of the request duration.
# TYPE http_request_duration_seconds histogram
http_request_duration_seconds_bucket{le="0.05"} 24054
http_request_duration_seconds_bucket{le="0.1"} 33444
http_request_duration_seconds_bucket{le="0.2"} 100392
http_request_duration_seconds_bucket{le="0.5"} 129389
http_request_duration_seconds_bucket{le="1"} 133988
http_request_duration_seconds_bucket{le="+Inf"} 144320
http_request_duration_seconds_sum53423
http_request_duration_seconds_count144320
```

```
# HELP rpc_duration_seconds A summary of the RPC duration in seconds.
# TYPE rpc_duration_seconds summary
rpc_duration_seconds{quantile="0.01"} 3102
rpc_duration_seconds{quantile="0.05"} 3272
rpc_duration_seconds{quantile="0.5"} 4773
rpc_duration_seconds{quantile="0.9"} 9001
rpc_duration_seconds{quantile="0.99"} 76656
rpc_duration_seconds_sum1.7560473e+07
rpc_duration_seconds_count2693
```

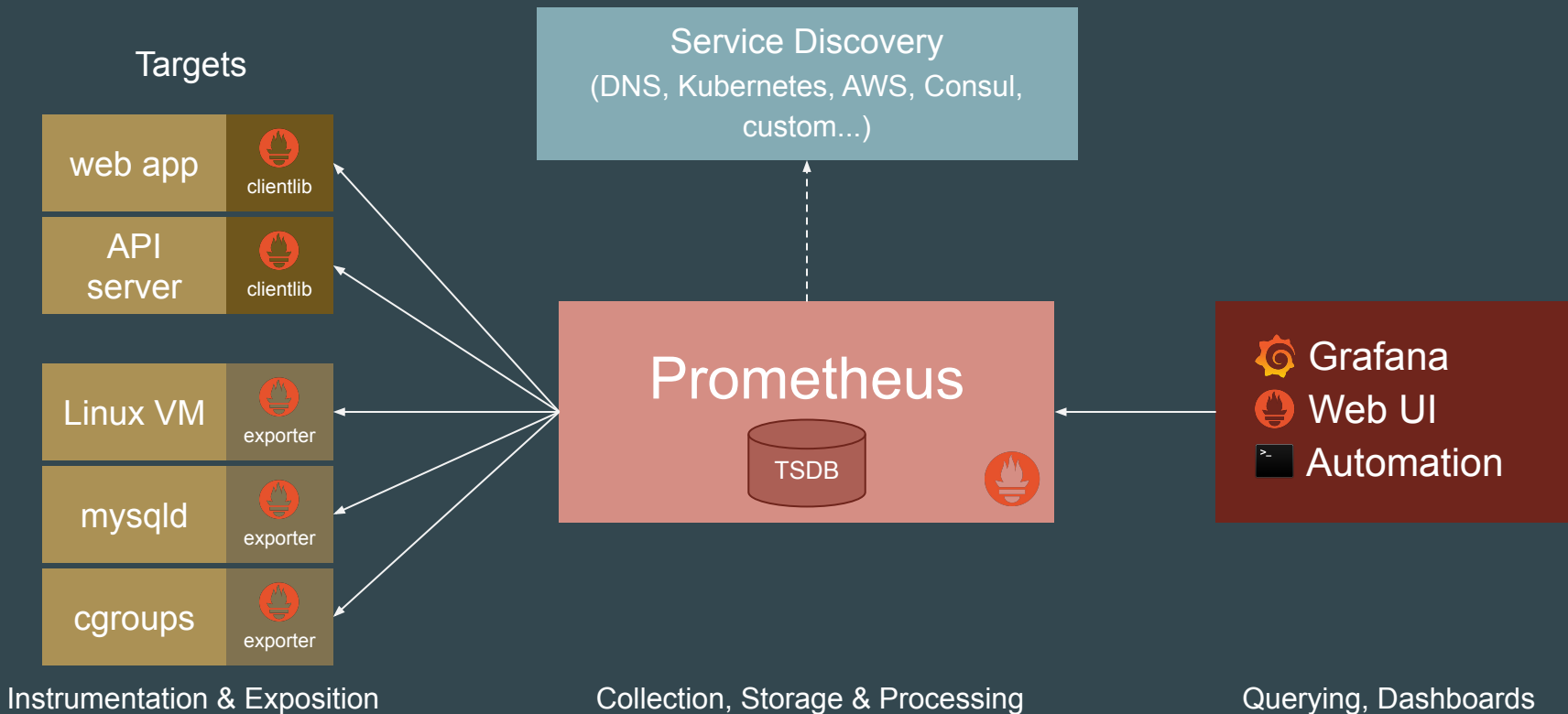
Architecture



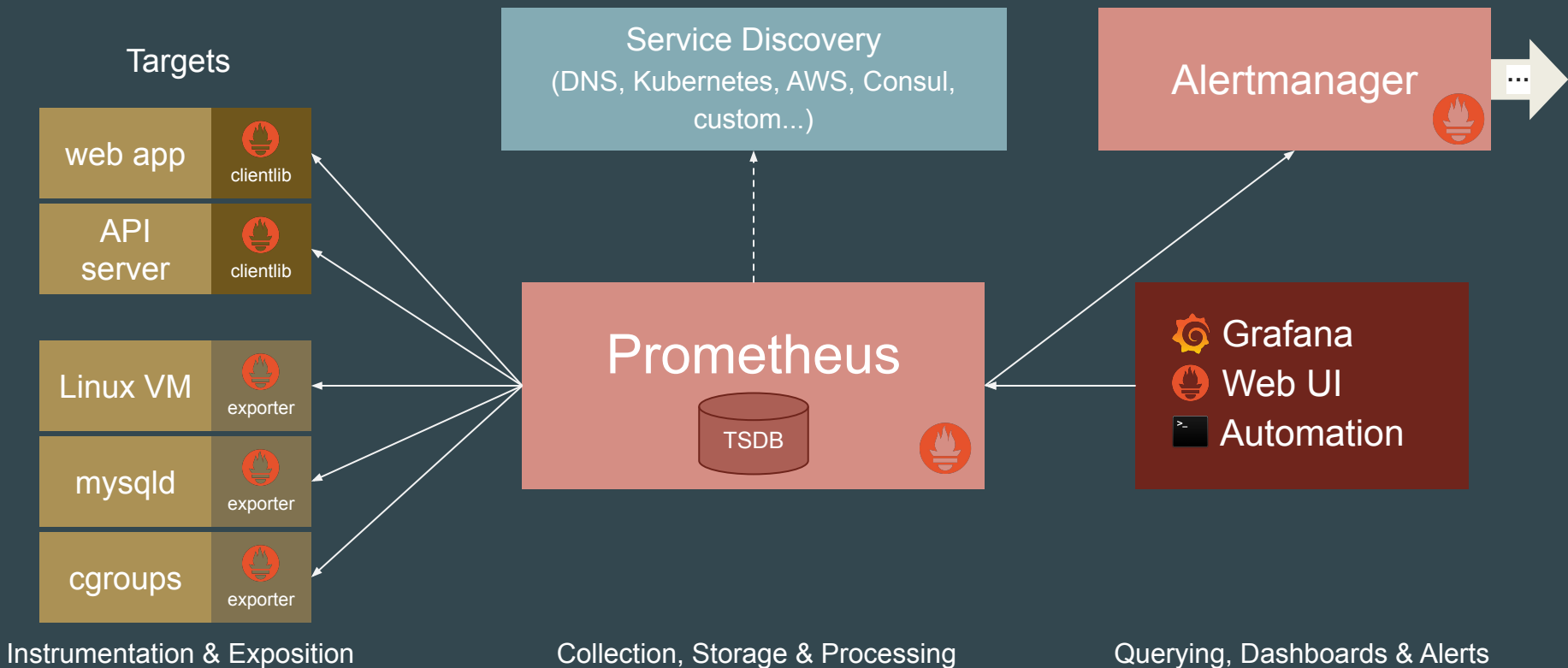
Architecture



Architecture



Architecture



Selling Points

- Dimensional data model
- Powerful query language
- Simple & efficient server
- Service discovery integration

Data Model

What is a time series?

`<identifier>` → [(t₀, v₀), (t₁, v₁), ...]

Data Model

What is a time series?

`<identifier>` → `[(t0, v0), (t1, v1), ...]`



What is this?



int64



float64

Data Model

What identifies a time series?

