



# Jaeger Project Intro

Pavol Loffay (Red Hat), Yuri Shkuro (Uber)

CloudNativeCon NA, San Diego, Nov-19-2019

# Agenda

- What is tracing
- Demo
- Project status
- New Features
- Roadmap
- Q & A

# About

- Pavol Loffay (<https://github.com/pavolloffay>)
  - Software engineer at Red Hat
  - Maintainer of Jaeger, OpenTracing, OpenTelemetry
- Yuri Shkuro (<https://github.com/yurishkuro>)
  - Software engineer at Uber Technologies
  - Maintainer of Jaeger, OpenTracing, OpenTelemetry
  - Author of “[Mastering Distributed Tracing](#)” book



**CLOUD NATIVE**  
COMPUTING FOUNDATION

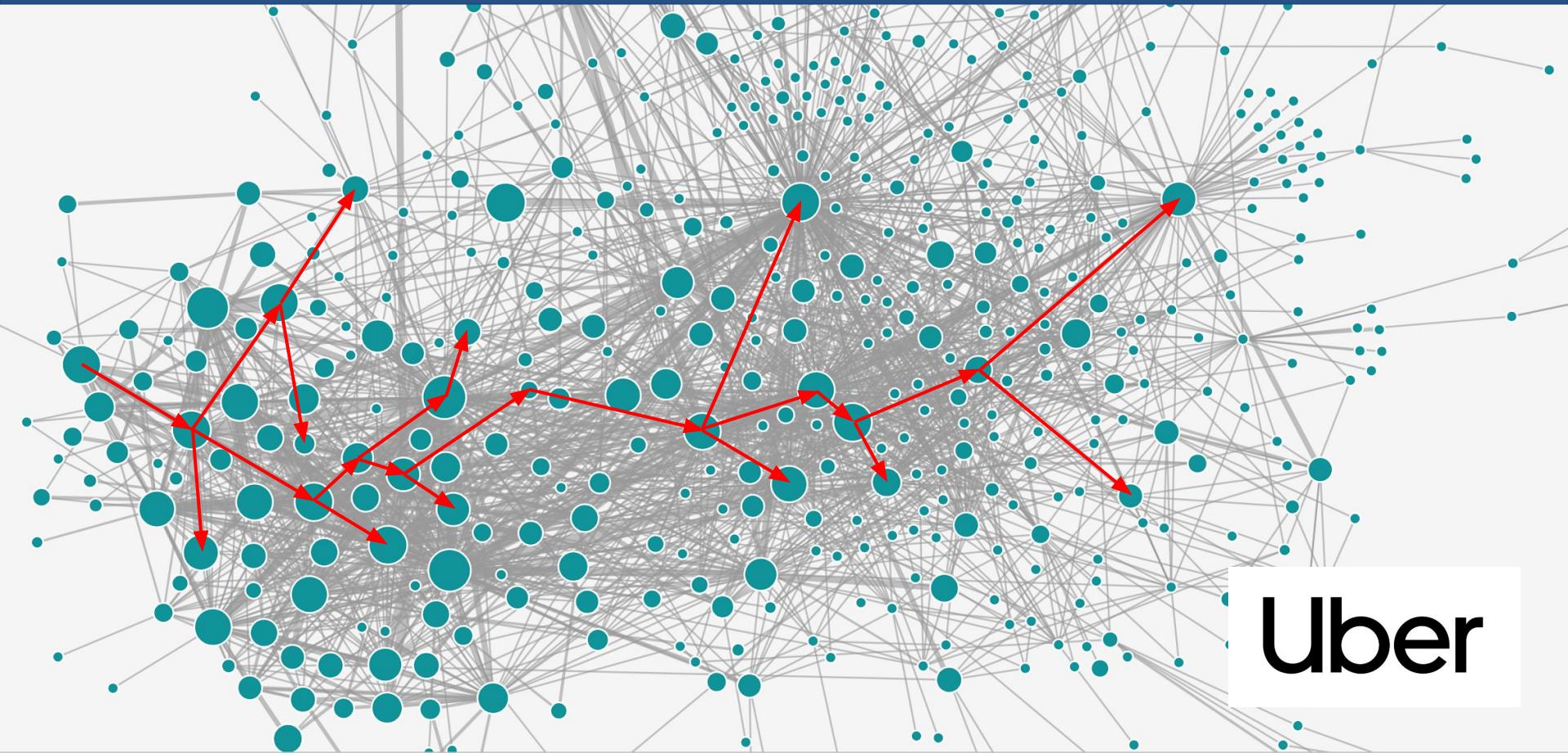
# What is Tracing & Why?

Concepts and terminology

# Modern Distributed Systems are COMPLEX

Loading Netflix or Facebook home page ⇒  
dozens of microservices, 100s of nodes

# BILLIONS of times a day!

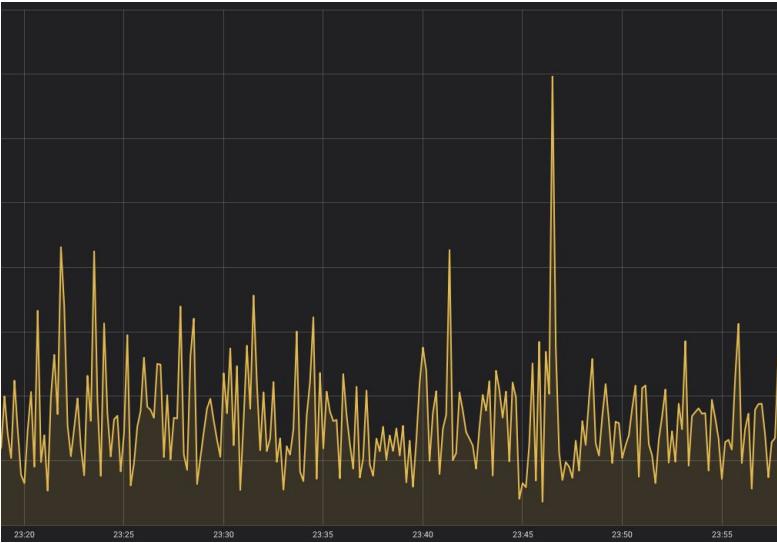


Uber

# How can we tell what is going on?

Which service is to blame  
when things go wrong or become slow?

# Traditional monitoring tools don't help



Metrics show something is wrong, but do not explain why.

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

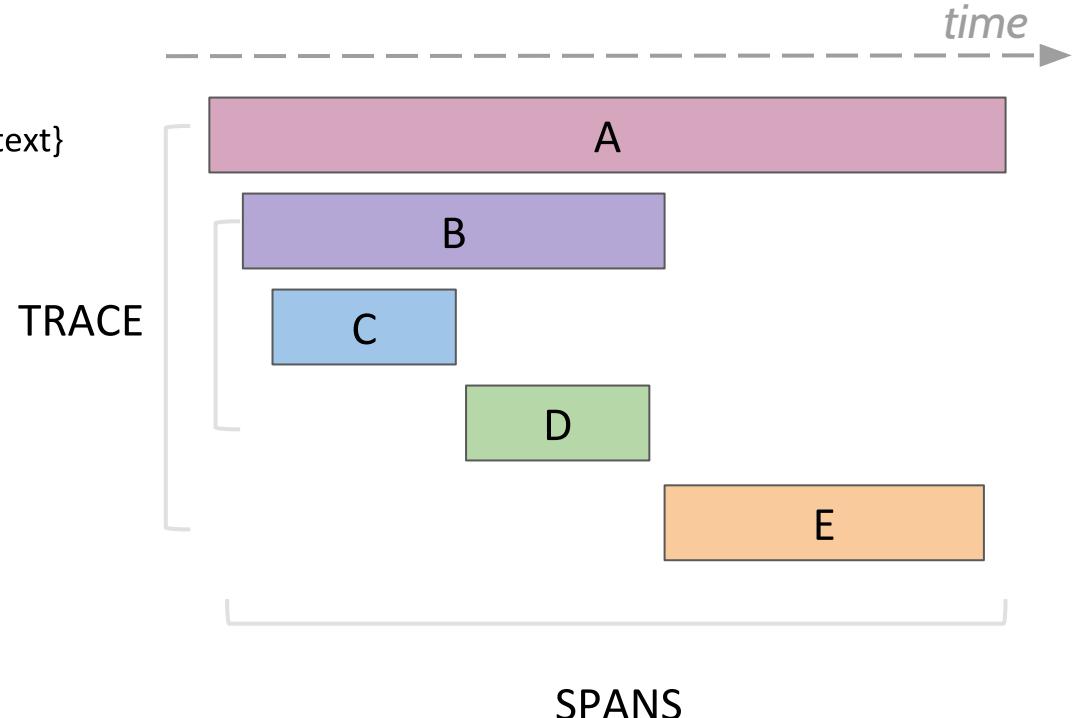
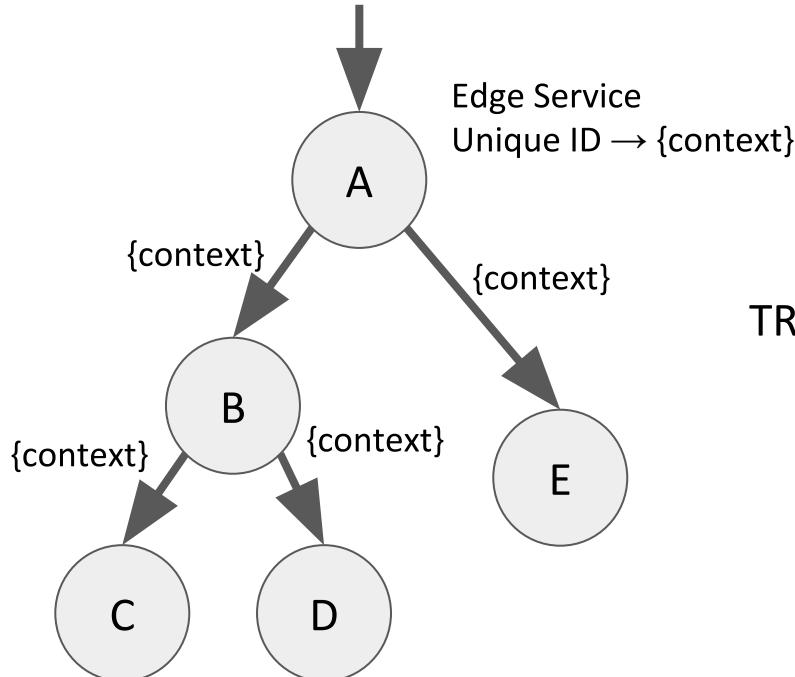
# Monitoring tools must tell stories!

Do you like debugging  
without a stack trace?

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



# Context Propagation & Distributed Tracing





Let's look at some traces

<http://bit.do/jaeger-hotrod>



# Service dependencies diagram

Jaeger UI

Lookup by Trace ID...

