



NABENIK

OpenTelemetry: A língua franca da observabilidade

Você não pode melhorar o que não pode medir

Nabenik

11 de junho de 2025

Sistema de TI

- Aplicações
- Serviços de terceiros
- Processos intermédios
- Hosts

Teoria: Coletamos dados, processamos para identificar KPIs (golden signals) e criamos alarmes

Fontes de informação

- Eventos
- Logs
- Métricas
- Traces

Sistema de TI

- Aplicações: Binário (Go, GraalVM Native), minified (JS), jar, war (Java), K8s (containers), FaaS
- Serviços de terceiros (PaaS especializados)
- Processos intermédios (Banco de dados, message queue)
- Hosts (LSB, Alpine, openrc, Windows, etc.)
- ElasticSearch, Grafana, Datadog, New Relic, Cloudwatch, etc.

Fontes de informação

- Logs: Arquivos (/var/log), Systemd, FluentD
- Metrics: Hosts, JMX, Prometheus endpoints (Spring Actuator, Microprofile Metrics)
- Traces: Jaeger, Zipkin

Precisamos dados de **fontes e formatos diversos**, os quais podem ser **enviados ou coletados** para ser analisados em **plataformas de observabilidade** com diversas capacidades e riscos -e.g. Vendor lock in, tecnologias muito específicas, incompatibilidade entre versões-

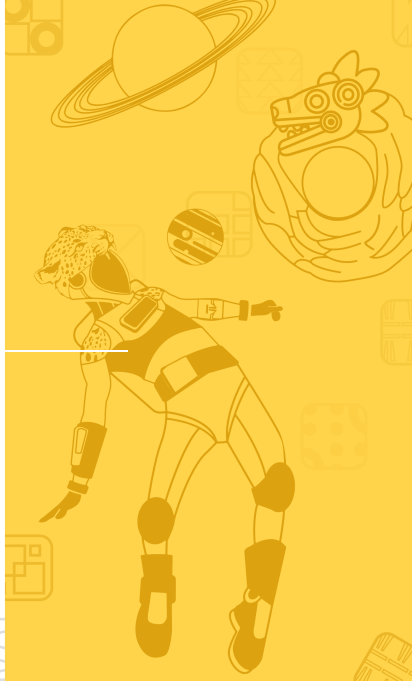
Precisamos ...

