Monitoring my application like a boss


fagossa


fabgutierr


fabian.gutierrez

github.com/fagossa/play-prometheus

Agenda

- Why are we here?
- Monitoring
- Logging
- Push model andThen demo
- Pull model andThen demo
- Questions

Why are we here?

because we need to know when the house is on fire


THIS IS FINE.


THIS IS
NOT
FINE!!

Monitoring

Follow-up the state of an application

Can be achieved by any/all of these three things?

- Logging
- Tracing (this one get philosophical)
- Metrics

Logging vs Tracing

What is the difference between Tracing and Logging?

6

1

From the terminology point of view and in general, what is the difference between a 'tracing' and a 'logging' ?

Thanks!

java .net windows logging tracing

share edit flag

asked Dec 2 '14 at 8:21

pencilCake

16.8k ● 53 ● 164 ● 286

Event Logging vs. Tracing sums it up nicely. – Filburt Dec 2 '14 at 8:28

add a comment

start a bounty

Logging - Solved problem

- No more ssh + tail -f output.log
- **Strings** containing diverse information
  - (level, user, host, etc)
- **Slf4j** (logback, others)
- Individual records matter
- **Direct business value** (€)
- Non-ephemeral
- Logs can be **used as metrics**

Logging - examples

1

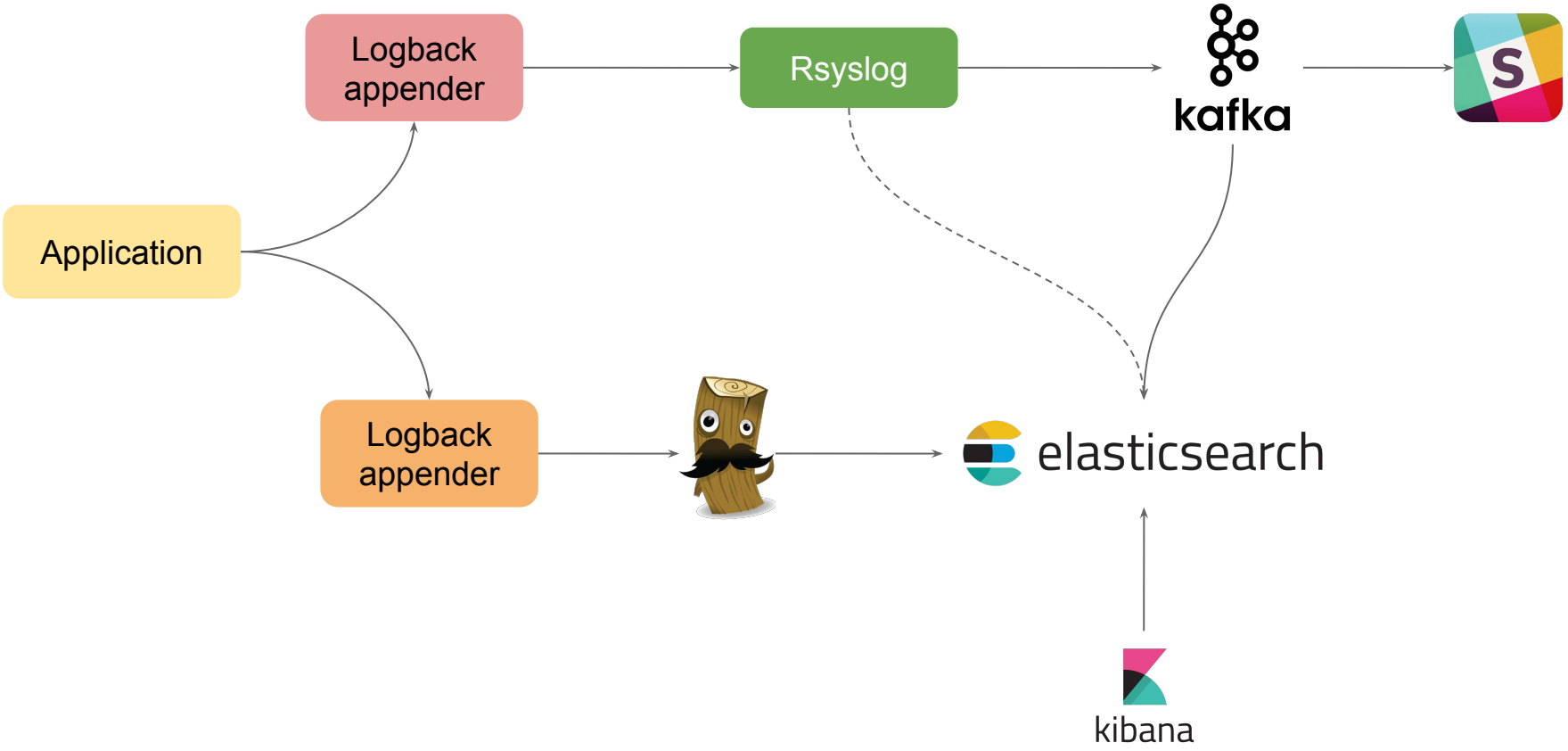
```
217.0.0.1 - jean [INFO] GET /products 200
```

2

```
host=217.0.0.1 user=jean method=GET url=/products time=200
```

3

```
{
 "host": "217.0.0.1",
 "user": "jean",
 "method": "GET",
 "url": "/products",
 "time": 200
}
```


Logback appender
Rsyslog
kafka
Application
Logback appender
elasticsearch
kibana

Metrics

- (hopefully) Multidimensional data
- **Direct tech value**
  - Response times
  - Complex flows
- Business value (?)
- **Ephemeral**
- Easier alerting
- Logs => metrics
  - https://github.com/google/**mtail**/wiki

