



Gain Better Observability with OpenTelemetry and InfluxDB

Suyash Joshi

sjoshi@influxdata.com

Sr. Developer Advocate

InfluxData

@suyashcjoshi

@influxdb



whoami



LinkedIn



Agenda

- Introduction to OpenTelemetry and logs, traces, and metrics
- Overview of InfluxDB Cloud powered by InfluxDB 3.0
- Live Demo of Observability App
 - Jaeger
 - Grafana
 - Hot R.O.D.
 - Telegraf
- Learning Resources
- Q&A

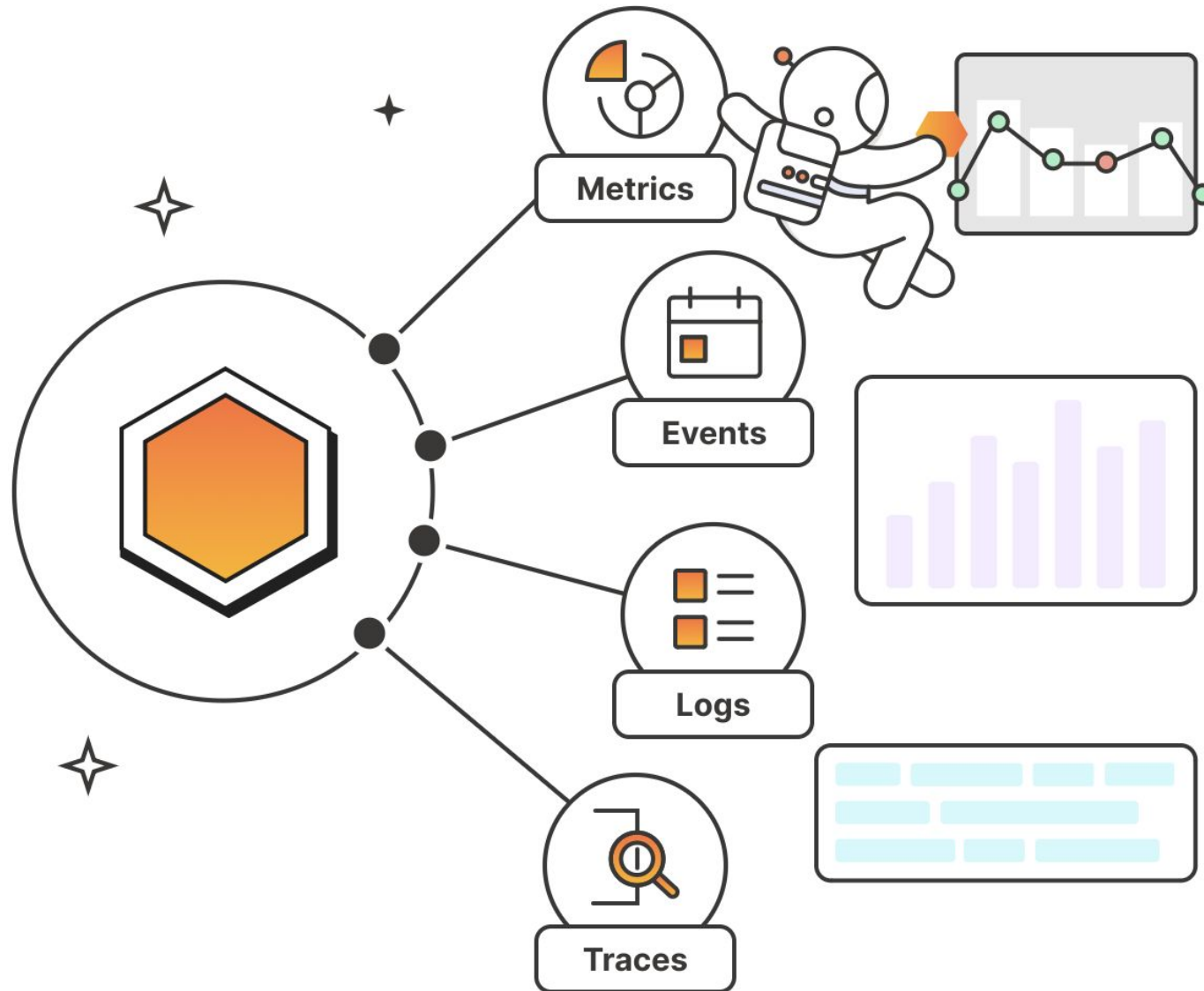
Introduction to OpenTelemetry



OpenTelemetry is an open-source CNCF project providing tools to collect and export telemetry data for analyzing software performance.

