# Container Native FS Interposer

Julia Hua, Jiawei Xiang, Hilario Gonzalez, Juncheng Cao
**Mentors:** Vasily Tarasov, Alex Merenstein

October 23, 2024

# Recap

Container
Native FS
Interposer

Recap

Jaeger

FUSE

Plans

## Last sprint

- Support CSI volume stacking
- CSI benchmark in CI
- Tracing FUSE filesytem

## This sprint

- Deploy Jaeger into test cluster
- Flesh out Throttling FS
- Integrate Tracing FS with Jaeger
- Integrate Faulty fs with Jaeger

# What is Jaeger

## Jaeger

open source, distributed tracing platform

- monitor distributed workflows
- find & fix performance bottlenecks
- track down root causes
- analyze service dependencies

## Span

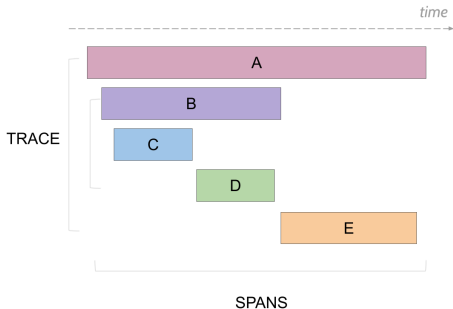Metric monitoring systems works on **Numbers**, but Jaeger operates on **Spans**
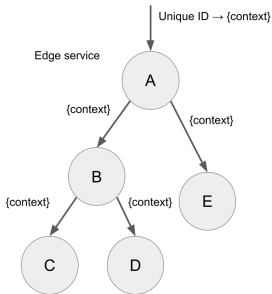
A **span** *represents a* **logical unit of work** *that has an operation name, the start time of the operation, and the duration. Spans may be* **nested** *and ordered to model causal relationships.*
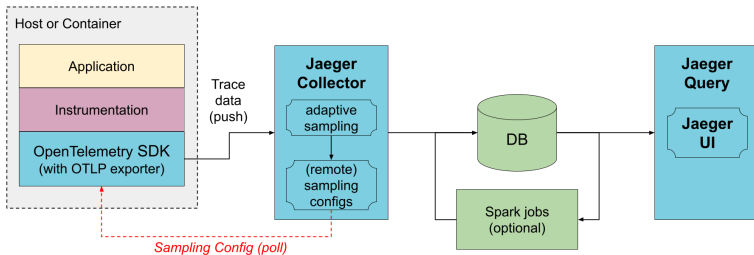
# Jaeger Deployment Architecture

Container
Native FS
Interposer

Recap
Jaeger
FUSE
Plans

# Workload Tracing
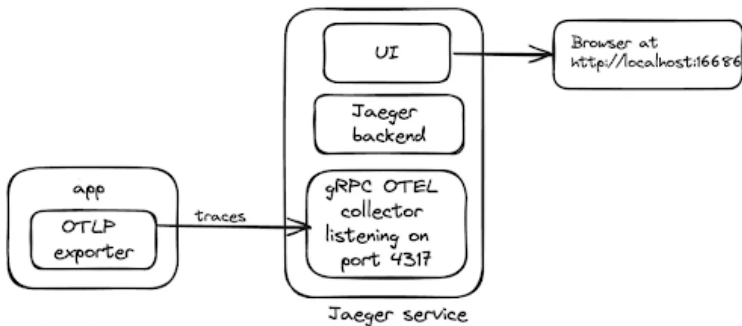
Container
Native FS
Interposer

Recap

Jaeger

FUSE

Plans

## Progress

- Low-level API
- Export tracing data to Jaeger

# Metric Collection

## What

Gather quantitative and aggregate data on the performance and behavior of an application over time

- Frequency of each file operation
- Error rate
- Latency distribution

## Why

- Understand type of workload
- Set up alerts when error rate or latency exceeds a threshold
- Monitor performance trends

# IO Throttling

## Problem

FUSE mandates IOs to be fully fulfilled (return exactly the number of bytes as requested)

## Solution

Block the operation until we have enough IO quota left

# Transition Faulty-fs Codebase to C++

### Process

There were some simple changes that needed to be implemented widely throughout the file system code

- Dockerfile Compilation
- Included headers
- NULL →nullptr
- Change IO interface

# OTEL for Simulated Error Logging

## Process

- Configure OTEL SDK for application
- Instrument the error logging with OTEL's API
- Set up exporter to backend (Jaeger)

## Why

- Integrate with OTEL used for tracing and metric collection
- Shared Jaeger backend

# Future Plans

Container
Native FS
Interposer

Recap
Jaeger
FUSE
Plans

- Implement metric collection layer in FUSE
- Grafana dashboard for the metrics
- Make filesystem attributes configurable
- Combine all fuse filesystems into a single binary
- Post processing script for resolving file path
- Create parent spans covering multiple operations

# Burndown

| Title | Date | Moved to Done Estimated Hours | Moved to Done Actual Hours | Ideal Hours | Burndown Estimated | Burndown Actual | Team # | Hours/Sprint |
|---|---|---|---|---|---|---|---|---|
| | 2024/10/08 | 0 | 0 | 100 | 100 | 100 | 5 | 20 |
| | 2024/10/09 | 0 | 0 | 93 | 93 | 93 | | |
| Throttle IO with token bucket | 2024/10/10 | 10 | 9 | 86 | 76 | 77 | | |
| Deploy jaeger in test cluster | 2024/10/11 | 3 | 4 | 79 | 76 | 75 | | |
| Implement more faults | 2024/10/12 | 6 | 6 | 71 | 65 | 65 | | |
| Add test for token bucket | 2024/10/13 | 4 | 4 | 64 | 60 | 60 | | |
| Make tracing work with k8s | 2024/10/14 | 5 | 4 | 57 | 52 | 53 | | |
| Integrate throttling | 2024/10/15 | 3 | 3 | 50 | 47 | 47 | | |
| | 2024/10/16 | 0 | 0 | 43 | 43 | 43 | | |
| | 2024/10/17 | 0 | 0 | 36 | 36 | 36 | | |
| Fully implement tracing | 2024/10/18 | 5 | 5 | 29 | 24 | 24 | | |
| Implement more delay | 2024/10/19 | 4 | 5 | 21 | 17 | 16 | | |
| Modularize tracing | 2024/10/20 | 3 | 4 | 14 | 11 | 10 | | |
| | 2024/10/21 | 0 | 0 | 7 | 7 | 7 | | |
| Prepare for pre | 2024/10/22 | 6 | 6 | 0 | -6 | -6 | | |

Moved to Done Estimated Hours, Actual Hours, Ideal Hours, Running
Total Estimate and Running Total Actual