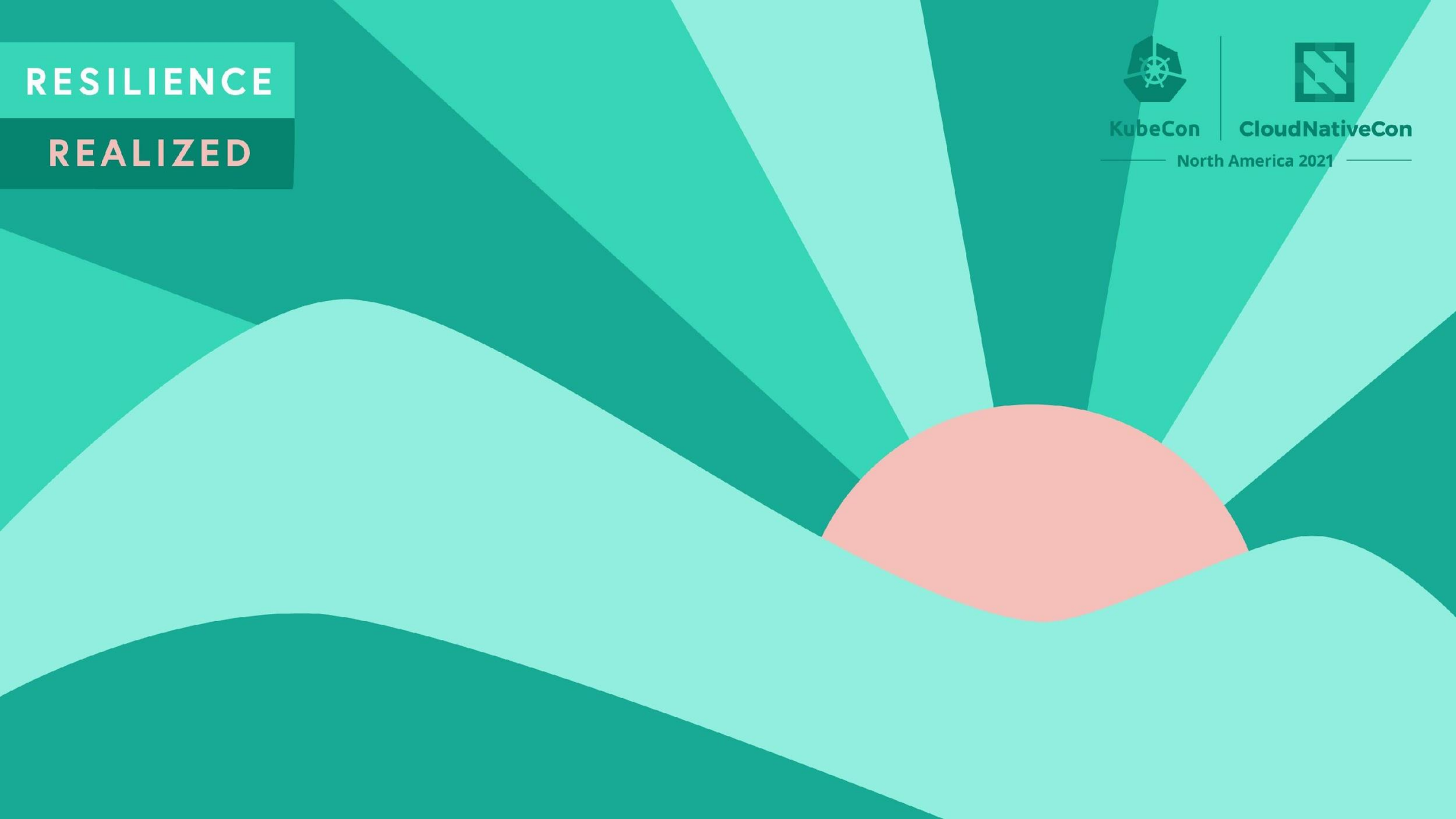

RESILIENCE
REALIZED
KubeCon
CloudNativeCon
North America 2021


KubeCon | CloudNativeCon

North America 2021

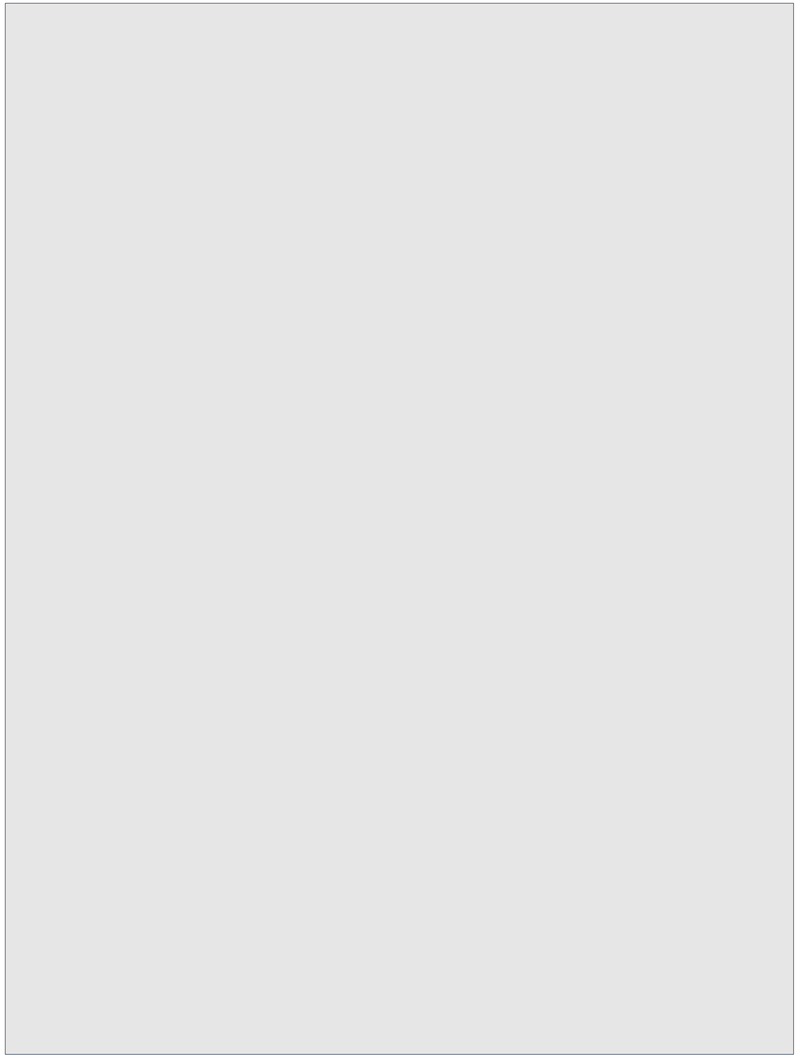
RESILIENCE REALIZED

OpenTelemetry Collector Deployment Patterns

Juraci Paixão Kröhling - Red Hat

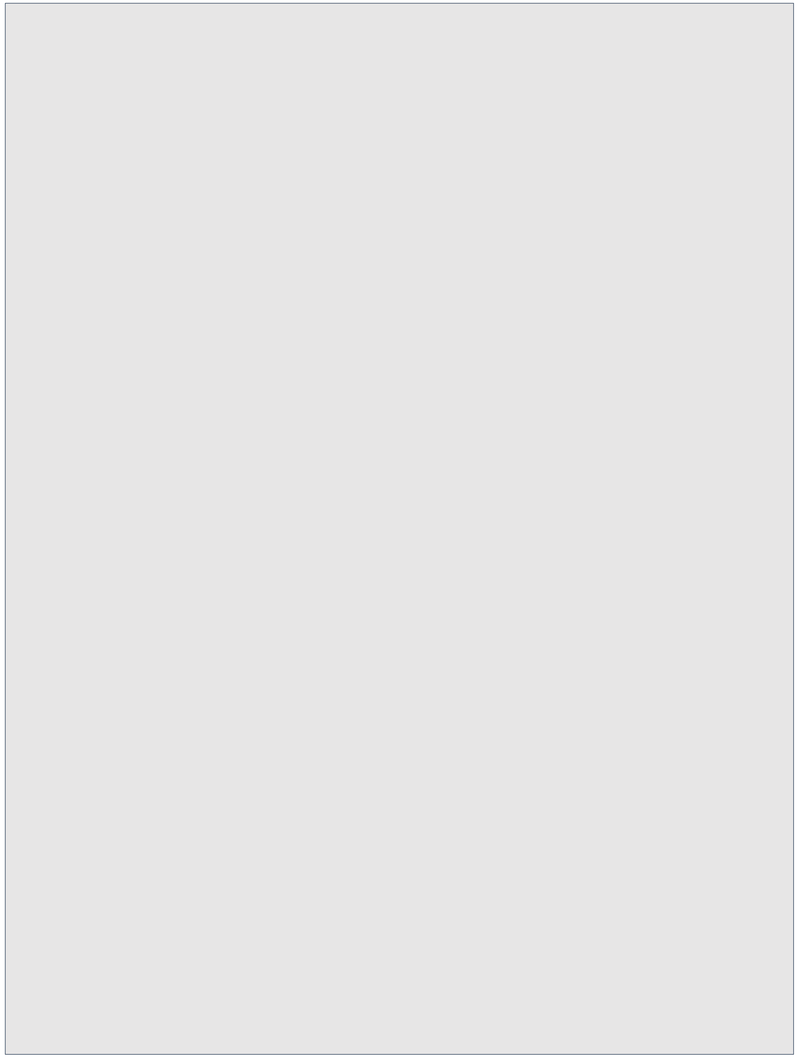
Who am I?