Search

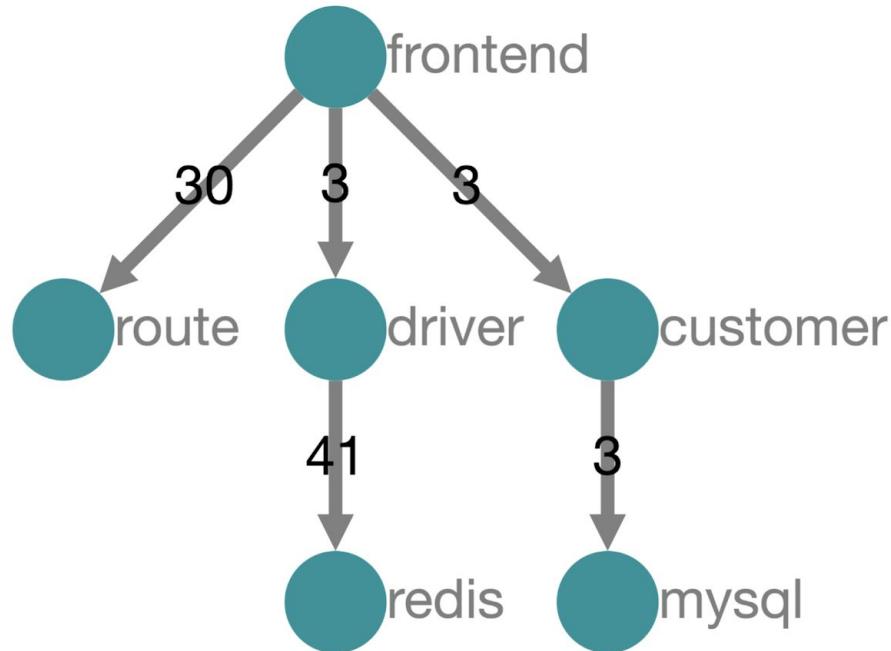
Compare

Dependencies

About Jaeger

Force Directed Graph

DAG



# Trace timeline

## frontend: HTTP GET /dispatch

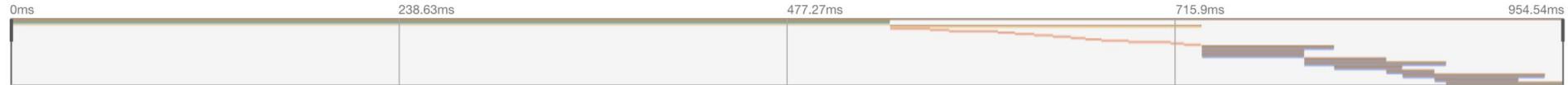


Search...



View Options

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



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		25.02ms			

# Trace timeline – Parent → Child → Grandchild

## ✓ frontend: HTTP GET /dispatch



Search...



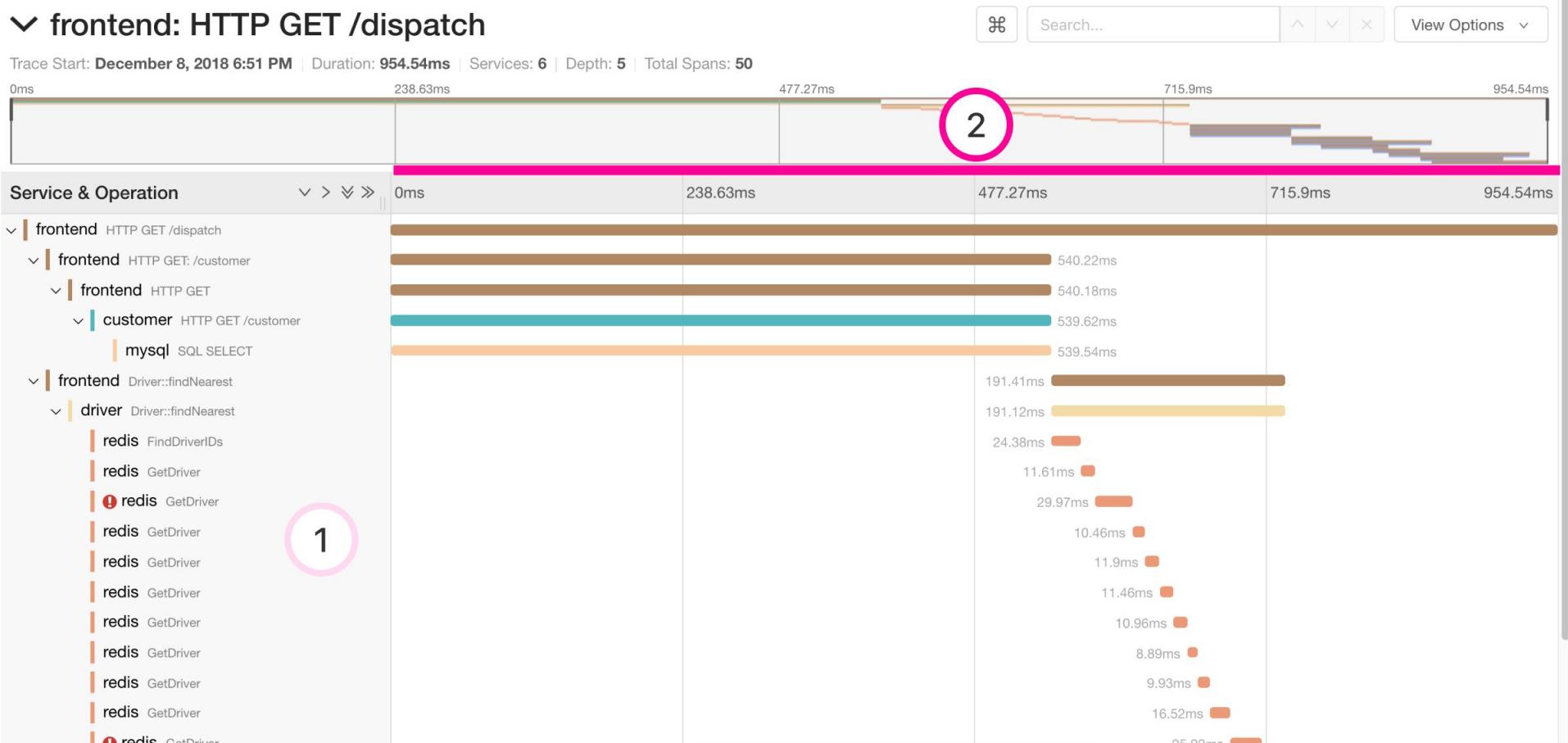
Trace Start: December 8, 2018 6:51 PM | Duration: 954.54ms | Services: 6 | Depth: 5 | Total Spans: 50



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				25.92ms	

1

# Trace timeline – Time + Mini-map



# Trace timeline – A blocking operation

## ✓ 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



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				25.92ms	

1

3

2

# Trace timeline – Sequential operations

## ✓ 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



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				25.92ms	
redis GetDriver					

1

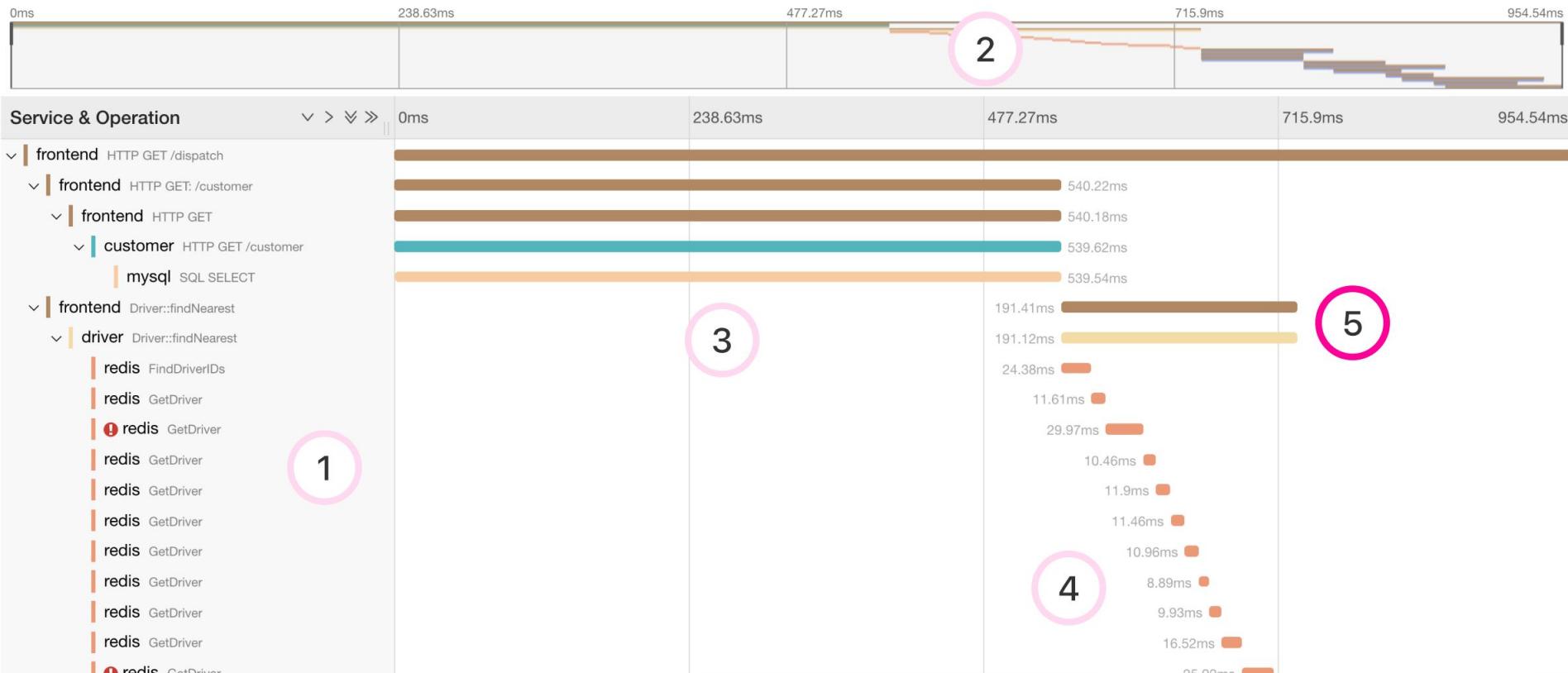
3

4

# Trace timeline – Parents encompass descendants (generally)

## ✓ frontend: HTTP GET /dispatch

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



# Span details

> frontend: HTTP GET /dispatch

Service & Operation    v > v > 0ms    238.63ms    477.27ms    715.9ms    954.54ms

Search...    View Options

frontend HTTP GET /dispatch

frontend HTTP GET: /customer

  frontend HTTP GET

    customer HTTP GET /customer

mysql SQL SELECT

**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

driver Driver::findNearest

# Span details – Database query

> frontend: HTTP GET /dispatch

Service & Operation    0ms    238.63ms    477.27ms    715.9ms    954.54ms

frontend HTTP GET /dispatch  
  frontend HTTP GET: /customer  
    frontend HTTP GET  
      customer HTTP GET /customer  
mysql SQL SELECT

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

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

SpanID: 7aecd811f9df684

frontend Driver::findNearest  
  driver Driver::findNearest

191.41ms  
191.12ms

# Span details – Lock contention

> frontend: HTTP GET /dispatch

Service & Operation    v > v > 0ms    238.63ms    477.27ms    715.9ms    954.54ms

frontend HTTP GET /dispatch  
frontend HTTP GET: /customer  
  frontend HTTP GET  
    customer HTTP GET /customer  
mysql SQL SELECT