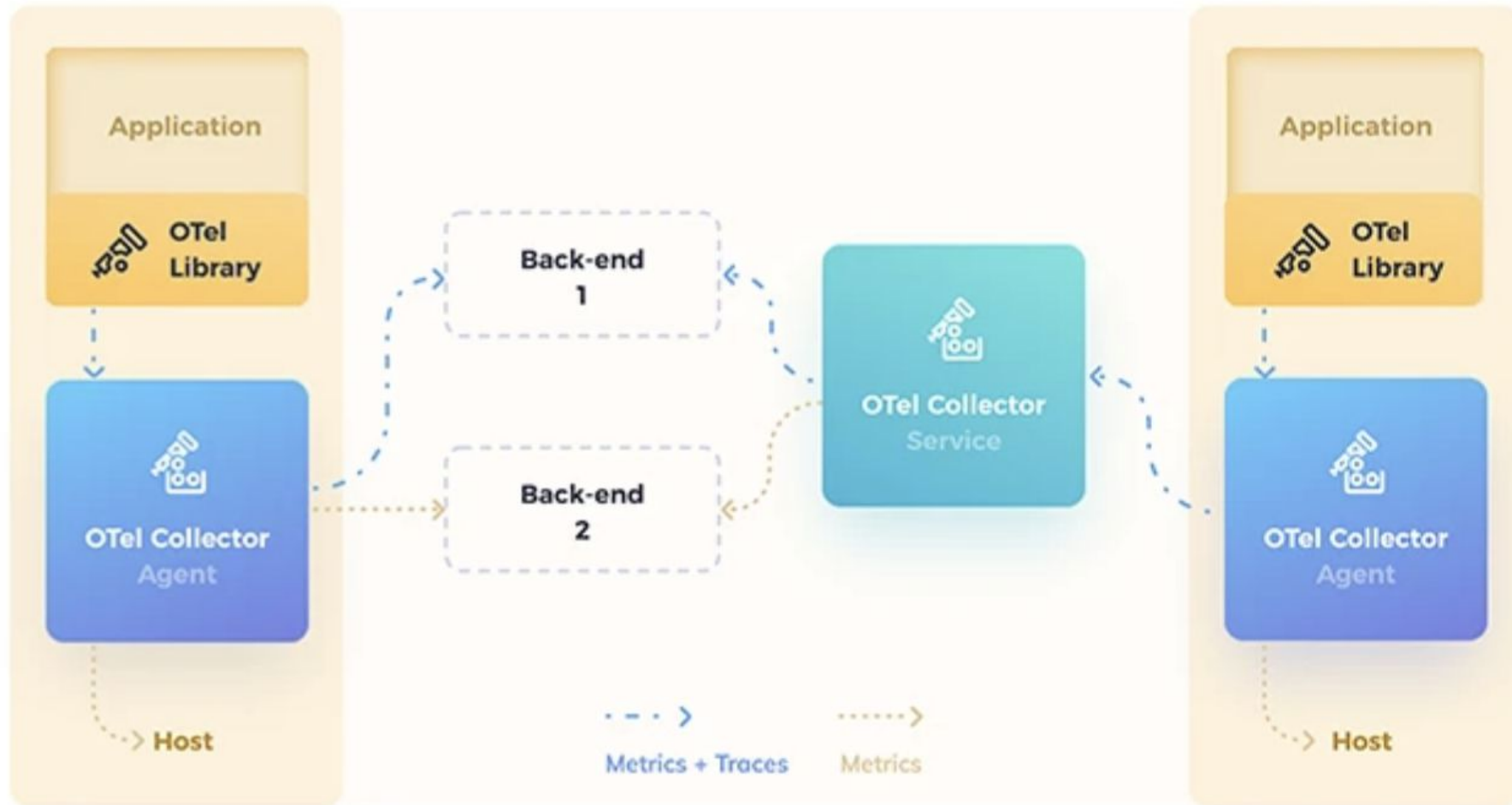
Reference: <https://opentelemetry.io/>

What is Observability?