- **Examples**

1

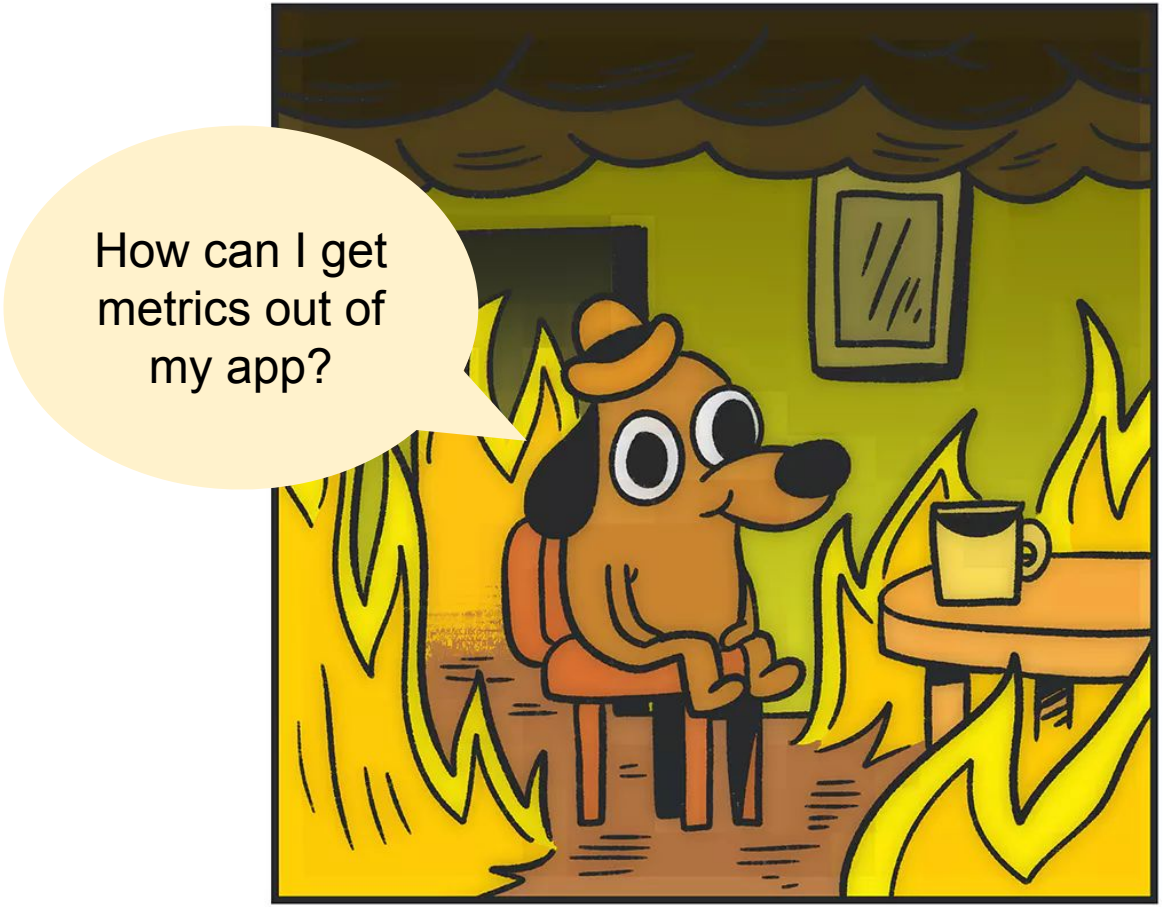
```
http_request_duration_seconds_count
{
    method="GET",
    path="/metrics",
    status="2xx"
} 4
```

