Jaeger Deep Dive

Won Jun Jang (Uber), Prithvi Raj (Uber)
Agenda

- Demo
- Architecture, data model
- Configuration
- New Features
- Roadmap
- Project governance, public meetings, contributions
- Discussion
Let’s look at some traces

demo time: http://bit.do/jaeger-hotrod
Distributed Tracing Systems

- Distributed transaction monitoring
- Performance and latency optimization
- Root cause analysis
- Service dependency analysis
- Distributed context propagation
Under the hood
Architecture, etc.
Technology Stack

- Go backend
- Pluggable storage
  - Cassandra, Elasticsearch, memory, ...
- React/Javascript frontend
- OpenTracing Instrumentation libraries
JAEGGER

Host or Container

Application
Instrumentation
OpenTracing API
jaeger-client

jaeger-agent (Go)

Trace Reporting
Thrift over TChannel

Trace Reporting
Thrift over UDP

jaeger-collector (Go)
memory queue

Adaptive Sampling

jaeger-ui (React)

jaeger-query (Go)

Data Store (Cassandra)

CLOUD NATIVE COMPUTING FOUNDATION

Architecture 2017
Configuration

- **Viper/Cobra**
  - Flag: `--span-storage.type=memory`
  - Env: `SPAN_STORAGE_TYPE=memory`

- **k8s ConfigMaps**

- **UI configuration**
  - Control the top menu
  - Google Analytics token
  - Other parameters in the future
Performance Settings

● Collector
  ○ --collector.num-workers
  ○ --collector.queue-size

● Agent
  ○ --processor.jaeger-compact.server-queue-size
  ○ --processor.jaeger-compact.workers
Monitoring

● **Metrics**
  ○ `--metrics-backend`
    ■ prometheus (default), expvar
  ○ `--metrics-http-route`
    ■ `/metrics` (default)

● **Scraping Endpoints**
  ○ Query service - API port 16686
  ○ Collector - HTTP API port 14268
  ○ Agent - sampler port 5778
New Features
Adaptive Sampling

- 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
Asynchronous Ingestion

- Synchronous Ingestion was unable to cope with traffic spikes
- Created a Ingester and Indexer component that uses Kafka as a buffer
Trace Comparisons
Jaeger Operator

- Jaeger Operator for Kubernetes is now available
- Can be used to bring up Jaeger for both demo purposes and Production environments
Roadmap

Things we are working on
Post Trace Sampling

- Jaeger currently only supports head based sampling
- Sample all traces and use heuristics to determine “interesting” traces.
Based on Kafka and Apache Flink
Support aggregations and data mining
Examples:
  ○ Pairwise dependencies diagram
  ○ Path-based dependencies diagram
  ○ Latency histograms
Incomplete Span Support

- Add Ability to store/retrieve partial spans
- Use case:
  - Certain workflows are hours long. Unfortunately spans are only emitted once after it’s Finished(). “Root span” is missing until the complete workflow is finished.
Project & Community
Contributors are welcome
Contributing

- make, glide
- 100% test coverage
- Agree to the Certificate of Origin
- Sign commits (-s)
- Backend, client libraries, k8s, data pipeline
Community

- GitHub: [https://github.com/jaegertracing](https://github.com/jaegertracing)
- Chat: [https://gitter.im/jaegertracing/](https://gitter.im/jaegertracing/)
- Mailing List - jaeger-tracing@googlegroups.com
- Blog: [https://medium.com/jaegertracing](https://medium.com/jaegertracing)
- Twitter: [https://twitter.com/JaegerTracing](https://twitter.com/JaegerTracing)
- Bi-Weekly Community Meetings
Q & A
Open Discussion