CLOUD NATIVE
COMPUTING FOUNDATION

Jaeger
Project Deep Dive

Annanay Agarwal (Grafana)
Pavol Loffay (Traceable.ai)
Yuri Shkuro (Facebook)

KubeCon + CloudNativeCon NA 2020 Virtual
Thu, Nov 19 • 2:55 pm - 3:30 pm

About

- Yuri Shkuro (https://github.com/yurishkuro)
  - Software engineer
  - Maintainer of Jaeger, OpenTracing, OpenTelemetry
  - Author of "Mastering Distributed Tracing" book

Agenda

- Observability and tracing
- Jaeger features
- Jaeger architecture
- Sampling
- Jaeger and OpenTelemetry
- Jaeger on Kubernetes

About

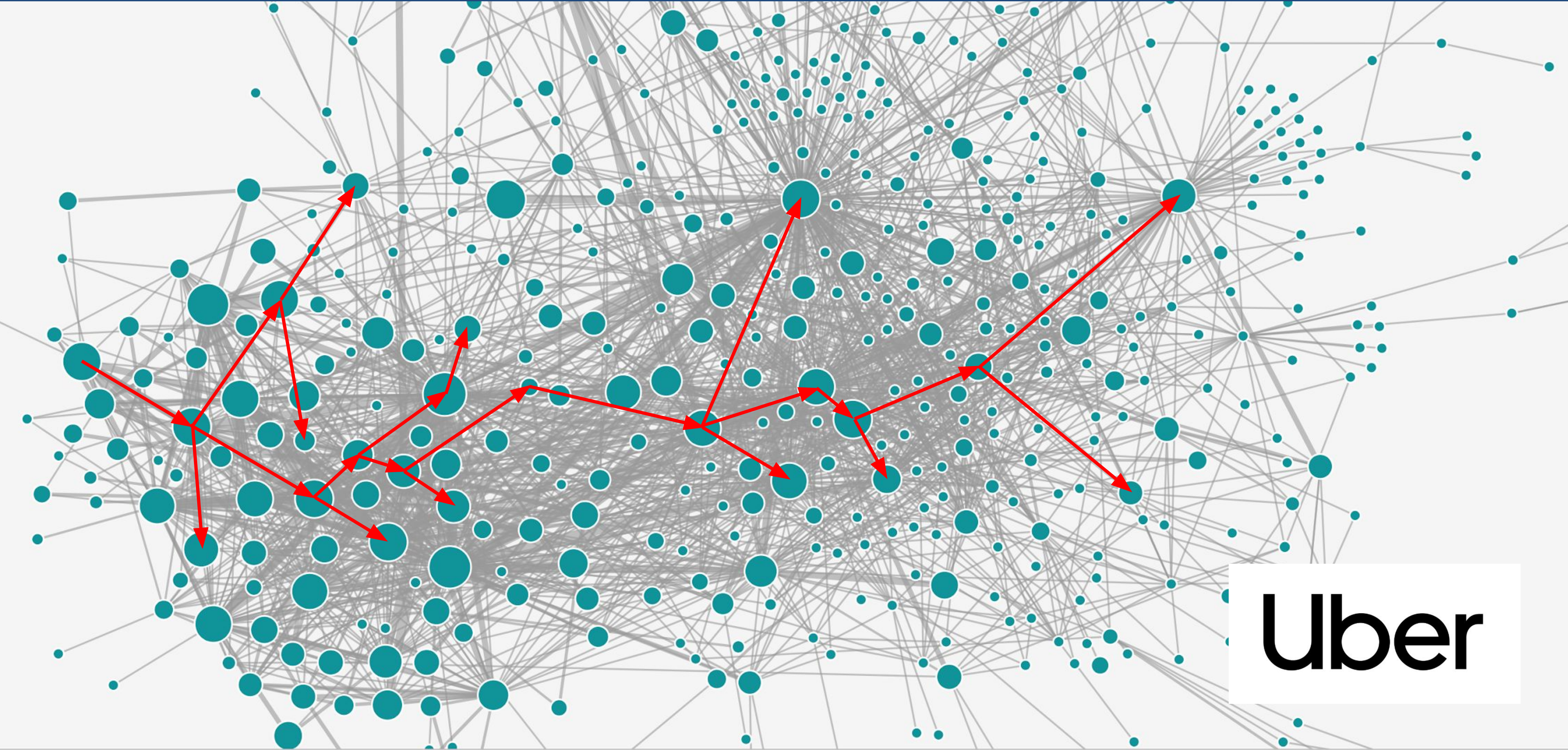
- Annanay Agarwal (https://github.com/annanay25)
  - Software developer at Grafana Labs
  - Contributor to Jaeger and OpenTelemetry projects

CLOUD NATIVE
COMPUTING FOUNDATION

What is Tracing & Why?

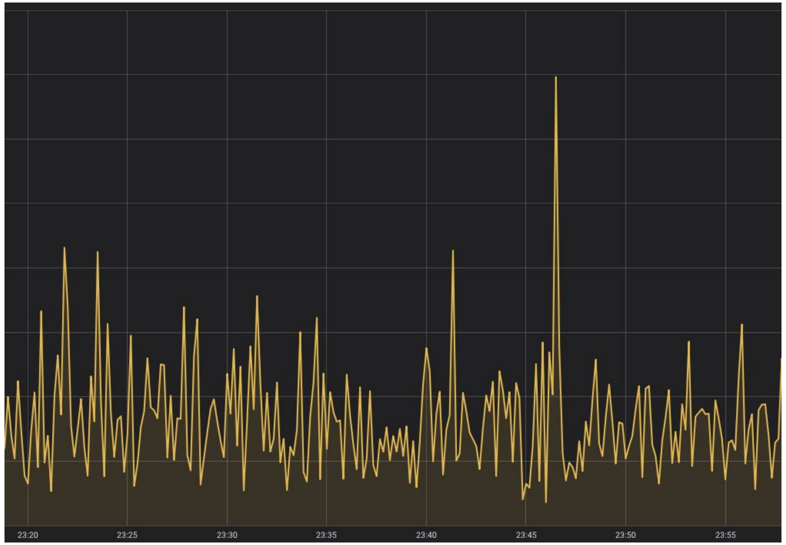
Concepts and terminology

BILLIONS of times a day!



Uber

Metrics

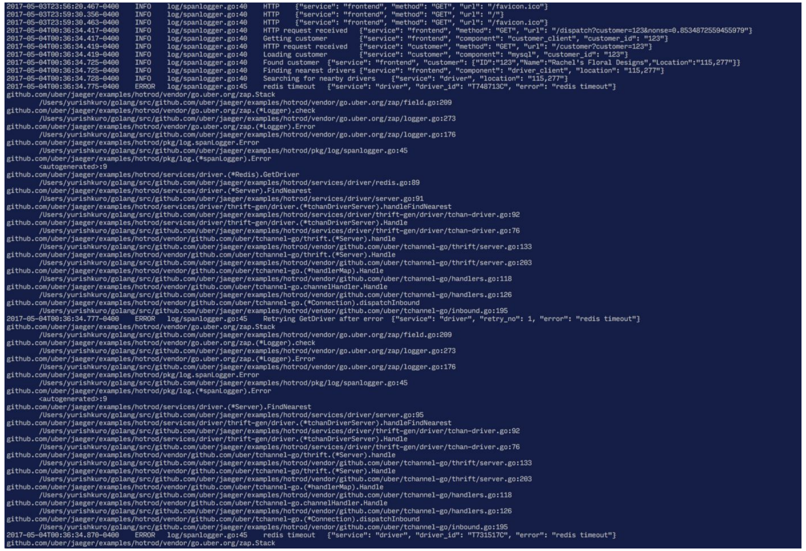


```
http_request_duration_sec{“app=ice-cream-shop”} 10s
```

Metrics - Cardinality

```
http_request_duration_sec{“app=ice-cream-shop”
datacenter=”us-central”, env=”production”,
service=”cart-manager”, path=”/api/order”,
func_name=”my-func”} 6s
```

Logs - stack trace?



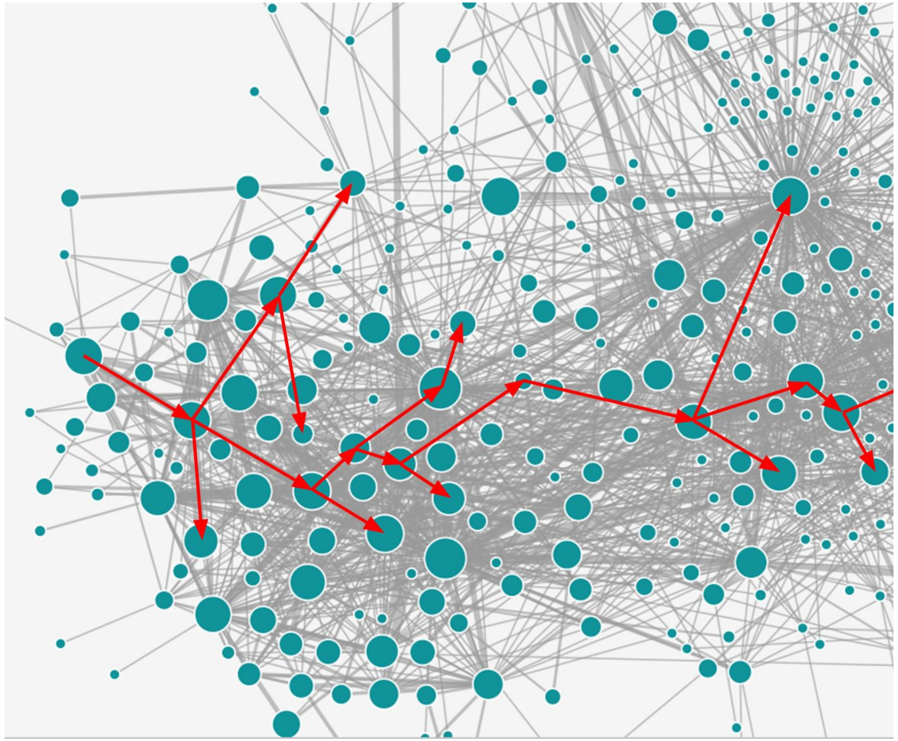
```
2017-05-03T23:56:20.467-0400    INFO    log/spanlogger.go:40    HTTP    {"service": "frontend", "method": "GET", "url": "/favicon.ico"}
2017-05-03T23:59:30.356-0400    INFO    log/spanlogger.go:40    HTTP    {"service": "frontend", "method": "GET", "url": "/"}
2017-05-03T23:59:30.463-0400    INFO    log/spanlogger.go:40    HTTP    {"service": "frontend", "method": "GET", "url": "/favicon.ico"}
2017-05-04T00:36:34.417-0400    INFO    log/spanlogger.go:40    HTTP request received   {"service": "frontend", "method": "GET", "url": "/dispatch?customer=123&nonse=0.8534872559455979"}
2017-05-04T00:36:34.417-0400    INFO    log/spanlogger.go:40    Getting customer        {"service": "frontend", "component": "customer_client", "customer_id": "123"}
2017-05-04T00:36:34.419-0400    INFO    log/spanlogger.go:40    HTTP request received   {"service": "customer", "method": "GET", "url": "/customer?customer=123"}
2017-05-04T00:36:34.419-0400    INFO    log/spanlogger.go:40    Loading customer        {"service": "customer", "component": "mysql", "customer_id": "123"}
2017-05-04T00:36:34.725-0400    INFO    log/spanlogger.go:40    Found customer  {"service": "frontend", "customer": {"ID":"123","Name":"Rachel's Floral Designs","Location":"115,277"}}
2017-05-04T00:36:34.725-0400    INFO    log/spanlogger.go:40    Finding nearest drivers {"service": "frontend", "component": "driver_client", "location": "115,277"}
2017-05-04T00:36:34.728-0400    INFO    log/spanlogger.go:40    Searching for nearby drivers    {"service": "driver", "location": "115,277"}
2017-05-04T00:36:34.775-0400    ERROR   log/spanlogger.go:45    redis timeout   {"service": "driver", "driver_id": "T748713C", "error": "redis timeout"}
github.com/uber/jaeger/examples/hotrod/vendor/go.uber.org/zap.Stack
        /Users/yurishkuro/golang/src/github.com/uber/jaeger/examples/hotrod/vendor/go.uber.org/zap/field.go:209
github.com/uber/jaeger/examples/hotrod/vendor/go.uber.org/zap.(*Logger).check
        /Users/yurishkuro/golang/src/github.com/uber/jaeger/examples/hotrod/vendor/go.uber.org/zap/logger.go:273
github.com/uber/jaeger/examples/hotrod/vendor/go.uber.org/zap.(*Logger).Error
        /Users/yurishkuro/golang/src/github.com/uber/jaeger/examples/hotrod/vendor/go.uber.org/zap/logger.go:176
github.com/uber/jaeger/examples/hotrod/pkg/log.spanLogger.Error
        /Users/yurishkuro/golang/src/github.com/uber/jaeger/examples/hotrod/pkg/log/spanlogger.go:45
github.com/uber/jaeger/examples/hotrod/pkg/log.(*spanLogger).Error
        <autogenerated>:9
github.com/uber/jaeger/examples/hotrod/services/driver.(*Redis).GetDriver
        /Users/yurishkuro/golang/src/github.com/uber/jaeger/examples/hotrod/services/driver/redis.go:89
github.com/uber/jaeger/examples/hotrod/services/driver.(*Server).FindNearest
        /Users/yurishkuro/golang/src/github.com/uber/jaeger/examples/hotrod/services/driver/server.go:91
github.com/uber/jaeger/examples/hotrod/services/driver/thrift-gen/driver.(*tchanDriverServer).handleFindNearest
        /Users/yurishkuro/golang/src/github.com/uber/jaeger/examples/hotrod/services/driver/thrift-gen/driver/tchan-driver.go:92
github.com/uber/jaeger/examples/hotrod/services/driver/thrift-gen/driver.(*tchanDriverServer).Handle
        /Users/yurishkuro/golang/src/github.com/uber/jaeger/examples/hotrod/services/driver/thrift-gen/driver/tchan-driver.go:76
github.com/uber/jaeger/examples/hotrod/vendor/github.com/uber/tchannel-go/thrift.(*Server).handle
        /Users/yurishkuro/golang/src/github.com/uber/jaeger/examples/hotrod/vendor/github.com/uber/tchannel-go/thrift/server.go:133
github.com/uber/jaeger/examples/hotrod/vendor/github.com/uber/tchannel-go/thrift.(*Server).Handle
        /Users/yurishkuro/golang/src/github.com/uber/jaeger/examples/hotrod/vendor/github.com/uber/tchannel-go/thrift/server.go:203
github.com/uber/jaeger/examples/hotrod/vendor/github.com/uber/tchannel-go.(*handlerMap).Handle
        /Users/yurishkuro/golang/src/github.com/uber/jaeger/examples/hotrod/vendor/github.com/uber/tchannel-go/handlers.go:118
github.com/uber/jaeger/examples/hotrod/vendor/github.com/uber/tchannel-go.channelHandler.Handle
        /Users/yurishkuro/golang/src/github.com/uber/jaeger/examples/hotrod/vendor/github.com/uber/tchannel-go/handlers.go:126
github.com/uber/jaeger/examples/hotrod/vendor/github.com/uber/tchannel-go.(*Connection).dispatchInbound
        /Users/yurishkuro/golang/src/github.com/uber/jaeger/examples/hotrod/vendor/github.com/uber/tchannel-go/inbound.go:195
2017-05-04T00:36:34.777-0400    ERROR   log/spanlogger.go:45    Retrying GetDriver after error  {"service": "driver", "retry_no": 1, "error": "redis timeout"}
github.com/uber/jaeger/examples/hotrod/vendor/go.uber.org/zap.Stack
        /Users/yurishkuro/golang/src/github.com/uber/jaeger/examples/hotrod/vendor/go.uber.org/zap/field.go:209
github.com/uber/jaeger/examples/hotrod/vendor/go.uber.org/zap.(*Logger).check
        /Users/yurishkuro/golang/src/github.com/uber/jaeger/examples/hotrod/vendor/go.uber.org/zap/logger.go:273
github.com/uber/jaeger/examples/hotrod/vendor/go.uber.org/zap.(*Logger).Error
        /Users/yurishkuro/golang/src/github.com/uber/jaeger/examples/hotrod/vendor/go.uber.org/zap/logger.go:176
github.com/uber/jaeger/examples/hotrod/pkg/log.spanLogger.Error
        /Users/yurishkuro/golang/src/github.com/uber/jaeger/examples/hotrod/pkg/log/spanlogger.go:45
github.com/uber/jaeger/examples/hotrod/pkg/log.(*spanLogger).Error
        <autogenerated>:9
github.com/uber/jaeger/examples/hotrod/services/driver.(*Server).FindNearest
        /Users/yurishkuro/golang/src/github.com/uber/jaeger/examples/hotrod/services/driver/server.go:95
github.com/uber/jaeger/examples/hotrod/services/driver/thrift-gen/driver.(*tchanDriverServer).handleFindNearest
        /Users/yurishkuro/golang/src/github.com/uber/jaeger/examples/hotrod/services/driver/thrift-gen/driver/tchan-driver.go:92
github.com/uber/jaeger/examples/hotrod/services/driver/thrift-gen/driver.(*tchanDriverServer).Handle
        /Users/yurishkuro/golang/src/github.com/uber/jaeger/examples/hotrod/services/driver/thrift-gen/driver/tchan-driver.go:76
github.com/uber/jaeger/examples/hotrod/vendor/github.com/uber/tchannel-go/thrift.(*Server).handle
        /Users/yurishkuro/golang/src/github.com/uber/jaeger/examples/hotrod/vendor/github.com/uber/tchannel-go/thrift/server.go:133
github.com/uber/jaeger/examples/hotrod/vendor/github.com/uber/tchannel-go/thrift.(*Server).Handle
        /Users/yurishkuro/golang/src/github.com/uber/jaeger/examples/hotrod/vendor/github.com/uber/tchannel-go/thrift/server.go:203
github.com/uber/jaeger/examples/hotrod/vendor/github.com/uber/tchannel-go.(*handlerMap).Handle
        /Users/yurishkuro/golang/src/github.com/uber/jaeger/examples/hotrod/vendor/github.com/uber/tchannel-go/handlers.go:118
github.com/uber/jaeger/examples/hotrod/vendor/github.com/uber/tchannel-go.channelHandler.Handle
        /Users/yurishkuro/golang/src/github.com/uber/jaeger/examples/hotrod/vendor/github.com/uber/tchannel-go/handlers.go:126
github.com/uber/jaeger/examples/hotrod/vendor/github.com/uber/tchannel-go.(*Connection).dispatchInbound
        /Users/yurishkuro/golang/src/github.com/uber/jaeger/examples/hotrod/vendor/github.com/uber/tchannel-go/inbound.go:195
2017-05-04T00:36:34.870-0400    ERROR   log/spanlogger.go:45    redis timeout   {"service": "driver", "driver_id": "T731517C", "error": "redis timeout"}
github.com/uber/jaeger/examples/hotrod/vendor/go.uber.org/zap.Stack
```

Logs are a mess: concurrent requests, multiple hosts, impossible to correlate.