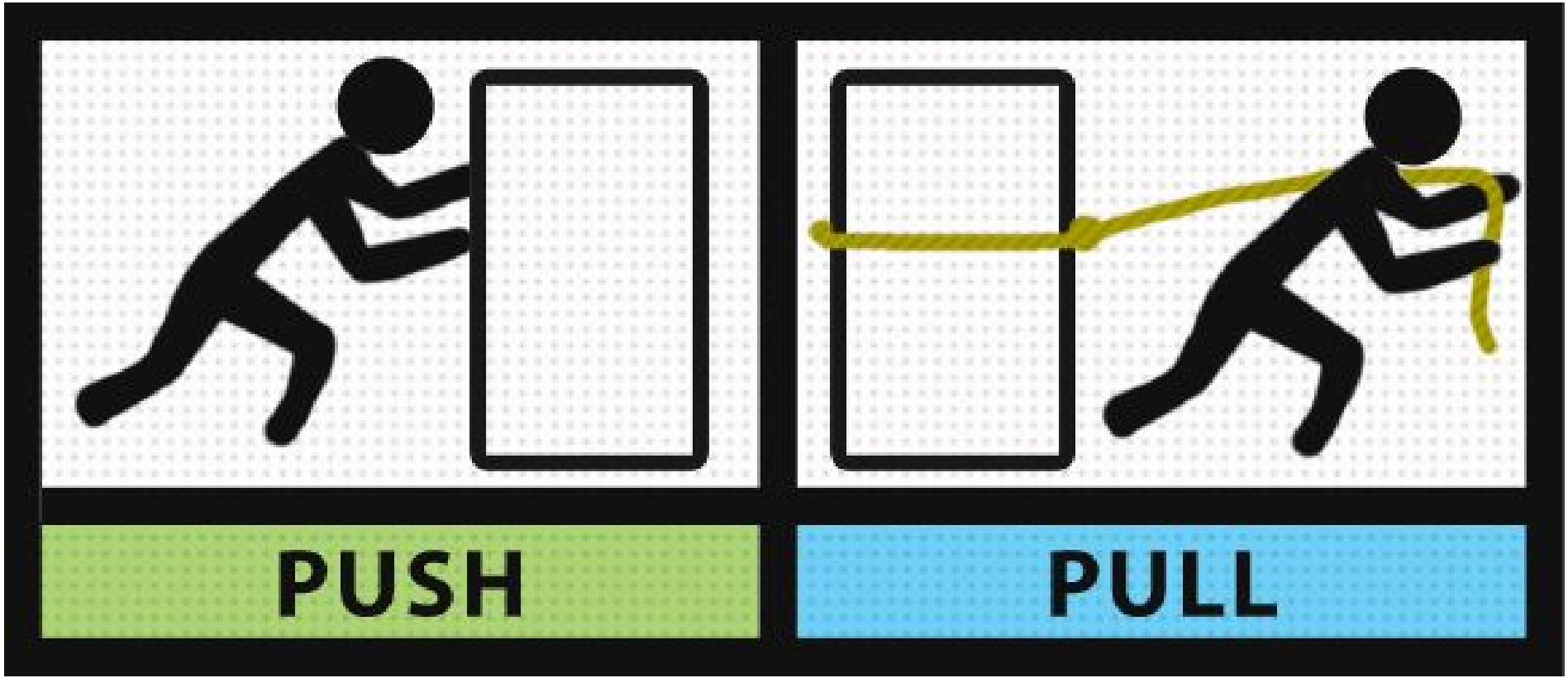
2

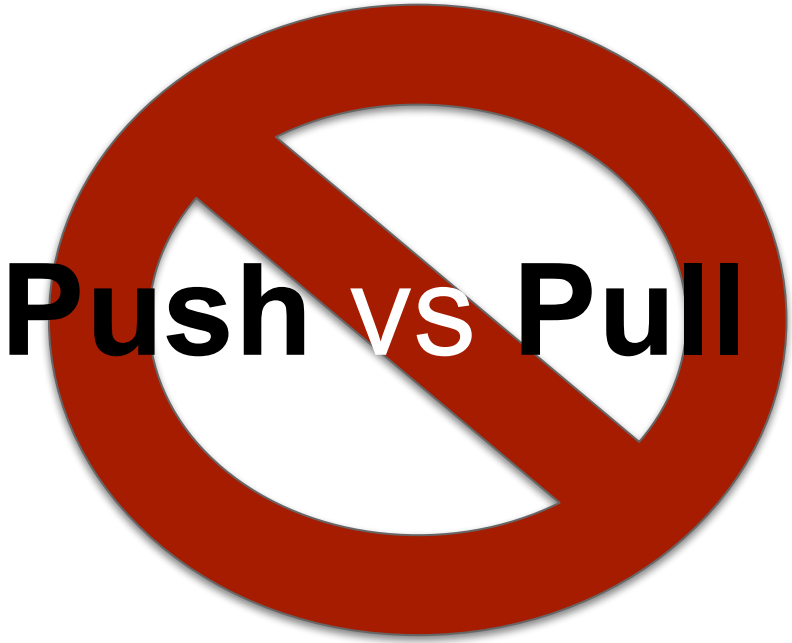
3

```
current_users    3.0
```

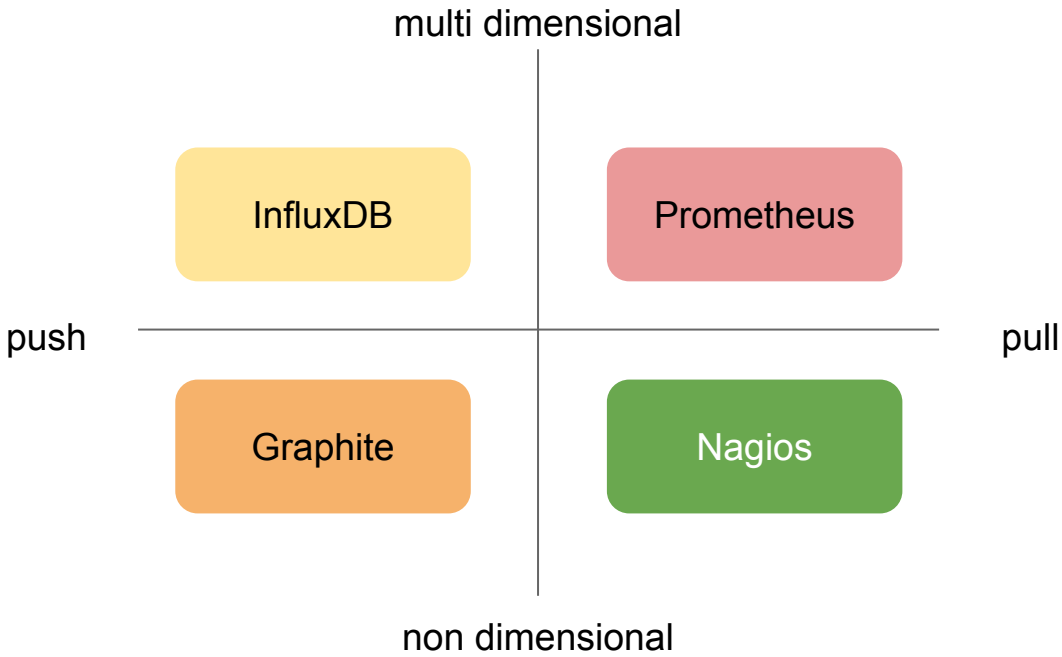
4


How can I get metrics out of my app?


