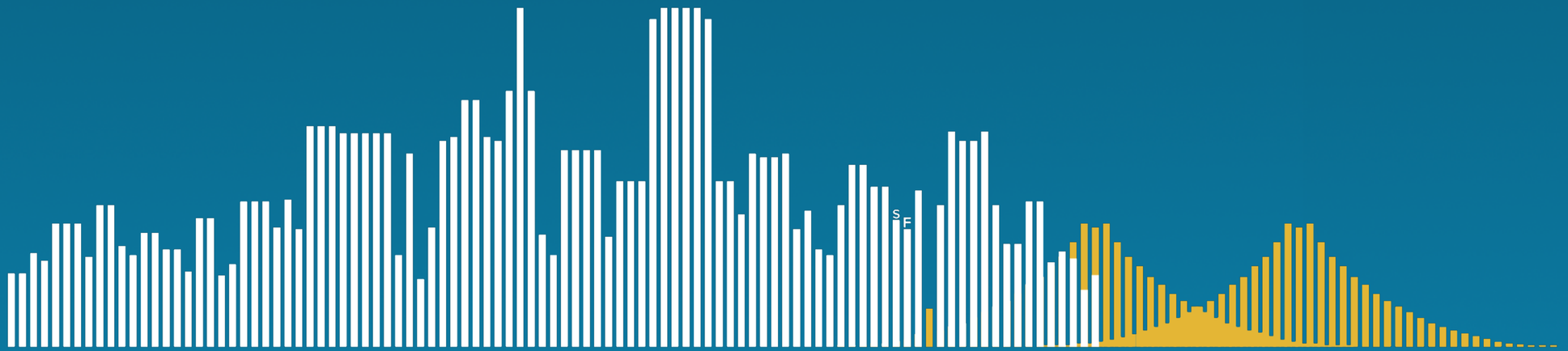
hi.

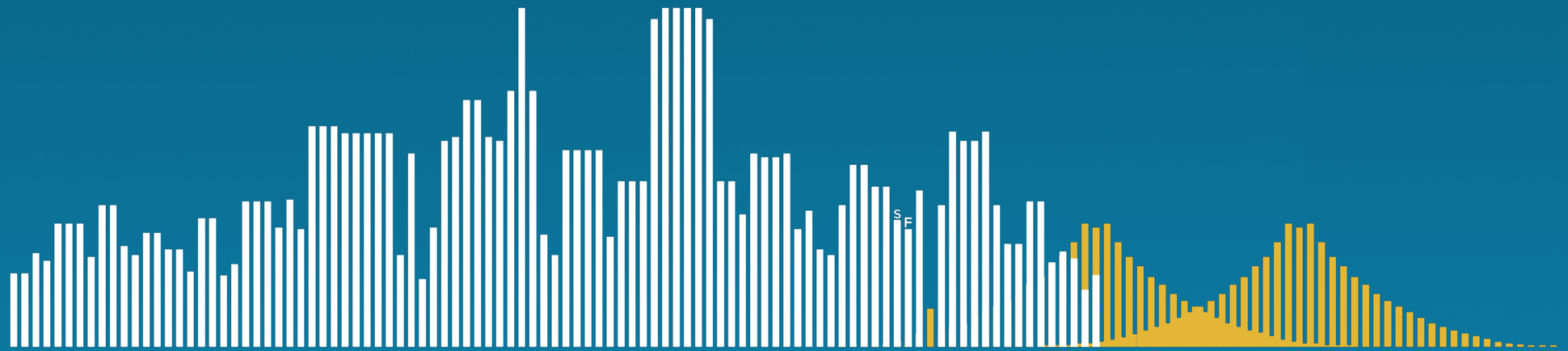


hi.

dave@librato.com

@davejosephsen

github: djosephsen

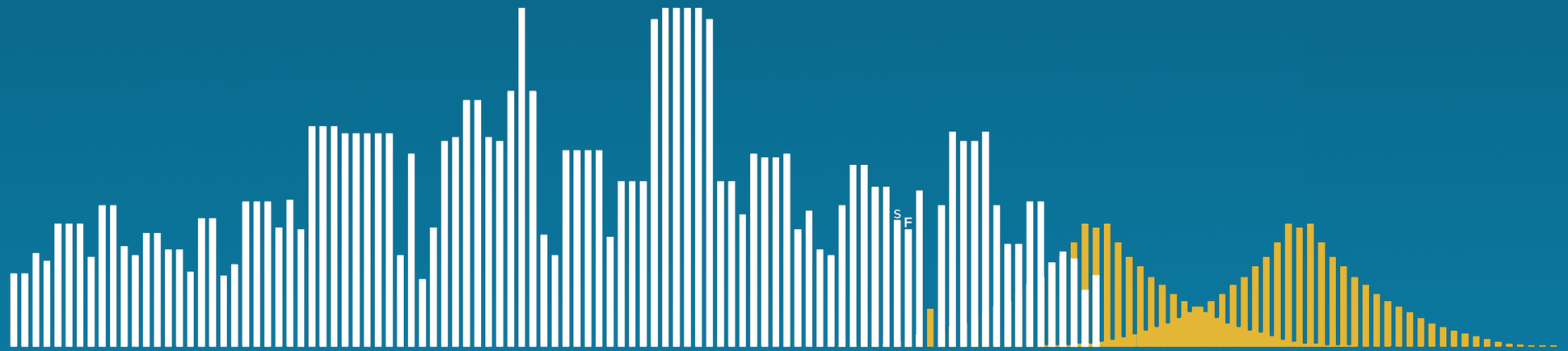


Graphing Nagios

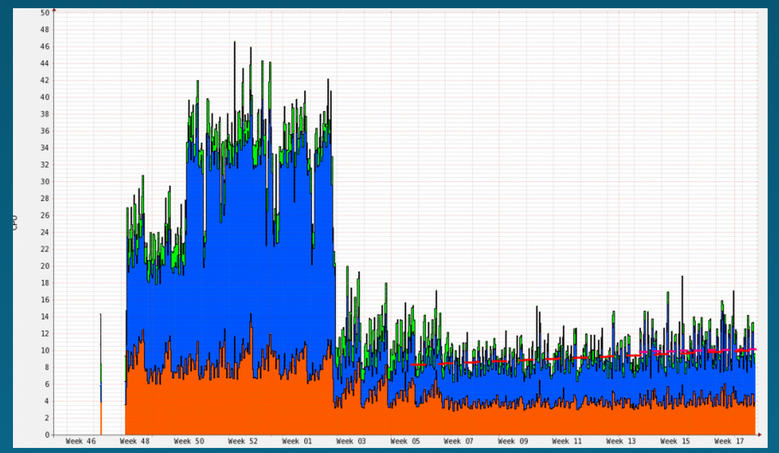
dave@librato.com

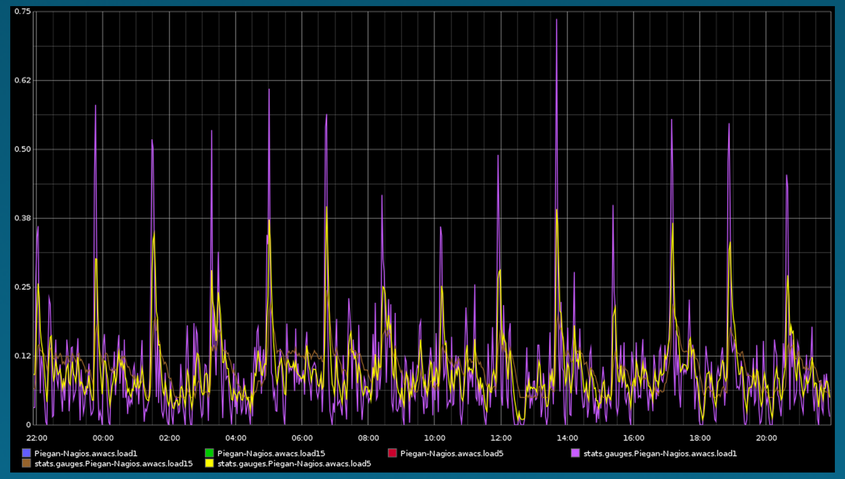
@davejosephsen

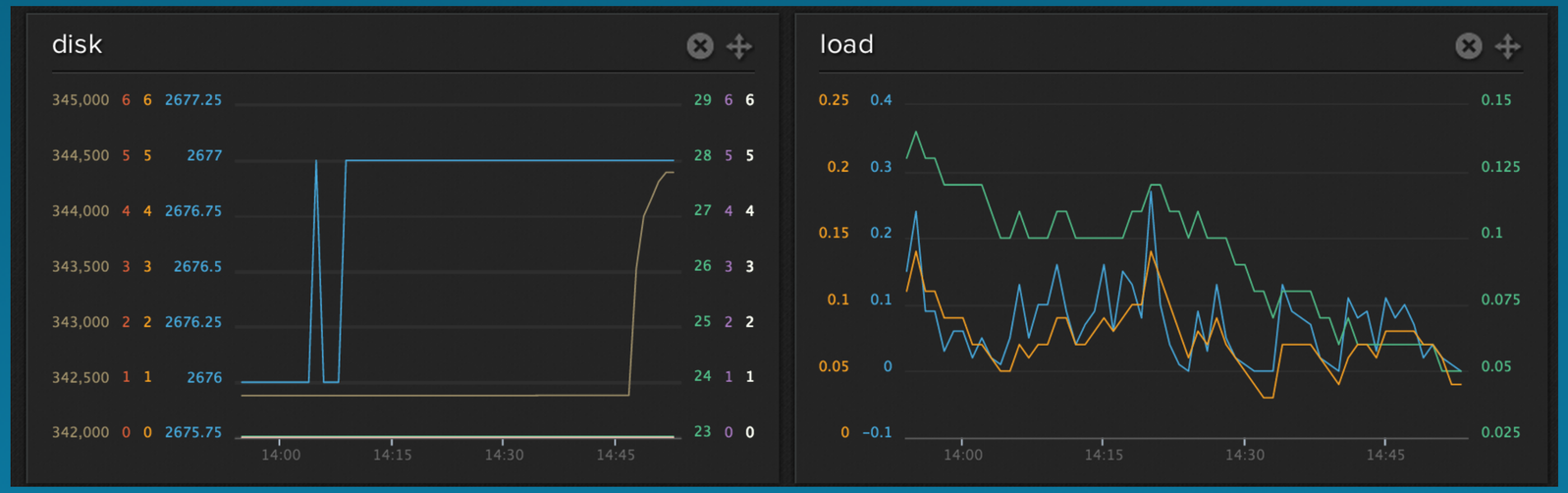
github: djosephsen

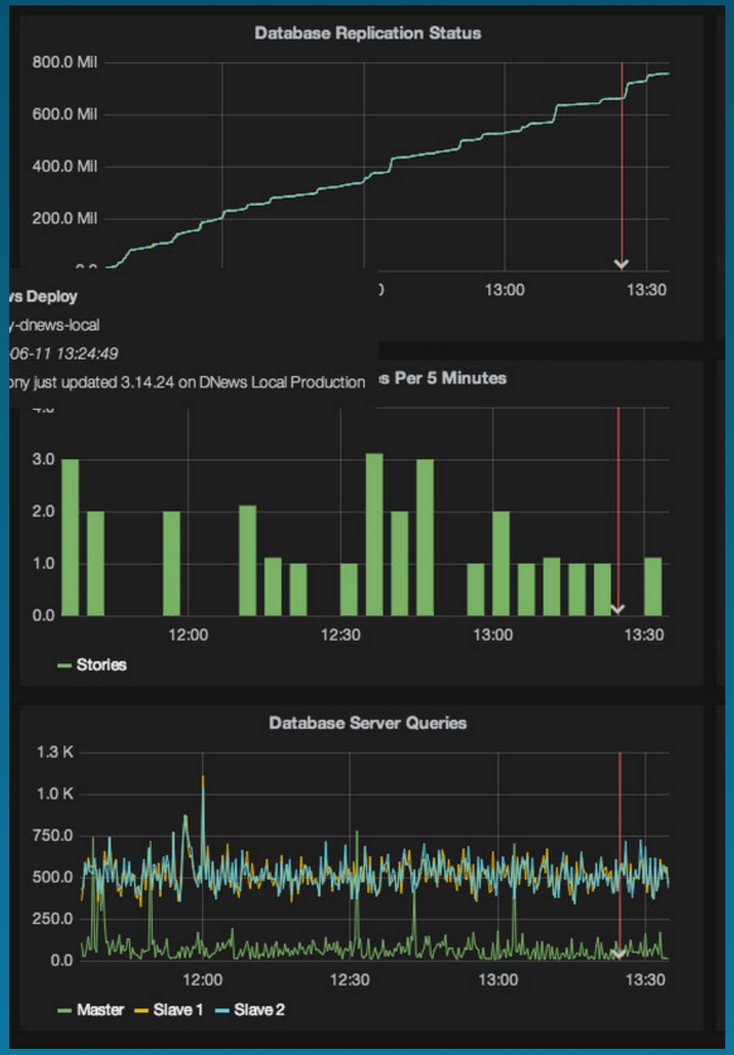


Graphing Nagios









Nagios®

World Conference
North America
2012

Speaker

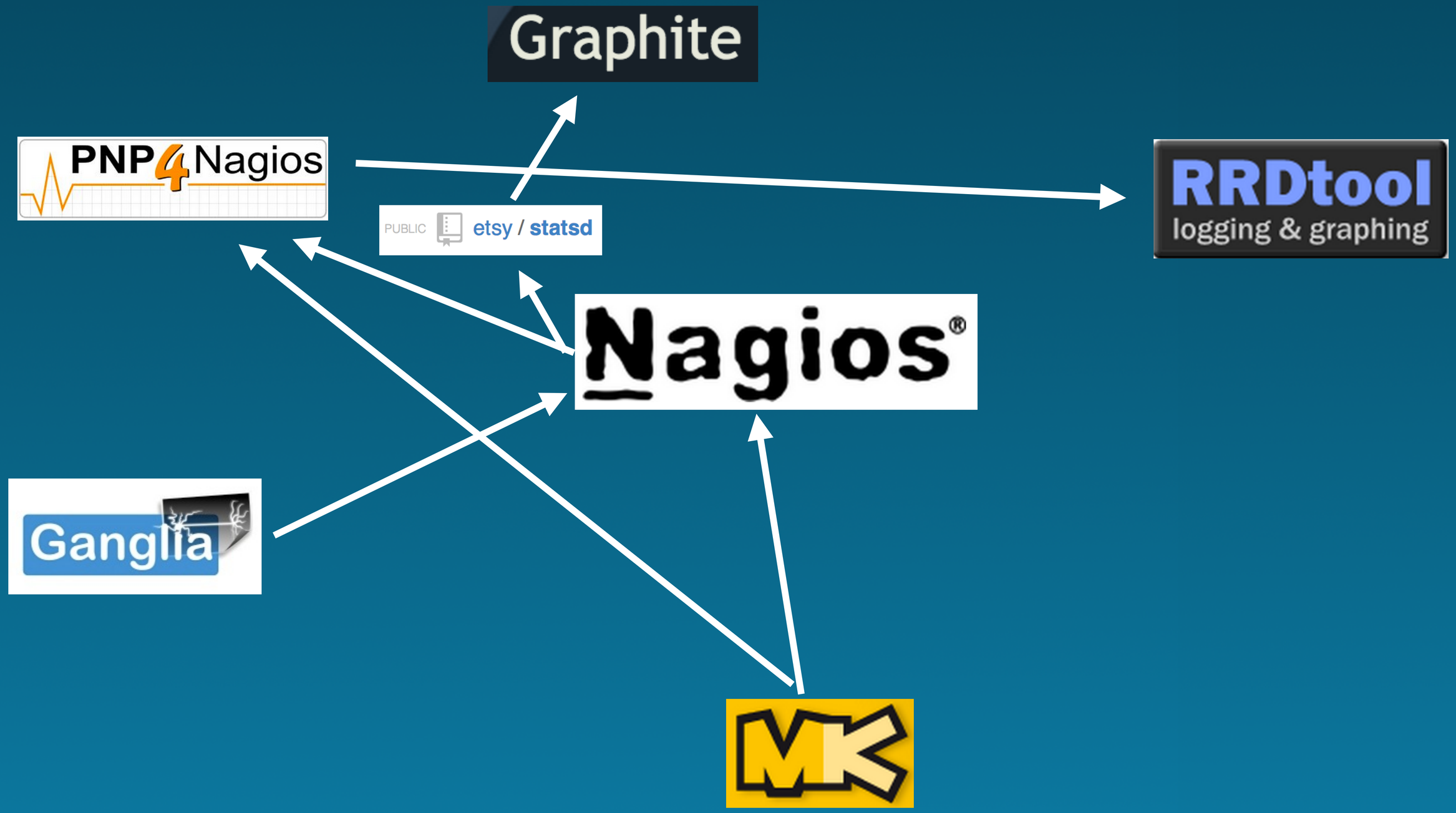
Dave Josephsen

Presentation Title

2002 Called; They Want Their RRDTool Shell Scripts Back

Jer, traditional needs for Fortune-500

- Suitcorp
  - >5000 hosts
  - >20,000 services
  - 1, 9-story office building
  - Plenty of Budget
    - Beefy Hardware
  - 1.5m/1000 hosts


Graphite
PNP4Nagios
RRDtool
logging & graphing
PUBLIC
etsy / statsd
Nagios®
Ganglia
MK

librato™

OpenTSDB

collectd

Metrics Mind the gap.

Graphite

PNP4Nagios

PUBLIC armon / statsite

Ground Control
Manage and monitor your Raspberry Pi with ease

PUBLIC etsy / statsd

DATADOG

Grafana

Nagios®

RIEMANN

PUBLIC ober / pinky

Ganglia

RRDtool
logging & graphing

{ GRAYLOG2

InfluxDB

PUBLIC eric / metriks

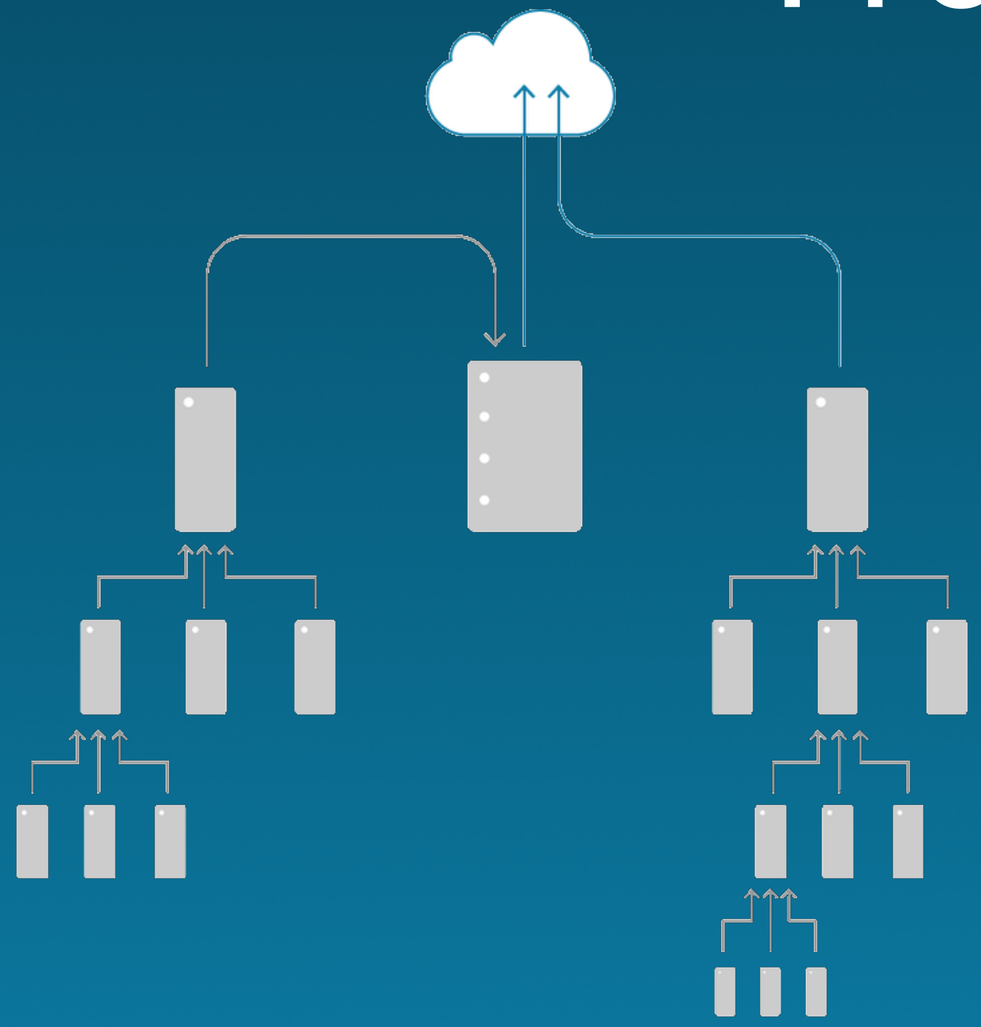
Heka

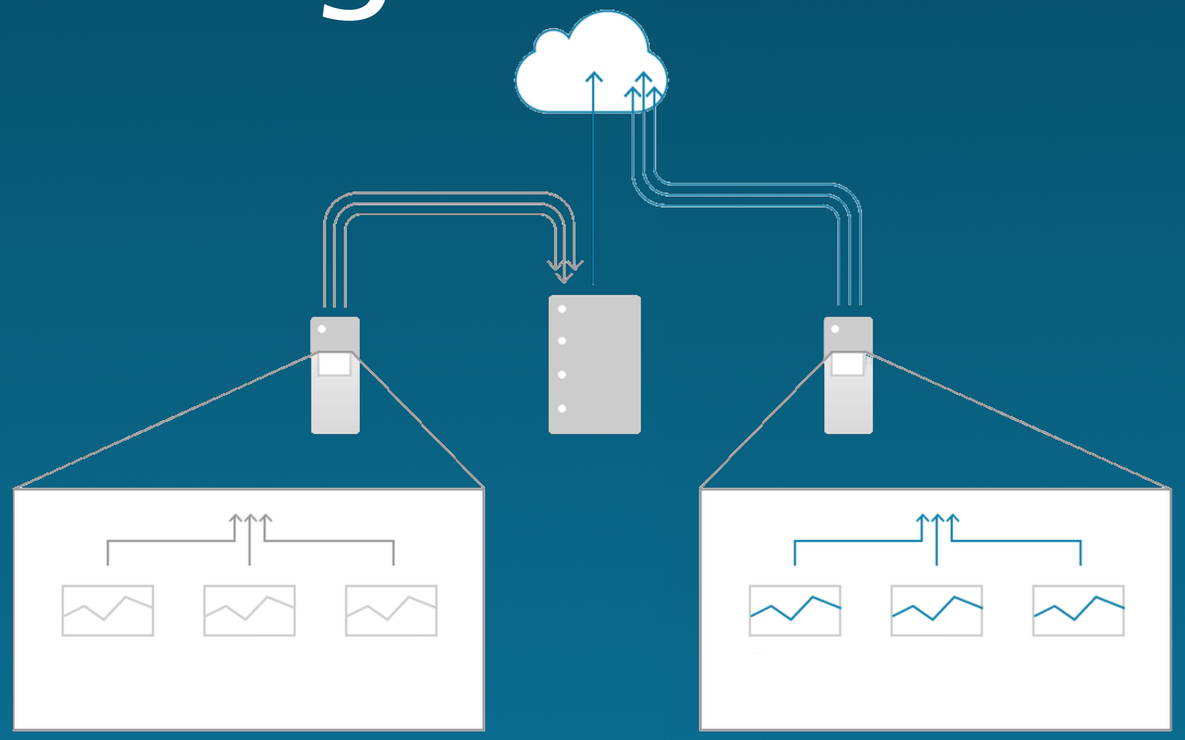
shawn-sterling / graphios

MK

fluentd

Design Patterns for Metrics Processing





Things that measure

Nagios®

PUBLIC eric / metriks

Metrics Mind the gap.

Ground Control
Manage and monitor your Raspberry Pi with ease

PUBLIC ober / pinky

collectd

Things that extract, transport and process

PUBLIC etsy / statsd

PUBLIC armon / statsite

PNP4Nagios

Heka

RIEMANN

shawn-sterling / graphios

GRAYLOG2

fluentd

All of the above

MK

Ganglia

DATADOG

Things that store and graph

RRDtool
logging & graphing

librato.™

Graphite

Grafana

OpenTSDB

InfluxDB

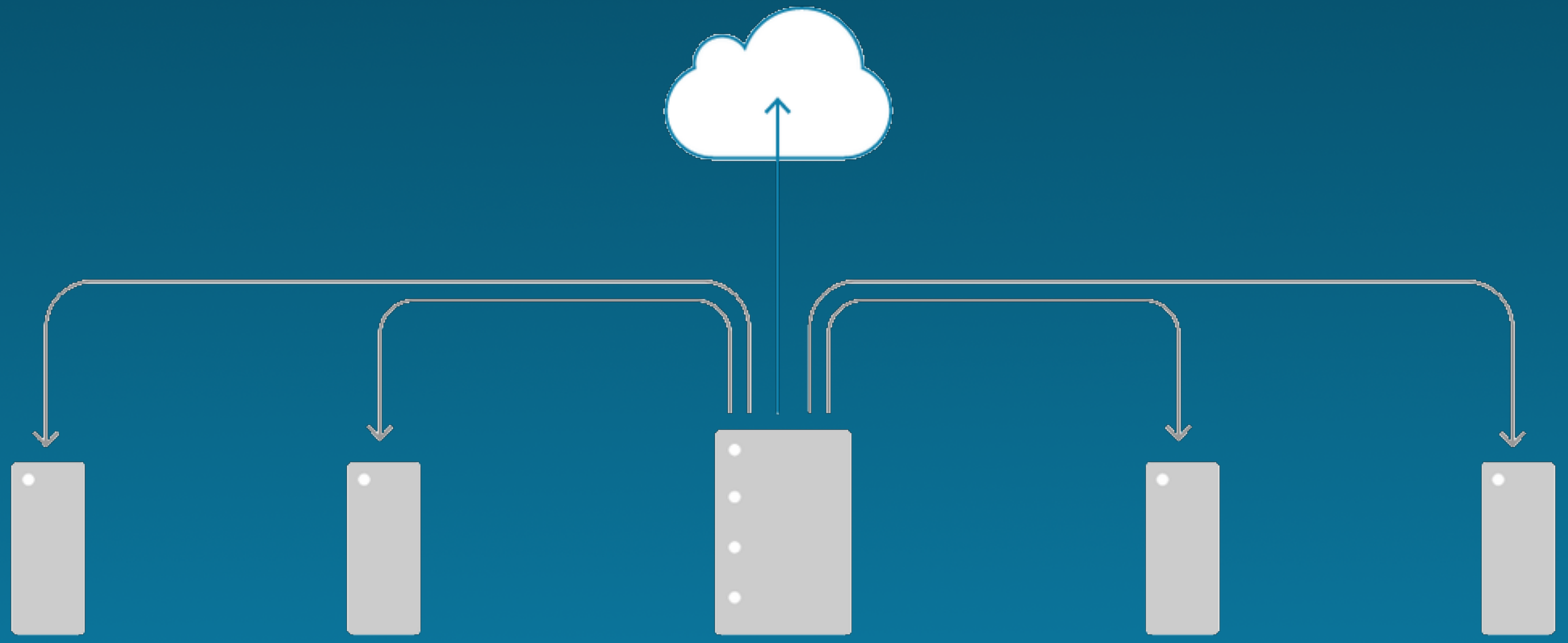
Today At 4:30

Alert On What You Draw

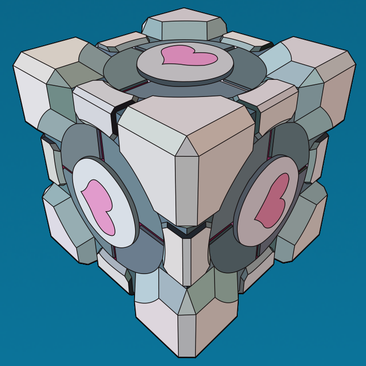
David Josephsen - Engineer/Evangelist @ Librato

The damage done by disparate signals and how to fix it by promoting metrics to first class citizens.