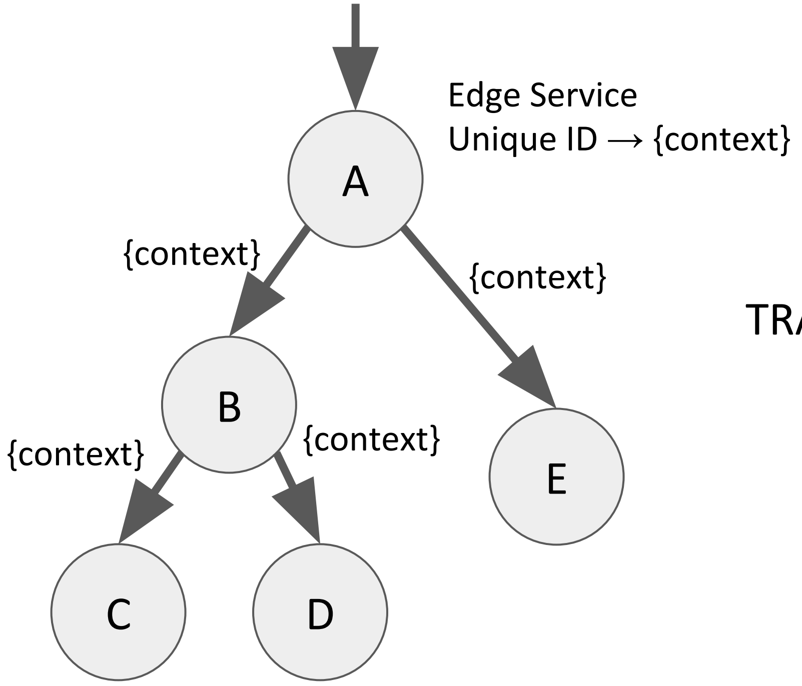
Monitoring tools must tell stories!

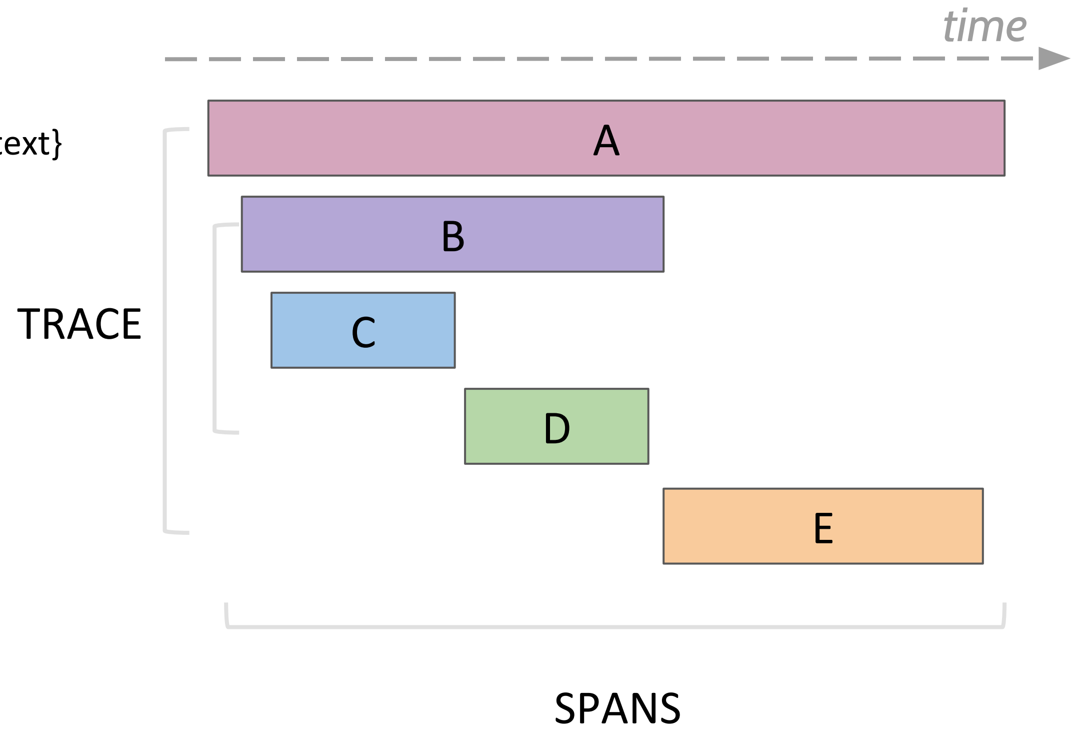
Do you like debugging without a stack trace?

We need to monitor distributed transactions ⇒ **distributed tracing**!



Context Propagation & Distributed Tracing



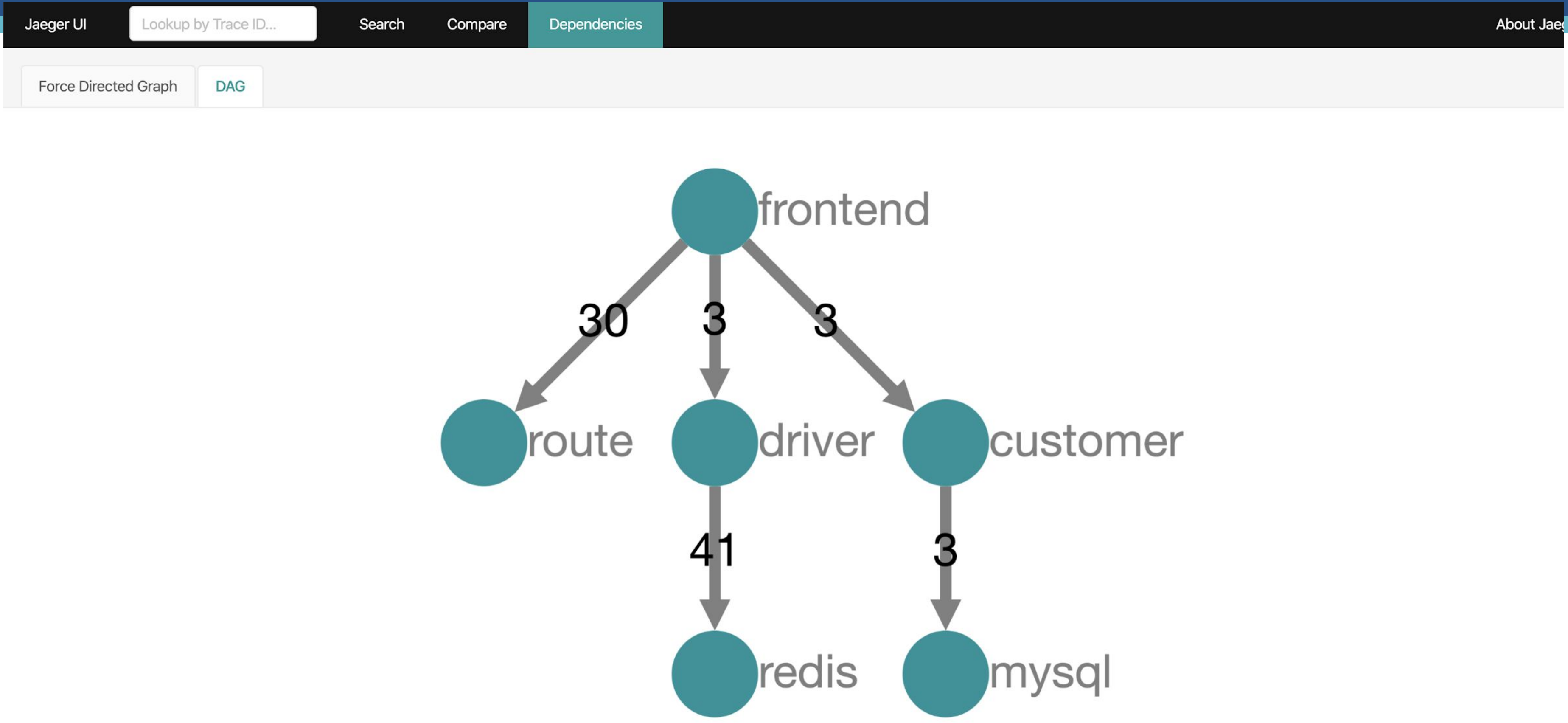


CLOUD NATIVE
COMPUTING FOUNDATION

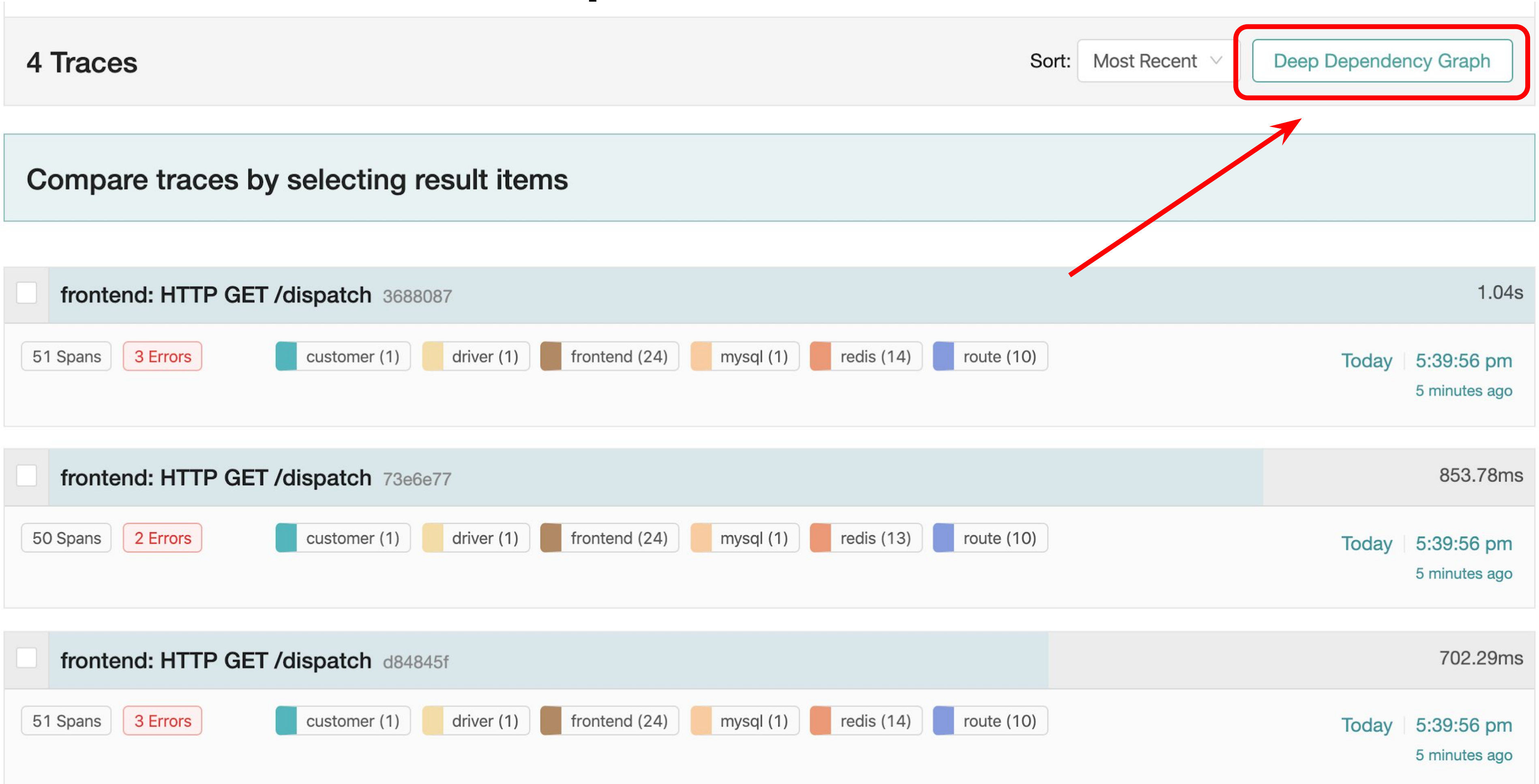
Let’s look at some traces

http://bit.do/jaeger-hotrod

Service dependencies diagram



Transitive Service Graphs



4 Traces

Sort: Most Recent

Compare traces by selecting result items

frontend: HTTP GET /dispatch 3688087 1.04s

51 Spans 3 Errors customer (1) driver (1) frontend (24) mysql (1) redis (14) route (10) Today | 5:39:56 pm 5 minutes ago

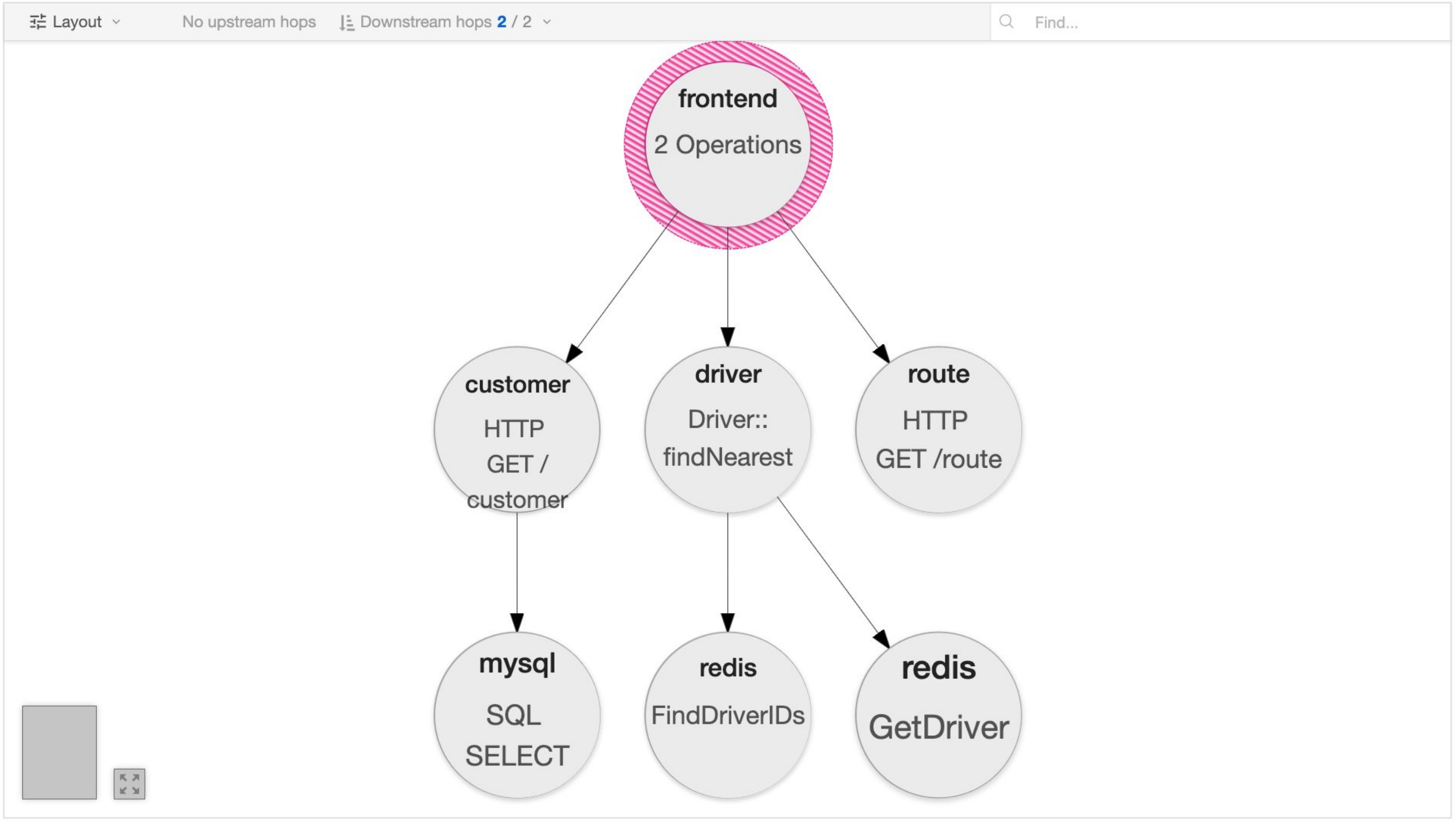
frontend: HTTP GET /dispatch 73e6e77 853.78ms