PUSH
PULL


Push vs Pull

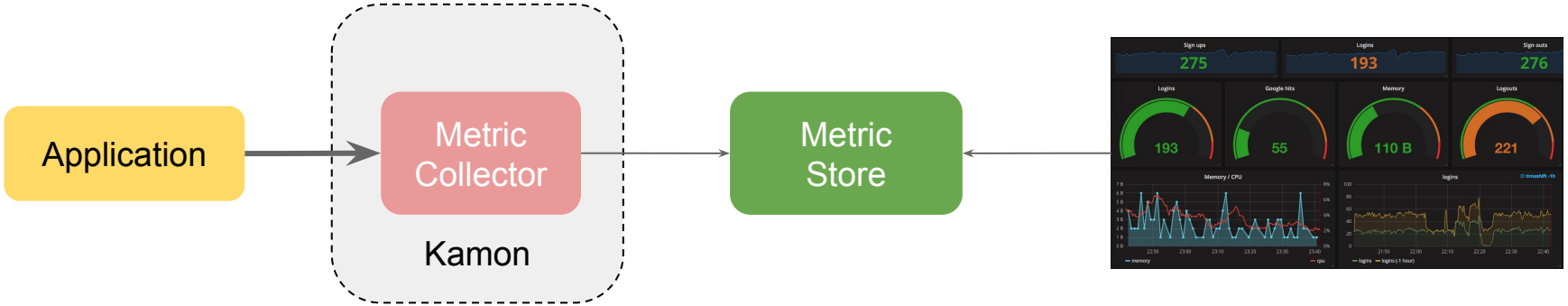
We are talking about this

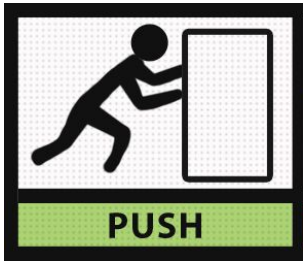


Push approach

Push those metrics my friend

Push approach



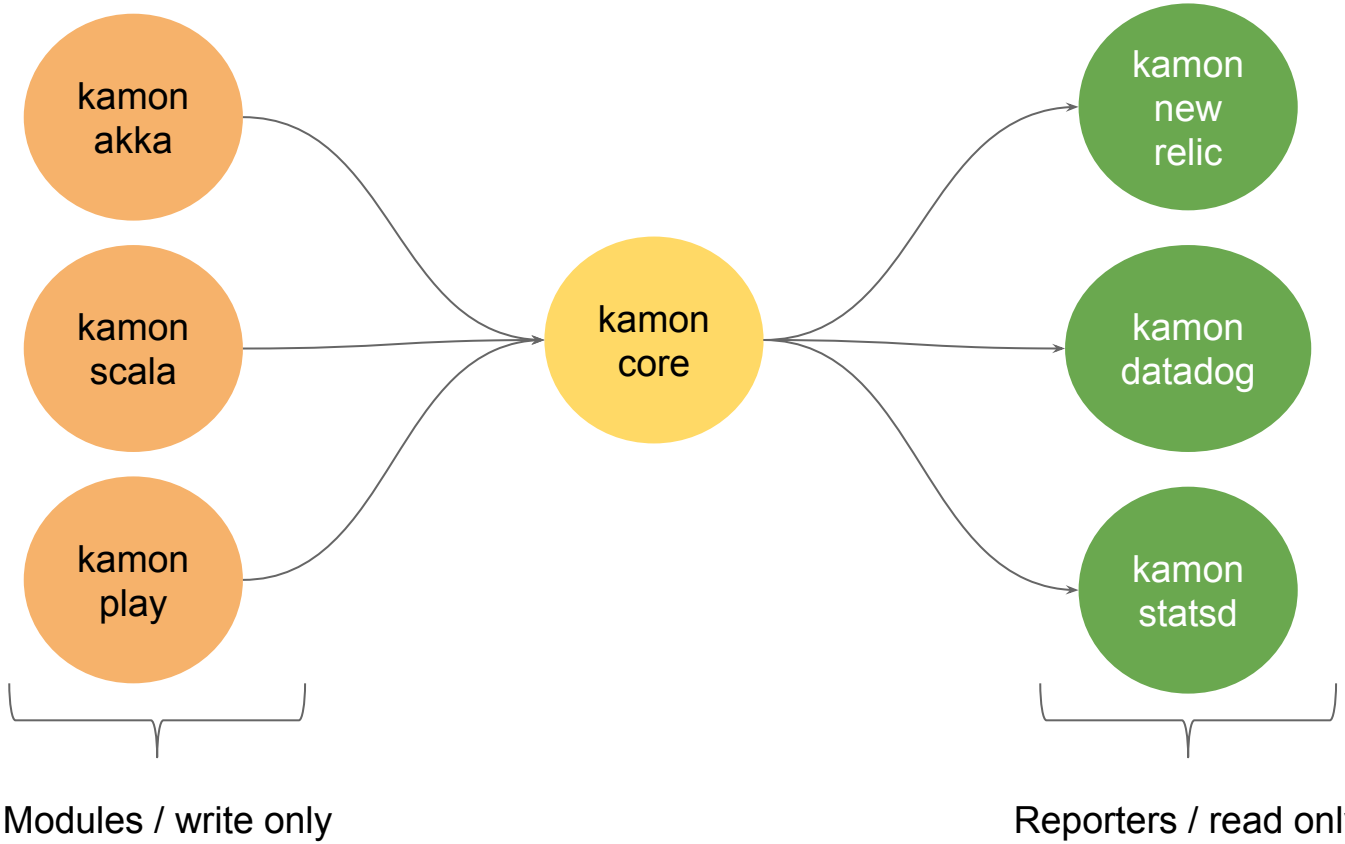


Kamon

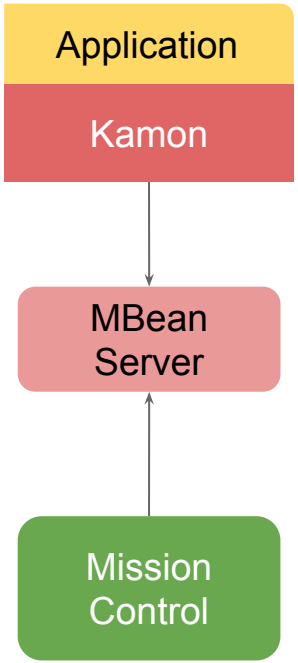
Monitoring tool for the JVM

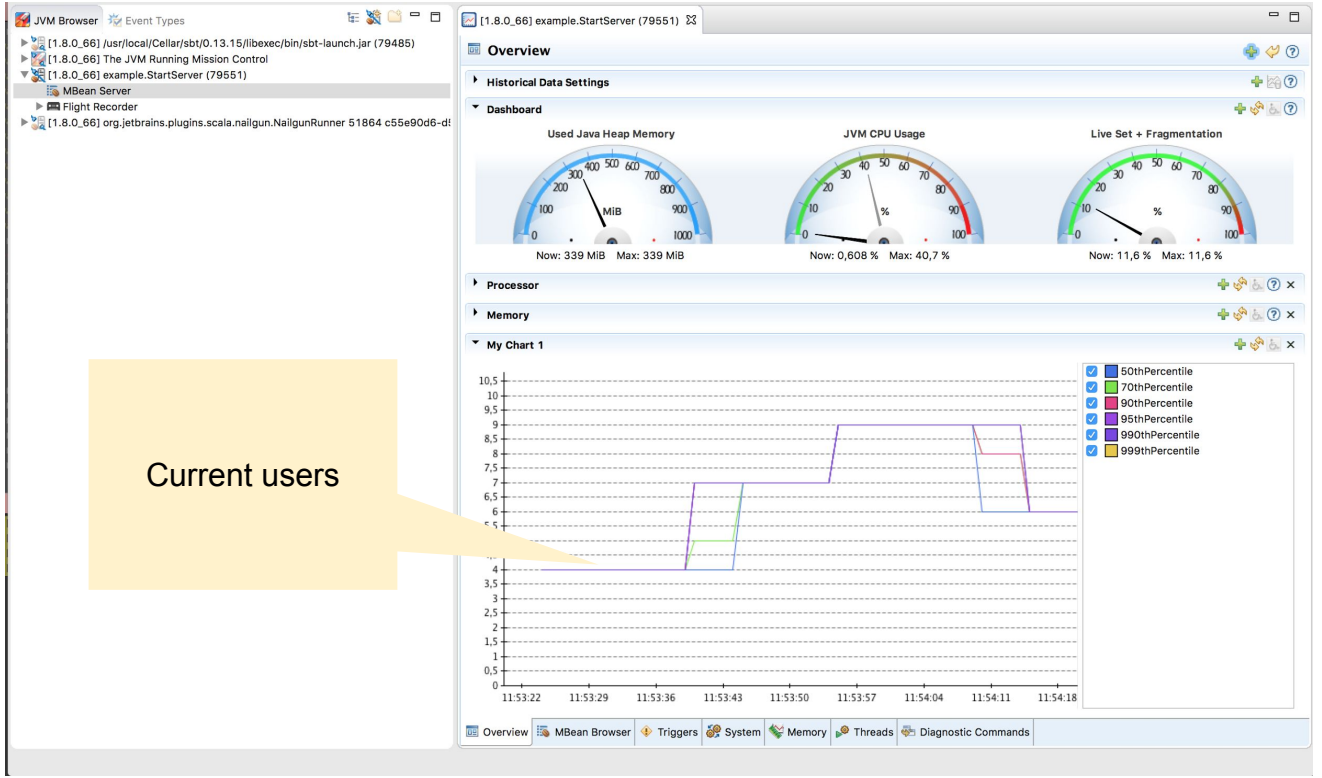
- Open source
- Metrics and tracing API
- **Instrumentation** for common libraries (akka, play, etc)
- **Collection** and **reporting** are **separate**
  - Instrument once, report anywhere

High Level view



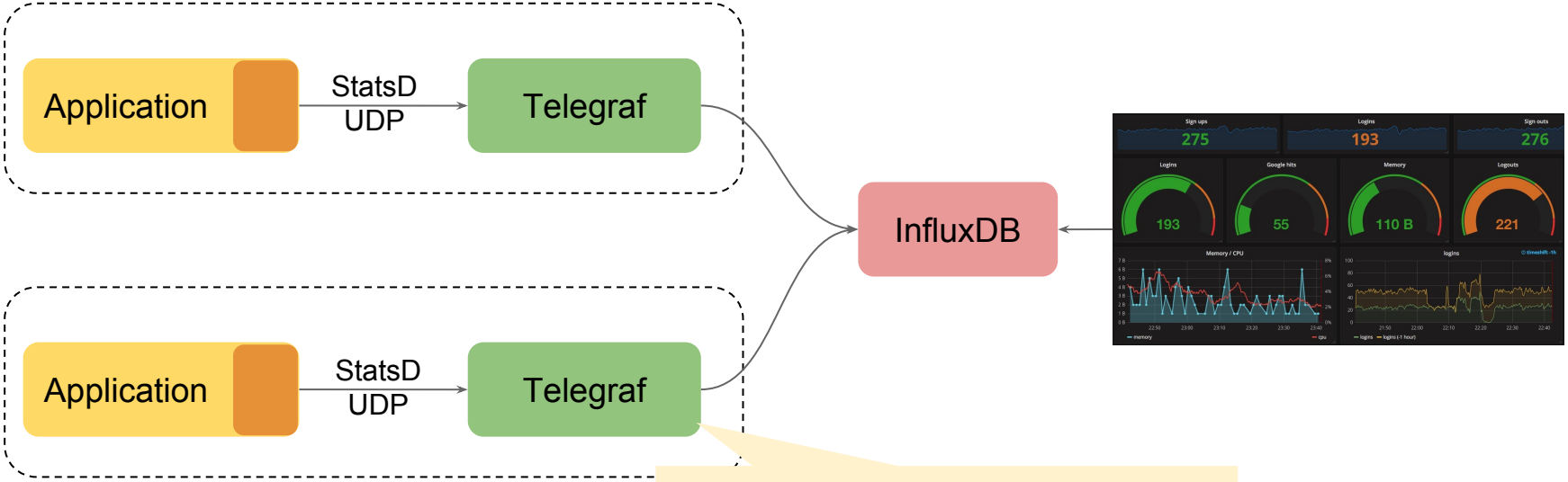
Kamon + JMX Reporter