data pipelines de observabilidade baseados numa **especificação** padrão



NABENIK
Tecnología para tu éxito

OpenTelemetry



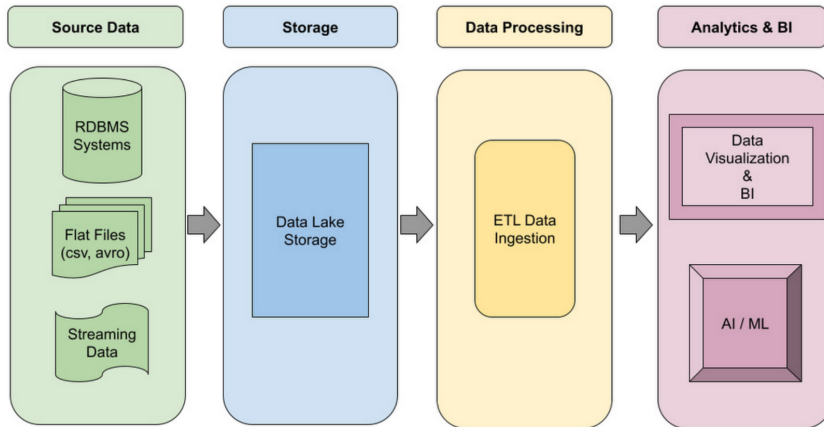
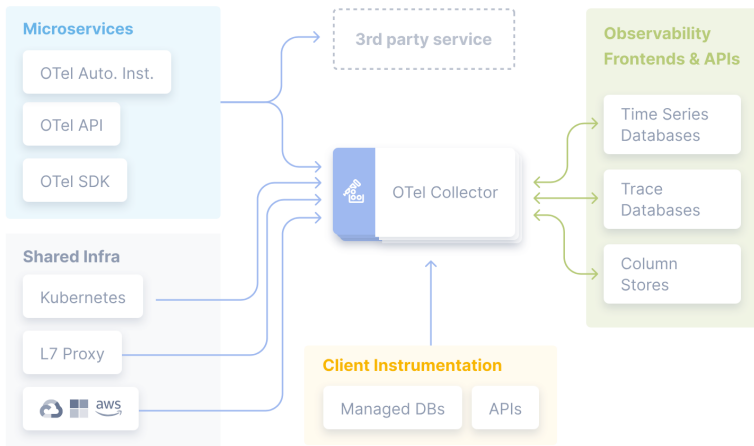
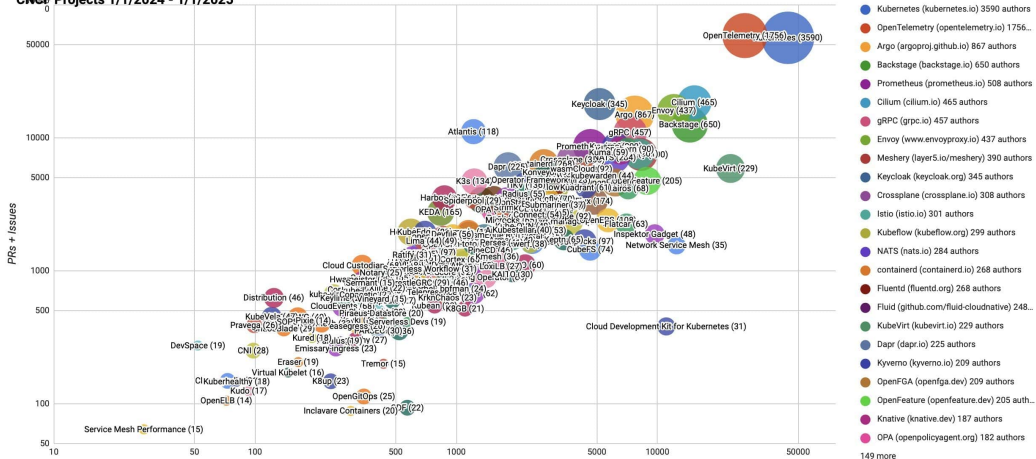


Figura 1: Data pipeline




1. Instrumentação
2. Coleta
3. Envio

CNCB Projects 1/1/2024 - 1/1/2025



- **</> Biblioteca (Library)**
 - Instrumentação manual usando APIs do OpenTelemetry diretamente no código
- **⚙ Framework**
 - Integração com frameworks que já oferecem suporte oficial
 - Ex: Spring Boot, Quarkus, ASP.NET Core
- **✂ Manipulação de artefato**
 - Instrumentação automática por meio de interceptores, agentes ou manipulação de bytecode
 - Ex: Java Agent (javaagent), AWS Lambda layer, JavaScript Zero code instrumentation

Otel Collector - Upstream


[Docs](#) [Ecosystem](#) [Status](#) [Community](#) [Training](#) [Blog](#) [English ▾](#)

Docs

- What is OpenTelemetry?
- ▶ Getting Started
- ▶ Concepts
- ▶ Demo
- ▶ Language APIs & SDKs
- ▶ Platforms
- ▶ Zero-code Instrumentation
- ▼ **Collector**
 - Quick start
 - Install the Collector
 - ▶ **Deployment**
 - Configuration
 - Management
 - Distributions
 - Internal telemetry
 - Troubleshooting
 - Scaling the Collector
 - Transforming telemetry
 - Architecture

Collector

Vendor-agnostic way to receive, process and export telemetry data.



