Conquering Microservices Complexity @Uber

With Distributed Tracing

Yuri Shkuro

SOFTWARE ENGINEER @ UBER

Agenda

Yuri Shkuro

Software Engineer
Uber Technologies

shkuro.com

Founder & Maintainer
of CNCF Jaeger

jaegertracing.io

Co-founder of
OpenTracing &
OpenTelemetry

Mastering
Distributed
Tracing

ing performance in microservi
lex systems

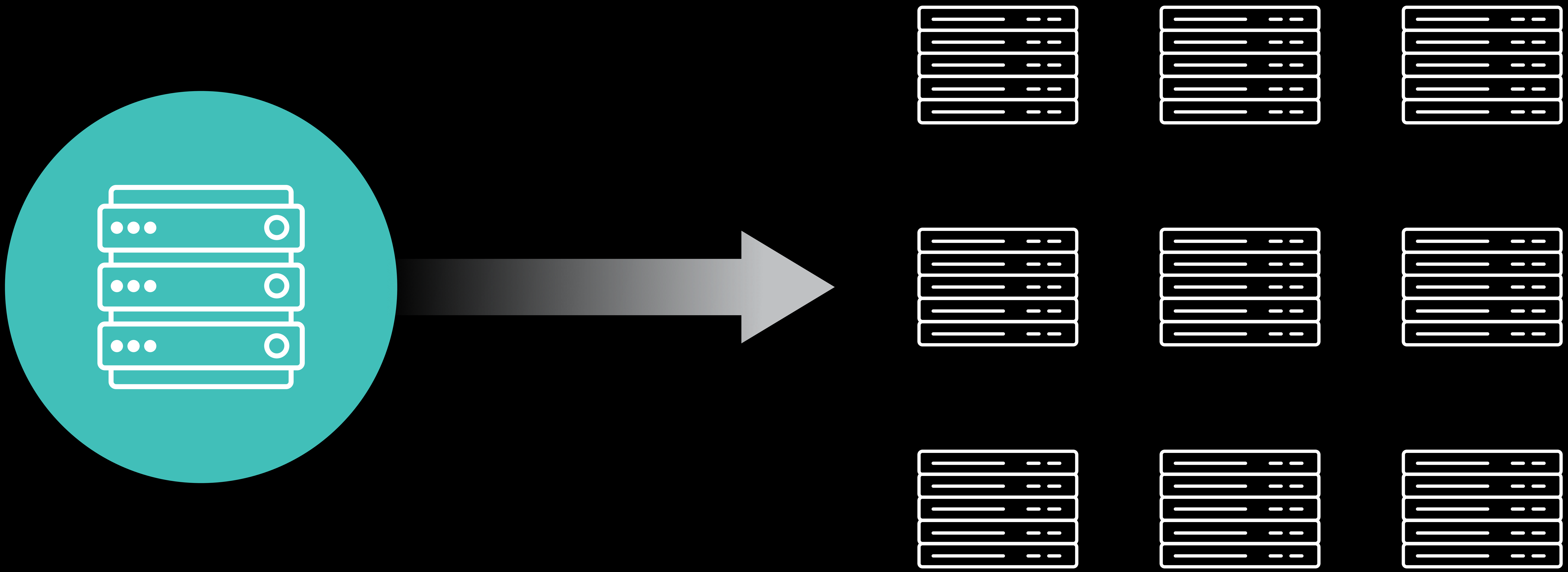
Author of "Mastering Distributed Tracing",
by Packt Publishing

Quick Poll

Why Distributed Tracing

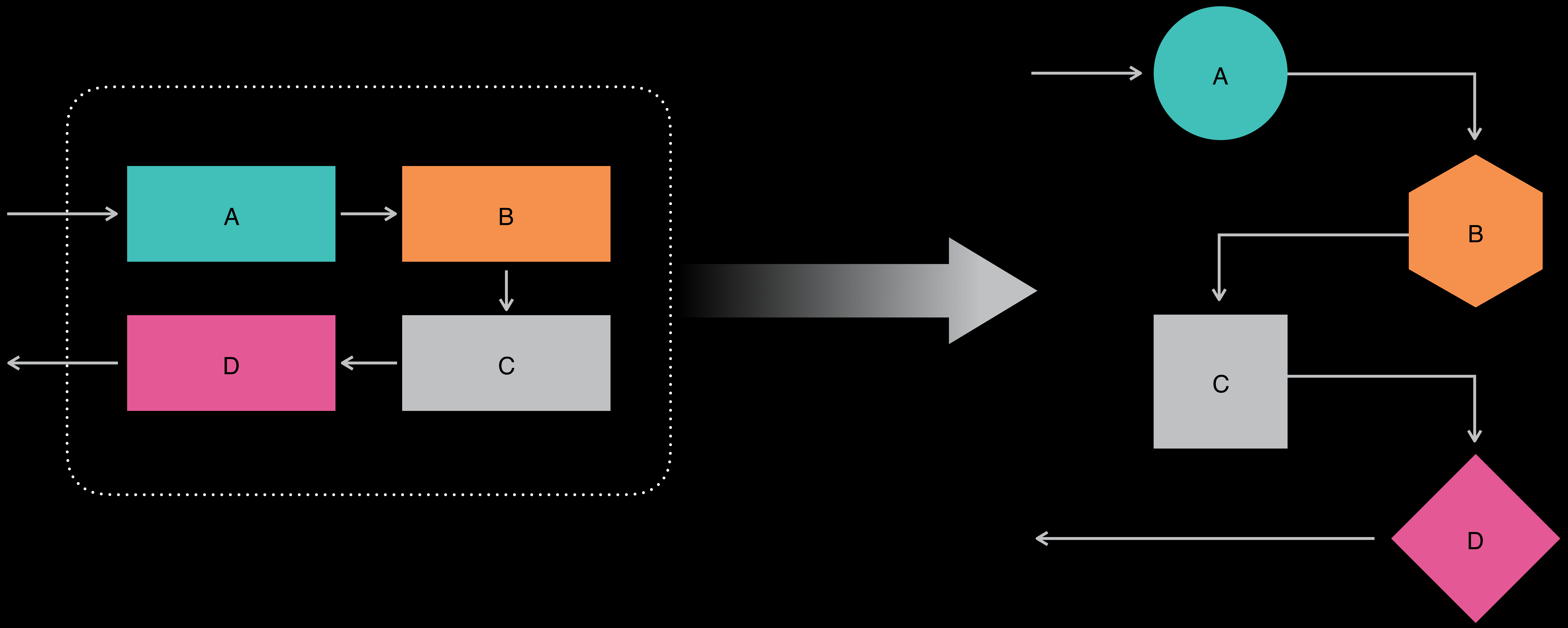
Scaling With Users

Distributed Systems



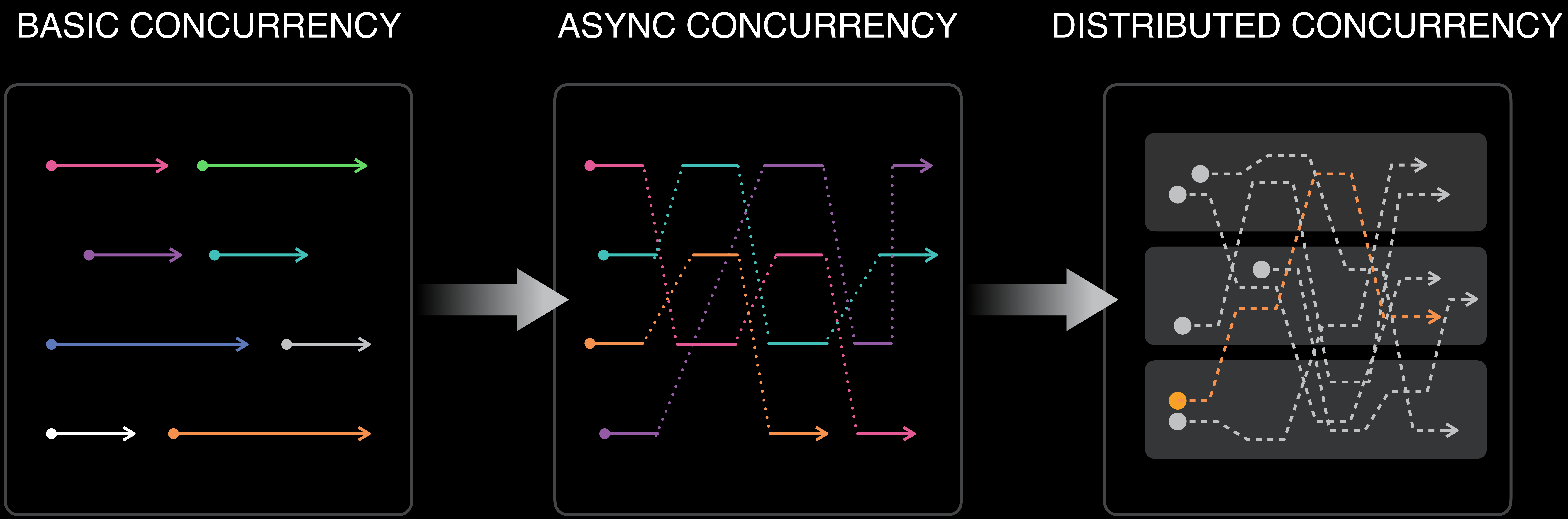
Scaling With Engineering Organization

Monoliths to Microservices



Scaling With CPU Cores

Asynchronous Programming Models, Distributed Concurrency



In microservices architectures
the number of failure modes
increases exponentially

Observability of
distributed transactions
is paramount!

Observability
vs.
monitoring

Observability vs. monitoring

Observability

System's ability to answer questions

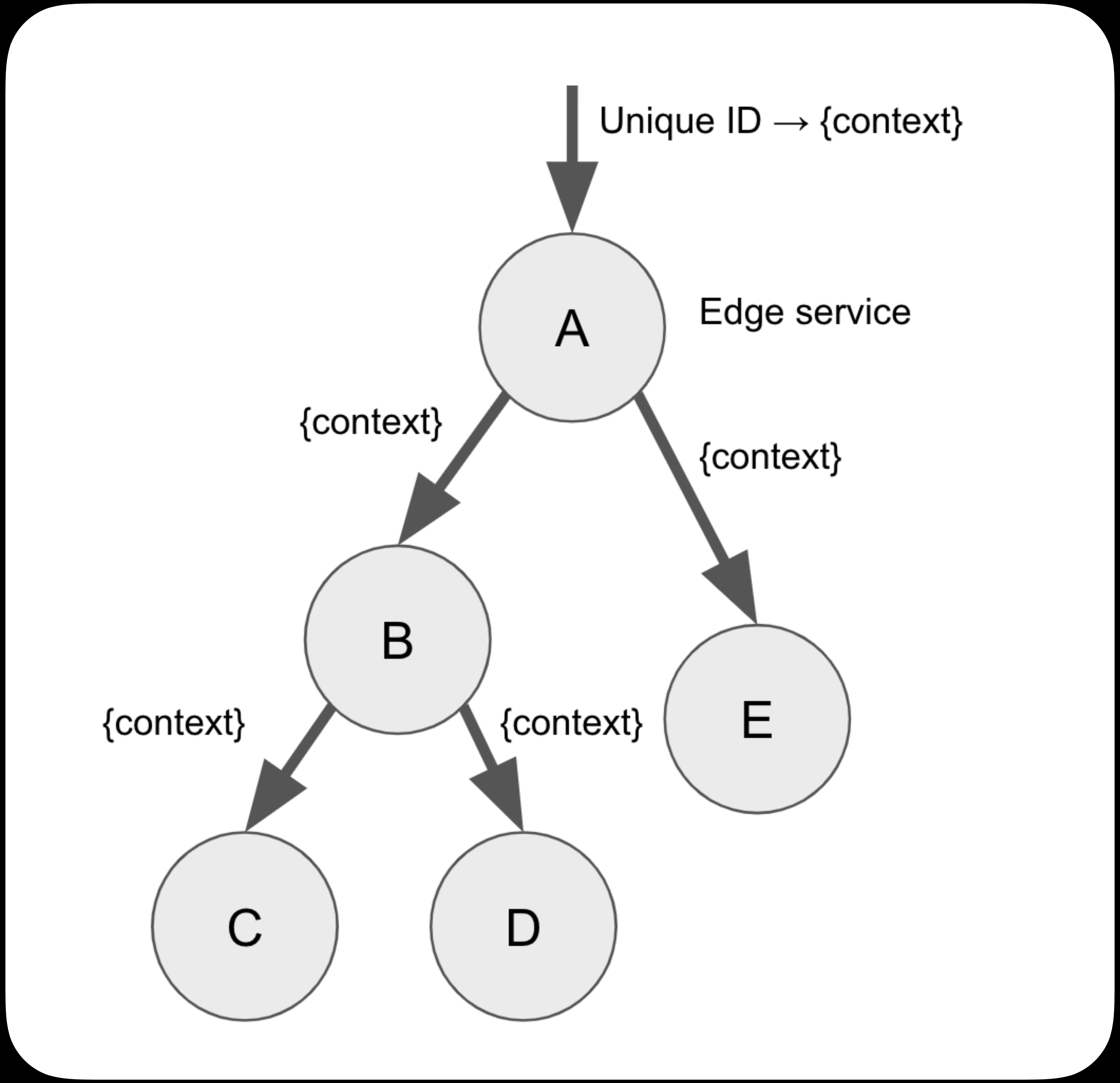
- Which services did the request go through
- What did every service do when processing the request
- If the request was slow, where were the bottlenecks
- If the request failed, where did the errors happen

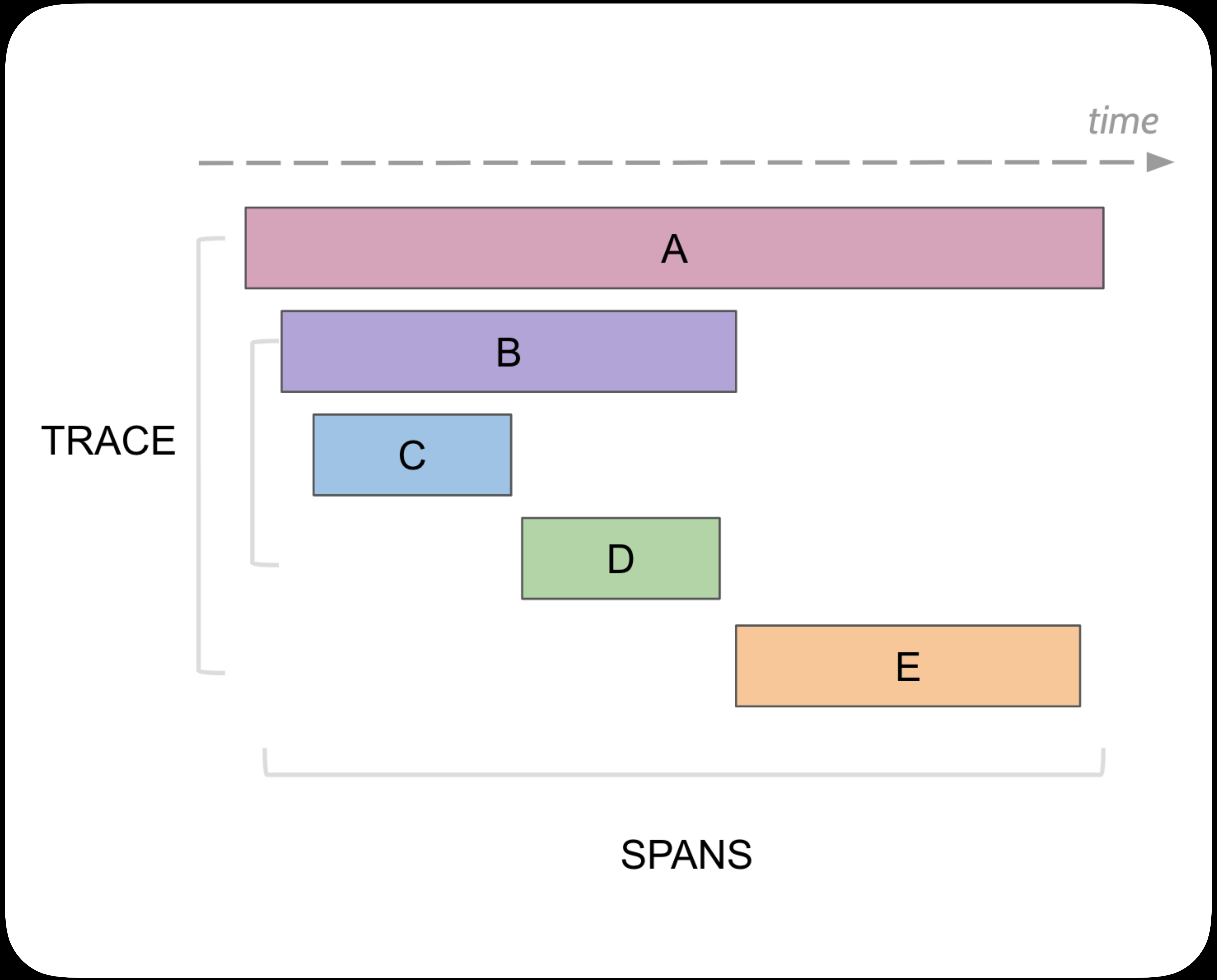
- How different was the execution from the normal system behavior
    - Structural differences
    - Performance differences
- What was on the critical path of the request

- Who should be paged

Distributed tracing
can answer these questions
and accelerate root cause analysis