DEMO GODS
PLEASE LET THIS DEMO WORK

Kamon + Telegraf + influxDB



- Agent that accepts StatsD protocol metrics
- Aggregates and parses metrics
- Periodically forwards the metrics to InfluxDB

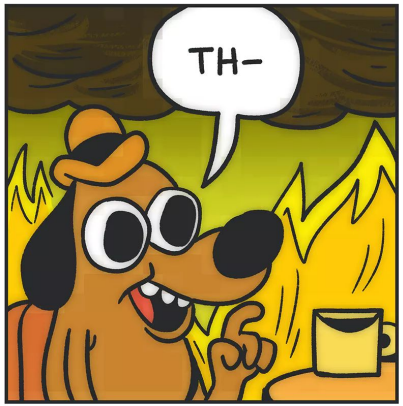
Kamon and Actors

For each actor you have access to **4 metrics**:

- Errors
- Mailbox-size
- Processing-time
- Time-in-mailbox

Kamon - The good

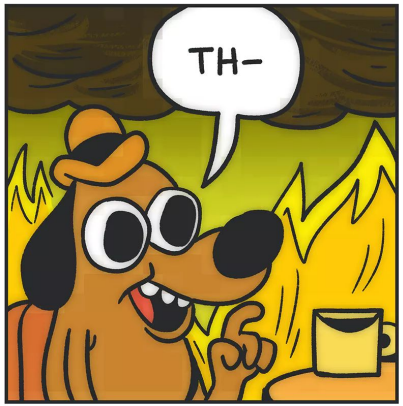
RECAP



- **Push** approach
- **Great integration** with the JVM
- Several modules (JMX, StatsD, etc)
- **Active project**: A new version (1.0.0) came out a couple of months ago

Kamon - The bad?