**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

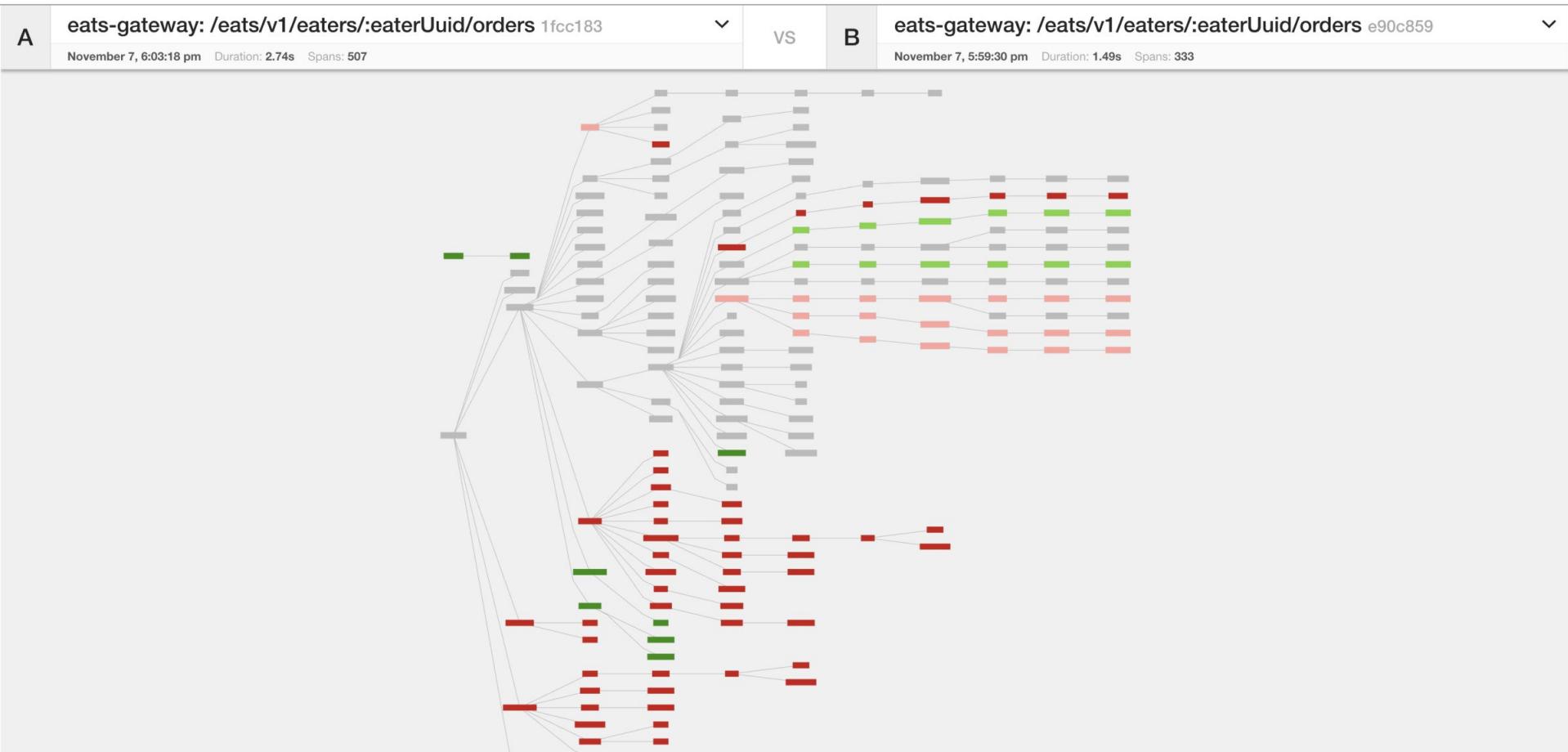
Log timestamps are relative to the start time of the full trace.

SpanID: 7aecd811f9df684

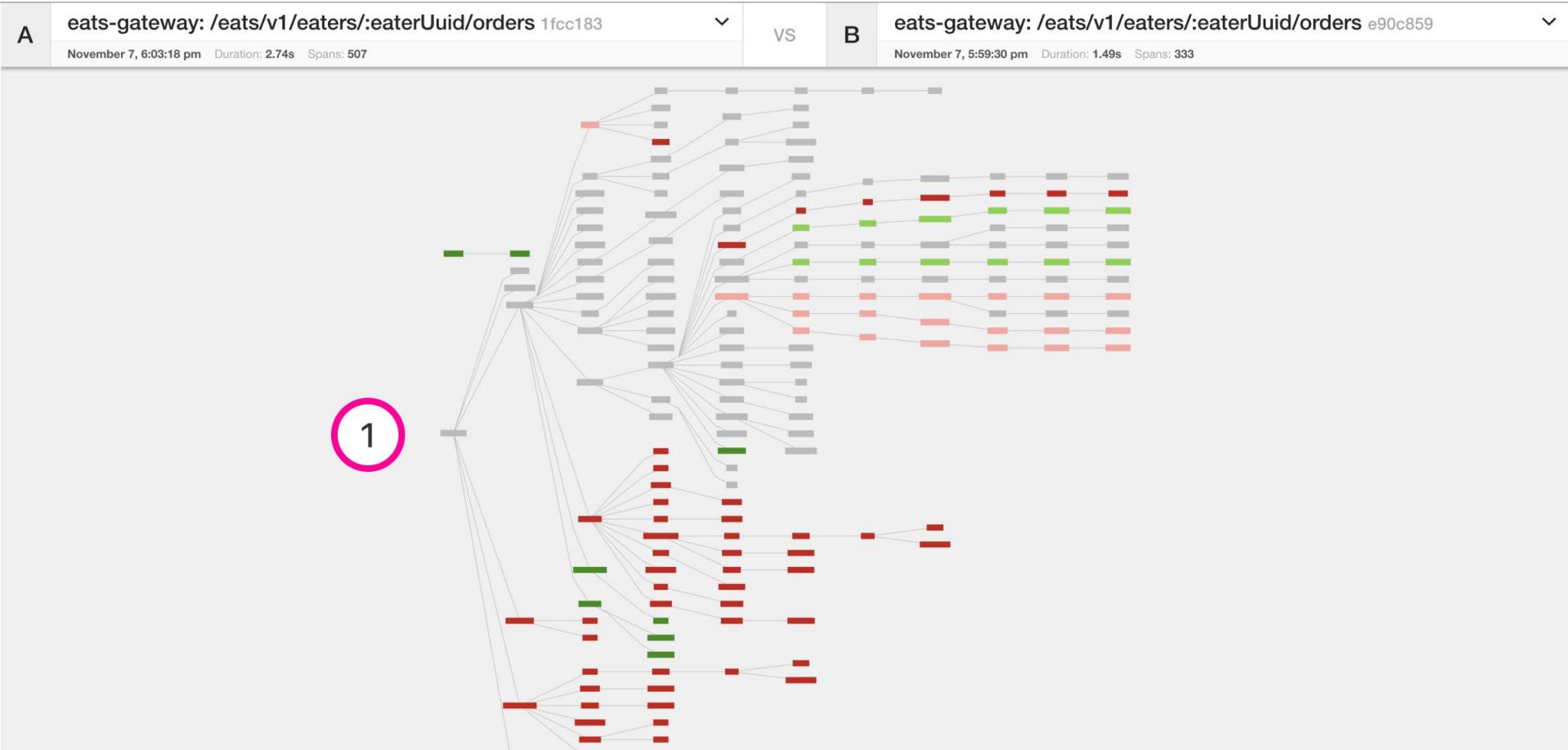
frontend Driver::findNearest  
driver Driver::findNearest

191.41ms  
191.12ms

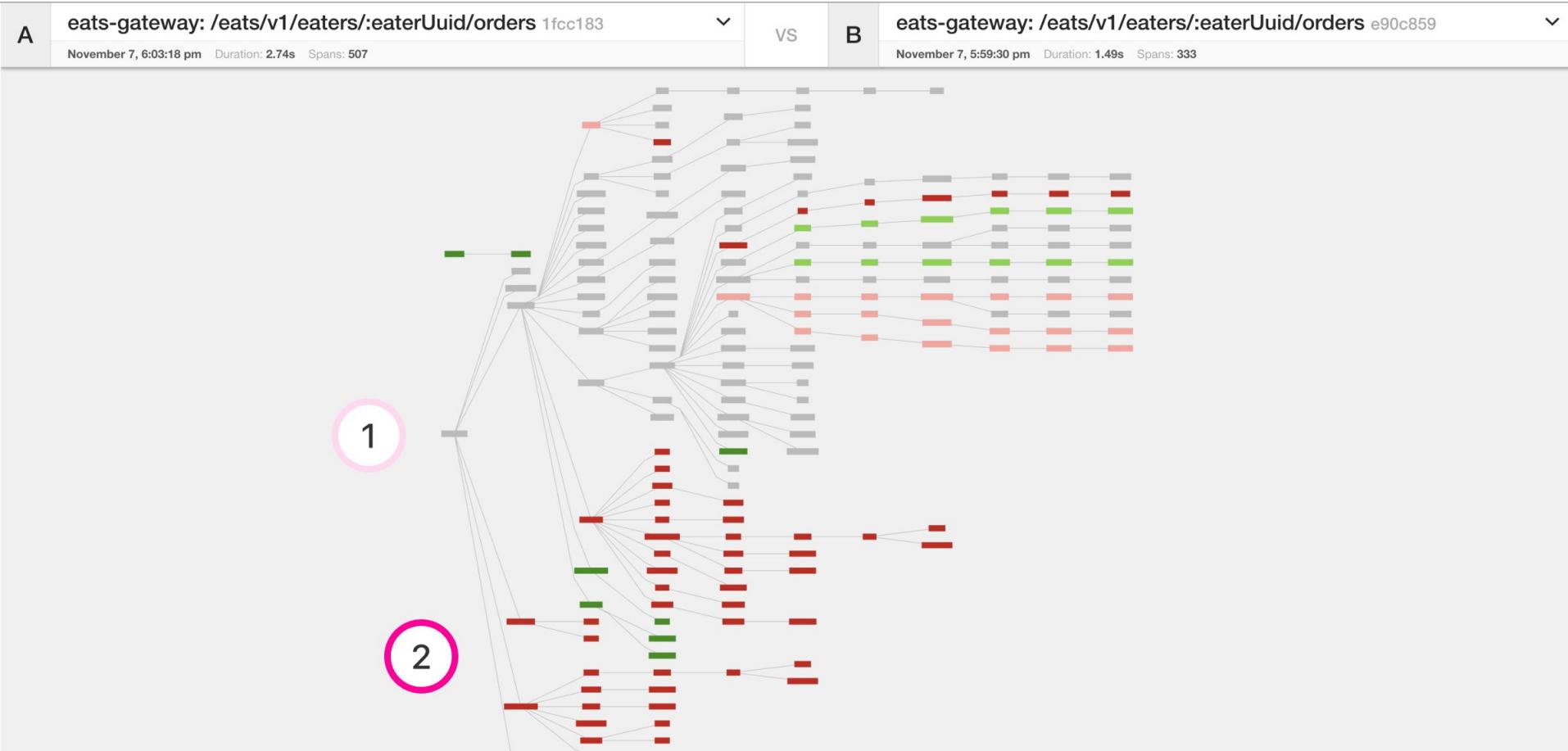
# Comparing trace structures – Unified diff