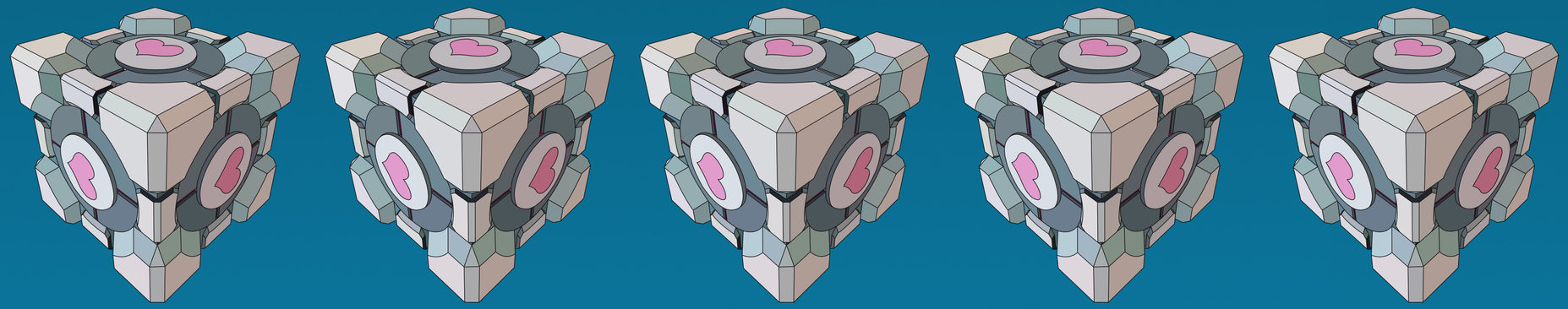
Pattern1: Centralized Polling



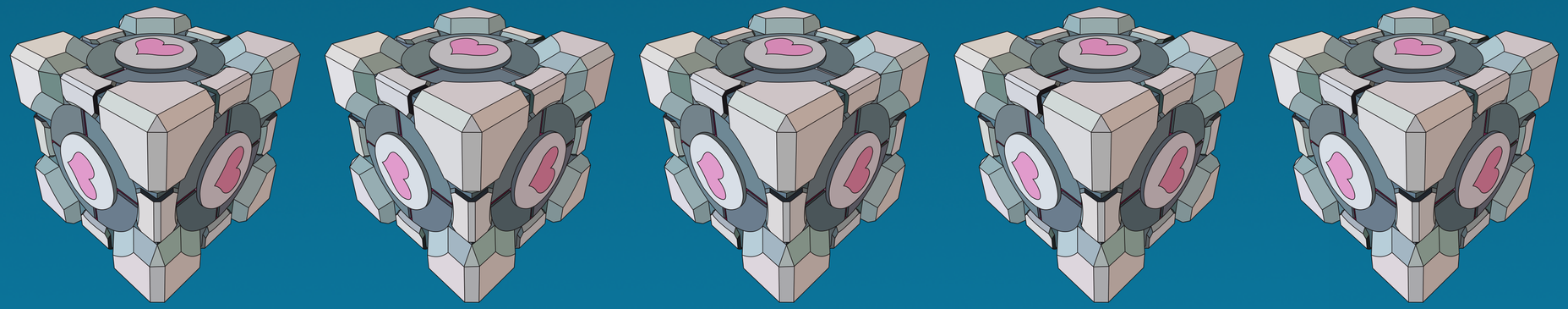
Pattern1: Centralized Polling



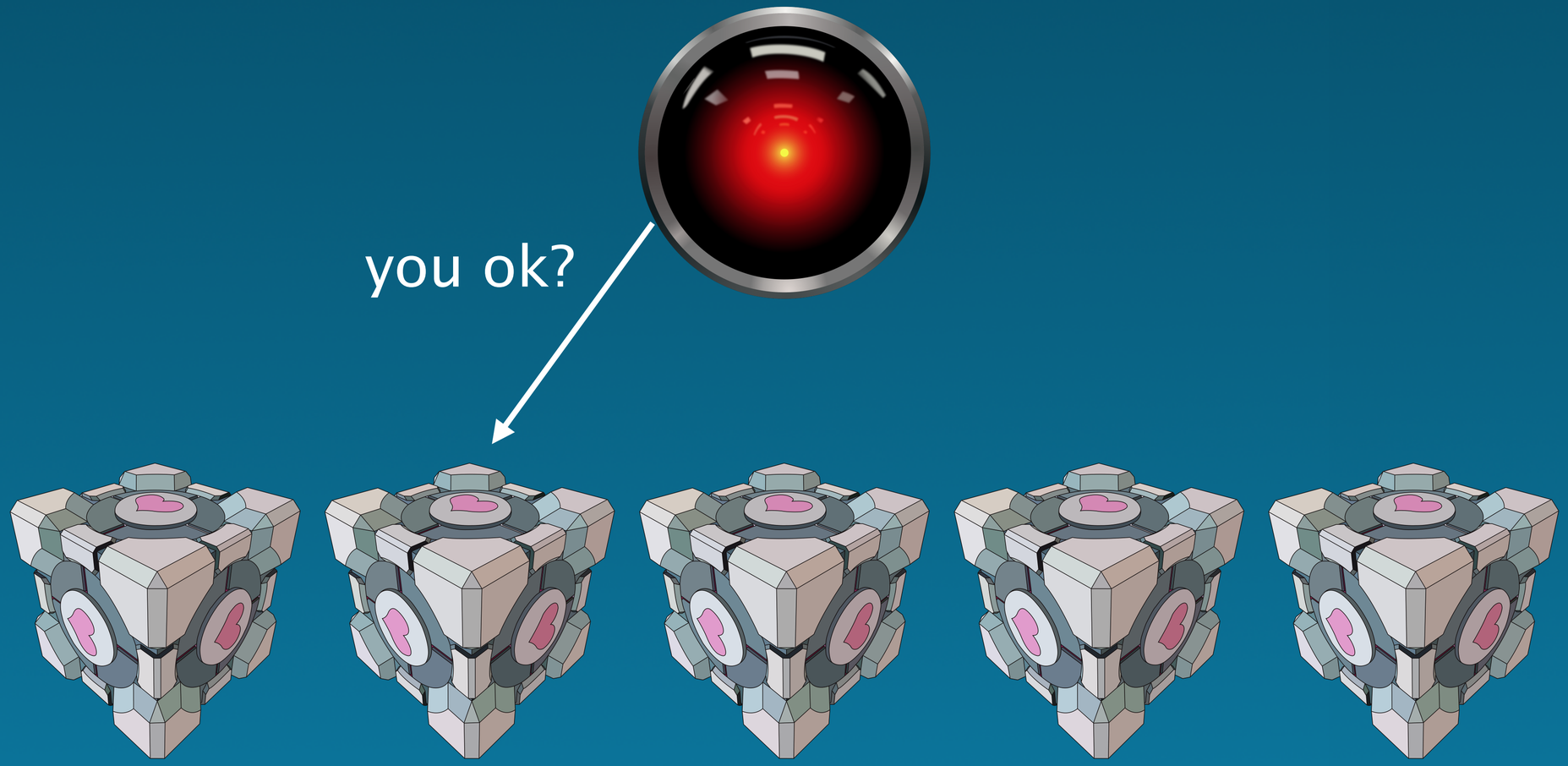
Pattern1: Centralized Polling



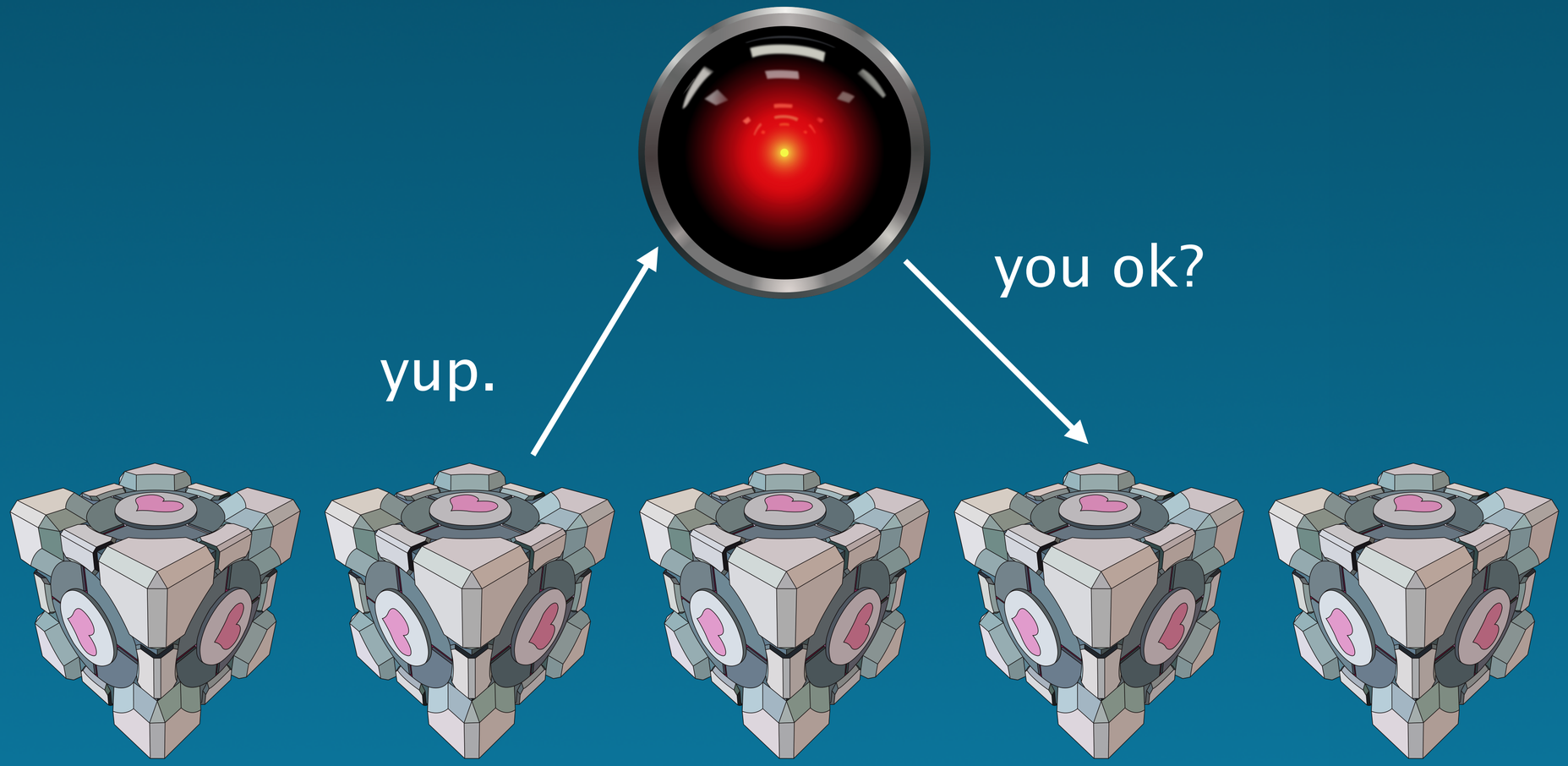
Pattern1: Centralized Polling



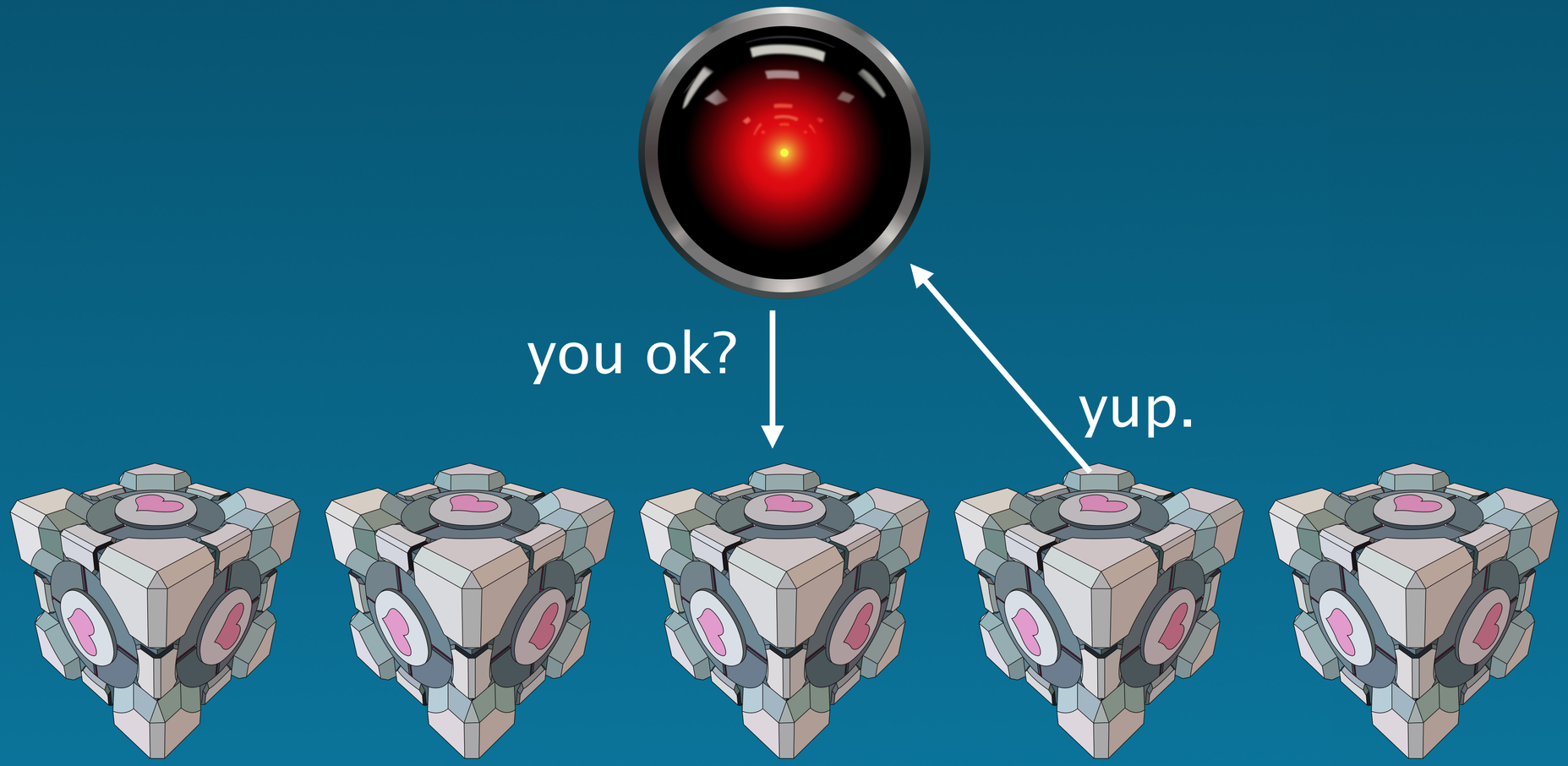
Pattern1: Centralized Polling



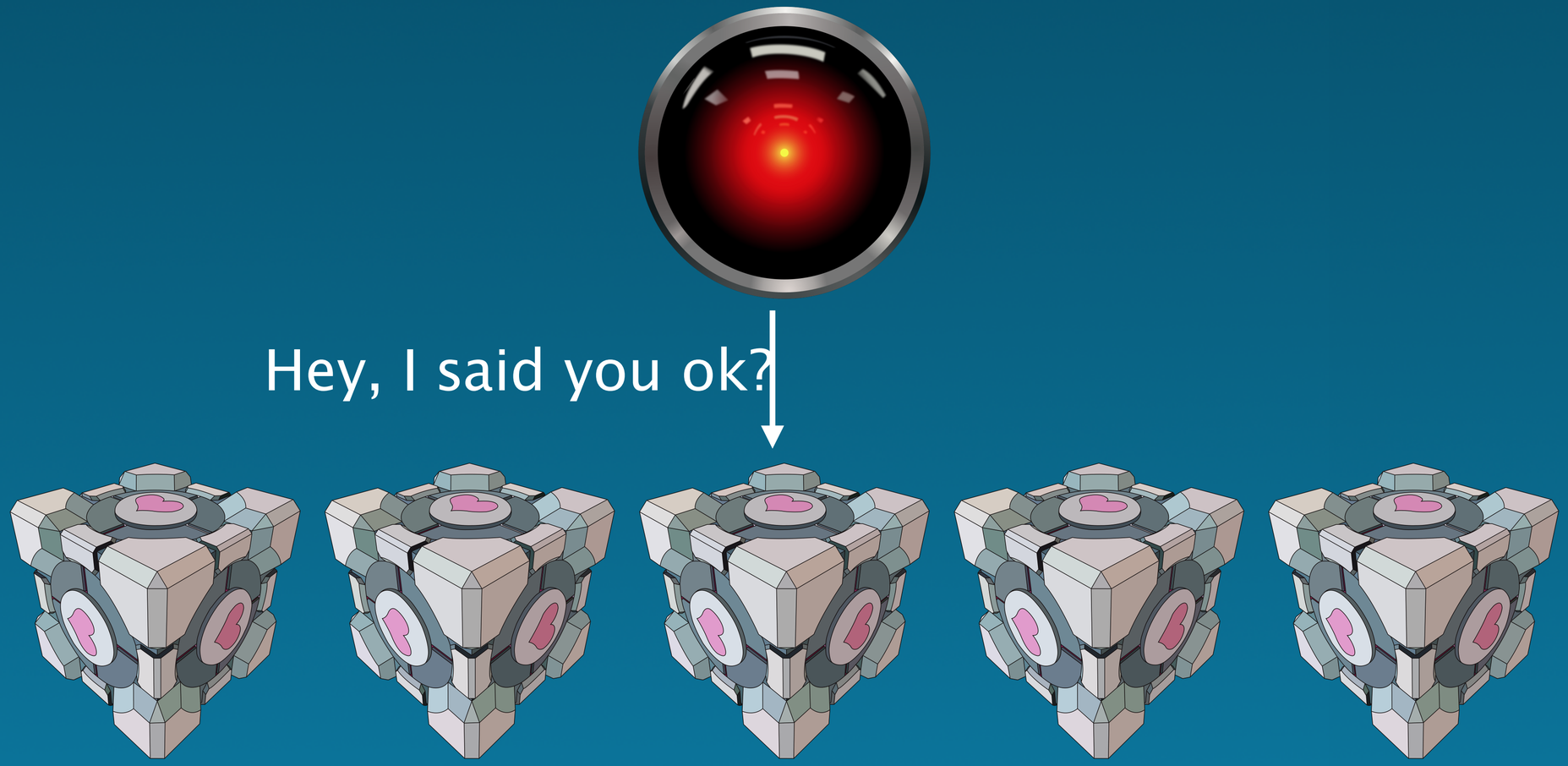
Pattern1: Centralized Polling



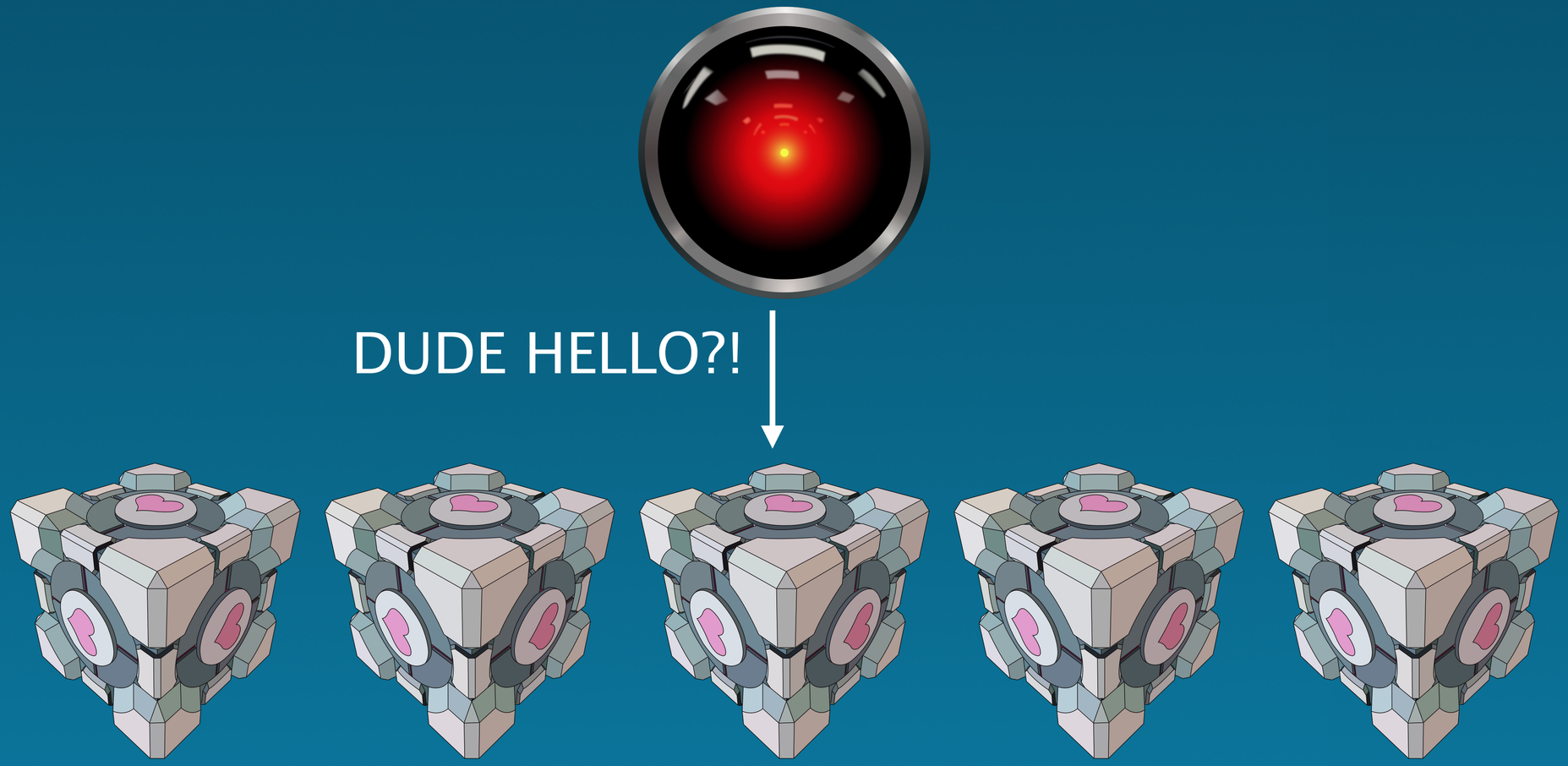
Pattern1: Centralized Polling



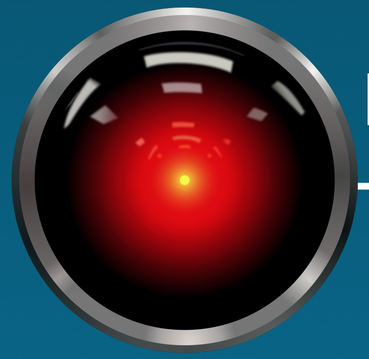
Pattern1: Centralized Polling



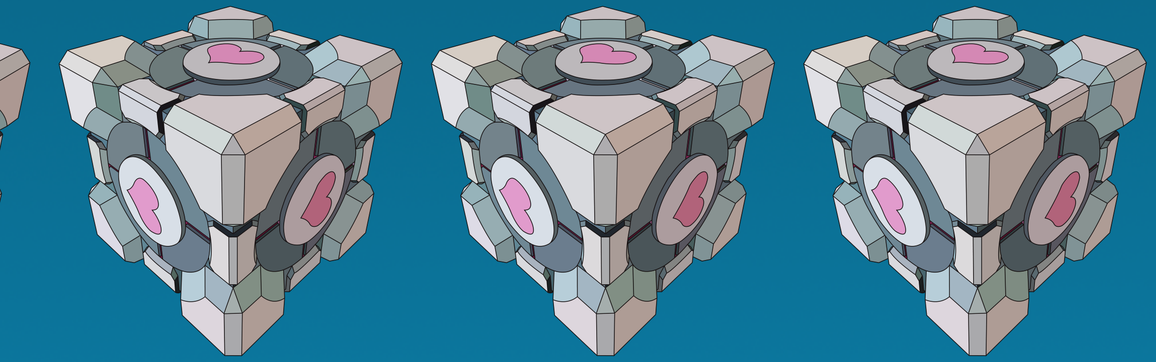
Pattern1: Centralized Polling



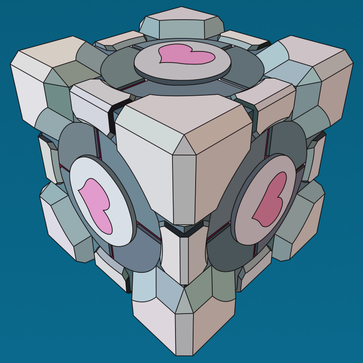
Pattern1: Centralized Polling



host[2] is NOT OK



Pattern1: Centralized Polling



ya ok?

Nagios®

Pattern1: Centralized Polling

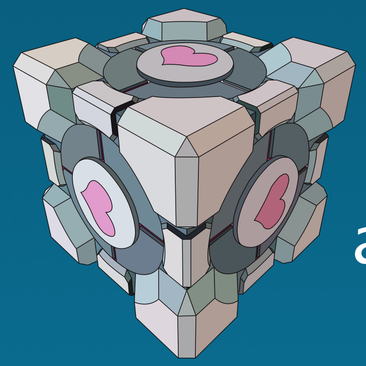
Yeah I’m ok | rta=50.582001ms; pl=0%;0

Pattern1: Centralized Polling

(performance data)

Yeah I'm ok | rta=50.582001ms; pl=0%;0

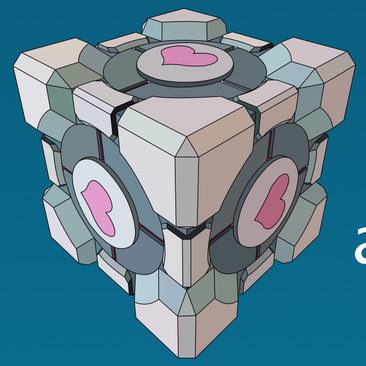
Pattern1: Centralized Polling



Nagios®

all good (perfdata)

Pattern1: Centralized Polling



all good (perfdata)

