Combining OpenTracing and Kernel Tracing for Performance Analysis of Distributed Applications

Progress Report Meeting - May 6, 2019

DORSAL

Ecole polytechnique de Montréal

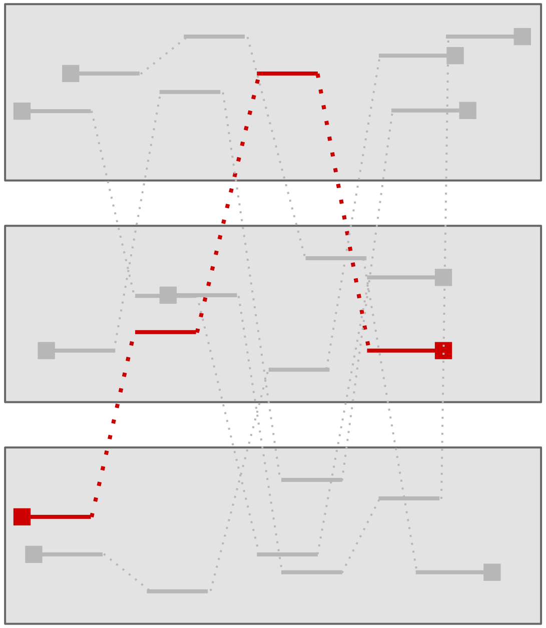
Loïc Gelle

Michel Dagenais

Context

OpenTracing: where does it help, where does it fail?

What distributed tracing is all about



- Single user request, multiple machines
- We want to tell the full story of a given request