Events	Metrics	Traces	Logs
Debugging & Correlation of incidents	Monitoring system health, overall trends	Debugging, Diagnosing latency issues	Debugging
Discrete occurrence or actions at irregular intervals	Numerical data point, scalar values at regular intervals	Record of a request's path through a distributed system using trace ID.	Structured / Unstructured event data with timestamp
HTTP Request/Responses, User Signups, Clicks etc	%CPU of Usage, No. of Errors	trace_id: abc123 ┌── frontend [100ms] ─┬── auth_service [50ms] ─┬── database [200ms]	<pre> {"type": "user_login","timestamp": "2024-03-15T10:30:00Z","u ser_id": "12345","device": "mobile","location": "US-NYC"} </pre>
InfluxDB	InfluxDB	New Relic, InfluxDB*	Splunk, InfluxDB*

OpenTelemetry key concepts

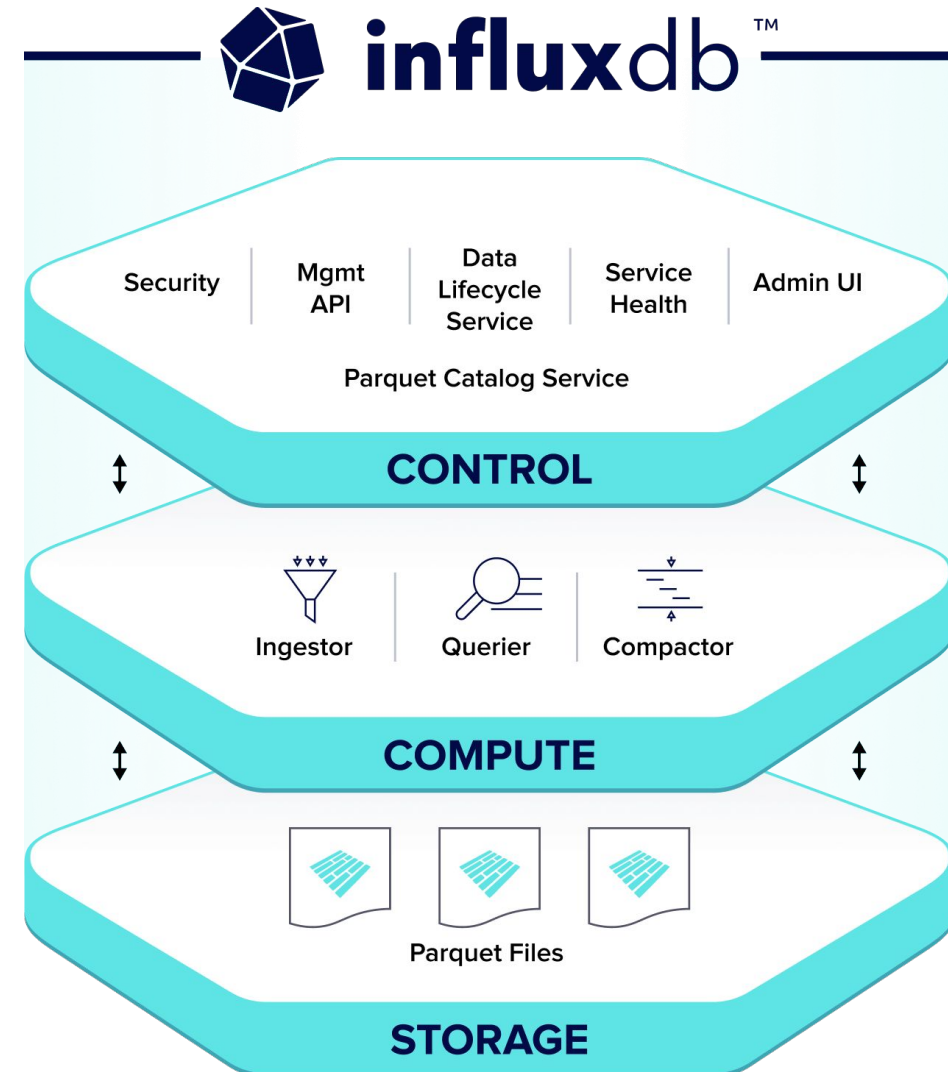


What can OpenTelemetry do for me?