50 Spans 2 Errors customer (1) driver (1) frontend (24) mysql (1) redis (13) route (10) Today | 5:39:56 pm 5 minutes ago

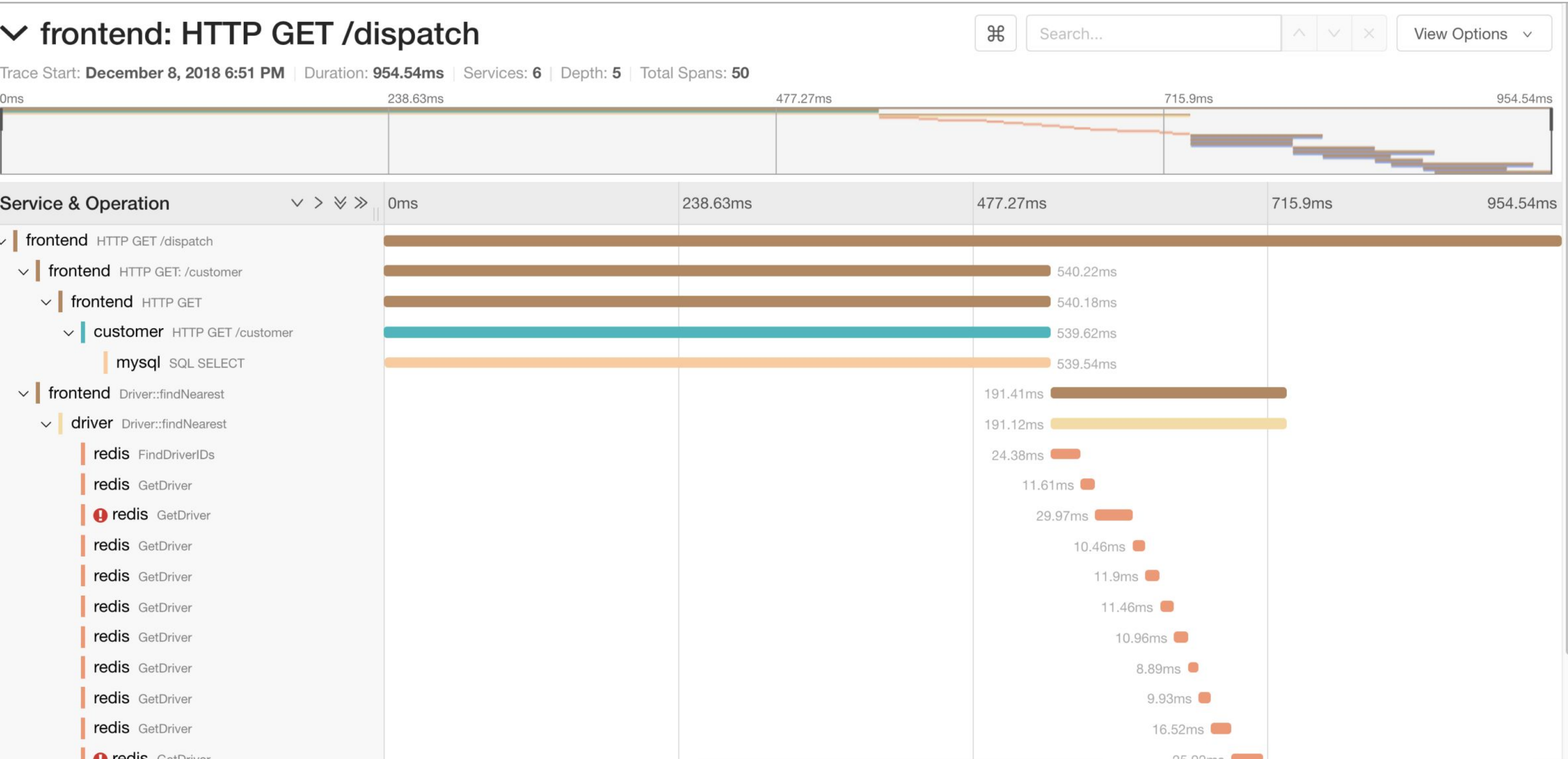
frontend: HTTP GET /dispatch d84845f 702.29ms

51 Spans 3 Errors customer (1) driver (1) frontend (24) mysql (1) redis (14) route (10) Today | 5:39:56 pm 5 minutes ago

Transitive Service Graphs



Trace timeline

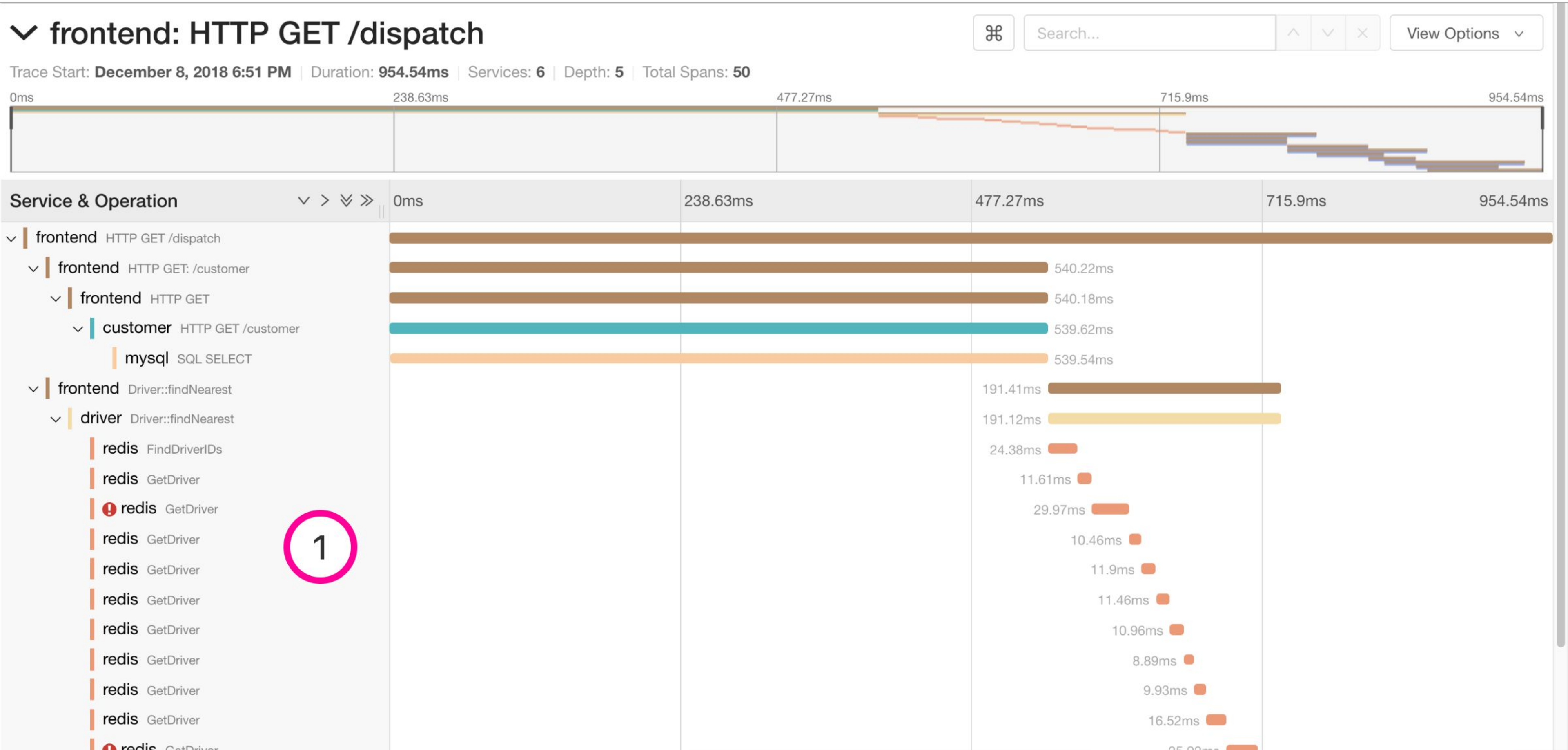


frontend: HTTP GET /dispatch

Trace Start: **December 8, 2018 6:51 PM** | Duration: **954.54ms** | Services: **6** | Depth: **5** | Total Spans: **50**

Service & Operation	Duration
frontend HTTP GET /dispatch	
frontend HTTP GET: /customer	540.22ms
frontend HTTP GET	540.18ms
customer HTTP GET /customer	539.62ms
mysql SQL SELECT	539.54ms
frontend Driver::findNearest	191.41ms
driver Driver::findNearest	191.12ms
redis FindDriverIDs	24.38ms
redis GetDriver	11.61ms
redis GetDriver	29.97ms
redis GetDriver	10.46ms
redis GetDriver	11.9ms
redis GetDriver	11.46ms
redis GetDriver	10.96ms
redis GetDriver	8.89ms
redis GetDriver	9.93ms
redis GetDriver	16.52ms
redis GetDriver	

Trace timeline – Parent → Child → Grandchild



frontend: HTTP GET /dispatch

Search... View Options

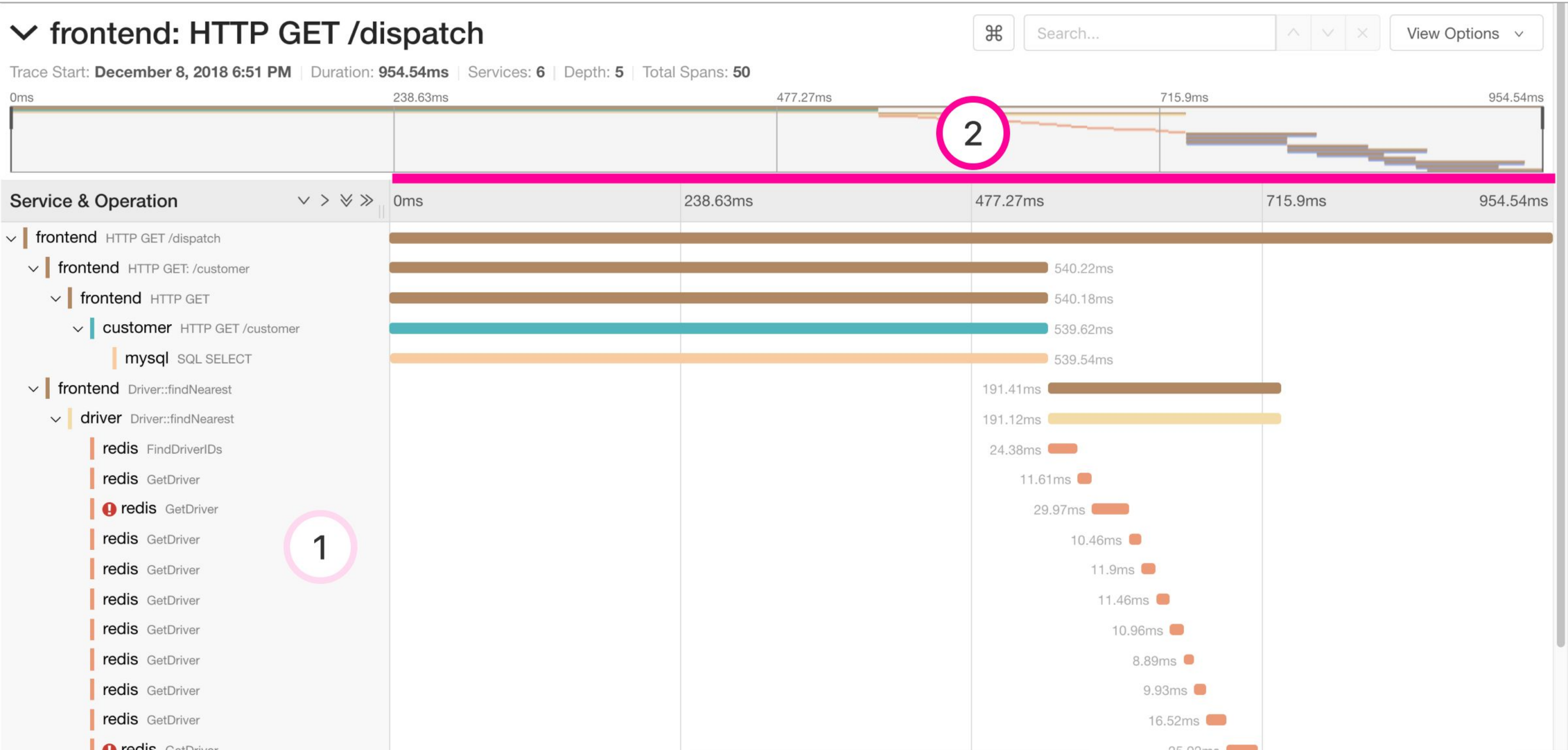
Trace Start: **December 8, 2018 6:51 PM** | Duration: **954.54ms** | Services: **6** | Depth: **5** | Total Spans: **50**

0ms 238.63ms 477.27ms 715.9ms 954.54ms

Service & Operation	0ms 238.63ms 477.27ms 715.9ms 954.54ms
frontend HTTP GET /dispatch	
frontend HTTP GET: /customer	540.22ms
frontend HTTP GET	540.18ms
customer HTTP GET /customer	539.62ms
mysql SQL SELECT	539.54ms
frontend Driver::findNearest	191.41ms
driver Driver::findNearest	191.12ms
redis FindDriverIDs	24.38ms
redis GetDriver	11.61ms
! redis GetDriver	29.97ms
redis GetDriver	10.46ms
redis GetDriver	11.9ms
redis GetDriver	11.46ms
redis GetDriver	10.96ms
redis GetDriver	8.89ms
redis GetDriver	9.93ms
redis GetDriver	16.52ms

