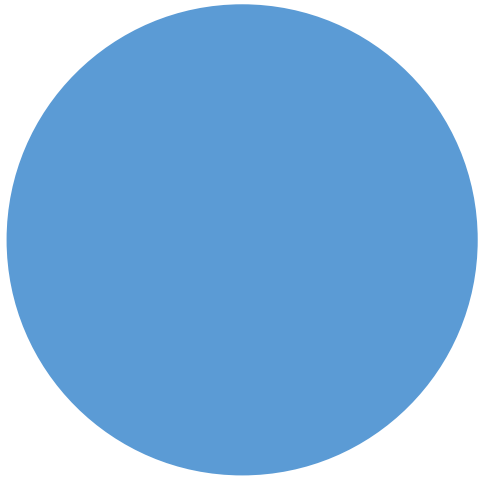
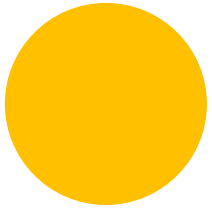



Observability in Distributed Systems

David Ostrovsky

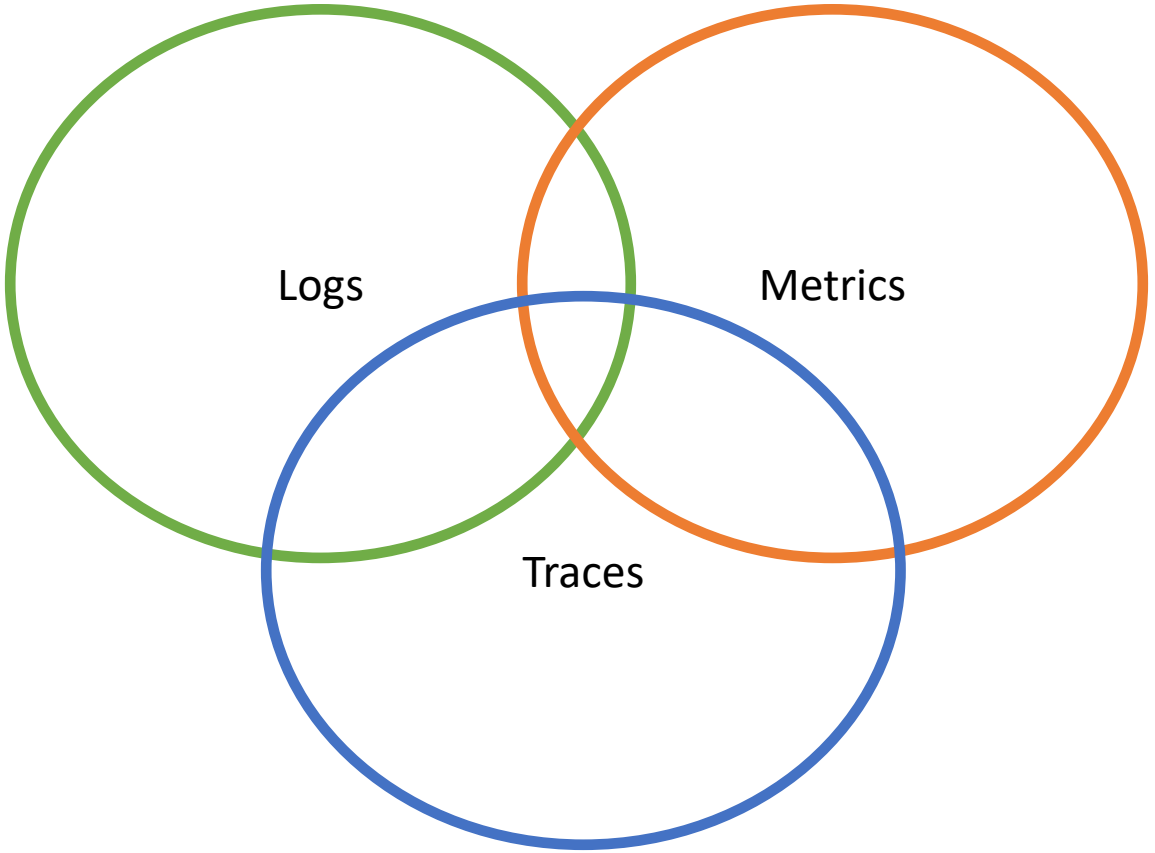
Principal Architect, Proofpoint

Observability is being able to answer

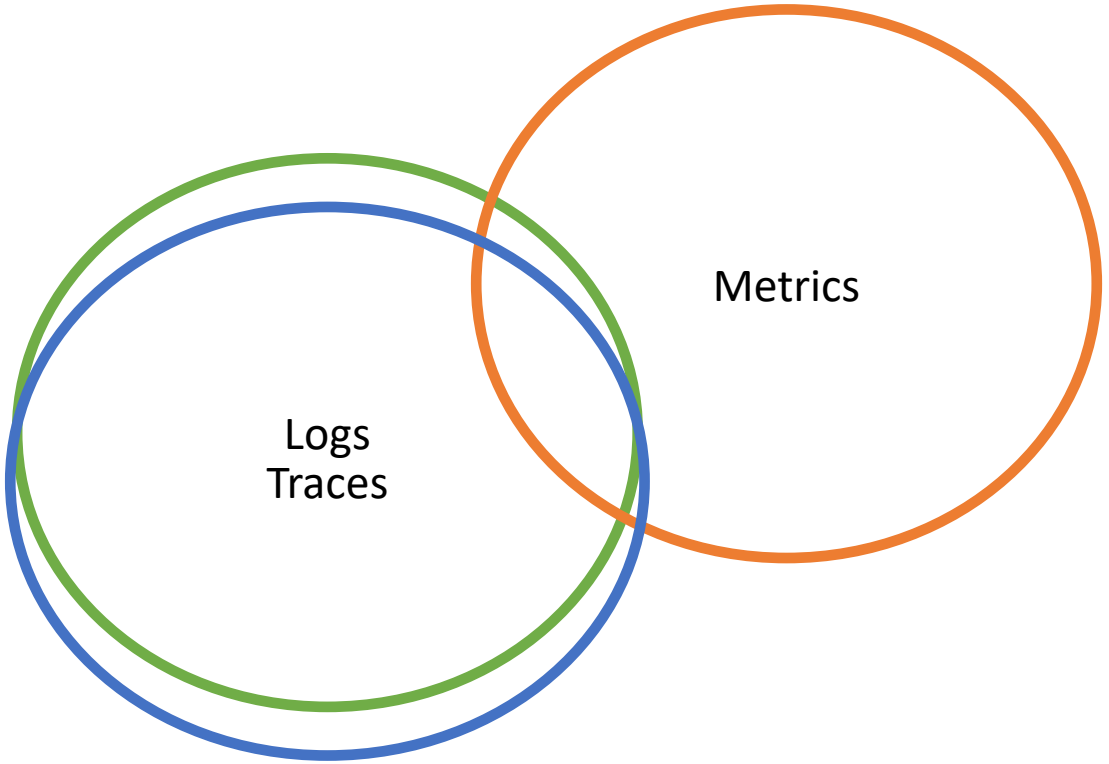
Is anything wrong?

What is wrong and **why**?

So Called ‘Observability Pillars’



Observability
(Proofpoint CASB)



Service Level Indicators

Business oriented

Simple and clear

As few as possible

SLI Example 1 (Old Proofpoint CASB)

TENANT	LOGIN EVENTS LAST	LOGIN EVENTS COUNTED	3PA EVENTS LAST	3PA EVENTS COUNTED	FILE EVENTS LAST	FILE EVENTS COUNTED	LAST UPDATED FILE	LAST UPDATED DLP FILE
	last happened at: 04/30/2019 13:41	(2929/50 in last 12h)	last happened at: 04/30/2019 14:02	(111/20 in last 2h)	no event in last 1h	(0/20000 in last 1h)	last happened at: 12/31/2036 03:35	last happened at: 04/30/2019 14:01
	last happened at: 04/30/2019 07:58	(3097/30 in last 12h)	last happened at: 04/30/2019 02:45	(36/50 in last 24h)	last happened at: 04/30/2019 13:31	(33/10 in last 4h)	last happened at: 04/30/2019 13:28	last happened at: 04/30/2019 13:28
	last happened at: 04/30/2019 13:43	(2583/2500 in last 1h)	last happened at: 04/30/2019 11:57	(286/50 in last 24h)	last happened at: 04/30/2019 14:00	(10091/10000 in last 1h)	last happened at: 02/07/2106 23:28	no file in last 6h