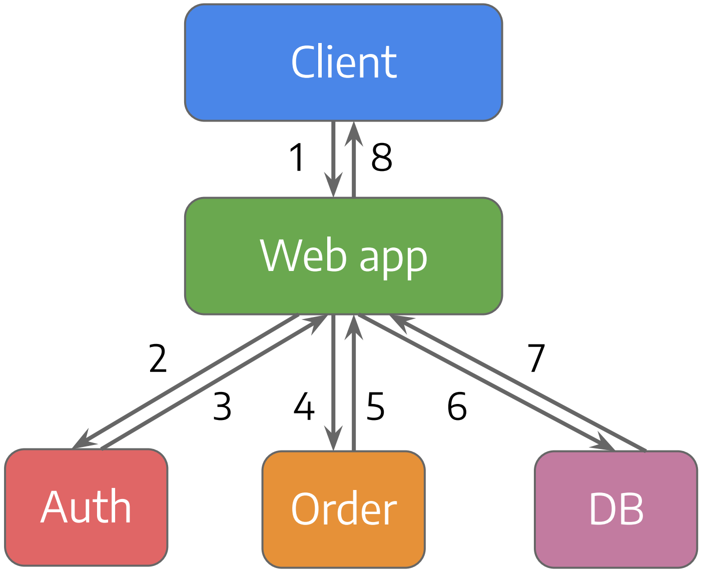
Illustration courtesy of Ben Sigelman

Key facts about OpenTracing

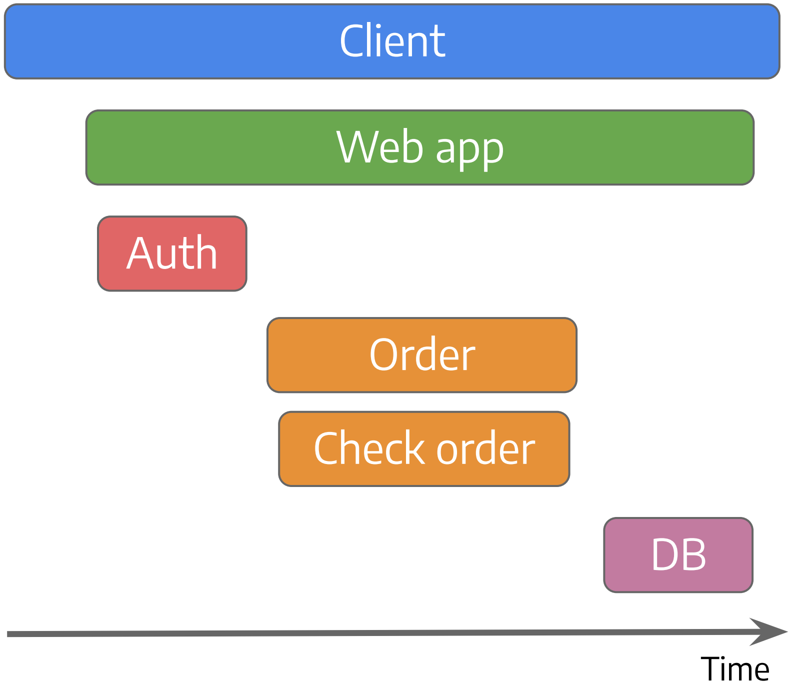
- An open-source **specification for distributed tracing**
- A **vendor-neutral API** for instrumenting libraries
  - API available for **popular languages** like Java, Go, C++, Python…
  - Lots of **libraries** like gRPC, NodeJS… are instrumented
- Many tracers (Jaeger, OpenZipkin, LightStep…) implement the OpenTracing specification
  - OpenTracing **leaves implementation details** to the tracers
  - Each tracer has **different purposes and analyses / UI**

Describing complex transactions

OpenTracing focuses on describing **tasks** instead of events.