Distributed Tracing in a Nutshell

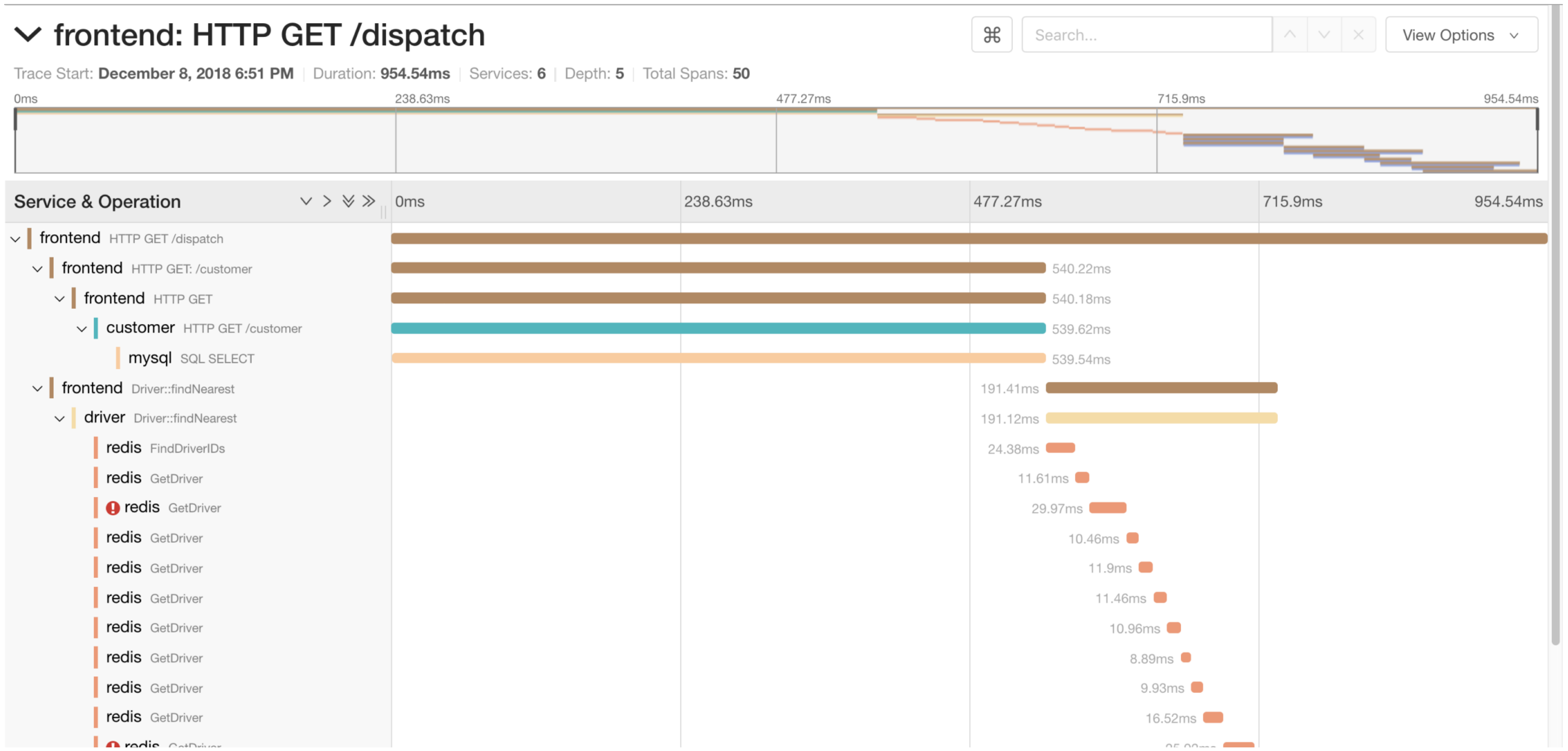




Trace as a narrative

Trace Timeline

Classic trace view as Gantt chart

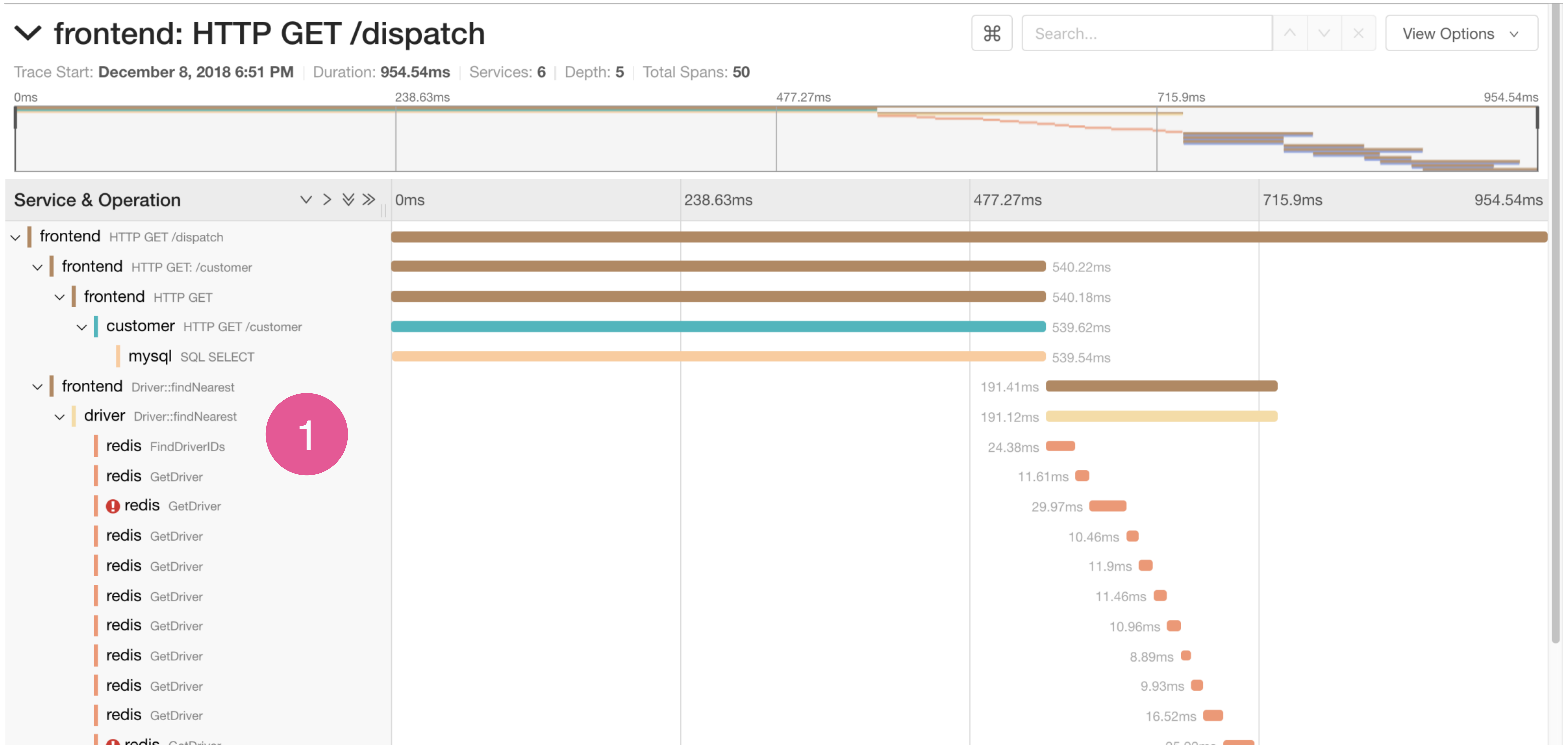

frontend: HTTP GET /dispatch

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

Parent → Child → Grandchild

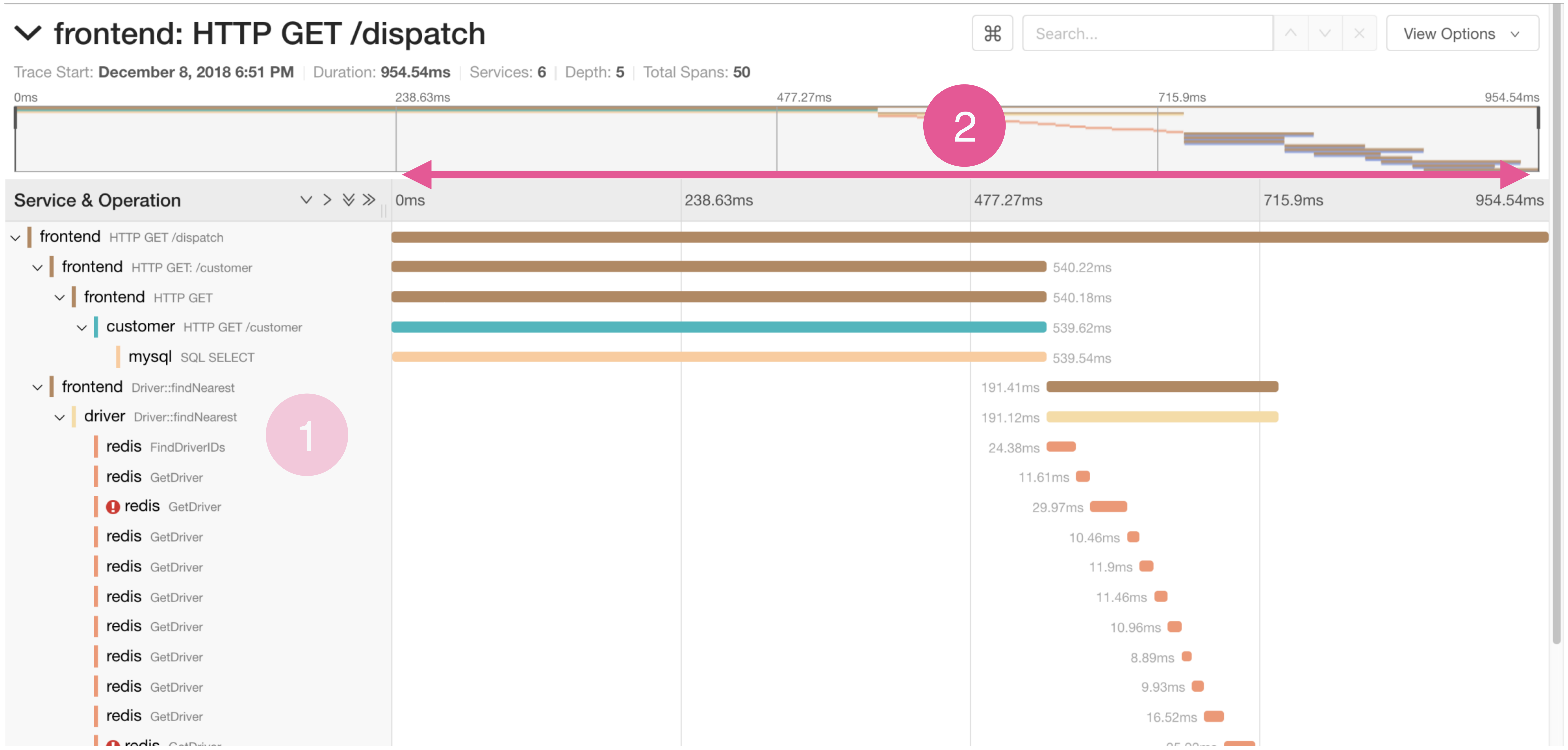

frontend: HTTP GET /dispatch

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

Time + Mini-Map



frontend: HTTP GET /dispatch

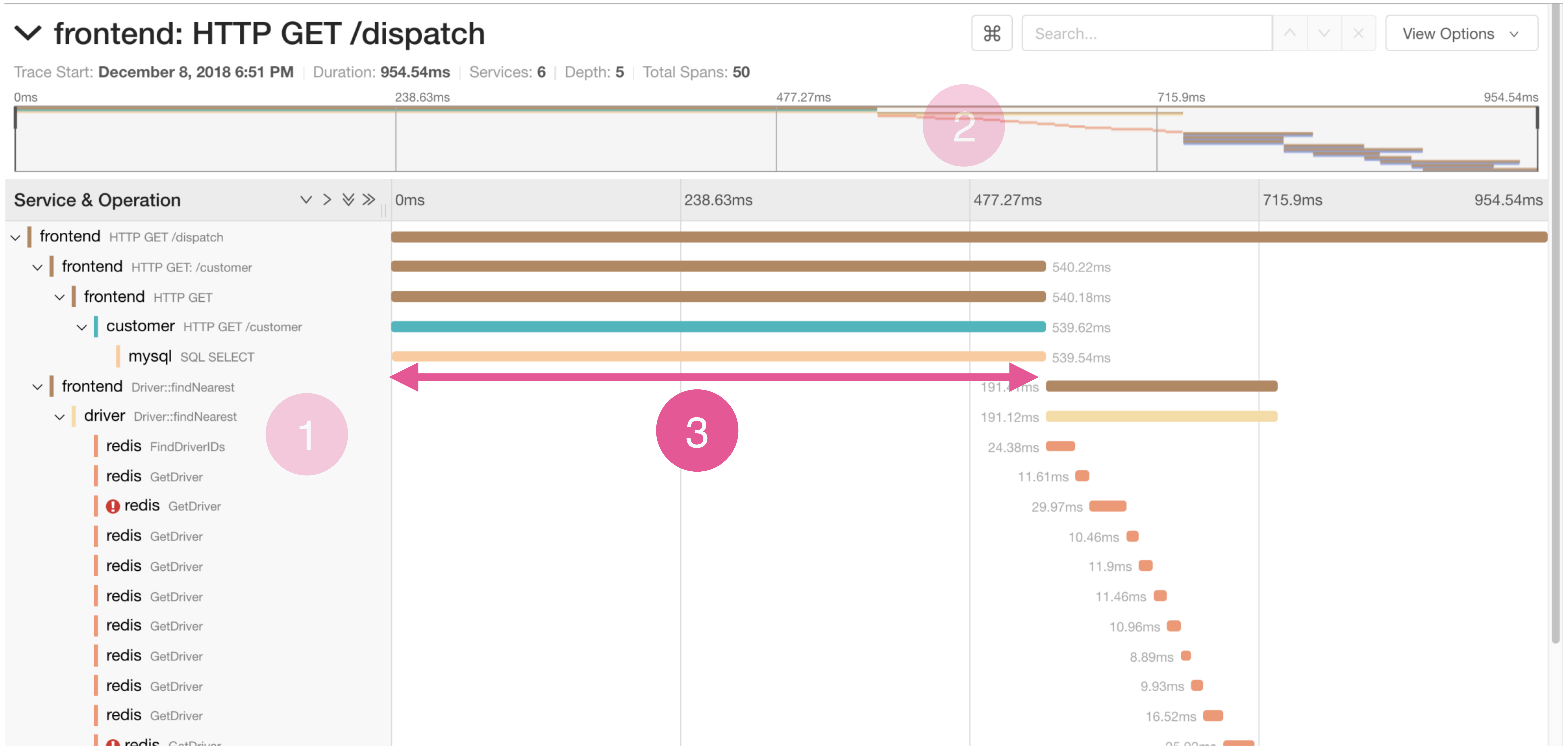
Trace Start: **December 8, 2018 6:51 PM** | Duration: **954.54ms** | Services: **6** | Depth: **5** | Total Spans: **50**

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

Trace Timeline

Blocking operation



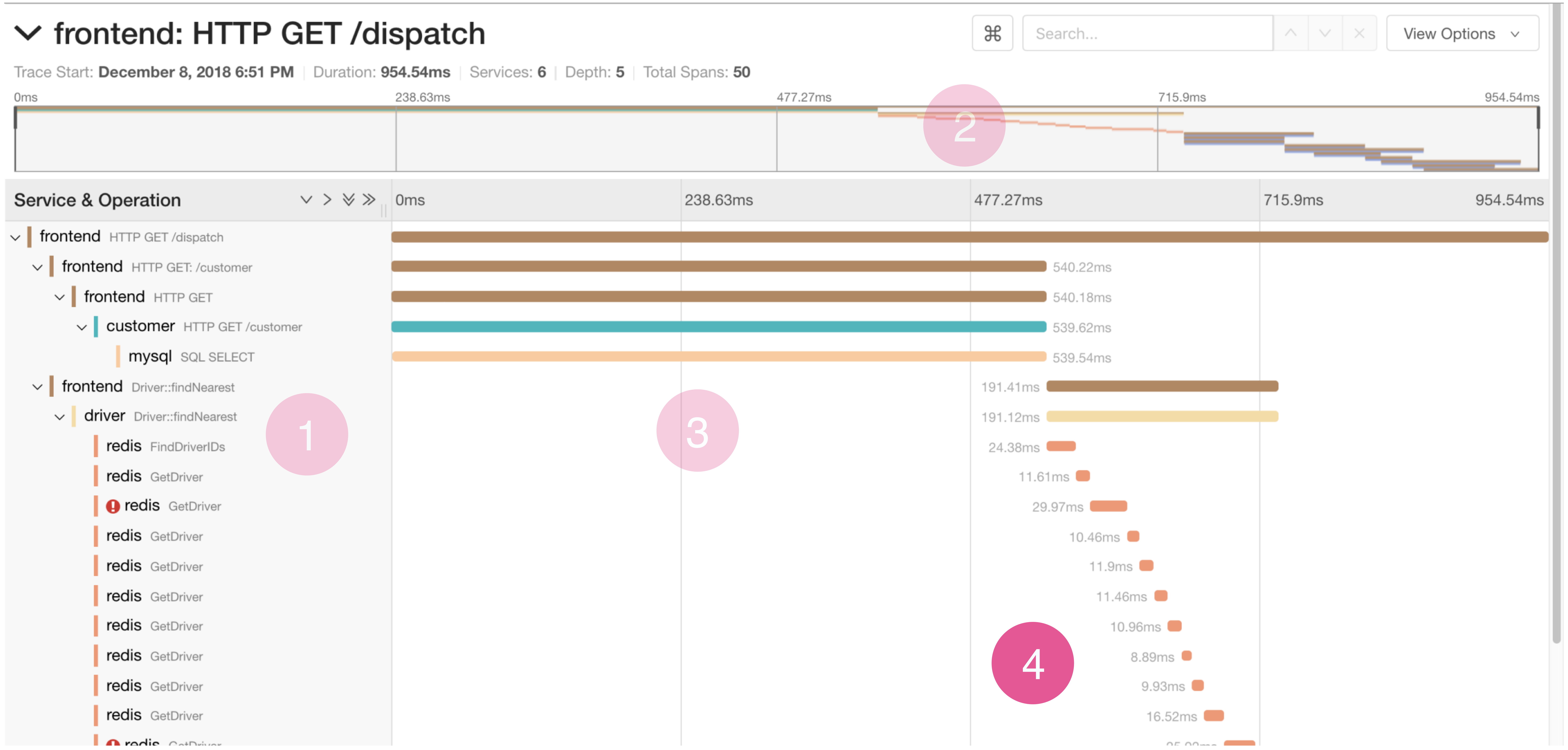
frontend: HTTP GET /dispatch

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

```
Service & Operation                          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
```

Trace Timeline

Sequential operations



frontend: HTTP GET /dispatch

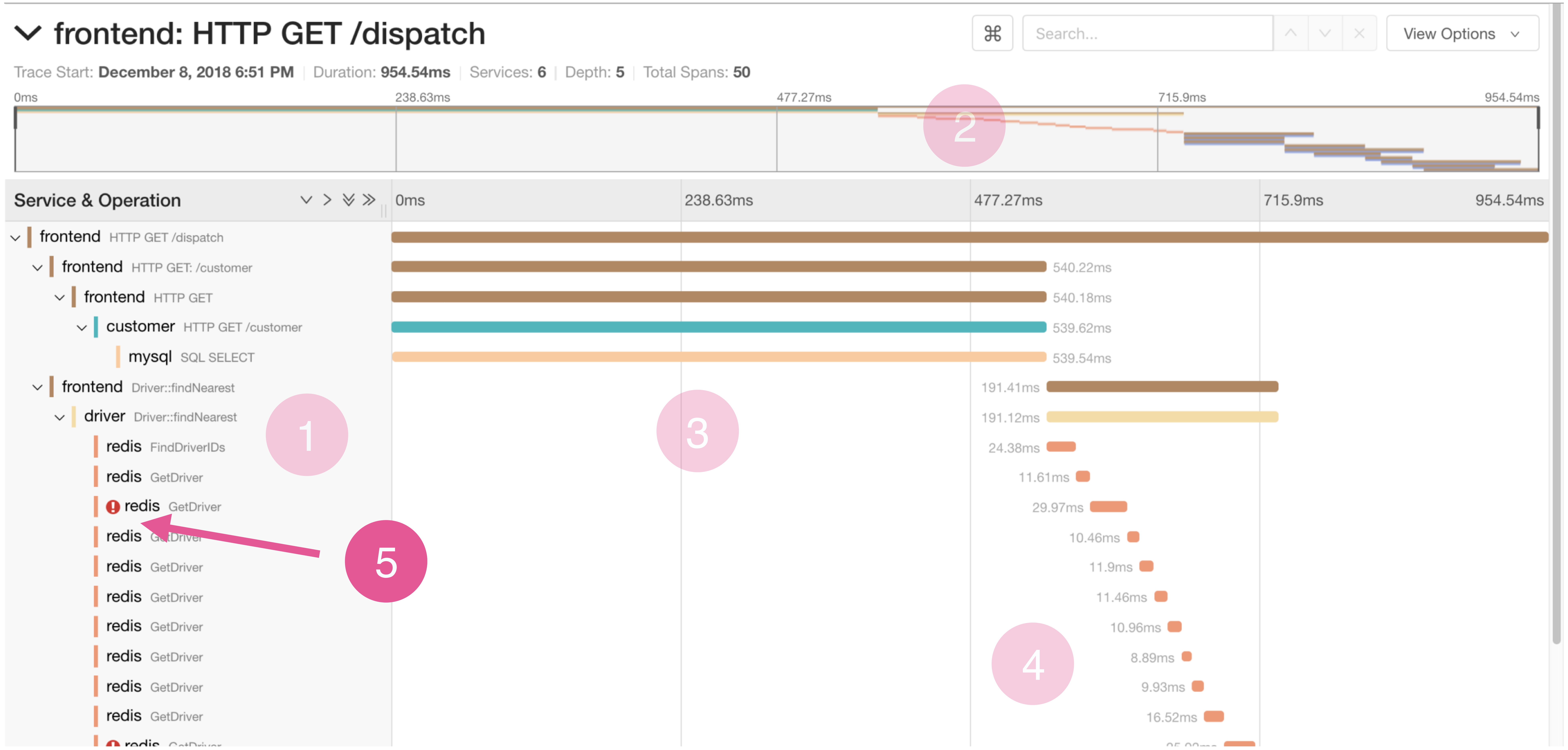
Trace Start: **December 8, 2018 6:51 PM** | Duration: **954.54ms** | Services: **6** | Depth: **5** | Total Spans: **50**

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

Trace Timeline

Errors

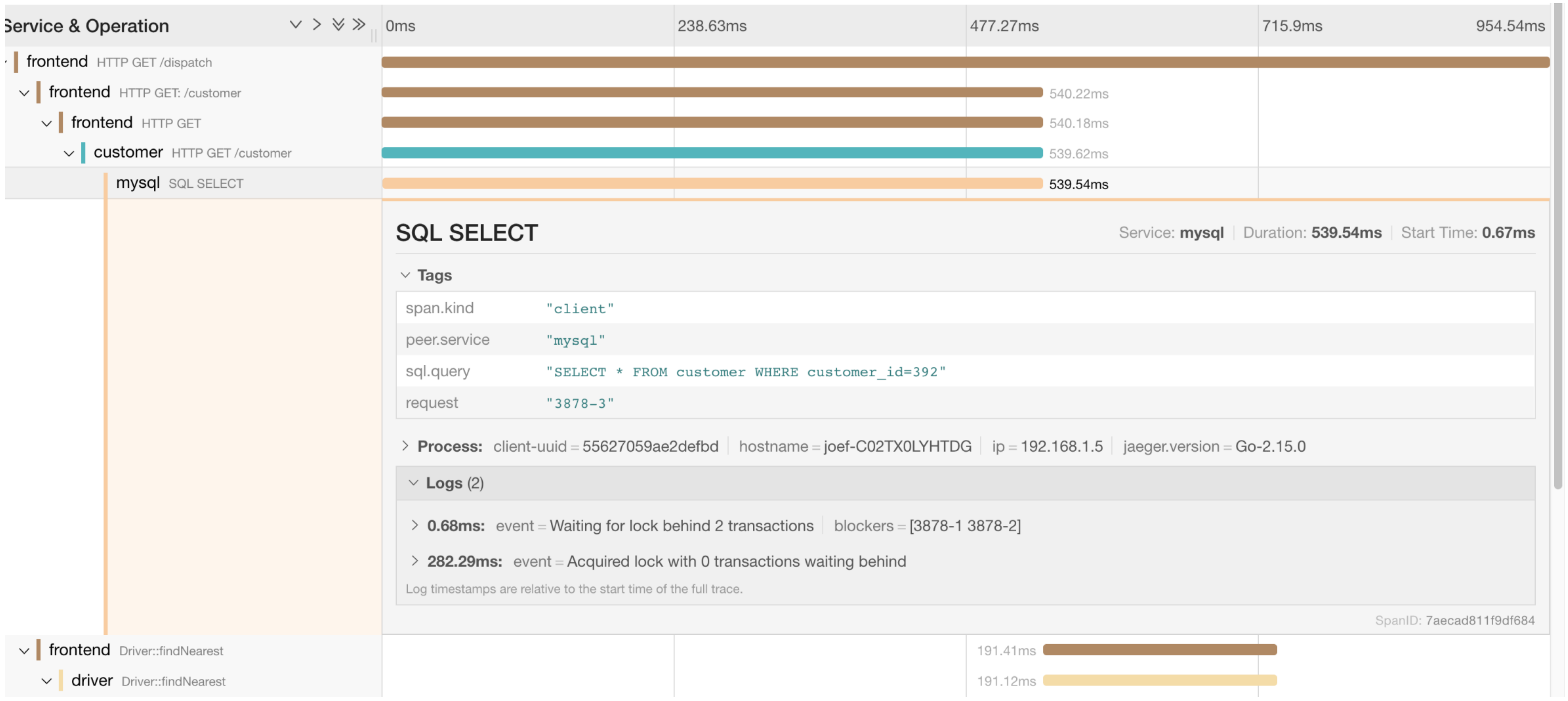


frontend: HTTP GET /dispatch

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

```
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
```

Span details



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

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

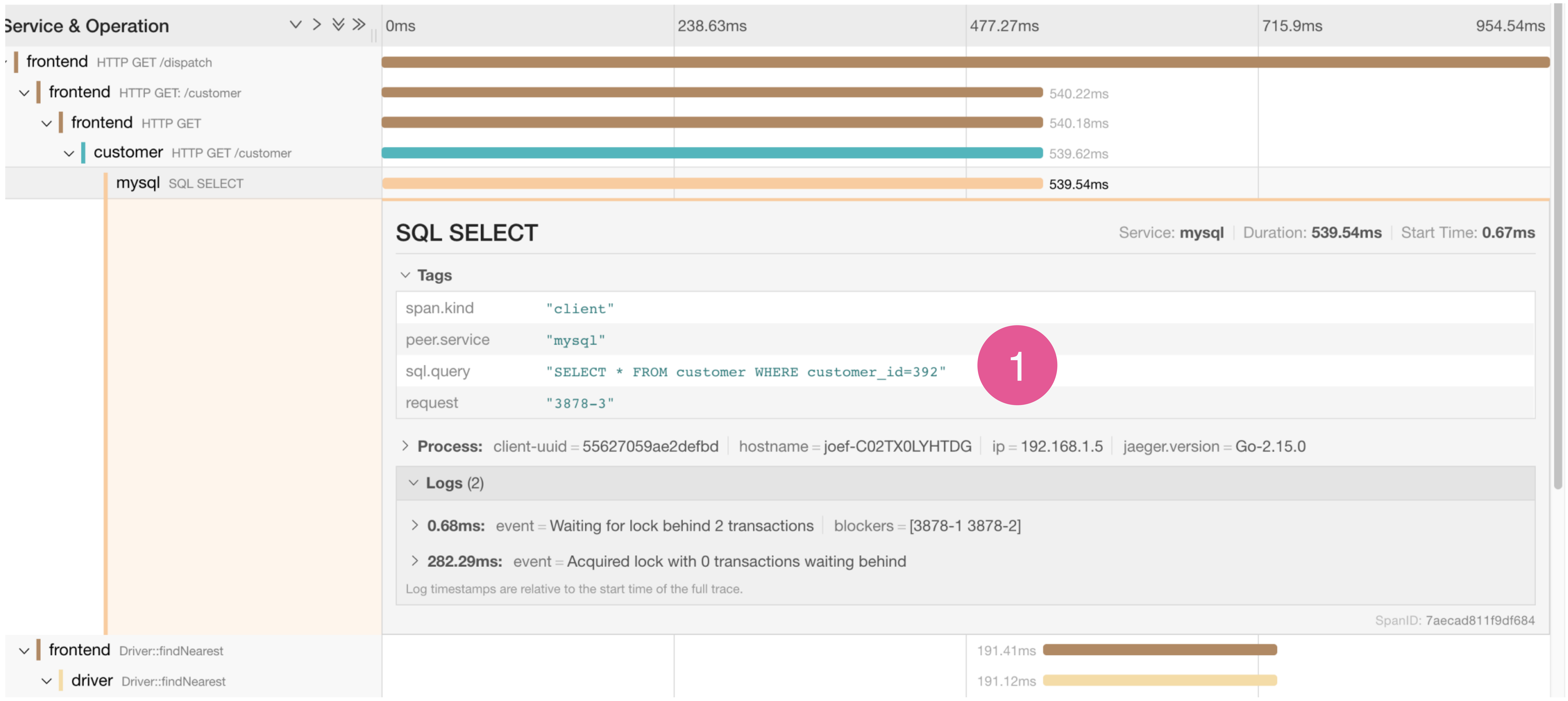
Log timestamps are relative to the start time of the full trace.