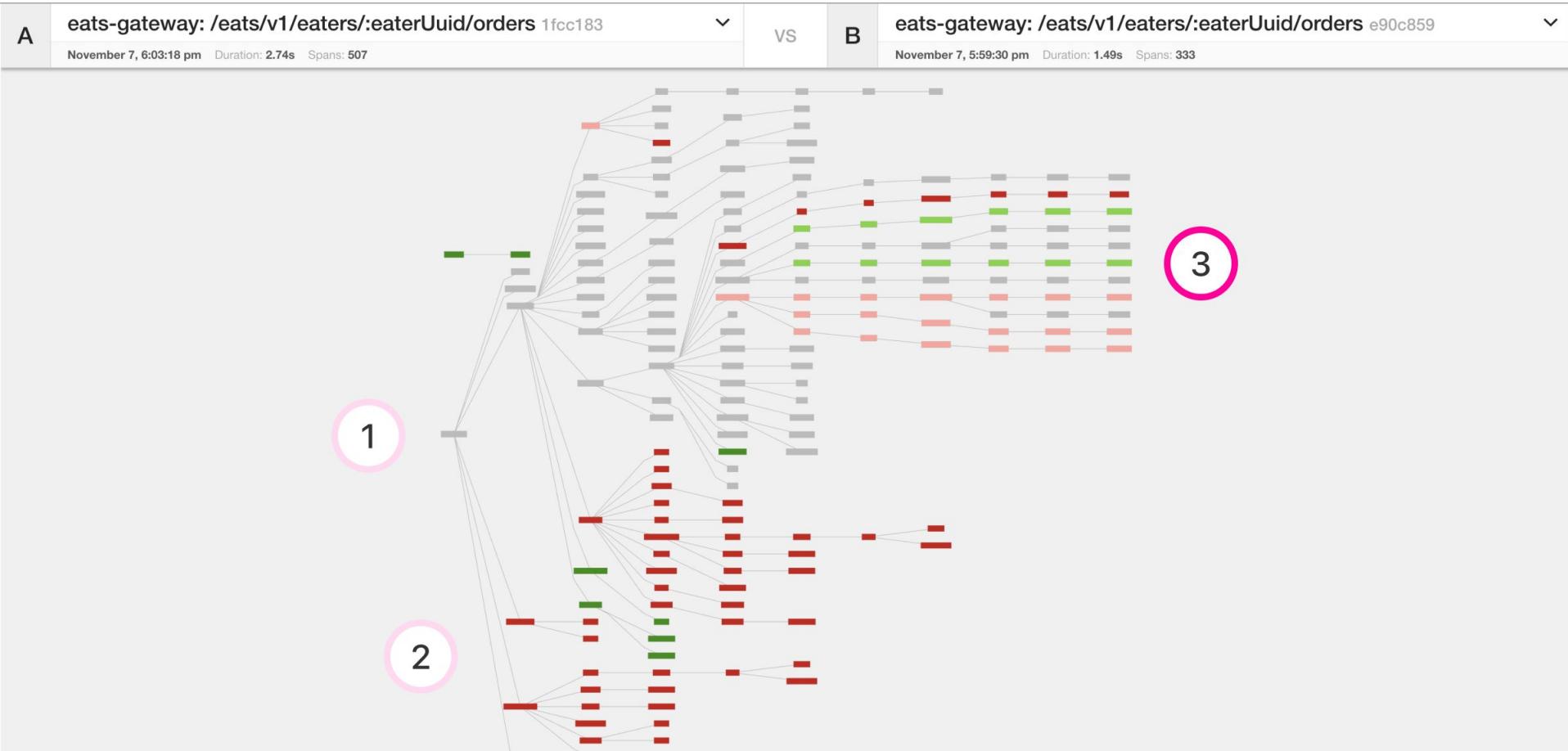
# Comparing trace structures – Shared structure



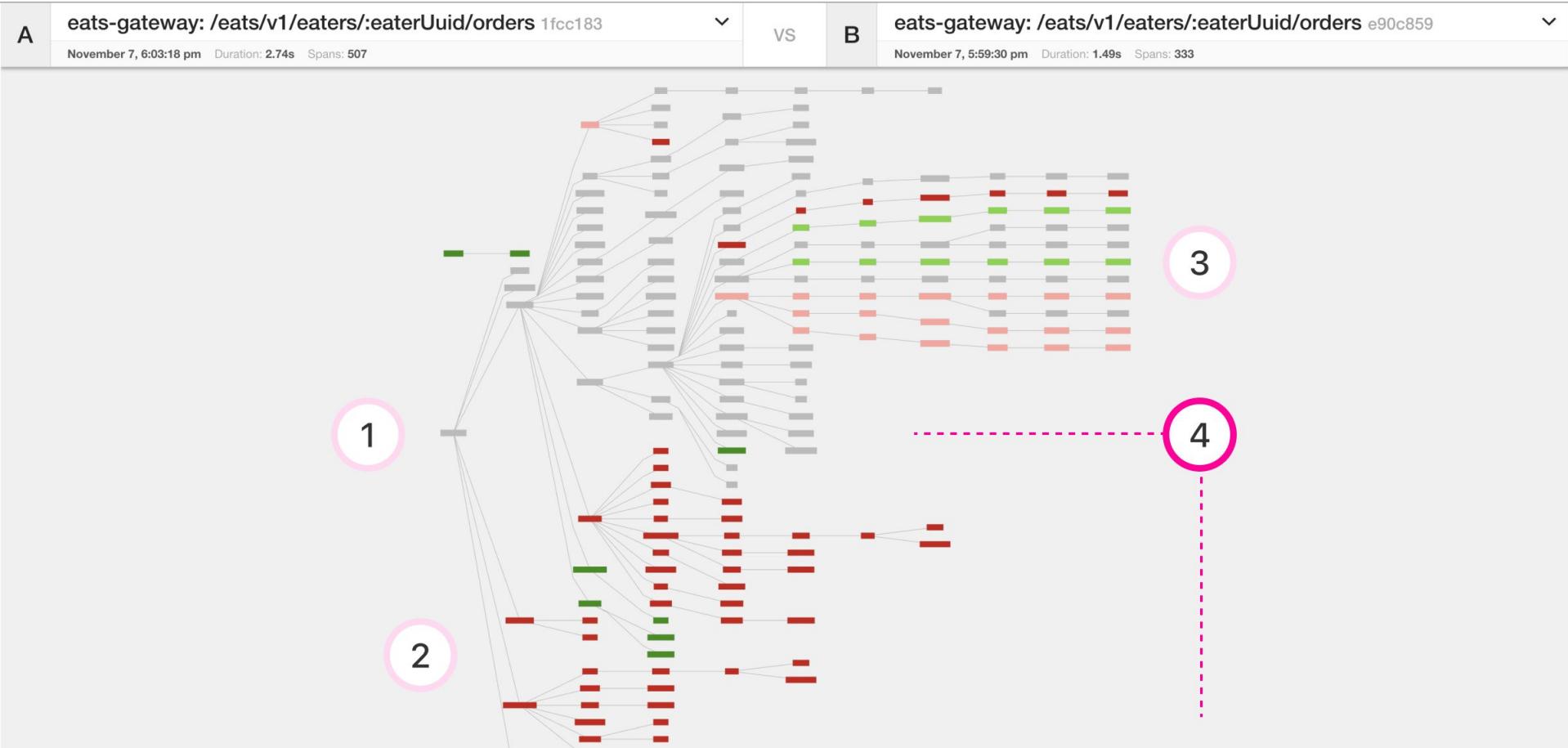
# Comparing trace structures – Absent in one or the traces



# Comparing trace structures – More or less within a node



# Comparing trace structures – Substantial divergence



# "You have an outstanding balance..."

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

Service & Operation    View Options    Archive Trace

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

0ms    371.25ms    742.5ms    1.11s    1.49s

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

> 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

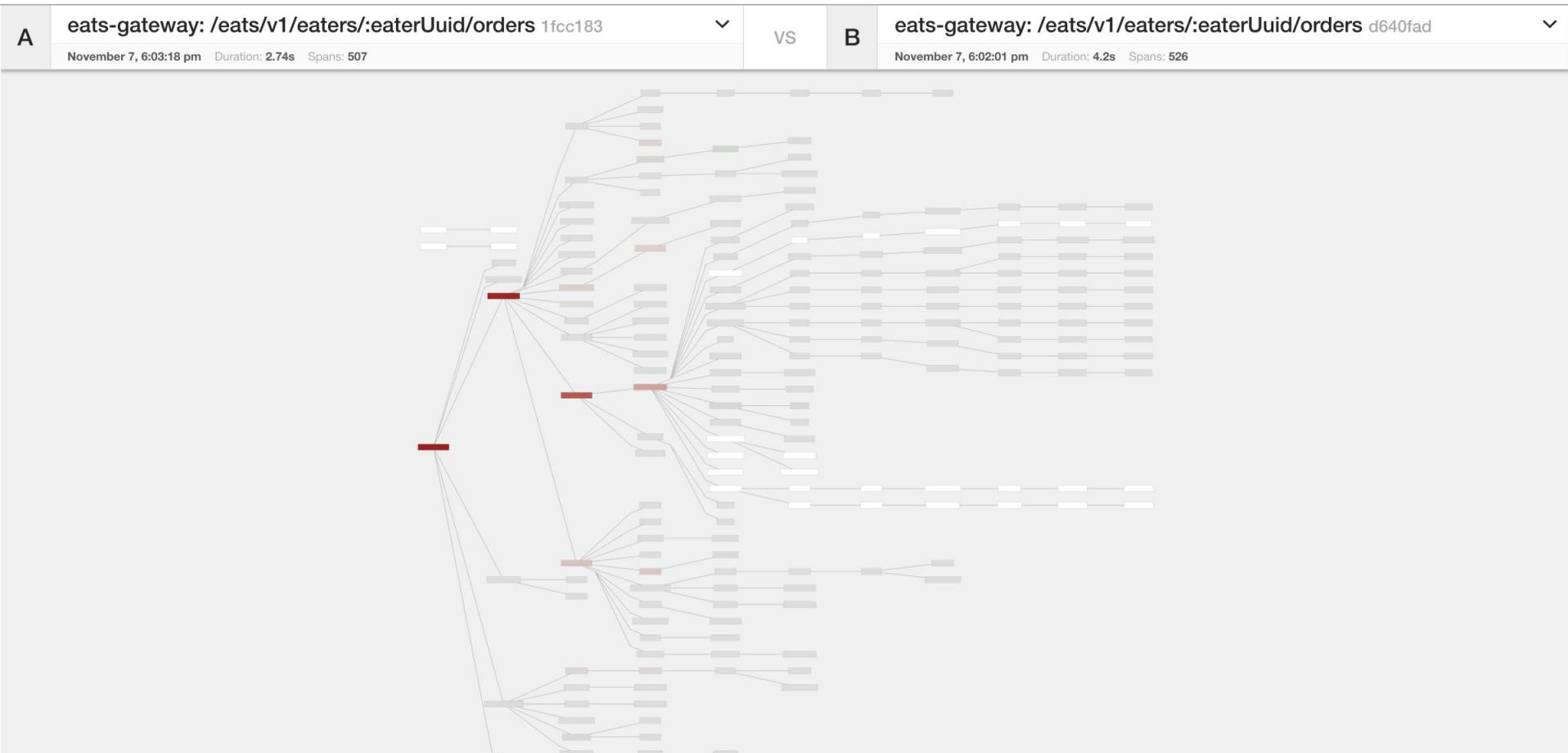
event "error"  
error.kind "TChannelError"  
error.object {  
    info: {  
        message: "Please verify payment information to secure your account",  
        statusCode: 403,  
        shouldRetry: false,  
        stack: "\*errors.errorString You have an outstanding balance due to a credit card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