- Juraci Paixão Kröhling
- @jpkrohling (github|twitter)
- Jaeger maintainer
- OpenTelemetry Collector approver



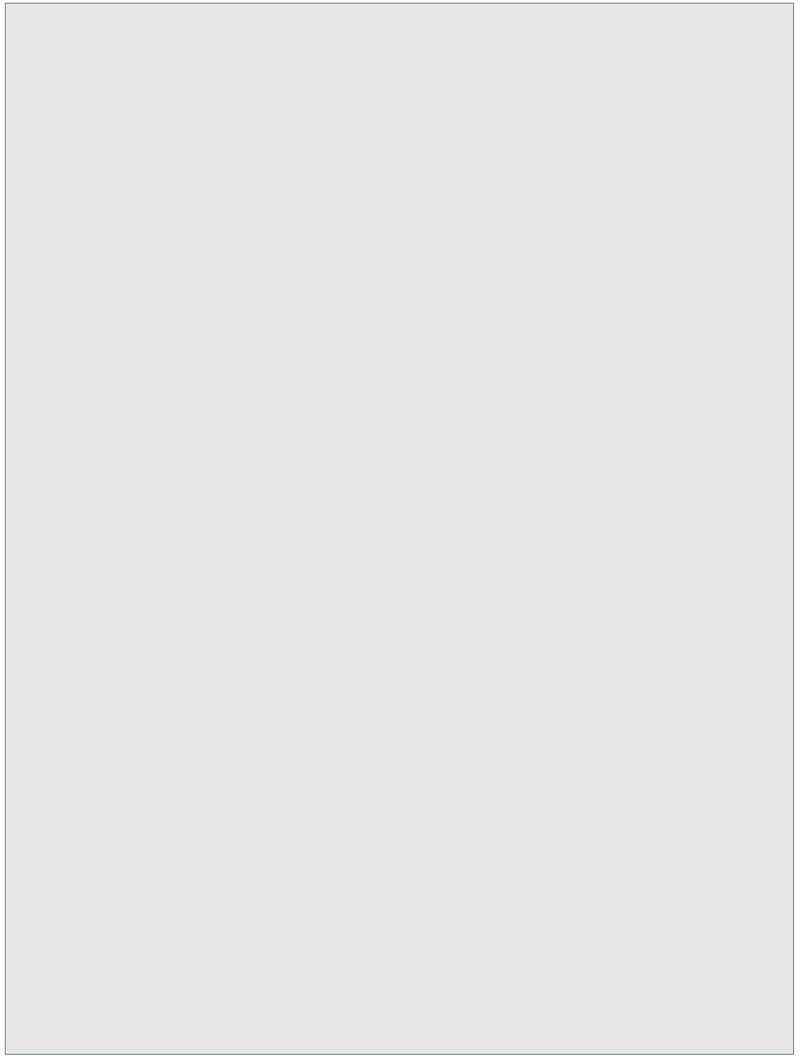
Agenda

- OpenTelemetry
- OpenTelemetry Collector
- Patterns!



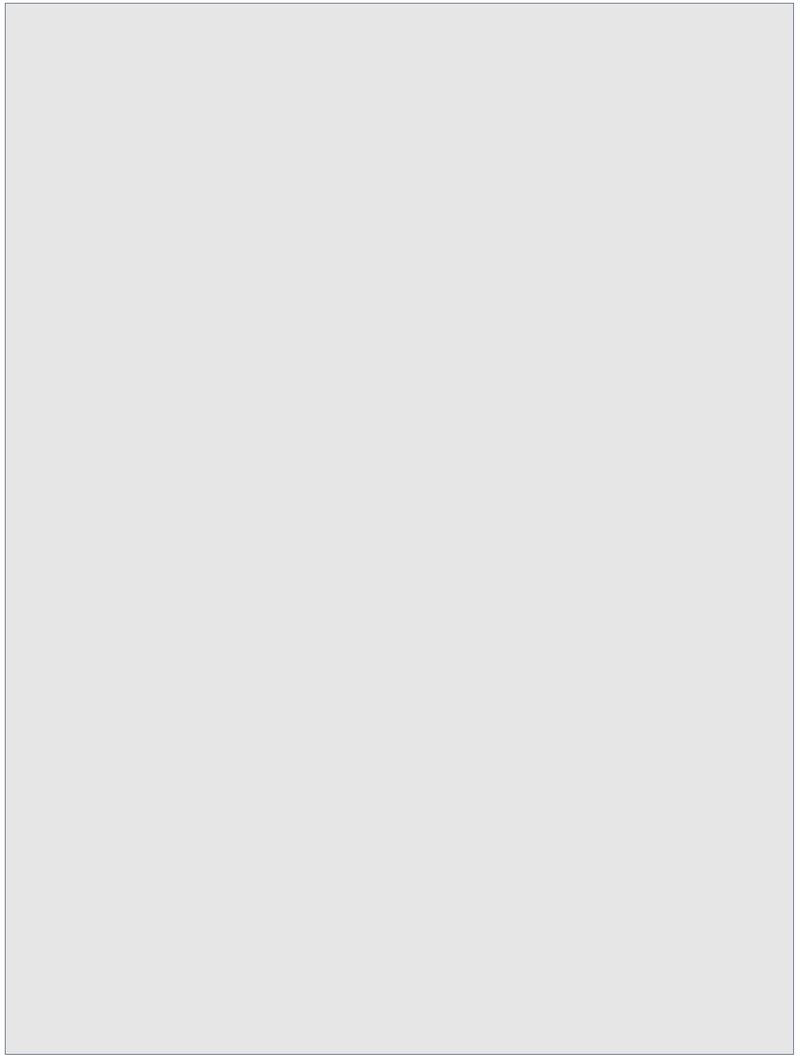
Patterns

- ➔ Pattern #1 - Basic I and II
- ➔ Pattern #2 - Normalizer
- ➔ Pattern #3 - On Kubernetes
- ➔ Pattern #4 - Load balancer
- ➔ Pattern #5 - Multi-cluster
- ➔ Pattern #6 - Multitenant