	no event in last 1h	(0/50 in last 1h)		(0/0 in last 24h)	no event in last 4h	(0/20 in last 4h)	last happened at: 01/01/4501 07:00	
	no event in last 10m	(179/200 in last 1h)	last happened at: 04/30/2019 13:53	(303/5 in last 1d)	last happened at: 04/30/2019 13:55	(1030/100 in last 3h)	last happened at: 04/30/2019 13:40	last happened at: 04/30/2019 00:28
	last happened at: 04/30/2019 13:43	(1232/2000 in last 1h)		(0/0 in last 1h)	last happened at: 04/30/2019 13:57	(4525/1000 in last 1h)	last happened at: 11/10/2019 01:18	no file in last 2h
	last happened at: 04/30/2019 13:44	(36958/10000 in last 1h)	last happened at: 04/30/2019 02:15	(7/2 in last 1d)	last happened at: 04/30/2019 13:57	(13573/2000 in last 1h)	last happened at: 11/23/2113 16:50	last happened at: 04/30/2019 13:52
	last happened at: 04/30/2019 13:43	(2670/2000 in last 1h)	last happened at: 04/30/2019 07:37	(6/2 in last 1d)	last happened at: 04/30/2019 13:59	(23608/10000 in last 1h)	last happened at: 01/01/2098 02:00	last happened at: 04/30/2019 14:01

SLI Example 2 (reddit.statuspage.io)

reddit

SUBSCRIBE TO UPDATES

All Systems Operational

Uptime over the past 90 days. View historical uptime.

reddit.com

Desktop Web	99.99 % uptime	Operational
Mobile Web	100 % uptime	Operational
Native Mobile Apps	100 % uptime	Operational
Vote Processing	100 % uptime	Operational
Comment Processing	100 % uptime	Operational
Spam Processing	100 % uptime	Operational
View Counts	100 % uptime	Operational