1

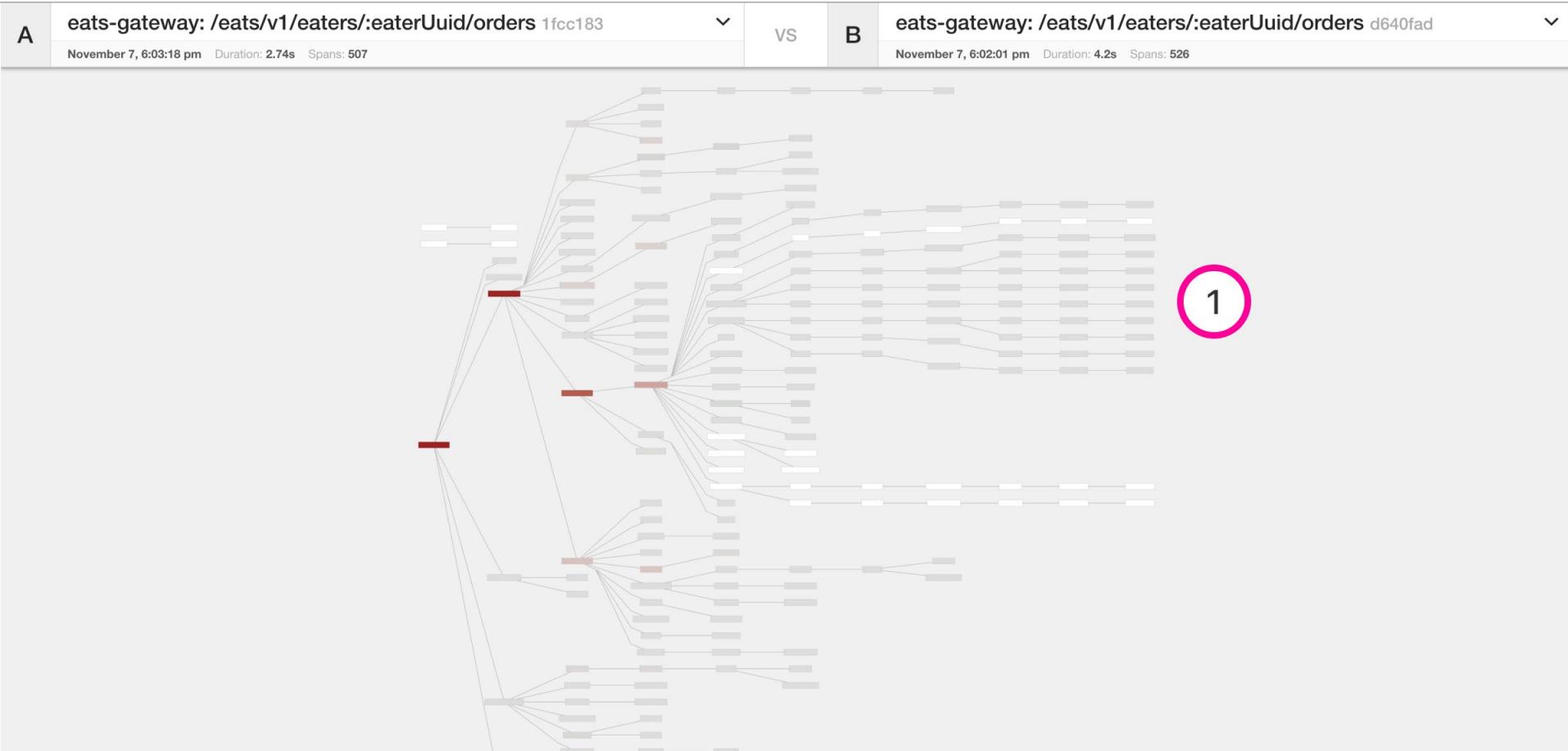
Log timestamps are relative to the start time of the full trace.

SpanID: 63bd06b7a7ed85b4

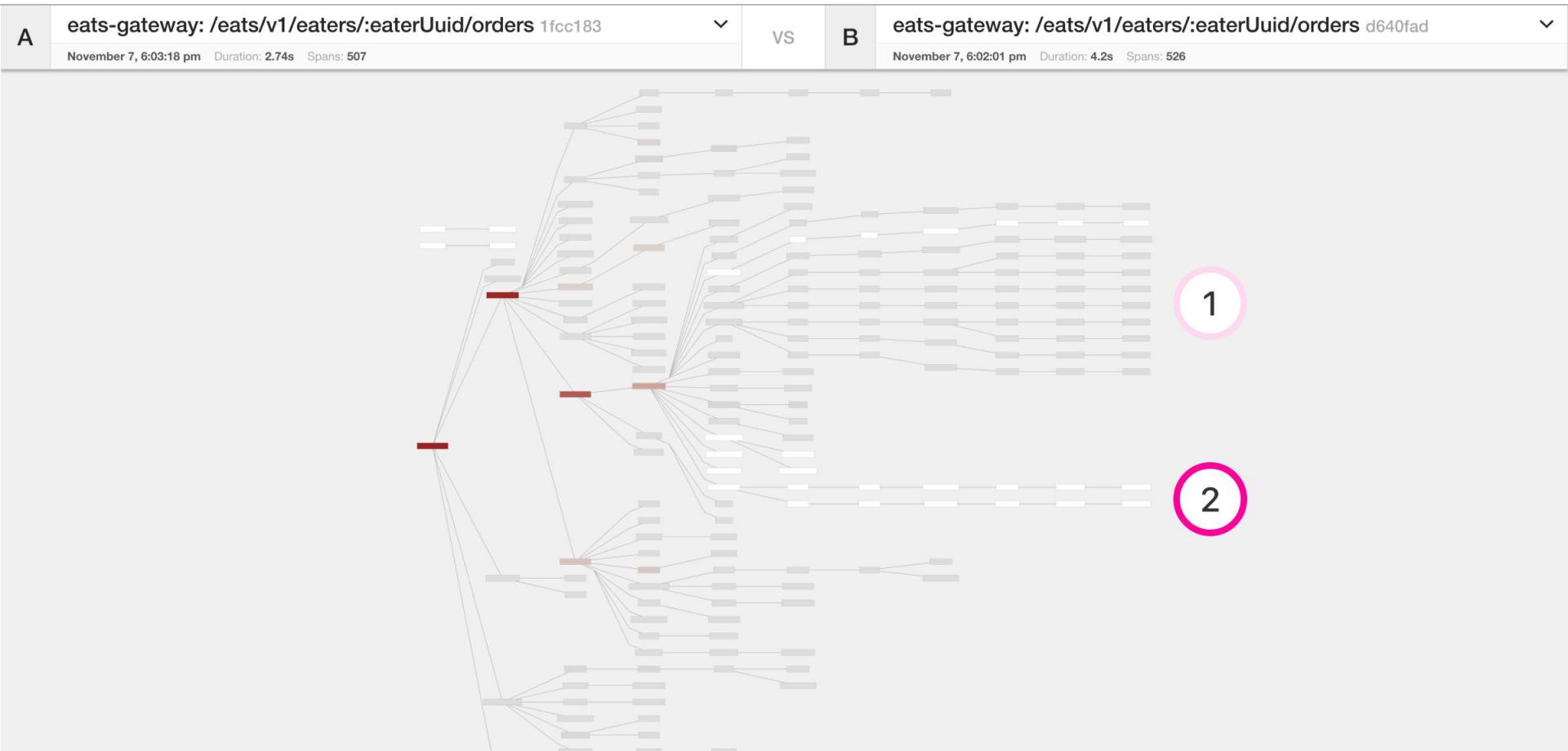
# Comparing span durations



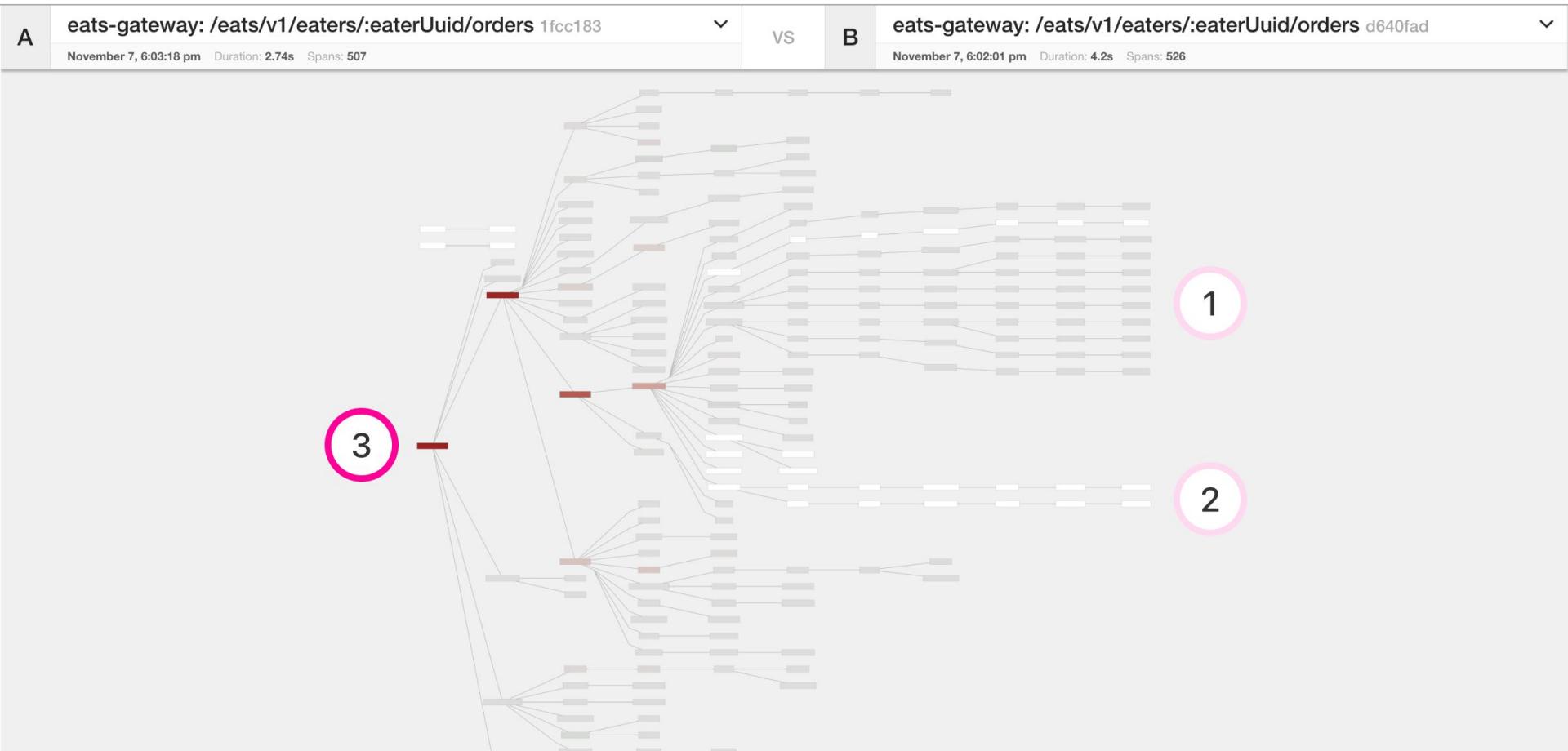
# Comparing span durations – Similar durations