SpanID: 7aecad811f9df684

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

Span details

Database query



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

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]

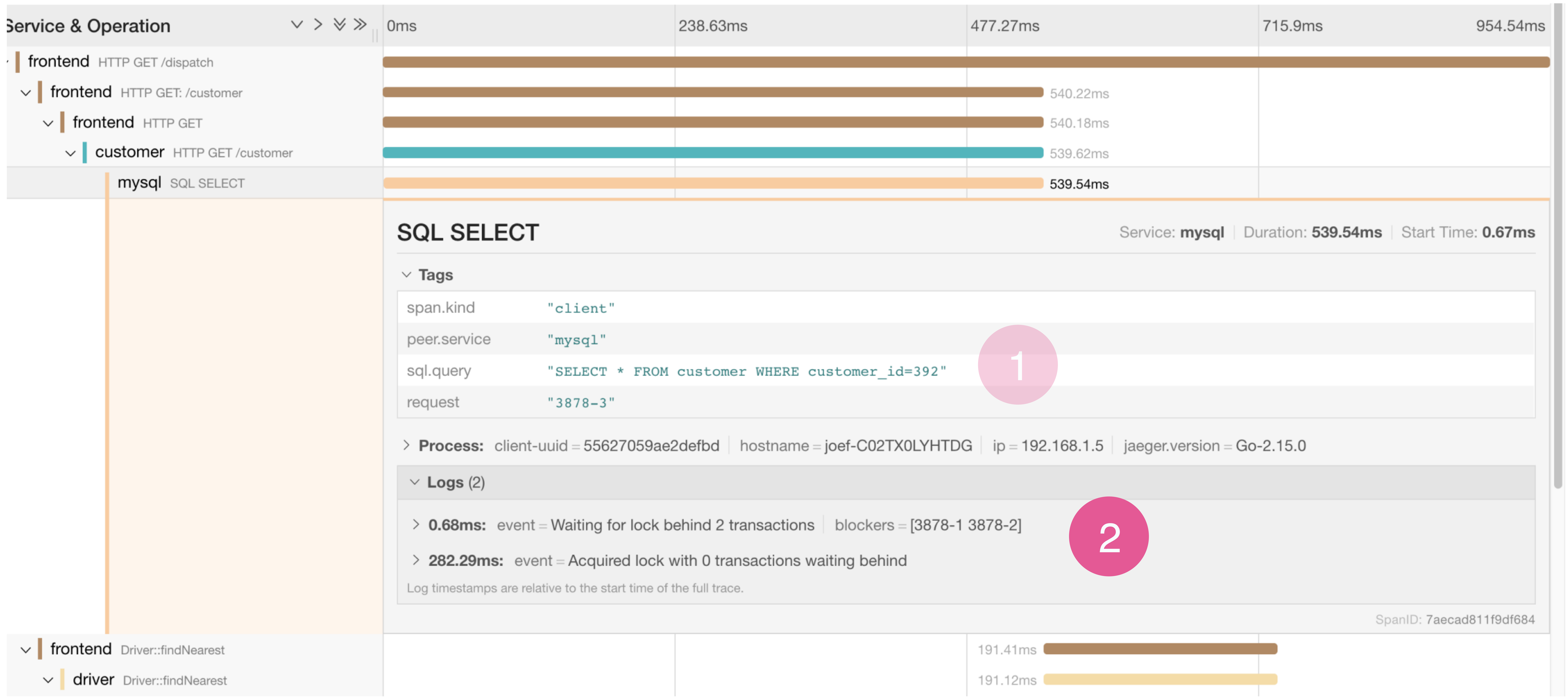
282.29ms: event = Acquired lock with 0 transactions waiting behind

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
Timed events (logs)



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

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"`

1

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

2

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

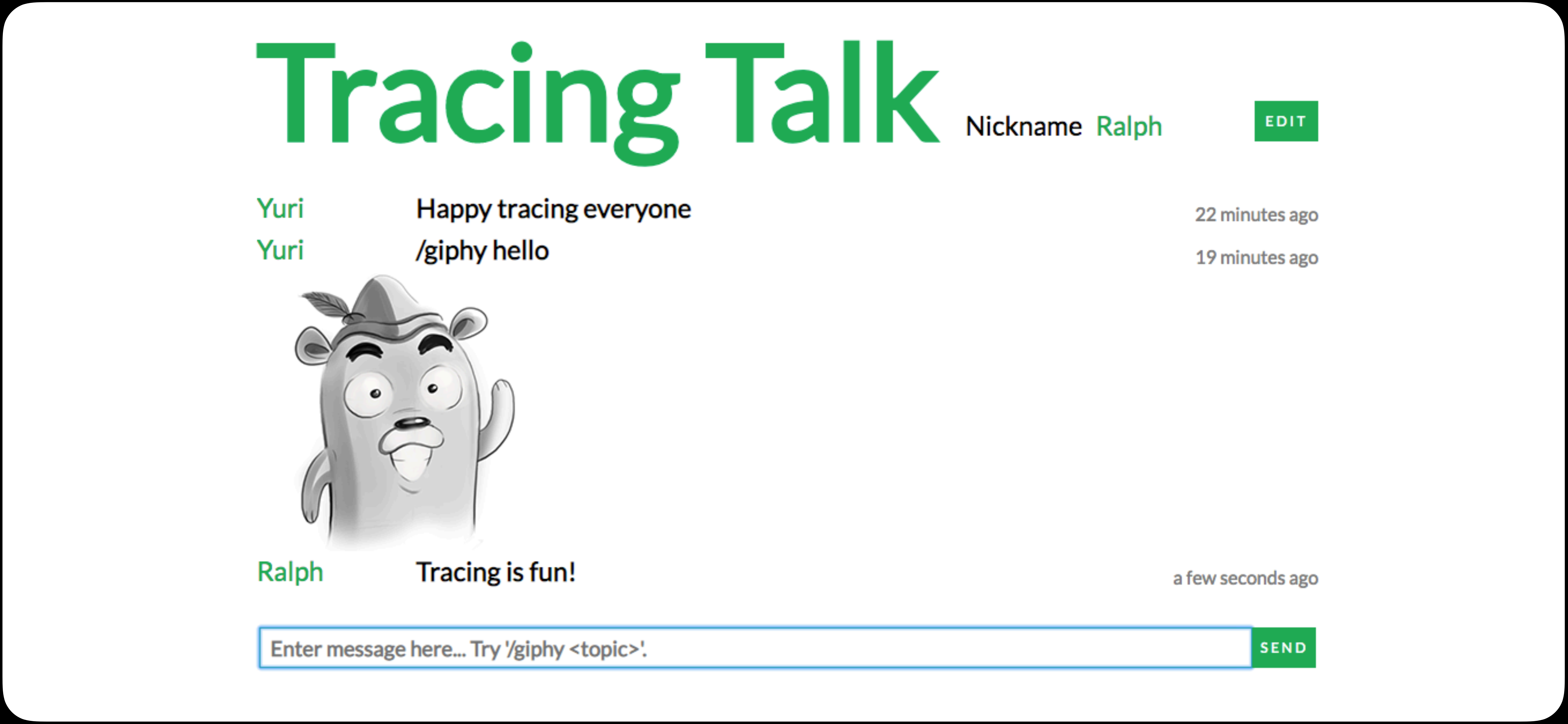
SpanID: 7aecad811f9df684

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

We can also trace
asynchronous workflows

Tracing Talk Application

Mastering Distributed Tracing, Chapter 5



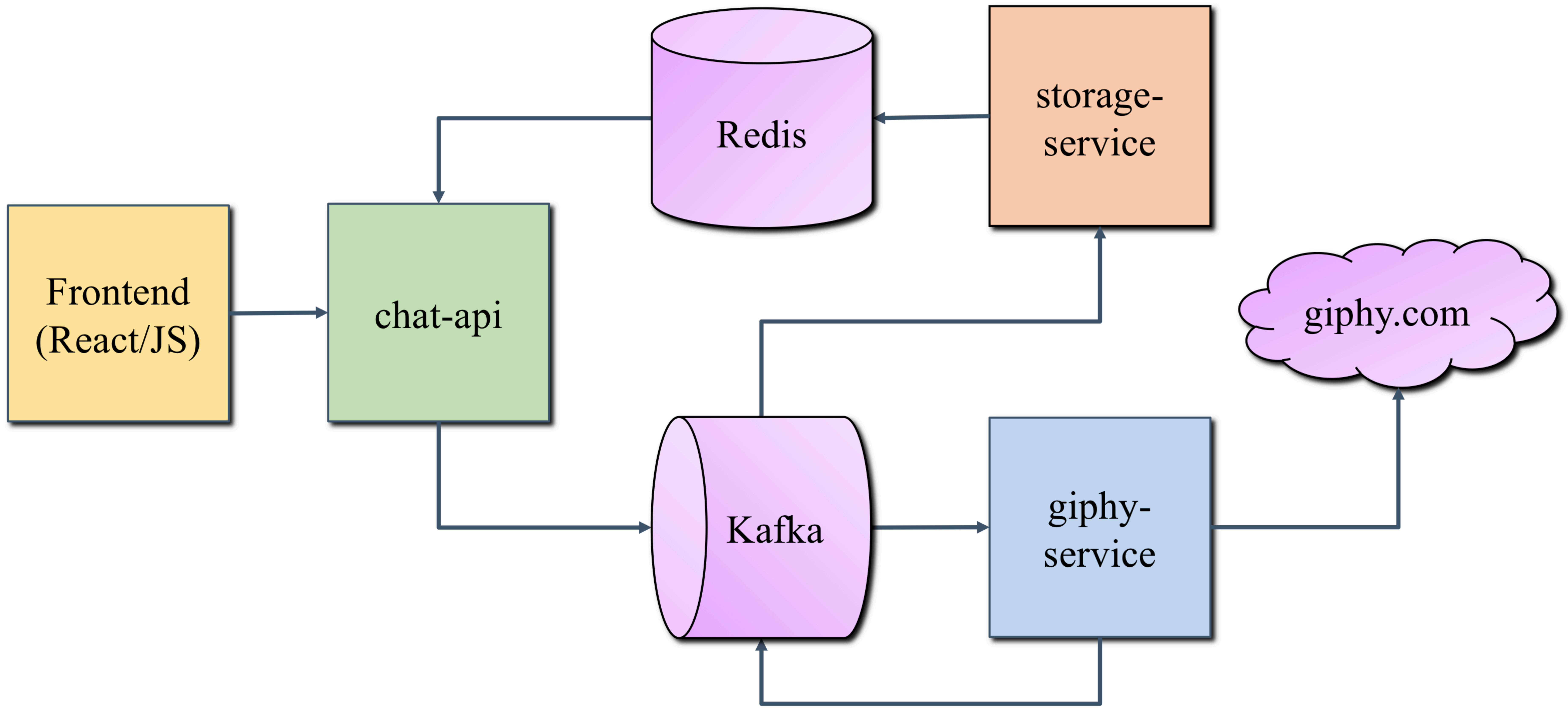
Yuri Happy tracing everyone 22 minutes ago

Yuri /giphy hello 19 minutes ago

Ralph Tracing is fun! a few seconds ago

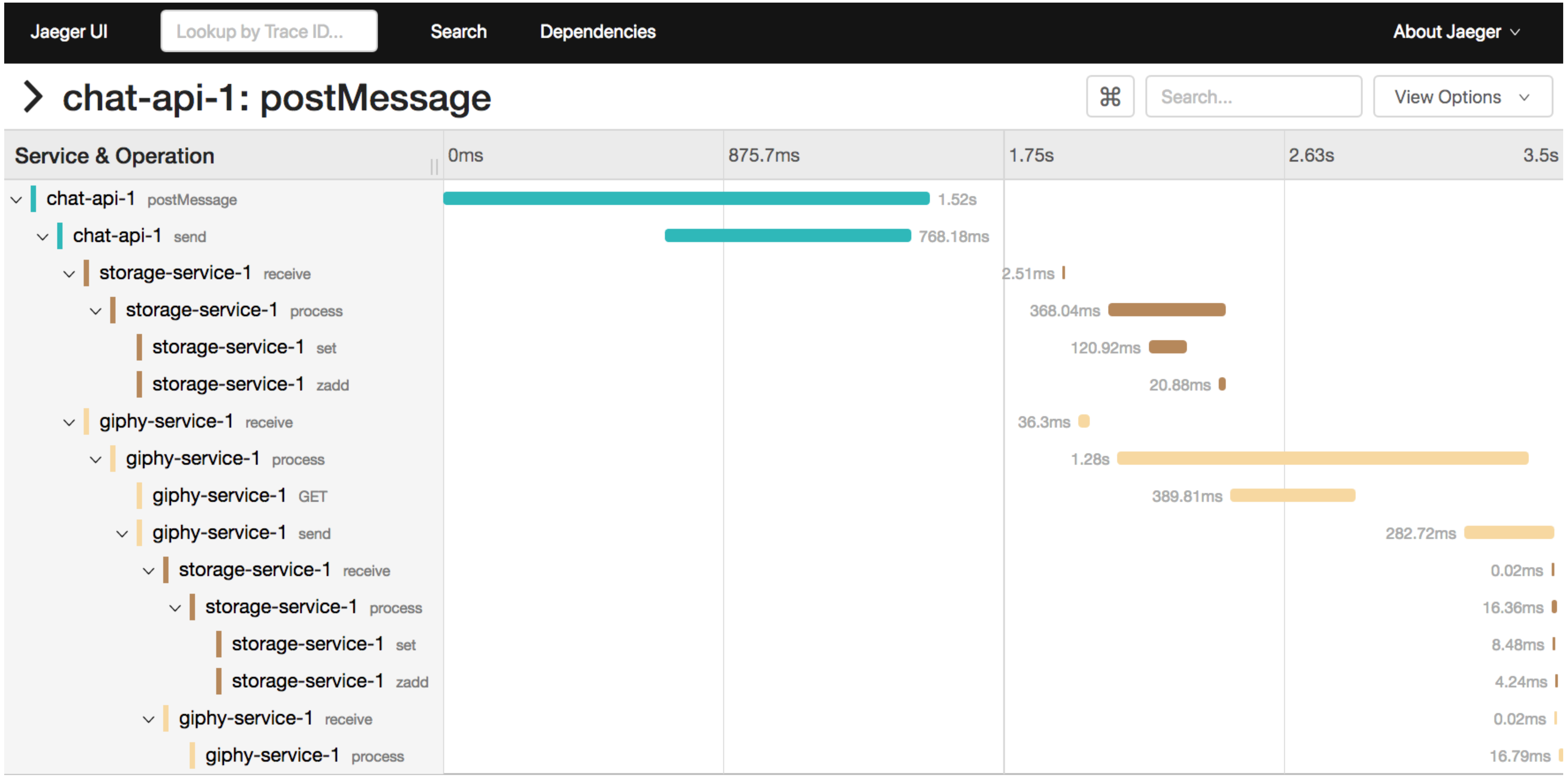
Tracing Talk Application

Architecture



Tracing Talk Application

Request trace



Jaeger UI | Lookup by Trace ID... | Search | Dependencies | About Jaeger

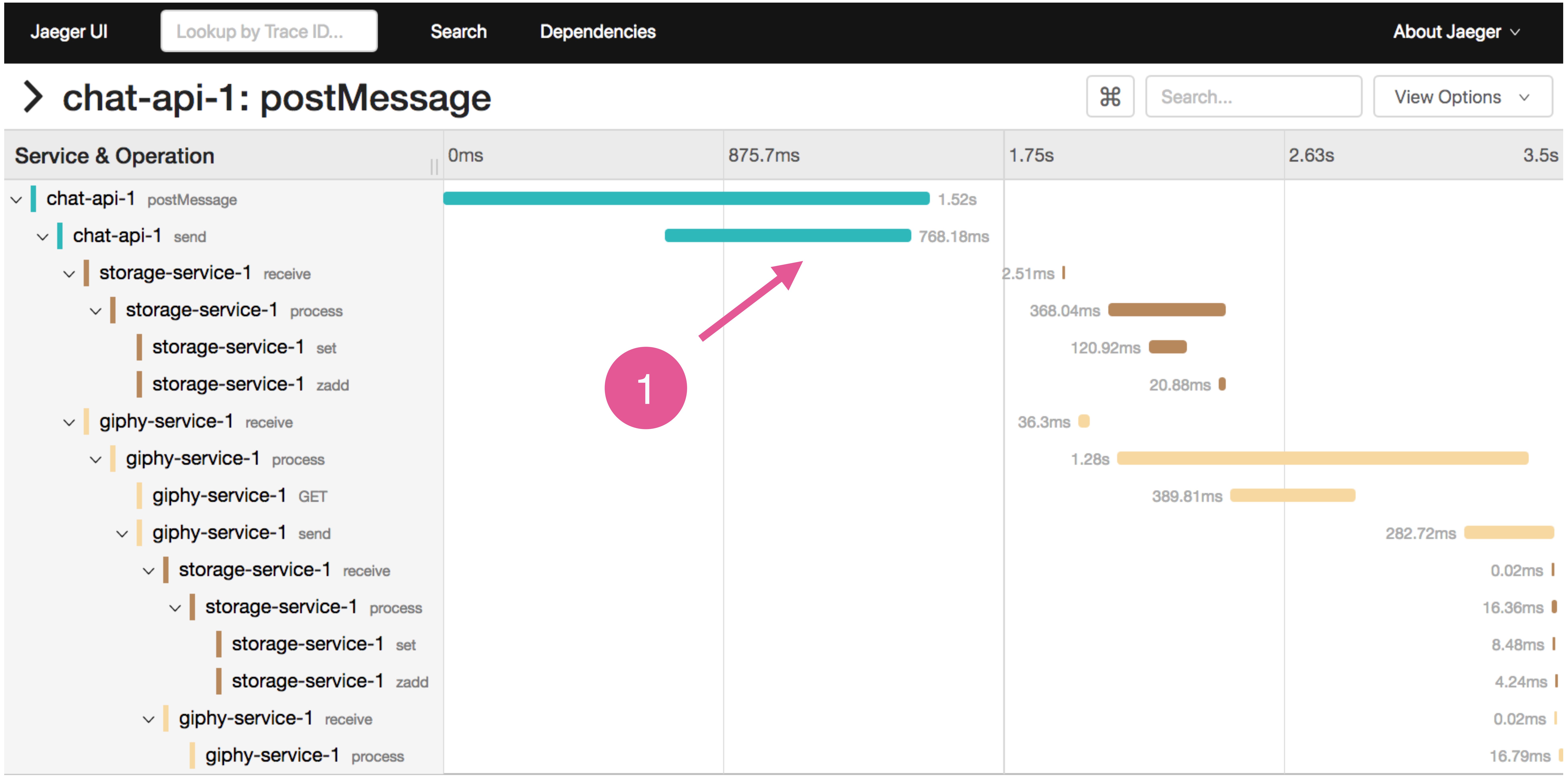
chat-api-1: postMessage

View Options

Service & Operation	Duration
chat-api-1 postMessage	1.52s
chat-api-1 send	768.18ms
storage-service-1 receive	2.51ms
storage-service-1 process	368.04ms
storage-service-1 set	120.92ms
storage-service-1 zadd	20.88ms
giphy-service-1 receive	36.3ms
giphy-service-1 process	1.28s
giphy-service-1 GET	389.81ms
giphy-service-1 send	282.72ms
storage-service-1 receive	0.02ms
storage-service-1 process	16.36ms
storage-service-1 set	8.48ms
storage-service-1 zadd	4.24ms
giphy-service-1 receive	0.02ms
giphy-service-1 process	16.79ms

Tracing Talk Application

Message sent

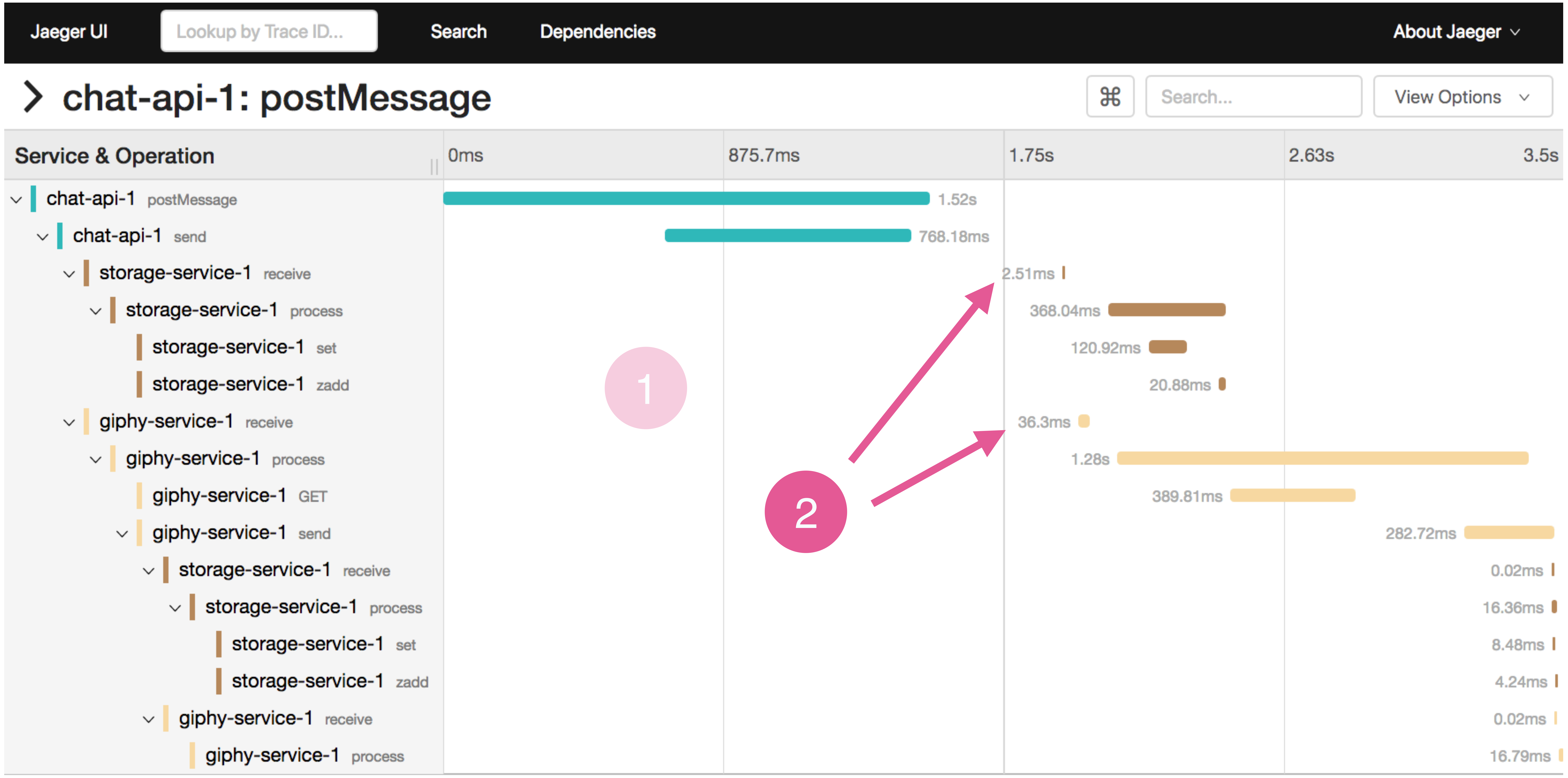


chat-api-1: postMessage

Service & Operation	0ms · 875.7ms · 1.75s · 2.63s · 3.5s
chat-api-1 postMessage	1.52s
chat-api-1 send	768.18ms
storage-service-1 receive	2.51ms
storage-service-1 process	368.04ms
storage-service-1 set	120.92ms
storage-service-1 zadd	20.88ms
giphy-service-1 receive	36.3ms
giphy-service-1 process	1.28s
giphy-service-1 GET	389.81ms
giphy-service-1 send	282.72ms
storage-service-1 receive	0.02ms
storage-service-1 process	16.36ms
storage-service-1 set	8.48ms
storage-service-1 zadd	4.24ms
giphy-service-1 receive	0.02ms
giphy-service-1 process	16.79ms

Tracing Talk Application

Message received



chat-api-1: postMessage

Service & Operation	0ms · 875.7ms · 1.75s · 2.63s · 3.5s
chat-api-1 postMessage	1.52s
chat-api-1 send	768.18ms
storage-service-1 receive	2.51ms
storage-service-1 process	368.04ms
storage-service-1 set	120.92ms
storage-service-1 zadd	20.88ms
giphy-service-1 receive	36.3ms
giphy-service-1 process	1.28s
giphy-service-1 GET	389.81ms
giphy-service-1 send	282.72ms
storage-service-1 receive	0.02ms
storage-service-1 process	16.36ms
storage-service-1 set	8.48ms
storage-service-1 zadd	4.24ms
giphy-service-1 receive	0.02ms
giphy-service-1 process	16.79ms

Single Trace

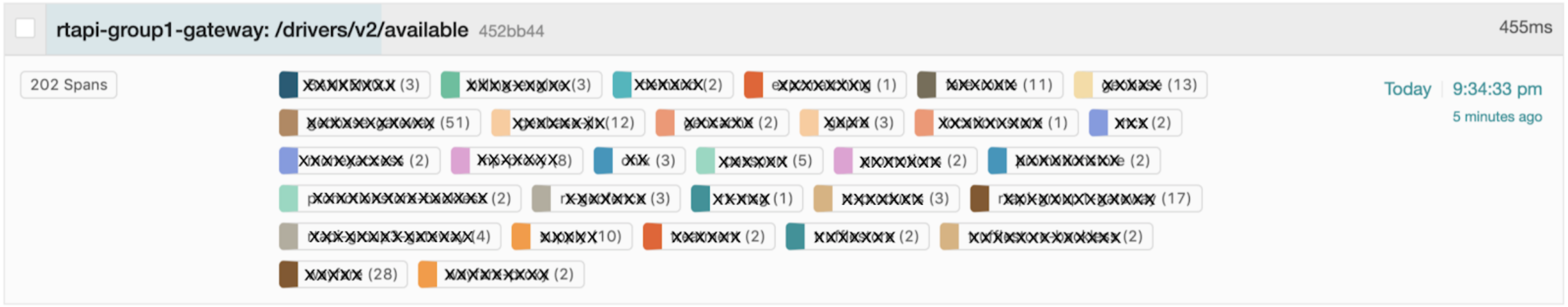
Pros and cons

- Tells a story about a single transaction
- Allows deep contextual drill-down
- Acts as a distributed stack trace

- Tells a story about a single transaction. What if it's an anomaly?
- One trace can be overwhelmingly complex

Too Much Complexity

One request - 30 services, 100+ RPCs

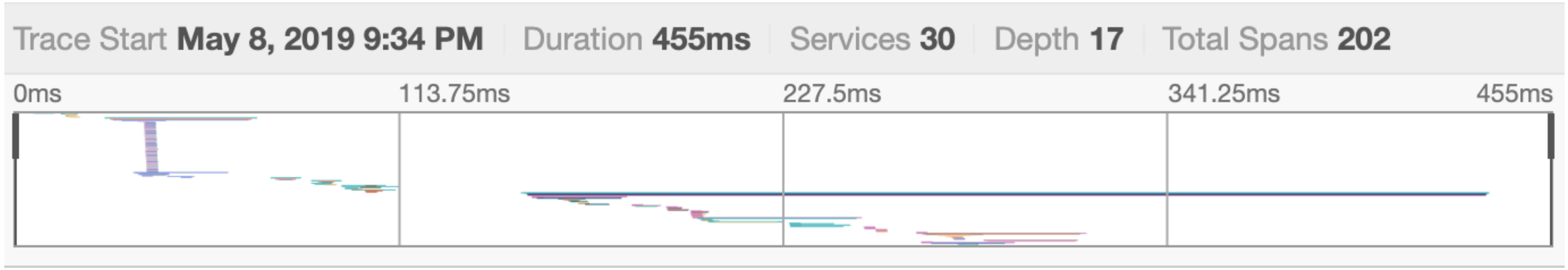


rtapi-group1-gateway: /drivers/v2/available 452bb44 455ms

202 Spans

[redacted] (3) [redacted] (3) [redacted] (2) [redacted] (1) [redacted] (11) [redacted] (13)
[redacted] (51) [redacted] (12) [redacted] (2) [redacted] (3) [redacted] (1) [redacted] (2)
[redacted] (2) [redacted] (8) [redacted] (3) [redacted] (5) [redacted] (2) [redacted] (2)
[redacted] (2) [redacted] (3) [redacted] (1) [redacted] (3) [redacted] (17)
[redacted] (4) [redacted] (10) [redacted] (2) [redacted] (2) [redacted] (2)
[redacted] (28) [redacted] (2)