1

Trace timeline – Time + Mini-map

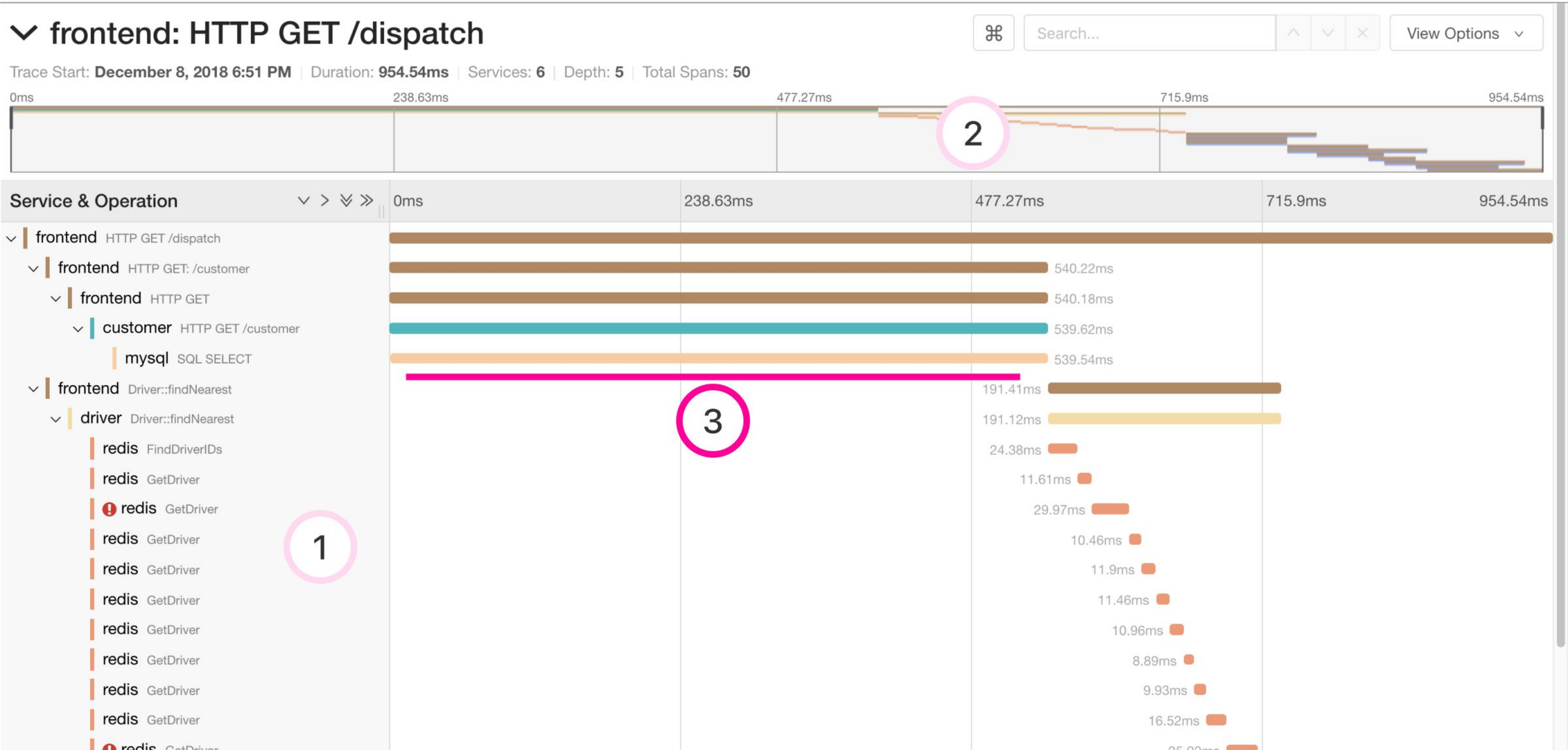


frontend: HTTP GET /dispatch

Trace Start: **December 8, 2018 6:51 PM** | Duration: **954.54ms** | Services: **6** | Depth: **5** | Total Spans: **50**

Service & Operation	Duration
frontend HTTP GET /dispatch	
frontend HTTP GET: /customer	540.22ms
frontend HTTP GET	540.18ms
customer HTTP GET /customer	539.62ms
mysql SQL SELECT	539.54ms
frontend Driver::findNearest	191.41ms
driver Driver::findNearest	191.12ms
redis FindDriverIDs	24.38ms
redis GetDriver	11.61ms
❗ redis GetDriver	29.97ms
redis GetDriver	10.46ms
redis GetDriver	11.9ms
redis GetDriver	11.46ms
redis GetDriver	10.96ms
redis GetDriver	8.89ms
redis GetDriver	9.93ms
redis GetDriver	16.52ms

Trace timeline – A blocking operation

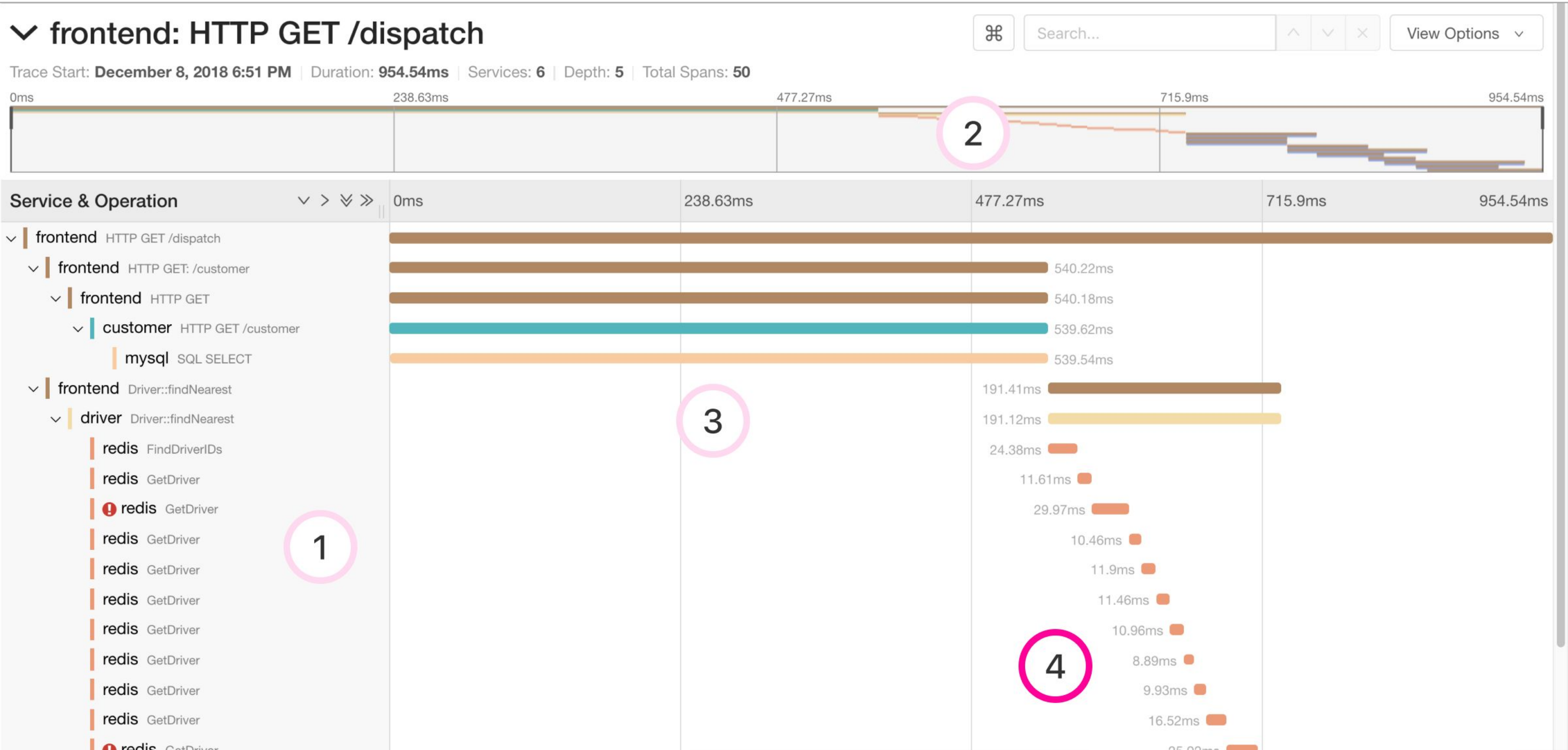


frontend: HTTP GET /dispatch

Trace Start: **December 8, 2018 6:51 PM** | Duration: **954.54ms** | Services: **6** | Depth: **5** | Total Spans: **50**

Service & Operation	Duration
frontend HTTP GET /dispatch	
frontend HTTP GET: /customer	540.22ms
frontend HTTP GET	540.18ms
customer HTTP GET /customer	539.62ms
mysql SQL SELECT	539.54ms
frontend Driver::findNearest	191.41ms
driver Driver::findNearest	191.12ms
redis FindDriverIDs	24.38ms
redis GetDriver	11.61ms
redis GetDriver	29.97ms
redis GetDriver	10.46ms
redis GetDriver	11.9ms
redis GetDriver	11.46ms
redis GetDriver	10.96ms
redis GetDriver	8.89ms
redis GetDriver	9.93ms
redis GetDriver	16.52ms

Trace timeline – Sequential operations

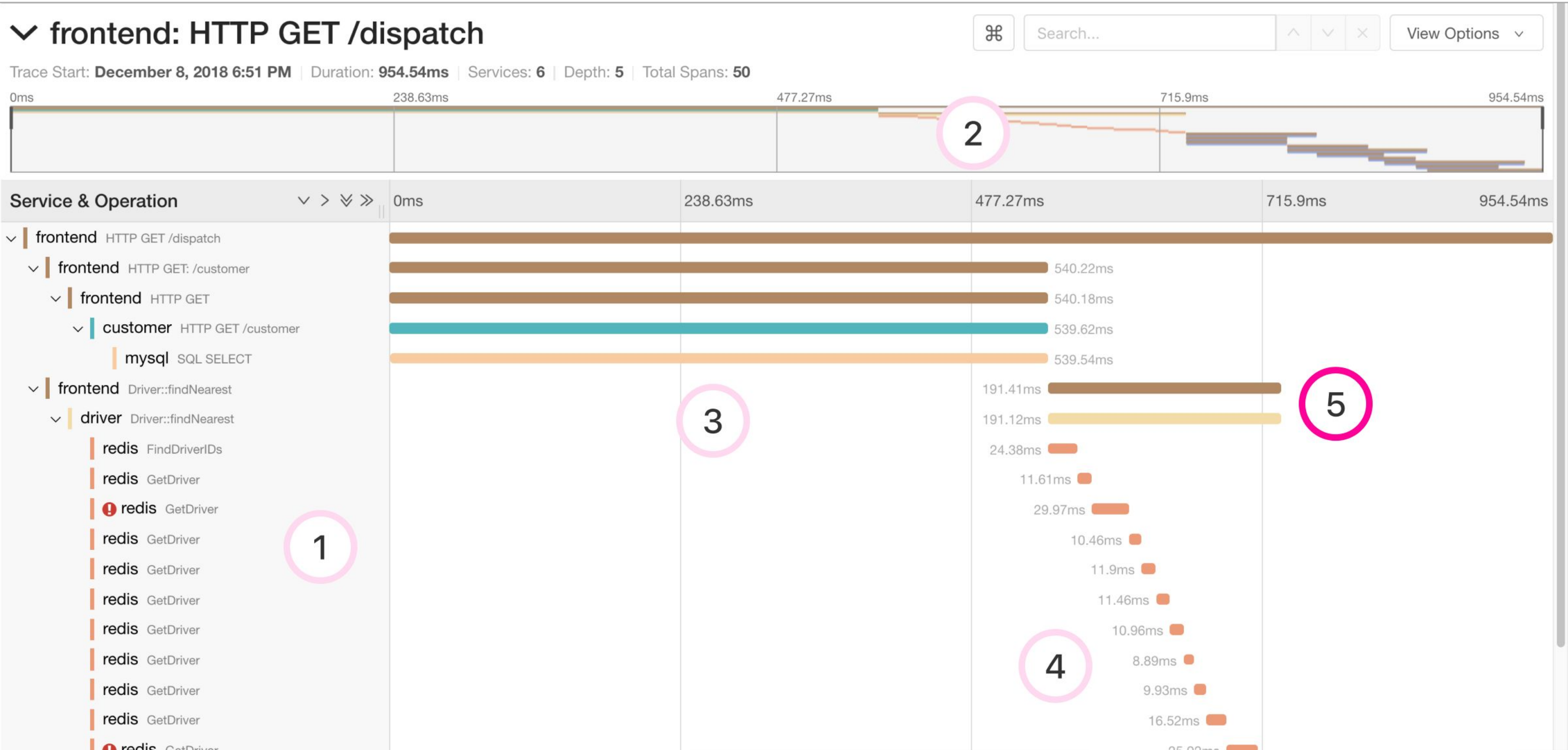


frontend: HTTP GET /dispatch

Trace Start: **December 8, 2018 6:51 PM** | Duration: **954.54ms** | Services: **6** | Depth: **5** | Total Spans: **50**

Service & Operation	Duration
frontend HTTP GET /dispatch	
frontend HTTP GET: /customer	540.22ms
frontend HTTP GET	540.18ms
customer HTTP GET /customer	539.62ms
mysql SQL SELECT	539.54ms
frontend Driver::findNearest	191.41ms
driver Driver::findNearest	191.12ms
redis FindDriverIDs	24.38ms
redis GetDriver	11.61ms
❗ redis GetDriver	29.97ms
redis GetDriver	10.46ms
redis GetDriver	11.9ms
redis GetDriver	11.46ms
redis GetDriver	10.96ms
redis GetDriver	8.89ms
redis GetDriver	9.93ms
redis GetDriver	16.52ms
❗ redis GetDriver	

Trace timeline – Parents encompass descendents (generally)



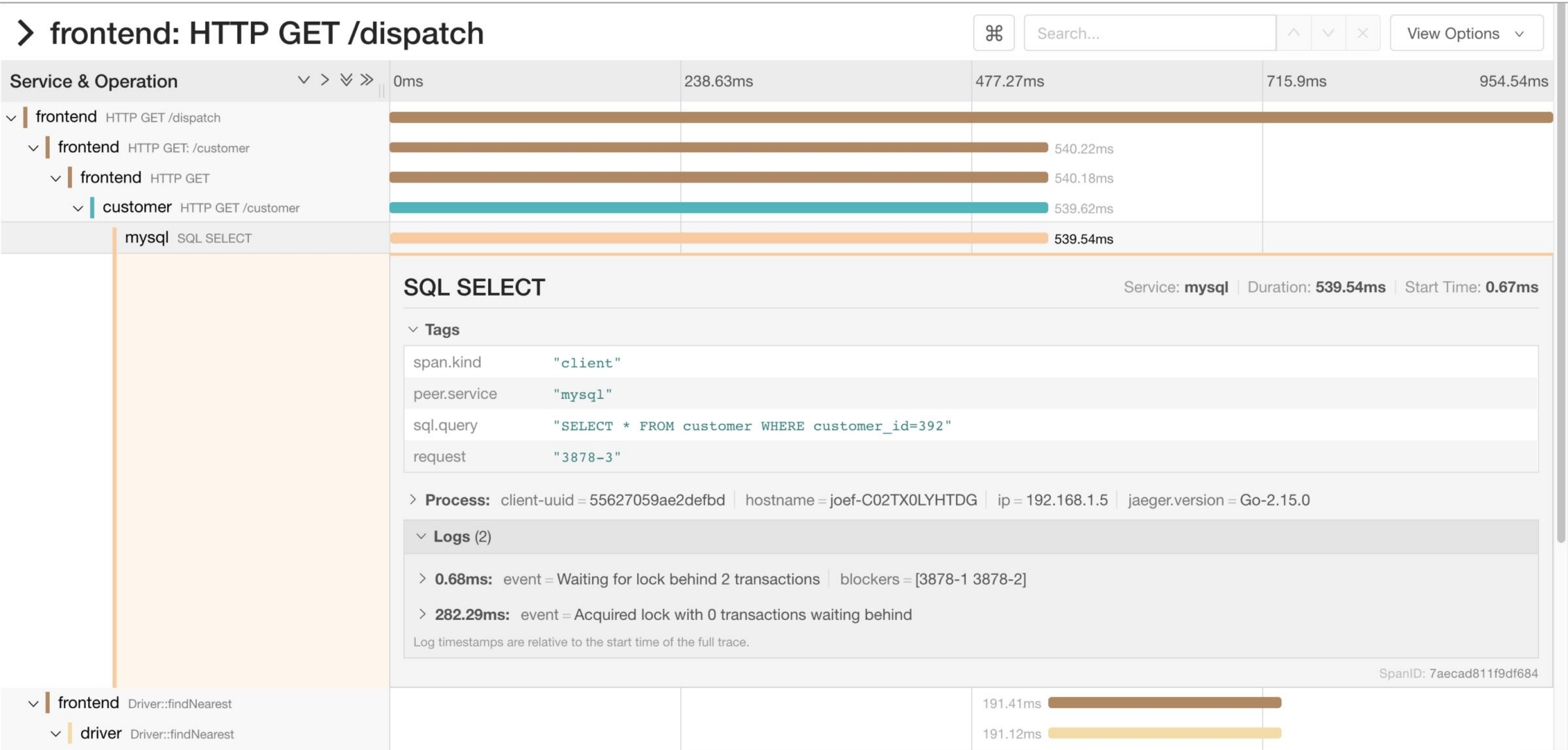
frontend: HTTP GET /dispatch

Trace Start: **December 8, 2018 6:51 PM** | Duration: **954.54ms** | Services: **6** | Depth: **5** | Total Spans: **50**