(90 days ago … Today)

reddit.com — Operational — 99.99 % uptime

redditgifts.com — Operational — 100 % uptime

Reddit Infrastructure — Operational

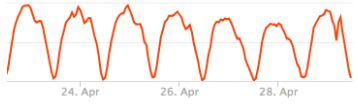
Reddit Media Storage — Operational

CDN - Fastly — Operational

System Metrics

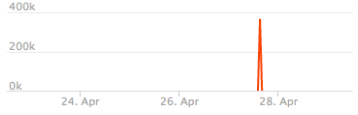
Day | Week | Month

reddit.com request rate

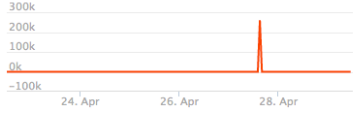


reddit.com error rate

vote backlog (posts) 2,327 votes



vote backlog (comments) 1,655 votes



comment tree backlog 1,212 comments

thumbnail and embed scraper backlog 155 posts

Metrics

Time-based

Aggregations, series calculations

Zoom-able

Relatively inexpensive

The Four "Golden Signals"

Latency

Traffic

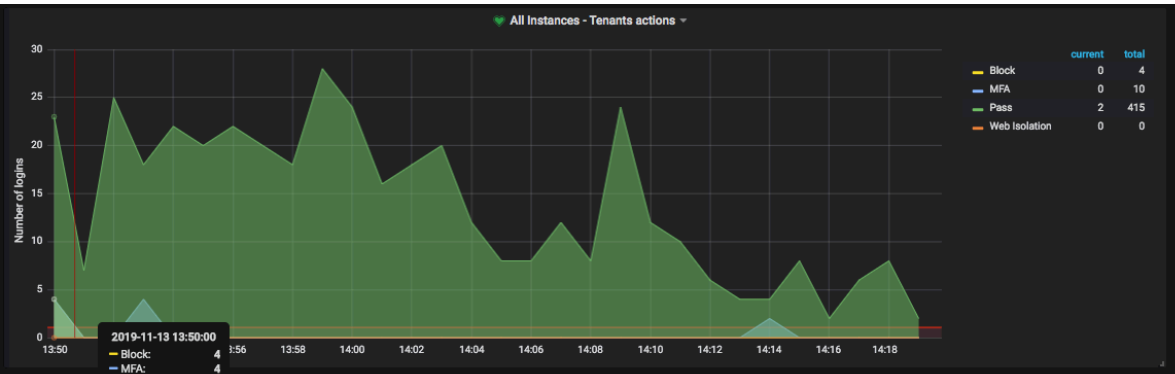
Error Rate

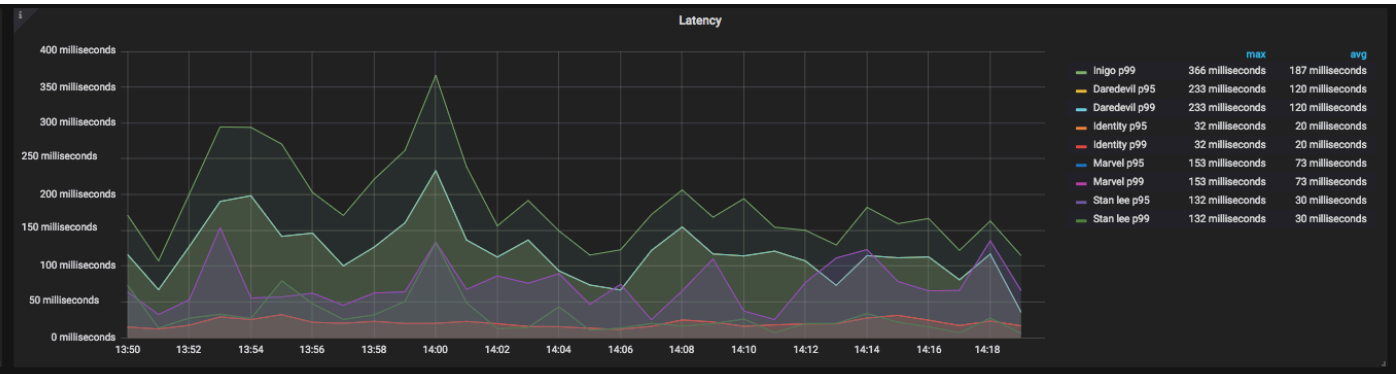
Saturation

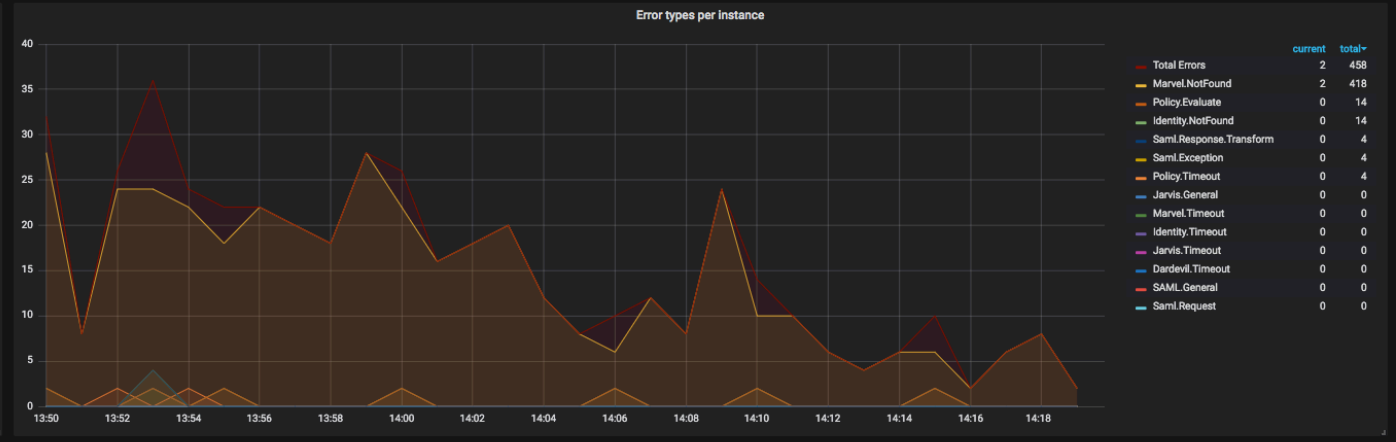
Google SRE Books: https://landing.google.com/sre/books/