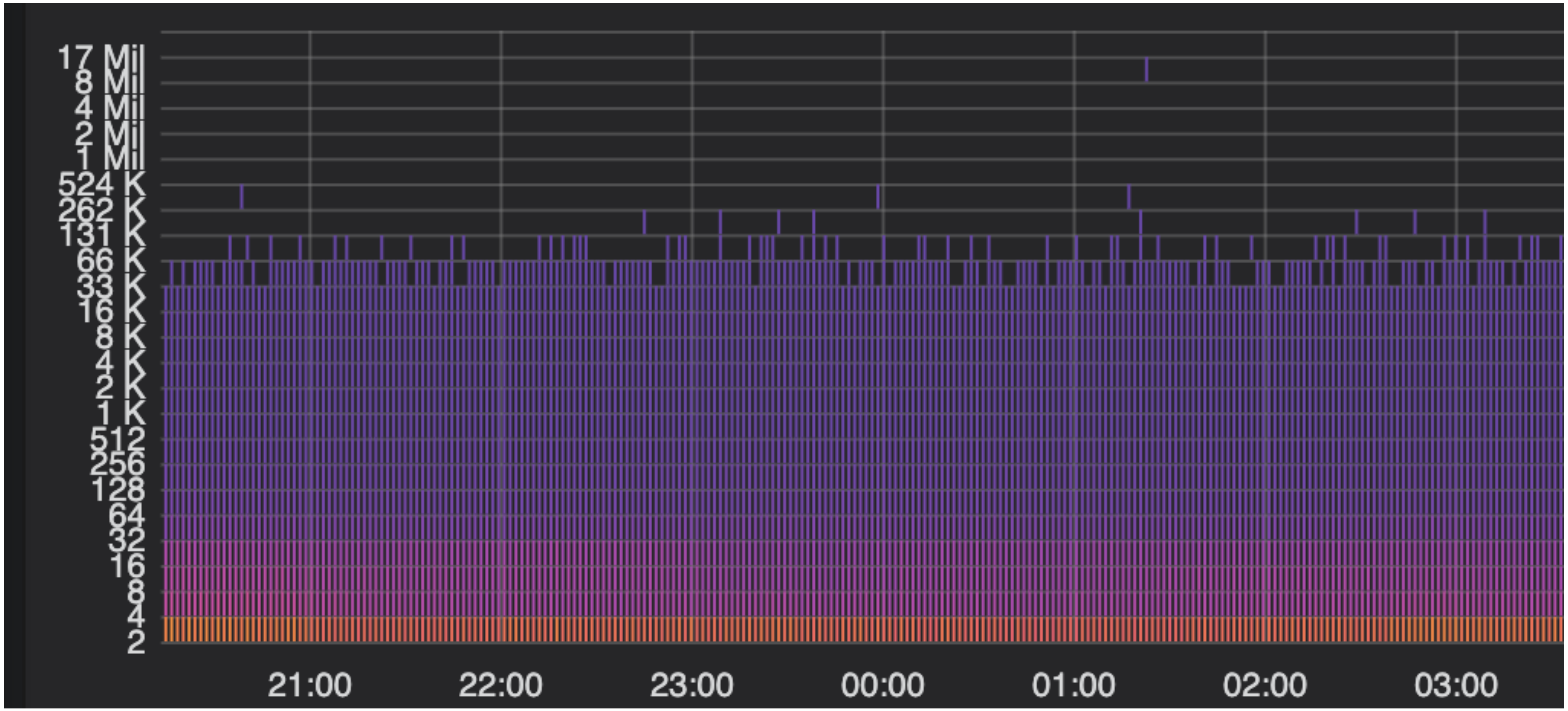
Today | 9:34:33 pm
5 minutes ago



Trace Start **May 8, 2019 9:34 PM** | Duration **455ms** | Services **30** | Depth **17** | Total Spans **202**

Too Much Complexity

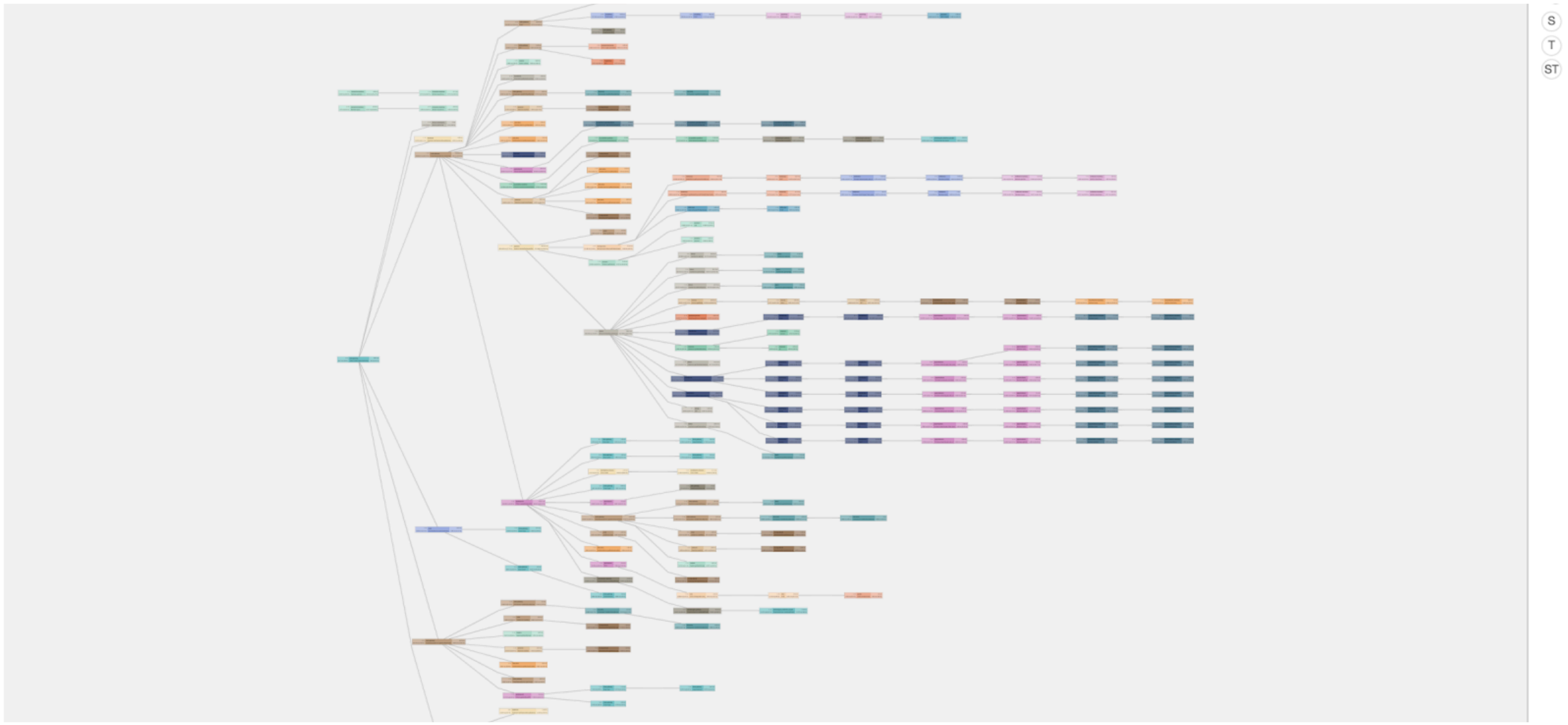
Some traces have hundreds of thousands spans



Reducing complexity by smarter visualizations

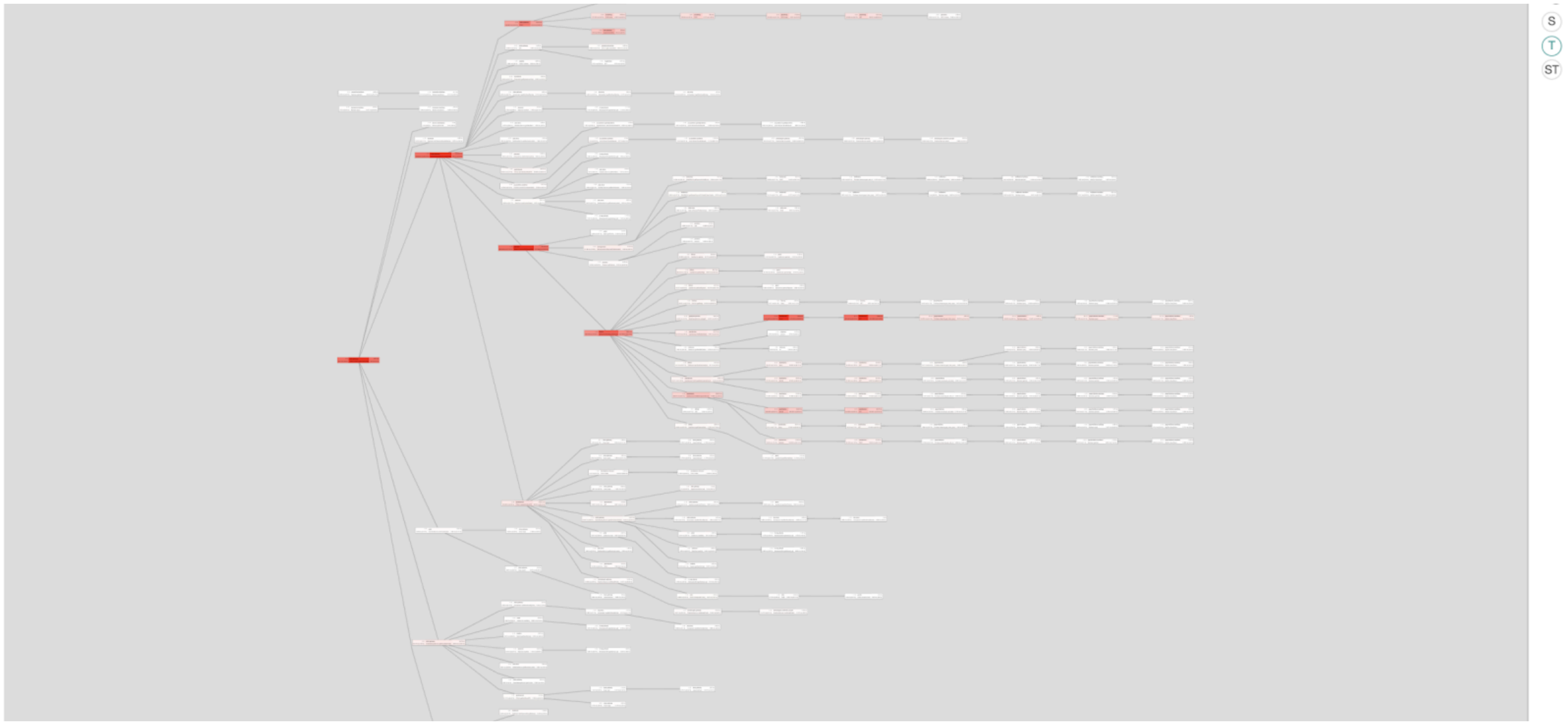
Trace graph

Time ordered, repeated edges collapsed



Trace graph

Latency heat map

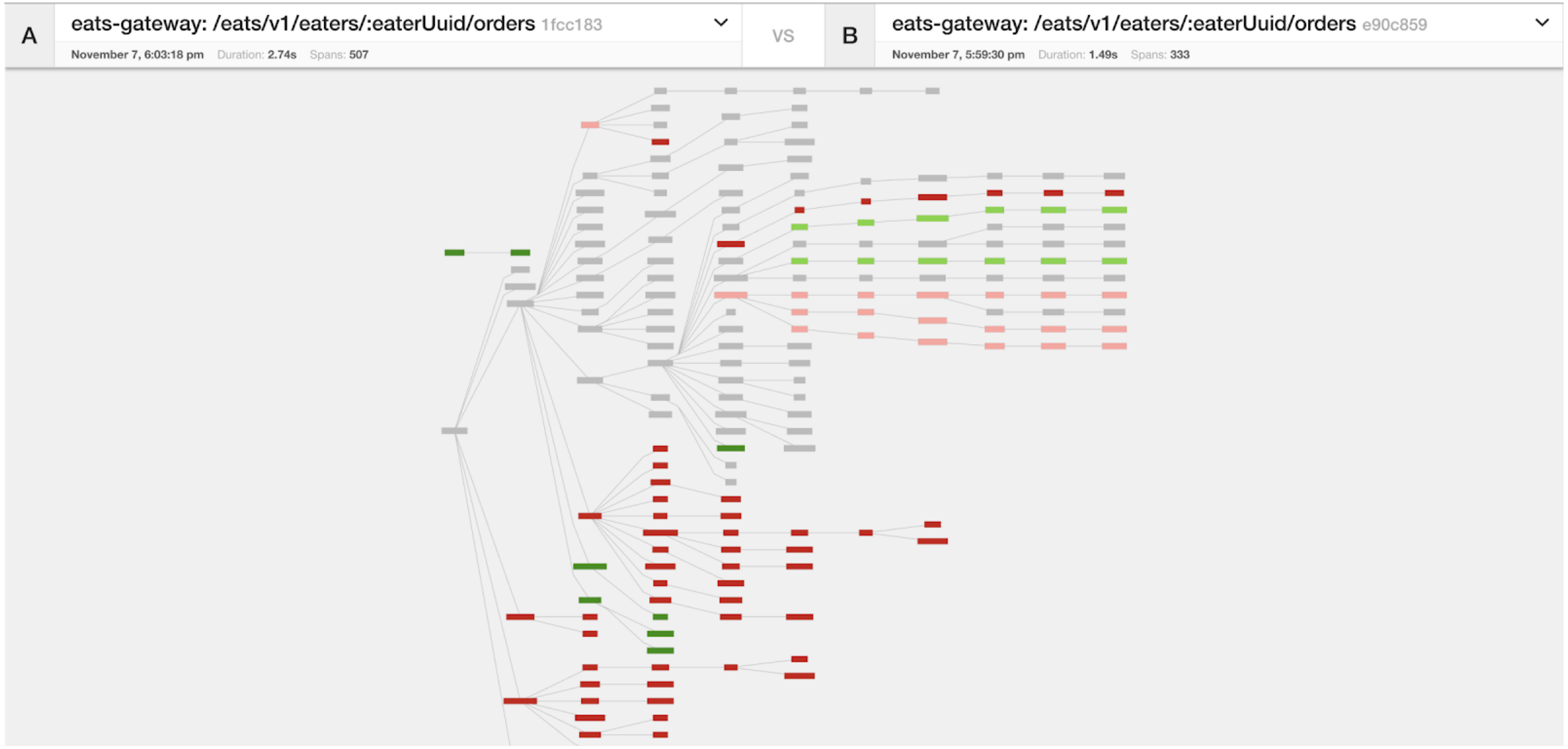


Finding anomalies is easier
when we look at differences
in performance profiles

Trace vs. Trace

Comparing Trace Structures

Just like a Code Diff



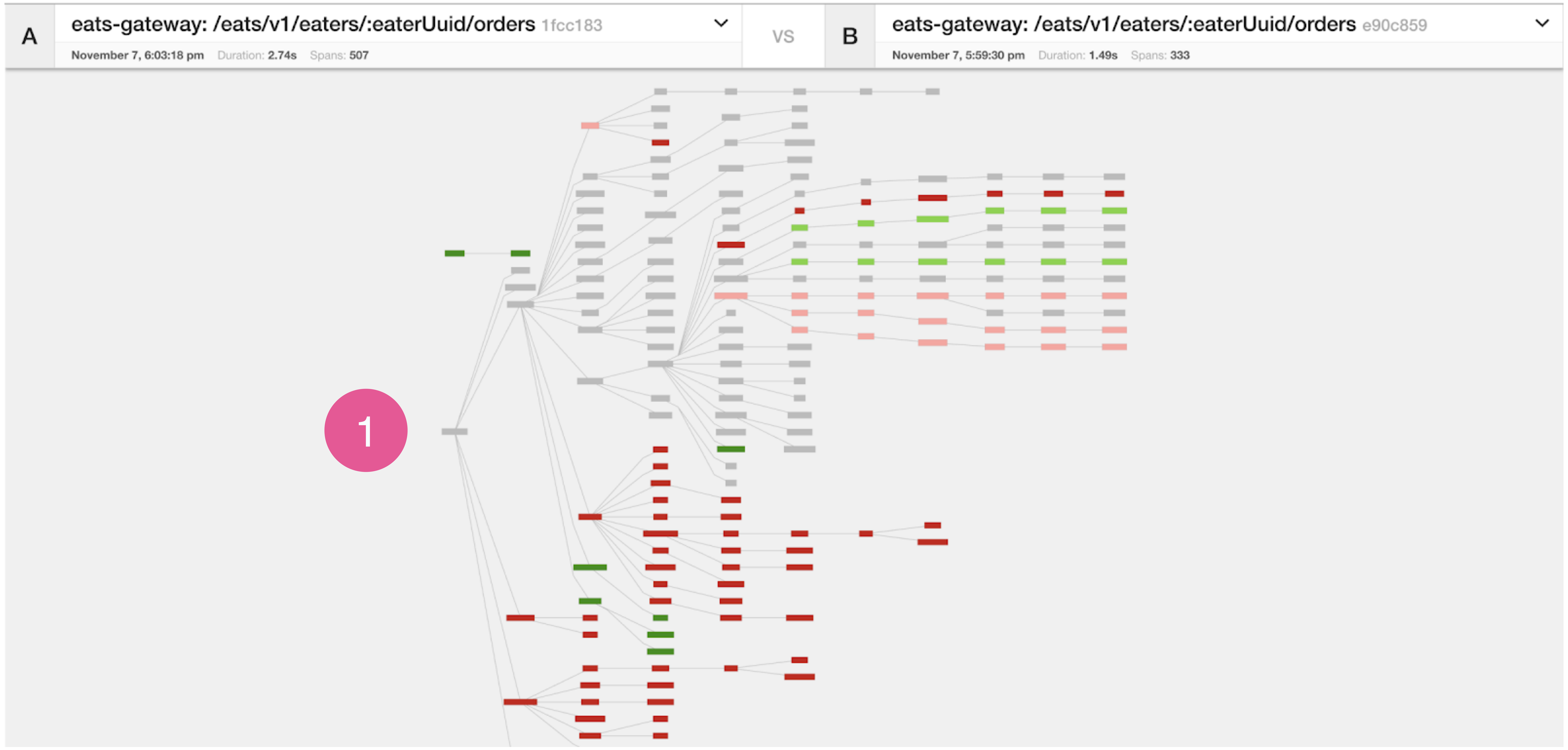
A eats-gateway: /eats/v1/eaters/:eaterUuid/orders 1fcc183
November 7, 6:03:18 pm Duration: 2.74s Spans: 507

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



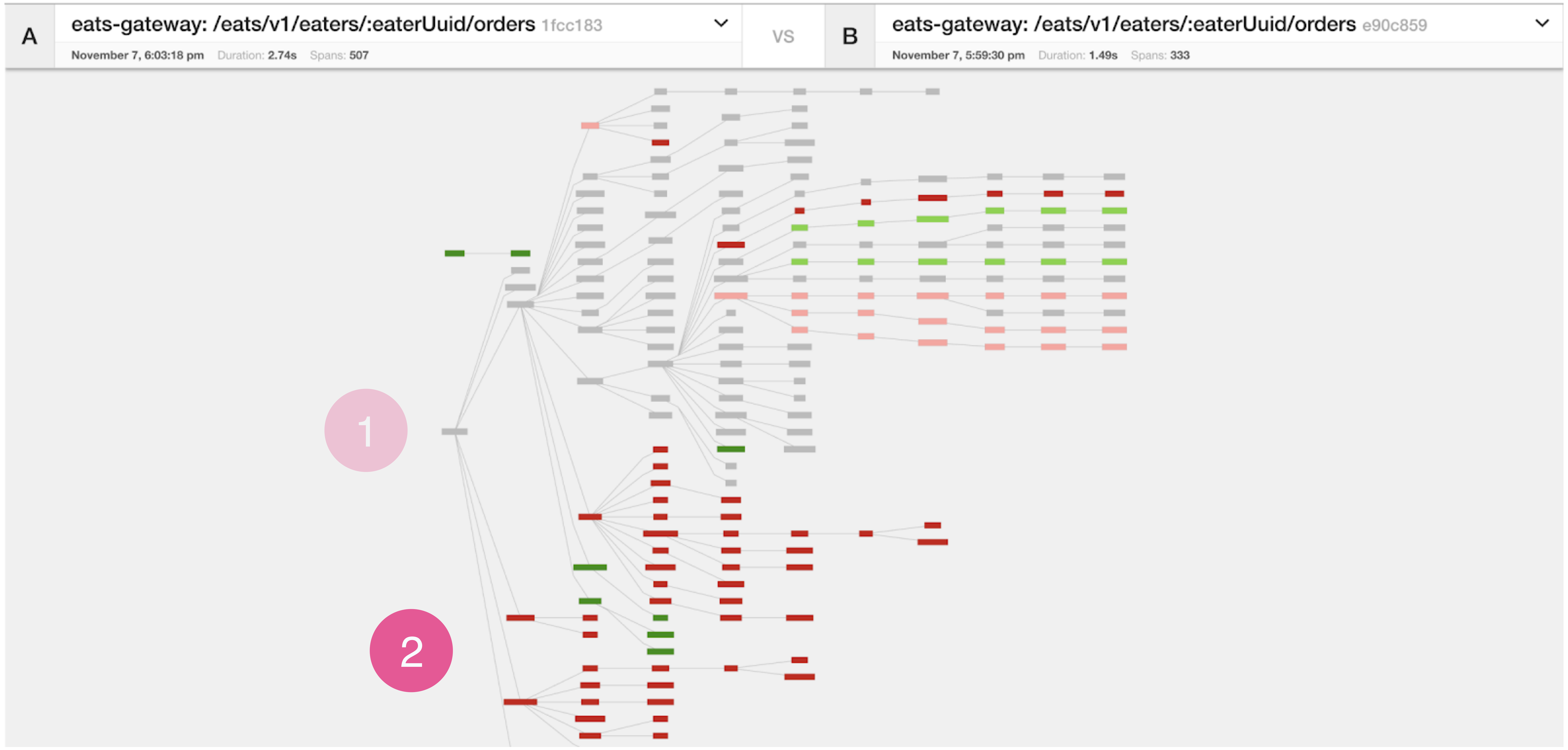
A eats-gateway: /eats/v1/eaters/:eaterUuid/orders 1fcc183
November 7, 6:03:18 pm Duration: 2.74s Spans: 507

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

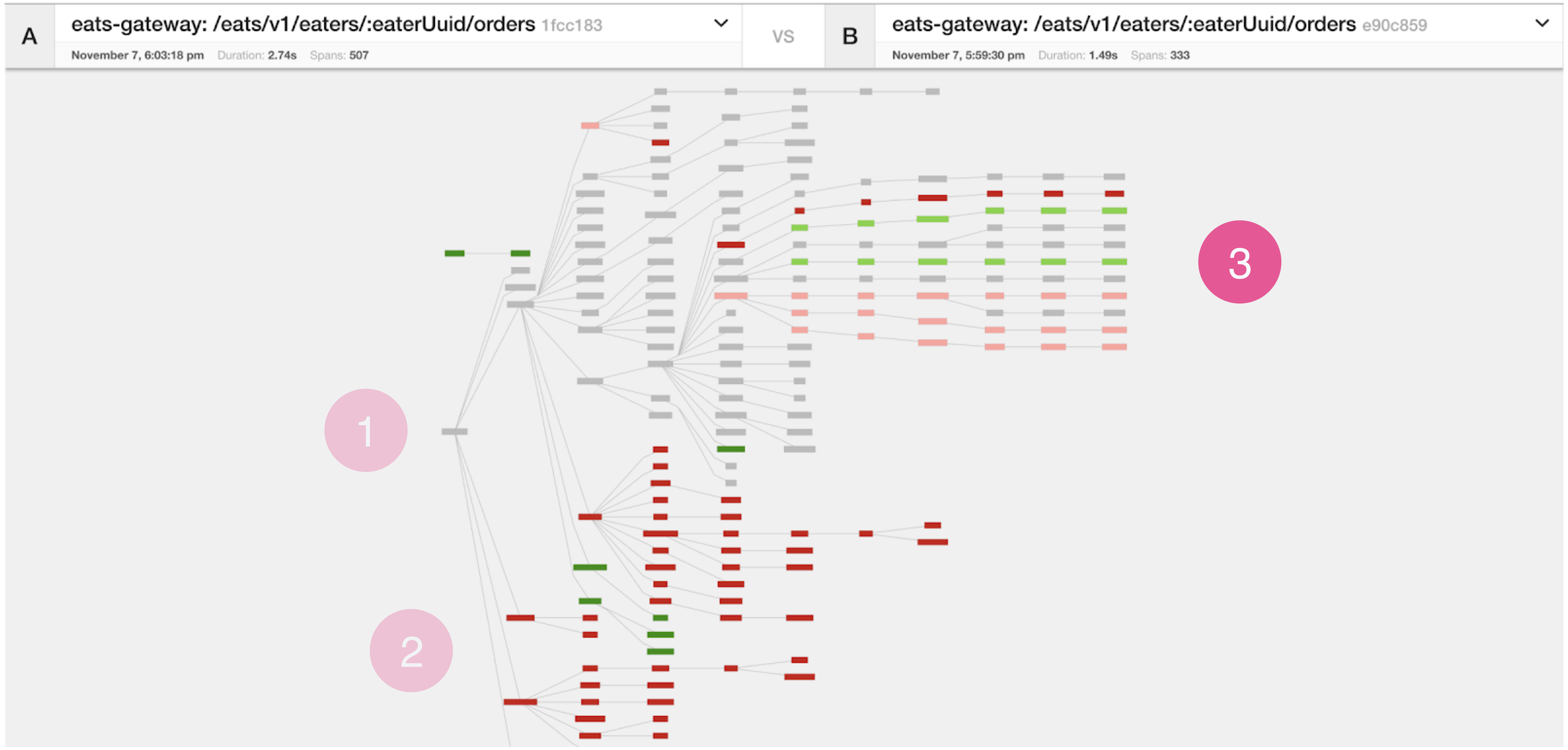

A eats-gateway: /eats/v1/eaters/:eaterUuid/orders 1fcc183
November 7, 6:03:18 pm Duration: 2.74s Spans: 507

