Observability strategies to not overload engineering teams.

Finding a balance between visibility and engineering efforts.

whoami

- Brazilian living in Braga
- Open Source Contributor
- Currently working @Coralogix
- Kubernetes, O11y, GitOps, Distributed Systems

Introduction

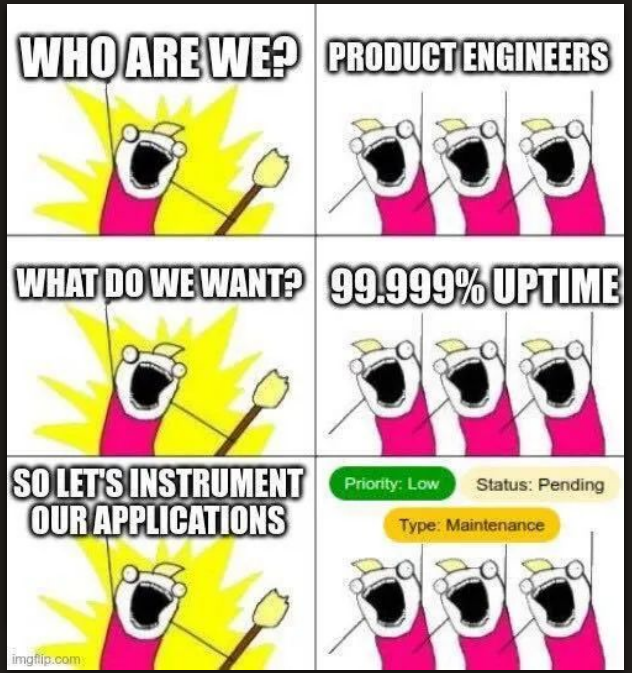
Observability - Engineer night watcher

In the context of software engineering, observability is crucial for understanding how a system is behaving, identifying issues and bottlenecks, and troubleshooting problems quickly.

Introduction

Observability - Engineer nightmare

Instrumenting code to collect all possible metrics and traces can quickly become overwhelming and lead to performance issues.



Introduction

Strategies to not overload engineering teams.

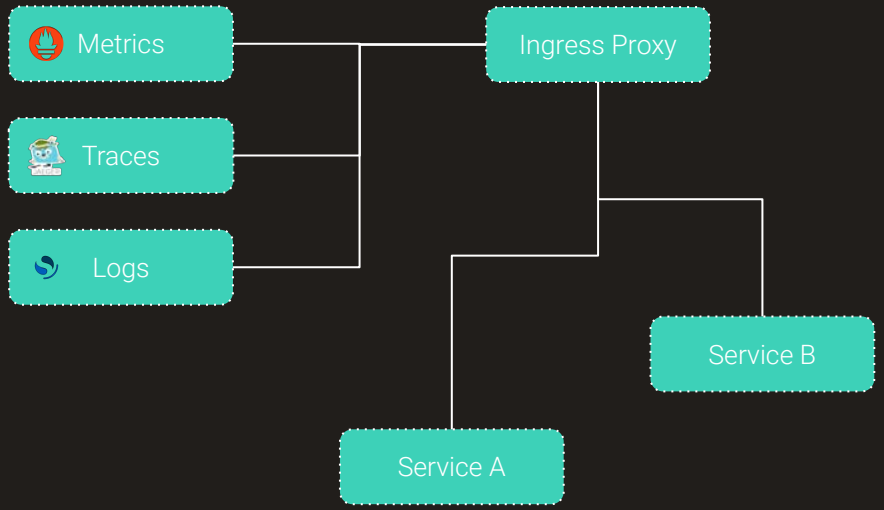
These strategies are designed to give your engineering teams a solid foundation of observability without having to change any code.

- Proxy
- OpenTelemetry
- eBPF

Strategies to not overload engineering teams.

The Proxy Strategy

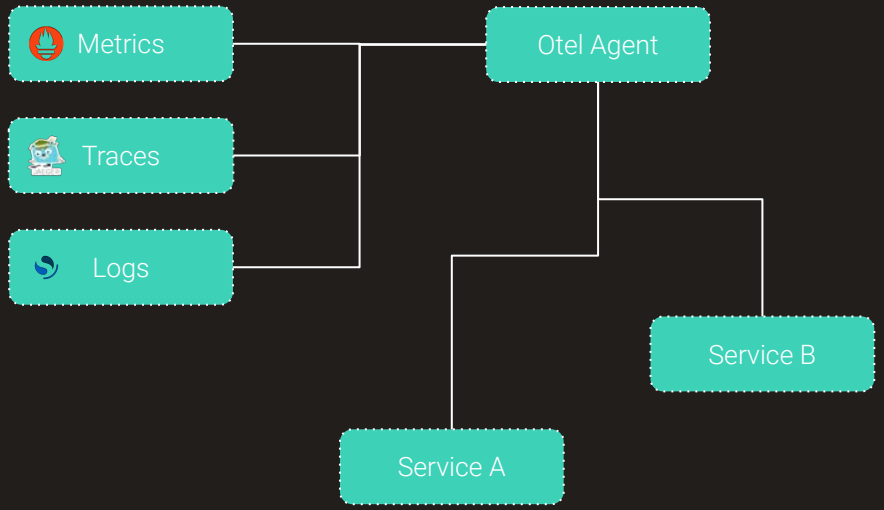
Web proxies are usually part of the platform and a great way to collect telemetry data without any extra effort.