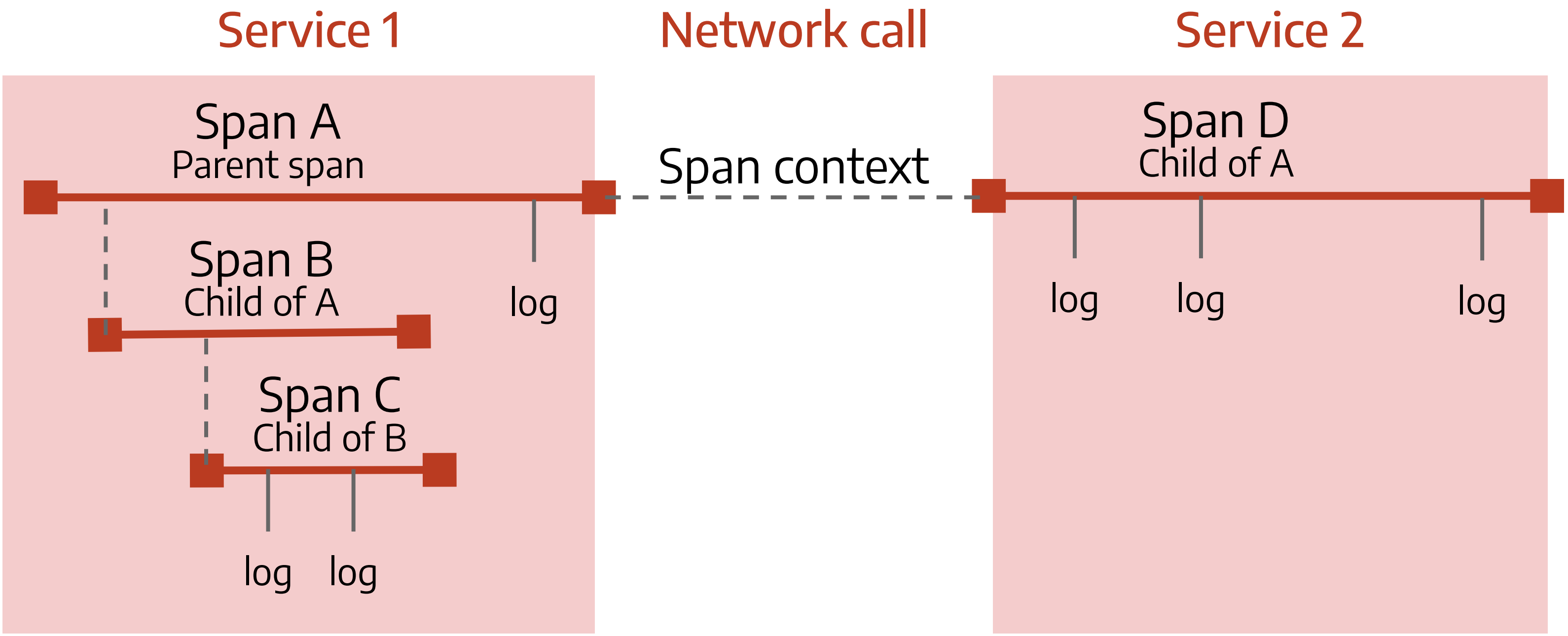
What the transaction looks like



What the trace looks like

Adapted from Ben Sigelman

Key concepts in OpenTracing



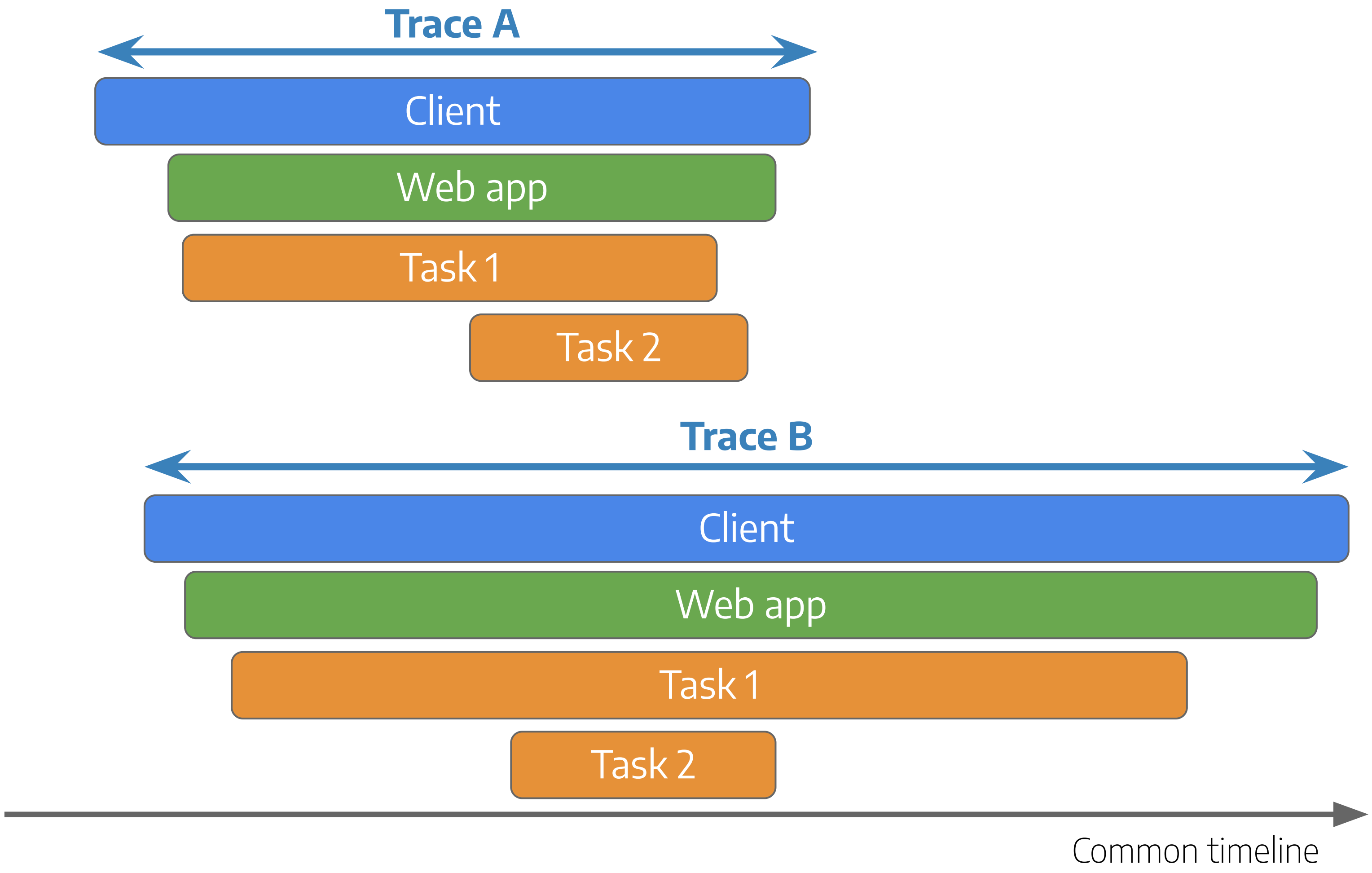
- A **span** has a name, a start, a duration, tags and attached logs.
- The **span context** identifies the trace; it is injected into requests.
- A **trace** is the recording of the whole transaction using the above!