Nagios®

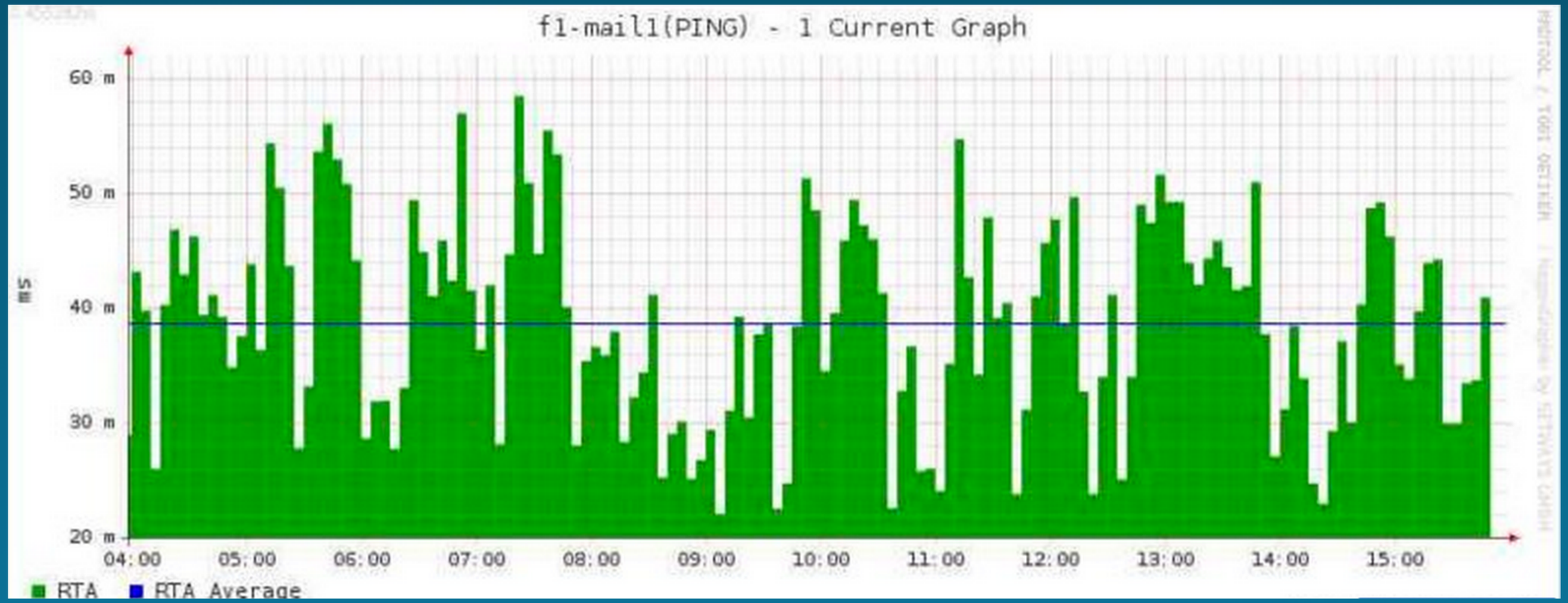
PNP4Nagios

RRDtool
logging & graphing

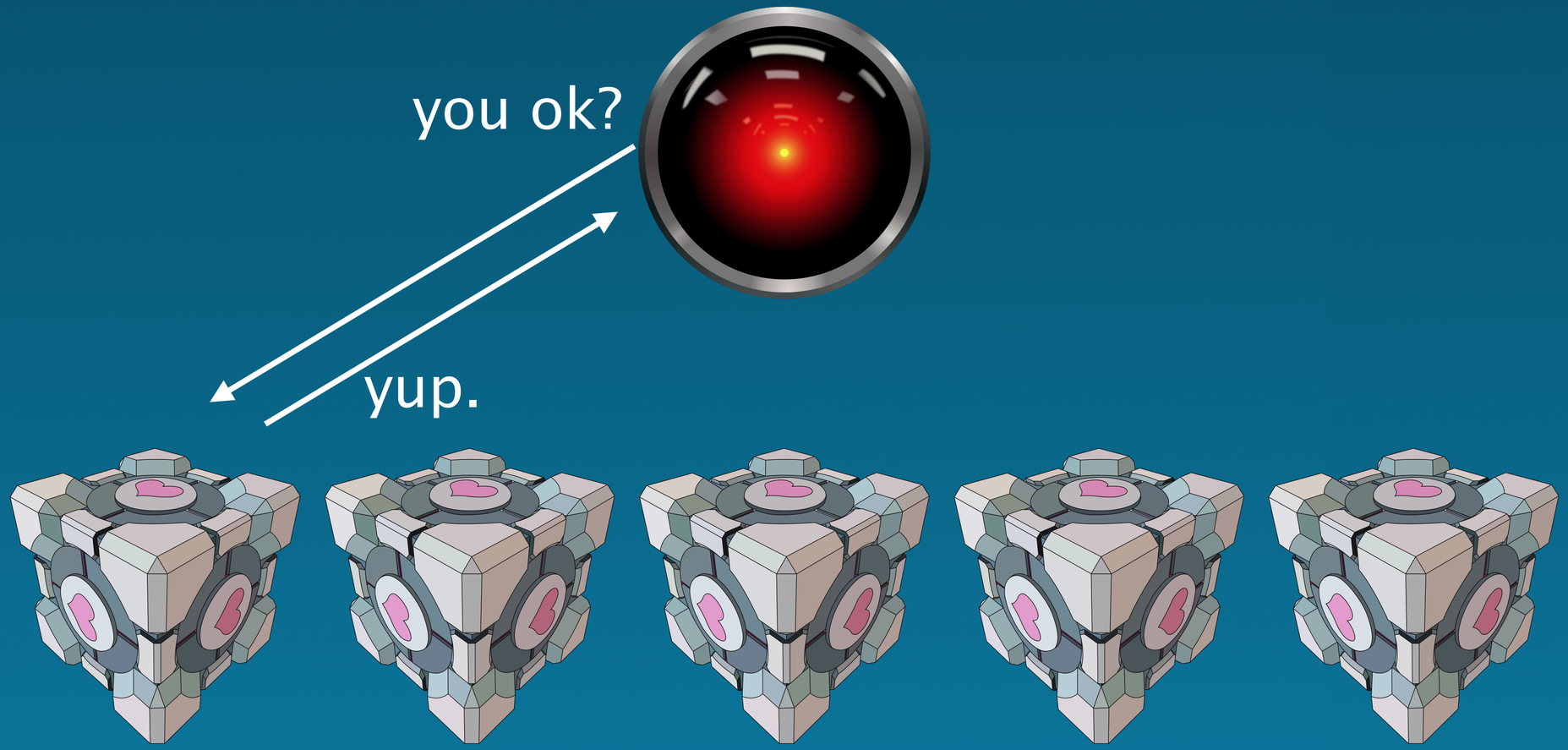
Pattern1: Centralized Polling

Status	Last Check	Duration	Status Information
UP	10-01-2014 15:11:59	2114d 23h 37m 47s	PING OK - Packet loss = 0%, RTA = 0.00 ms