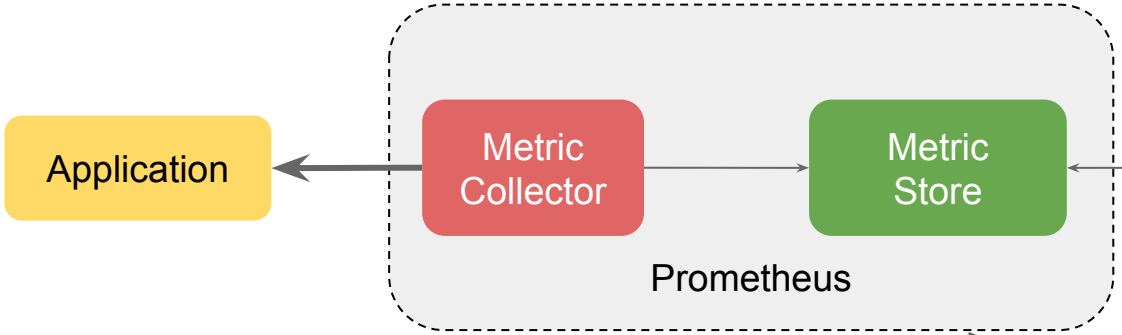
RECAP

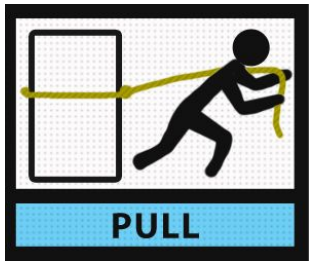


- **Bytecode** instrumentation (?)
- Working with modules is sometimes **confusing** (à la Spring)
- Potential **bytecode incompatibilities**
- Outside the JVM

Pull approach

Pull approach





- You can run your **monitoring on your laptop** when developing changes
- You can more easily **tell if a target is down**
- You can **manually go** to a target and **inspect its health with a web browser**

Prometheus

System monitoring tool with built-in timeseries DB

- **Integrates collecting** and **reporting**
- Metric API
- **Alerting already provided**
- Only **numeric timeseries** metrics

It is not

- Don’t do logging or tracing
- Not distributed storage (**only local**) by design!

/metrics

1. `http_request_duration_seconds_count { method="GET", path="/metrics", status="2xx"}` ----------> **2**

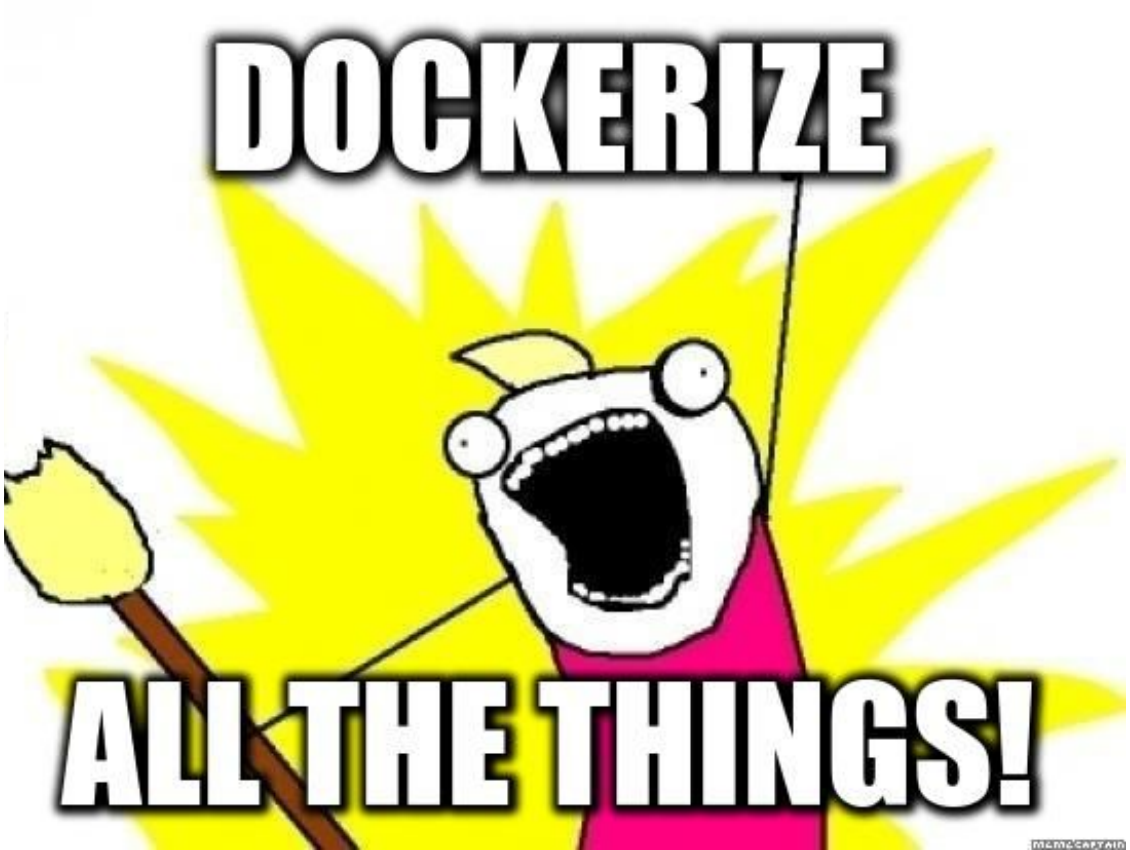
2. `http_request_duration_seconds_sum { method="GET", path="/metrics", status="2xx"}` ----------> **0.065599873**