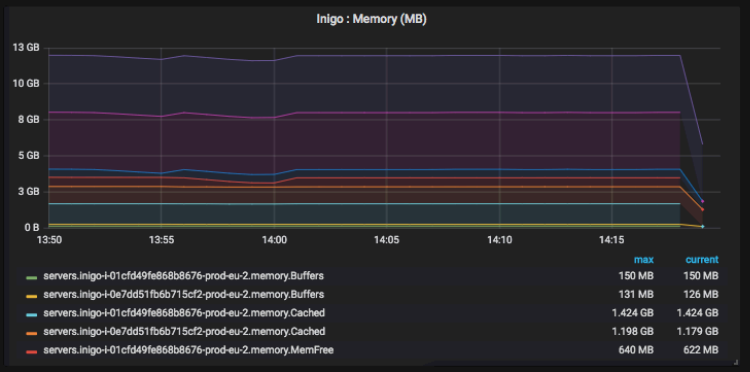
Bad Example (Old Proofpoint CASB)

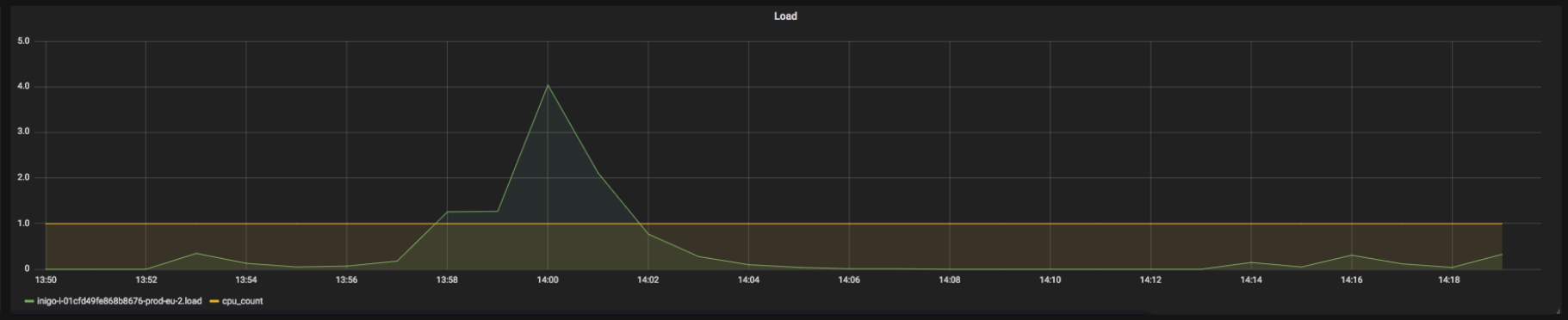
Good Example (Current Proofpoint CASB)











Logs

Centralized*

Limited retention

Relatively expensive

Tricky to reconcile in a distributed system

From Logging to Distributed Tracing

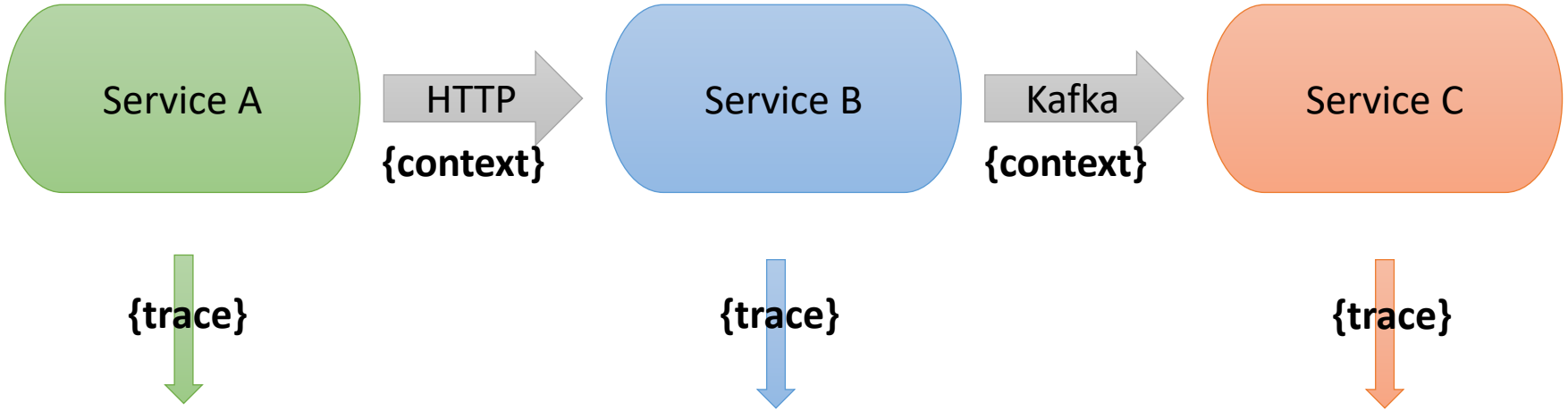
Put transaction context information in logs