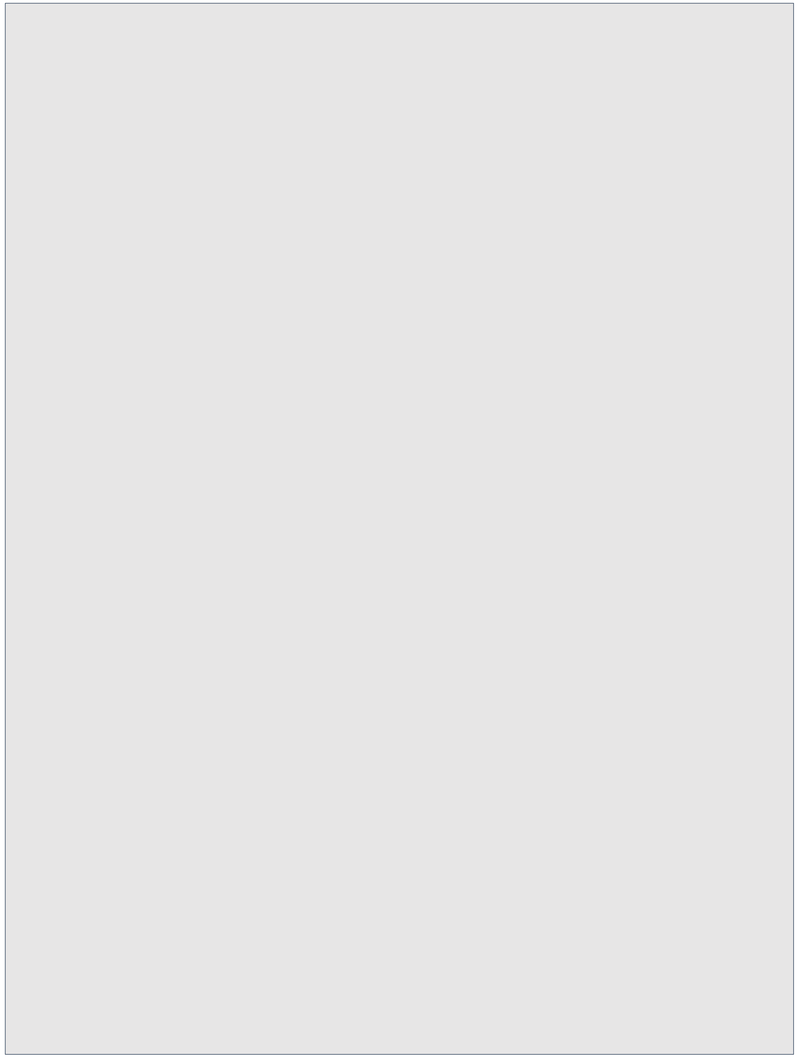
Patterns

➔ Take a look at the GitHub repository for detailed explanations and working examples.



OpenTelemetry

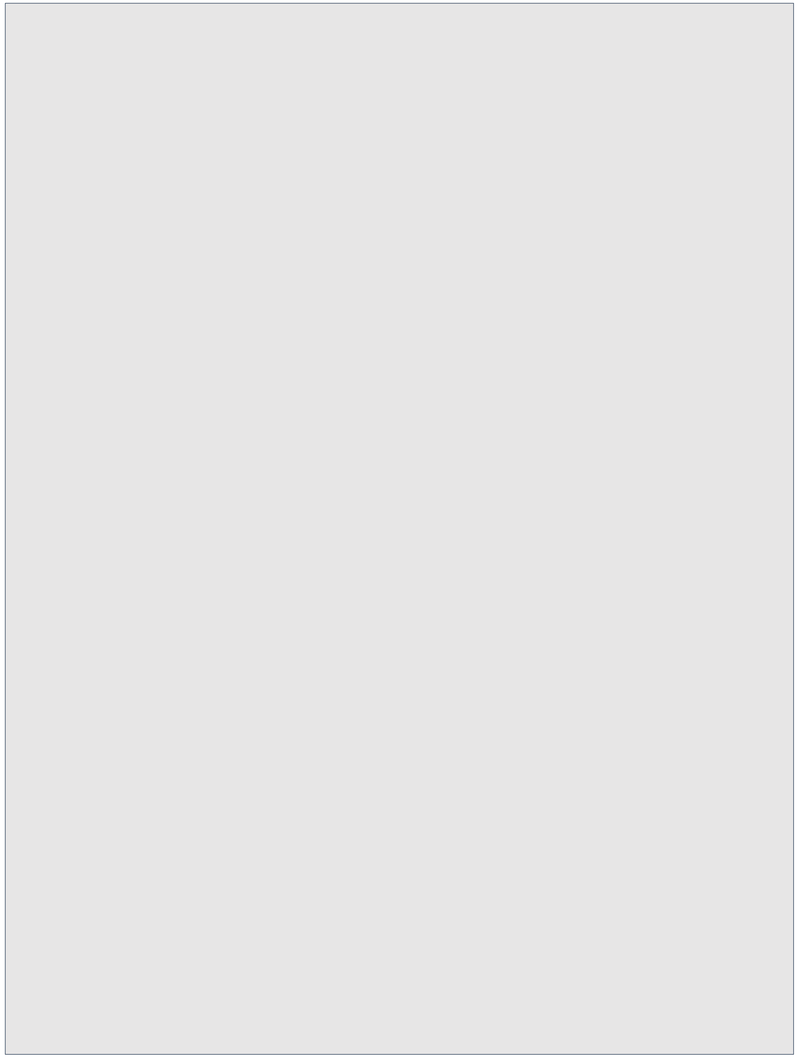
- Fusion of OpenTracing and OpenCensus
- Specifications and conventions
- Client APIs and SDKs
- OTLP (Line Protocol)
- Collector (or “otelcol”)



OpenTelemetry Collector

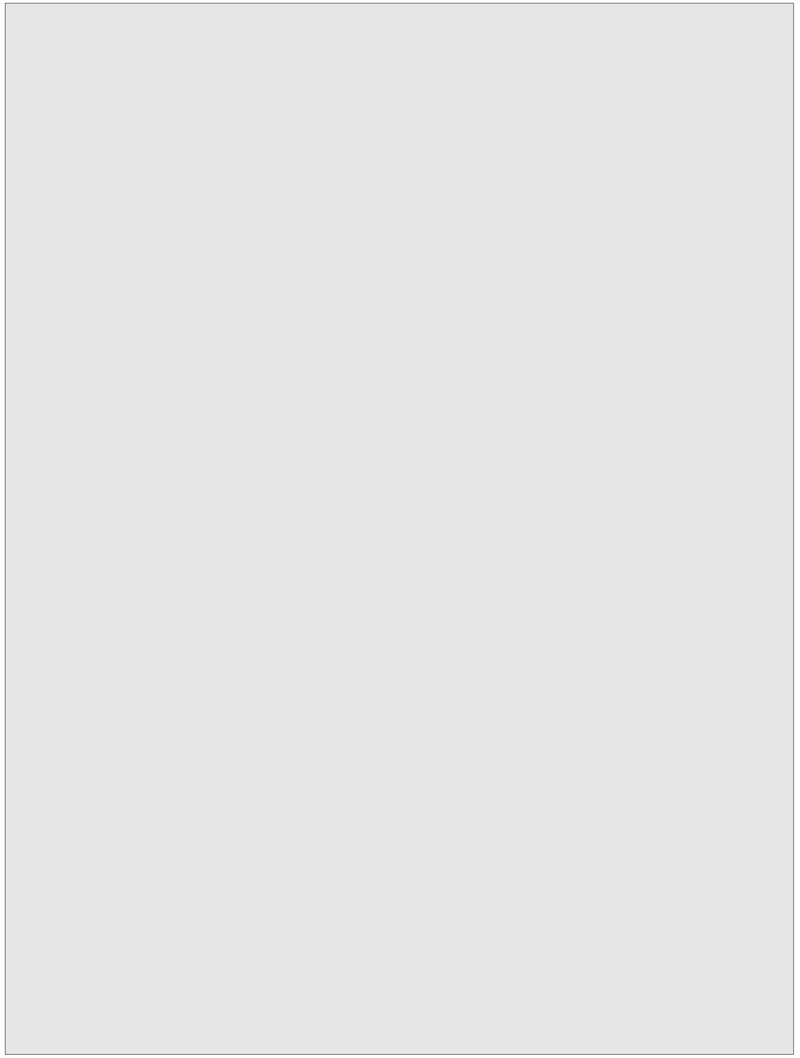
“Vendor-agnostic way to receive, process and export telemetry data.”