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 Fewer Spans Within a Node



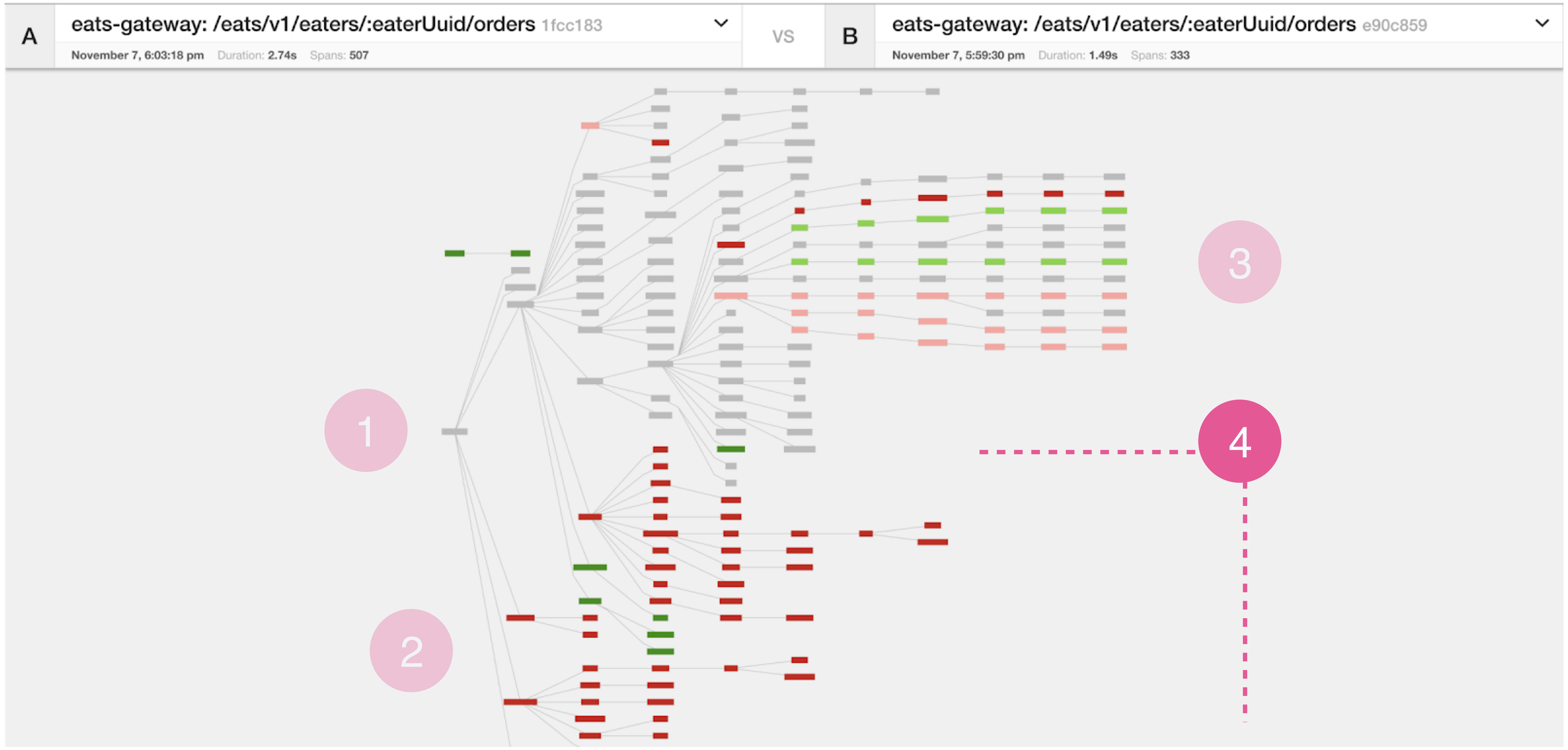
A eats-gateway: /eats/v1/eaters/:eaterUuid/orders 1fcc183
November 7, 6:03:18 pm Duration: 2.74s Spans: 507

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



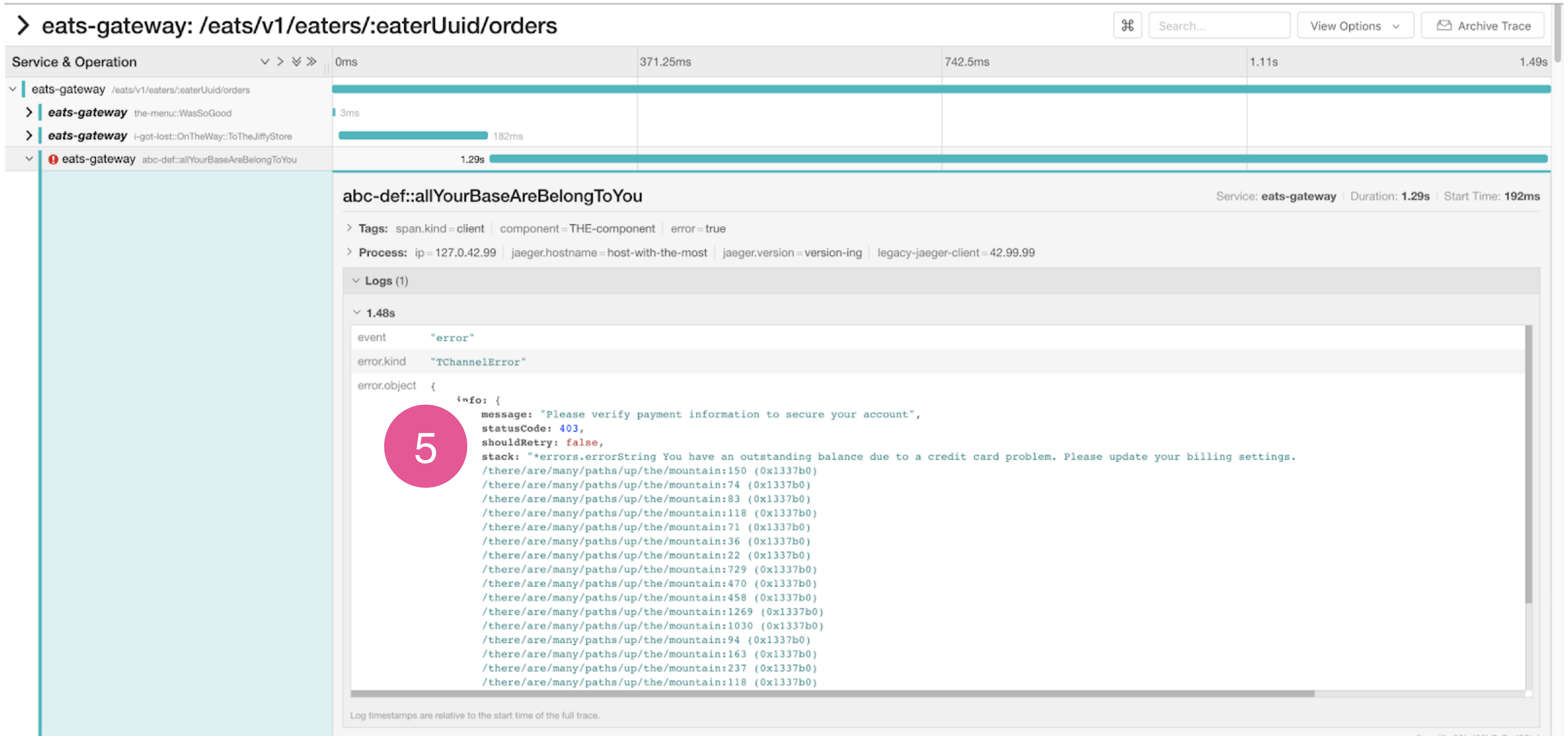
A eats-gateway: /eats/v1/eaters/:eaterUuid/orders 1fcc183
November 7, 6:03:18 pm Duration: 2.74s Spans: 507

VS

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

Deep Linking to Raw Traces & Spans

Error: ”You have an outstanding balance…"



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

Service & Operation	
eats-gateway	/eats/v1/eaters/:eaterUuid/orders
eats-gateway	the-menu::WasSoGood
eats-gateway	i-got-lost::OnTheWay::ToTheJiffyStore
eats-gateway	abc-def::allYourBaseAreBelongToYou

abc-def::allYourBaseAreBelongToYou

Service: **eats-gateway** | Duration: **1.29s** | Start Time: **192ms**

Tags: span.kind = client | component = THE-component | error = true

Process: ip = 127.0.42.99 | jaeger.hostname = host-with-the-most | jaeger.version = version-ing | legacy-jaeger-client = 42.99.99

Logs (1)

1.48s

```
event         "error"
error.kind    "TChannelError"
error.object  {
                  info: {
                      message: "Please verify payment information to secure your account",
                      statusCode: 403,
                      shouldRetry: false,
                      stack: "*errors.errorString You have an outstanding balance due to a credit card problem. Please update your billing settings.
                      /there/are/many/paths/up/the/mountain:150 (0x1337b0)
                      /there/are/many/paths/up/the/mountain:74 (0x1337b0)
                      /there/are/many/paths/up/the/mountain:83 (0x1337b0)
                      /there/are/many/paths/up/the/mountain:118 (0x1337b0)
                      /there/are/many/paths/up/the/mountain:71 (0x1337b0)
                      /there/are/many/paths/up/the/mountain:36 (0x1337b0)
                      /there/are/many/paths/up/the/mountain:22 (0x1337b0)
                      /there/are/many/paths/up/the/mountain:729 (0x1337b0)
                      /there/are/many/paths/up/the/mountain:470 (0x1337b0)
                      /there/are/many/paths/up/the/mountain:458 (0x1337b0)
                      /there/are/many/paths/up/the/mountain:1269 (0x1337b0)
                      /there/are/many/paths/up/the/mountain:1030 (0x1337b0)
                      /there/are/many/paths/up/the/mountain:94 (0x1337b0)
                      /there/are/many/paths/up/the/mountain:163 (0x1337b0)
                      /there/are/many/paths/up/the/mountain:237 (0x1337b0)
                      /there/are/many/paths/up/the/mountain:118 (0x1337b0)
```

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

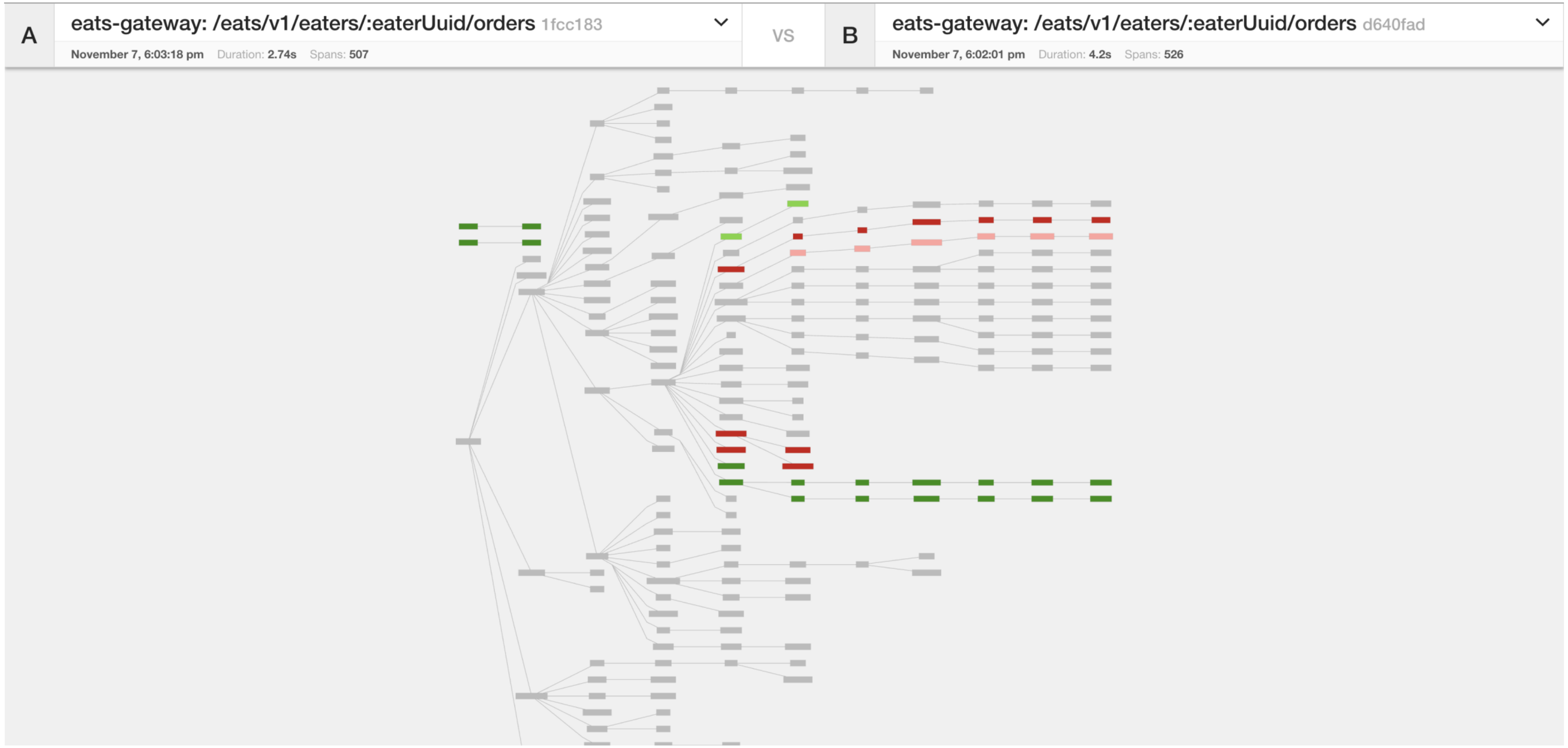
Production story

Migrating services to a nearby datacenter

Request latency doubles

Investigating latency

Structural comparison not always useful



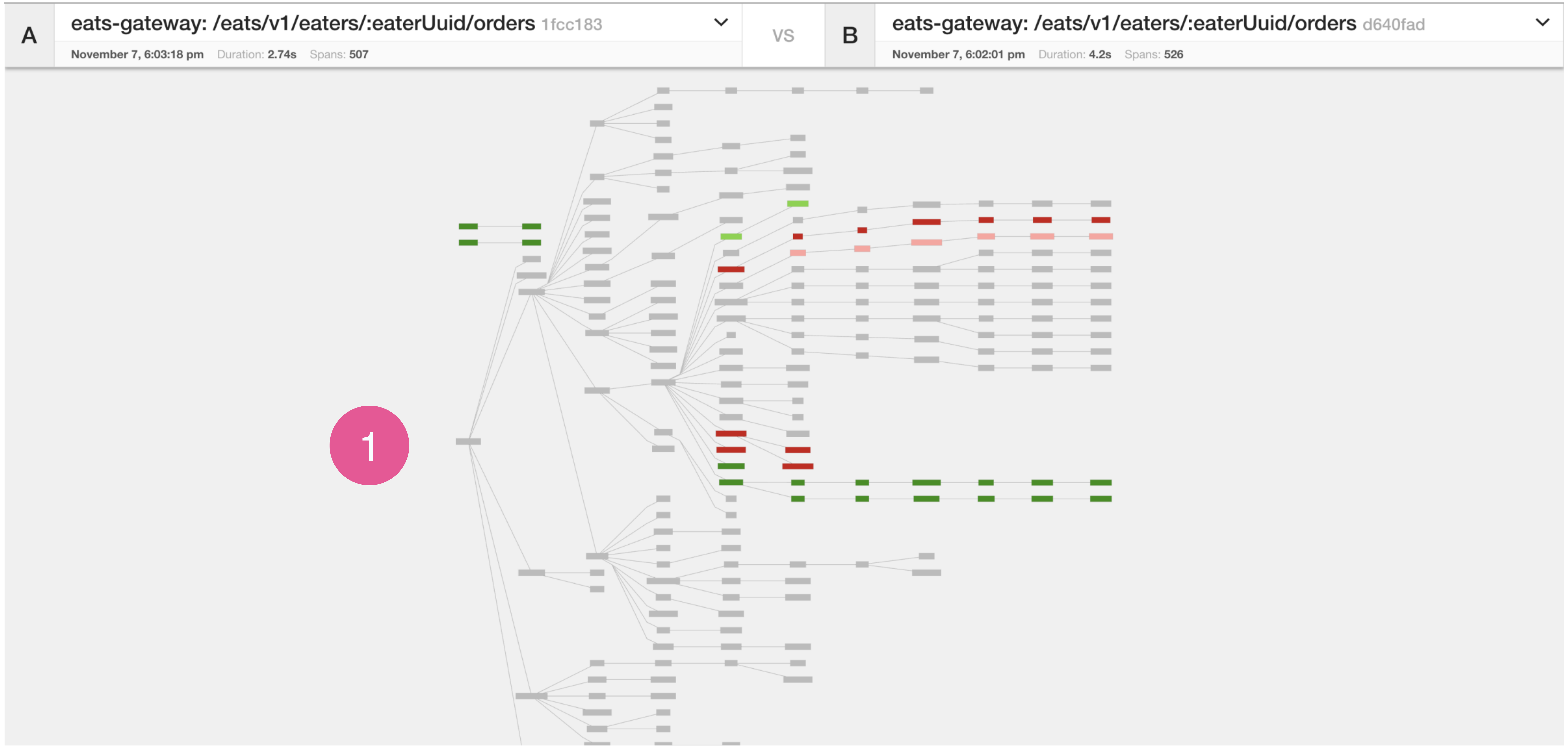
A eats-gateway: /eats/v1/eaters/:eaterUuid/orders 1fcc183
November 7, 6:03:18 pm Duration: 2.74s Spans: 507

VS

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

Investigating latency

Very similar structure



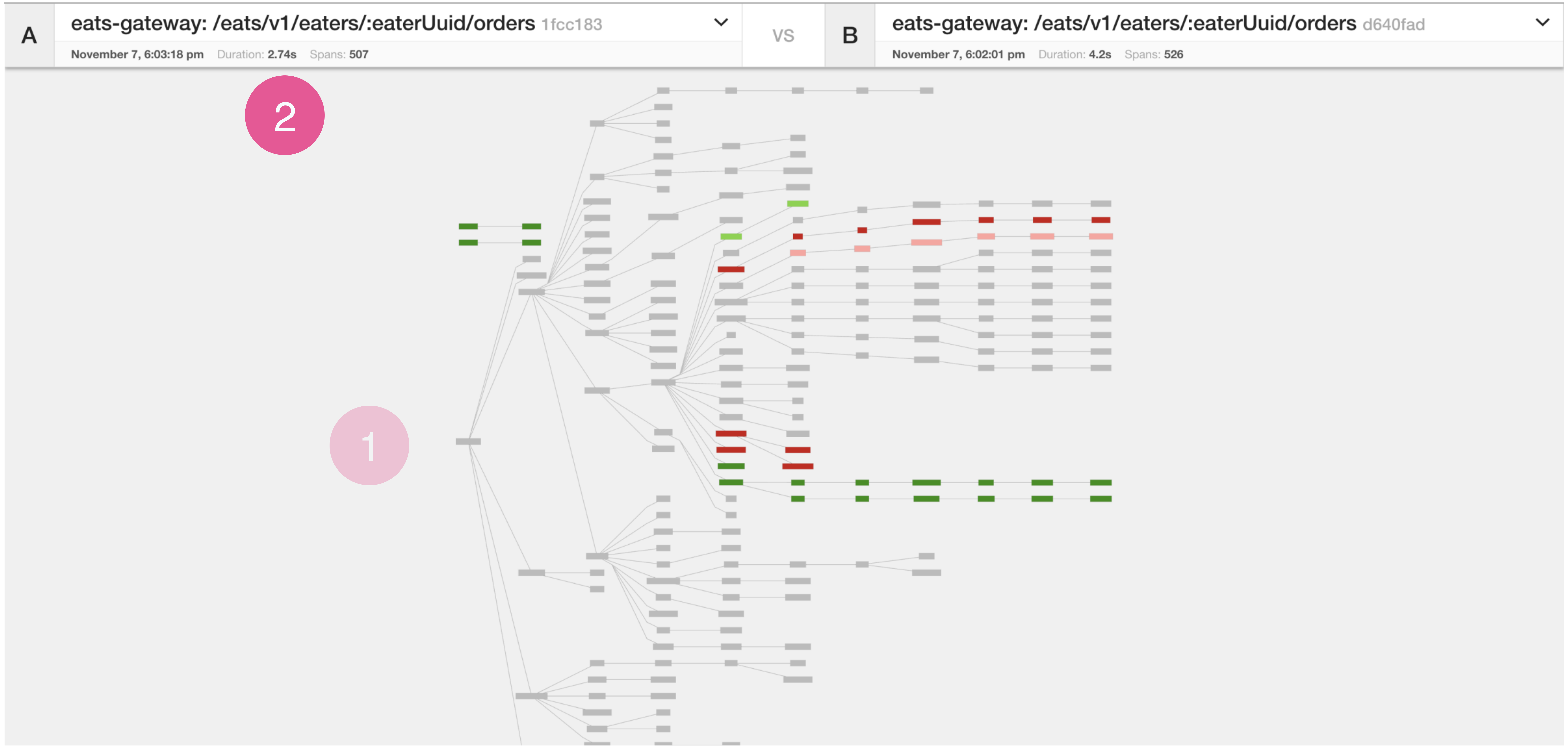
A | eats-gateway: /eats/v1/eaters/:eaterUuid/orders 1fcc183
November 7, 6:03:18 pm Duration: 2.74s Spans: 507

VS

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

Investigating latency

Left trace 2.74 seconds



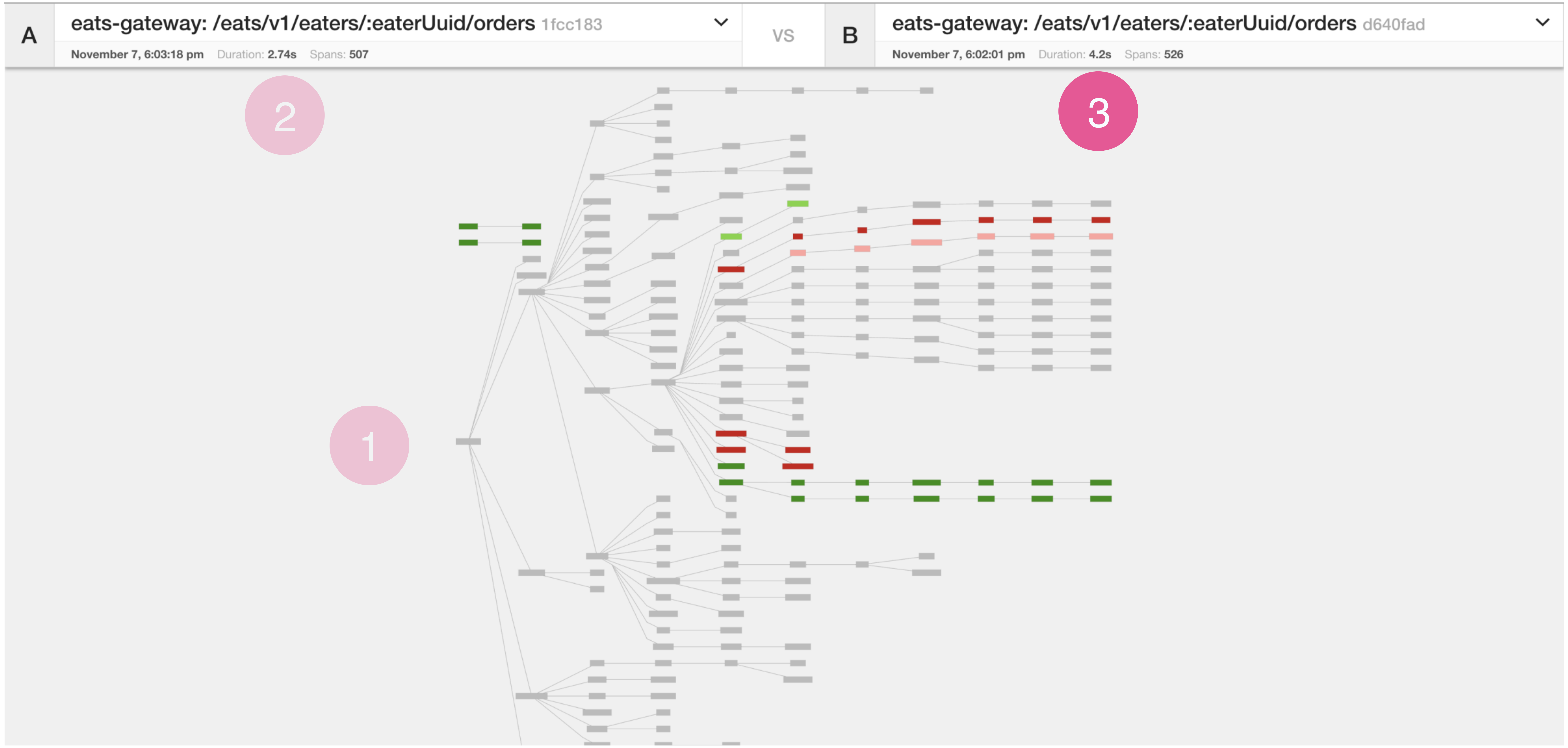
A | eats-gateway: /eats/v1/eaters/:eaterUuid/orders 1fcc183
November 7, 6:03:18 pm Duration: 2.74s Spans: 507