Pattern1: Centralized Polling

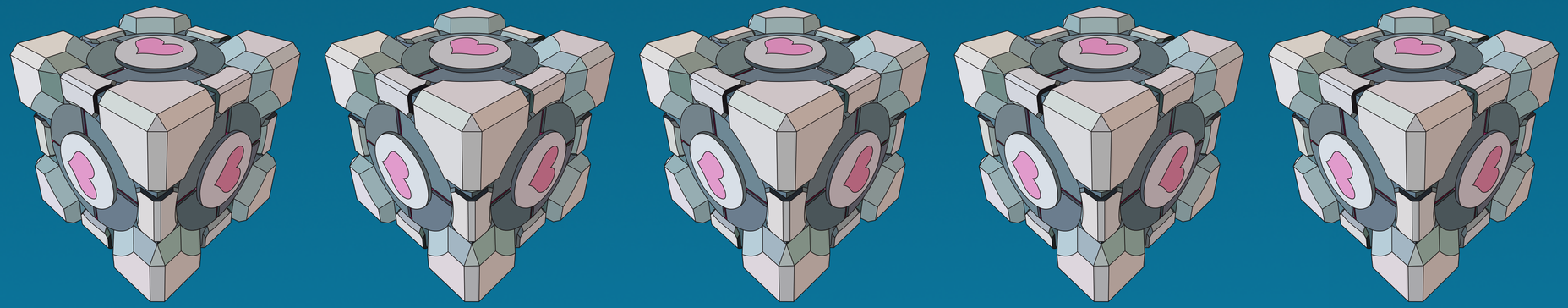


Pattern1: Centralized Polling

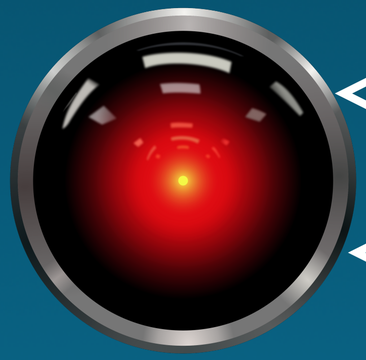


Pattern1: Centralized Polling

<——Limits Resolution

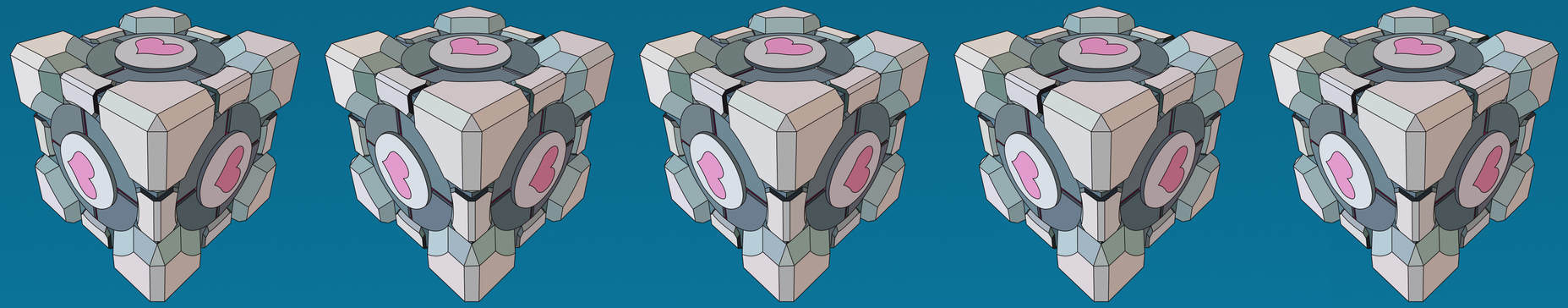


Pattern1: Centralized Polling

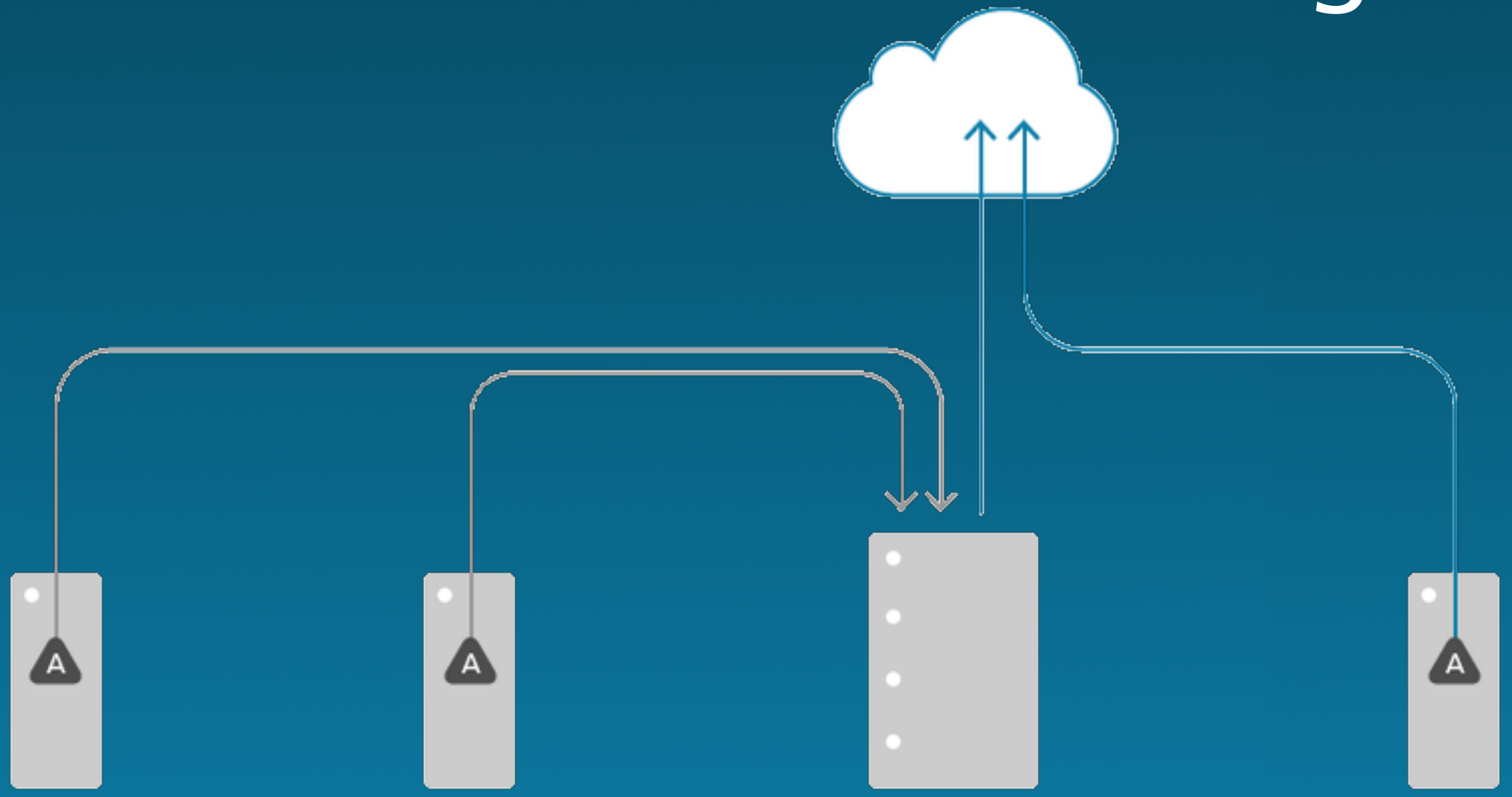


<——Limits Resolution

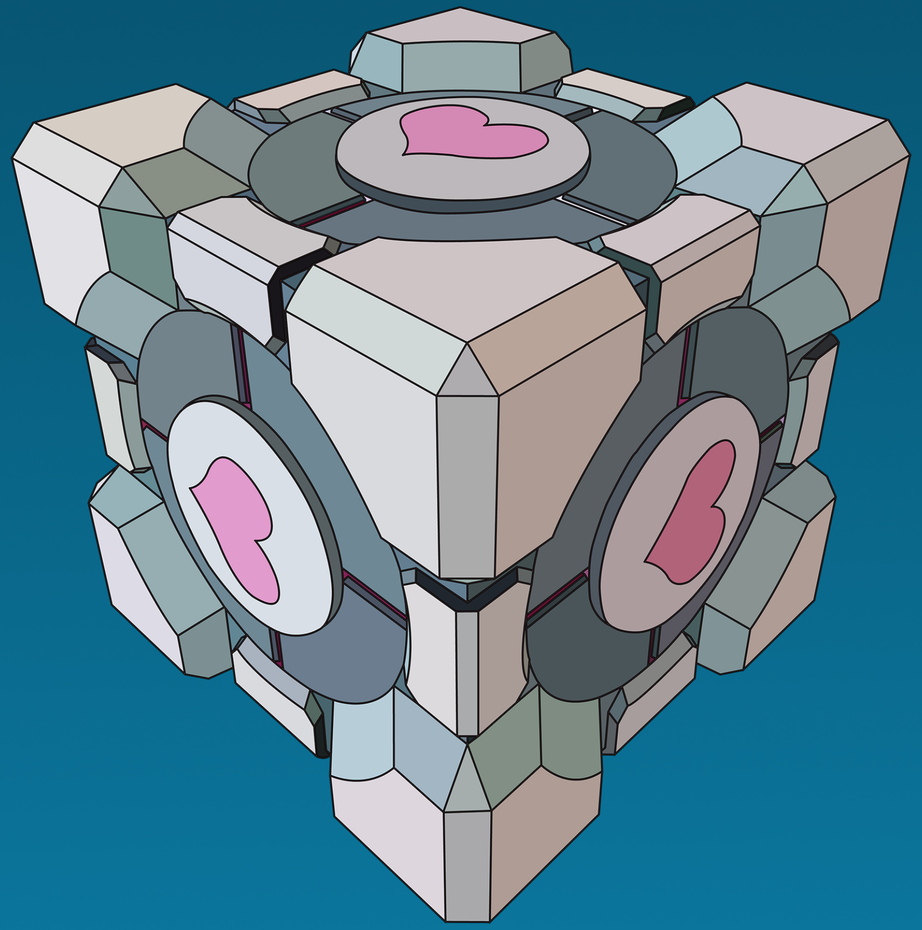
<——Limits Scalability



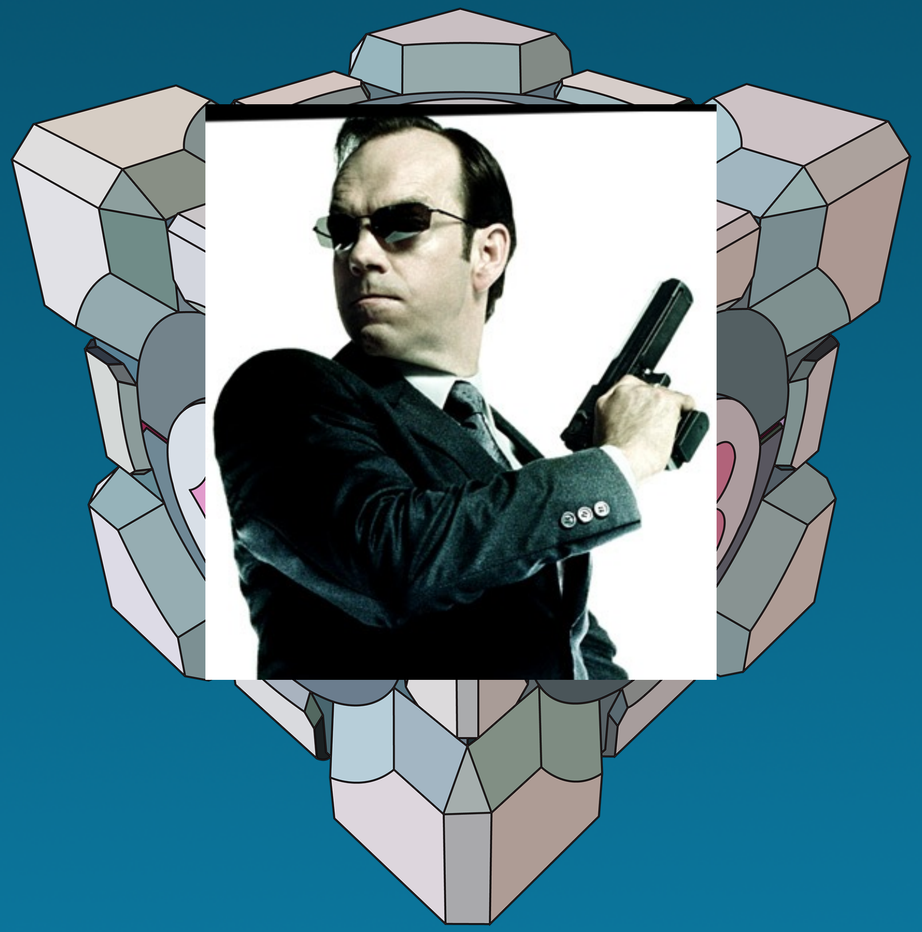
Pattern2: Autonomous Agents



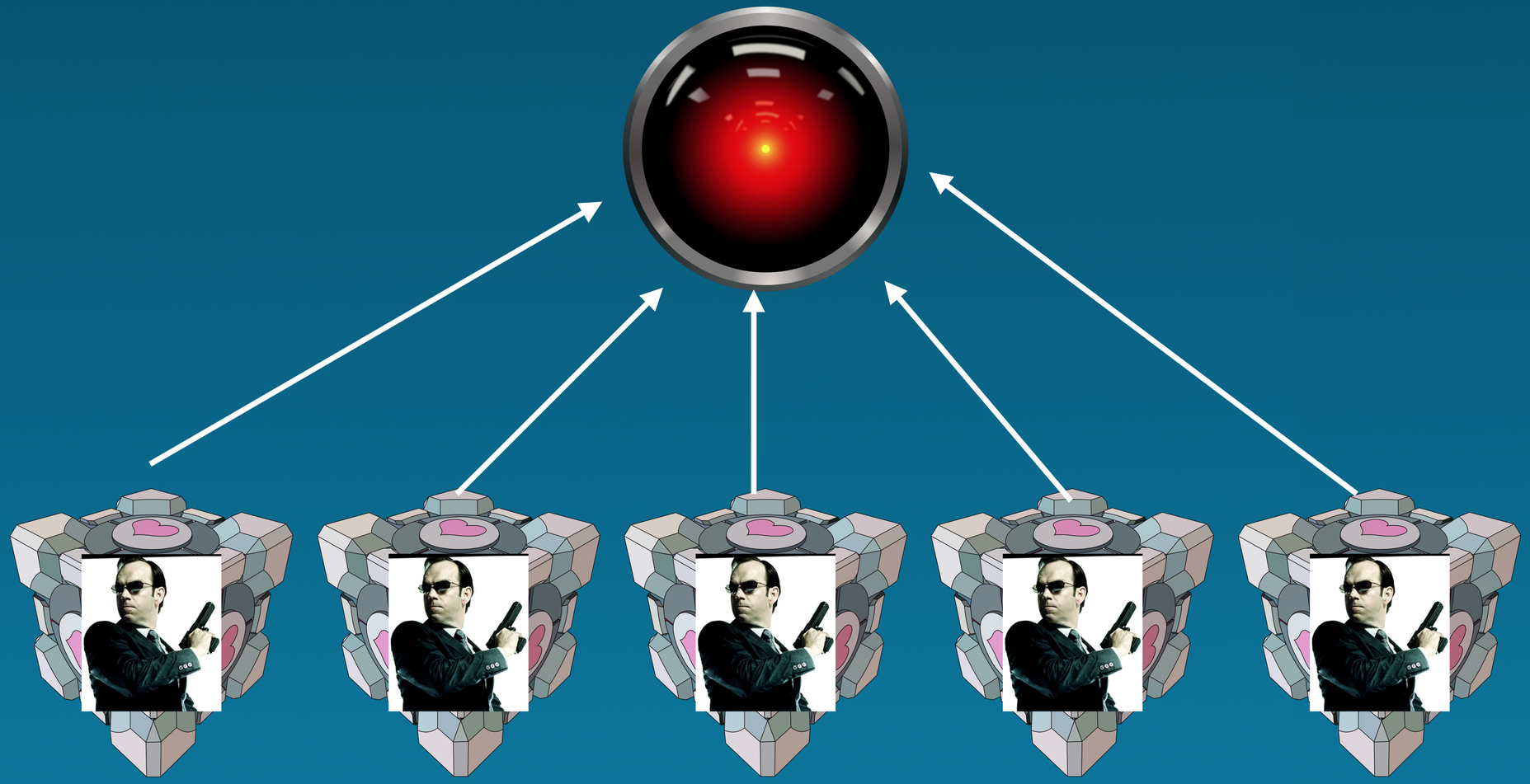
Pattern2: Autonomous Agents



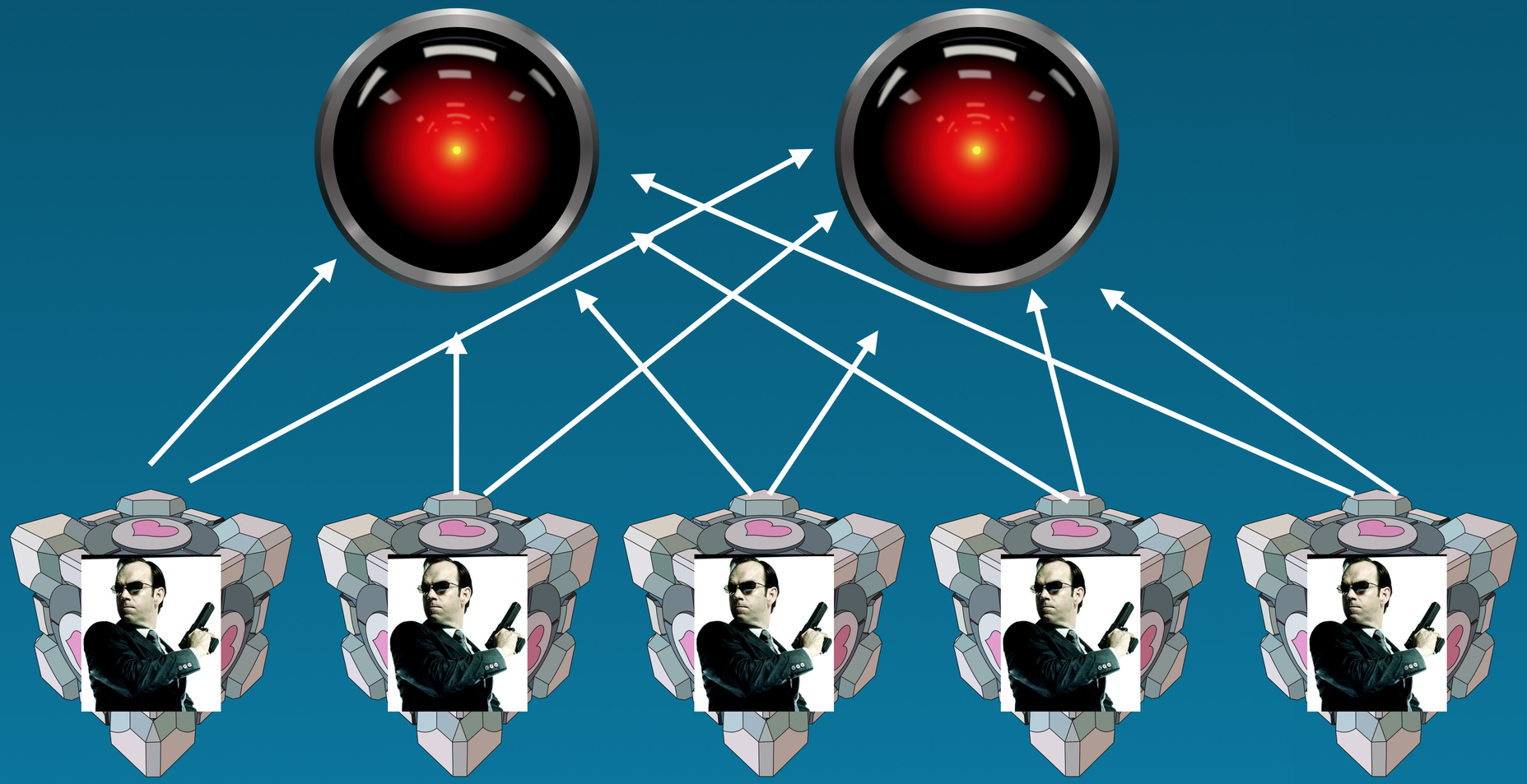
Pattern2: Autonomous Agents



Pattern2: Autonomous Agents



Pattern2: Autonomous Agents

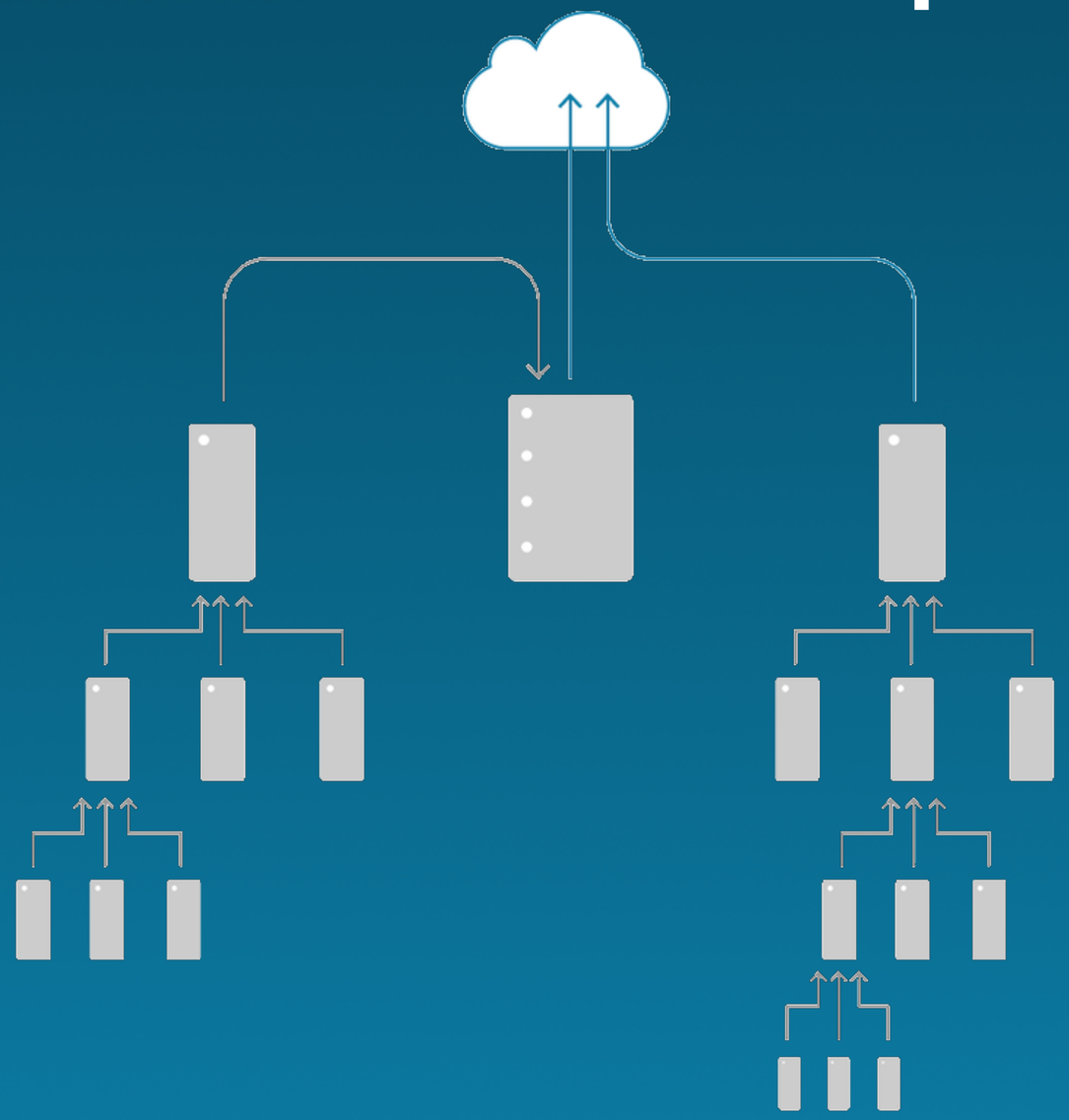


Pattern2: Autonomous Agents

Pattern2: Autonomous Agents

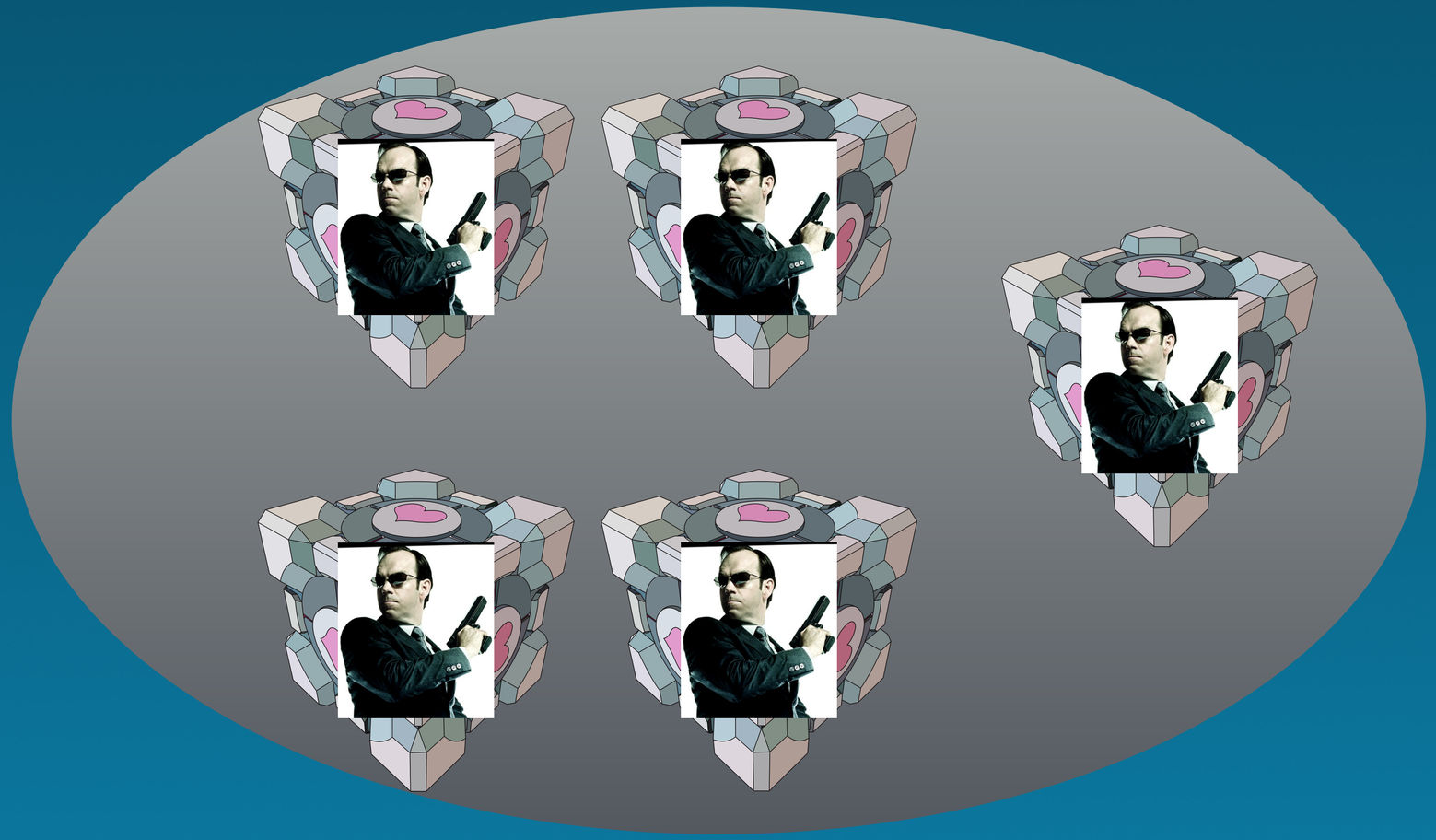
Still Doesn't Scale

Pattern3: The Rollup Pattern

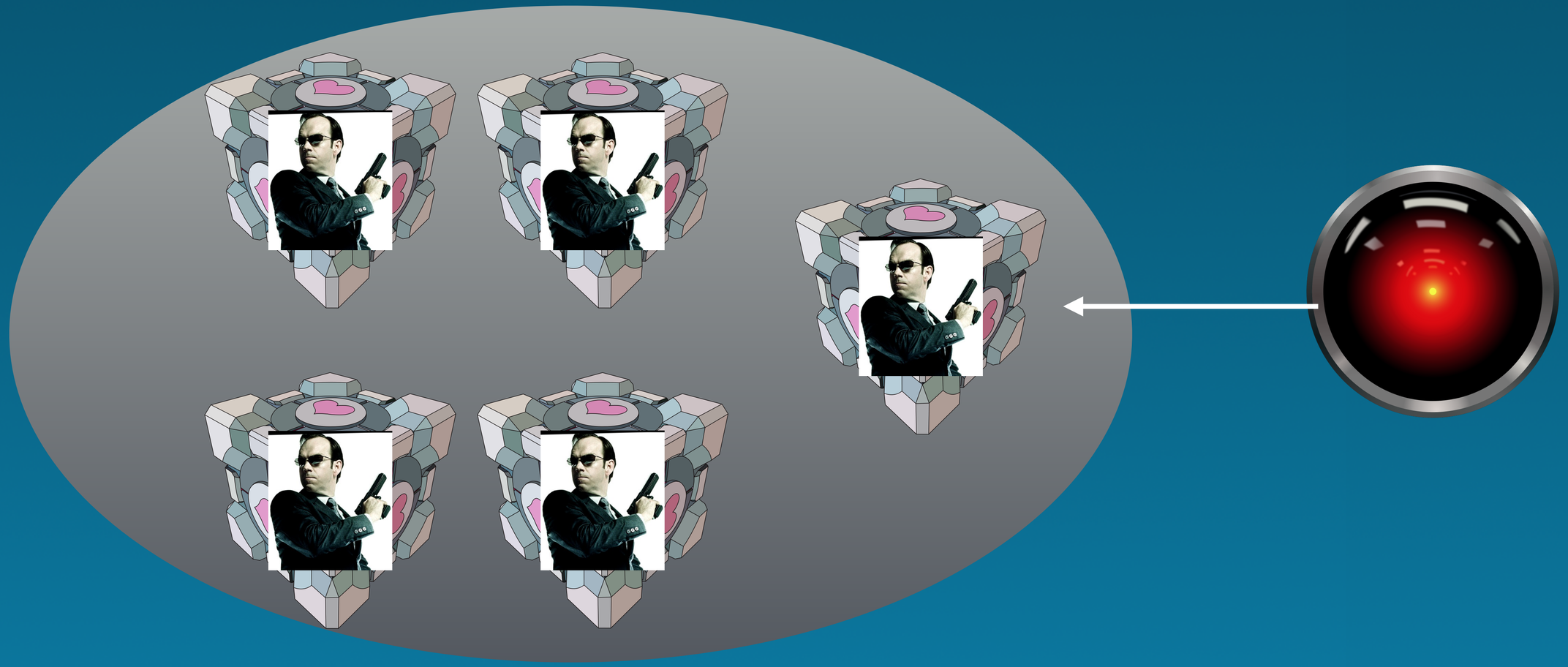


Pattern3: The Rollup Pattern

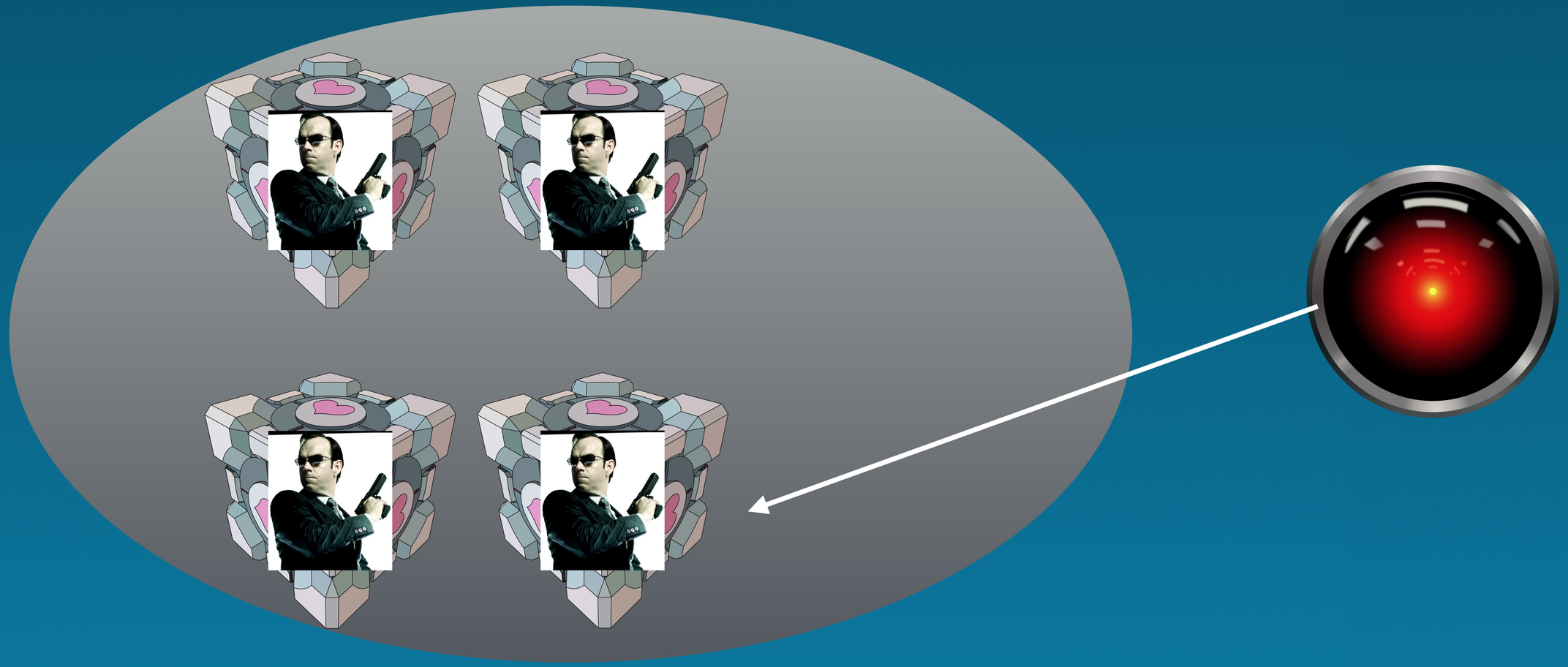
Pattern3: The Rollup Pattern



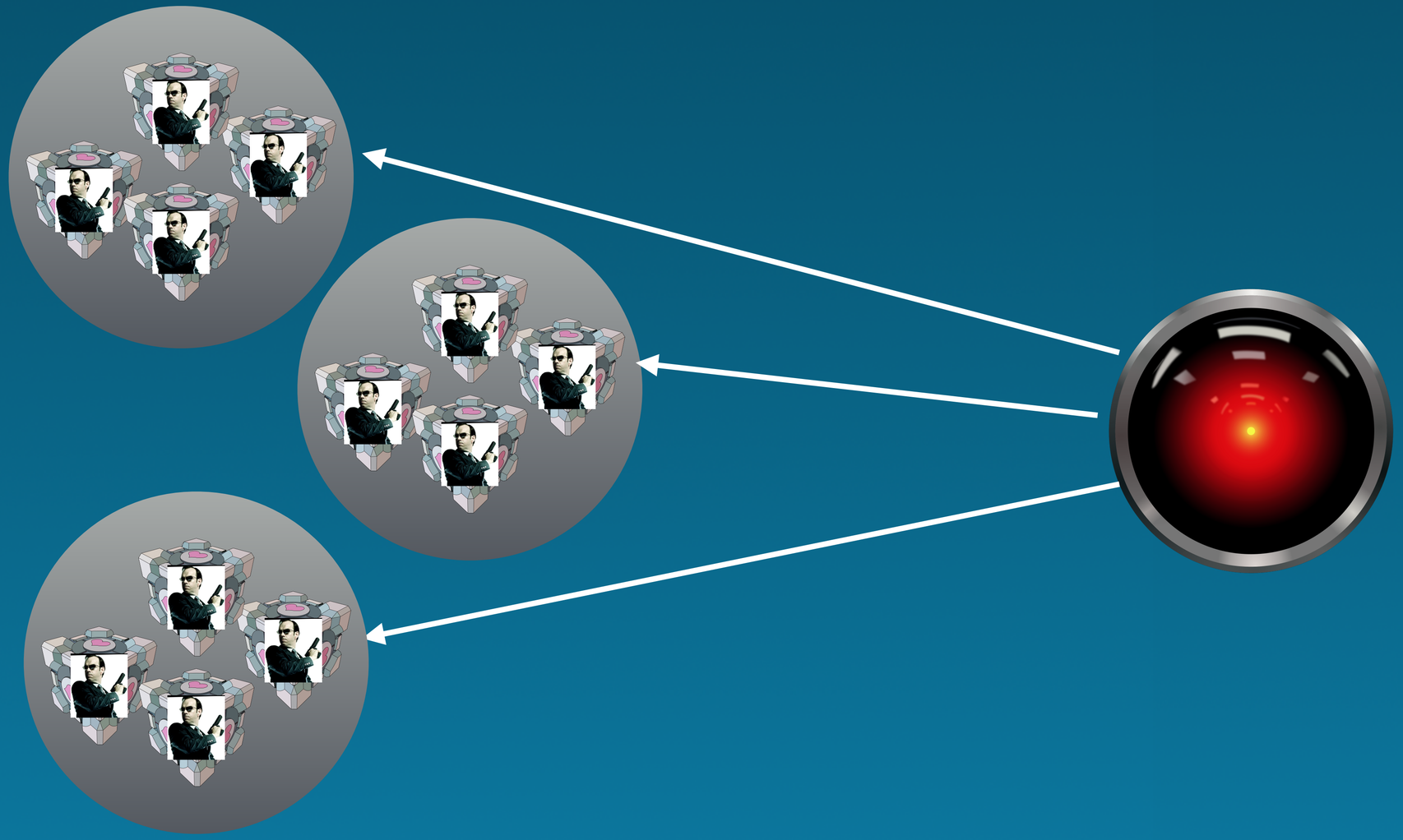
Pattern3: The Rollup Pattern



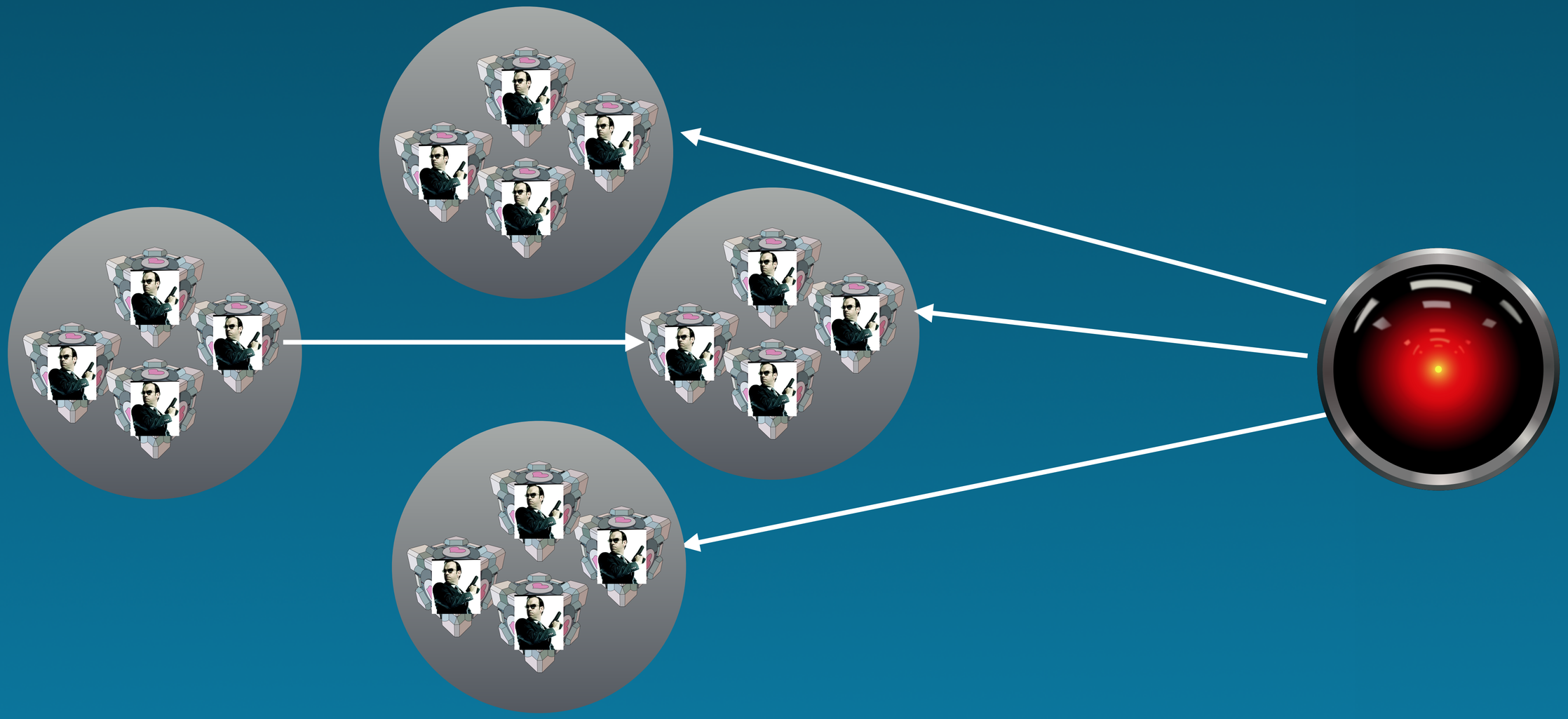
Pattern3: The Rollup Pattern



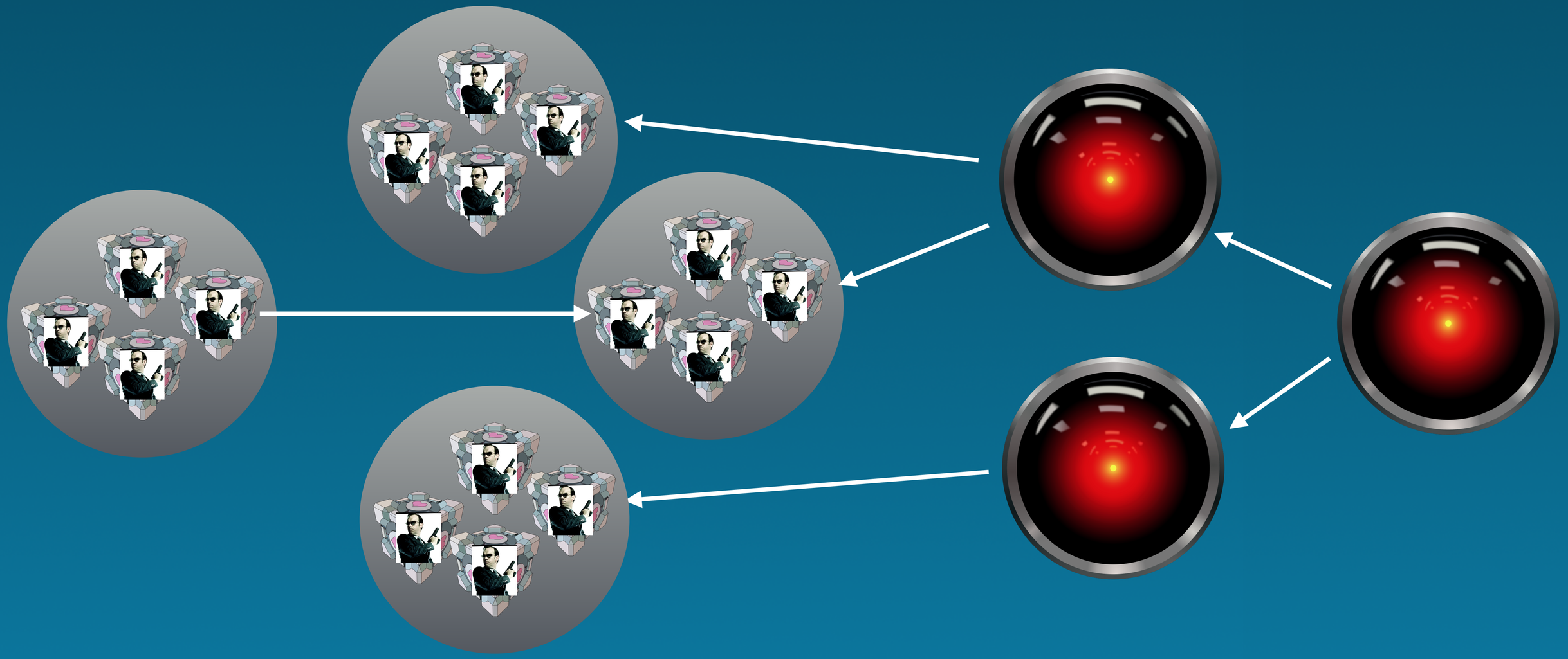
Pattern3: The Rollup Pattern



Pattern3: The Rollup Pattern



Pattern3: The Rollup Pattern



Pattern3: The Rollup Pattern