```
http_requests_total{job="nginx",instance="1.2.3.4:80",path="/home",status="200"}
```

Data Model

What identifies a time series?

```
http_requests_total{job="nginx",instance="1.2.3.4:80",path="/home",status="200"}
```

↑
metric name

↑
labels

Data Model

What identifies a time series?

```
http_requests_total{job="nginx",instance="1.2.3.4:80",path="/home",status="200"}
```

↑
metric name

↑
labels

- Flexible
- No hierarchy
- Explicit dimensions

Querying

PromQL

- New query language
- Great for time series computations
- Not SQL-style

Querying

All partitions in my entire infrastructure with more than 100GB capacity that are not mounted on root?

```
node_filesystem_bytes_total{mountpoint!=" /"} / 1e9 > 100
```

```
{device="sda1", mountpoint="/home", instance="10.0.0.1"}      118.8
{device="sda1", mountpoint="/home", instance="10.0.0.2"}      118.8
{device="sdb1", mountpoint="/data", instance="10.0.0.2"}       451.2
{device="xdvc", mountpoint="/mnt", instance="10.0.0.3"}       320.0
```

Querying

What's the ratio of request errors across all service instances?

```
sum(rate(http_requests_total{status="500"}[5m]))  
/ sum(rate(http_requests_total[5m]))
```

```
{}
```

```
0.029
```


Querying

What's the ratio of request errors across all service instances?

```
sum by(path) (rate(http_requests_total{status="500"}[5m]))  
/ sum by(path) (rate(http_requests_total[5m]))
```

```
{path="/status"}           0.0039  
{path="/" }               0.0011  
{path="/api/v1/topics/:topic"} 0.087  
{path="/api/v1/topics"}    0.0342
```

Querying

99th percentile request latency across all instances?

```
histogram_quantile(0.99,  
  sum without(instance) (rate(request_latency_seconds_bucket[5m]))  
)
```

```
{path="/status", method="GET"}           0.012  
{path="/", method="GET"}                 0.43  
{path="/api/v1/topics/:topic", method="POST"} 1.31  
{path="/api/v1/topics", method="GET"}      0.192
```

Expression browser

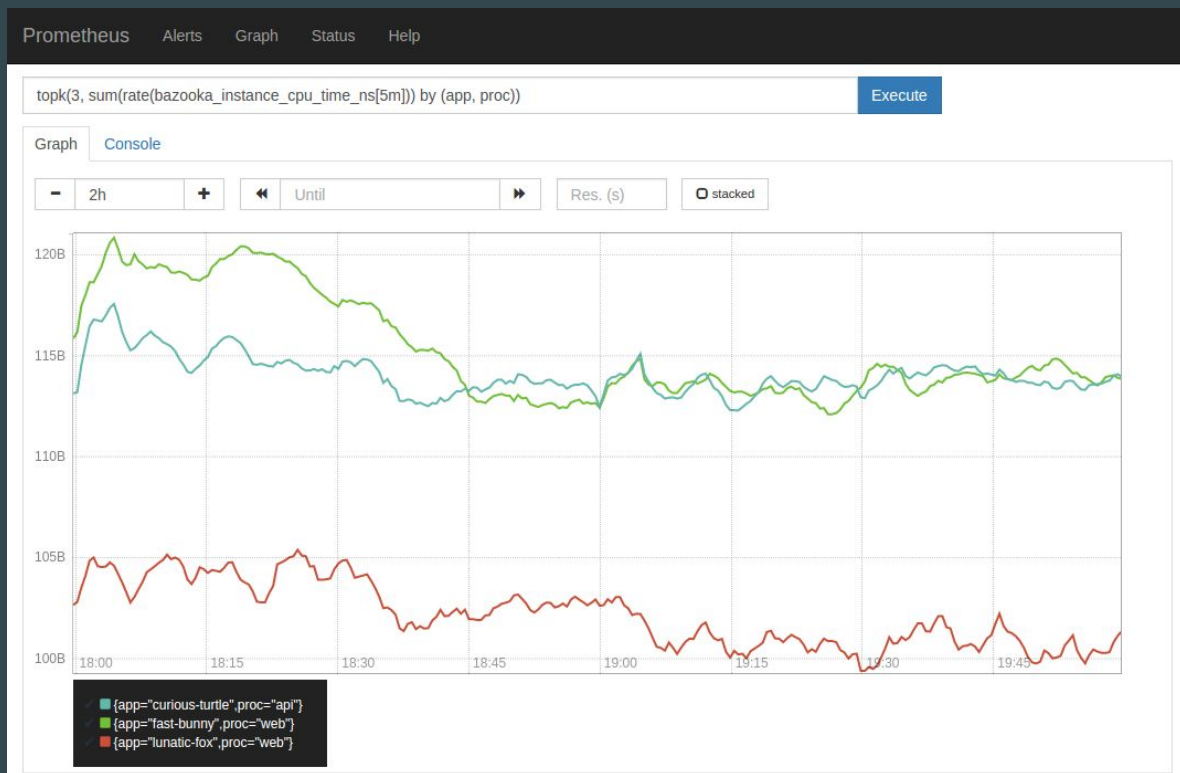
[Prometheus](#)[Alerts](#)[Graph](#)[Status](#)[Help](#)