VS

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

Investigating latency

Right trace 4.2 seconds



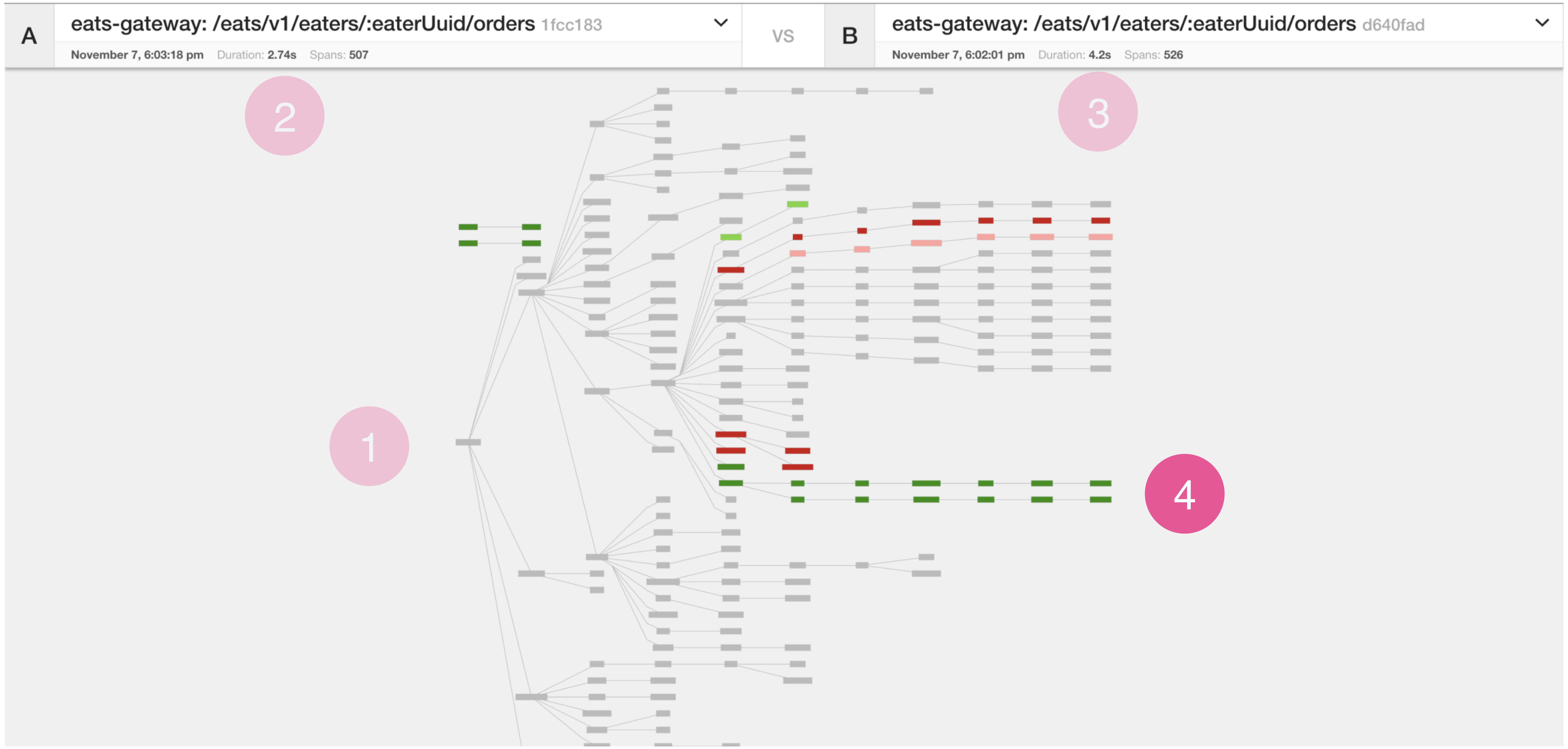
A | eats-gateway: /eats/v1/eaters/:eaterUuid/orders 1fcc183
November 7, 6:03:18 pm Duration: 2.74s Spans: 507

VS

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

Investigating latency

Due to structural differences?



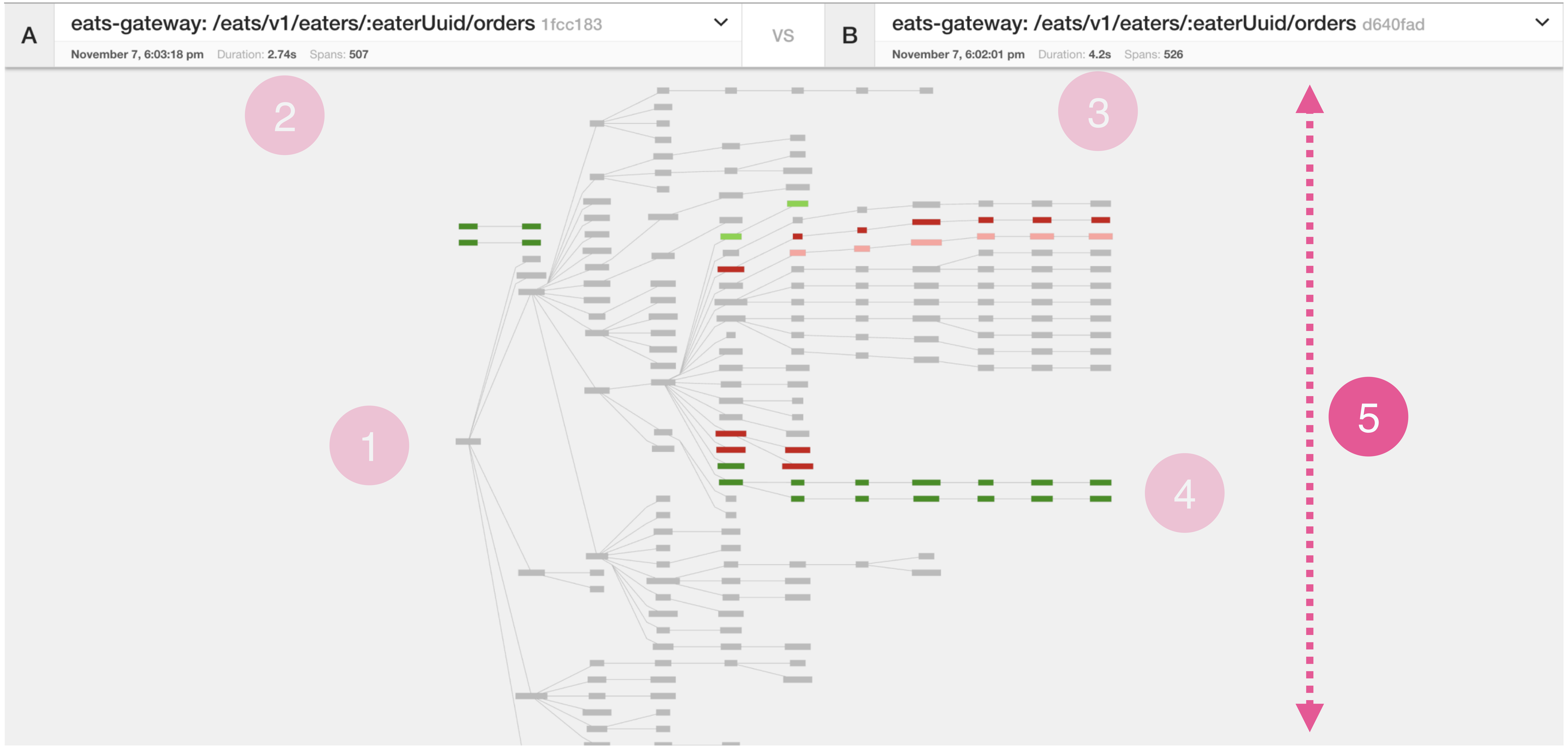
A eats-gateway: /eats/v1/eaters/:eaterUuid/orders 1fcc183
November 7, 6:03:18 pm Duration: 2.74s Spans: 507

VS

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

Investigating latency

Or dispersed contributors?



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

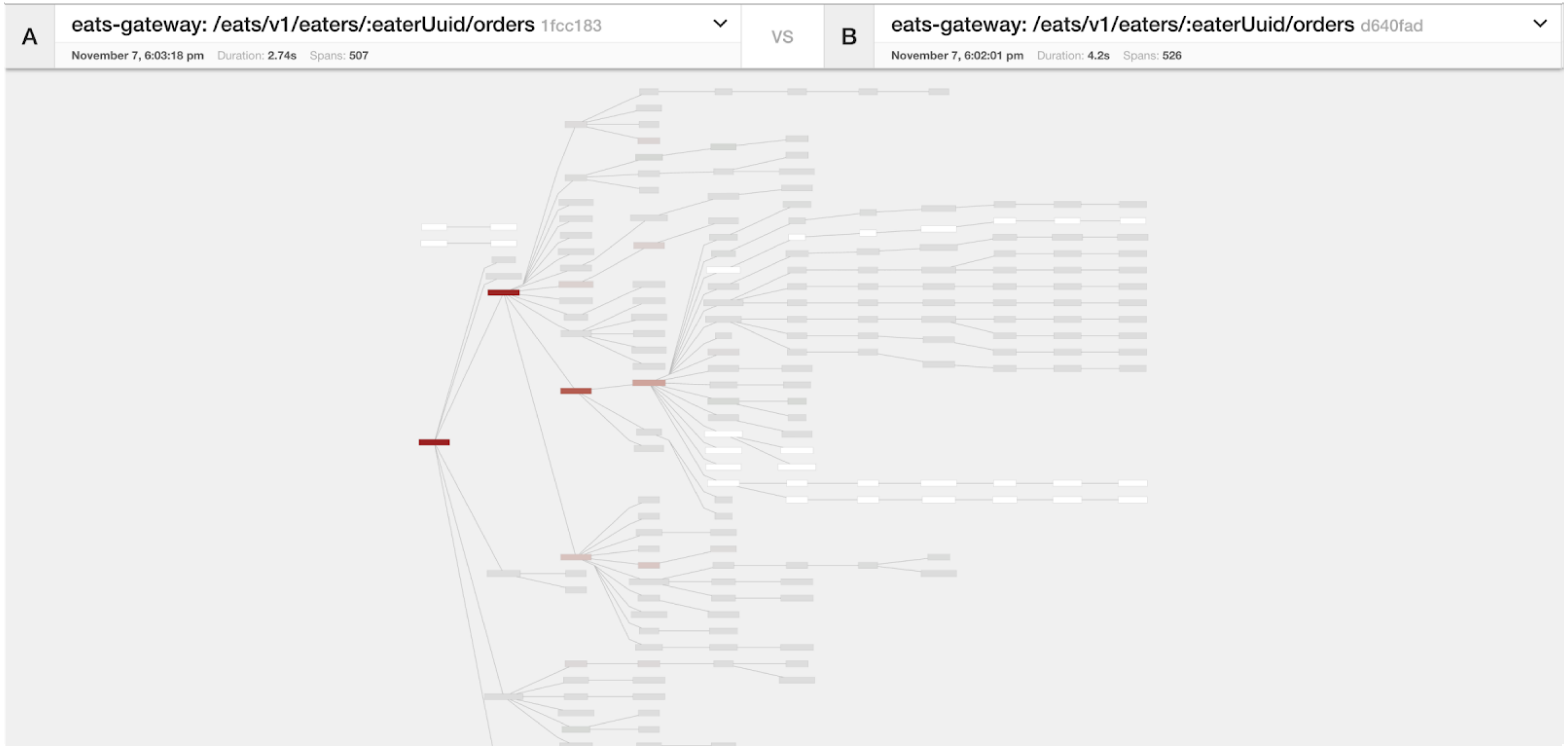
VS

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

Heat-maps!

Comparing trace durations

Heat-map of latencies



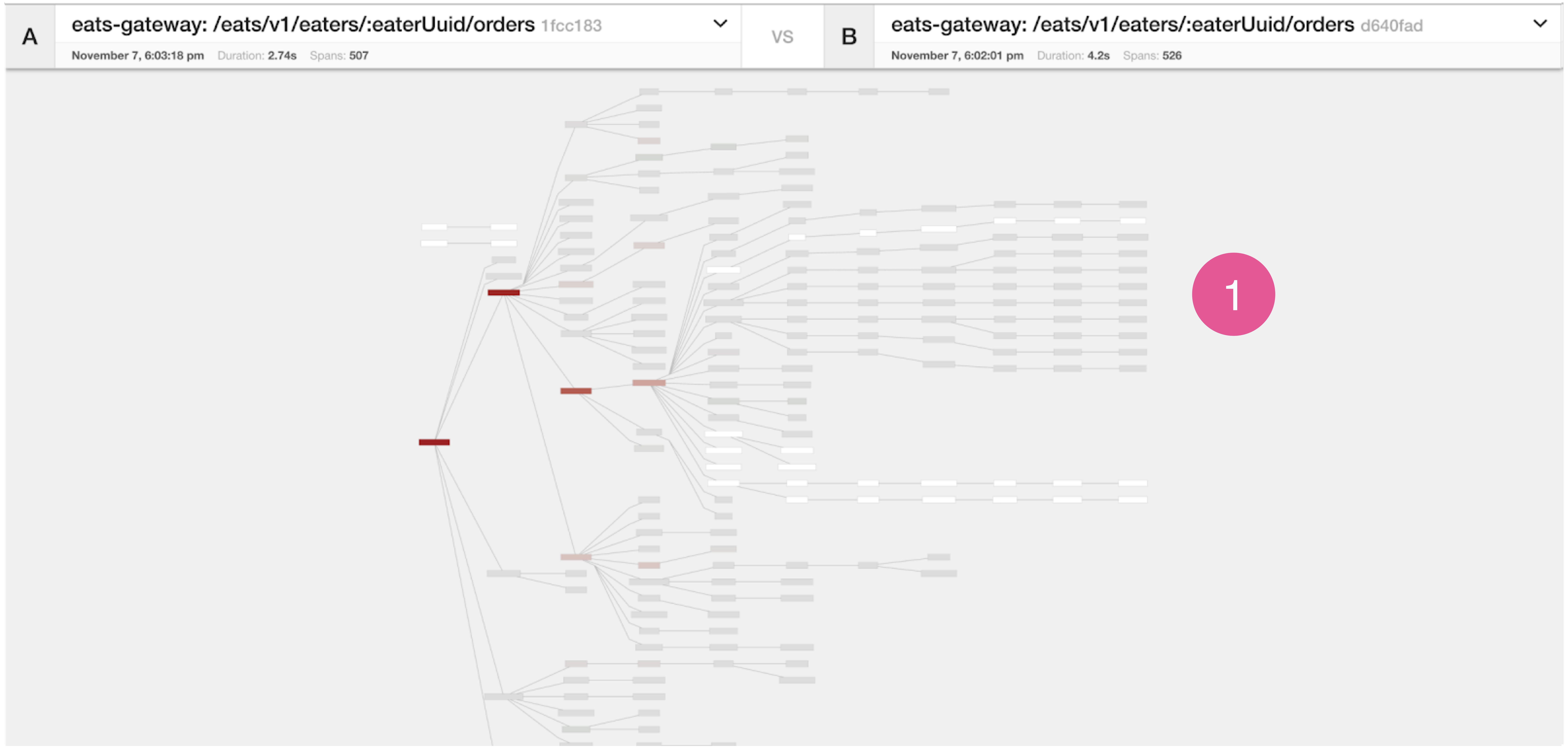
A eats-gateway: /eats/v1/eaters/:eaterUuid/orders 1fcc183
November 7, 6:03:18 pm Duration: 2.74s Spans: 507

VS

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

Comparing trace durations

Similar durations (grey)

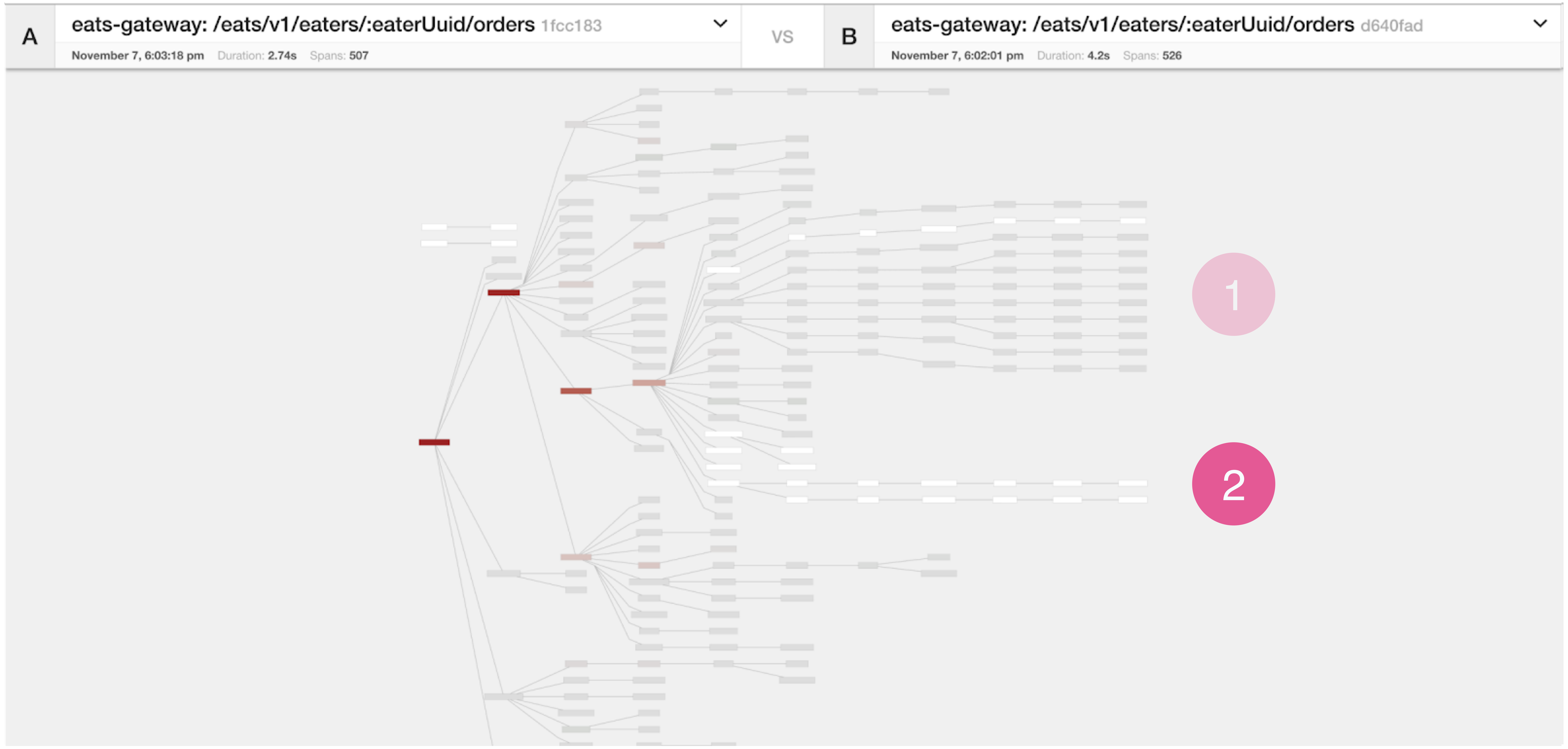


A | eats-gateway: /eats/v1/eaters/:eaterUuid/orders 1fcc183
November 7, 6:03:18 pm Duration: 2.74s Spans: 507

VS

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

Comparing trace durations
Nodes that are not shared (white)

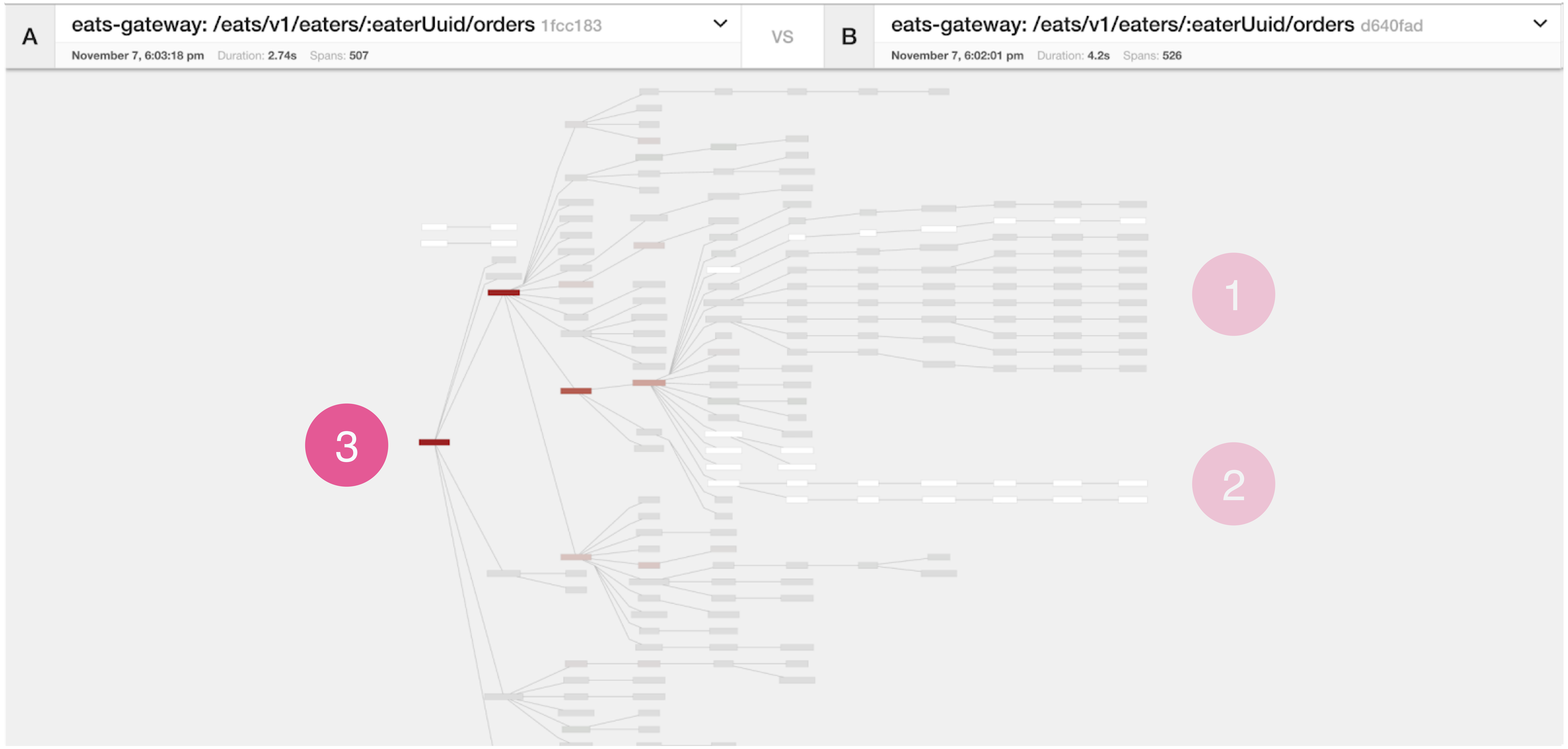

A | eats-gateway: /eats/v1/eaters/:eaterUuid/orders 1fcc183
November 7, 6:03:18 pm Duration: 2.74s Spans: 507