-- https://opentelemetry.io/docs/collector/

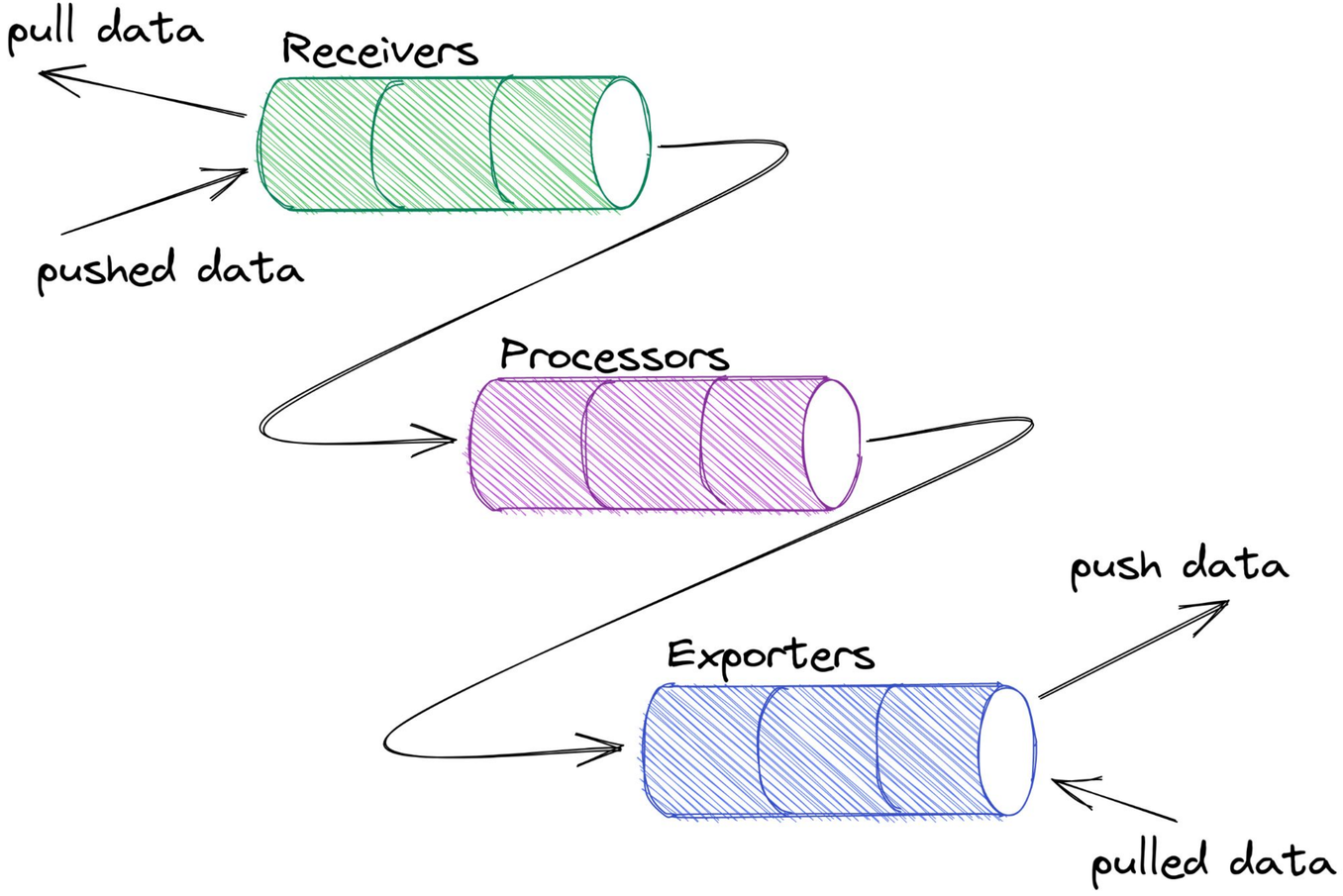


OpenTelemetry Collector

- Receivers
- Processors
- Exporters
- Extensions

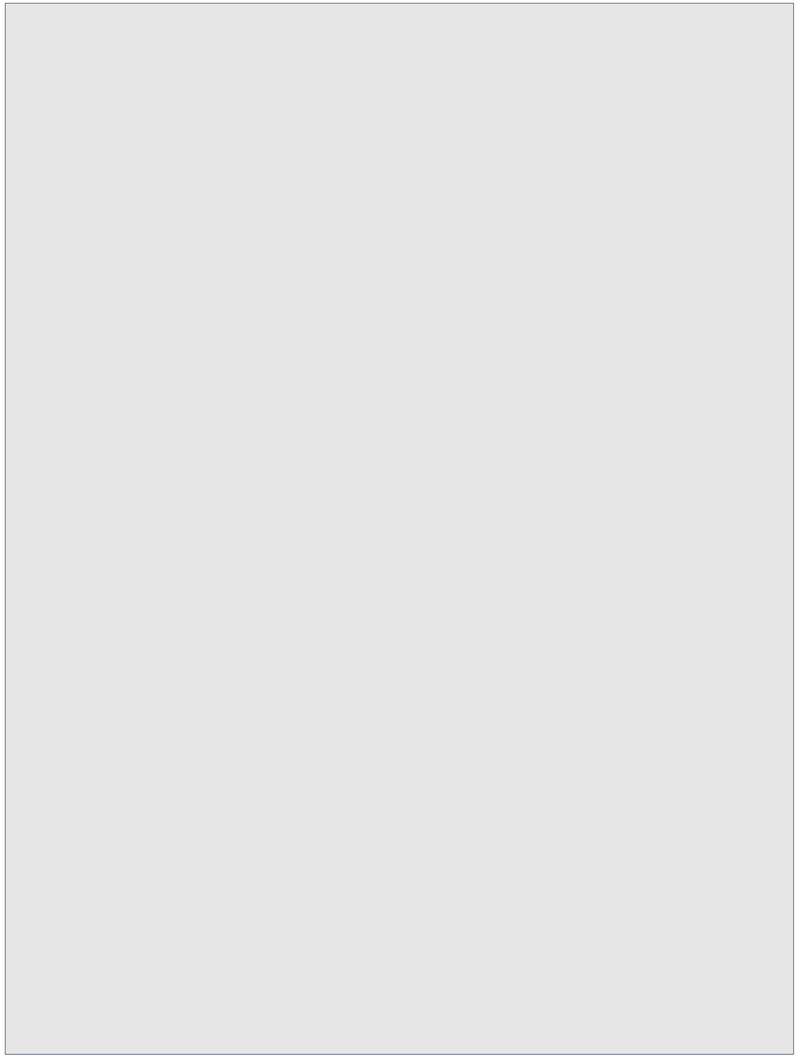


OpenTelemetry Collector



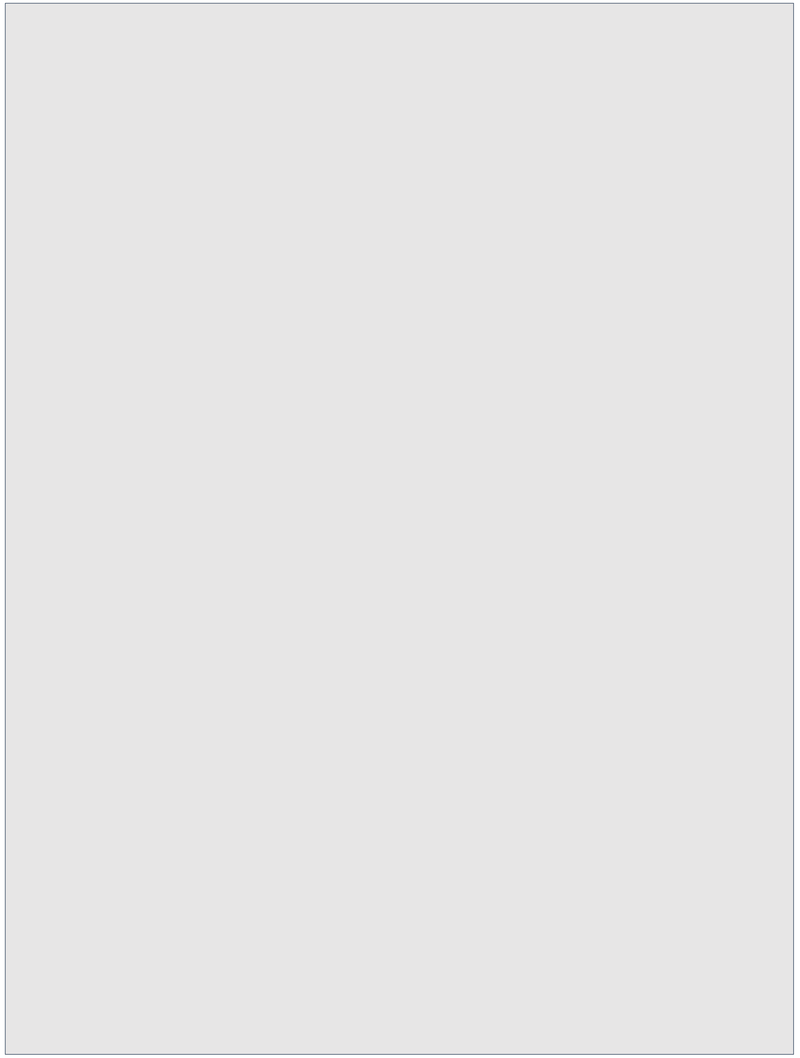
OpenTelemetry Collector

- Receivers
  - Jaeger
  - Prometheus
  - OTLP
  - Zipkin
  - ...