3. `http_request_mismatch_total` ----------> **1.0**

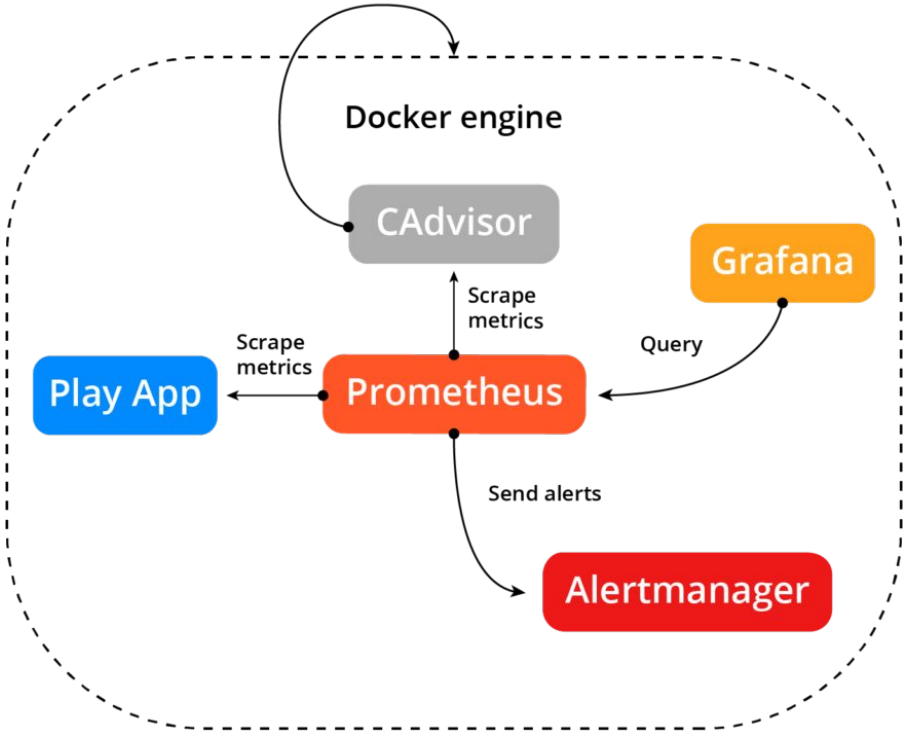
4. `play_current_users` ----------> **3.0**

5. `play_requests_total` ----------> **1.0**

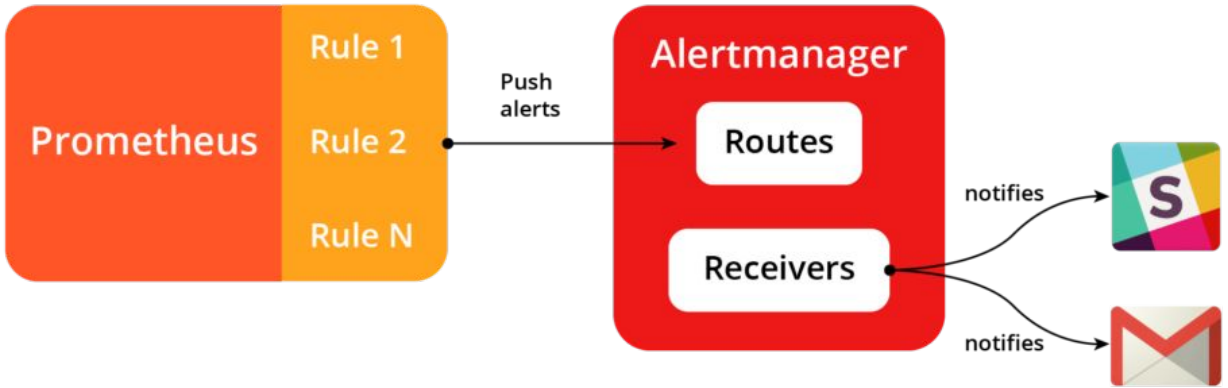
Prometheus + docker



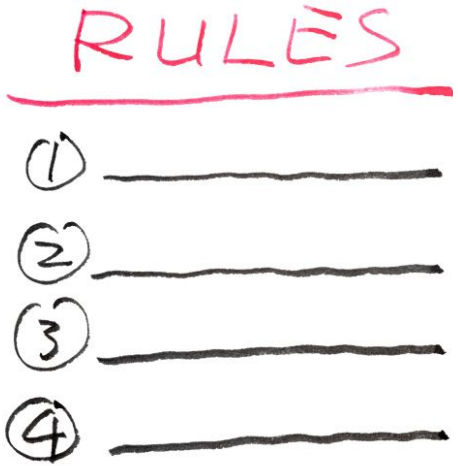
High level view



Alerting


Prometheus
Rule 1
Rule 2
Rule N
Push
alerts
Alertmanager
Routes
Receivers
notifies
notifies

Alerting



1

```
ALERT low_connected_users
  IF play_current_users < 2
  FOR 30s
  LABELS {
    severity = "warning"
  }
  ANNOTATIONS {
    summary = "Instance {{ $labels.instance }} under lower load",
    description =
          "{{ $labels.instance }} of job {{ $labels.job }} is under lower load.",
  }
```