VS

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

Comparing trace durations

Red heat-map for latency differences



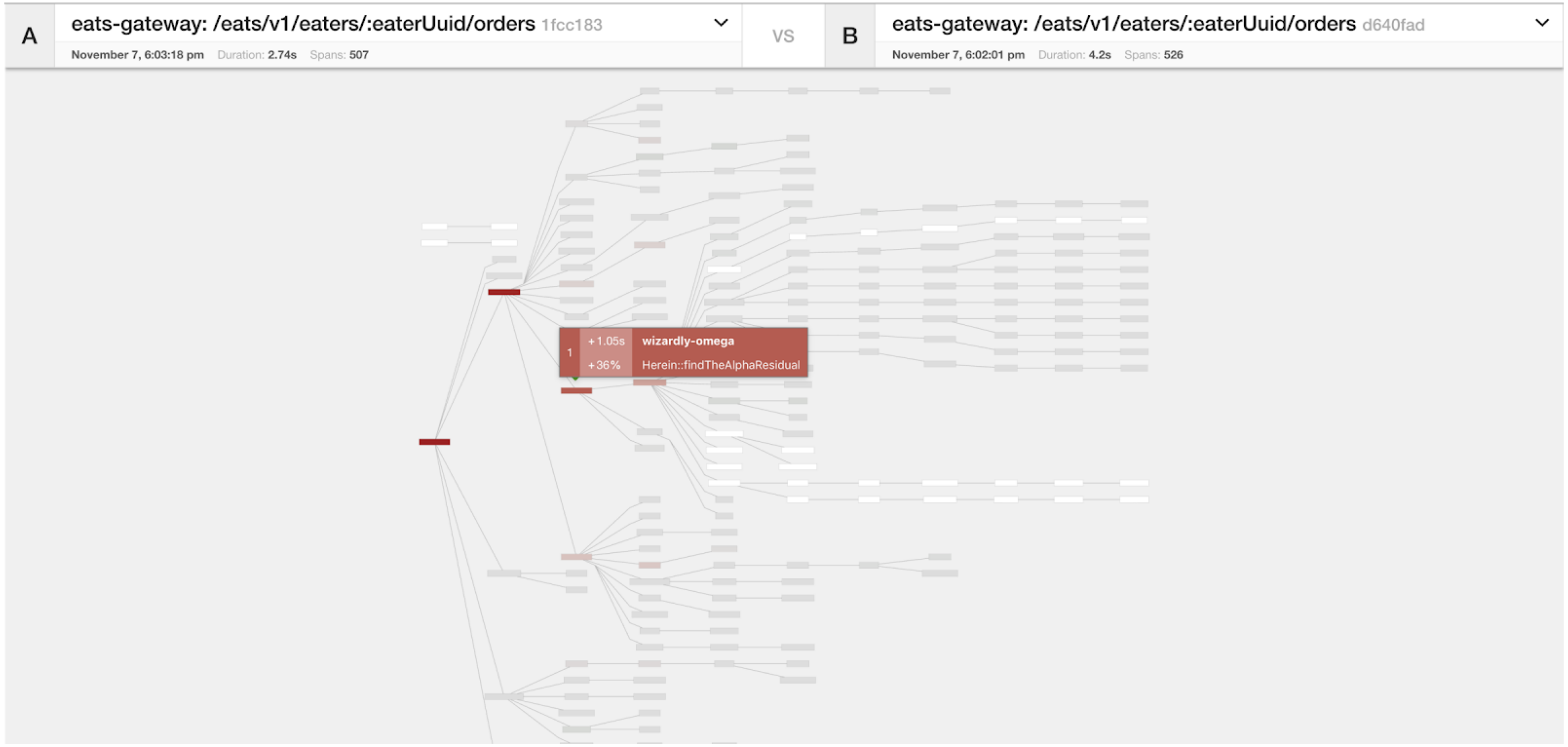
A eats-gateway: /eats/v1/eaters/:eaterUuid/orders 1fcc183
November 7, 6:03:18 pm Duration: 2.74s Spans: 507

VS

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

Comparing trace durations

Details on Mouse-Over



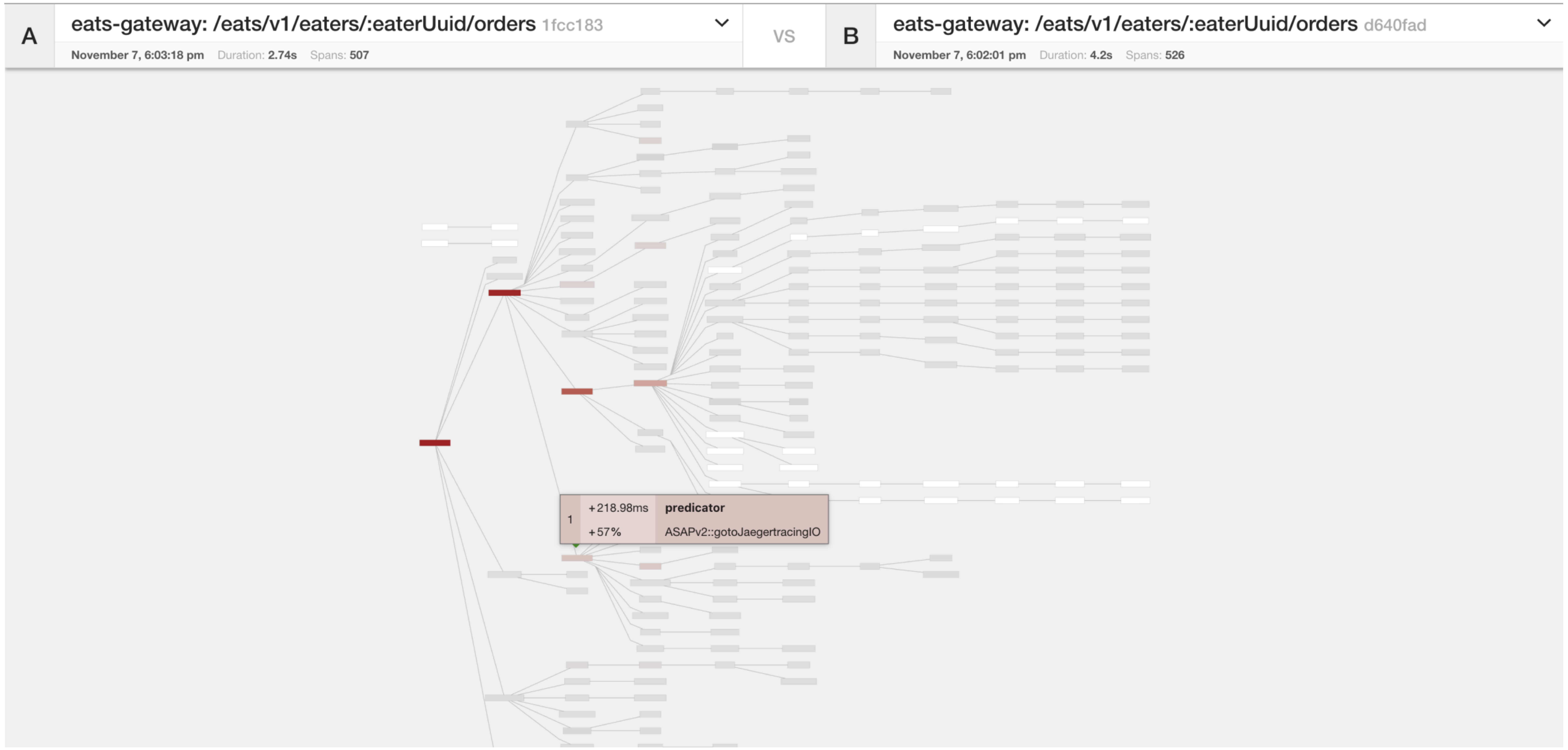
A eats-gateway: /eats/v1/eaters/:eaterUuid/orders 1fcc183
November 7, 6:03:18 pm Duration: 2.74s Spans: 507

VS

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

Comparing trace durations

Details on Mouse-Over



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

1	+218.98ms	**predicator**
	+57%	ASAPv2::gotoJaegertracingIO

How Are These Approach Different?

Summary

Surface less information

Condense the structural representation

Emphasize the differences

Distinct comparison modes simplify the comparisons

Challenges

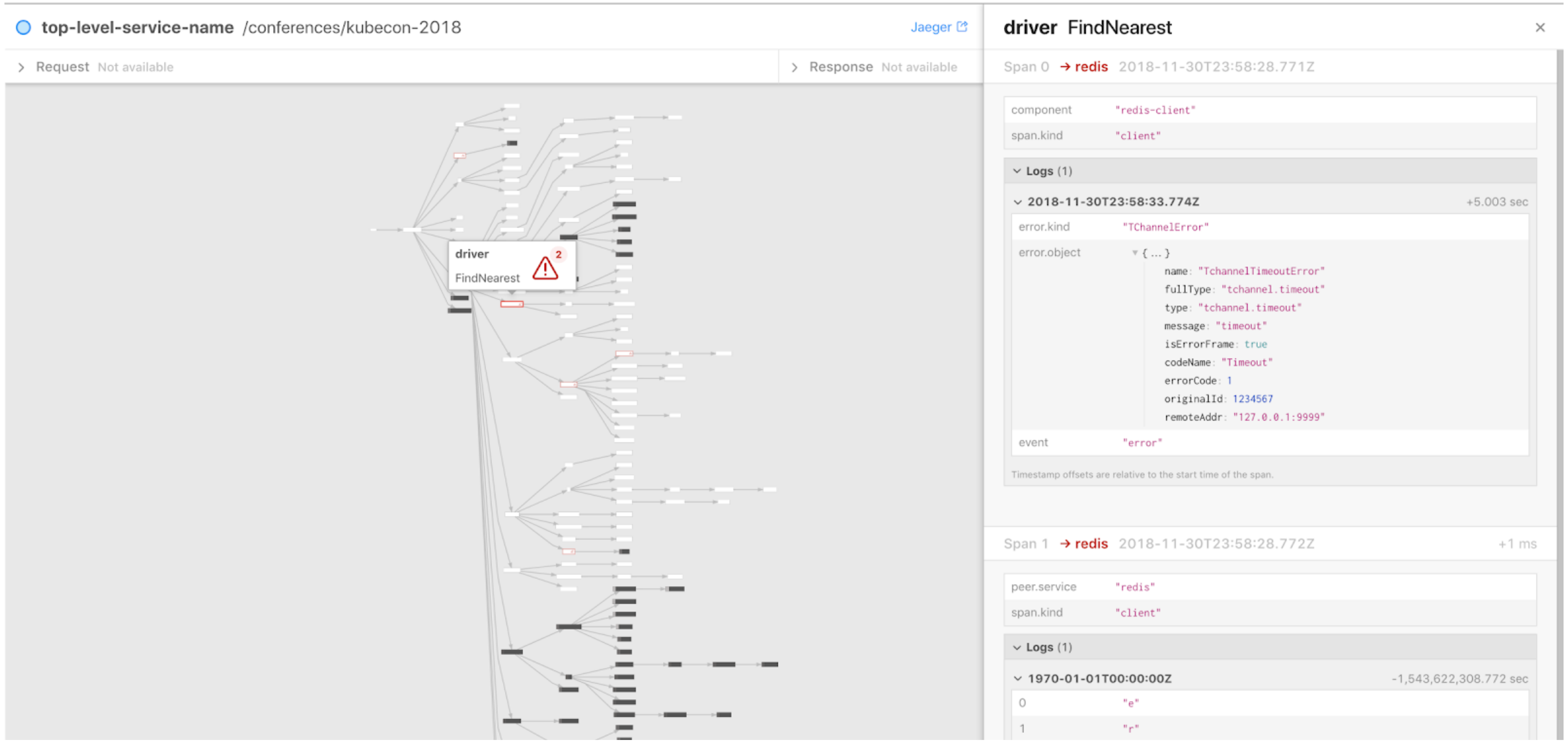
Individual traces can be an outliers.

User must find the right baseline.

Traces vs. Trace

What Went Wrong?

Root Cause Analysis



top-level-service-name /conferences/kubecon-2018 Jaeger

› Request Not available › Response Not available

driver FindNearest ×

Span 0 → **redis** 2018-11-30T23:58:28.771Z

```
component    "redis-client"
span.kind    "client"
```

⌄ **Logs** (1)

⌄ **2018-11-30T23:58:33.774Z** +5.003 sec

```
error.kind     "TChannelError"
error.object   ▾ { … }
                   name: "TchannelTimeoutError"
                   fullType: "tchannel.timeout"
                   type: "tchannel.timeout"
                   message: "timeout"
                   isErrorFrame: true
                   codeName: "Timeout"
                   errorCode: 1
                   originalId: 1234567
                   remoteAddr: "127.0.0.1:9999"
event          "error"
```

Timestamp offsets are relative to the start time of the span.

Span 1 → **redis** 2018-11-30T23:58:28.772Z +1 ms

```
peer.service   "redis"
span.kind      "client"
```

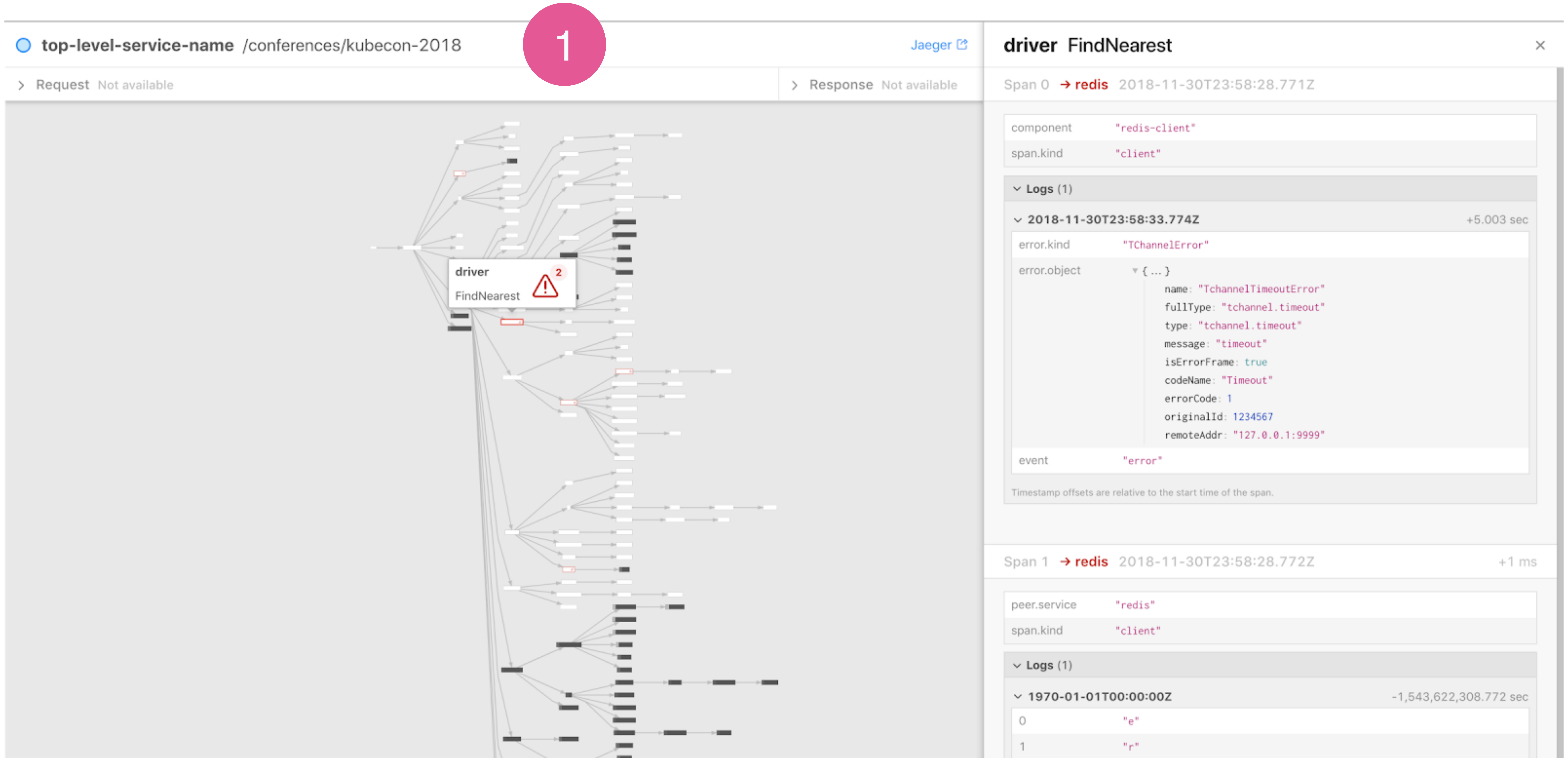
⌄ **Logs** (1)

⌄ **1970-01-01T00:00:00Z** -1,543,622,308.772 sec

```
0    "e"
1    "r"
```

Top Level Outcome

Including Request/Response Payloads



driver FindNearest

Span 0 → redis 2018-11-30T23:58:28.771Z

component	"redis-client"
span.kind	"client"

Logs (1)

2018-11-30T23:58:33.774Z +5.003 sec

```
error.kind      "TChannelError"
error.object    { ... }
                  name: "TchannelTimeoutError"
                  fullType: "tchannel.timeout"
                  type: "tchannel.timeout"
                  message: "timeout"
                  isErrorFrame: true
                  codeName: "Timeout"
                  errorCode: 1
                  originalId: 1234567
                  remoteAddr: "127.0.0.1:9999"
event           "error"
```

Timestamp offsets are relative to the start time of the span.

Span 1 → redis 2018-11-30T23:58:28.772Z +1 ms

peer.service	"redis"
span.kind	"client"

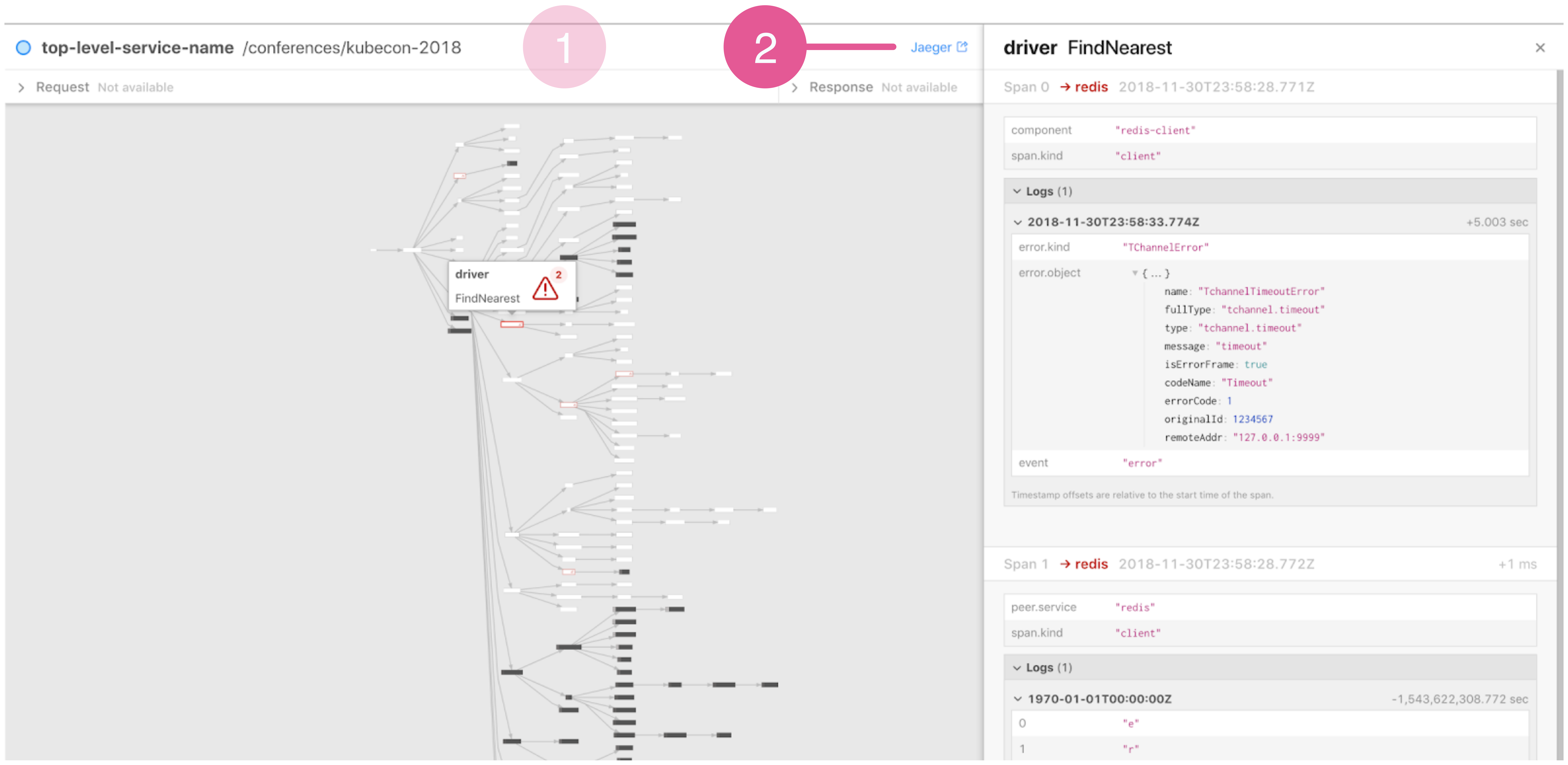
Logs (1)

1970-01-01T00:00:00Z -1,543,622,308.772 sec

0	"e"
1	"r"

Link to the Trace

Can Always Go Back to Raw Data



top-level-service-name /conferences/kubecon-2018

Jaeger

> Request Not available

> Response Not available

driver FindNearest

driver FindNearest ⚠ 2

Span 0 → redis 2018-11-30T23:58:28.771Z

component	"redis-client"
span.kind	"client"

Logs (1)

2018-11-30T23:58:33.774Z +5.003 sec

```
error.kind      "TChannelError"
error.object    { ... }
                  name: "TchannelTimeoutError"
                  fullType: "tchannel.timeout"
                  type: "tchannel.timeout"
                  message: "timeout"
                  isErrorFrame: true
                  codeName: "Timeout"
                  errorCode: 1
                  originalId: 1234567
                  remoteAddr: "127.0.0.1:9999"
event           "error"
```

Timestamp offsets are relative to the start time of the span.

Span 1 → redis 2018-11-30T23:58:28.772Z +1 ms

peer.service	"redis"
span.kind	"client"

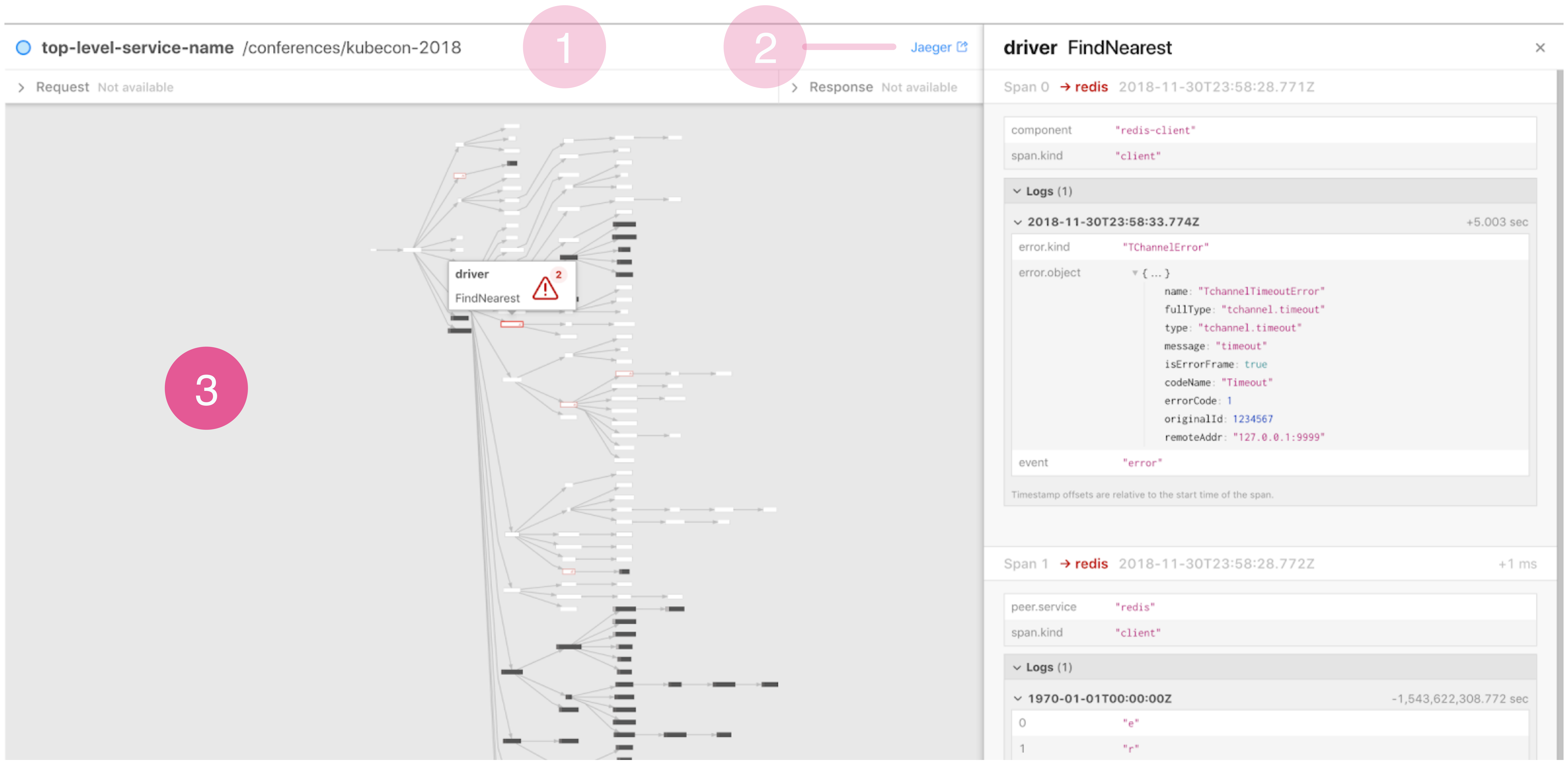
Logs (1)

1970-01-01T00:00:00Z -1,543,622,308.772 sec

```
0    "e"
1    "r"
```

Trace Structure

Nodes Are Sorted Chronologically



top-level-service-name /conferences/kubecon-2018

Request Not available

Response Not available

driver FindNearest

Span 0 → redis 2018-11-30T23:58:28.771Z

component	"redis-client"
span.kind	"client"

Logs (1)

2018-11-30T23:58:33.774Z +5.003 sec

```
error.kind      "TChannelError"
error.object    { ... }
                    name: "TchannelTimeoutError"
                    fullType: "tchannel.timeout"
                    type: "tchannel.timeout"
                    message: "timeout"
                    isErrorFrame: true
                    codeName: "Timeout"
                    errorCode: 1
                    originalId: 1234567
                    remoteAddr: "127.0.0.1:9999"
event           "error"
```

Timestamp offsets are relative to the start time of the span.