0ms 238.63ms 477.27ms 715.9ms 954.54ms

Service & Operation	0ms 238.63ms 477.27ms 715.9ms 954.54ms
frontend HTTP GET /dispatch	
frontend HTTP GET: /customer	540.22ms
frontend HTTP GET	540.18ms
customer HTTP GET /customer	539.62ms
mysql SQL SELECT	539.54ms
frontend Driver::findNearest	191.41ms
driver Driver::findNearest	191.12ms
redis FindDriverIDs	24.38ms
redis GetDriver	11.61ms
redis GetDriver	29.97ms
redis GetDriver	10.46ms
redis GetDriver	11.9ms
redis GetDriver	11.46ms
redis GetDriver	10.96ms
redis GetDriver	8.89ms
redis GetDriver	9.93ms
redis GetDriver	16.52ms
redis GetDriver	[illegible]

Span details



frontend: HTTP GET /dispatch

frontend HTTP GET /dispatch
frontend HTTP GET: /customer 540.22ms
frontend HTTP GET 540.18ms
customer HTTP GET /customer 539.62ms
mysql SQL SELECT 539.54ms

SQL SELECT

Service: **mysql** | Duration: **539.54ms** | Start Time: **0.67ms**

Tags

span.kind	"client"
peer.service	"mysql"
sql.query	"SELECT * FROM customer WHERE customer_id=392"
request	"3878-3"

Process: client-uuid = 55627059ae2defbd | hostname = joef-C02TX0LYHTDG | ip = 192.168.1.5 | jaeger.version = Go-2.15.0

Logs (2)

0.68ms: event = Waiting for lock behind 2 transactions | blockers = [3878-1 3878-2]

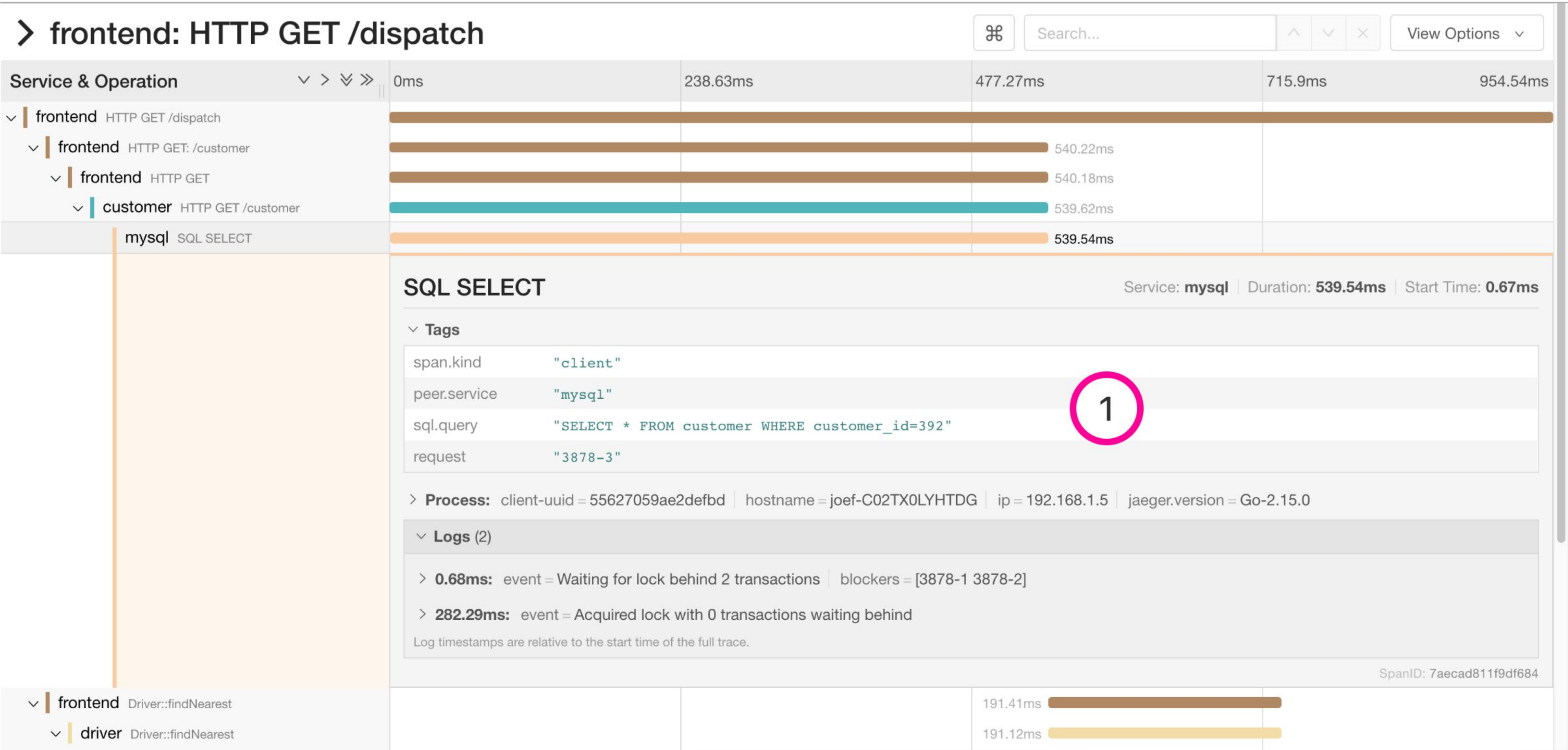
282.29ms: event = Acquired lock with 0 transactions waiting behind

Log timestamps are relative to the start time of the full trace.

SpanID: 7aecad811f9df684

frontend Driver::findNearest 191.41ms
driver Driver::findNearest 191.12ms

Span details – Database query



frontend: HTTP GET /dispatch

```
frontend HTTP GET /dispatch
  frontend HTTP GET: /customer      540.22ms
    frontend HTTP GET               540.18ms
      customer HTTP GET /customer   539.62ms
        mysql SQL SELECT            539.54ms
```

SQL SELECT

Service: **mysql** | Duration: **539.54ms** | Start Time: **0.67ms**

Tags

span.kind	"client"
peer.service	"mysql"
sql.query	"SELECT * FROM customer WHERE customer_id=392"
request	"3878-3"

Process: client-uuid = 55627059ae2defbd | hostname = joef-C02TX0LYHTDG | ip = 192.168.1.5 | jaeger.version = Go-2.15.0

Logs (2)

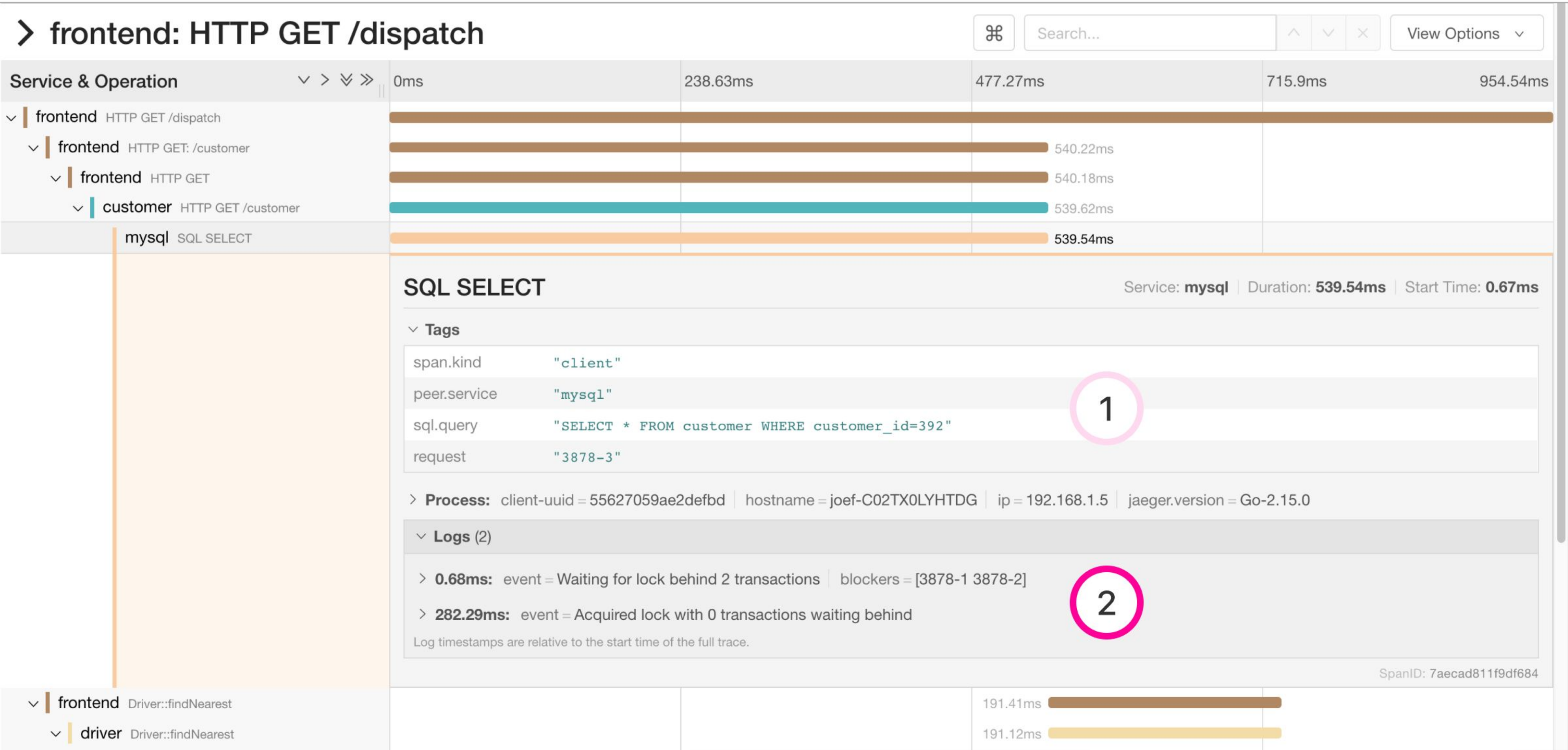
```
0.68ms: event = Waiting for lock behind 2 transactions | blockers = [3878-1 3878-2]
282.29ms: event = Acquired lock with 0 transactions waiting behind
```

Log timestamps are relative to the start time of the full trace.

SpanID: 7aecad811f9df684

```
  frontend Driver::findNearest      191.41ms
    driver Driver::findNearest      191.12ms
```

Span details – Lock contention



frontend: HTTP GET /dispatch

Service & Operation	Duration
frontend HTTP GET /dispatch	
frontend HTTP GET: /customer	540.22ms
frontend HTTP GET	540.18ms
customer HTTP GET /customer	539.62ms
mysql SQL SELECT	539.54ms
frontend Driver::findNearest	191.41ms
driver Driver::findNearest	191.12ms

SQL SELECT

Service: **mysql** | Duration: **539.54ms** | Start Time: **0.67ms**

Tags

span.kind	"client"
peer.service	"mysql"
sql.query	"SELECT * FROM customer WHERE customer_id=392"
request	"3878-3"

Process: client-uuid = 55627059ae2defbd | hostname = joef-C02TX0LYHTDG | ip = 192.168.1.5 | jaeger.version = Go-2.15.0

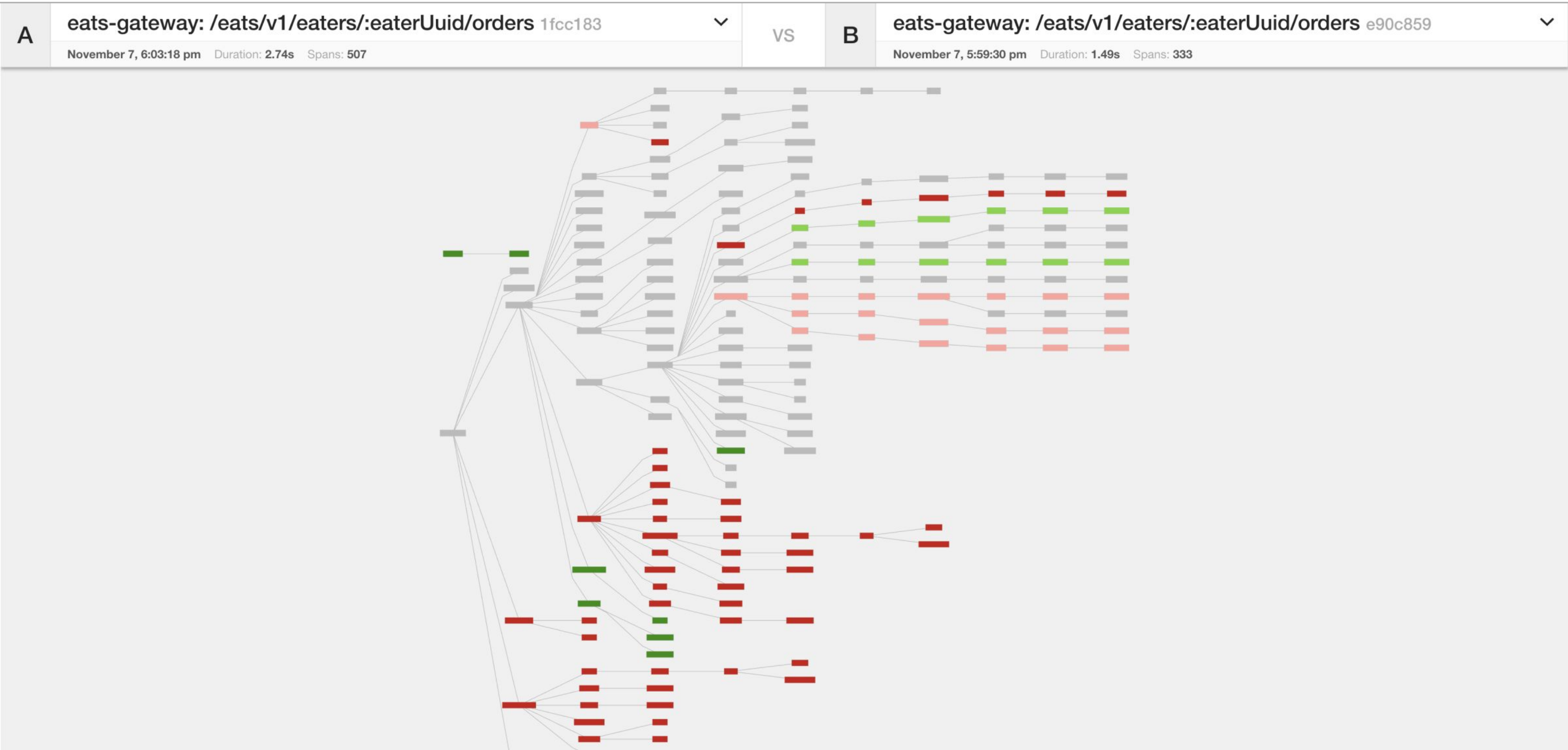
Logs (2)

```
0.68ms: event = Waiting for lock behind 2 transactions | blockers = [3878-1 3878-2]
282.29ms: event = Acquired lock with 0 transactions waiting behind
```

Log timestamps are relative to the start time of the full trace.