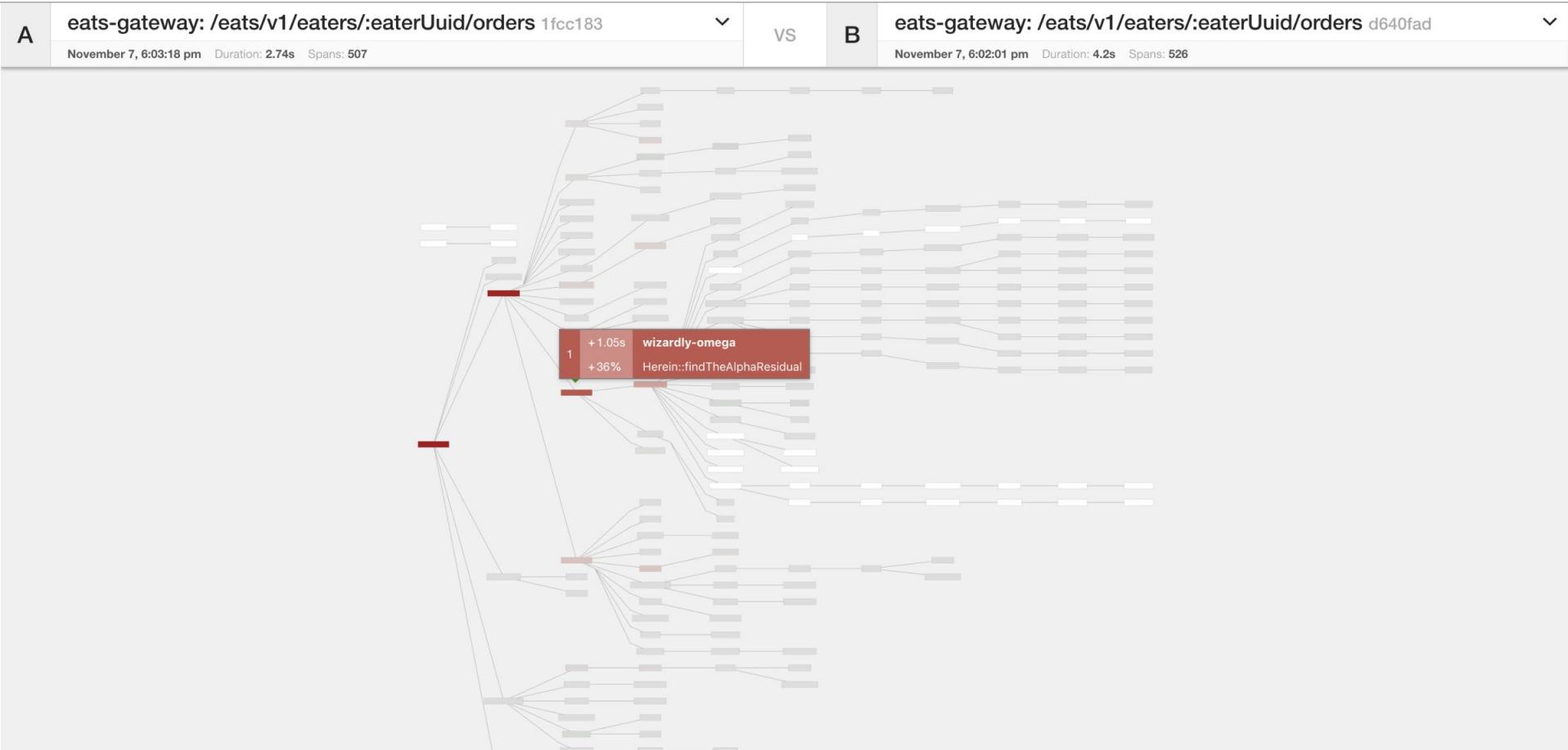
# Comparing span durations – Nodes that aren't shared



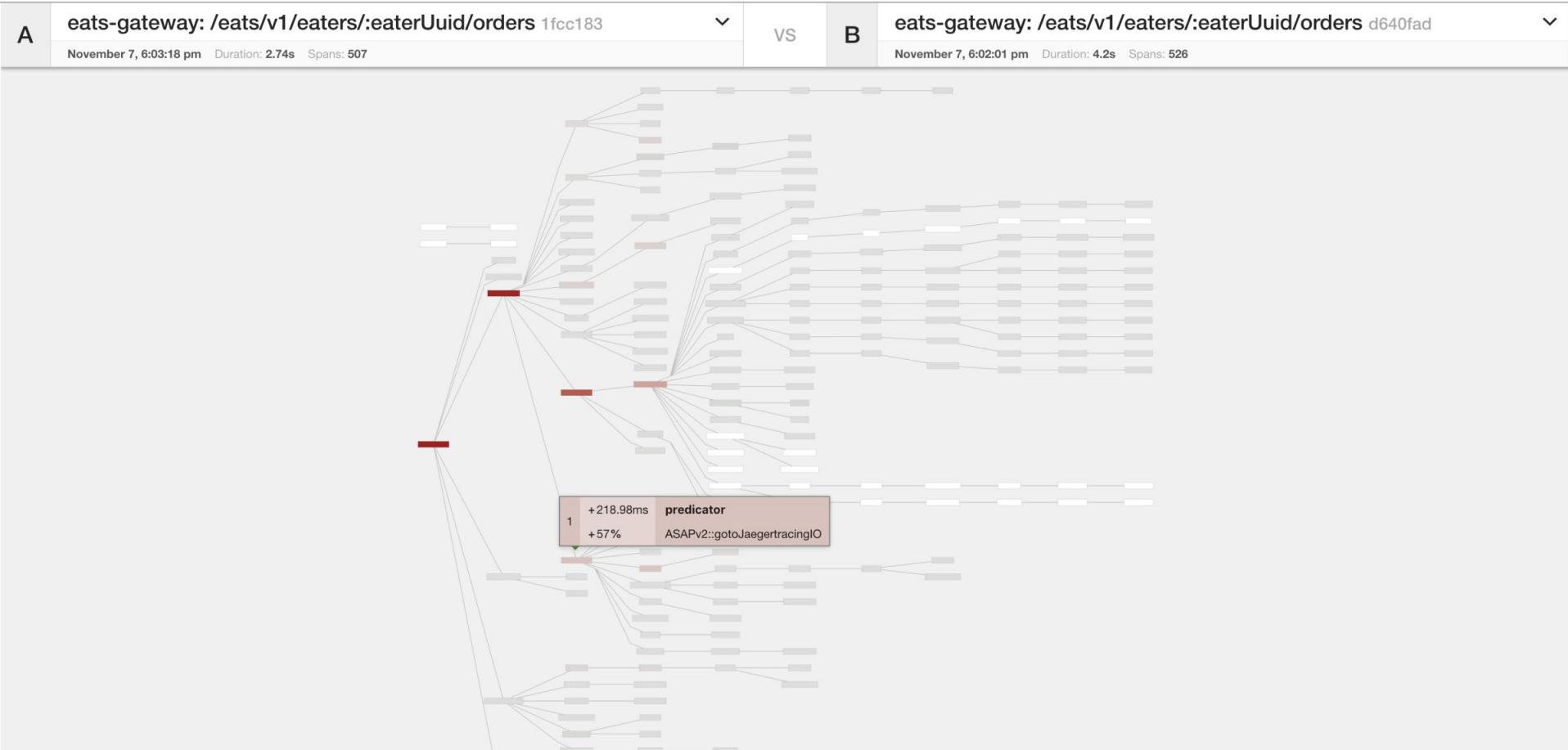
# Comparing span durations – Follow the slower nodes



# Comparing span durations



# Comparing span durations



# Graph Visualizations

Gantt chart is not great for traces with many 100s of spans

- Trace Diffs
  - Compare two traces
  - Compare one trace against a group of traces (coming soon)
- Trace Graph
  - Call graph visualization with mini-aggregations
  - Showing paths rather than individual RPCs

# Graph Visualizations

- Surface less information
- Condense the structural representation
- Emphasize the differences
- Distinct comparison modes simplify the comparisons