Pattern3: The Rollup Pattern

gmond

gmetad

Pattern3: The Rollup Pattern

gmond

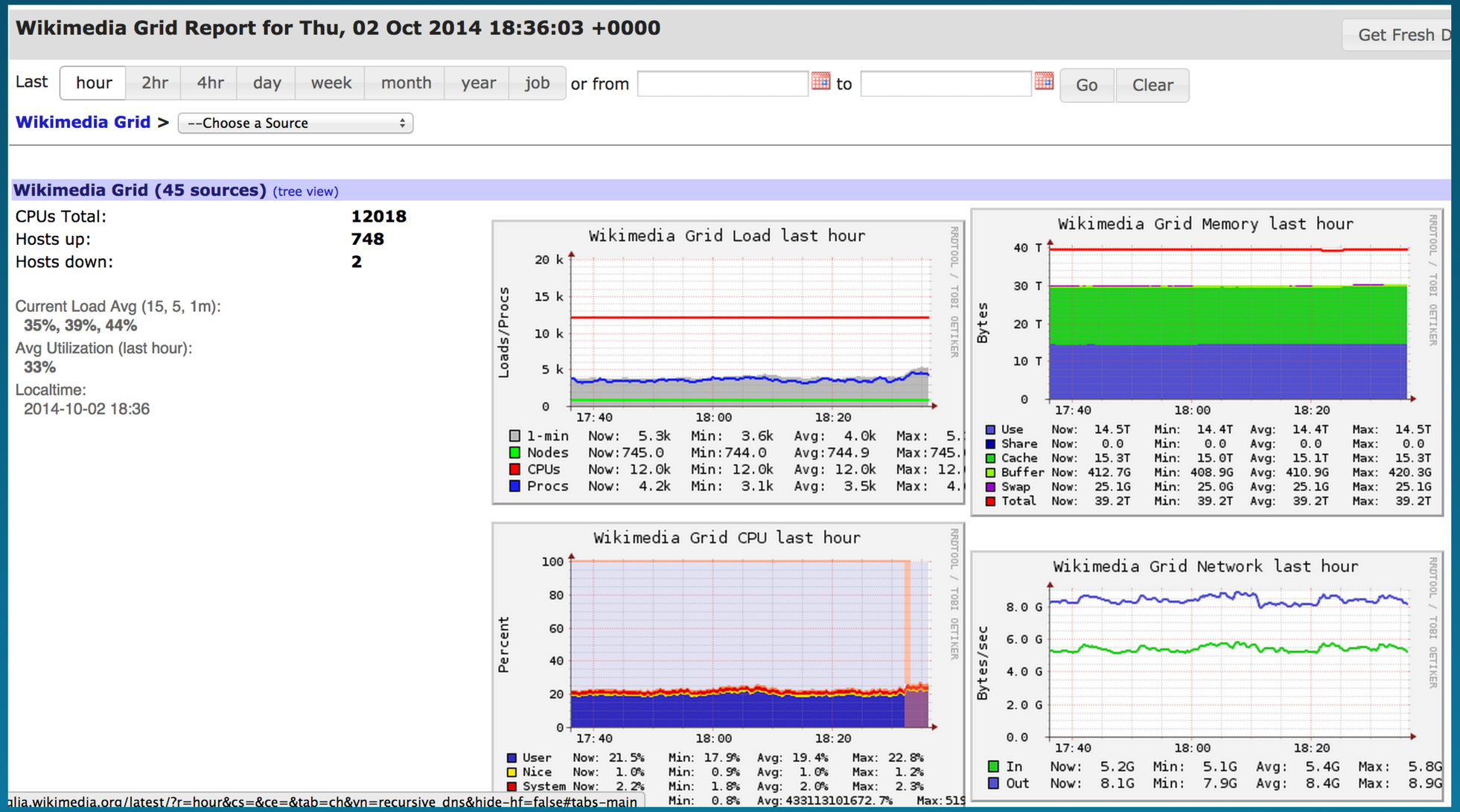
gmetad

Pattern3: The Rollup Pattern

gmond

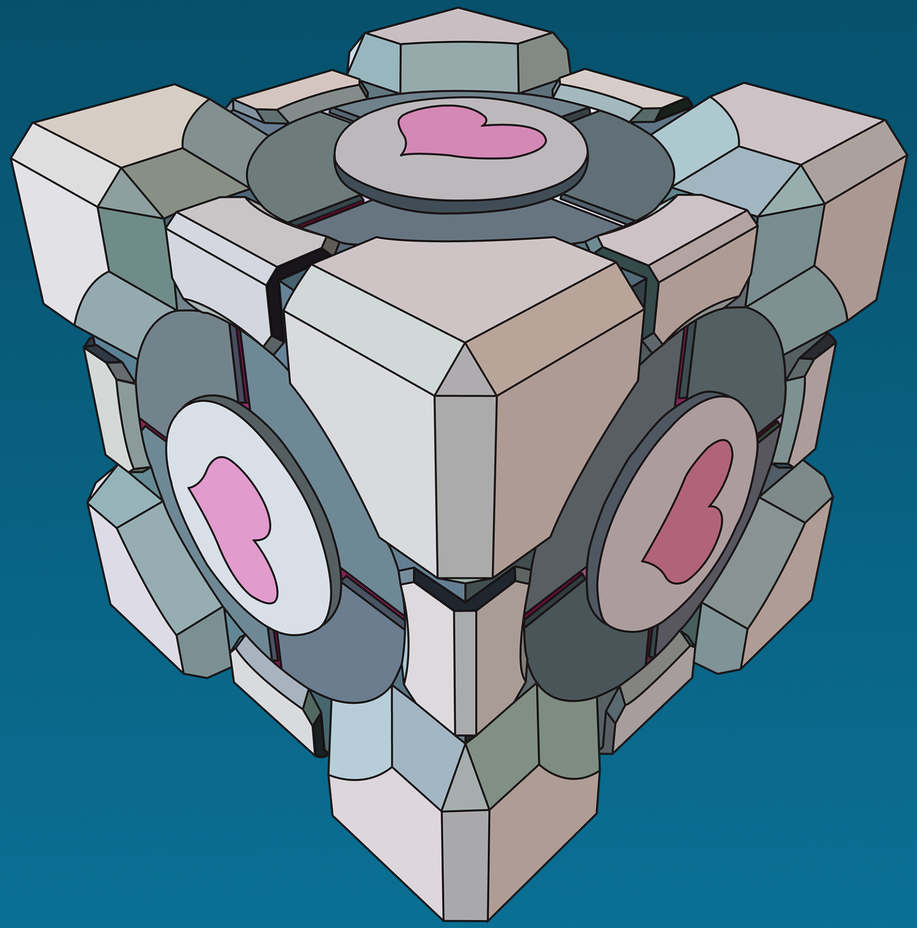
gmetad

Pattern3: The Rollup Pattern

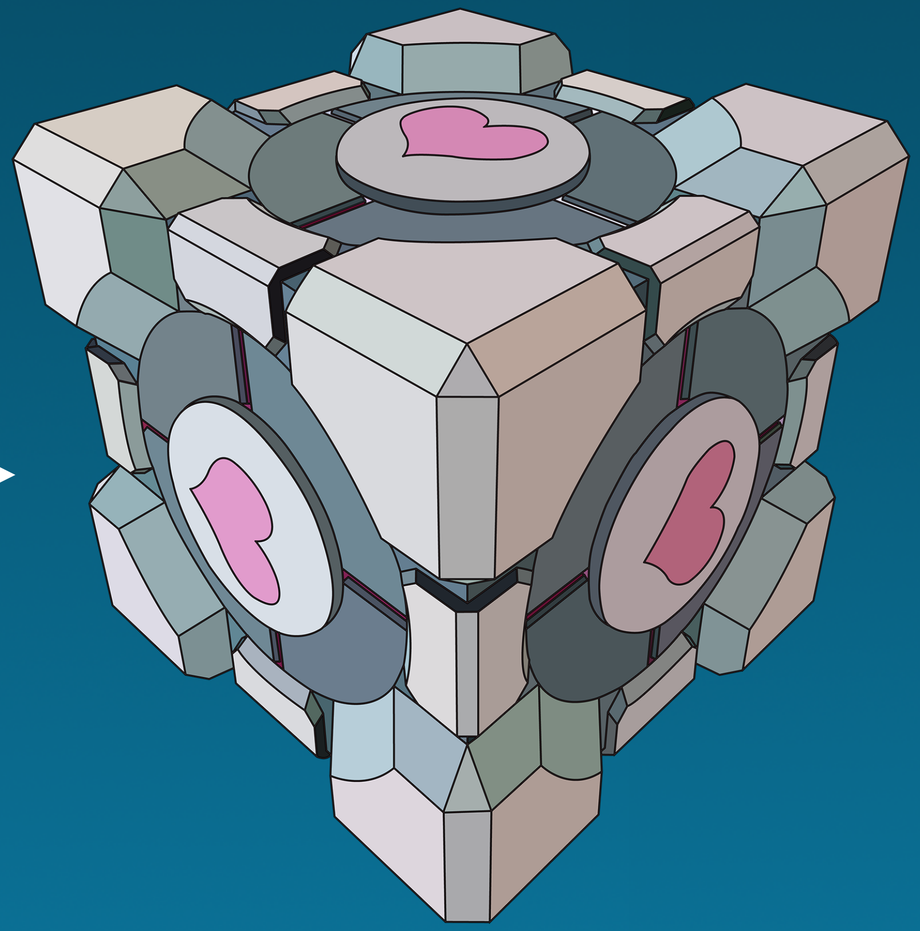


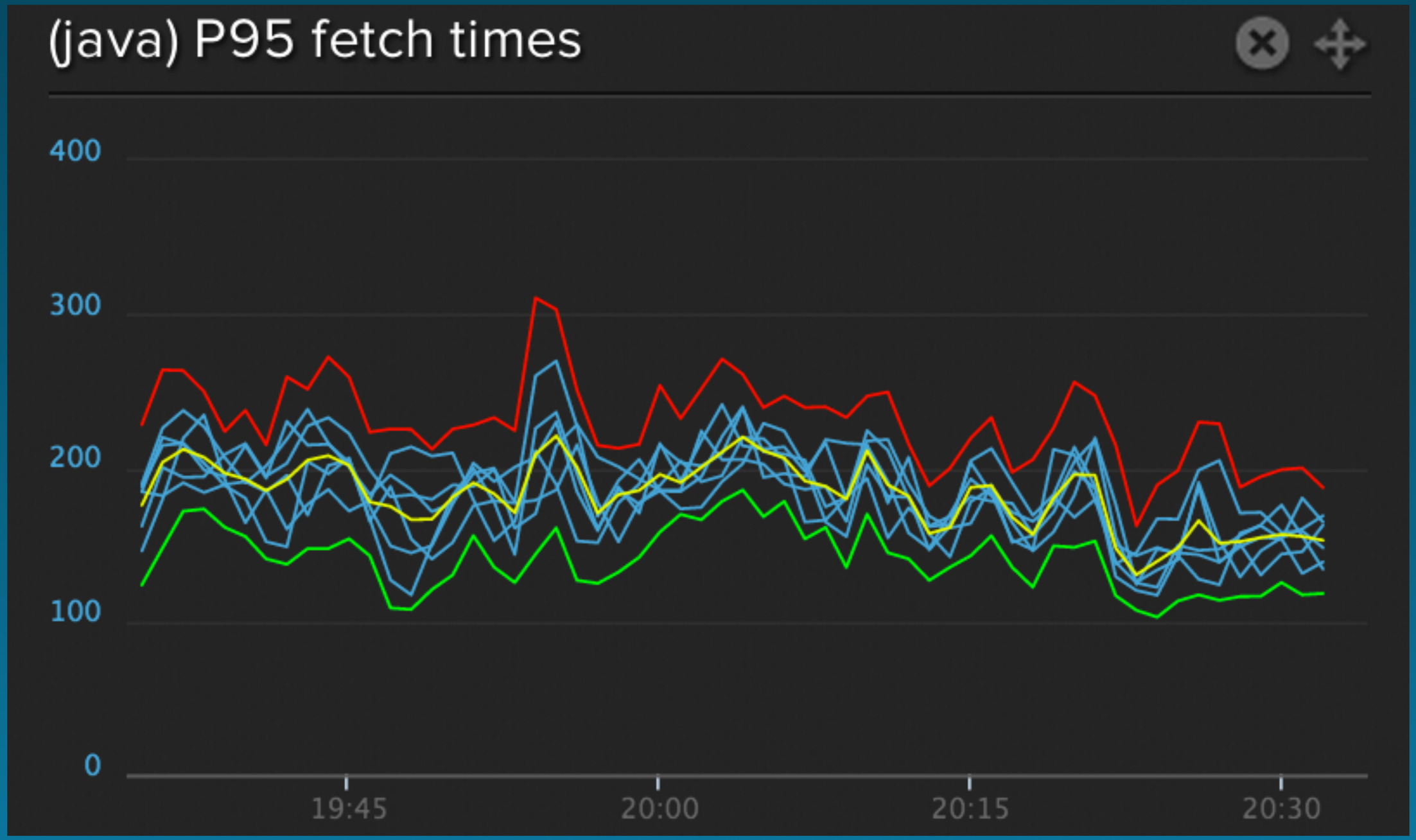
Pattern3: The Rollup Pattern

Complicated, and therefore
Hard to reason about, and therefore
Hard to Maintain

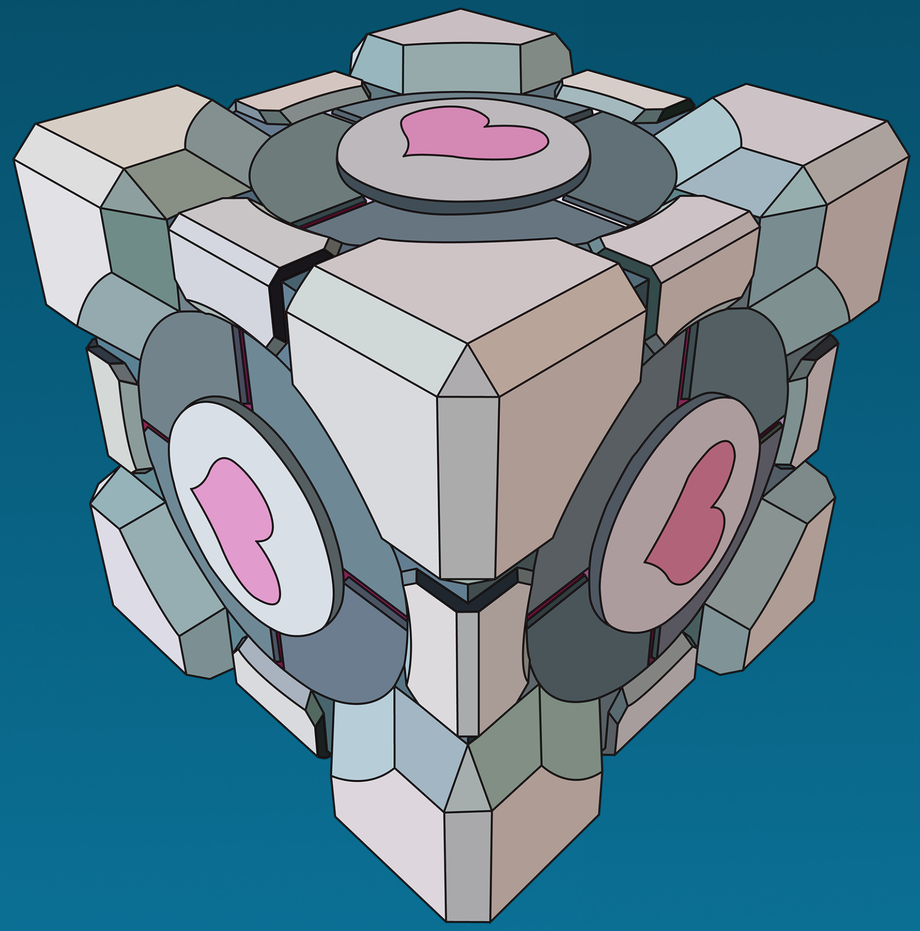


That’s a server right? —>



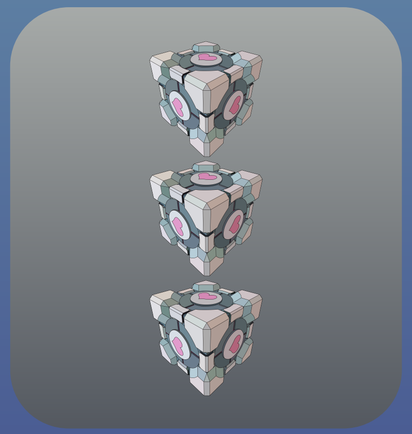

(java) P95 fetch times
400
300
200
100
0
19:45
20:00
20:15
20:30

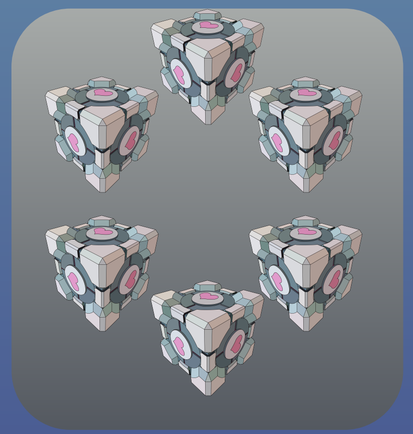
That’s a thread —>



Auto-Scaling Group 1b-17 east

instance-eaa6238ad5ff00





instance-d3b07384d11

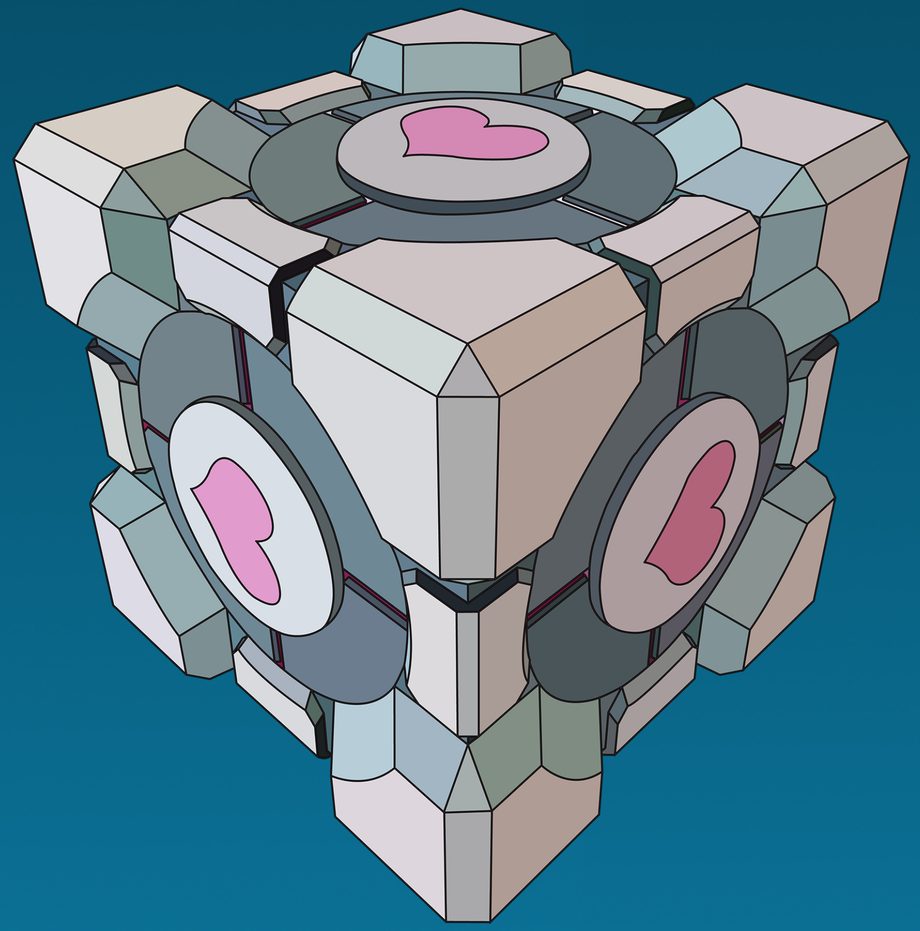
instance-3edec49

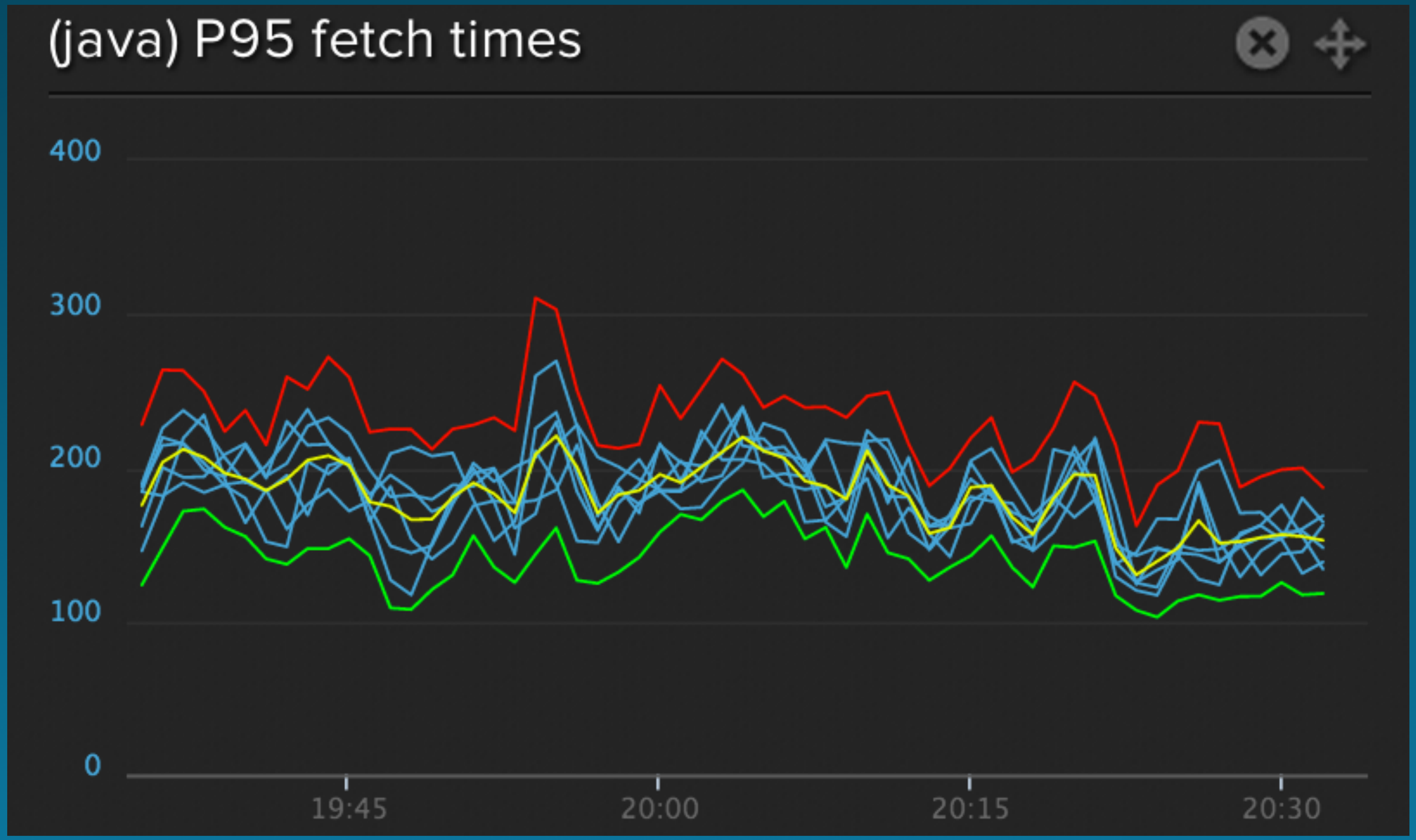
instance-d3b07384d11

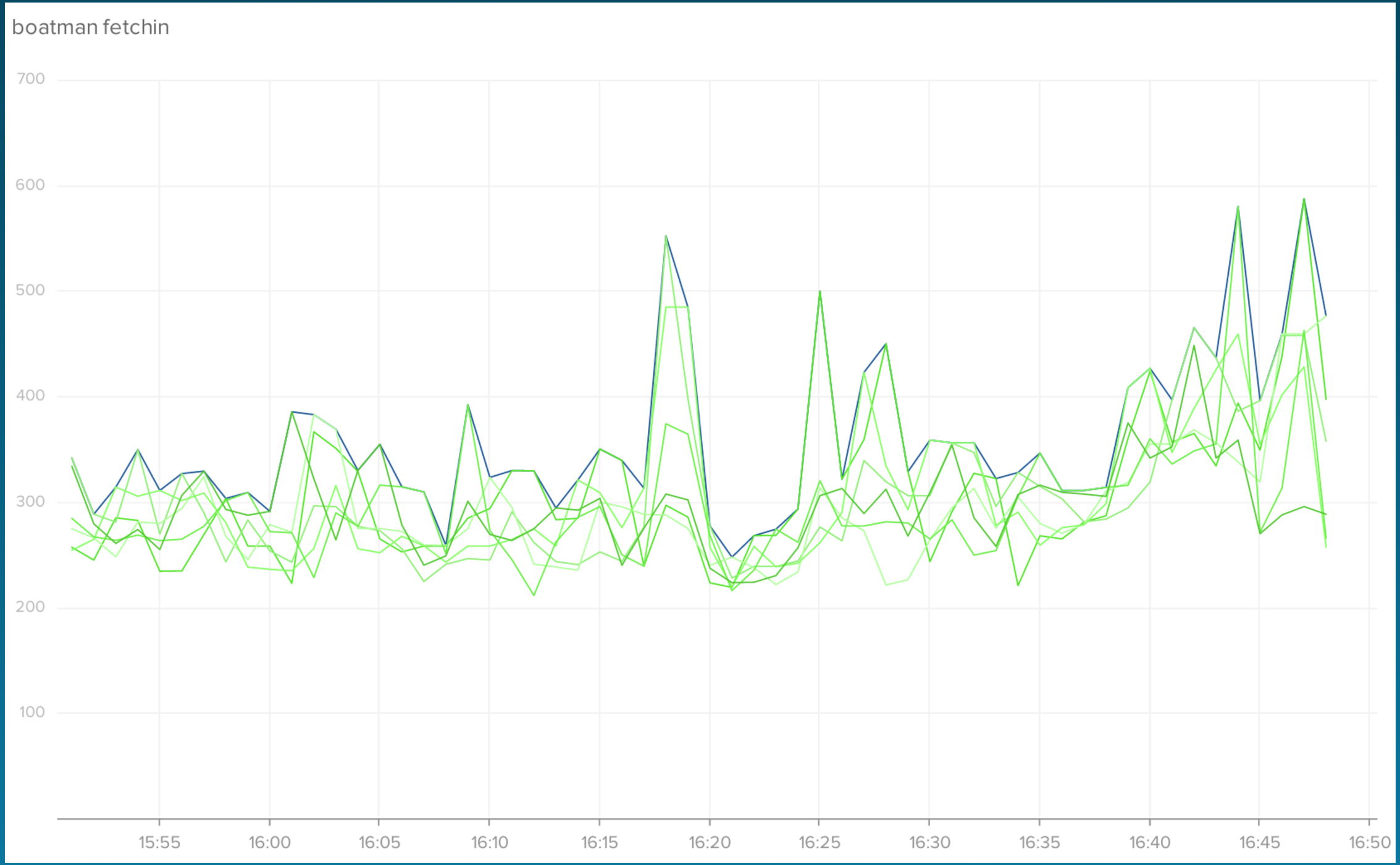
instance-3edec49

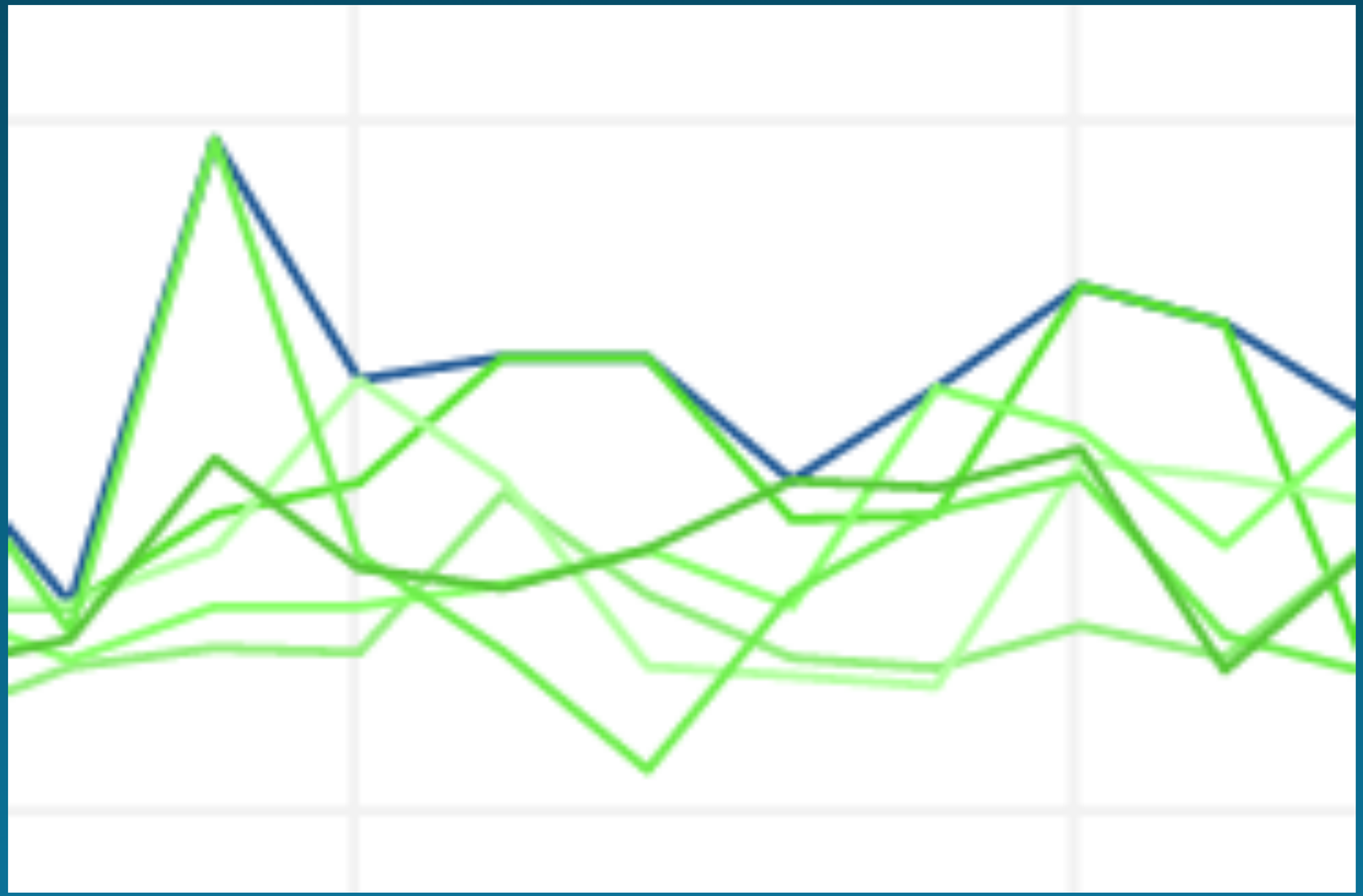
ZOMG IT'S 4:00 ON FRIDAY! FINISH ALL THE WORKS!!

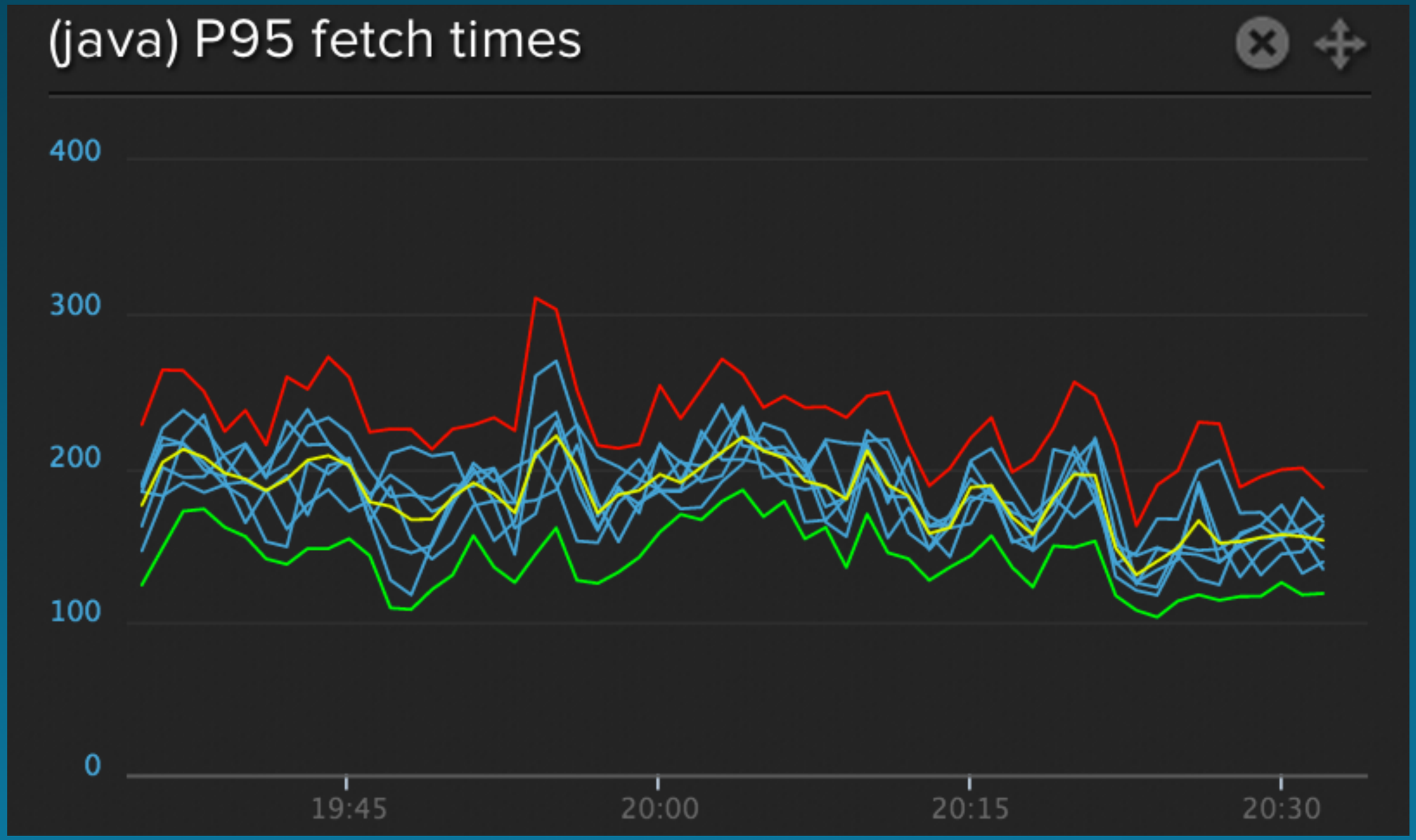
That already doesn’t exist anymore —>



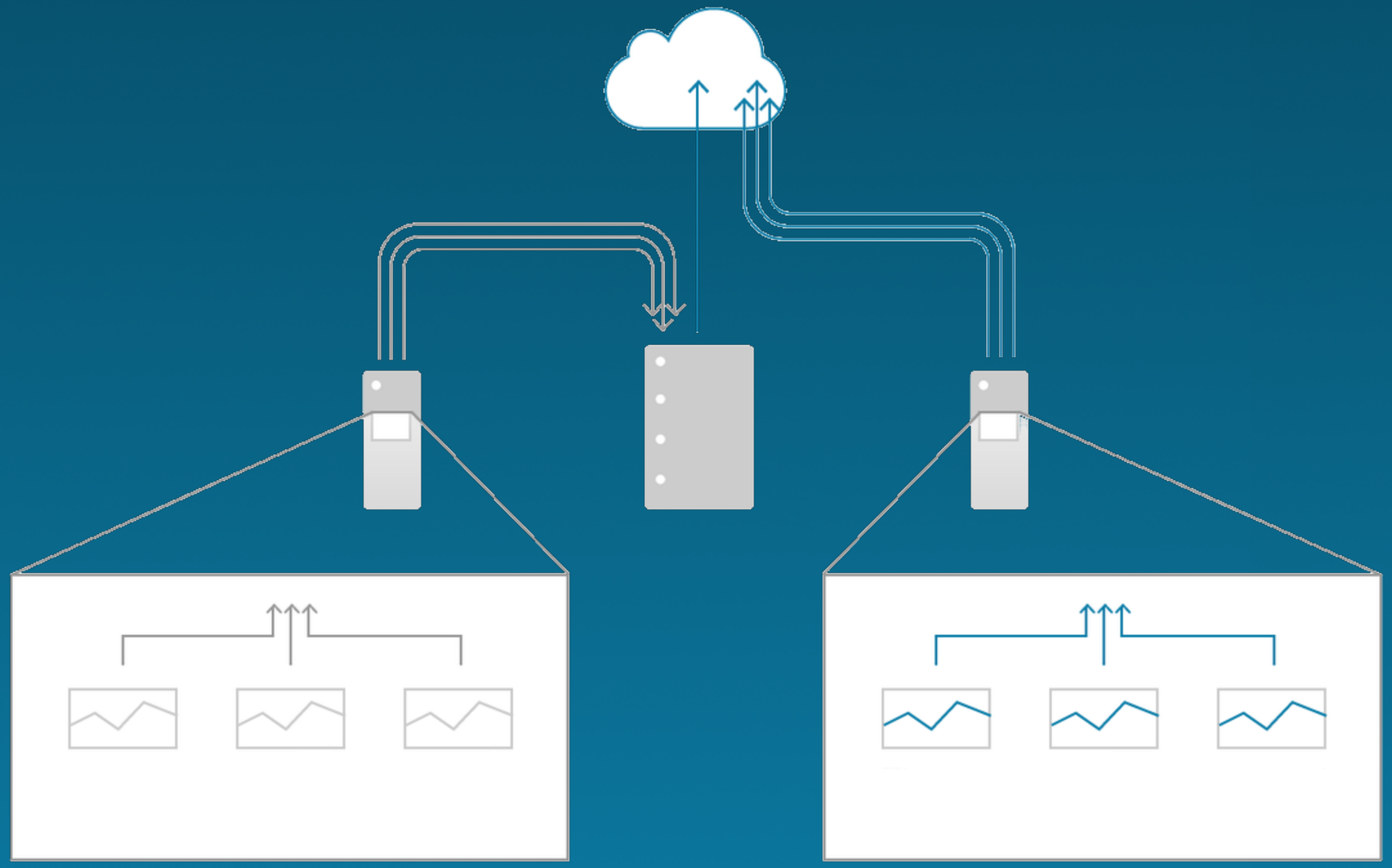

(java) P95 fetch times
400
300
200
100
0
19:45
20:00
20:15
20:30


boatman fetchin
700
600
500
400
300
200
100
15:55
16:00
16:05
16:10
16:15
16:20
16:25
16:30
16:35
16:40
16:45
16:50