SpanID: 7aecad811f9df684

Comparing trace structures – Unified diff

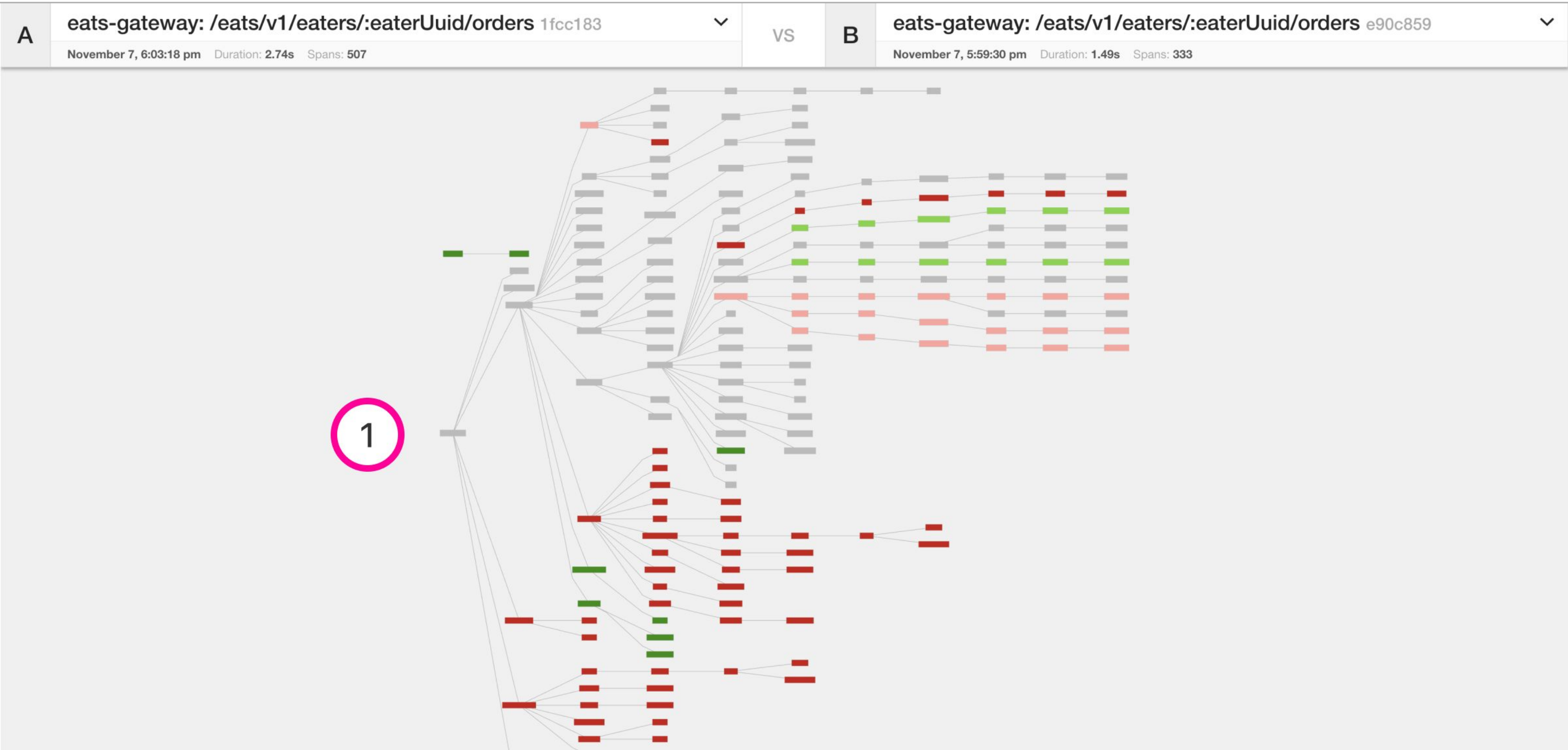


A | eats-gateway: /eats/v1/eaters/:eaterUuid/orders 1fcc183
November 7, 6:03:18 pm Duration: 2.74s Spans: 507

VS

B | eats-gateway: /eats/v1/eaters/:eaterUuid/orders e90c859
November 7, 5:59:30 pm Duration: 1.49s Spans: 333

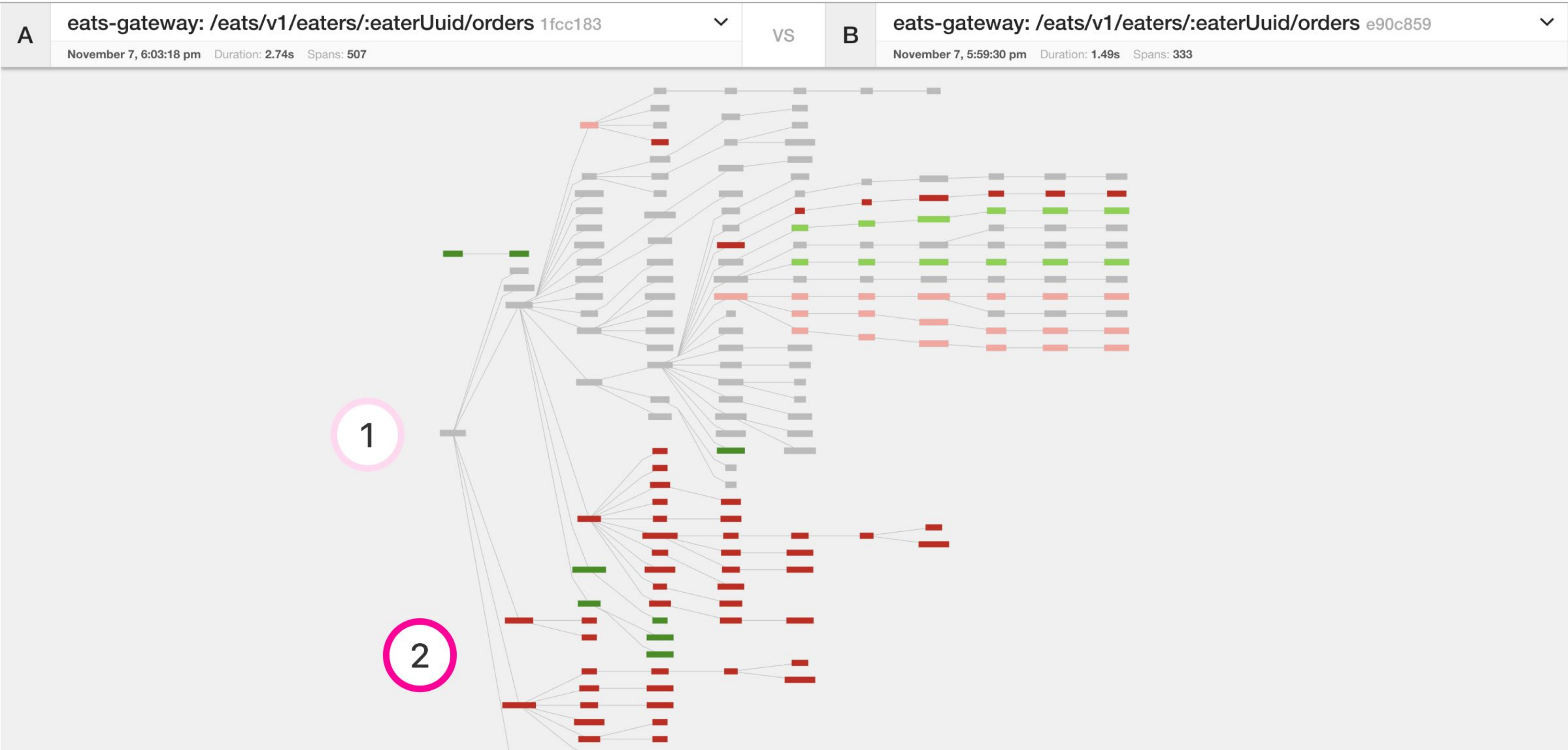
Comparing trace structures – Shared structure


A | eats-gateway: /eats/v1/eaters/:eaterUuid/orders 1fcc183
November 7, 6:03:18 pm Duration: 2.74s Spans: 507

VS

B | eats-gateway: /eats/v1/eaters/:eaterUuid/orders e90c859
November 7, 5:59:30 pm Duration: 1.49s Spans: 333

Comparing trace structures – Absent in one or the traces

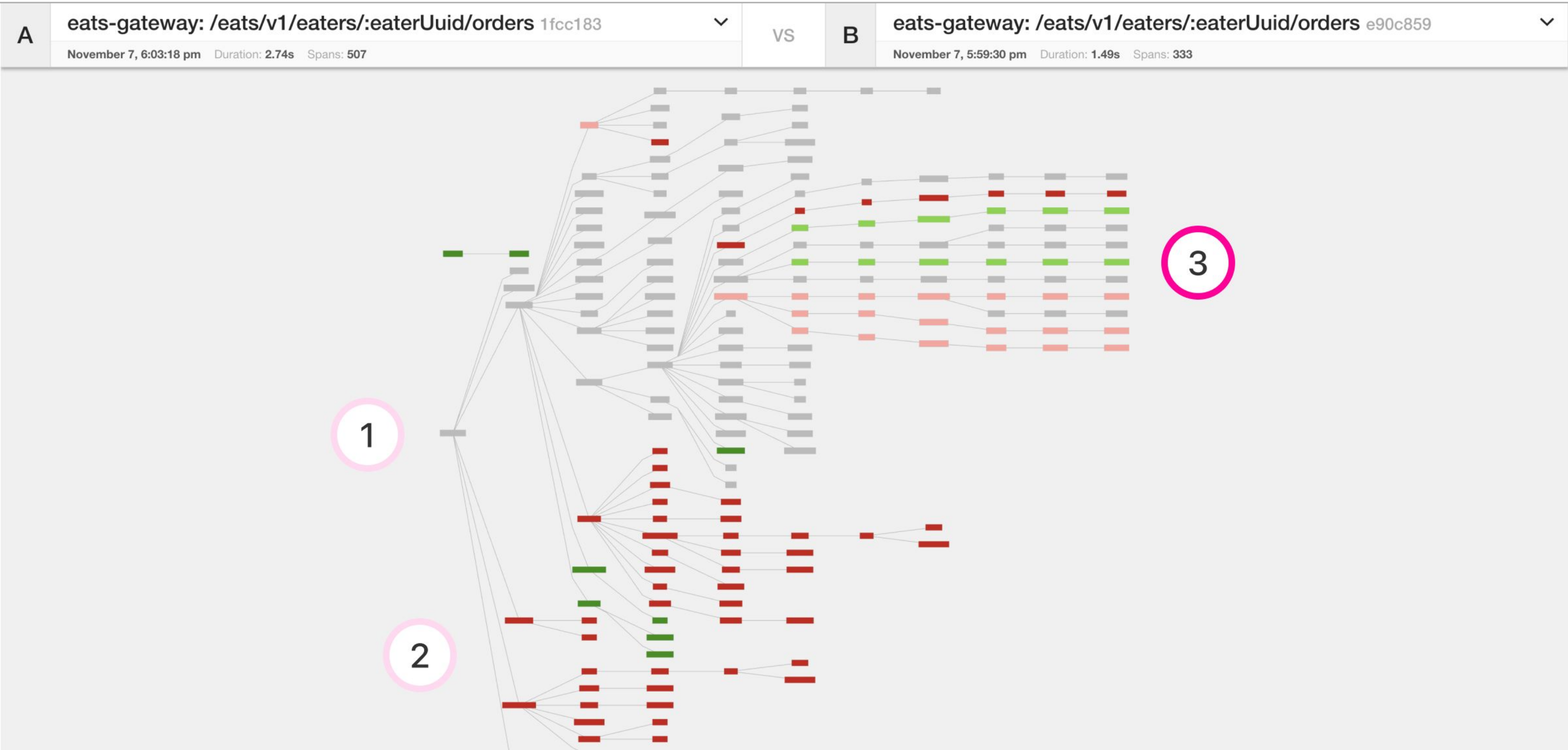


A | eats-gateway: /eats/v1/eaters/:eaterUuid/orders 1fcc183
November 7, 6:03:18 pm Duration: 2.74s Spans: 507

VS

B | eats-gateway: /eats/v1/eaters/:eaterUuid/orders e90c859
November 7, 5:59:30 pm Duration: 1.49s Spans: 333

Comparing trace structures – More or less within a node

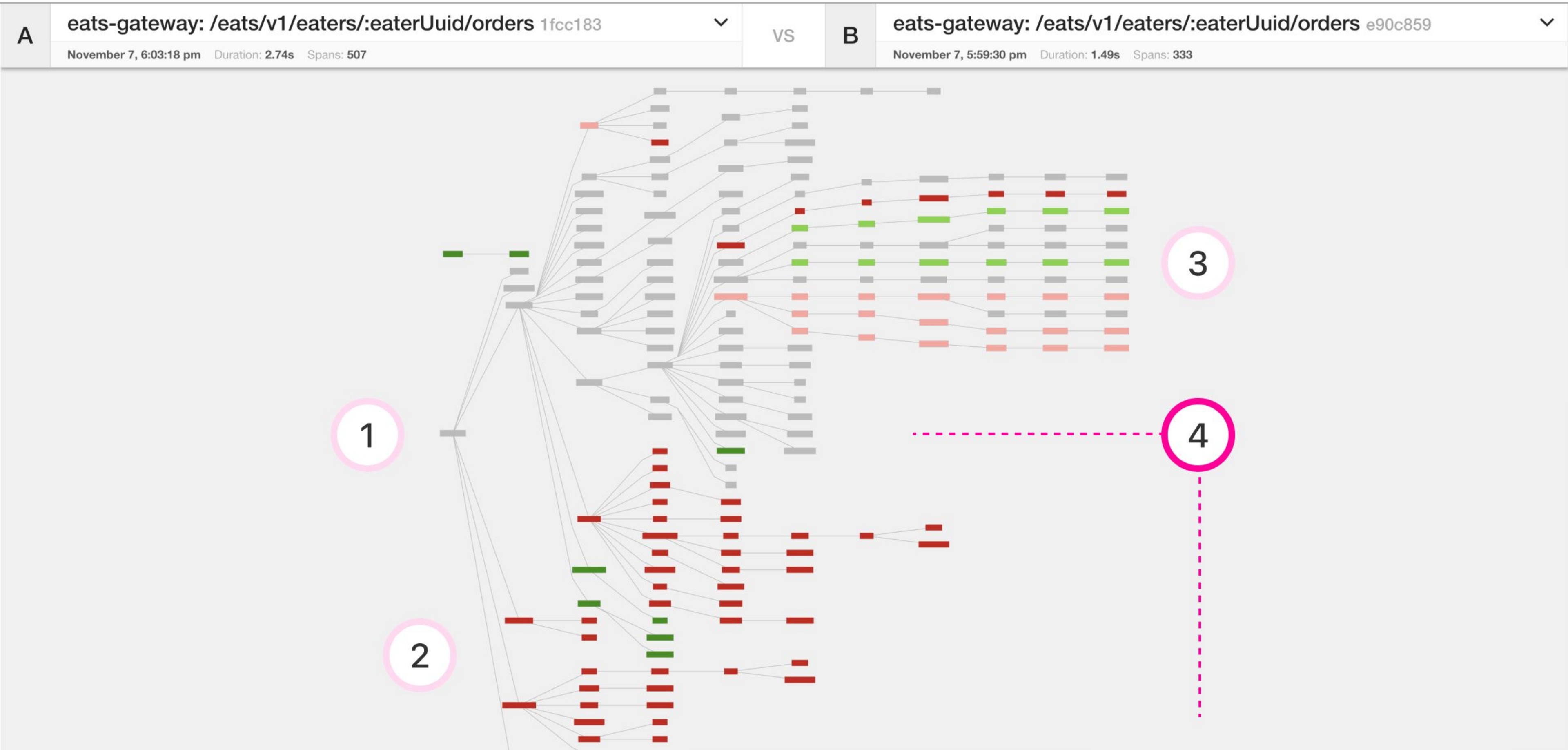


A | eats-gateway: /eats/v1/eaters/:eaterUuid/orders 1fcc183
November 7, 6:03:18 pm Duration: 2.74s Spans: 507

VS

B | eats-gateway: /eats/v1/eaters/:eaterUuid/orders e90c859
November 7, 5:59:30 pm Duration: 1.49s Spans: 333

Comparing trace structures – Substantial divergence

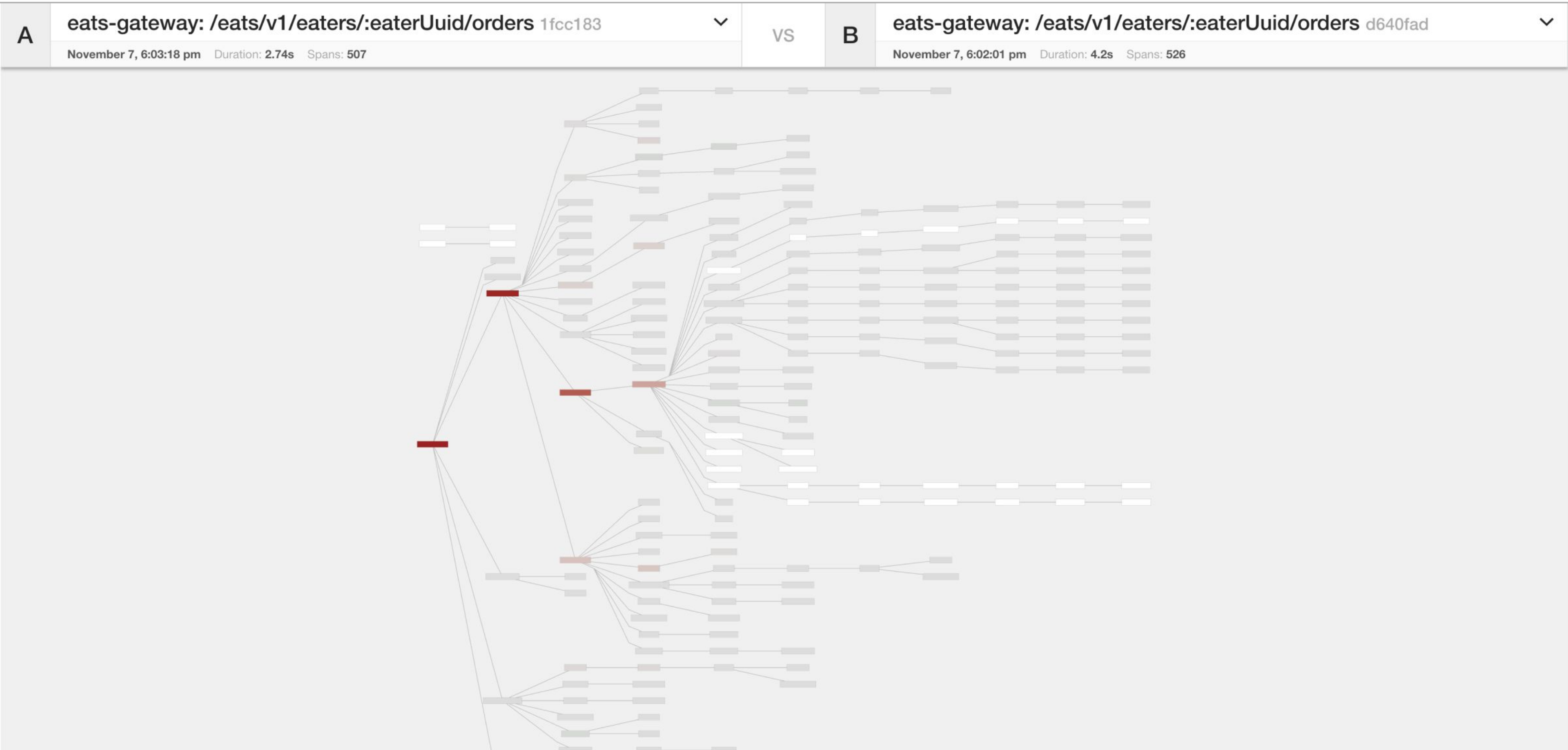


A eats-gateway: /eats/v1/eaters/:eaterUuid/orders 1fcc183
November 7, 6:03:18 pm Duration: 2.74s Spans: 507