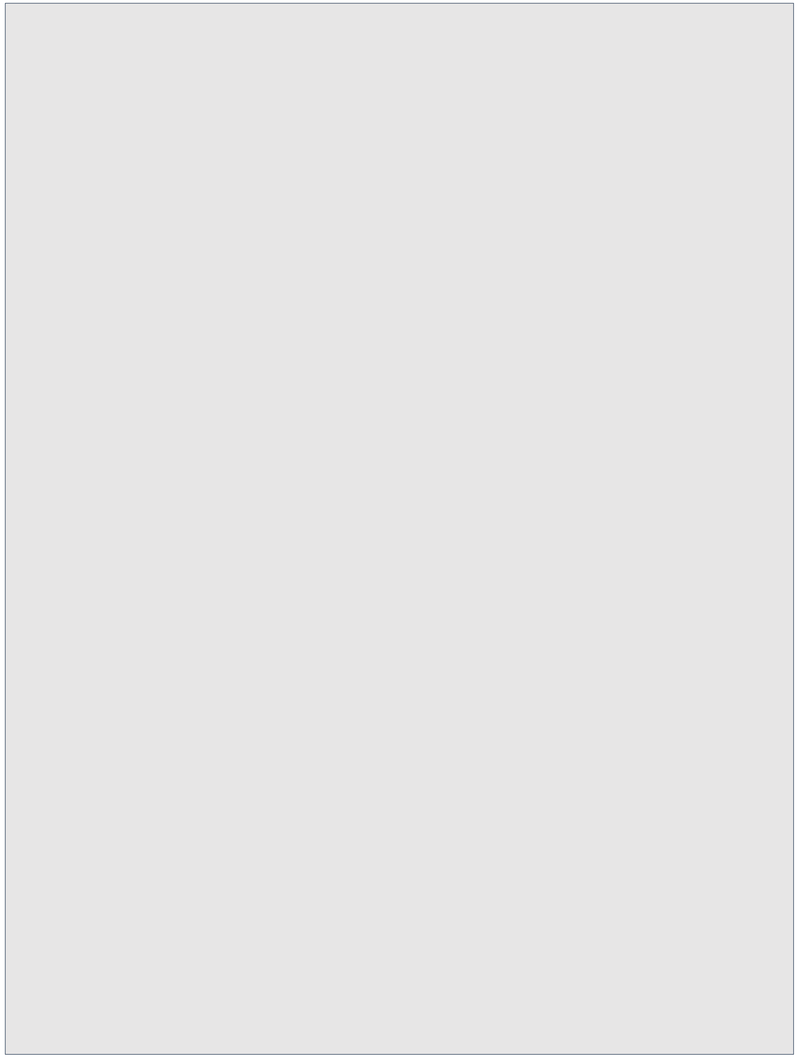
OpenTelemetry Collector

- Processors
  - Sampling
  - Change attributes
  - Batching
  - ...



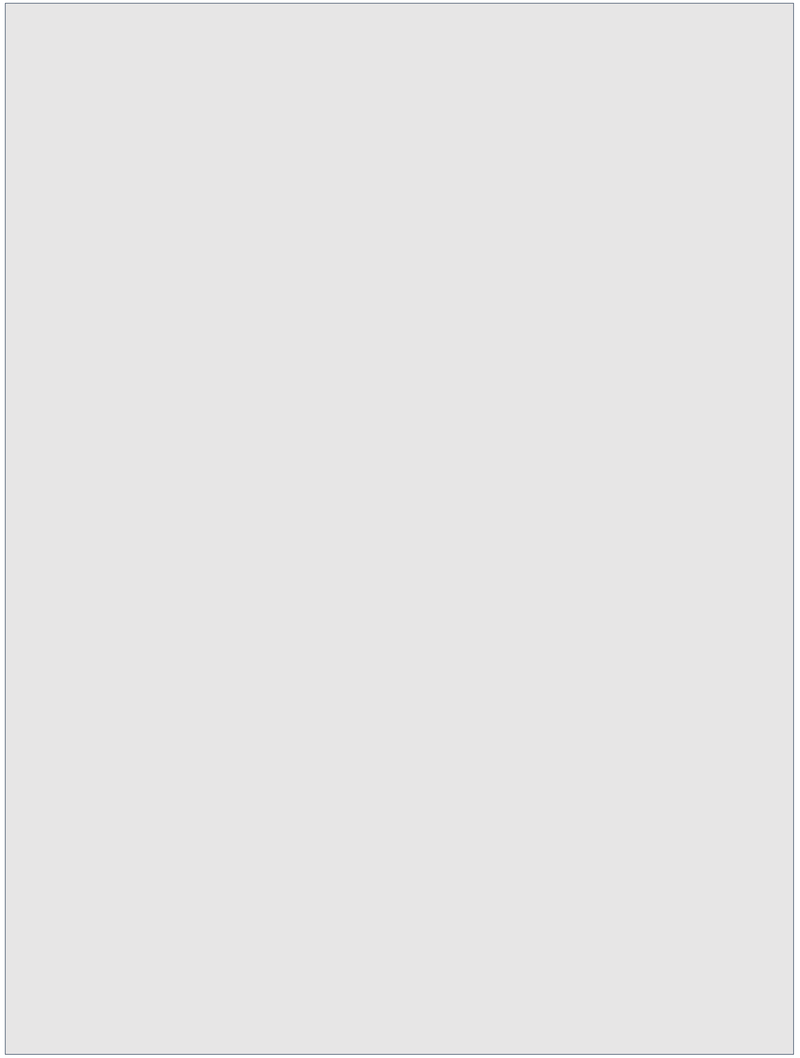
OpenTelemetry Collector

- → Exporters
  - Jaeger, Zipkin, ...
  - Prometheus
  - OTLP
  - Pretty much all commercial vendors...



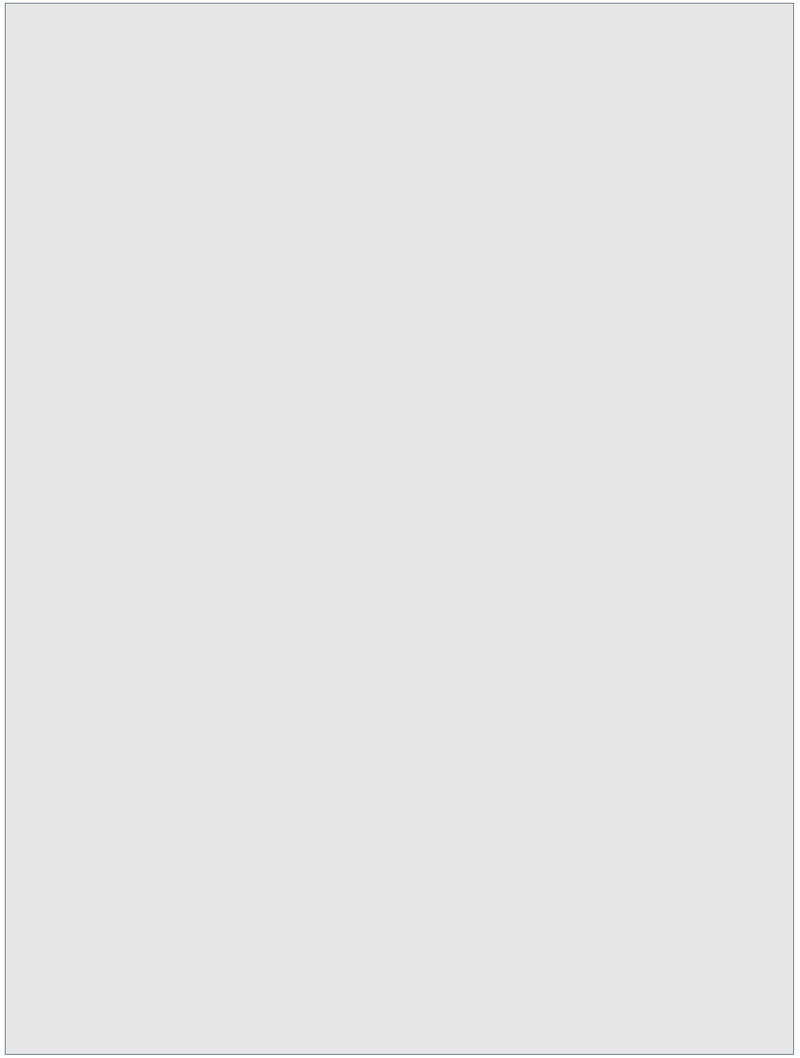
OpenTelemetry Collector

- ➔ Contrib is where all non-core components live, including vendor-specific ones
- ➔ Builder allows you to create your own distribution