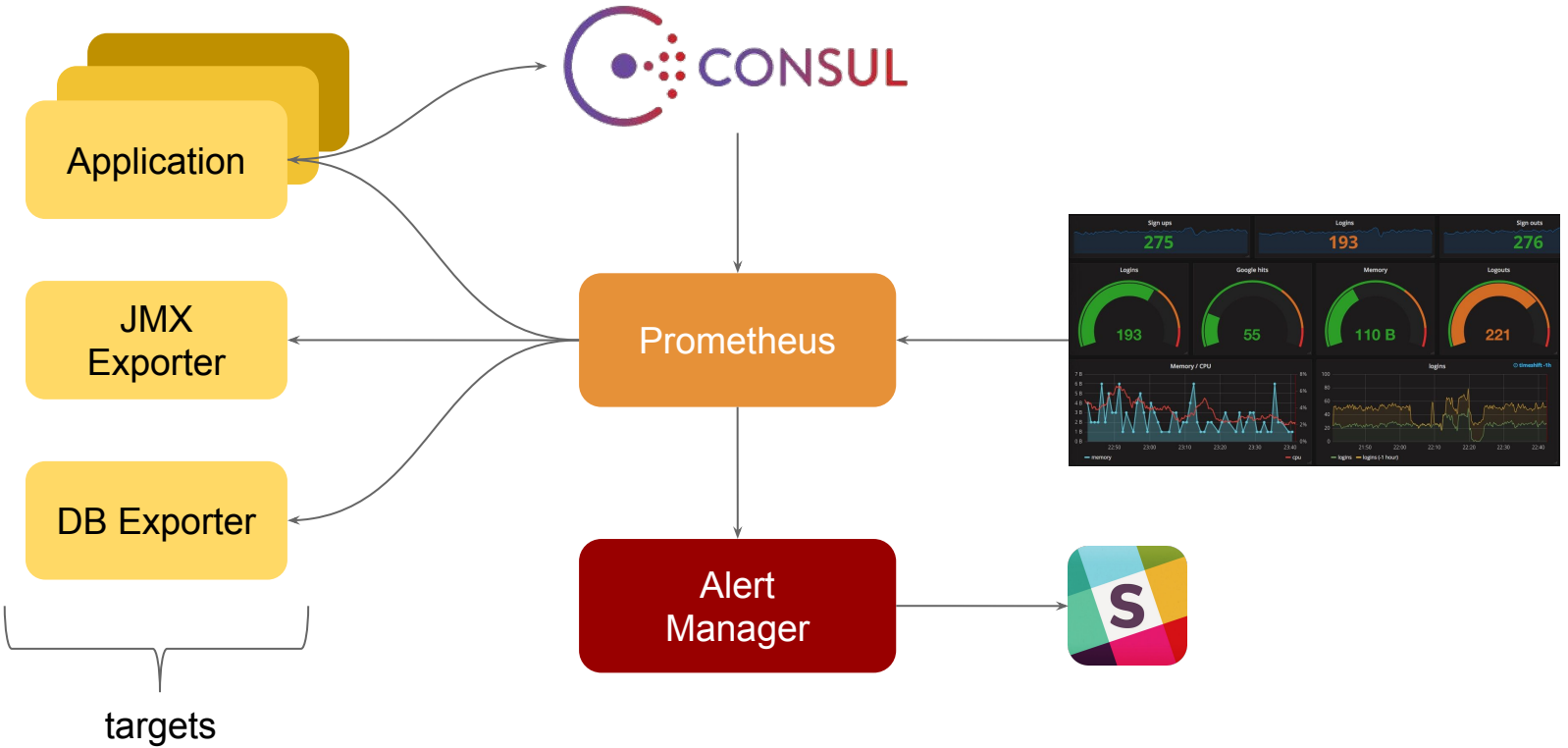
2

3

4

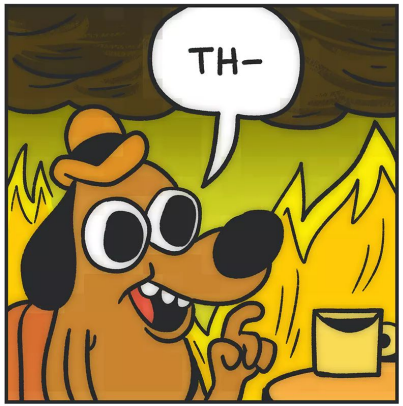

DEMO GODS
PLEASE LET THIS DEMO WORK

A more complex architecture



Prometheus - The good

RECAP

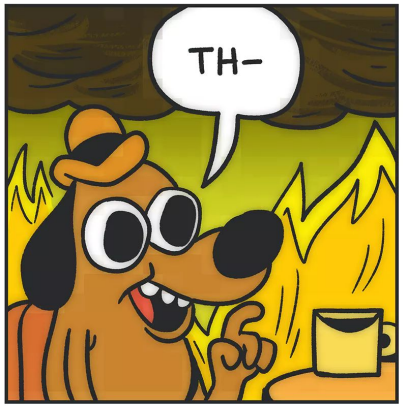


- **Pull** approach
- **Prepackaged** solution (collect + storage)
- Easy to start with
- Simple metric API
- **Active project** (version 2.0 just came out)
- Lots of **exporters**
- prometheus-akka seems nice

MySQL elasticsearch StatsD CONSUL kafka redis

Prometheus - The bad?

RECAP



- **Ephemeral** persistence (?)
- What to do after a **few weeks of logs**? (existing adaptors to influxDB)
- App **overhead**?
- Kamon-prometheus bridge :(

From zero to hero

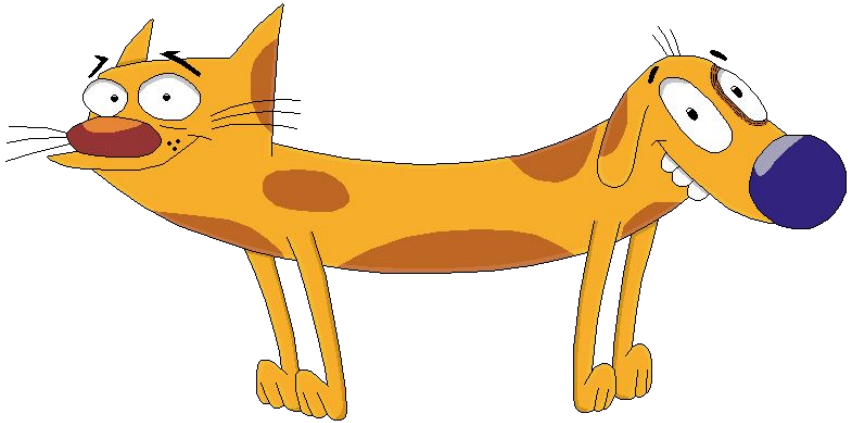

THIS IS FINE.

Mixing things up

https://github.com/lonelyplanet/prometheus-akka-http

https://github.com/Workday/prometheus-akka

https://github.com/MonsantoCo/kamon-prometheus



Conclusions

- Both approaches are **robust** enough
- Good **integrations** for both
- **Don’t guess … monitor**
- Does not matter which approach … **choose one**

Going further

- https://github.com/fagossa/play-prometheus
- http://blog.xebia.fr/2017/07/28/superviser-mon-application-play-avec-prometheus
- https://en.fabernovel.com/insights/tech-en/alerting-in-prometheus-or-how-i-can-sleep-well-at-night