Reconstruct transaction flow from logs

Query and visualize transaction-related logs

You just invented distributed tracing

Distributed Tracing in a Nutshell

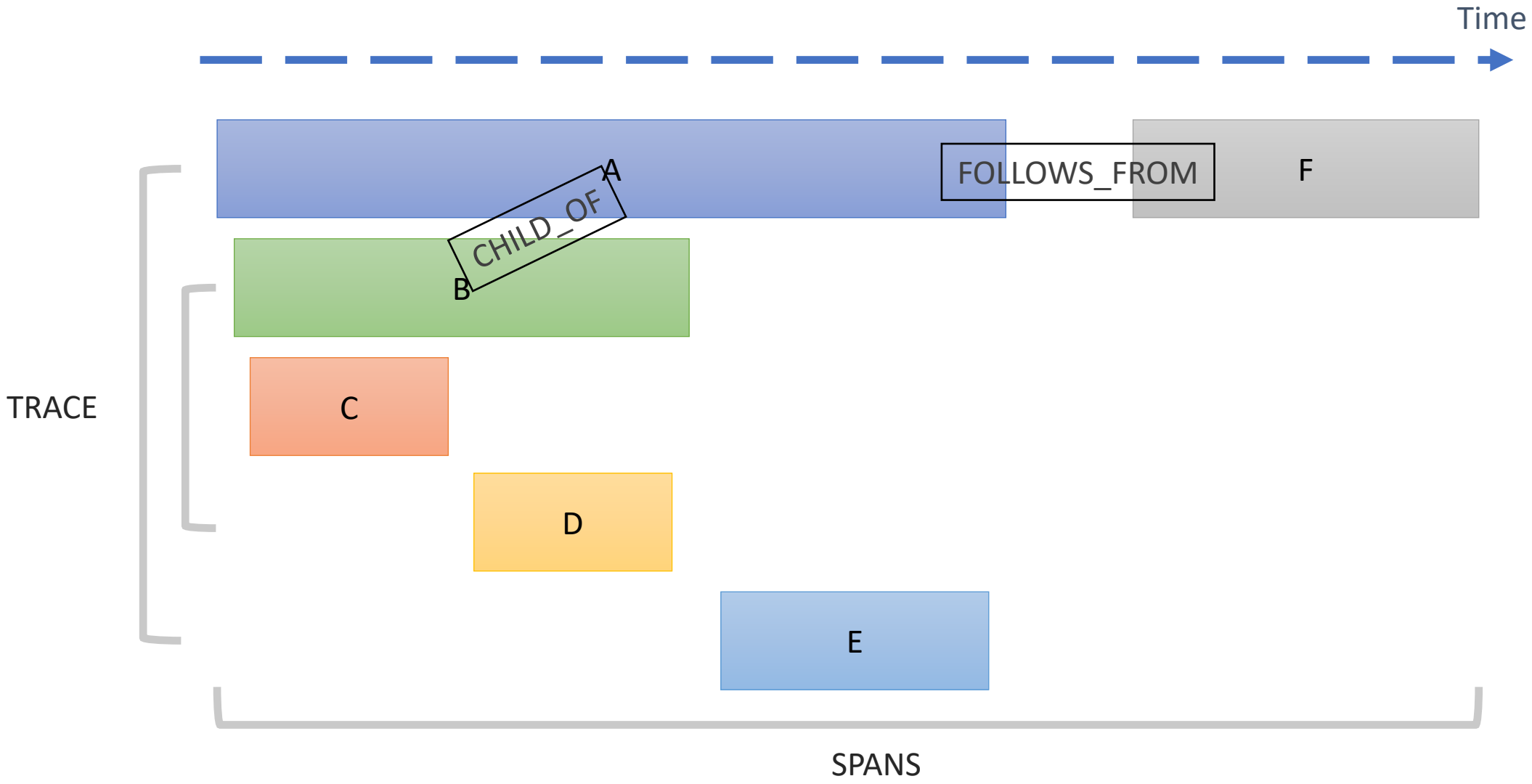



```
{
    "operation": "op1",
    "trace_id": 123,
    "span_id": "spanA",
    "start_time": 100000000,
    "end_time": 100010000,
    "tags" : {…}
}
```

```
{
    "operation": "op2",
    "trace_id": 123,
    "span_id": " spanB",
    "child_of": "spanA",
    "start_time": 100020000,
    "end_time": 100030000,
    "tags" : {…}
}
```

```
{
    "operation": "op3",
    "trace_id": 123,
    "span_id": "spanC",
    "follows_from": "spanB",
    "start_time": 100040000,
    "end_time": 100050000,
    "tags" : {…}
}
```