Adapted from Ben Sigelman

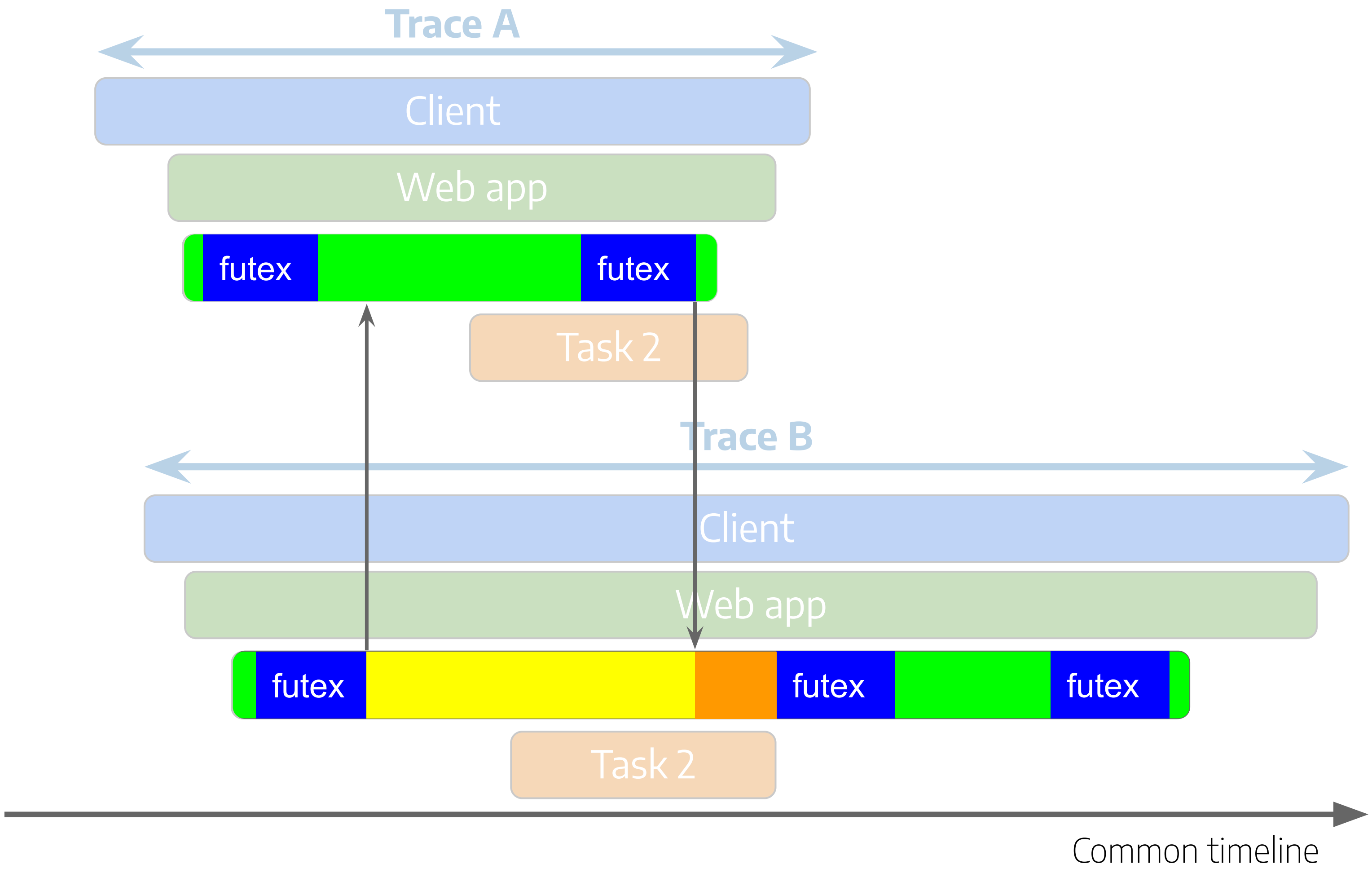
The benefits of OpenTracing

- The community is growing
- The traces provide useful **high-level context** for debugging applications
- The tracers provide the machinery to **collect the traces and display them**
- Use and deployment are fairly easy

Where does OpenTracing fail?