```
sort_desc(sum(bazooka_instance_memory_limit_bytes - bazooka_instance_memory_usage_bytes) by (app, proc)) / 1024 / 1024 / 1024
```

[Execute](#)[Graph](#)[Console](#)

Element	Value
{app="harsh-dagger",proc="api"}	132.720802
{app="quality-locomotive",proc="web"}	89.547081
{app="husky-long-oyster",proc="web"}	68.982738
{app="vital-albatross",proc="api"}	48.033772
{app="autopsy-gutsy",proc="widget"}	47.410583
{app="western-python",proc="cruncher"}	40.126926
{app="harsh-dagger",proc="api"}	28.527714
{app="outstanding-dagger",proc="api"}	26.119423
{app="gruesome-waterbird",proc="web"}	17.666714
{app="gutsy-square",proc="public"}	15.296242
{app="harsh-dagger",proc="web"}	14.738327
{app="northern-electron",proc="api"}	13.349815

Built-in graphing



Dashboarding



Alerting

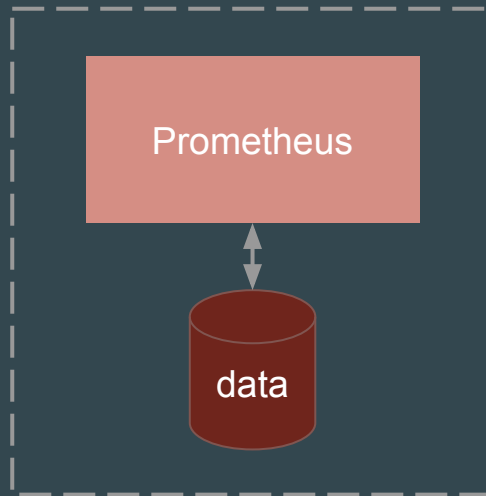
generate an alert for each path with an error rate of >5%