Anatomy of a Transaction Trace



Value of Distributed Tracing

Answer business oriented questions

Root cause investigation

Application performance metrics

Distributed Tracing

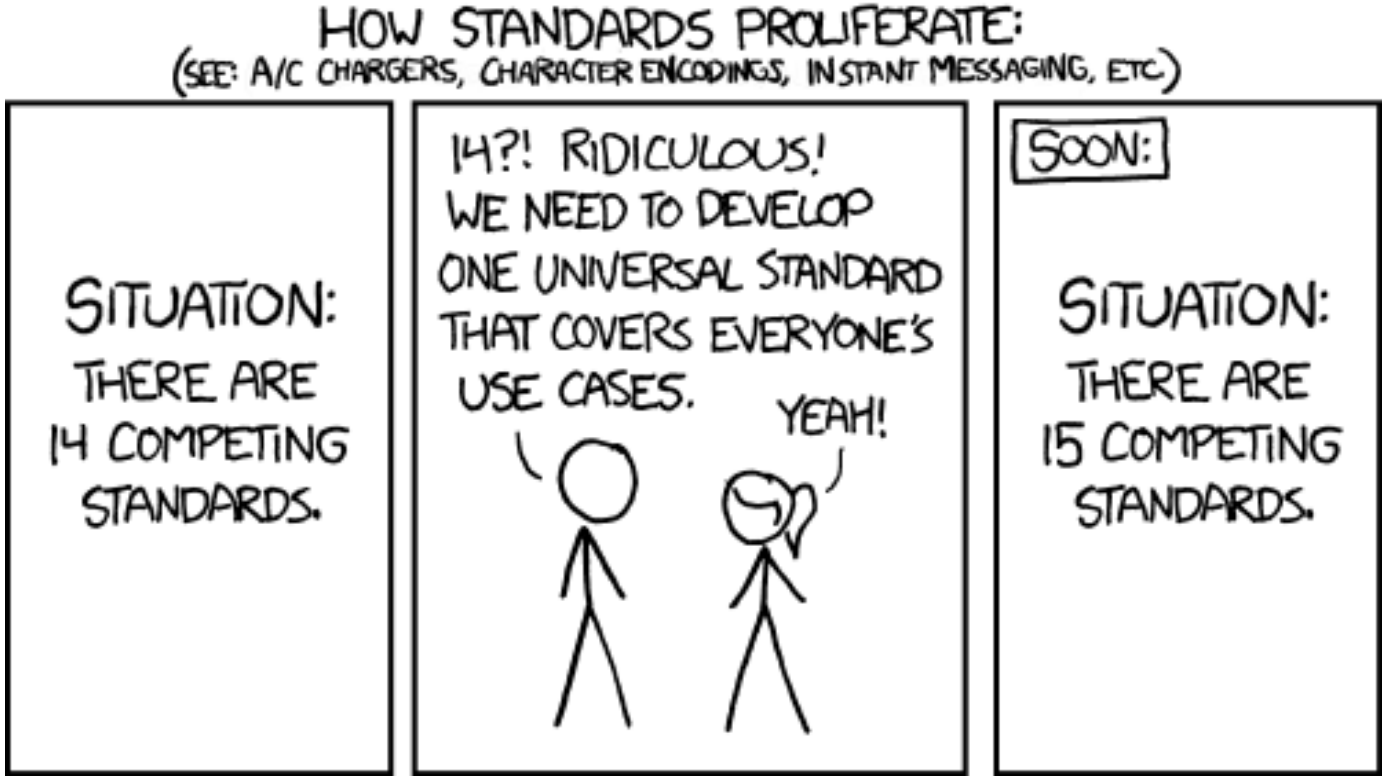
Instrument

Collect and Sample

Index

Query and Visualize

OpenTelemetry *OpenTracing*, *OpenCensus*, OpenMetrics*



https://xkcd.com/927/

Instrumenting and Collecting with OpenTracing

Vendor neutral

Multiple platforms

Solves a narrow and specific problem

Sampling and Indexing Traces

Too much data to store all traces for long

Sampling strategies

Indexing backend depends on use-case

Common tools use Elasticsearch and Cassandra

Visualizing Distributed Traces

Latency waterfall and flame graphs

Data flow diagrams

Time-series histograms

Use your imagination

Transaction sample

View transaction in Discover | View full trace

Timestamp
2 hours ago (November 1st 2018, 11:34:28.798)