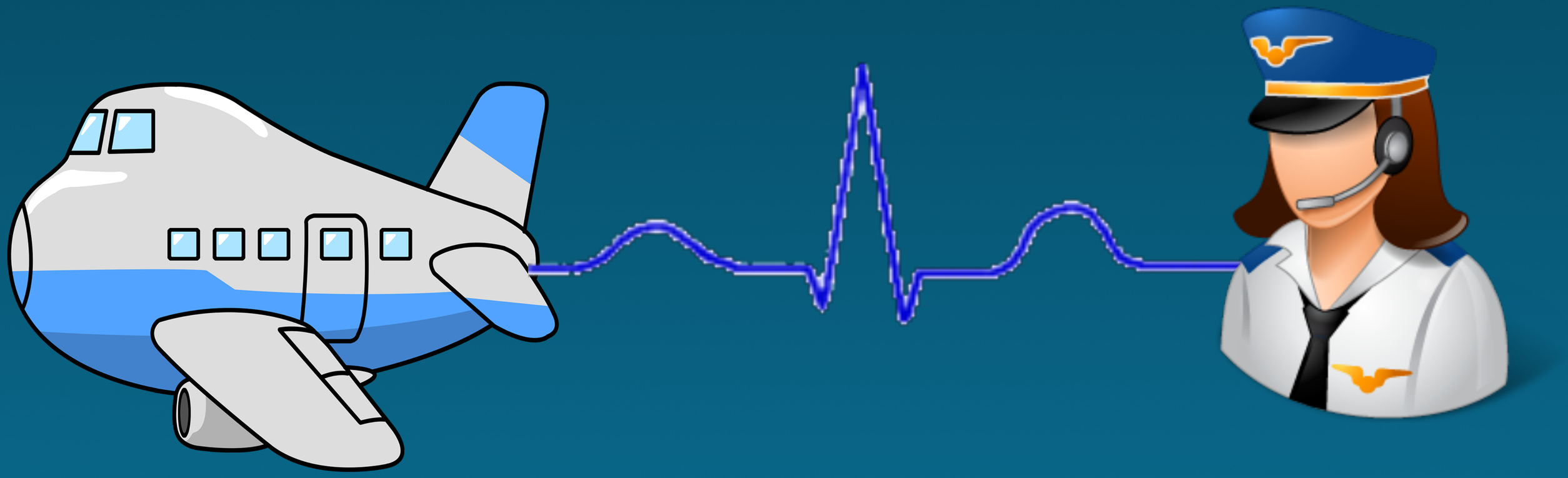



(java) P95 fetch times
400
300
200
100
0
19:45
20:00
20:15
20:30

Pattern 4: Emitter/Reporter



Instrumentation as code

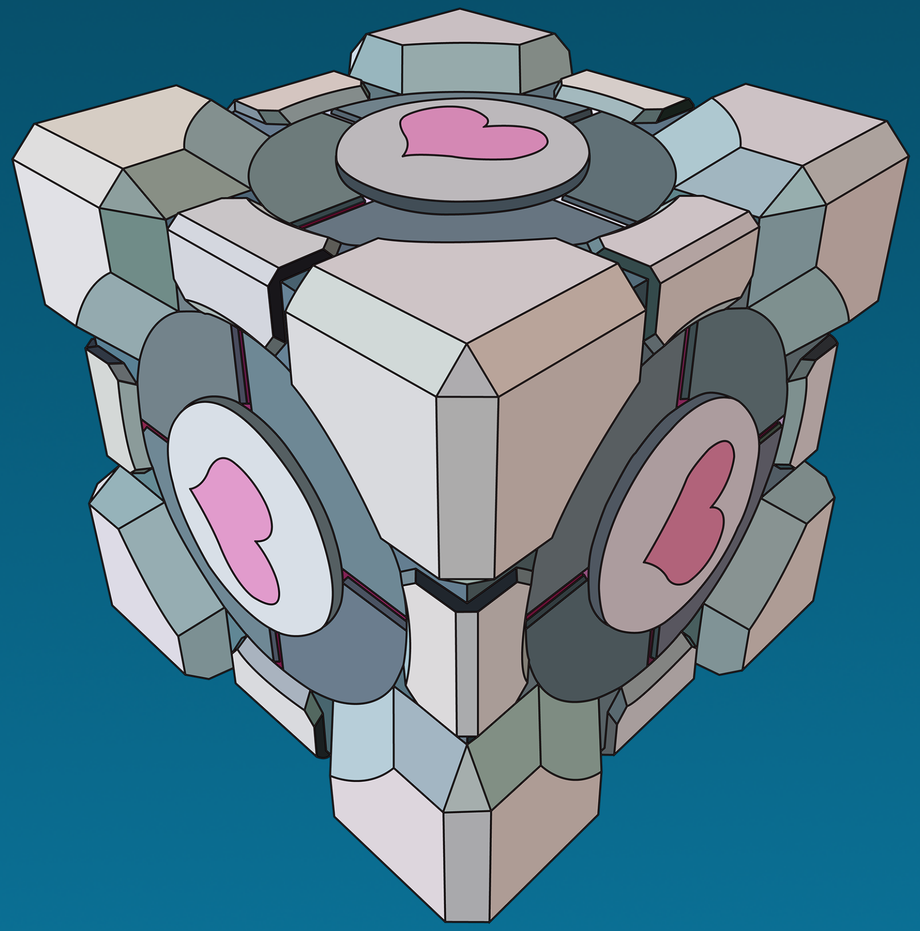


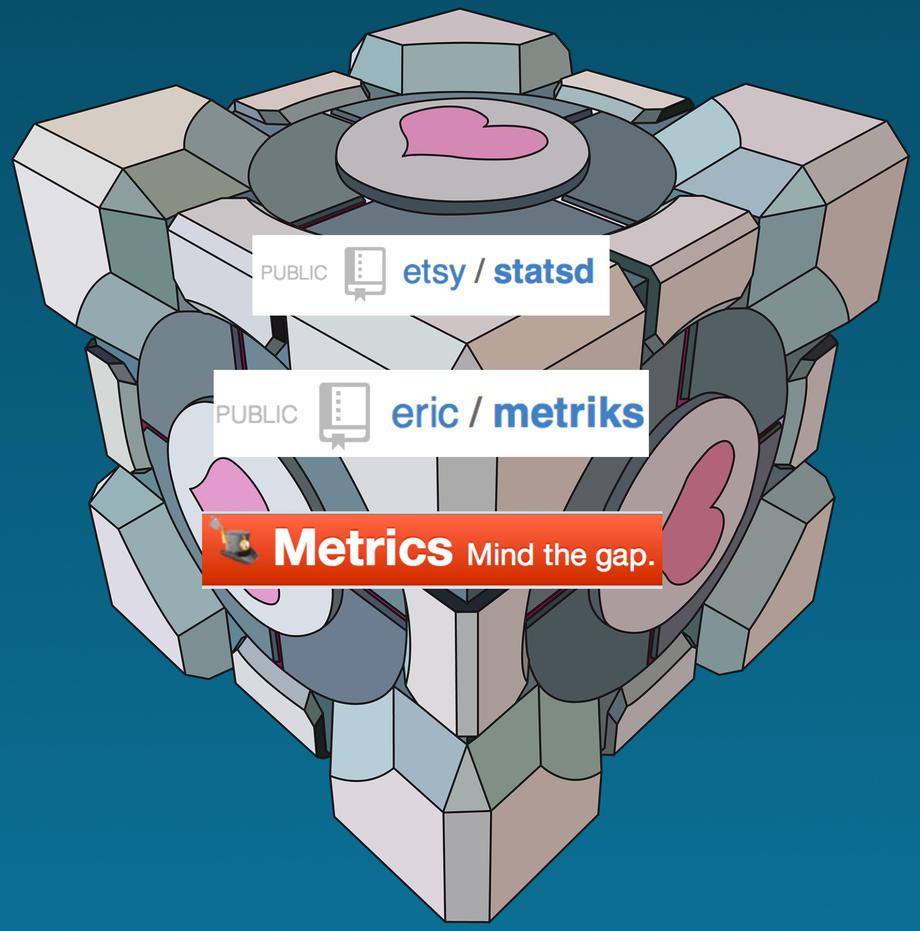
```
output, _ := daves.function(input)
```

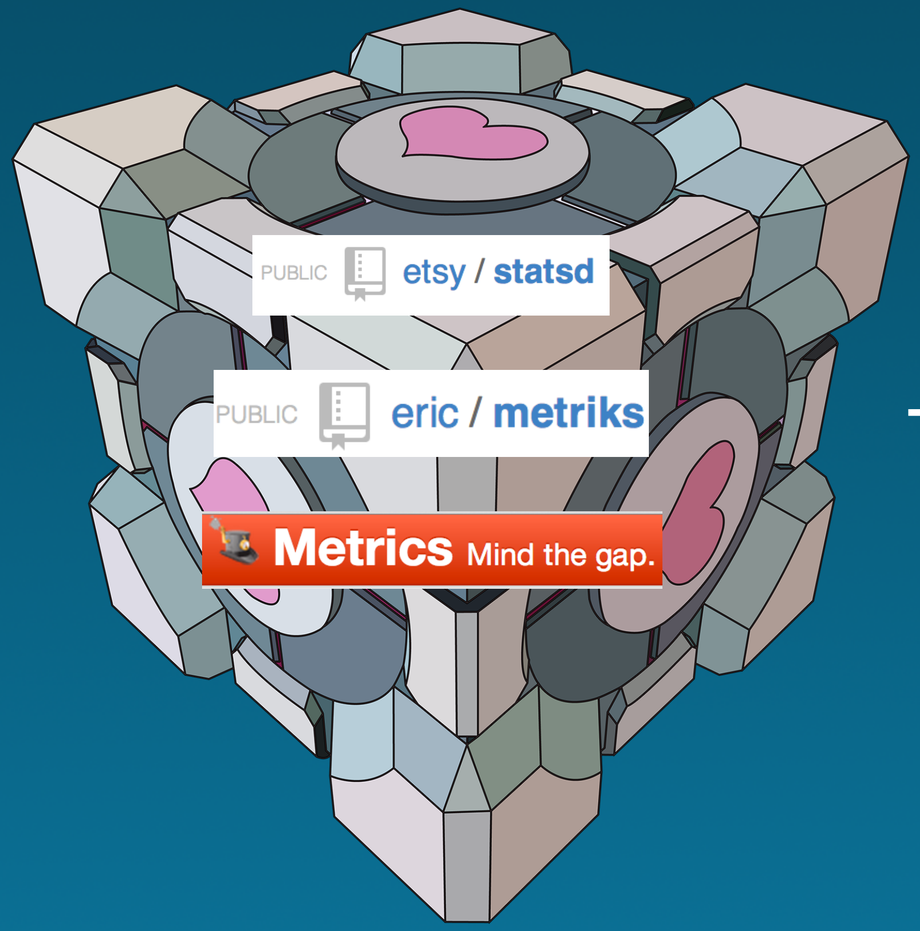
```
metric.time( output, _ := daves.function(input))
```

```
func handleInput(inBytes *[]byte, type string){

    if type == “A” {
        handleTypeA(inBytes)

    } else if type == “B” {
        handleTypeB(inBytes)

    }else if type == “C” {
        handleTypeC(inBytes)

    }
}
```

```
func handleInput(inBytes *[]byte, type string){

    if type == “A” {
        handleTypeA(inBytes)
        aCounter.Inc(1)
    } else if type == “B” {
        handleTypeB(inBytes)
        bCounter.Inc(1)
    }else if type == “C” {
        handleTypeC(inBytes)
        cCounter.Inc(1)
    }
}
```




PUBLIC
etsy / statsd
PUBLIC
eric / metriks
Metrics Mind the gap.

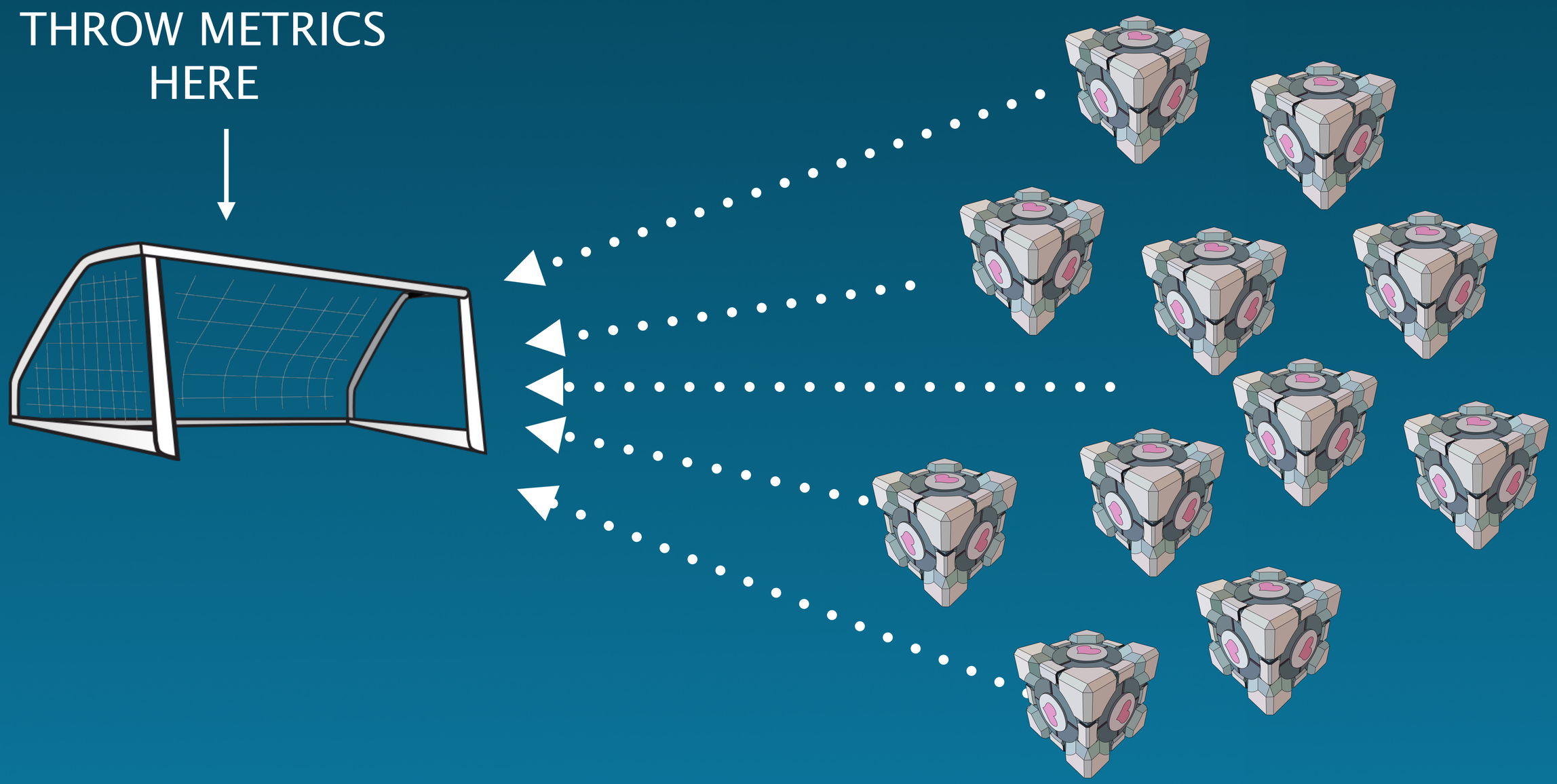


RRDtool
logging & graphing

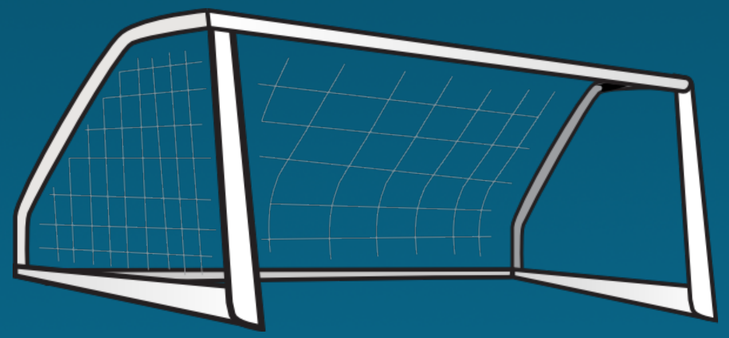
```
rrdtool create target.rrd \
        --start 1023654125 \
            --step 300 \
    DS:mem:GAUGE:600:0:671744 \
      RRA:AVERAGE:0.5:12:24 \
       RRA:AVERAGE:0.5:288:31
```

```
[root@YHG-I6-05-xs-back06 rra]# rrdtool fetch /var/www/cacti/rra/localhost_listen_21.rrd AVERAGE
                        listen          established          time_wait          fin_wait1          syn_se
recv

1407144960: -nan -nan -nan -nan -nan -nan -nan -nan
1407145320: -nan -nan -nan -nan -nan -nan -nan -nan
1407145680: -nan -nan -nan -nan -nan -nan -nan -nan
1407146040: -nan -nan -nan -nan -nan -nan -nan -nan
1407146400: -nan -nan -nan -nan -nan -nan -nan -nan
1407146760: -nan -nan -nan -nan -nan -nan -nan -nan
1407147120: -nan -nan -nan -nan -nan -nan -nan -nan
1407147480: -nan -nan -nan -nan -nan -nan -nan -nan
1407147840: -nan -nan -nan -nan -nan -nan -nan -nan
1407148200: -nan -nan -nan -nan -nan -nan -nan -nan
1407148560: -nan -nan -nan -nan -nan -nan -nan -nan
1407148920: -nan -nan -nan -nan -nan -nan -nan -nan
1407149280: -nan -nan -nan -nan -nan -nan -nan -nan
1407149640: -nan -nan -nan -nan -nan -nan -nan -nan
1407150000: -nan -nan -nan -nan -nan -nan -nan -nan
1407150360: -nan -nan -nan -nan -nan -nan -nan -nan
1407150720: -nan -nan -nan -nan -nan -nan -nan -nan
1407151080: -nan -nan -nan -nan -nan -nan -nan -nan
```

BOB.SOMEWHERE.COM
RRDtool
logging & graphing

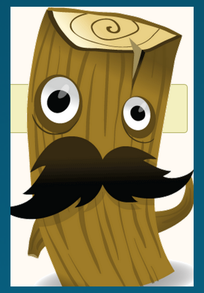

THROW METRICS
HERE

THROW METRICS HERE



Metrics Mind the gap.

fluentd



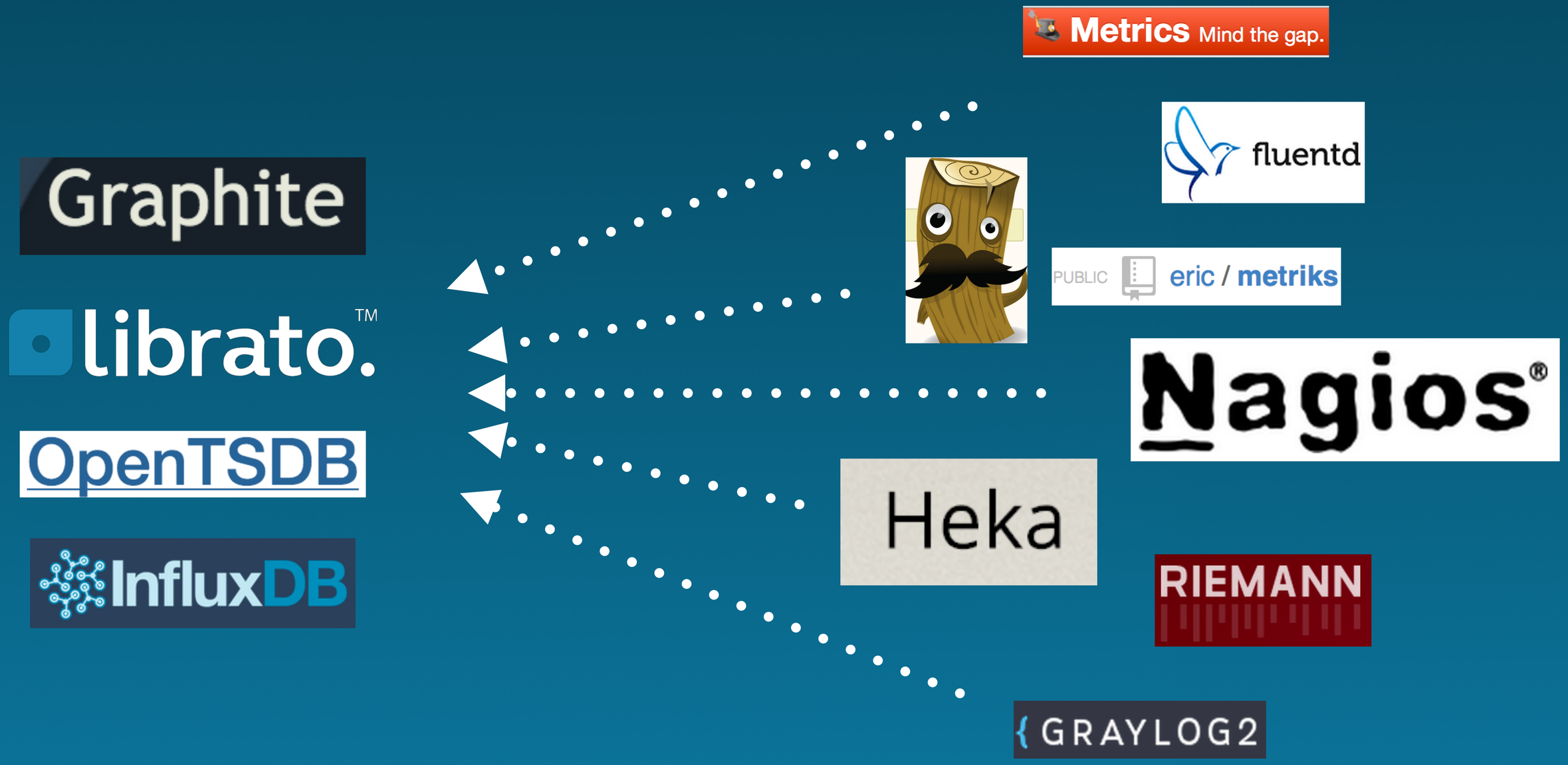
PUBLIC eric / metriks

Nagios®

Heka

RIEMANN

{ GRAYLOG2


Metrics Mind the gap.
fluentd
Graphite
PUBLIC
eric / metriks
librato.™
Nagios®
OpenTSDB
Heka
InfluxDB
RIEMANN
GRAYLOG2