```
alert: Many500Errors
expr: |
  (
    sum by(path) (rate(http_requests_total{status="500"}[5m]))
    /
    sum by(path) (rate(http_requests_total[5m]))
  ) * 100 > 5
for: 5m
labels:
  severity: "critical"
annotations:
  summary: "Many 500 errors for path {{$labels.path}} ({{$value}}%)"
```

Operational Simplicity

- Local storage, no clustering
- HA by running two
- Go: static binary



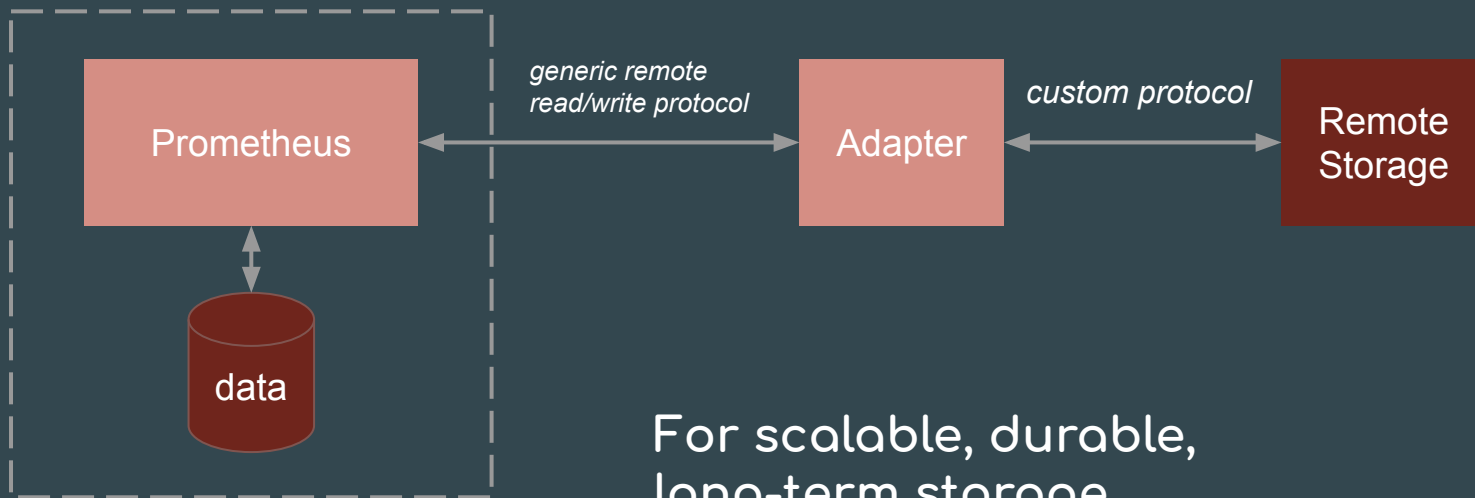
Efficiency

Local storage is scalable enough for many orgs:

- 1 million+ samples/s
- Millions of series
- 1-2 bytes per sample

Good for keeping a few weeks or months of data.

Decoupled Remote Storage



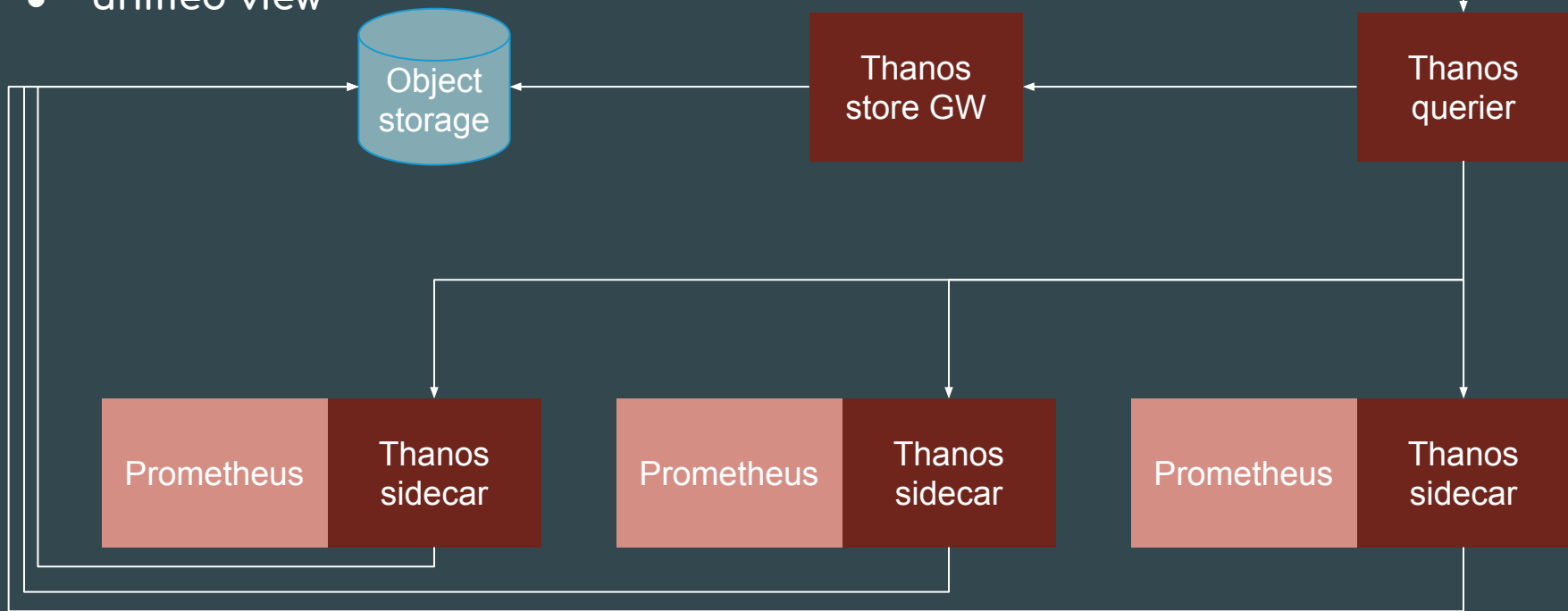
For scalable, durable,
long-term storage.

E.g.: Cortex, InfluxDB

Or: thanos.io

- long-term storage
- durability
- unified view

Thanos



Dynamic Environments

...pose new challenges:

- Dynamic VMs
- Cluster schedulers
- Microservices

→ many services, dynamic hosts, and ports

How to make sense of this all?

Service Discovery

Use service discovery to:

- ...know what *should* be there
- ...decide where to pull from
- ...add dimensional metadata to series

Service Discovery

Prometheus has built-in support for:

- VM providers (AWS, Azure, Google, ...)
- Cluster managers (Kubernetes, Marathon, ...)
- Generic mechanisms (DNS, Consul, Zookeeper, custom, ...)

Conclusion

Prometheus helps you make sense of complex dynamic environments via its:

- Dimensional data model
- Powerful query language
- Simplicity + efficiency
- Service discovery integration

Thanks!