- A single vendor-neutral collector binary and a vendor-agnostic instrumentation library that can be deployed in a variety of ways with support for both automatic and manual instrumentation.
- An end-to-end implementation to generate, emit, collect, process and export telemetry data.
- Full control of your data with the ability to send data to multiple destinations in parallel through configuration.
- Open-standard semantic conventions to ensure vendor-agnostic data collection
- A path forward no matter where you are on your observability journey.

Overview of InfluxDB Cloud

InfluxDB 3.0 : Built on Modern Open Source Stack



Apache Flight

Transport columnar data at high speeds based on Arrow format

Apache DataFusion

Fast query execution engine written in Rust

Apache Arrow

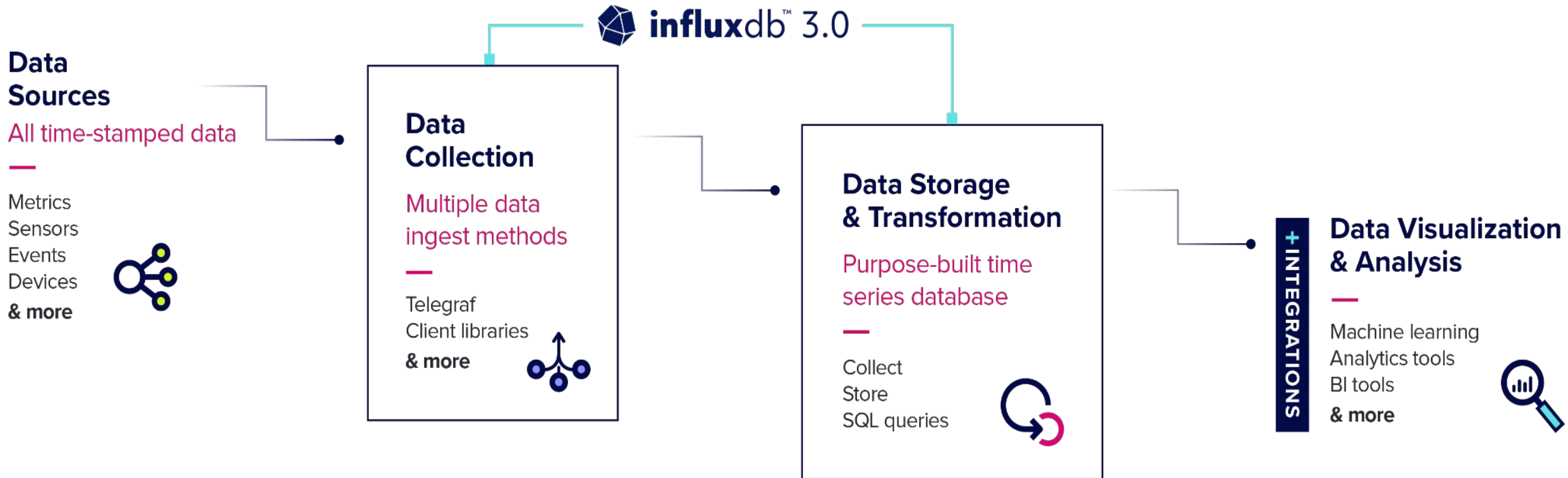
Optimized for running large analytical workloads

Apache Parquet

Open column-oriented file format designed for efficient data storage and retrieval

Typical Architecture & Deployment

InfluxDB Platform



Unlimited cardinality

InfluxDB 3.0's database engine is designed to handle high cardinality datasets without compromising performance. Track your devices or software instances on many dimensions. This is what will allow us to store logs and traces, instead of only metrics.

Native SQL support

InfluxDB 3.0 features native SQL support. Most developers (and all data scientists) are already familiar with SQL, so teams can gather insights faster and with less friction. We are also working on expanding what you can do with SQL to allow downsampling, aggregation, and other key features. However, InfluxDB IOx also supports InfluxQL and Flux. Once again, we're taking backwards compatibility seriously.

High-performance data ingestion

InfluxDB 3.0 is designed to handle high write and query loads, making it suitable for processing massive amounts of monitoring data in real-time. Its high-performance capabilities help developers capture and analyze data to identify patterns, trends, and anomalies, which leads to faster and more informed decision-making.