Strategies to not overload engineering teams.

The OpenTelemetry Strategy

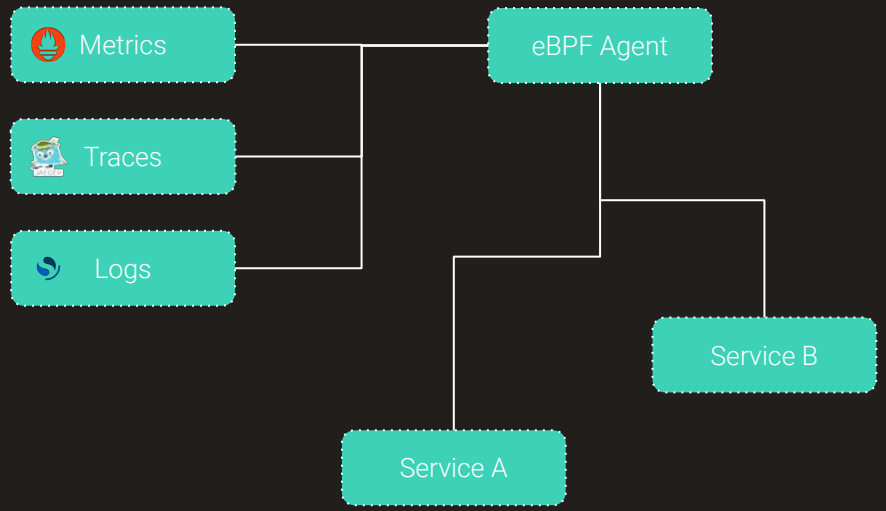
OpenTelemetry auto instrumentation automatically adds instrumentation code to your application, making it easier to collect telemetry data from your distributed systems.



Strategies to not overload engineering teams.

The eBPF Strategy

eBPF (extended Berkeley Packet Filter) is a Linux kernel technology used to trace, monitor, and analyze system performance and behavior, providing observability insights to developers and operators.



What's the best option?

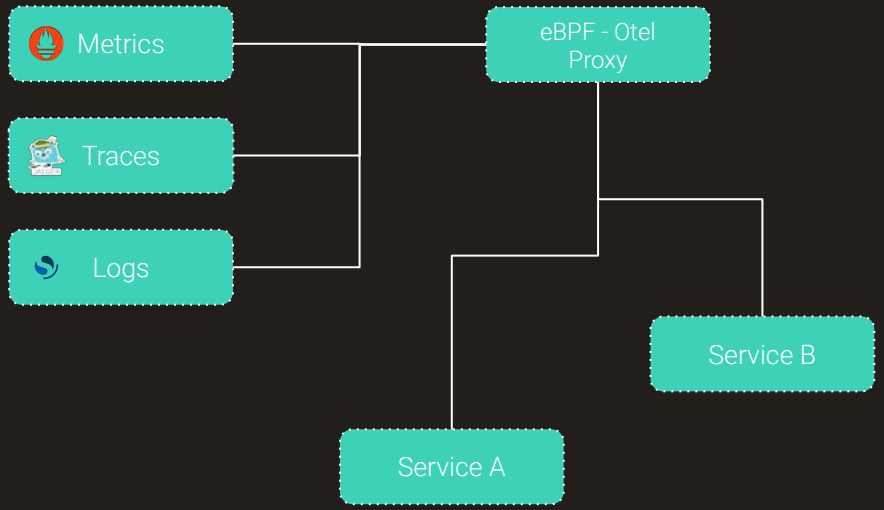
As always, it depends.

	Proxy	OpenTelemetry Auto Instrumentation	eBPF
Tech agnostic	👍	👎	👍
Ensure context propagation	👎	👍	👍
Environment Agnostic	👍	👍	👍
Multi Telemetry Data	👍	👍	👍

Strategies to not overload engineering teams.

The all in one strategy

Why choose only one strategy if you can combine multiple or maybe all strategies, to provide deep visibility about different layers of your system?



Q&A

Thank you!