VS

B eats-gateway: /eats/v1/eaters/:eaterUuid/orders e90c859
November 7, 5:59:30 pm Duration: 1.49s Spans: 333

Comparing span durations

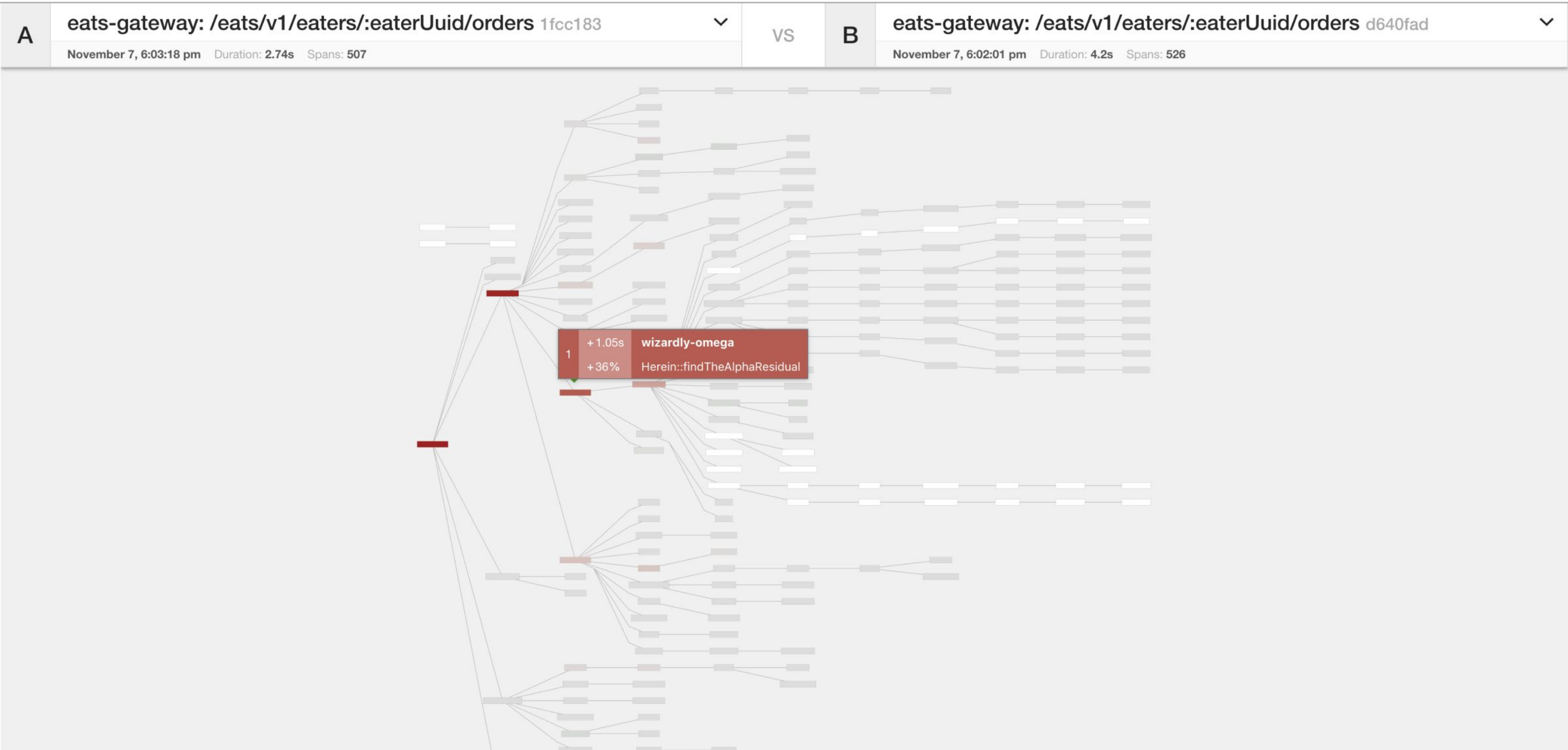


A eats-gateway: /eats/v1/eaters/:eaterUuid/orders 1fcc183
November 7, 6:03:18 pm Duration: 2.74s Spans: 507

VS

B eats-gateway: /eats/v1/eaters/:eaterUuid/orders d640fad
November 7, 6:02:01 pm Duration: 4.2s Spans: 526

Comparing span durations



A eats-gateway: /eats/v1/eaters/:eaterUuid/orders 1fcc183

November 7, 6:03:18 pm Duration: 2.74s Spans: 507

VS

B eats-gateway: /eats/v1/eaters/:eaterUuid/orders d640fad

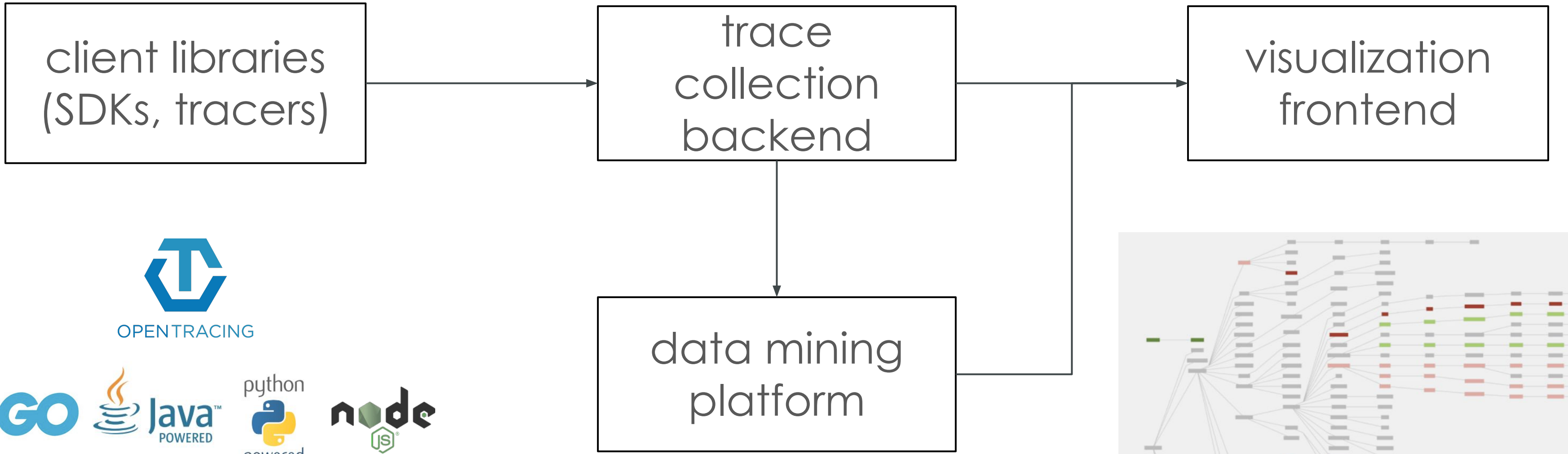
November 7, 6:02:01 pm Duration: 4.2s Spans: 526

CLOUD NATIVE
COMPUTING FOUNDATION

Jaeger

Architecture

Jaeger, a Distributed Tracing Platform



Instrumentation not included

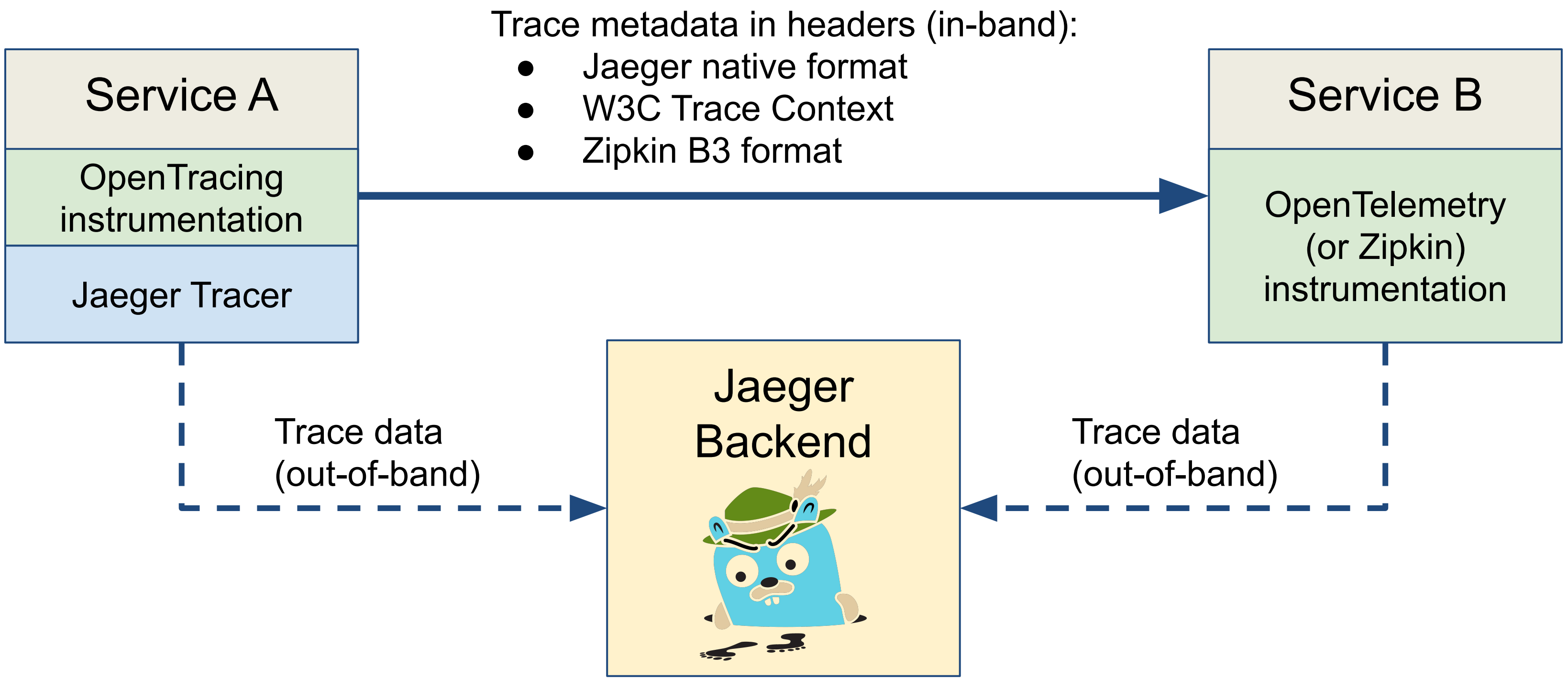
Jaeger project does not provide instrumentation!

Use OpenTracing or OpenTelemetry.

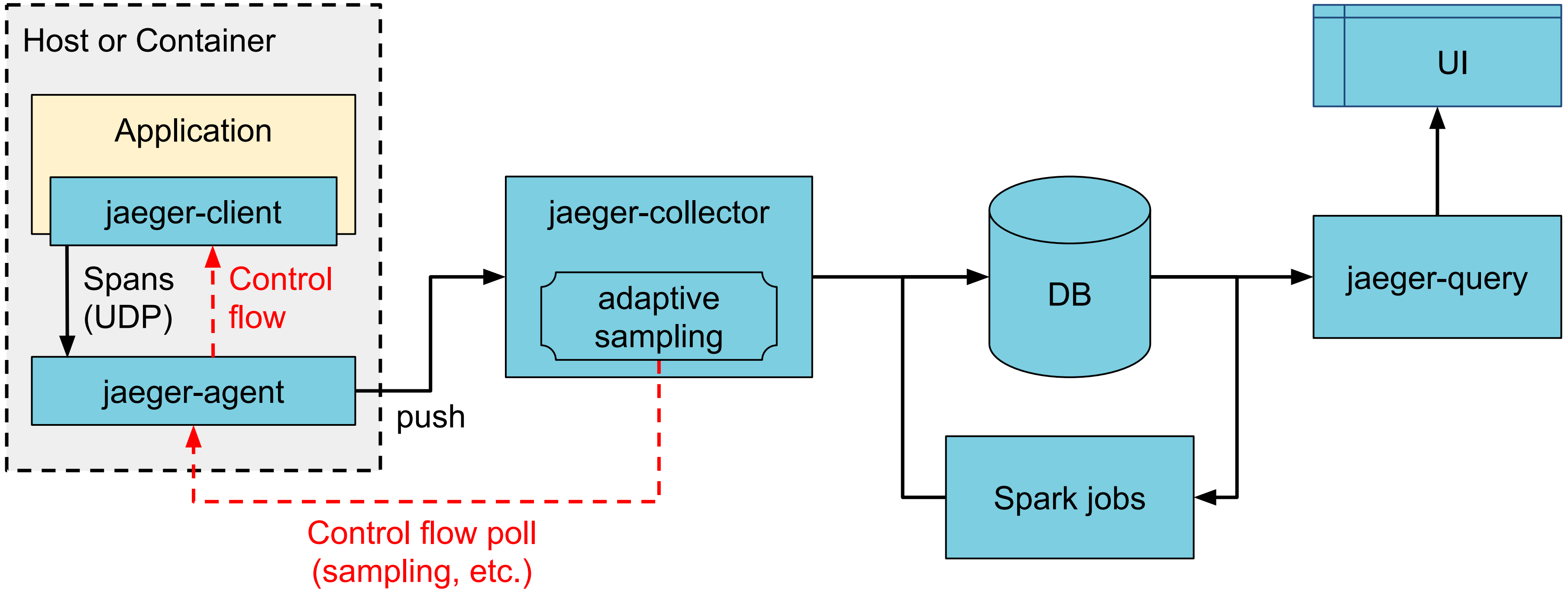
Jaeger - /ˈyāgər/, *noun*: hunter

- Inspired by Google's Dapper and OpenZipkin
- Created at Uber in August 2015 (blog)
- Open sourced in April 2017
- Joined CNCF in Sep 2017 (as incubating)
- Graduated to top-level CNCF project Oct 2019 (CNCF announcement)

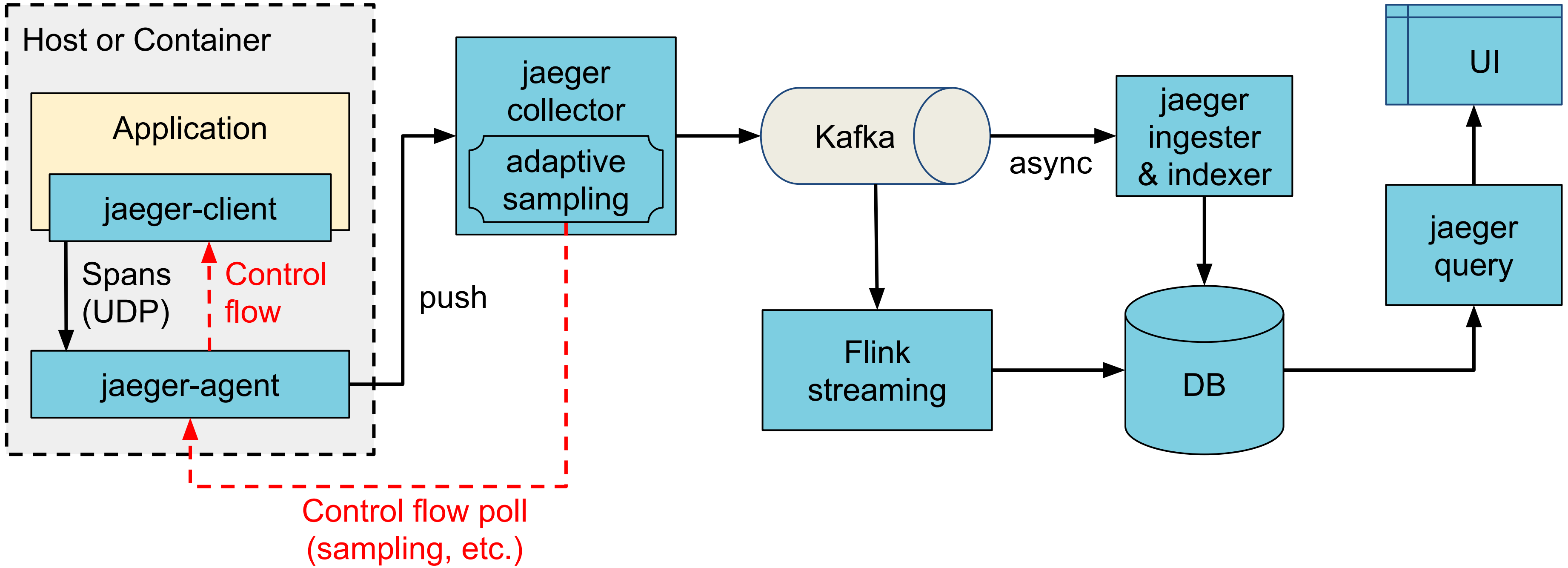
Jaeger and your system



Architecture 2017: Push



Architecture now: Push+Async+Streaming



Technology Stack

- Go backend
- Pluggable storage
  - Cassandra, Elasticsearch, badger, memory
- React/Javascript frontend
- OpenTracing Instrumentation libraries
- Integration with Kafka, Apache Flink

CLOUD NATIVE
COMPUTING FOUNDATION

Jaeger

And Sampling

Sampling

Sampling is the selection of a subset (a statistical sample) of individuals from within a (statistical) population to estimate characteristics of the whole population.