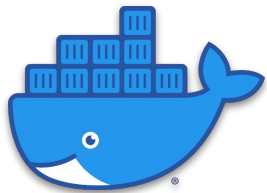
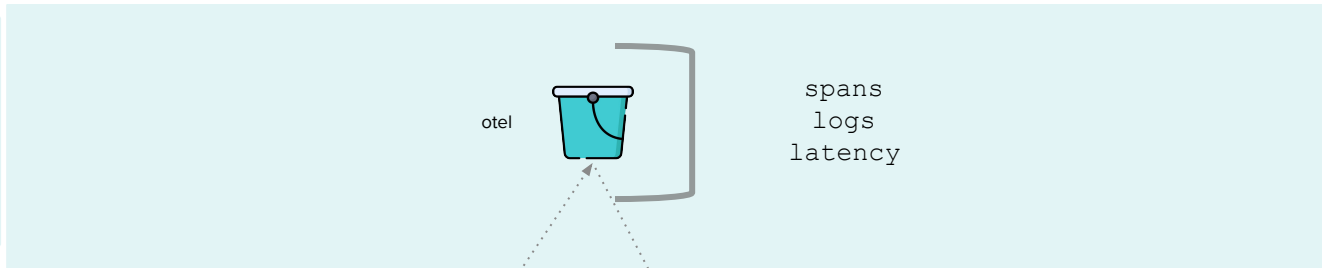
Seamless integration with observability tools

InfluxDB 3.0 integrates with popular observability tools such as Grafana, Jaeger, and OpenTelemetry to create a powerful ecosystem for monitoring, tracing, and visualizing your infrastructure. It also integrates with pandas and other data science tools to take action on your data.

Introduction to InfluxDB Observability Project

OpenTelemetry with InfluxDB

This demo provides a practical example of integrating InfluxDB, a high-performance time series database, with OpenTelemetry, an open-source observability framework, to achieve real-time monitoring and tracing of a distributed application.



Hot R.O.D



otelcol-influxdb



Jaeger-query



Jaeger-UI

- Aims to provide a standard for converting **OTEL -> InfluxDB Schema** and **InfluxDB Schema -> OTEL**
- Parts of **otelcol-influxdb** can be replaced with **Telegraf**



<https://github.com/influxdata/influxdb-observability>

Hot R.O.D - Rides on Demand

Your web client's id: 2131

Hot R.O.D.

 Rides On Demand 

Rachel's Floral Designs

Trom Chocolatier

Japanese Desserts

Amazing Coffee Roasters

Click on customer name above to order a car.