Same events, different perspective

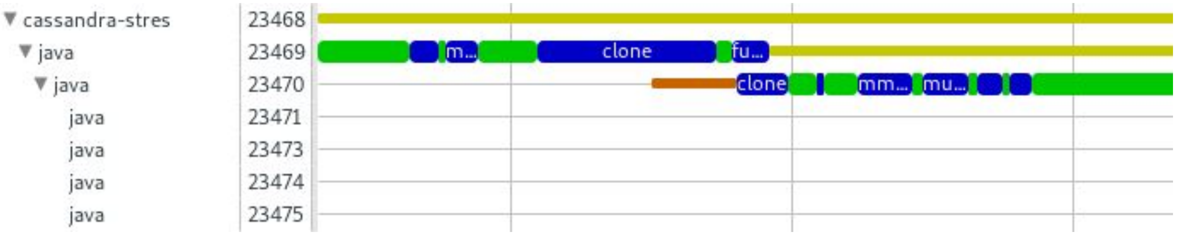


The approach

Combining OpenTracing and kernel traces

Bridging the gap

- On the one side: threads, nanosecond-precise events



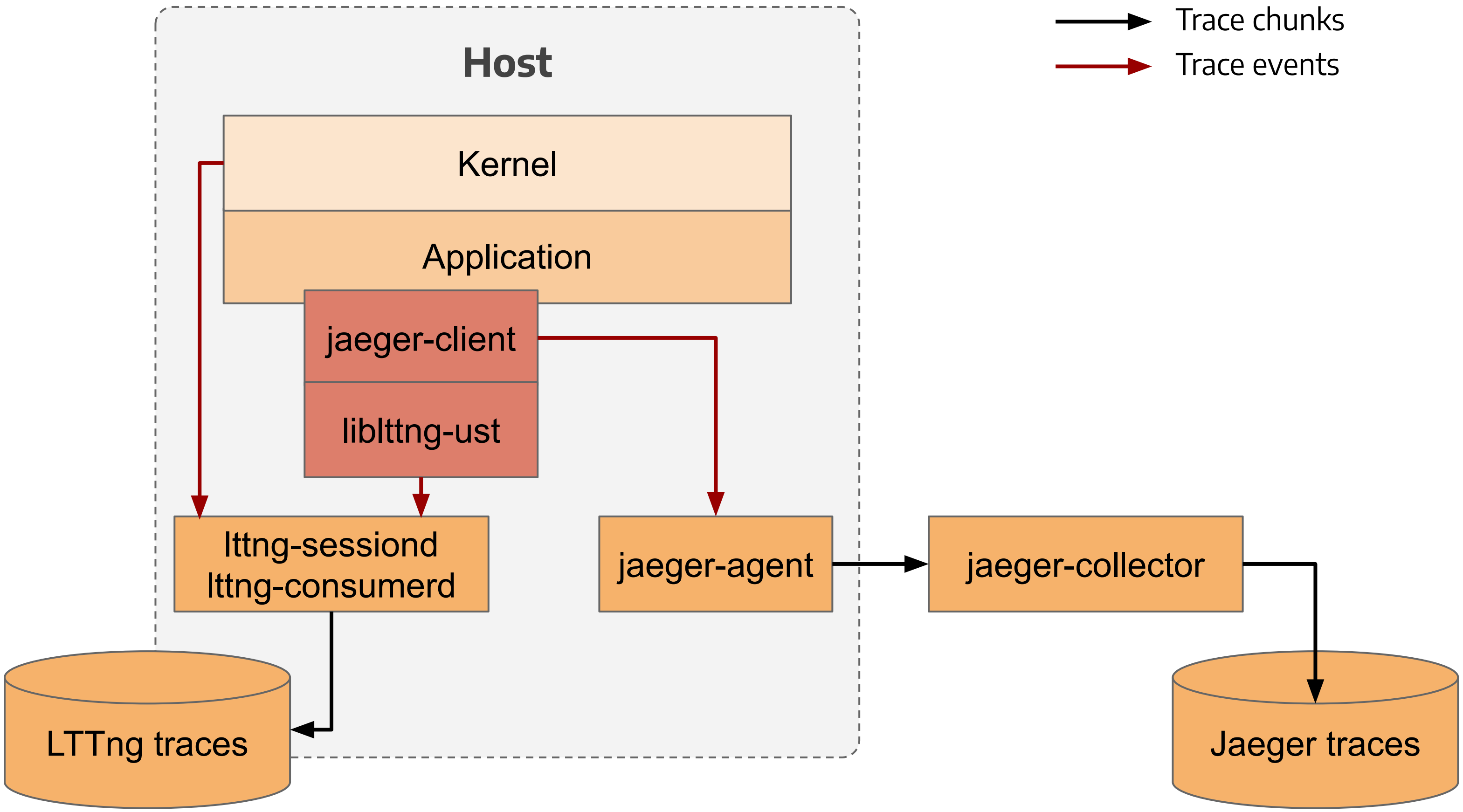
▼ cassandra-stres	23468
▼ java	23469
▼ java	23470
java	23471
java	23473
java	23474
java	23475

- On the other side: tasks, microsecond-precise events
    - We need to synchronize events
    - We need to relate tasks back to their thread(s)

Techniques for synchronization

- “Fake syscall” *(Google)*
- Kernel module + added LTTng kernel context *(Boston University)*
- Instrumentation of the OpenTracing tracer using LTTng-UST *(what we use)*