# Transitive Service Graphs

4 Traces

Sort: Most Recent

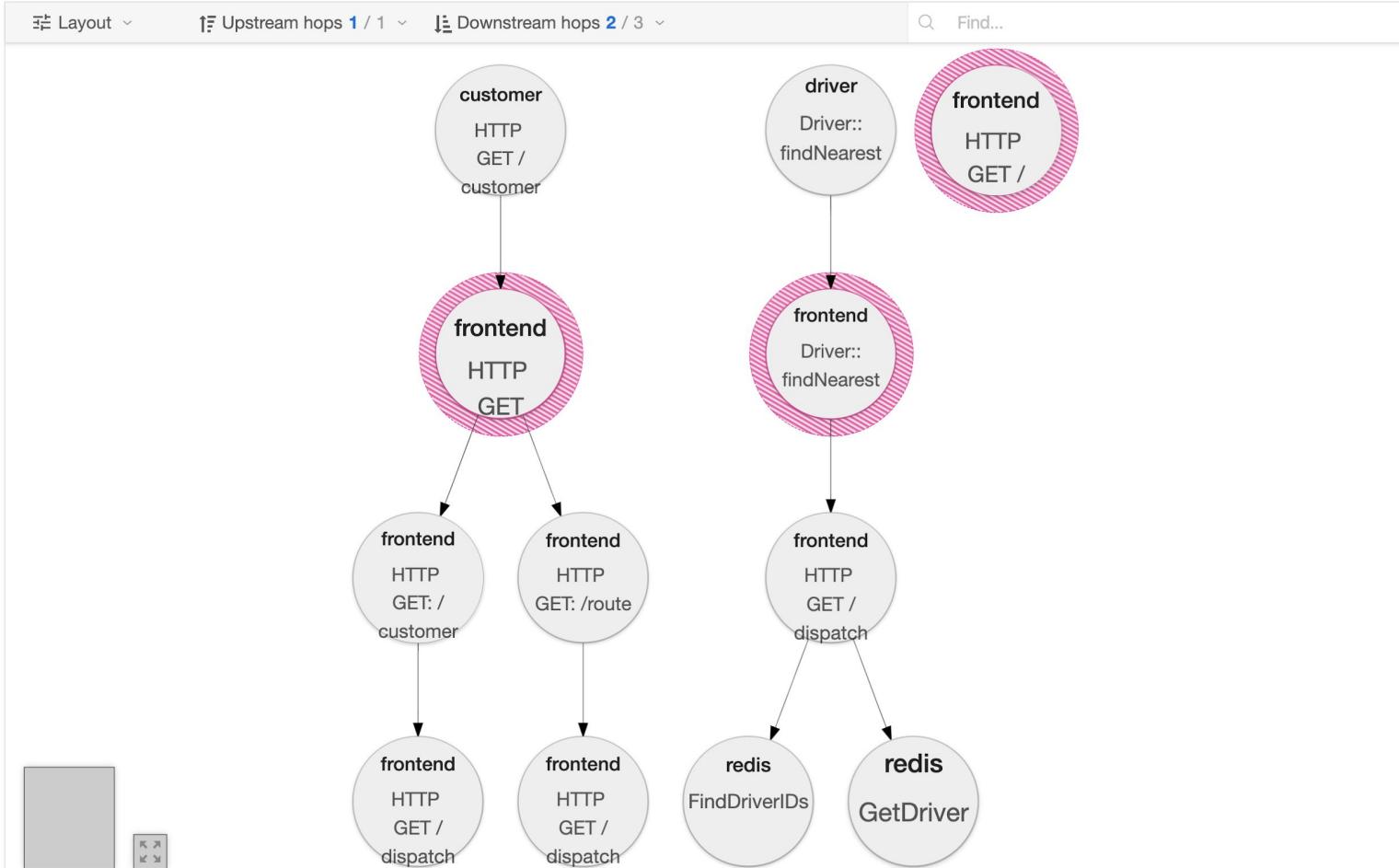
Deep Dependency Graph

Compare traces by selecting result items



<input type="checkbox"/> frontend: HTTP GET /dispatch 3688087	1.04s
51 Spans <span>3 Errors</span> customer (1) driver (1) frontend (24) mysql (1) redis (14) route (10)	Today   5:39:56 pm 5 minutes ago
<input type="checkbox"/> frontend: HTTP GET /dispatch 73e6e77	853.78ms
50 Spans <span>2 Errors</span> customer (1) driver (1) frontend (24) mysql (1) redis (13) route (10)	Today   5:39:56 pm 5 minutes ago
<input type="checkbox"/> frontend: HTTP GET /dispatch d84845f	702.29ms
51 Spans <span>3 Errors</span> customer (1) driver (1) frontend (24) mysql (1) redis (14) route (10)	Today   5:39:56 pm 5 minutes ago

# Transitive Service Graphs



# Distributed Tracing Systems

distributed  
transaction  
monitoring

performance  
and latency  
optimization

root cause  
analysis

service  
dependency  
analysis

distributed context propagation



# Jaeger

# Jaeger, a Distributed Tracing Platform



# Jaeger - /'yāgər/, noun: hunter

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

Oct 31, 2019 ([CNCF announcement](#))



# OpenTracing

- **Instrumentation API**
  - Context propagation
  - Distributed tracing
  - Contextualized logging
  - Contextualized metrics
- Vendor neutral
- Cross language
- CNCF top-level project



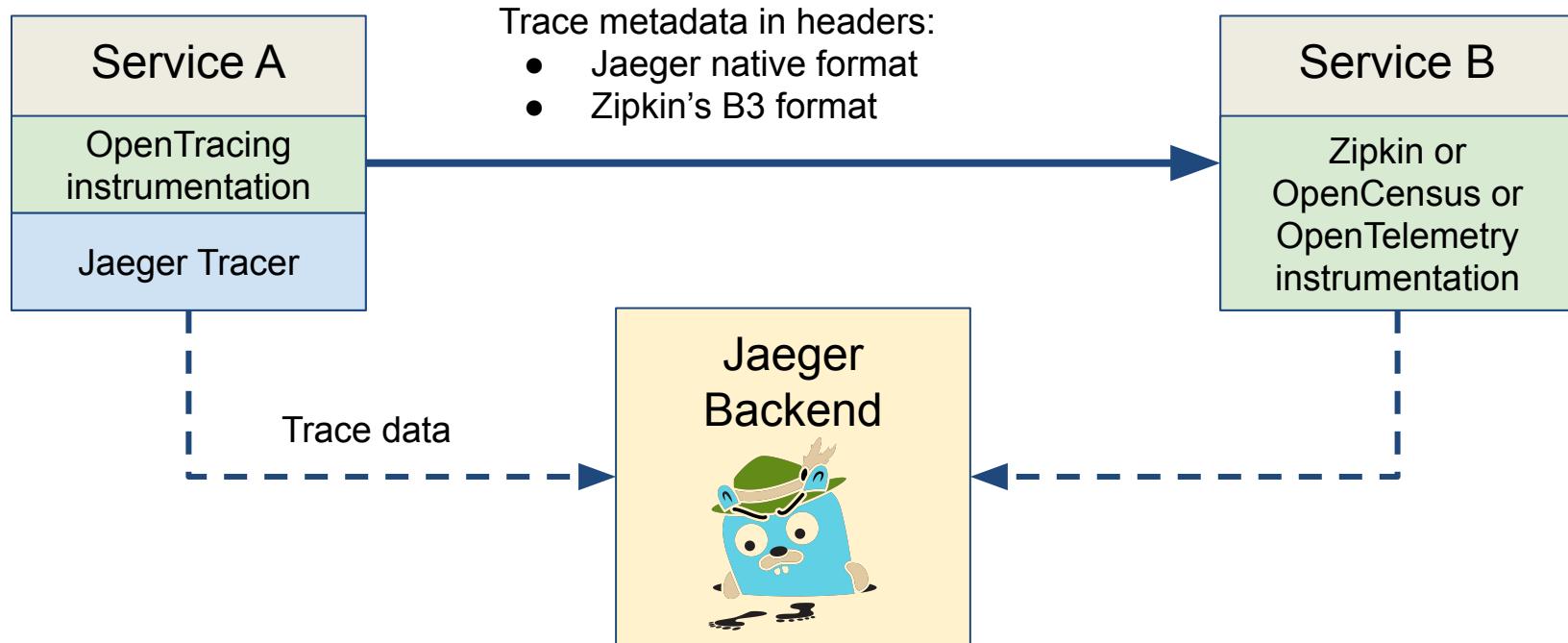
OPENTRACING

<http://opentracing.io>

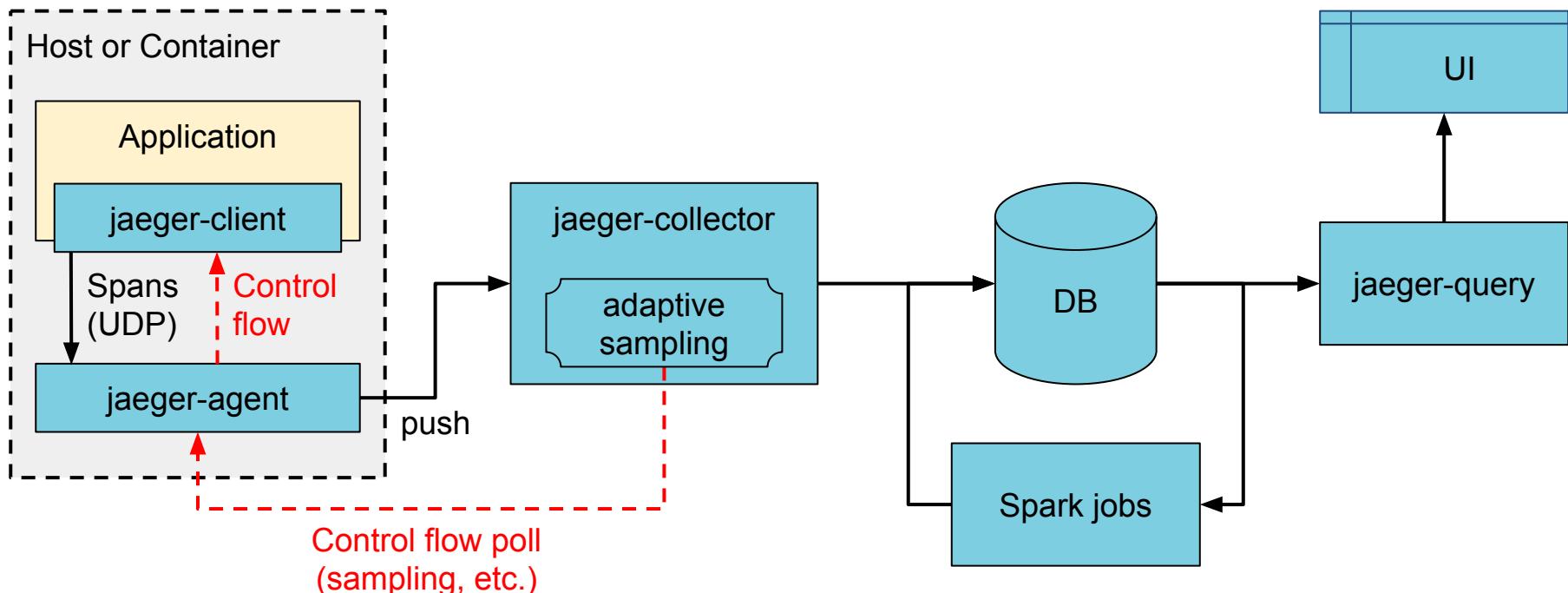


CLOUD NATIVE  
COMPUTING FOUNDATION

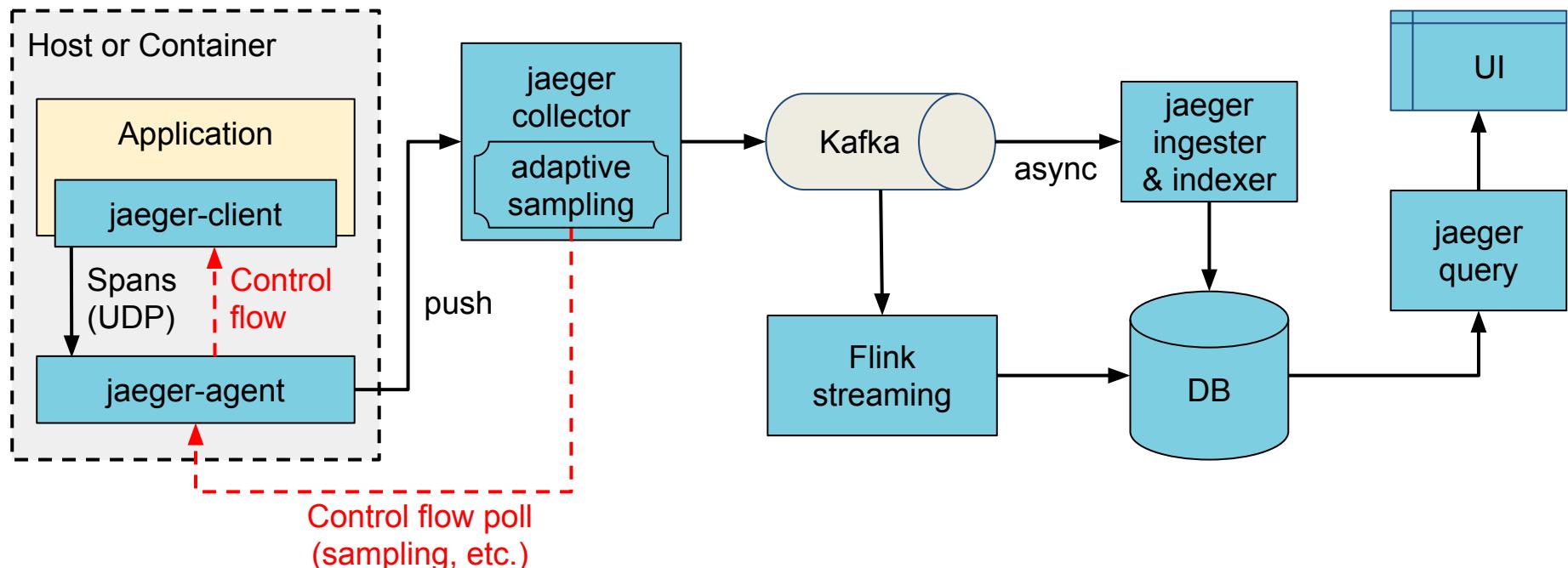
# Jaeger Architecture



# 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



Go



Java™  
POWERED

python  
powered



OPENTRACING

# Project & Community

- **9,000+ GH stars**
- **1200+ contributors**
- **375 authors of commits and pull requests**
- **15 maintainers** across all components from 5+ companies (backend: 7 and 3 respectively)
- **815 Gitter channel members**
- **2,800+ Twitter followers**
- **15 releases since incubation**
- **10M+ Docker pulls**

The screenshot shows the GitHub repository page for `jaegertracing / jaeger`. Key statistics displayed include 858 commits, 6 branches, 27 releases, 1 environment, 109 contributors, and an Apache-2.0 license. A recent commit by `vprithvi` and `yurishkuro` (refactor) pass `saramaMessageWrapper` by value (#1818) was made 7 hours ago. The repository is associated with the CNCF Jaeger, a Distributed Tracing Platform, and has a link to <https://jaegertracing.io/>.