URL
http://172.18.0.8:3000/api/types

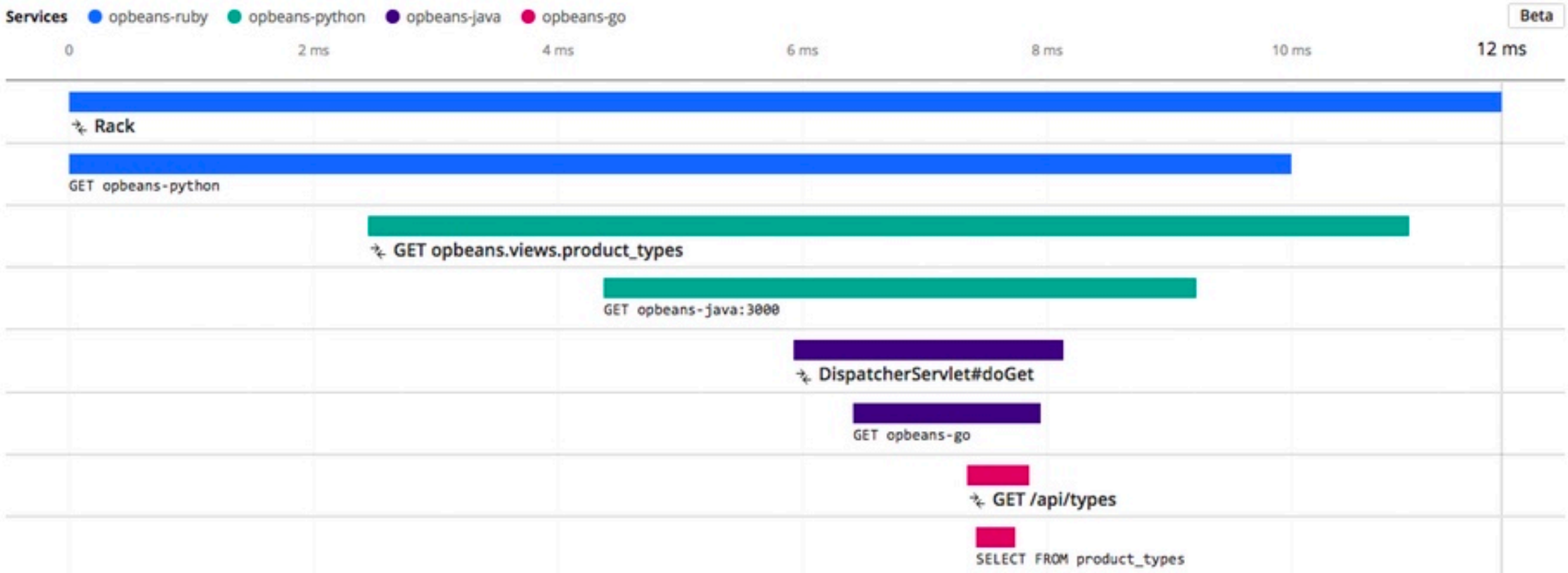
Duration
12 ms

% of trace
100.00%

Result
HTTP 2xx

User ID
N/A

Timeline | Request | Response | System | Service | Process | User | Tags | Custom