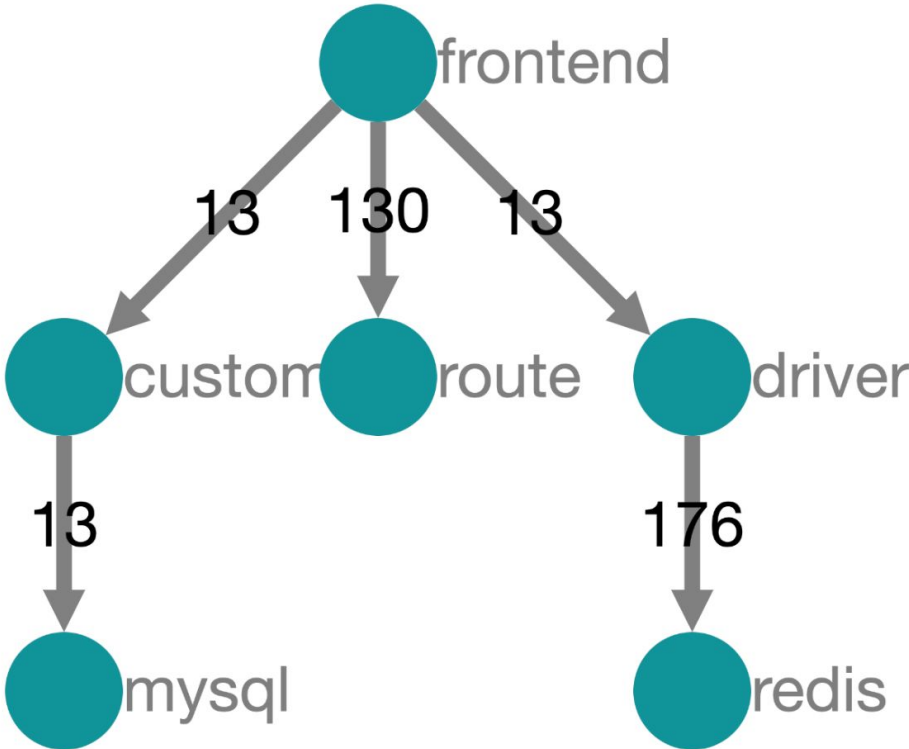
HotROD **T773436C** arriving in 2min [req: 2131-13, latency: 682ms] [\[find trace\]](#)
HotROD **T765760C** arriving in 2min [req: 2131-12, latency: 693ms] [\[find trace\]](#)
HotROD **T711432C** arriving in 2min [req: 2131-11, latency: 698ms] [\[find trace\]](#)
HotROD **T701938C** arriving in 2min [req: 2131-10, latency: 683ms] [\[find trace\]](#)
HotROD **T712496C** arriving in 2min [req: 2131-9, latency: 742ms] [\[find trace\]](#)
HotROD **T794861C** arriving in 2min [req: 2131-8, latency: 711ms] [\[find trace\]](#)
HotROD **T769365C** arriving in 2min [req: 2131-7, latency: 641ms] [\[find trace\]](#)
HotROD **T784896C** arriving in 2min [req: 2131-6, latency: 681ms] [\[find trace\]](#)
HotROD **T781260C** arriving in 2min [req: 2131-5, latency: 690ms] [\[find trace\]](#)
HotROD **T778376C** arriving in 2min [req: 2131-4, latency: 727ms] [\[find trace\]](#)
HotROD **T791772C** arriving in 2min [req: 2131-3, latency: 747ms] [\[find trace\]](#)
HotROD **T722133C** arriving in 2min [req: 2131-2, latency: 779ms] [\[find trace\]](#)
HotROD **T713042C** arriving in 2min [req: 2131-1, latency: 771ms] [\[find trace\]](#)