The diagram illustrates the Otel Collector architecture. It shows a central processing pipeline flanked by Receivers and Exporters. On the left, under the 'Receivers' column, are OTLP, Jaeger, and Prometheus. On the right, under the 'Exporters' column, are OTLP, Jaeger, and Prometheus. The central pipeline consists of two rows of components: the top row includes 'Extensions: health, pprof, zpages', 'Batch', '...', and 'Attributes'; the bottom row includes 'Batch', '...', and 'Filter'. Between these rows is a 'Processors' block. Arrows indicate the flow of data from Receivers through the central pipeline to Exporters.

Splunk Observability Cloud

[Splunk Observability Cloud](#) > [Manage Data](#) > [Splunk OpenTelemetry Collector](#)

Get started with the Splunk Distribution of the OpenTelemetry Collector

- > Get started: Understand the Collector
- > Collector components
- > Collector for Kubernetes
- > Collector for Linux
- > Collector for Windows
- > Splunk Add-On for OpenTelemetry Collector
- > Other deployment tools: EC2, Fargate, Nomad, P
- > Automatic discovery of and services
- Use the Universal Forwarder

Get started with the Splunk OpenTelemetry Collector

A banner for the AWS Distro for OpenTelemetry. It features a dark blue background with a white line graph icon in the top left. The text 'AWS Distro for OpenTelemetry' is prominently displayed in white. Below it, a smaller line of text reads 'Secure, production-ready open source distribution with predictable performance'. At the bottom, there is a yellow button with the text 'DOWNLOAD NOW'.



Grafana Alloy

Grafana Alloy combines the strengths of the leading collectors into one place. Whether observing applications, infrastructure, or both, Grafana Alloy can collect, process, and export telemetry signals to scale and future-proof your observability approach.

Overview

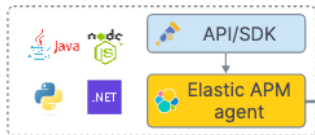


Introduction

AWS Distro for OpenTelemetry is a secure, production-ready, AWS-supported distribution of the OpenTelemetry project. Part of the Cloud Native Computing Foundation, OpenTelemetry provides open source APIs, libraries, and agents to collect distributed traces and metrics for application monitoring. With AWS Distro for OpenTelemetry, you can instrument your applications just once to send correlated metrics and traces to multiple AWS and Partner monitoring

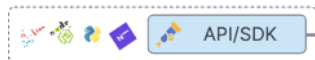
OpenTelemetry API/SDK with Elastic APM agents

Available in Java, .NET, Node.js, and Python



OpenTelemetry Agents

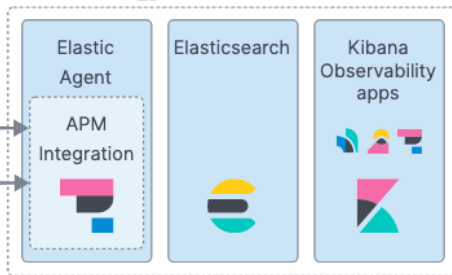
Click [here](#) to see all supported languages