Graphite

InfluxDB

OpenTSDB

librato™

Nagios®

Graphite

InfluxDB

librato.™

OpenTSDB

http://github.com/shawn-sterling/graphios

Nagios®

↓

shawn-sterling / graphios

→

Graphite

InfluxDB

librato.™

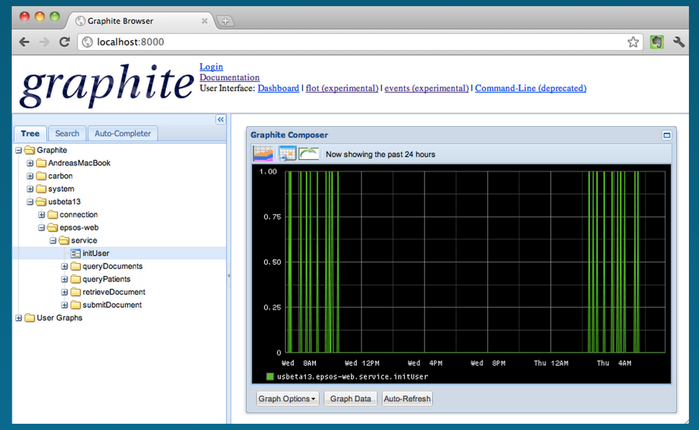
OpenTSDB

Graphite

Graphite

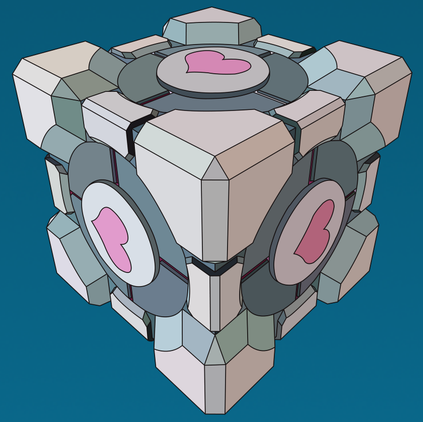
Carbon

Whisper



Graphite

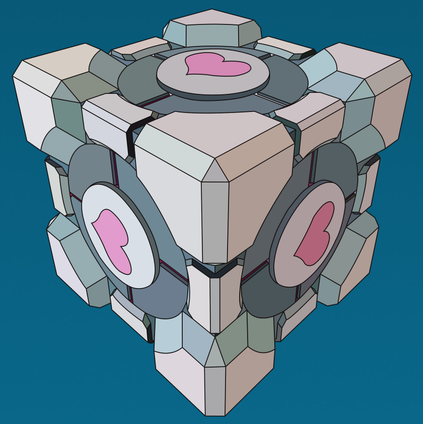
Graphite



2003/TCP

Carbon

Graphite



thing.cpu.load .021 1412366606

2003/TCP

Carbon

format:
<metric path>SPACE<metric value>SPACE<metric timestamp>

Graphite

2003/TCP

06

Carbon

Whisper

Graphite

Whisper

Graphite

Whisper

Graphite

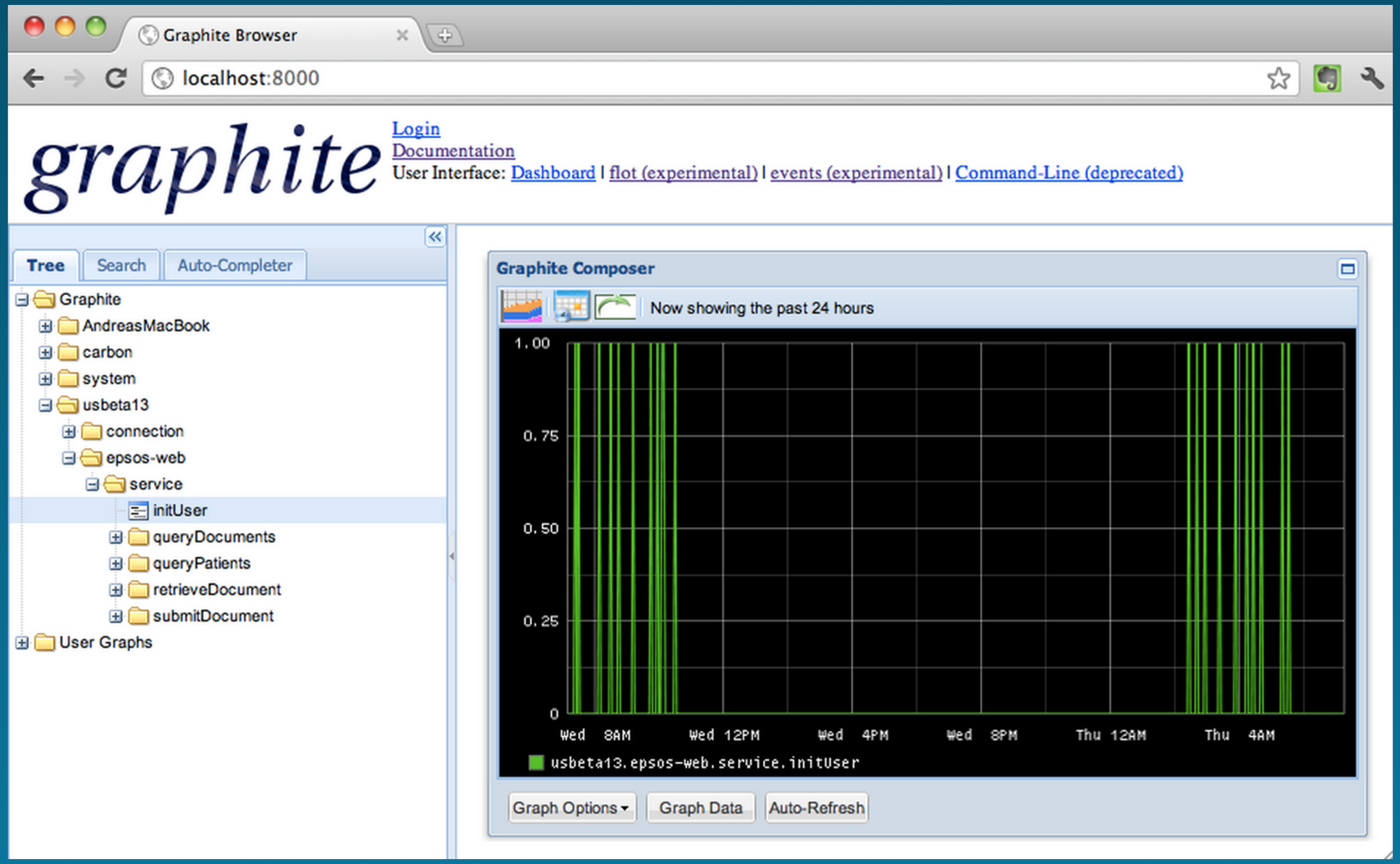
Whisper

pyr / cyanite

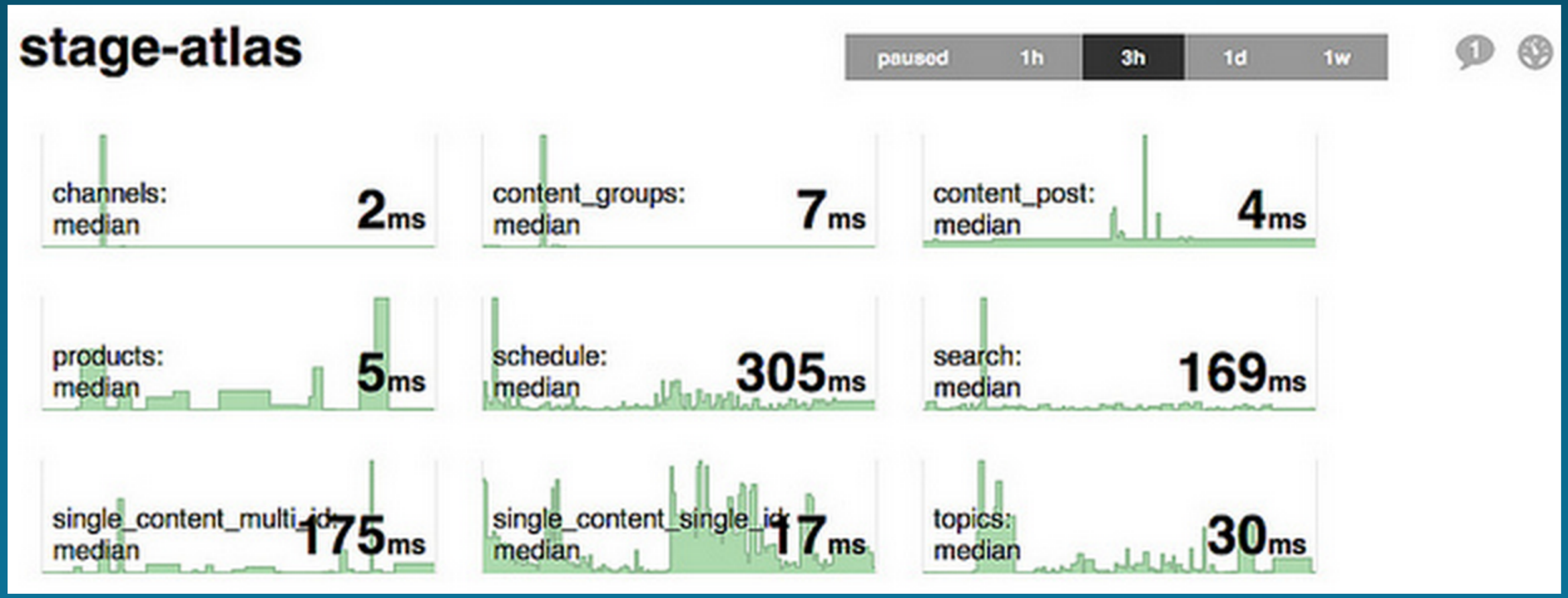
Cassandra

Graphite

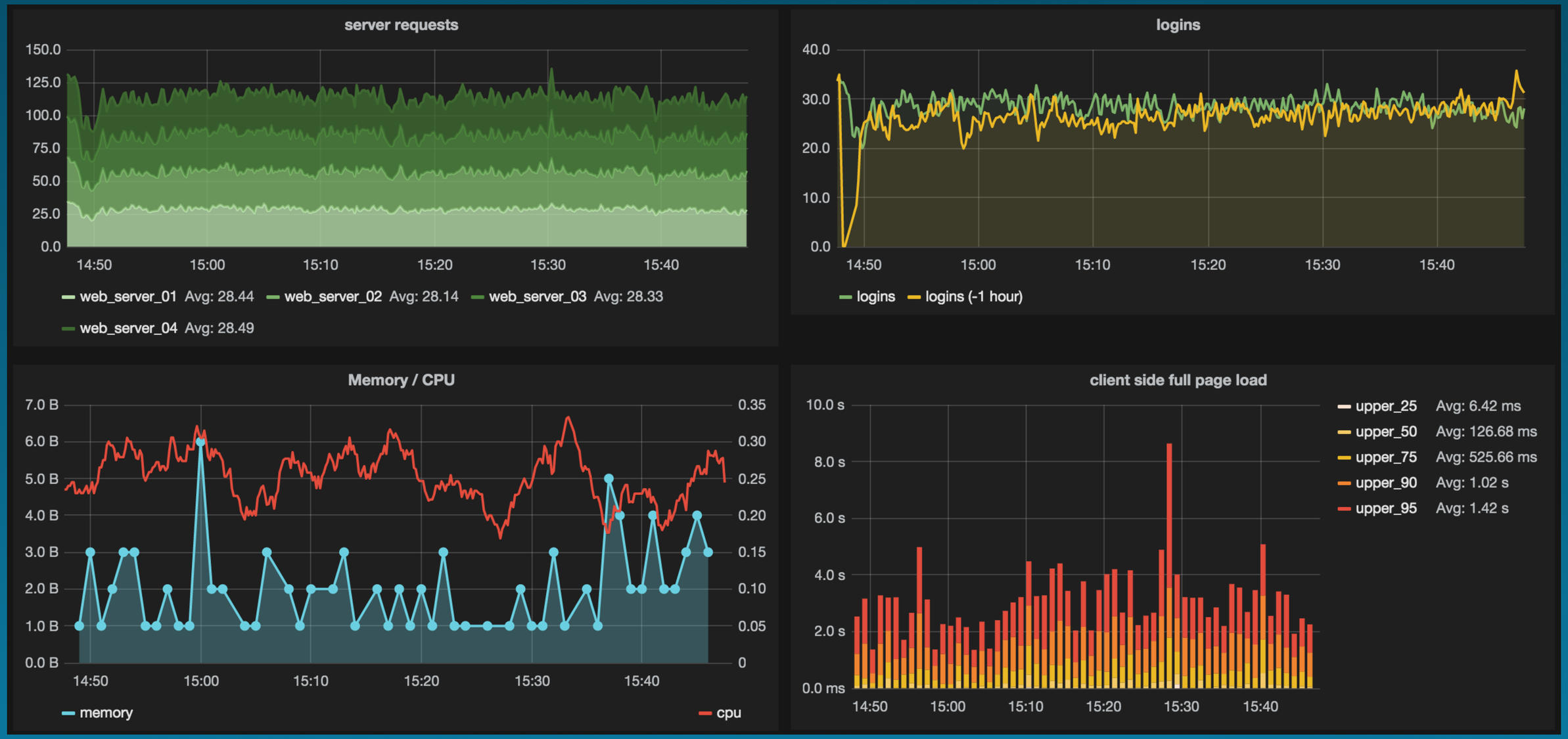
Whisper



obfuscurity / tasseo



http://play.grafana.org



Graphite Woes

Mitch Garnaat @garnaat · 7h

InfluxDB or Graphite?

Reply Retweet Favorite Buffer More

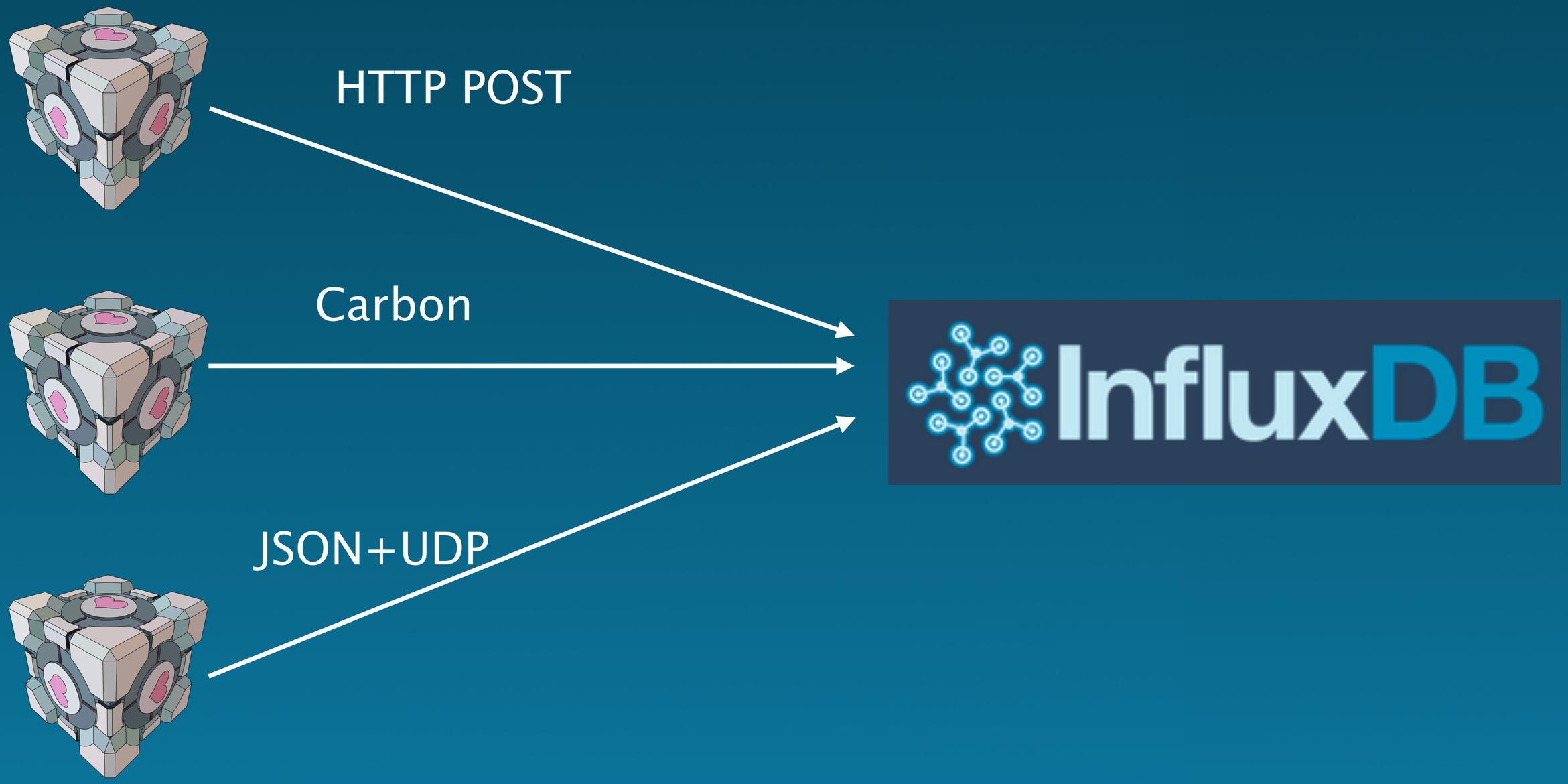
Greg Taylor

@gctaylor

Follow

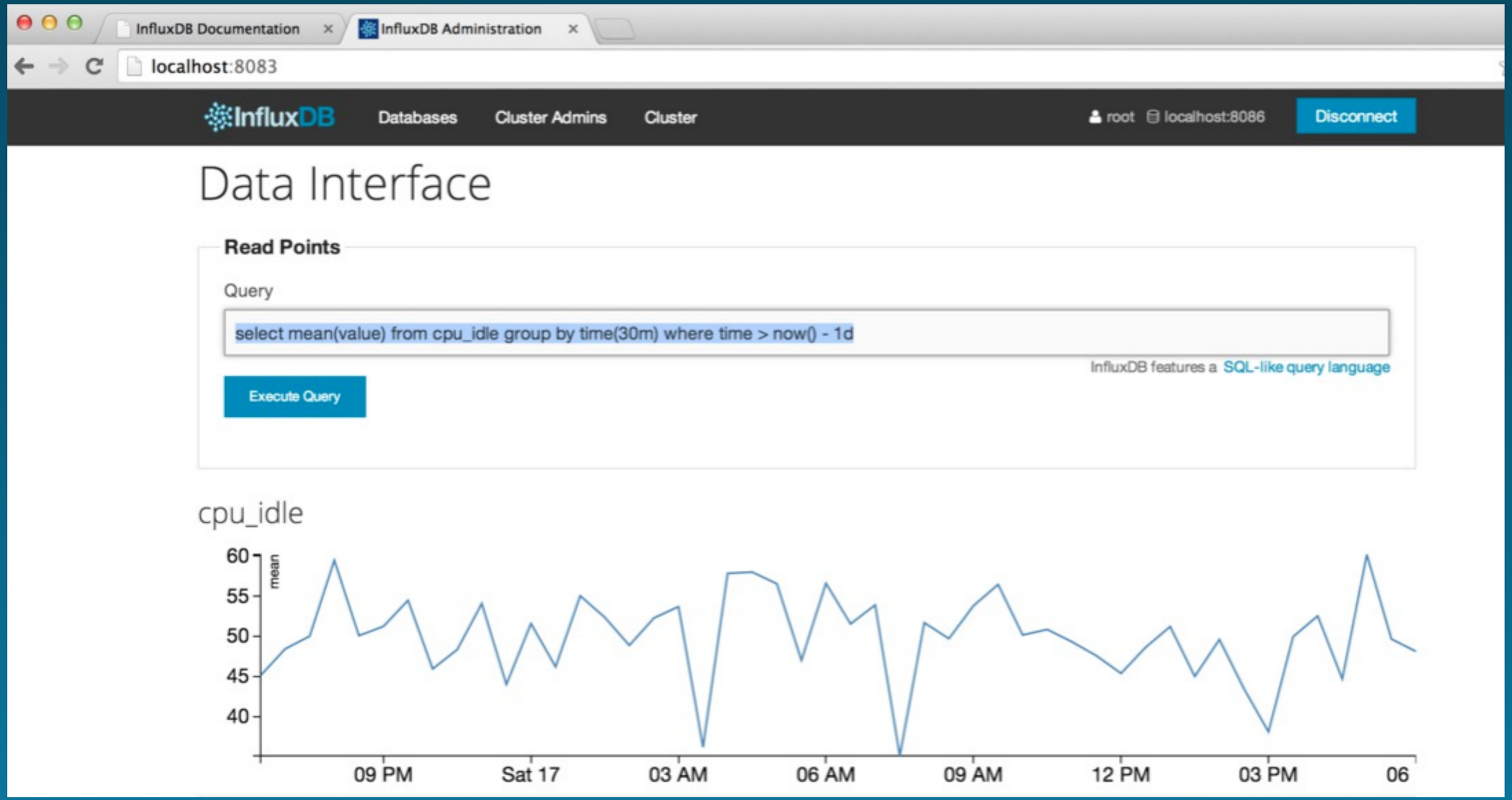
@garnaat Librato, so you don't have to maintain a graphite cluster :)

InfluxDB

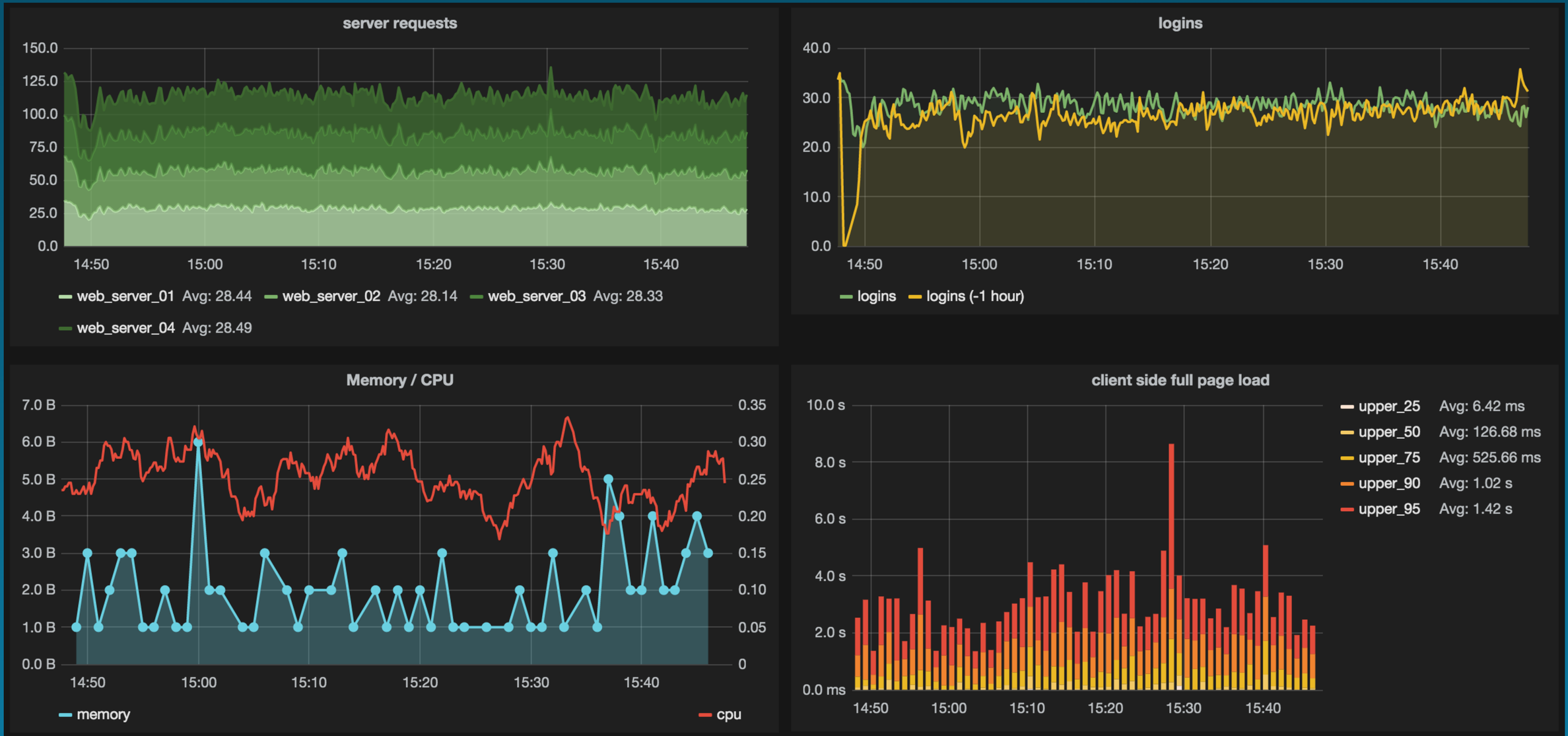

HTTP POST
Carbon
JSON+UDP
InfluxDB

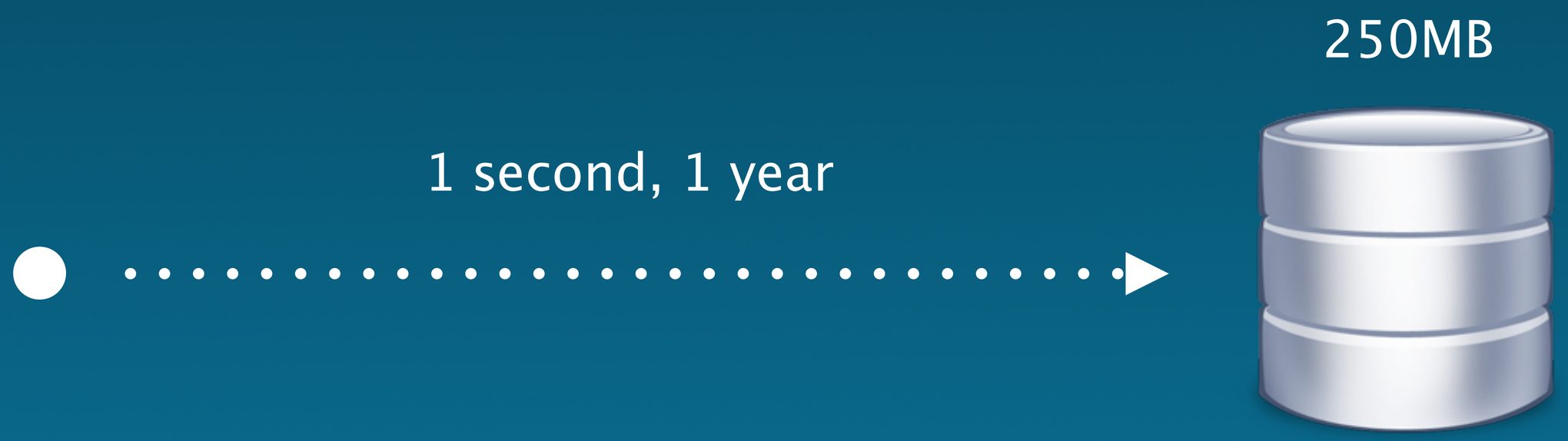
LevelDB, RocksDB, HyperLevelDB, or LMDB

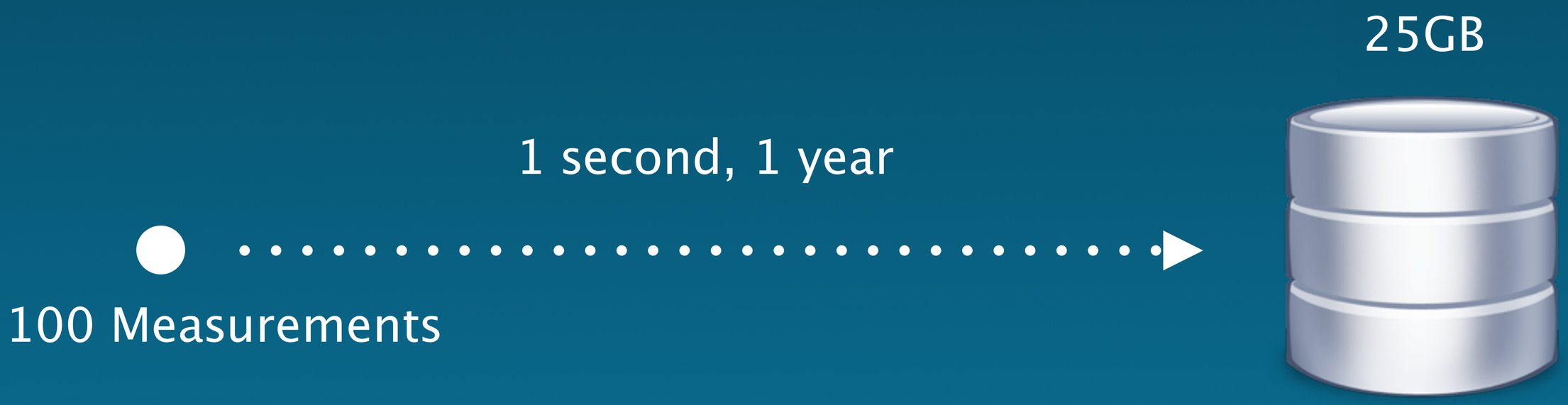
```
select value from response_times
where time > '2013-08-12 23:32:01.232' and
time < '2013-08-13';
```

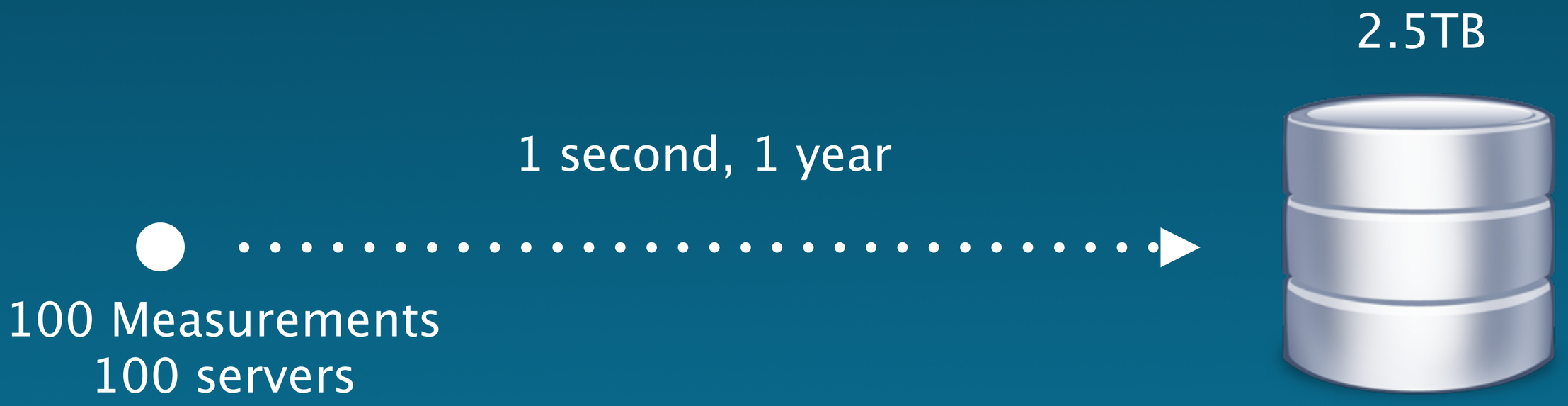

InfluxDB Documentation
InfluxDB Administration
localhost:8083
InfluxDB
Databases
Cluster Admins
Cluster
root
localhost:8086
Disconnect
Data Interface
Read Points
Query
select mean(value) from cpu_idle group by time(30m) where time > now() - 1d
InfluxDB features a SQL-like query language
Execute Query
cpu_idle
mean
60
55
50
45
40
09 PM
Sat 17
03 AM
06 AM
09 AM
12 PM
03 PM
06