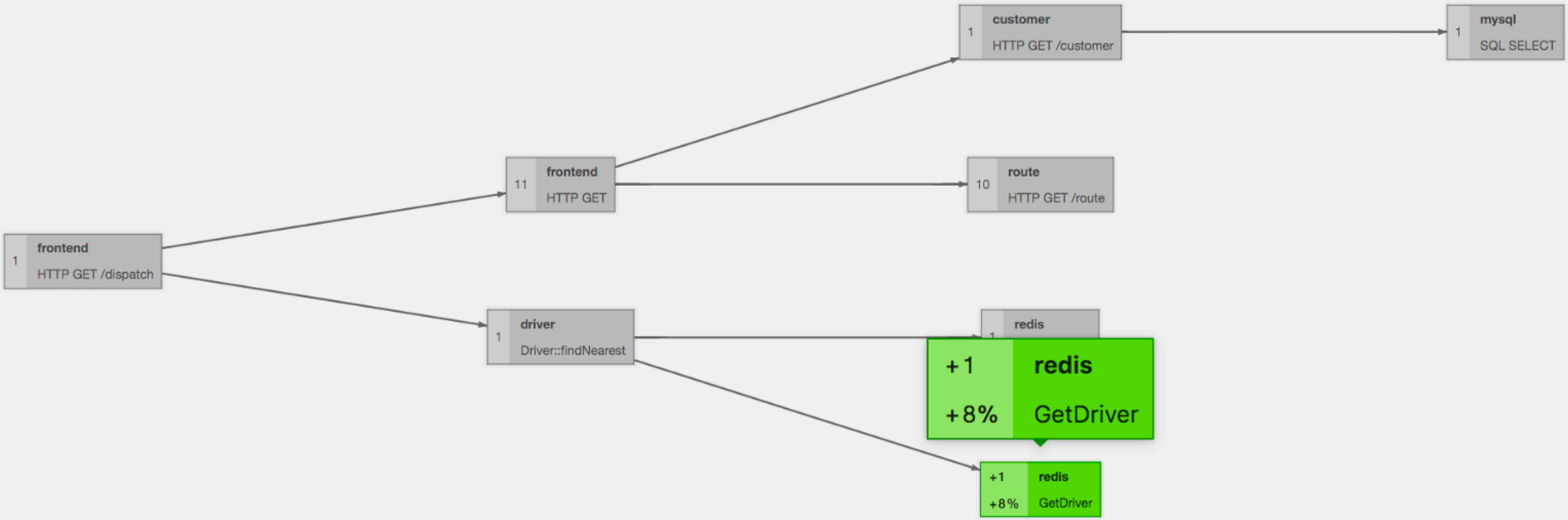
A frontend: HTTP GET / dispatch 6401e43

Today, 2:00:43 pm Duration: 1.46s Spans: 50

VS

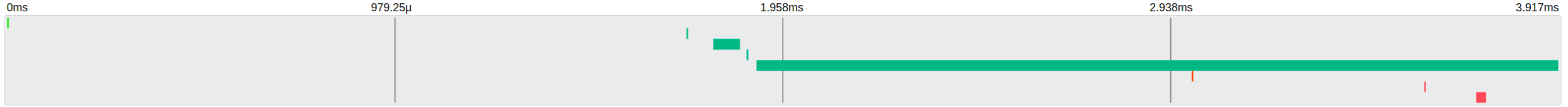
B frontend: HTTP GET / dispatch 6461459

Today, 2:00:43 pm Duration: 1.31s Spans: 51



5c902ec22bc6e32079fc1df1ca56f6d0

Duration: 3.917ms Services: 5 Depth: 5 Total Spans: 8



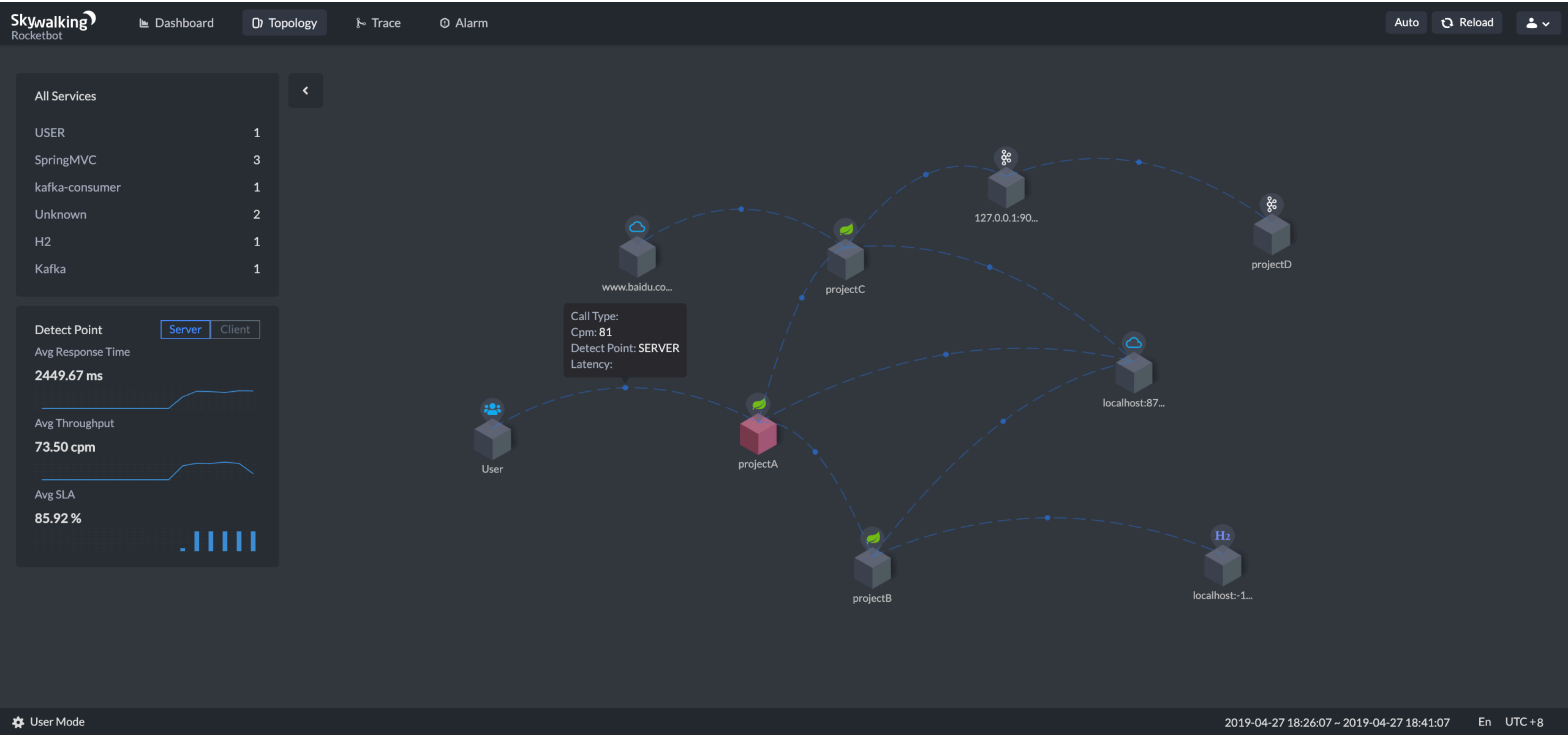
Service Name	Span Name	0ms	979.25µ	1.958ms	2.938ms
kafka-connect-twitter	on_send	1µs			
stream-transform	poll		1µs		
stream-transform	parse_json		67µs		
stream-transform	json_to_avro		3µs		
stream-transform	send		2.025ms		
kafka-connect-jdbc	on_consume			1µs	
twitter-console-consumer	poll				1µs
twitter-console-consumer	print-hello				25µs

stream-transform: send

kafka stream-transform

Date Time	Relative Time	Annotation	Address
03/19 00:50:26.062902	1.892ms	Producer Start	172.18.0.13 (stream-transform)
03/19 00:50:26.064927	3.917ms	Producer Finish	172.18.0.13 (stream-transform)

Key	Value
kafka.topic	twitter_avro_v1
Broker Address	kafka


Skywalking
Rocketbot
Dashboard
Topology
Trace
Alarm
Auto
Reload
All Services
USER 1
SpringMVC 3
kafka-consumer 1
Unknown 2
H2 1
Kafka 1
Detect Point
Server
Client
Avg Response Time
2449.67 ms
Avg Throughput
73.50 cpm
Avg SLA
85.92 %
Call Type:
Cpm: 81
Detect Point: SERVER
Latency:
www.baidu.co...
projectC
127.0.0.1:90...
projectD
User
projectA
localhost:87...
projectB
localhost:-1...
User Mode
2019-04-27 18:26:07 ~ 2019-04-27 18:41:07
En
UTC +8

Enter Service Name, Annotation, Trace ID or click the Help icon for additional details

Last 5 Minutes

Service map

Updated on 2016/12/04 03:14:26 (UTC -08:00)

Map legend