Collection of traces



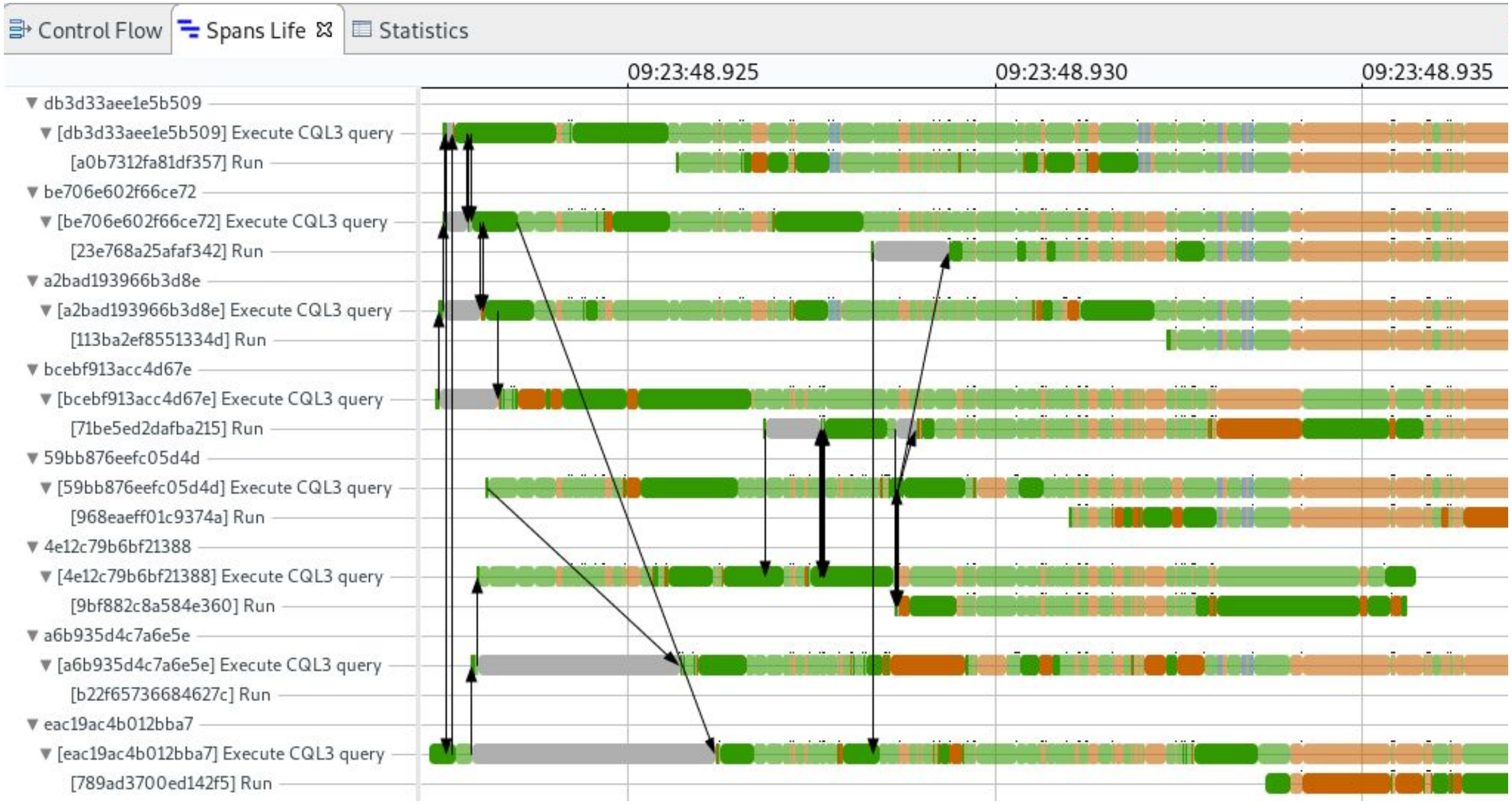
Analyses

TraceCompass views

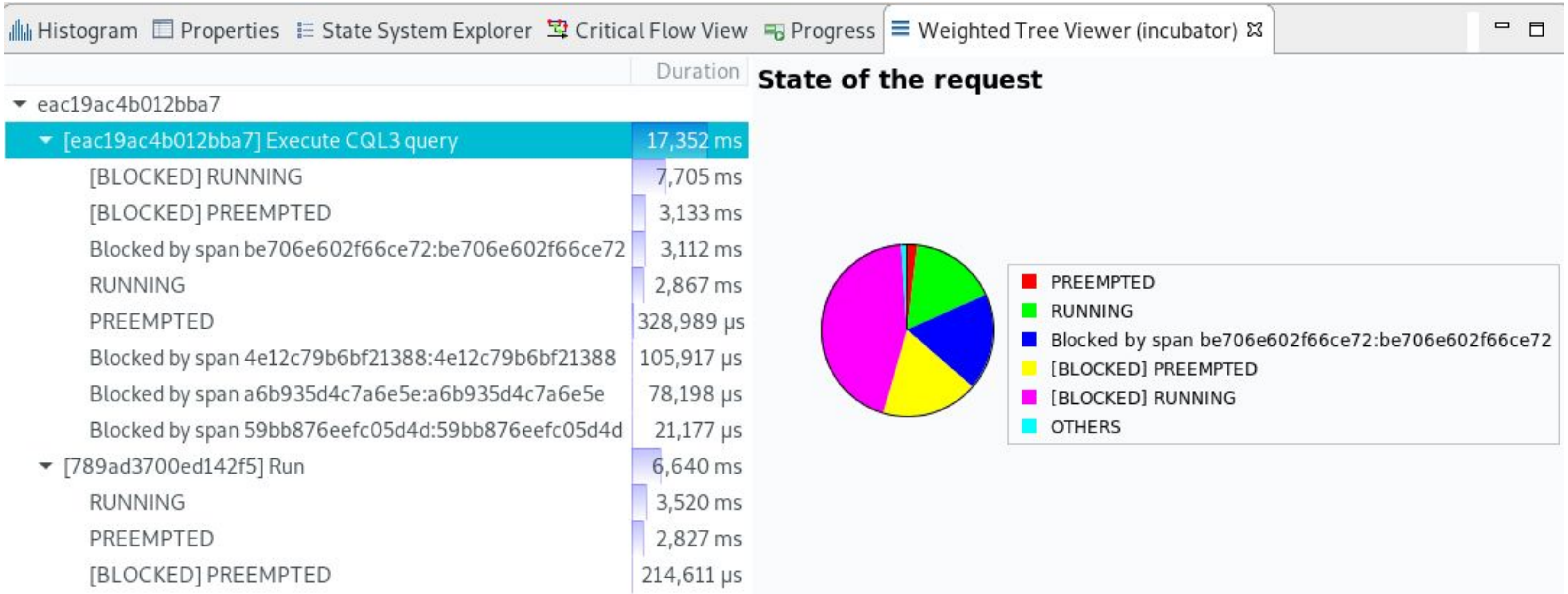
Proof of concept in TraceCompass

- Two views to validate the approach
    - Critical path of requests
    - Aggregated view per request of the critical path
- Based on prior work from Ericsson
- The instrumented application is Cassandra

Critical path of requests



Aggregated information



	Duration
eac19ac4b012bba7	
[eac19ac4b012bba7] Execute CQL3 query	17,352 ms
[BLOCKED] RUNNING	7,705 ms
[BLOCKED] PREEMPTED	3,133 ms
Blocked by span be706e602f66ce72:be706e602f66ce72	3,112 ms
RUNNING	2,867 ms
PREEMPTED	328,989 µs
Blocked by span 4e12c79b6bf21388:4e12c79b6bf21388	105,917 µs
Blocked by span a6b935d4c7a6e5e:a6b935d4c7a6e5e	78,198 µs
Blocked by span 59bb876eefc05d4d:59bb876eefc05d4d	21,177 µs
[789ad3700ed142f5] Run	6,640 ms
RUNNING	3,520 ms
PREEMPTED	2,827 ms
[BLOCKED] PREEMPTED	214,611 µs

State of the request

- 🟥 PREEMPTED
- 🟩 RUNNING
- 🟦 Blocked by span be706e602f66ce72:be706e602f66ce72
- 🟨 [BLOCKED] PREEMPTED
- 🟪 [BLOCKED] RUNNING
- 🟦 OTHERS

Conclusions and future work

Limitations and remarks

- We need developers to provide a good instrumentation of their application
- Analyses limited to a single machine
- The volume of the traces can be tough to handle and sampling is not straightforward
- Benchmarking has yet to be done

Future work

- Adapt the trace collection / analysis to applications hosted in containers
- Bring the analyses to UIs widely used by the OpenTracing community (Jaeger, Kibana)
- Work with the community to integrate the changes to the OpenTracing tracers

Thank you!

Questions, ideas, remarks?

✉ loic.gelle@polymtl.ca

⚙ Github: @loicgelle