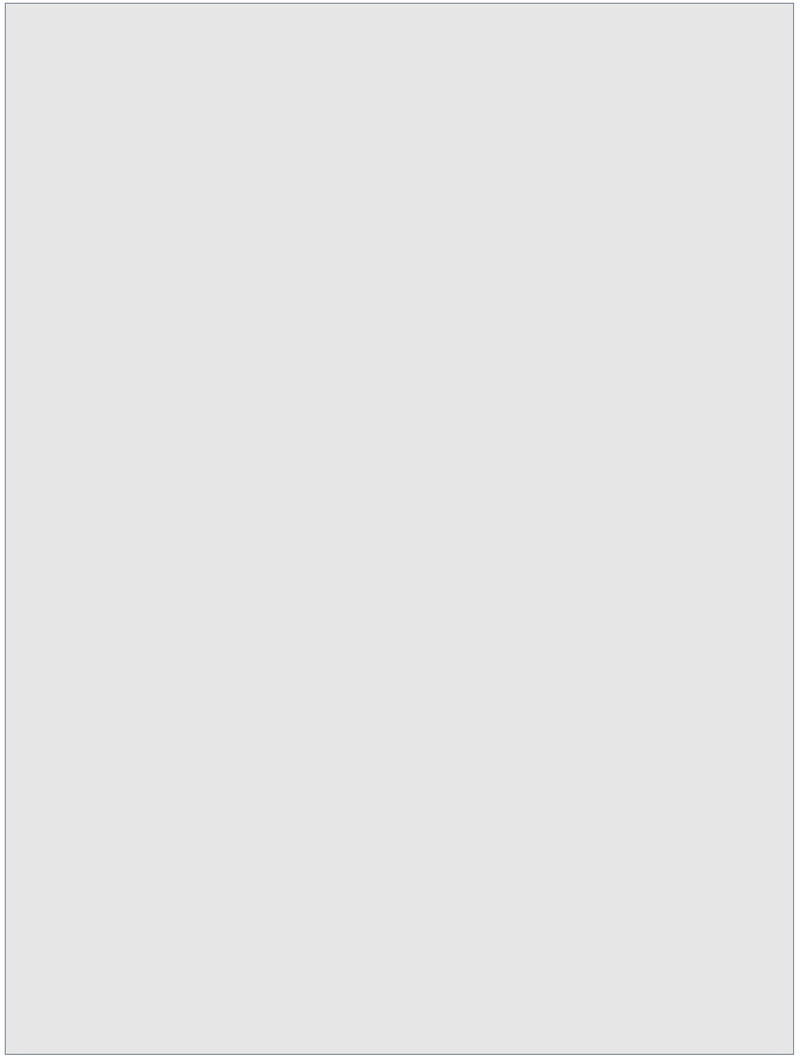
OpenTelemetry Collector


```
$ cat opentelemetry-collector-config.yaml
extensions:
 health_check:

receivers:
 otlp:
   protocols:
     grpc:

processors:
 batch:

exporters:
 logging:

service:
 extensions: [health_check]
 pipelines:
   traces:
     receivers: [otlp]
     processors: [batch]
     exporters: [logging]
```