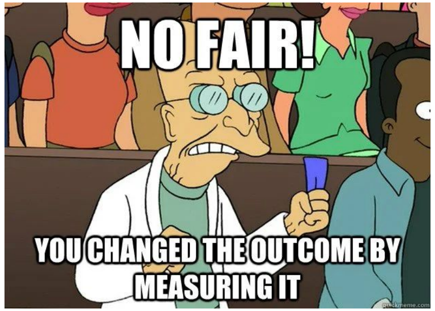
traces

all possible traces

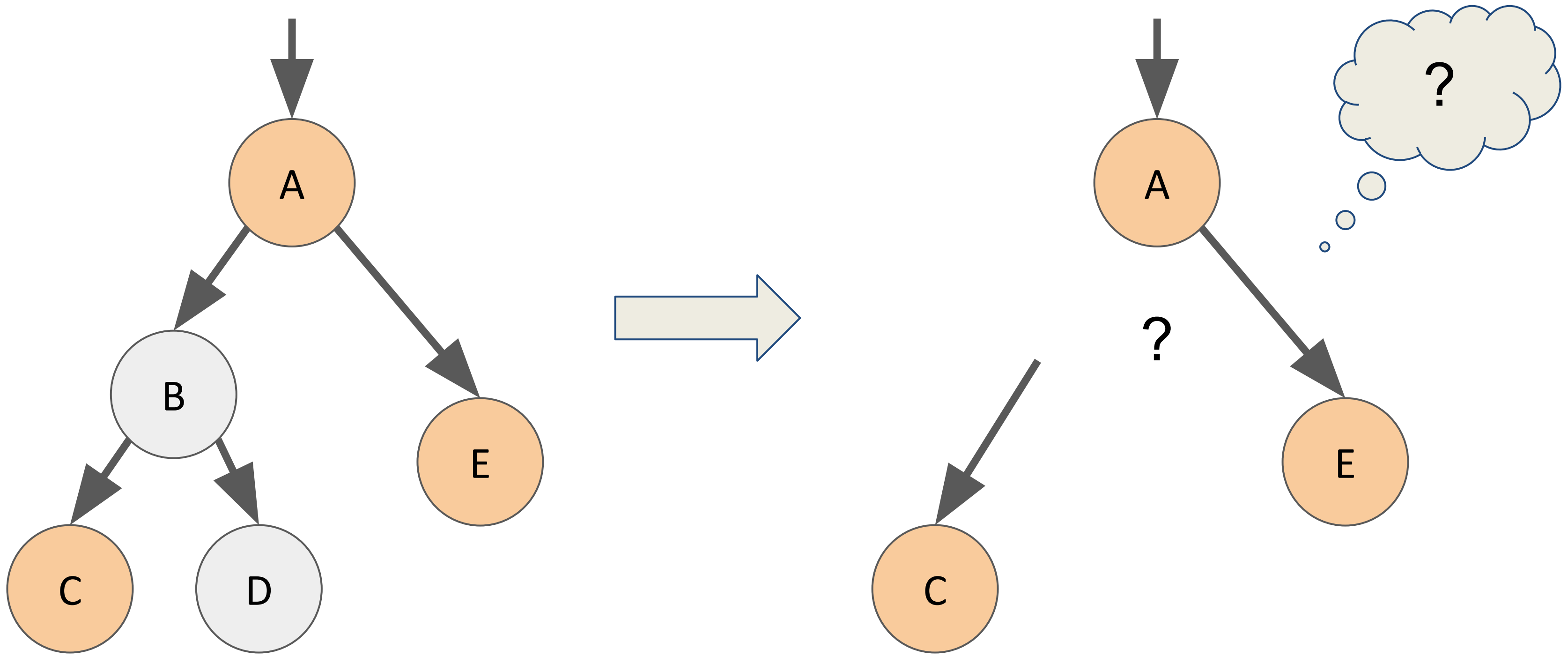
reason about application performance

Why do we sample

1. Saving everything incurs large storage costs
   - 2 KB / span on `10k` QPS server ⇒ `20` MB/s
   - `x100` instances ⇒ 2 GB/s ≅ 170 PB/day (for one service!)
2. Performance overhead from instrumentation
   - 10k QPS server ⇒ `100µs` / req budget
   - Trace instrumentation: `5µs` ⇒ 5% overhead
3. Trace data is very repetitive



Goal: Consistent (all or nothing) Sampling



Sampling techniques

- Head-based sampling
  - Most popular in the industry
- Tail-based sampling
  - Gaining popularity recently

Head-based (upfront) sampling

Sampling decision is made at the start of the trace and propagated in the trace context.

- ✓ Minimal perf overhead when trace is not sampled
- ✓ Easy to implement, supported by Jaeger SDKs
- ❌ Can easily miss rare anomalies/outliers
  - Prob. of catching p99 latency with 1% sampling rate ⇒ 1/10,000
- ❌ Cannot "sample on errors"

Head-based sampling in Jaeger

- SDKs can be configured with different samplers (always on / off, probabilistic, rate limiting, etc.)
  - ✓ Easy to implement
  - ❌ Spread-out configuration in the hands of developers
- SDKs default to "remote" sampler that allows centralized configuration (polled from collectors)

Jaeger sampling configuration

Applies to all other services

Custom strategy per service

```
"default_strategy": {
  "type": "probabilistic",
  "param": 0.5,
  "operation_strategies": [
    {
      "operation": "/health",
      "type": "probabilistic",
      "param": 0.0
    },
    {
      "operation": "/metrics",
      "type": "probabilistic",
      "param": 0.0
    }
  ]
}
```

Overrides for specific endpoints

```
"service_strategies": [
  {
    "service": "foo",
    "type": "probabilistic",
    "param": 0.8,
    "operation_strategies": [
      {
        "operation": "bar",
        "type": "probabilistic",
        "param": 0.2
      }
    ]
  }
]
```

Overrides for specific endpoints

CLOUD NATIVE
COMPUTING FOUNDATION

Tail-based (post-trace) sampling

Sampling decision is made at the end of the trace:

- ✓ Can be much more intelligent, based on observed latency, errors, unusual call patterns & graph shapes, etc.
- ✓ Can catch anomalies
- ✓ Can perform aggregations before sampling
- ✗ Requires temporary storage of all traces
- ✗ Applications incur performance overhead even for traces that may be later discarded

Tail-based sampling in Jaeger

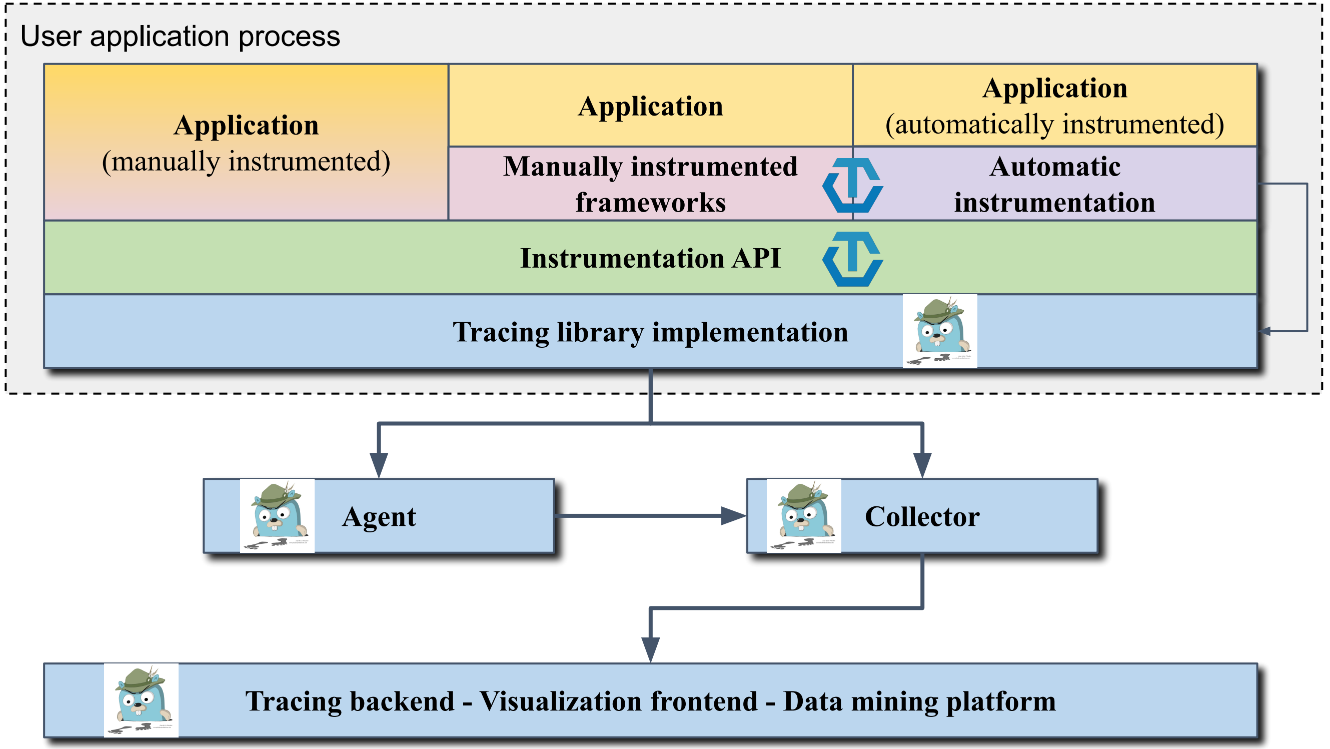
- ✔️ Supported in `jaeger-opentelemetry-collector`
- ✔️ Configurable sampling rules: latency, certain tags
- ❌ Single-node mode only, multi-node sharded solution will be available in the future

CLOUD NATIVE COMPUTING FOUNDATION

Jaeger

And OpenTelemetry

Jaeger with OpenTracing



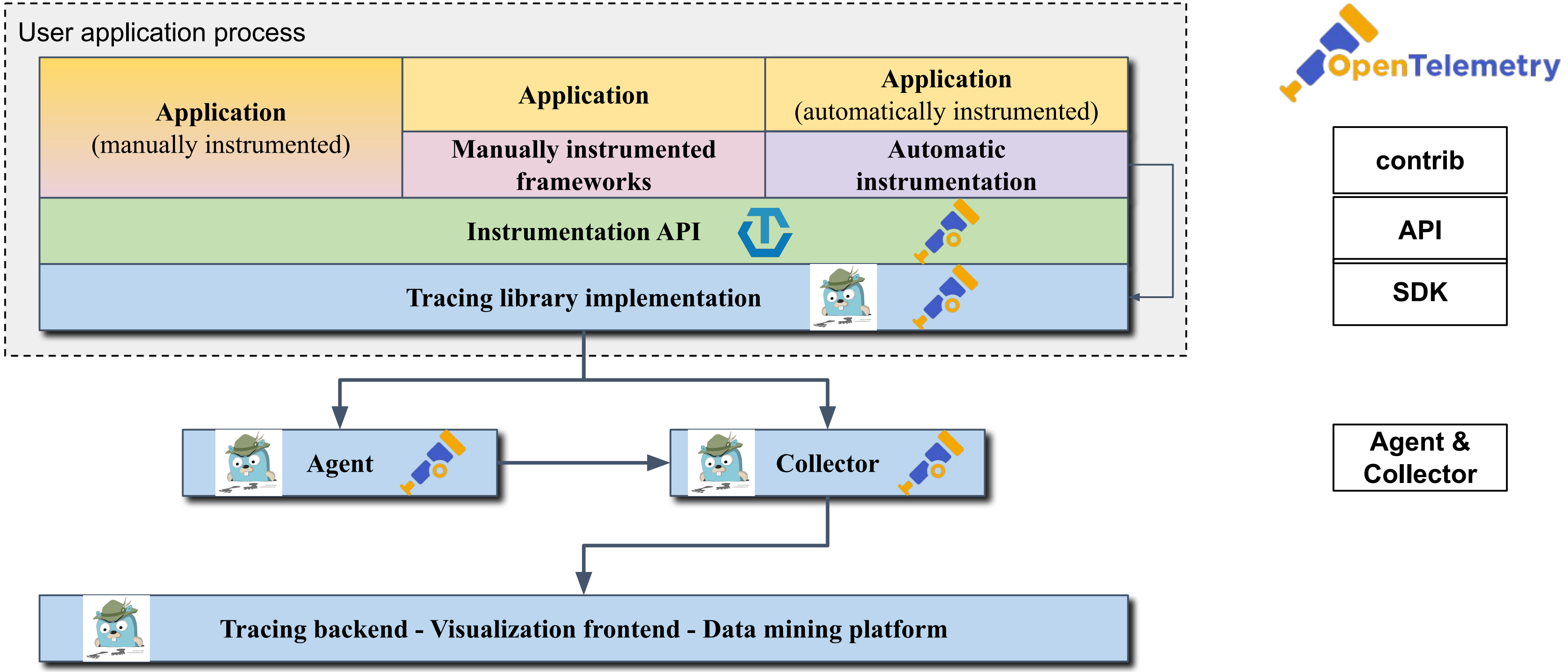
User application process
Application (manually instrumented)
Application
Application (automatically instrumented)
Manually instrumented frameworks
Automatic instrumentation
Instrumentation API
Tracing library implementation
Agent
Collector
Tracing backend - Visualization frontend - Data mining platform

OPENTRACING

opentracing contrib

API

Jaeger with OpenTelemetry


User application process
Application
(manually instrumented)
Application
Manually instrumented frameworks
Application
(automatically instrumented)
Automatic instrumentation
Instrumentation API
Tracing library implementation
Agent
Collector
Tracing backend - Visualization frontend - Data mining platform
OpenTelemetry
contrib
API
SDK
Agent & Collector

Jaeger components on OpenTelemetry

- OpenTelemetry Collector is written in Go
- We built Jaeger-specific versions
  - Have the same capabilities as upstream OTel
  - With Jaeger extensions, e.g. storage
- We're converting Jaeger storage implementation to OTel data model for better compatibility

CLOUD NATIVE
COMPUTING FOUNDATION

Jaeger

And OpenTelemetry SDKs

Jaeger and OpenTelemetry SDKs

- OpenTelemetry SDK support
  - Jaeger gRPC exporter
  - Jaeger propagation
- OpenTracing SHIM
  - use OTel SKD with OpenTracing instrumentations
- Jaeger client libraries support W3C Trace Context

CLOUD NATIVE
COMPUTING FOUNDATION

Jaeger

And Kubernetes

Deploying Jaeger on Kubernetes

- Helm charts
- Jaeger Operator
  - allInOne and production deployment
  - auto provisioning of Kafka (Strimzi)
- Plain Kubernetes manifest files

Getting in Touch

- GitHub: https://github.com/jaegertracing
- Chat: https://gitter.im/jaegertracing/
- Mailing List - jaeger-tracing@googlegroups.com
- Blog: https://medium.com/jaegertracing
- Twitter: https://twitter.com/JaegerTracing
- Bi-Weekly Community Meetings