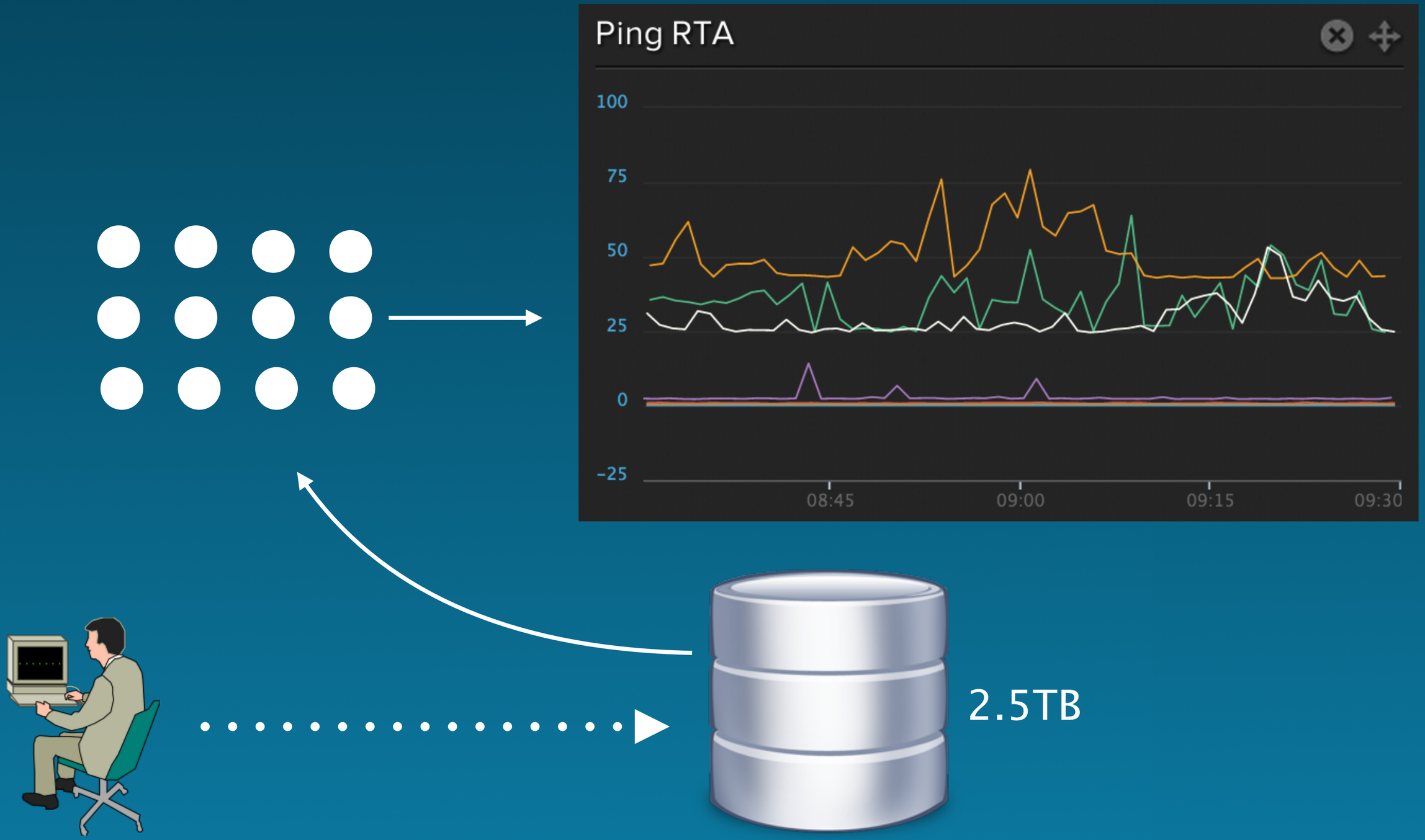
InfluxDB


server requests
150.0
125.0
100.0
75.0
50.0
25.0
0.0
14:50
15:00
15:10
15:20
15:30
15:40
web_server_01 Avg: 28.44
web_server_02 Avg: 28.14
web_server_03 Avg: 28.33
web_server_04 Avg: 28.49
logins
40.0
30.0
20.0
10.0
0.0
14:50
15:00
15:10
15:20
15:30
15:40
logins
logins (-1 hour)
Memory / CPU
7.0 B
6.0 B
5.0 B
4.0 B
3.0 B
2.0 B
1.0 B
0.0 B
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0
14:50
15:00
15:10
15:20
15:30
15:40
memory
cpu
client side full page load
10.0 s
8.0 s
6.0 s
4.0 s
2.0 s
0.0 ms
14:50
15:00
15:10
15:20
15:30
15:40
upper_25 Avg: 6.42 ms
upper_50 Avg: 126.68 ms
upper_75 Avg: 525.66 ms
upper_90 Avg: 1.02 s
upper_95 Avg: 1.42 s

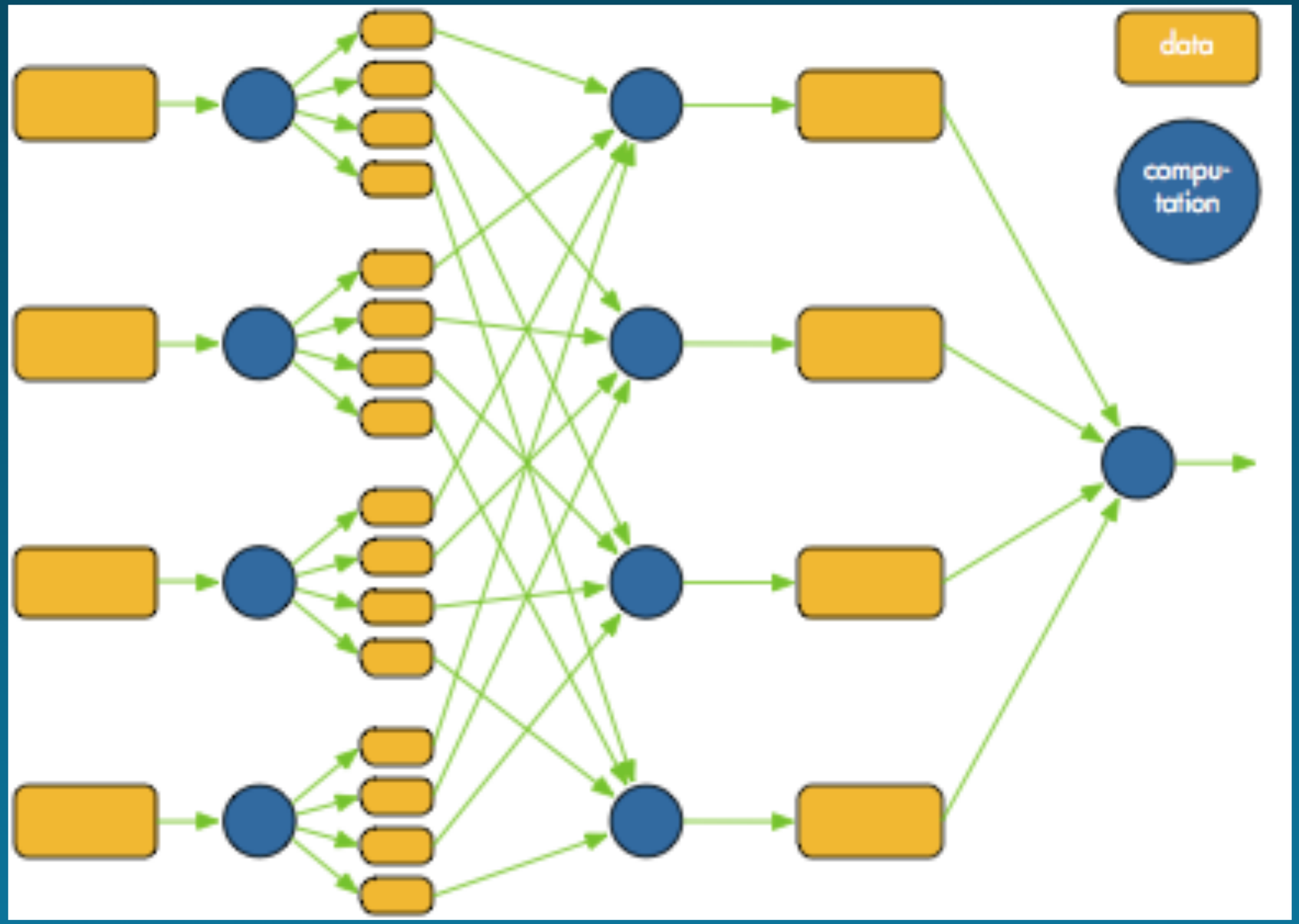

250MB
1 second, 1 year


25GB
1 second, 1 year
100 Measurements

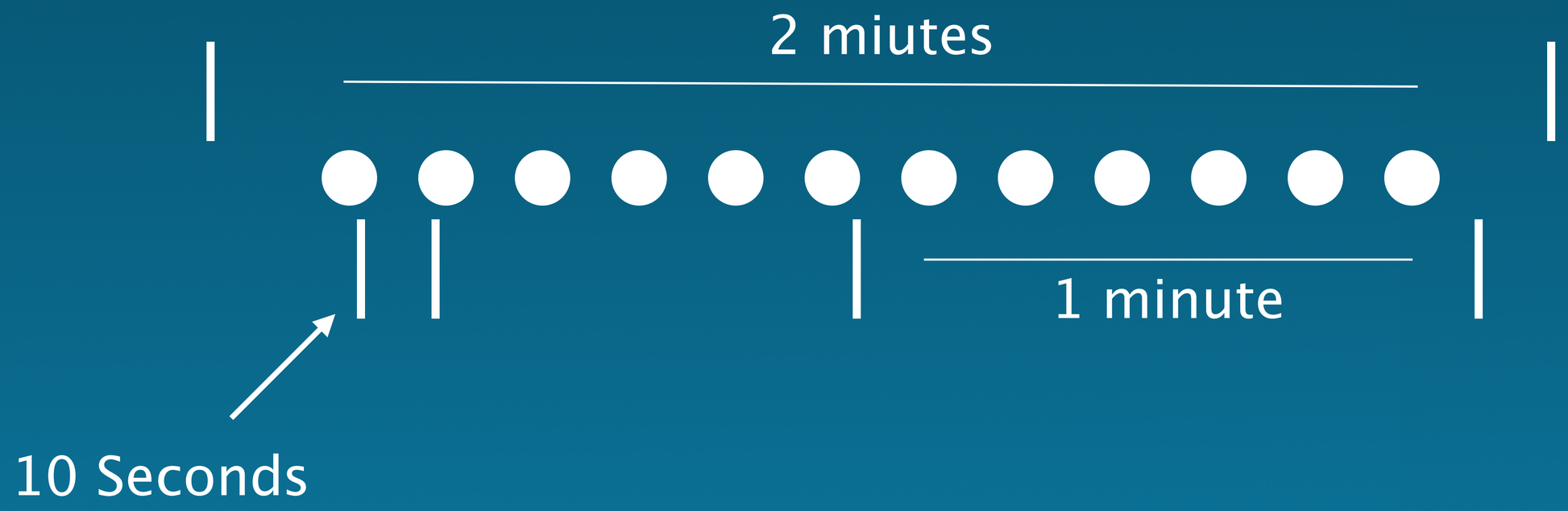

2.5TB
1 second, 1 year
100 Measurements
100 servers


Ping RTA
100
75
50
25
0
-25
08:45
09:00
09:15
09:30
2.5TB

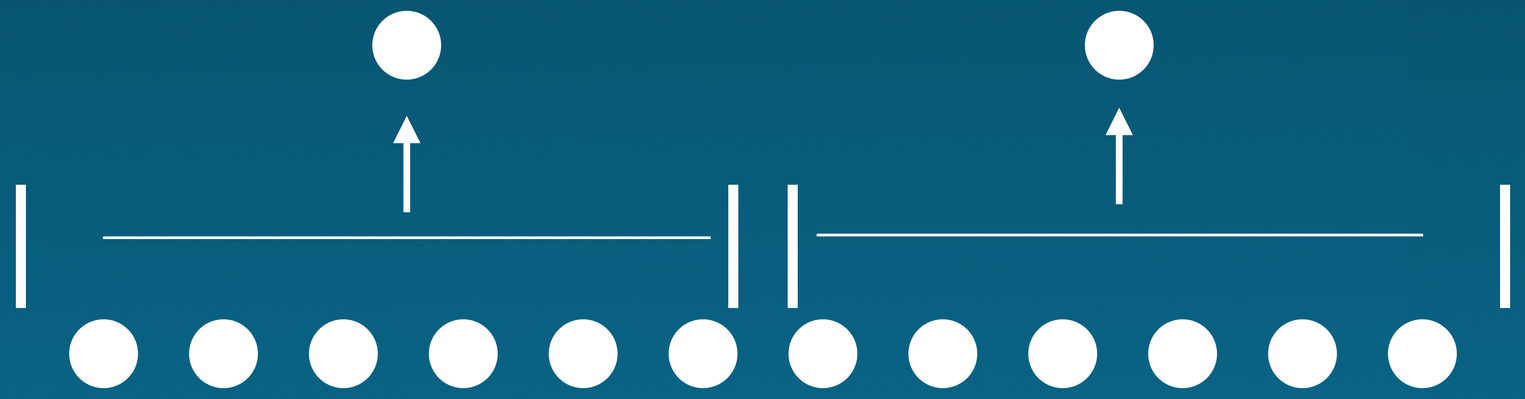
U NEED BIG DATA?
WAIT, I GET IT 4 U


data
compu-
tation

• • • • • • • • • • • •


2 miutes
1 minute
10 Seconds

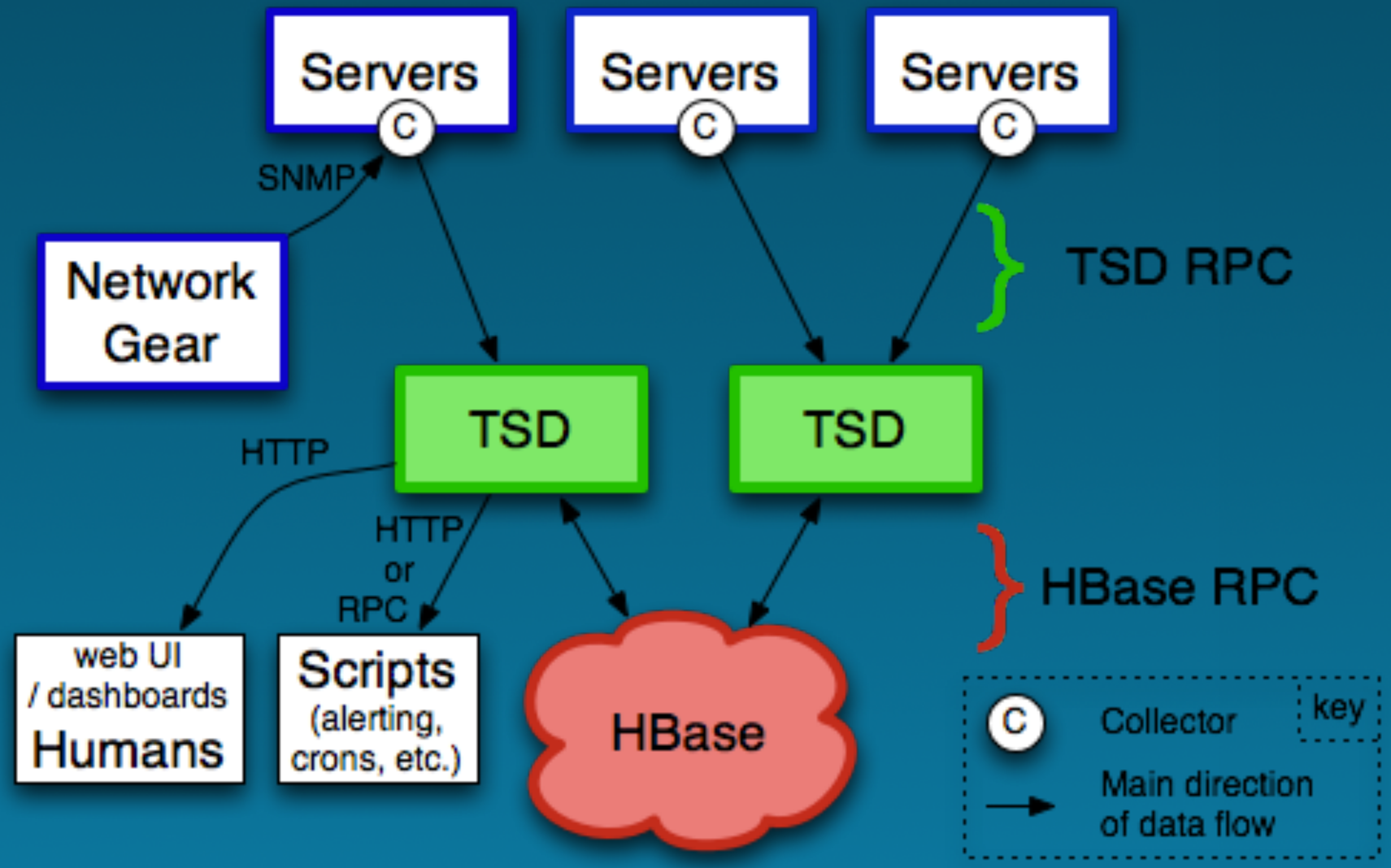
(stored for 1 year)
1-minute resolution data

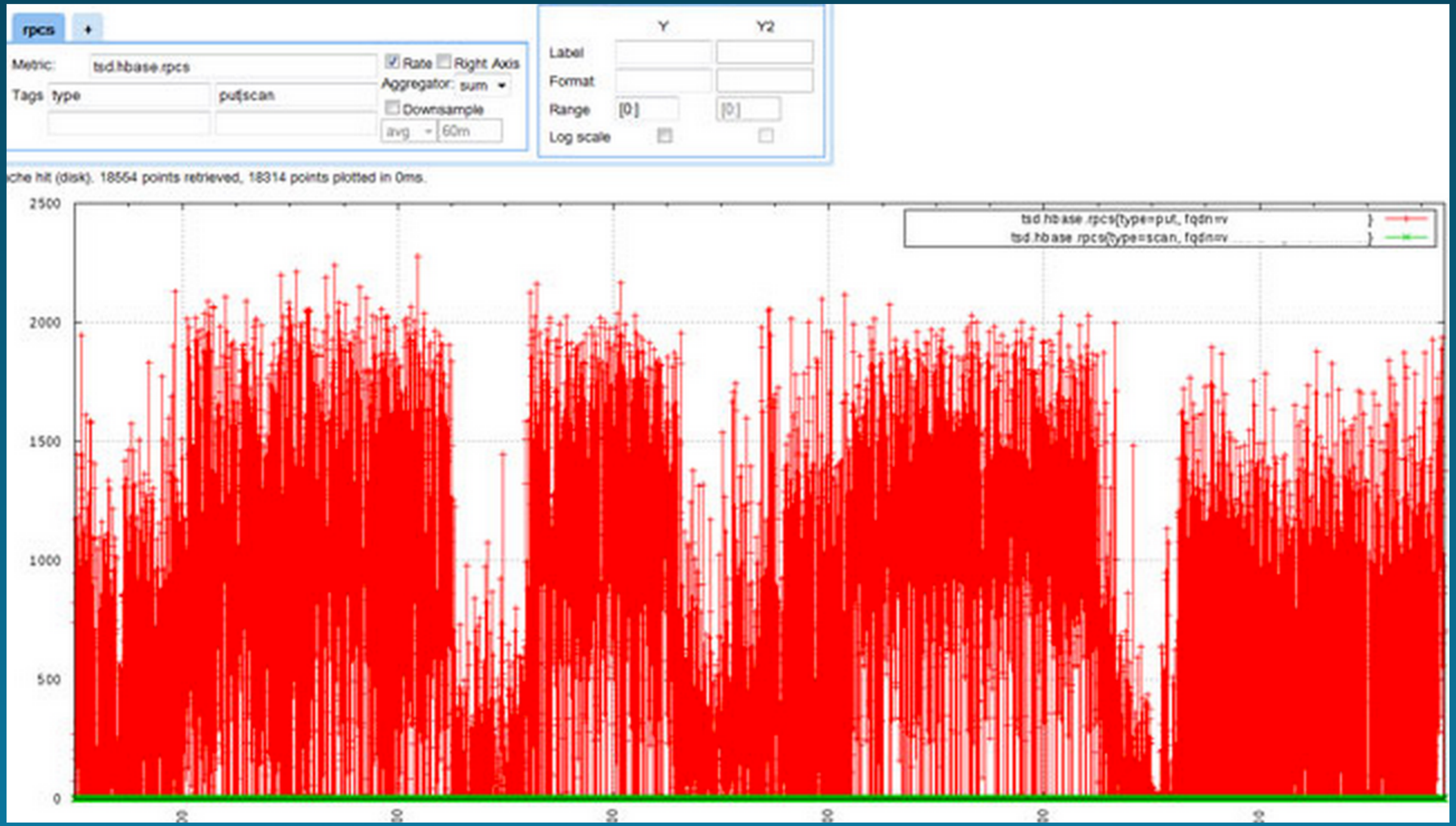


10-second resolution data
(stored for 24 hours)

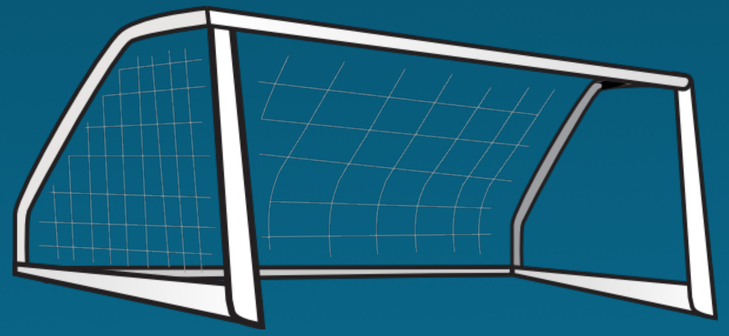
OpenTSDB

OpenTSDB


Servers
C
Servers
C
Servers
C
SNMP
Network Gear
TSD RPC
TSD
TSD
HTTP
HTTP or RPC
HBase RPC
web UI / dashboards
Humans
Scripts
(alerting, crons, etc.)
HBase
key
C
Collector
Main direction of data flow

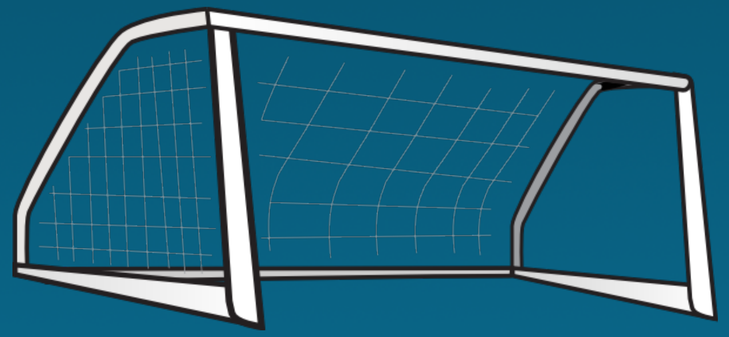

rpcs
+
Metric: tsd.hbase.rpcs
Tags type put|scan
Rate
Right Axis
Aggregator: sum
Downsample
avg
60m
Y
Y2
Label
Format
Range [0:]
Log scale
che hit (disk). 18564 points retrieved, 18314 points plotted in 0ms.
2500
2000
1500
1000
500
0
tsd.hbase.rpcs{type=put, fqdn=v
tsd.hbase.rpcs{type=scan, fqdn=v

librato™



librato.™

← shawn-sterling / **graphios** ← Nagios®



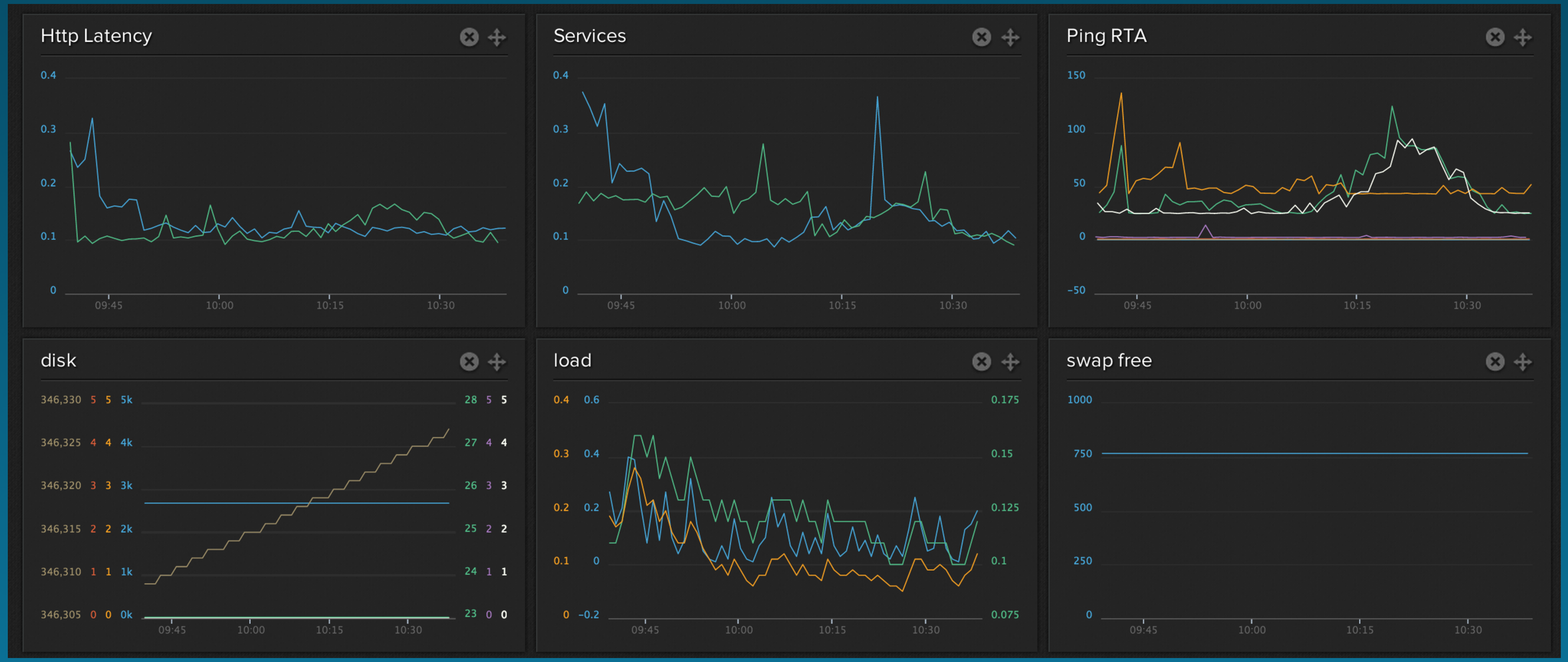
librato.™

1-minute, 10cents/month
5-minute, 5cents/month

shawn-sterling / graphios

Whitelist metrics here

Nagios®


Http Latency
0.4
0.3
0.2
0.1
0
09:45
10:00
10:15
10:30
Services
0.4
0.3
0.2
0.1
0
09:45
10:00
10:15
10:30
Ping RTA
150
100
50
0
-50
09:45
10:00
10:15
10:30
disk
346,330 5 5 5k
346,325 4 4 4k
346,320 3 3 3k
346,315 2 2 2k
346,310 1 1 1k
346,305 0 0 0k
28 5 5
27 4 4
26 3 3
25 2 2
24 1 1
23 0 0
09:45
10:00
10:15
10:30
load
0.4 0.6
0.3 0.4
0.2 0.2
0.1 0
0 -0.2
0.175
0.15
0.125
0.1
0.075
09:45
10:00
10:15
10:30
swap free
1000
750
500
250
0
09:45
10:00
10:15
10:30

Condition Definition

Condition type: goes above

set alert on AWS.EC2.CPUUtilization

from (source) alerts-prod*

when the average what do these options mean?

crosses threshold 40 preview for a duration

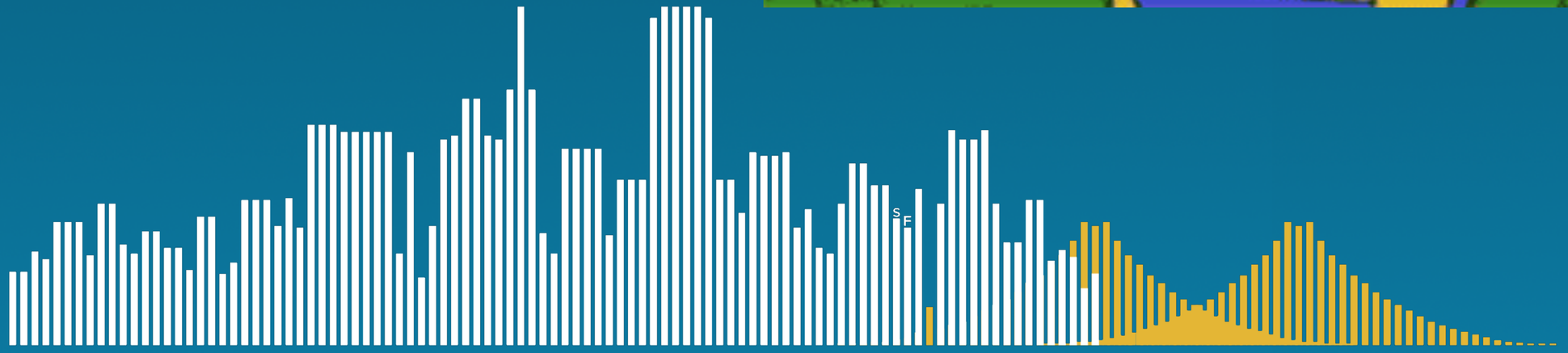
of 10 minutes

Alert Conditions | Notification Services

Select one or more services to notify when this alert fires.

Test Fire Alert

Create a new service

Campfire Data Room enabled	Customer.io 1 service available	flowdock 1 service available
Atlassian HipChat HipChat Librato Room enabled	Email 9 services available	OpsGenie 2 services available
pagerduty 2 services available	Slack 1 service available	Webhook 2 services available

Questions?