Instrumentation with OpenTelemetry and GAE

What is code instrumentation?

- Instrumentation refers to the measure of product performance, to diagnose errors, and to write trace information - https://en.wikipedia.org/wiki/Instrumentation_(computer_programming)

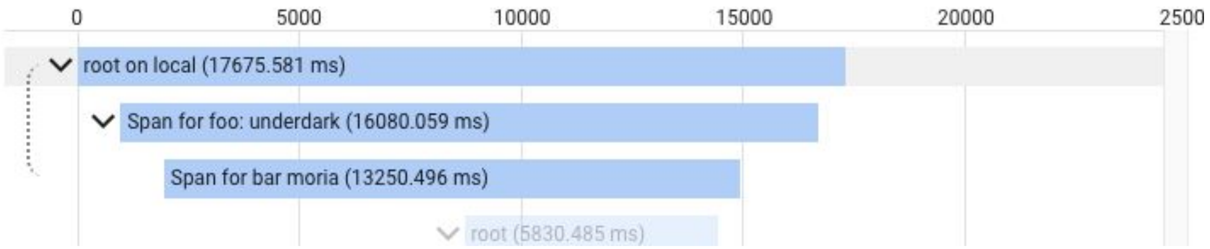
- Instrumention technology is designed to reduce both performance and memory ovehead to a minumum - https://www.ibm.com/support/knowledgecenter/SSSHUF_8.0.0/com.ibm.rational.testrt.doc/topics/cinstruovw.html

You did this before!

- print(“I am here”)
- print(“WTF?”)
- print (“This should not execute”)

What is tracing?

- Specialized use of logging to record information about program’s execution - https://en.wikipedia.org/wiki/Tracing_(software)#:~:text=In%20software%20engineering%2C%20tracing%20involves,information%20about%20a%20program's%20execution.&text=Logs%20that%20record%20program%20usage,into%20a%20terminological%20gray%20area.


0
5000
10000
15000
20000
2500
root on local (17675.581 ms)
Span for foo: underdark (16080.059 ms)
Span for bar moria (13250.496 ms)
root (5830.485 ms)

Part of larger concept - Observability.

- We need to answer questions about our systems.

What characteristics did the queries that timed out at 500ms share in common? Service versions? Browser plugins?

- Instrumentation produces data.
- Querying data answers our questions.

Source: https://opentelemetry.io/docs/workshop/resources/

What is OpenTelemetry

Project that provides a set of APIs, libraries in various languages, for capturing traces, and exporting them to other services for further analysis.

- https://opentelemetry.io/
- https://github.com/open-telemetry
- https://github.com/open-telemetry/opentelemetry-python

Currently in Beta.

Trace concepts

- Span
  - Represents a single unit of work in a system.
  - Typically encapsulates: operation name, a start and finish timestamp, the parent span identifier, the span identifier, and context items.
- Trace
  - Defined implicitly by its spans. A trace can be thought of as a directed acyclic graph of spans where the edges between spans are defined as parent/child relationships.
- DistributedContext
  - Contains the tracing identifiers, tags, and options that are propagated from parent to child spans

Source: https://opentelemetry.io/docs/workshop/resources/

Console tracing example

How to get the best results - visualize it!

- Possible places to export your traces
    - Honeycomb.io https://www.honeycomb.io/
    - CloudTrace https://cloud.google.com/trace
    - Jaeger https://www.jaegertracing.io/
    - Prometheus https://prometheus.io/

Cloud Trace

Cloud trace is a distributed tracing system that collects data from various different sources and presents them in a fairly easy to understand way :)

Cloud trace requirements

- A Google Cloud project. Otherwise the traces have nowhere to go.
- Enable Cloud Trace API. Go to the list of APIs available, it’s possible that it’s already activated
- Connect your application to the project. This is done by either
  - Setting the GOOGLE_APPLICATION_CREDENTIALS to point to the service account file
  - Installing the gcloud sdk, and authenticating yourself by calling gcloud auth application-default login
  - Taken care of by Google if you are running your project on GAE (which is what we will do), or GCE

GAE Cloud trace examples.

Tip of the iceberg