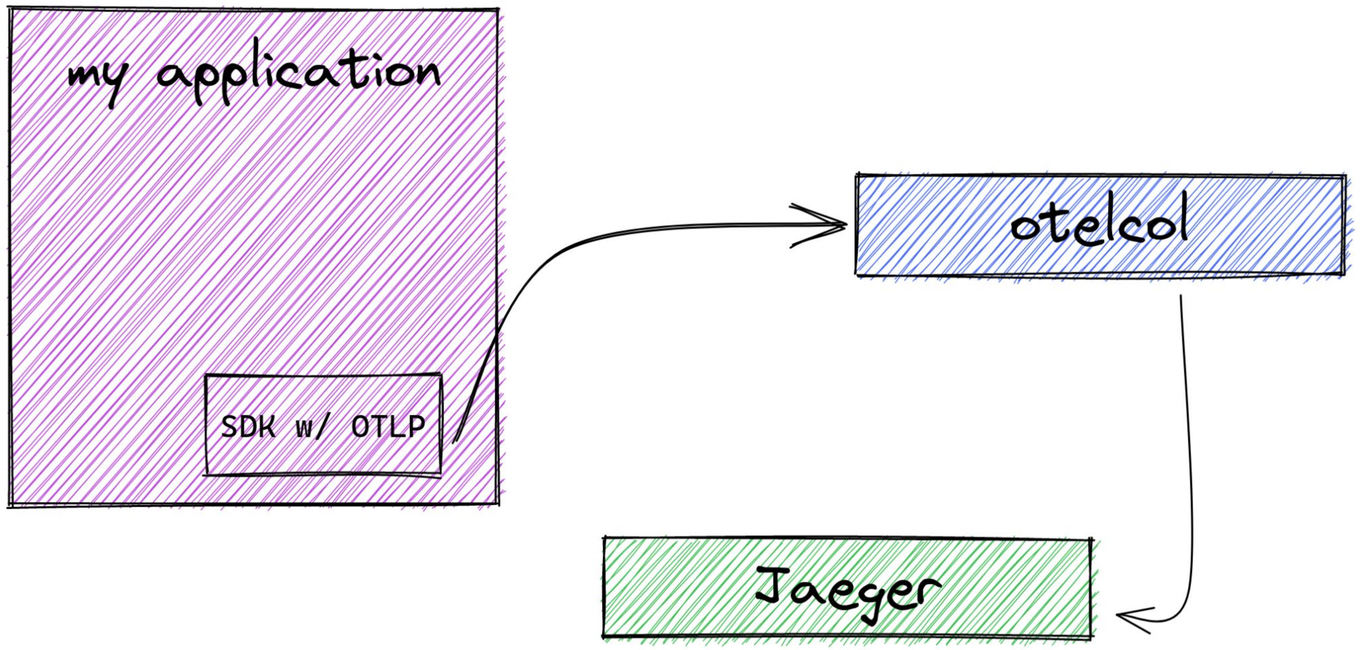


Pattern #1 - Basic I

- ➔ OpenTelemetry SDK with OTLP
- ➔ OpenTelemetry Collector
- ➔ Jaeger

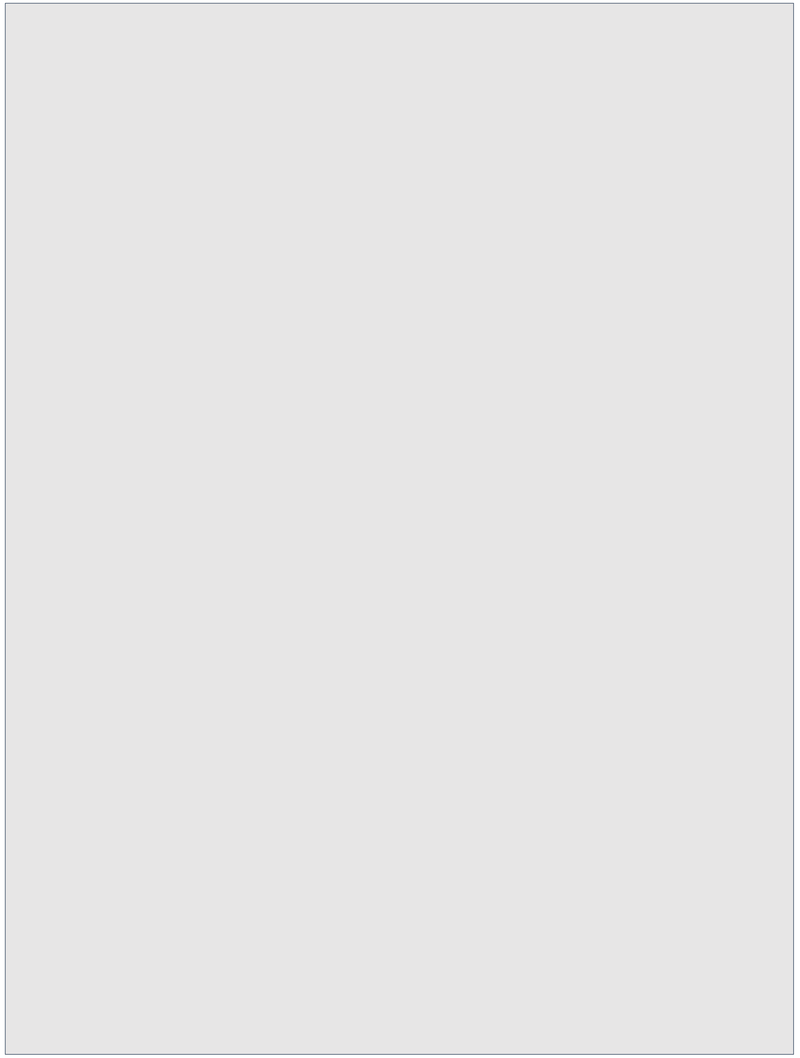


Pattern #1 - Basic I

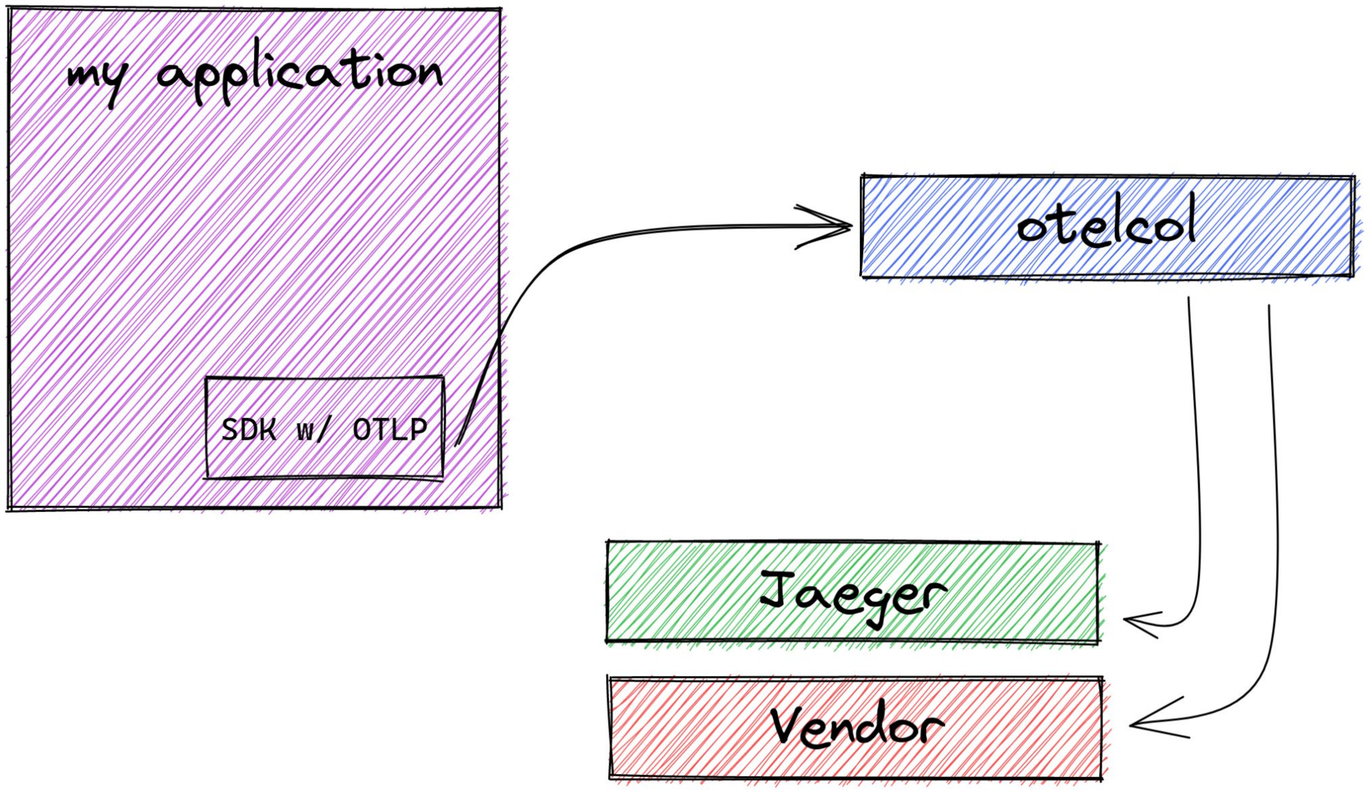


Pattern #1 - Basic II - Fanout

- ➔ OpenTelemetry SDK with OTLP
- ➔ OpenTelemetry Collector
- ➔ Jaeger
- ➔ Some vendor

