Distributed Tracing with OpenShift

Red Hat

Introduction to OpenTelemetry

What is OpenTelemetry?

OpenTelemetry (or "Otel") is an open-source observability framework for instrumenting, generating, collecting, and exporting telemetry data.

It's a collection of tools, APIs, and SDKs that helps you understand the performance and behavior of your software.

Think of it as the "plumbing" that standardizes how you get data out of your applications.

The Three Pillars of Observability

Traces: Represent the end-to-end flow of a request through a distributed system. They show the path of a transaction across multiple services. Essential for distributed tracing and performance analysis.

Metrics: A collection of aggregated measurements that represent a service's state over a period of time. Think of things like CPU usage, request rates, or memory consumption.

Logs: Timestamped text records of events that happen within an application. They are crucial for debugging and understanding specific events.

How It Works: The OpenTelemetry Architecture

Instrumentation: This is the process of generating telemetry data from your code. OpenTelemetry provides SDKs for multiple programming languages (Java, Python, Go, etc.) to automatically or manually instrument your application.

OpenTelemetry Collector: A vendor-agnostic intermediary that receives, processes, and exports telemetry data. It can be deployed as a single agent per host or as a gateway for multiple services.

Backend Analysis: The Collector exports the data to a backend, such as Prometheus, Tempo, or a commercial observability platform. This is where you can visualize, analyze, and query your data.

Why Use OpenTelemetry?

Vendor Neutrality: Avoids vendor lock-in by providing a standardized way to collect telemetry data. You can switch your backend without changing your application code.

Consistency: Ensures all teams across an organization are using the same format for observability data.

Rich Ecosystem: Supported by a large, active community and integrated with numerous tools and platforms.

Distributed Tracing: Provides the best-in-class solution for understanding complex microservice architectures.

Getting Started

Step 1: Choose Your Language: Find the OpenTelemetry SDK for your programming language.

Step 2: Instrument Your Code: Add the necessary libraries to your application. For many common frameworks, this can be done with automatic instrumentation.

Step 3: Deploy the Collector: Set up an OpenTelemetry Collector to receive data from your application.

Step 4: Choose a Backend: Configure the Collector to send data to your observability backend of choice.

Thank you for your attention, and now the interesting part … Hands on time

Questions ?

Link public git repository:
https://github.com/michaelalang/openshift-anwendertreffen-wien-25-09-2025

li n/ mpany/rec t

youtube.com/user/R os

facebook.c ic

witter.com/R at

Red Hat