OpenTelemetry Collectors



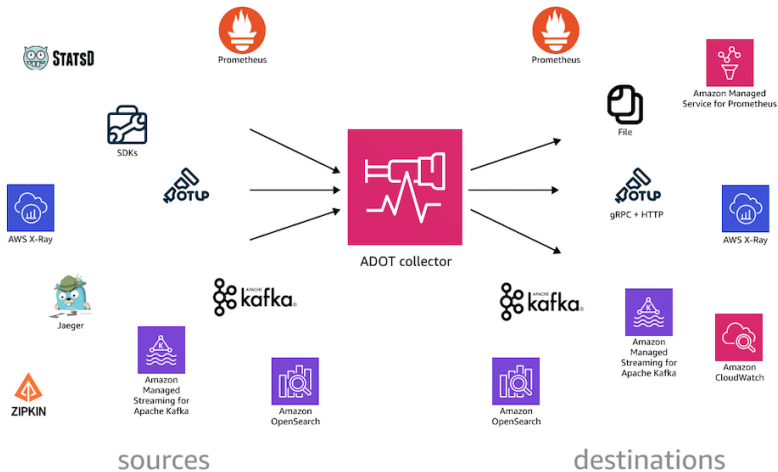
Elastic Observability



Edge machines

Protocol

Hosted on Elastic Cloud

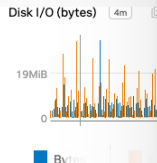


Fatos aleatórios que eu aprendi nesta jornada

1. A instrumentação via *agent* é particularmente boa no Java
2. A instrumentação de framework tem uma limitante, quando você pega bibliotecas não -opinionated- ... ela não funciona
3. Para alguns frameworks -e.g. Spring Boot- existe a instrumentação oficial do OpenTelemetry e a instrumentação oficial do projeto
4. A instrumentação via *agent* precisa fine-tuning em FaaS, -e.g. No AWS lambda o boot fica (ainda mais) lento sem tuning-
5. As *distribuições* do collector facilitam o envio de dados
6. O barco da *padronização* já esta em alto mar, ele chama-se OpenTelemetry

∇_+ Ad

EC2 instances



By AWSUniqueid

4MiB

Wed 25 Sep 2024 10:13:00

By AWSUniqueid

2MiB

+

disk utilization metrics, or it might be under a different identifier within the observability platform.

Given this situation, there are a few steps we can take:

- 1. Verify Integration and Configuration:** Ensure that the AWS instance is correctly integrated with Splunk Observability and that disk utilization metrics are being collected and reported.
- 2. Check for Recent Data:** It's possible that the instance has not reported any recent data for `disk.utilization`, or there might be a delay in data reporting or collection.
- 3. Alternative Metrics or Logs:** Consider checking other related metrics or logs that might provide insights into the instance's performance and health.

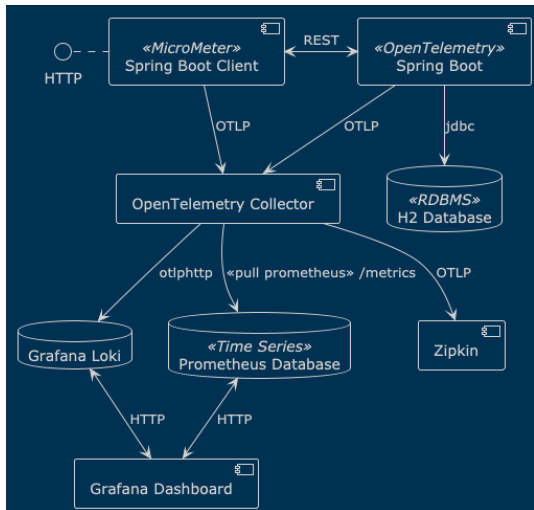
If you have specific concerns or if there's another metric you would like to explore, please let me know. Additionally, if you believe the instance should be reporting `disk.utilization` and it's not showing up, it might be helpful to review the configuration for metric collection on this instance.



NABENIK
Tecnología para tu éxito

Exemplo





A Practical Guide to OpenTelemetry With Spring Boot Workloads

This tutorial demonstrates setting up OpenTelemetry with Spring Boot for observability, including metrics, traces, and logs, using tools like Grafana, Loki, and Tempo.

By Victor Orozco · Apr. 08, 25 · Tutorial

👍 Likes (5) 💬 Comment (0) ⭐ Save 🐦 Tweet 🔗 Share 👁 5.7K Views ⚙

In this tutorial, we consolidated some practical approaches regarding OpenTelemetry and how to use it with Spring Boot. This tutorial is composed of four primary sections:

1. OpenTelemetry practical concepts
2. Setting up an observability stack with OpenTelemetry Collector, Grafana, Loki, Tempo, and Podman
3. Instrumenting Spring Boot applications for OpenTelemetry





Oracle ACE
Pro



- vorozco@nabenik.com
- @tuxtor
- <https://voroeco.com>
- <https://tuxtor.shekalug.org>



Este trabalho está licenciado sob a
licença Creative Commons
Atribuição-NãoComercial-
Compartilhual 3.0 Guatemala
(CC BY-NC-SA 3.0 GT).