The screenshot shows the GitHub repository page for `jaegertracing / jaeger-collector`. It has 10 stars, 10M+ downloads, and is marked as PUBLIC. The repository was updated 7 hours ago.



# Jaeger 1.15

New Features



# New Features

- Kubernetes Operator
- Badger storage
- Storage plugins: Couchbase, InfluxDB
- Visual trace comparisons
- Security improvements
  - TLS with gRPC, Kafka, Elasticsearch

# Documentation Website

- Releases & Downloads
- Architecture
- Deployment
- Command line options
- Client features

# Integrations

- Jaeger Operator for Kubernetes
  - <https://github.com/jaegertracing/jaeger-operator>
- OpenTelemetry libraries and collector ship with exporters for Jaeger
  - <https://opencensus.io/guides/exporters/supported-exporters/java/jaeger/>
- Istio comes with Jaeger included
  - <https://istio.io/docs/tasks/telemetry/distributed-tracing/>
- Envoy works with Jaeger native C++ client
  - [https://www.envoyproxy.io/docs/envoy/latest/start/sandboxes/jaeger\\_native\\_tracing](https://www.envoyproxy.io/docs/envoy/latest/start/sandboxes/jaeger_native_tracing)
- Eclipse Trace Compass incubator supports importing Jaeger traces
  - <https://github.com/tuxology/tracevizlab/tree/master/labs/303-jaeger-opentracing-traces>

# Asynchronous span ingestion

- Push model was struggling to keep up with traffic spikes
  - Because of sync storage writes
  - Collectors had to drop data randomly
- Kafka is much more elastic for writes
  - Just raw bytes, no schema, no indexing
  - A lot less overhead on the write path
- Data in Kafka allows for streaming data mining & aggregations
- Two new components: **jaeger-ingester** and **jaeger-indexer**

# Protobuf & gRPC

- Internal data model generated from Protobuf IDL
- gRPC connection between `jaeger-agent` and `jaeger-collector`

Why

- gRPC plays better with modern routing than TChannel
- Path to official data model and collector/query APIs
- Protobuf-based JSON API
- Unblock development of storage plugins
- (Thrift still supported for backwards compatibility)

# Zipkin Compatibility

- Clients
  - Zipkin B3-\*\*\* headers for context propagation
  - Interop between Jaeger-instrumented and Zipkin-instrumented apps
- Collector
  - Zipkin Thrift, Protobuf, and JSON v2 span format
  - Use Zipkin instrumentation (e.g. Brave) to send traces to Jaeger
- Kafka



**CLOUD NATIVE**  
COMPUTING FOUNDATION

## Roadmap

<http://bit.do/jaeger-roadmap>



# Roadmap

- Trace DSL, jupyter notebooks and where we are heading
- Delayed & ad-hoc sampling
- Tail-based sampling
- OpenTelemetry

# Adaptive Sampling

## Problem

- APIs have endpoints with different QPS
- Service owners do not know the full impact of sampling probability

Adaptive Sampling is per service + endpoint,  
decided by Jaeger backend based on traffic

# Adaptive Sampling Status

- Jaeger clients support per service/endpoint sampling strategies
- Can be statically configured in collector
- Pull requests for dynamic recalculations

# Data Pipeline

- Based on Kafka and Apache Flink
- Support aggregations and data mining
- Examples:
  - Pairwise service graph (dependencies diagram)
  - Path-based service graphs
  - Latency histograms



# 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

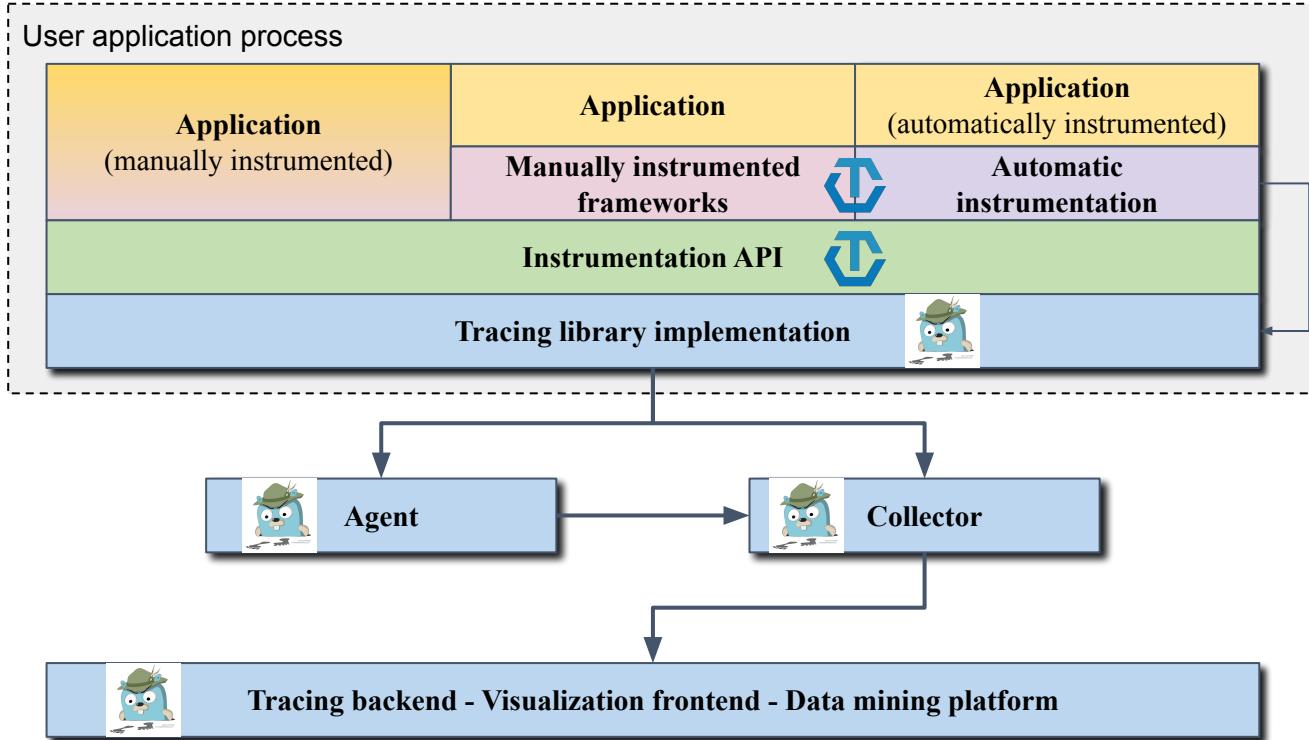
# Q&A

- Jaeger Deep Dive - Wed, November 20, 2:25pm

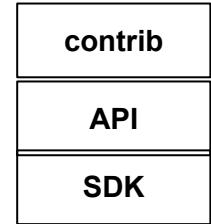
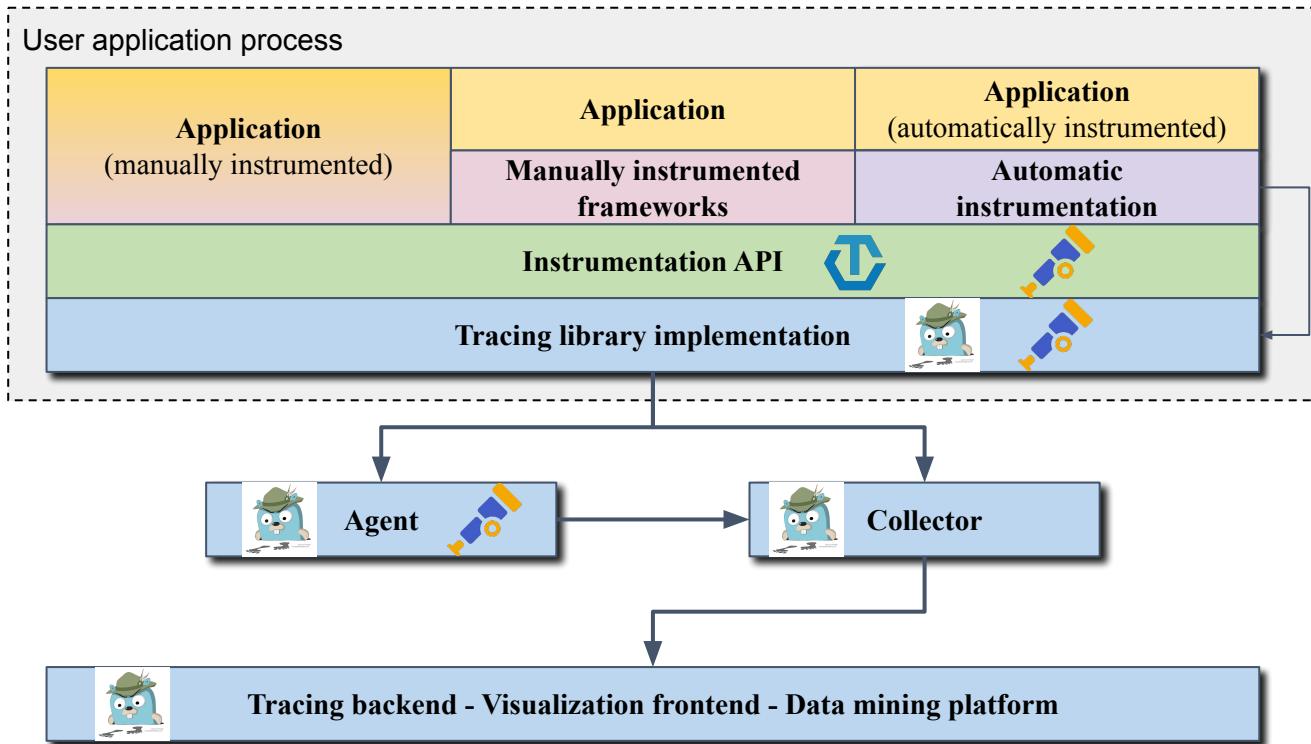


<https://jaegertracing.io>

# Jaeger vs. OpenTracing



# Jaeger vs. OpenTracing, OpenCensus, OpenTelemetry





**CLOUD NATIVE**  
COMPUTING FOUNDATION

## Learn More

Website: [jaegertracing.io/](https://jaegertracing.io/)

Blog: [medium.com/jaegertracing](https://medium.com/jaegertracing)