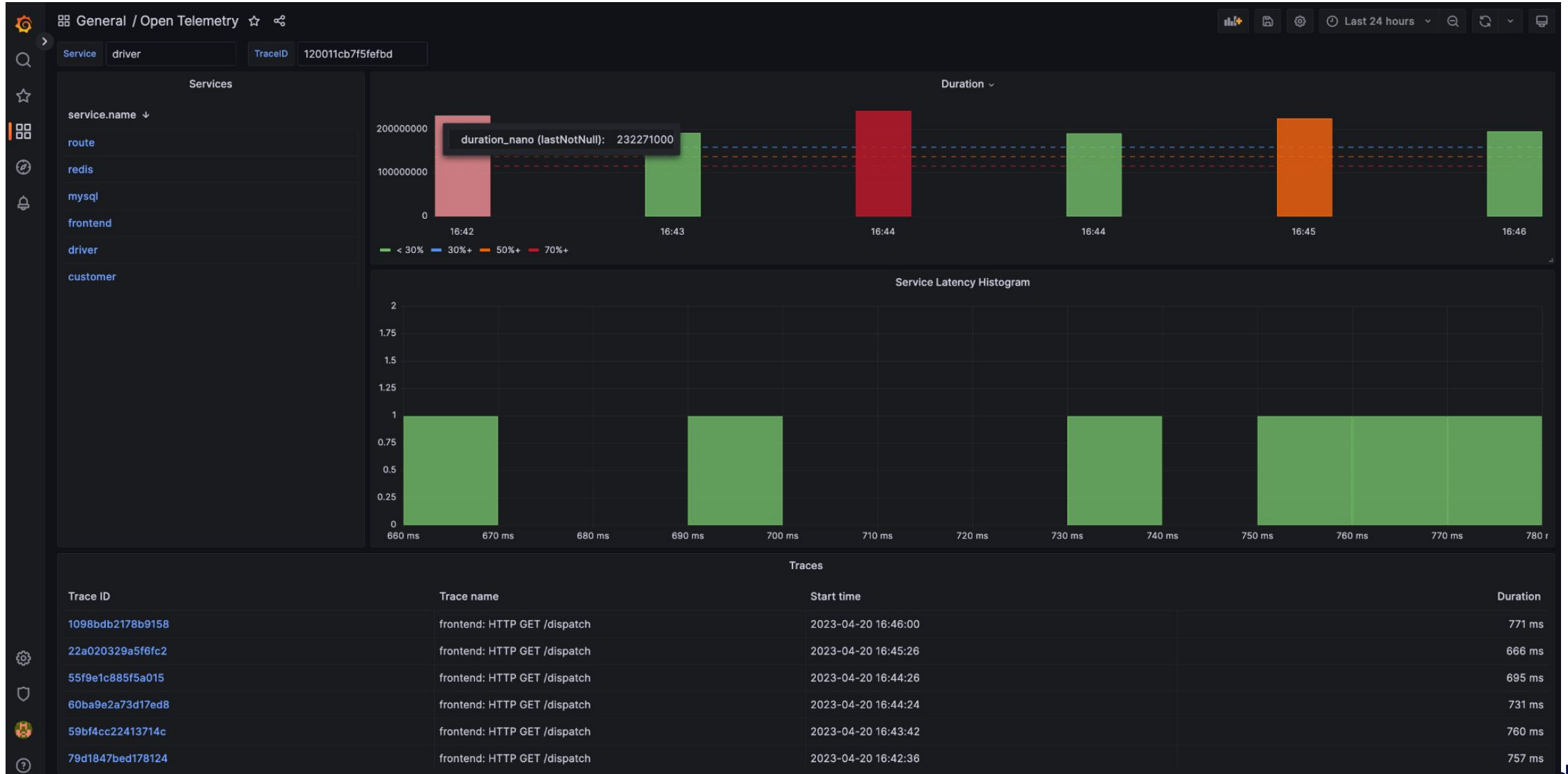
Jaeger

JAEGER UI Search Compare **System Architecture**

Force Directed Graph **DAG**



Grafana



InfluxDB Cloud

The screenshot displays the InfluxDB Cloud Data Explorer interface. On the left, the 'Schema Browser' shows the 'otel' bucket and the 'logs' measurement selected. The main area contains a SQL editor with the query `SELECT * from "logs"`. Below the editor, a 'Ready (295ms)' status is shown, along with 'CSV' and 'Past 1h' options, and a 'RUN' button. The results are displayed in a table view, showing 1 table and 839 rows. The table has two columns: 'attributes' and 'name'. The first row shows a 'redis timeout' error, while the subsequent rows show 'HTTP request rec' entries.

table	attributes	name
_result	no group string	no group string
0	{"driver_id":"T756842C","error":"redis timeout","level":"error"}	redis timeout
0	{"level":"info","method":"GET","url":"/route?dropoff=211%2C653\u0026pickup=393%2C398"}	HTTP request rec
0	{"level":"info","method":"GET","url":"/route?dropoff=211%2C653\u0026pickup=320%2C366"}	HTTP request rec
0	{"level":"info","method":"GET","url":"/route?dropoff=211%2C653\u0026pickup=14%2C235"}	HTTP request rec

Running InfluxDB Observability Demo

1. Sign up / Login to [InfluxDB Cloud account](#)
2. Create 2 buckets, “otel” and “otel-archival” where “otel-archival” has a longer retention policy.
3. Clone/Download sample app from [GitHub](#)
4. Update “.env” file with the cloud account information.
5. Install flight-sql as per the README.md.
6. Build and run the docker images as per the README.md.
7. Import your dashboard with the JSON at **demo/grafana/dashboards/** into Grafana.
8. Create traces by clicking on a customer on the HotRod application.

Live Demo

Resources

Sign up for Free

❖ www.influxdata.com/cloud

❖ via cloud marketplace



Resources

- Sample app demo (GitHub): <https://github.com/InfluxCommunity/influxdb-observability>
- OpenTelemetry Demo: <https://opentelemetry.io/docs/demo/>
- Jaeger: <https://www.jaegertracing.io/>
- Blogs : <https://www.influxdata.com/blog>
- Documentation: <https://docs.influxdata.com>
- InfluxDB University (free training): <https://influxdbu.com>
- Community: <https://influxcommunity.slack.com> & <https://community.influxdata.com>



THANK YOU