Span 1 → redis 2018-11-30T23:58:28.772Z +1 ms

peer.service	"redis"
span.kind	"client"

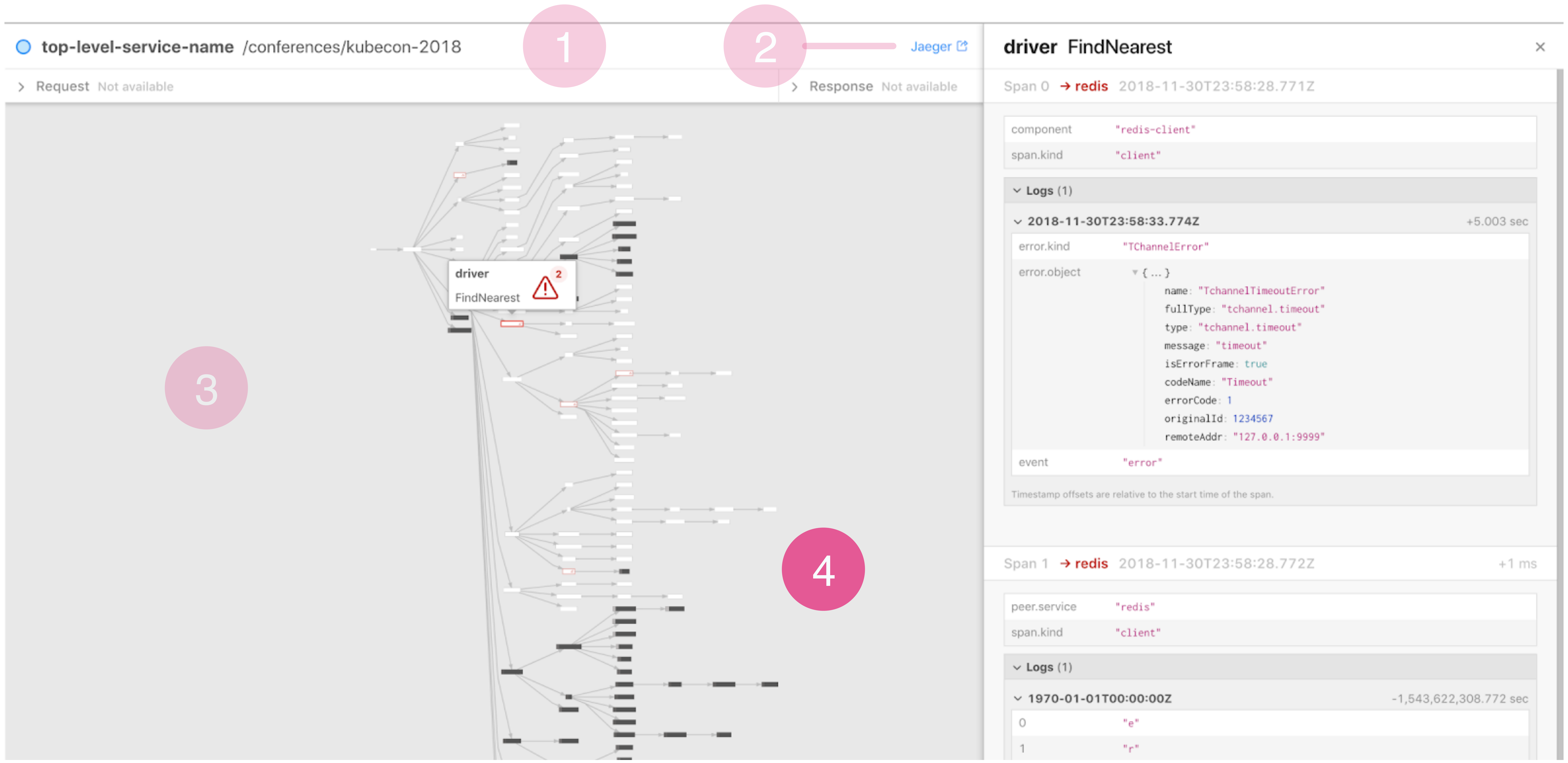
Logs (1)

1970-01-01T00:00:00Z -1,543,622,308.772 sec

0	"e"
1	"r"

Present and Missing Nodes

Color-Coding



top-level-service-name /conferences/kubecon-2018

Jaeger

> Request Not available

> Response Not available

driver FindNearest ×

Span 0 → redis 2018-11-30T23:58:28.771Z

```
component        "redis-client"
span.kind        "client"
```

⌄ Logs (1)

⌄ 2018-11-30T23:58:33.774Z +5.003 sec

```
error.kind       "TChannelError"
error.object     ▾ { … }
                     name: "TchannelTimeoutError"
                     fullType: "tchannel.timeout"
                     type: "tchannel.timeout"
                     message: "timeout"
                     isErrorFrame: true
                     codeName: "Timeout"
                     errorCode: 1
                     originalId: 1234567
                     remoteAddr: "127.0.0.1:9999"
event            "error"
```

Timestamp offsets are relative to the start time of the span.

Span 1 → redis 2018-11-30T23:58:28.772Z +1 ms

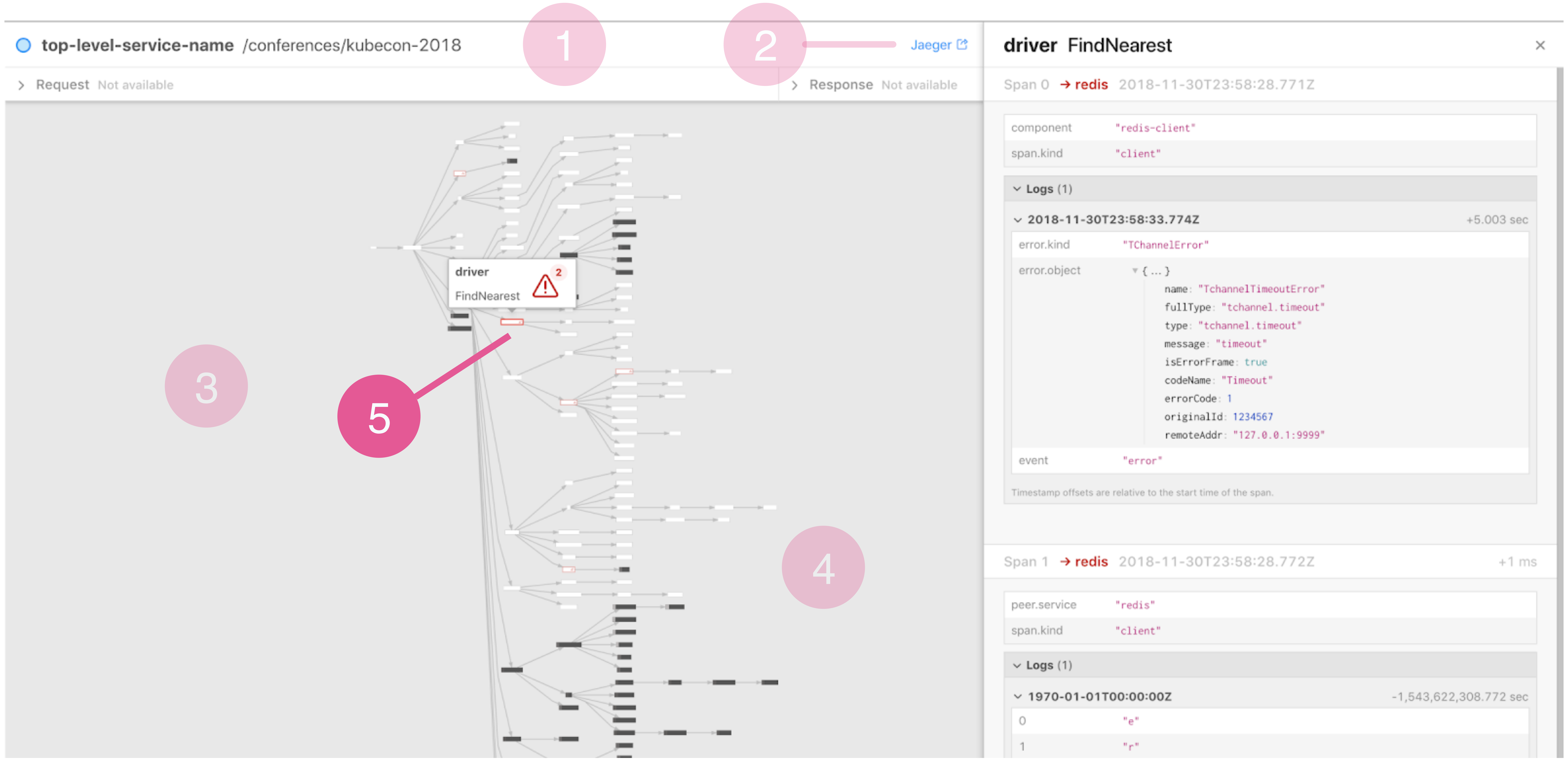
```
peer.service     "redis"
span.kind        "client"
```

⌄ Logs (1)

⌄ 1970-01-01T00:00:00Z -1,543,622,308.772 sec

```
0                "e"
1                "r"
```

A Node With Error Data



driver FindNearest

Span 0 → **redis** 2018-11-30T23:58:28.771Z

component	"redis-client"
span.kind	"client"

Logs (1)

2018-11-30T23:58:33.774Z +5.003 sec

```
error.kind      "TChannelError"
error.object    { ... }
                  name: "TchannelTimeoutError"
                  fullType: "tchannel.timeout"
                  type: "tchannel.timeout"
                  message: "timeout"
                  isErrorFrame: true
                  codeName: "Timeout"
                  errorCode: 1
                  originalId: 1234567
                  remoteAddr: "127.0.0.1:9999"
event           "error"
```

Timestamp offsets are relative to the start time of the span.

Span 1 → **redis** 2018-11-30T23:58:28.772Z +1 ms

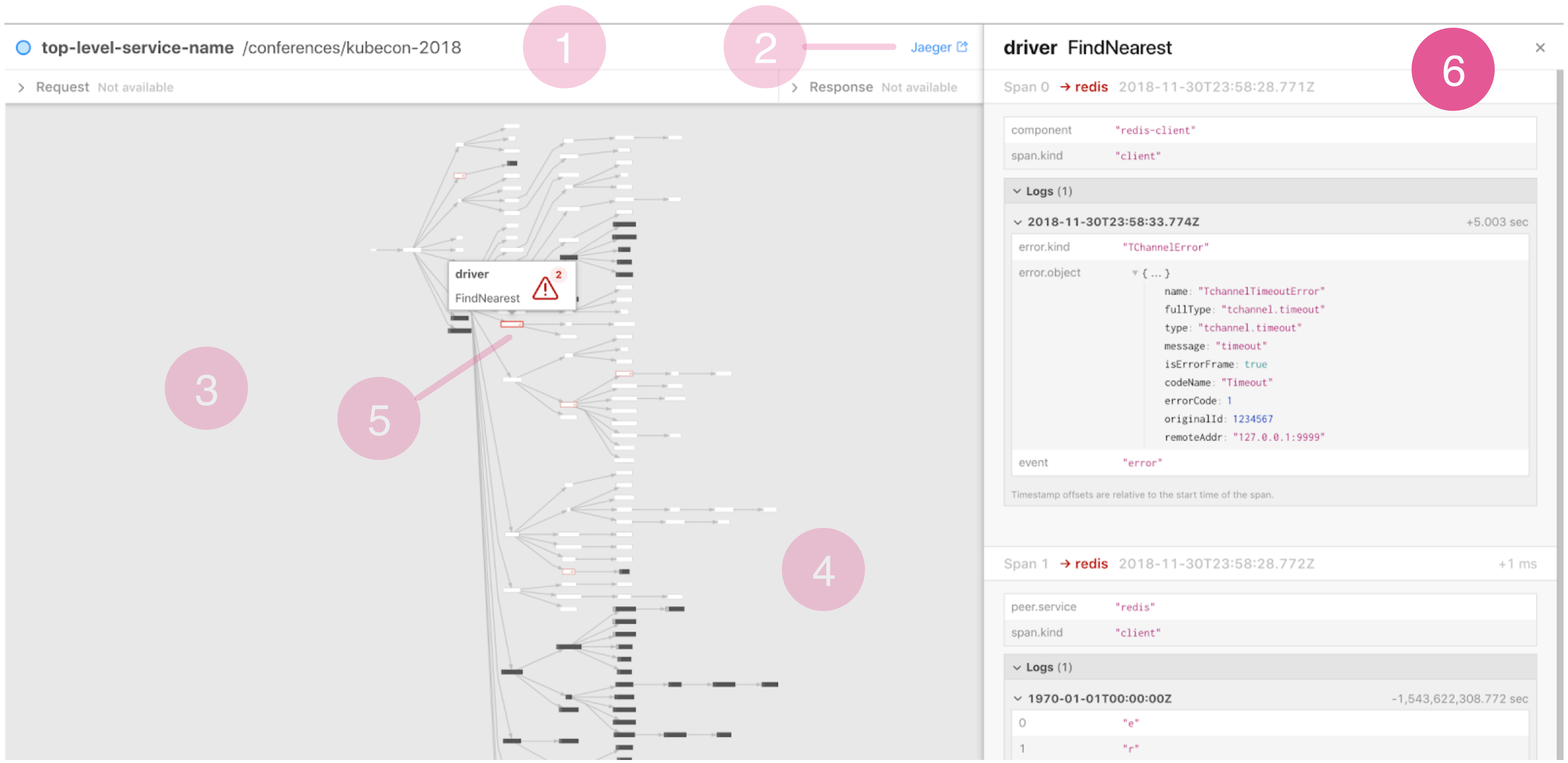
peer.service	"redis"
span.kind	"client"

Logs (1)

1970-01-01T00:00:00Z -1,543,622,308.772 sec

0	"e"
1	"r"

Error Data Panel



top-level-service-name /conferences/kubecon-2018

> Request Not available

> Response Not available

driver FindNearest ×

Span 0 → **redis** 2018-11-30T23:58:28.771Z

component	"redis-client"
span.kind	"client"

⌄ **Logs** (1)

⌄ **2018-11-30T23:58:33.774Z** +5.003 sec

```
error.kind      "TChannelError"
error.object    ▾ { … }
                    name: "TchannelTimeoutError"
                    fullType: "tchannel.timeout"
                    type: "tchannel.timeout"
                    message: "timeout"
                    isErrorFrame: true
                    codeName: "Timeout"
                    errorCode: 1
                    originalId: 1234567
                    remoteAddr: "127.0.0.1:9999"
event           "error"
```

Timestamp offsets are relative to the start time of the span.

Span 1 → **redis** 2018-11-30T23:58:28.772Z +1 ms

peer.service	"redis"
span.kind	"client"

⌄ **Logs** (1)

⌄ **1970-01-01T00:00:00Z** -1,543,622,308.772 sec

0	"e"
1	"r"

How Is This Approach Different?

Summary

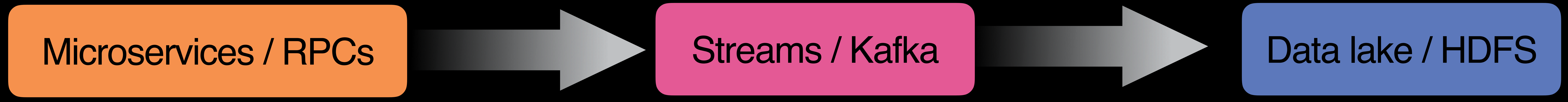
Much broader context: aggregate vs. one trace

One purpose: root cause analysis of reliability issues

Tackling Data Complexity

Uber is a data company

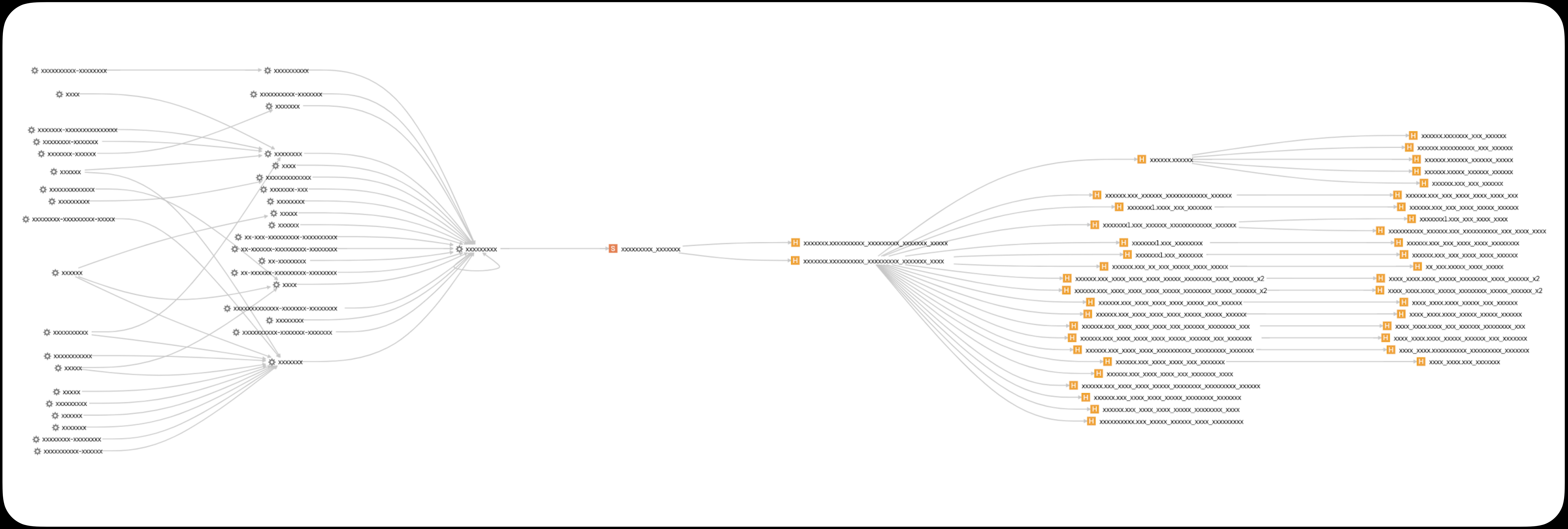
OK, and a transportation company

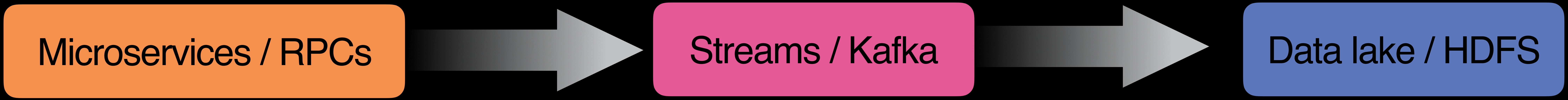


- Data undergoes many transformations
- More data is derived from other data
- Debugging data quality is difficult

Data Lineage

Debugging Data Quality

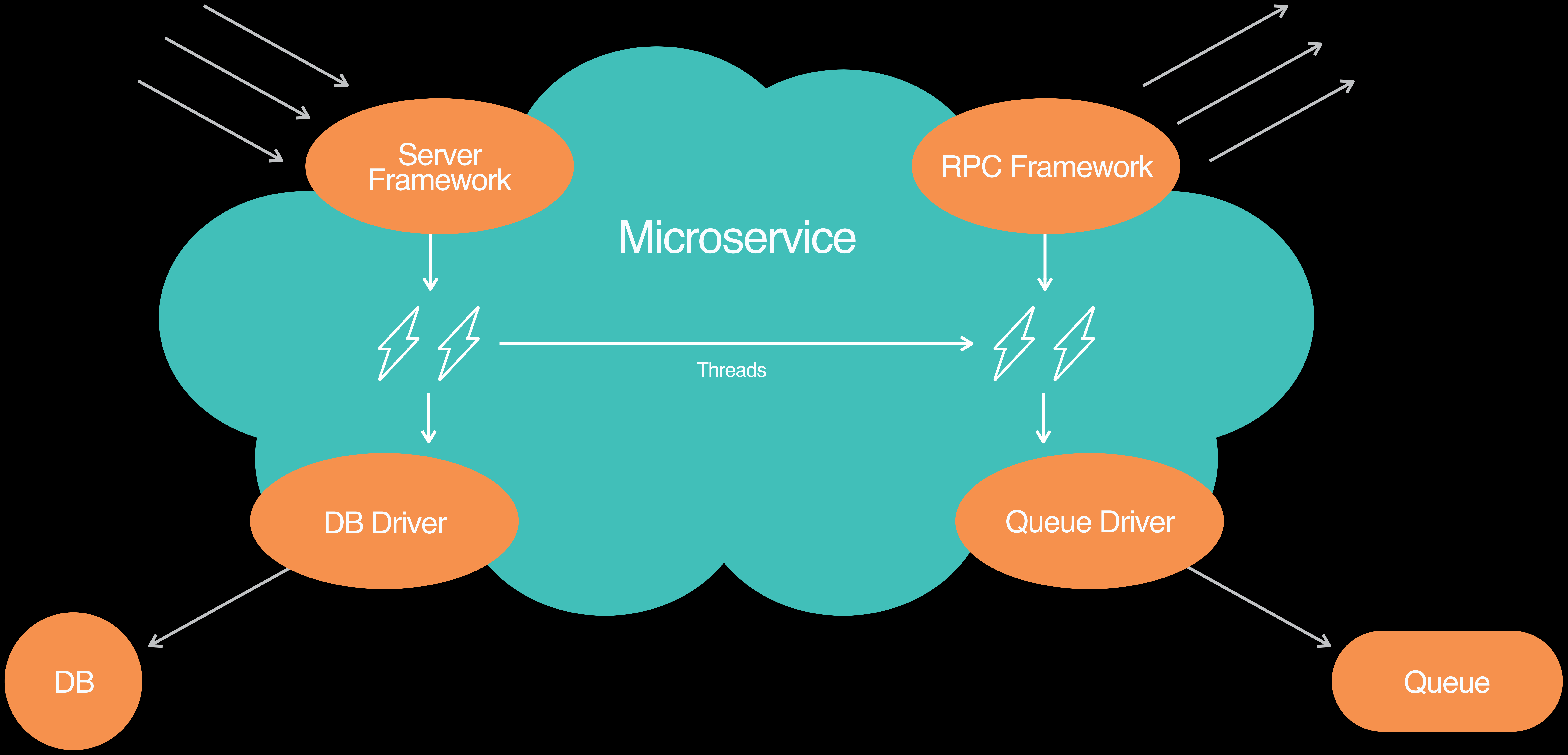




Observability requires
high quality instrumentation.

Our Software Is Highly Composable

Often from Open Source Components



Tracing breaks if components don’t understand each other.

Standardization Efforts
Instrumentation and Data Formats

OpenTelemetry

- Effective observability requires high-quality telemetry.
- OpenTelemetry makes robust, portable telemetry a built-in feature of cloud-native software.

W3C®

- Distributed Tracing Working Group
- Data formats for on-the-wire trace context & correlation-context, and out-of-band trace data.

In Summary

Distributed tracing helps us
to deal with the overwhelming
complexity of microservices

In Summary

Creative visualizations
are essential
in performance analysis

In Summary

Distributed tracing empowers
unparalleled insights
into our distributed systems

Thank You

Find me @ shkuro.com

Q&A