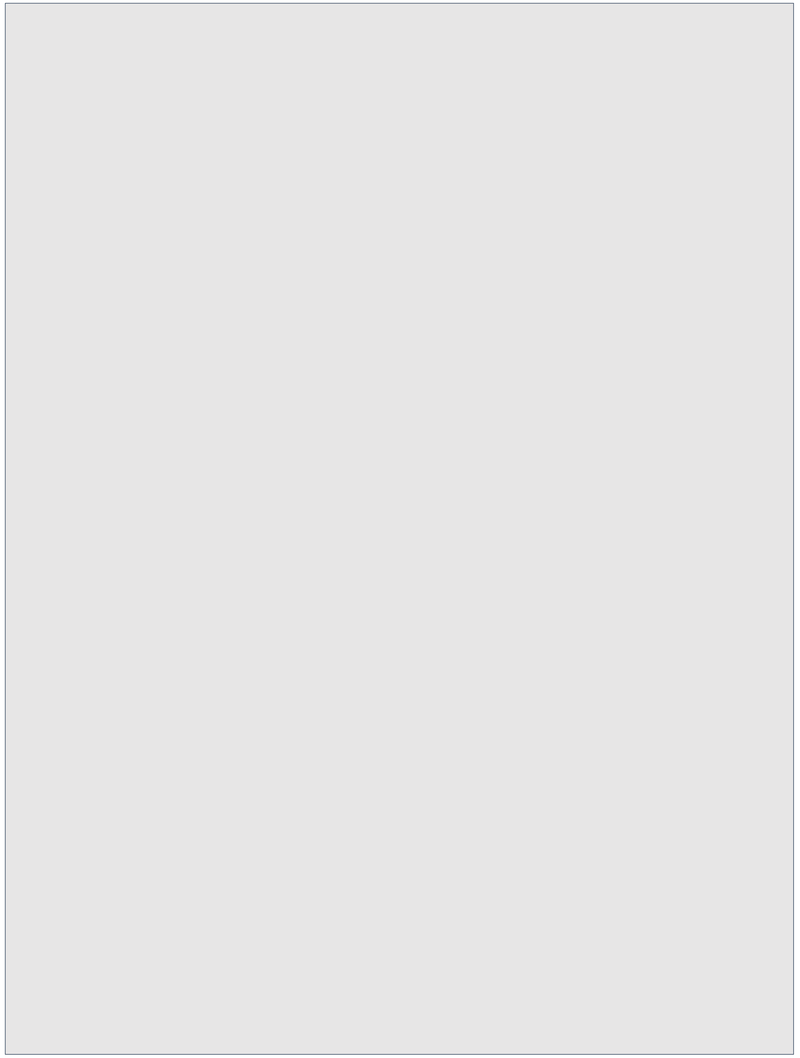


Pattern #1 - Basic II - Fanout

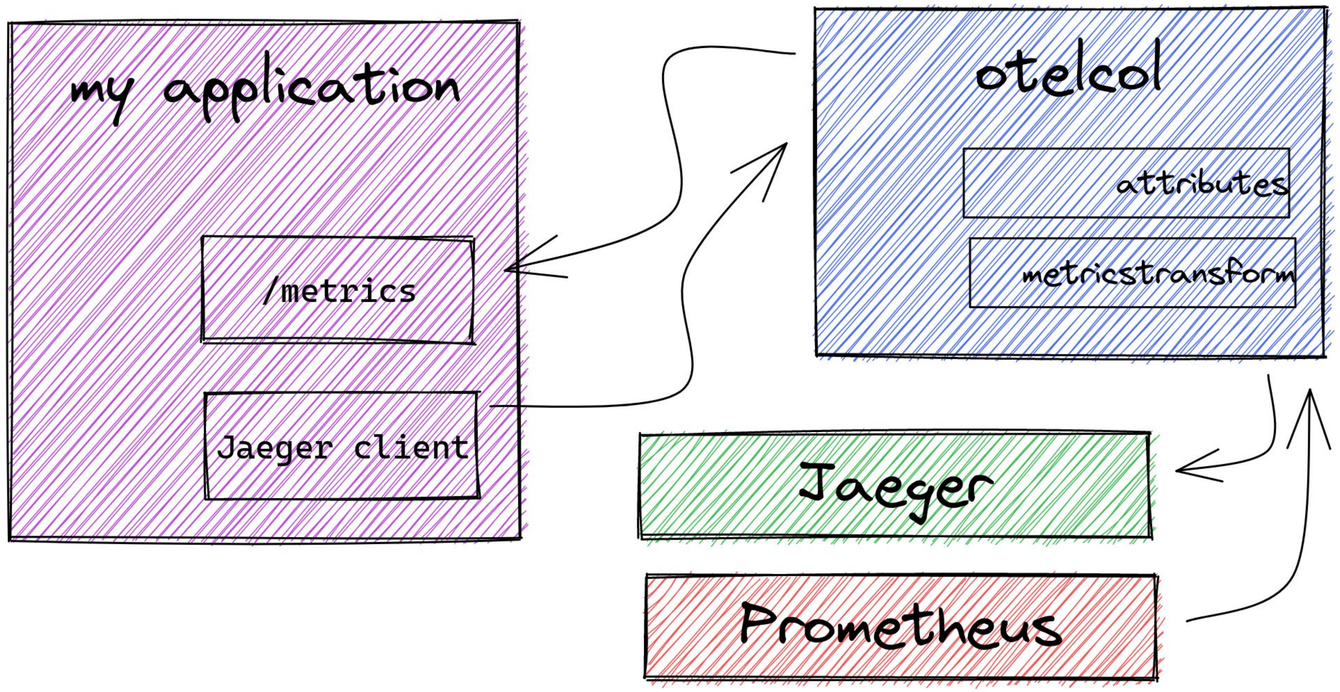


Pattern #2 - Normalizer

- ➔ Prometheus client
- ➔ Jaeger client
- ➔ Collector with attributes processor
- ➔ Jaeger
- ➔ Prometheus

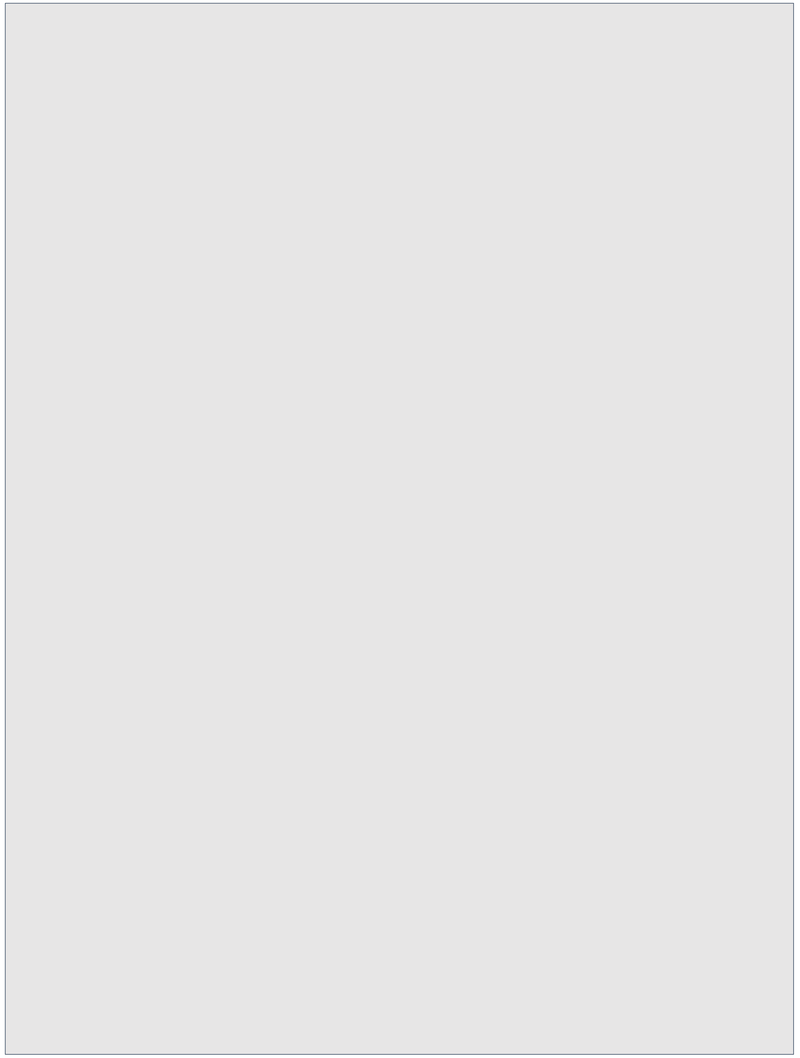


Pattern #2 - Normalizer

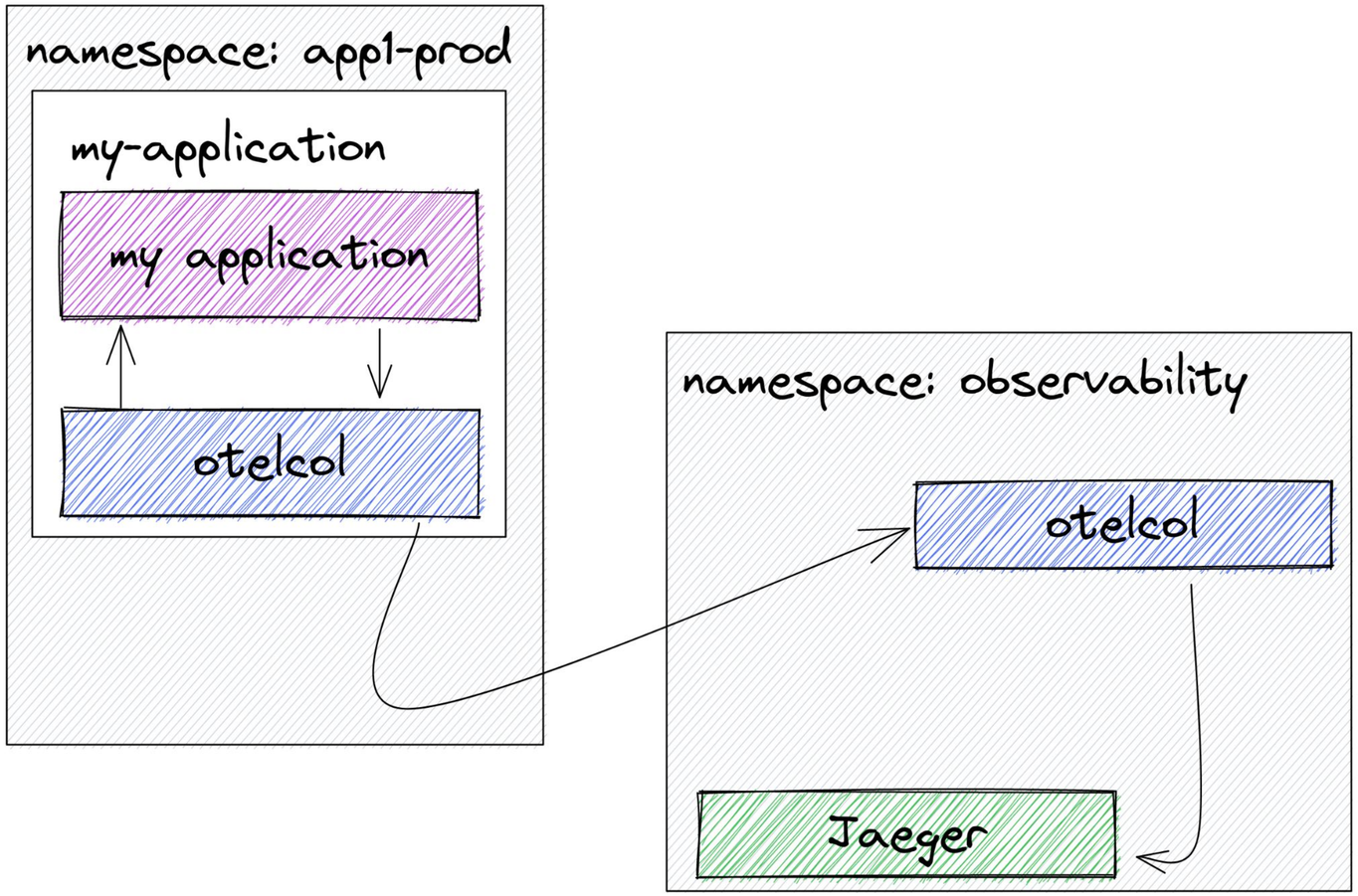


Pattern #3 - On Kubernetes I

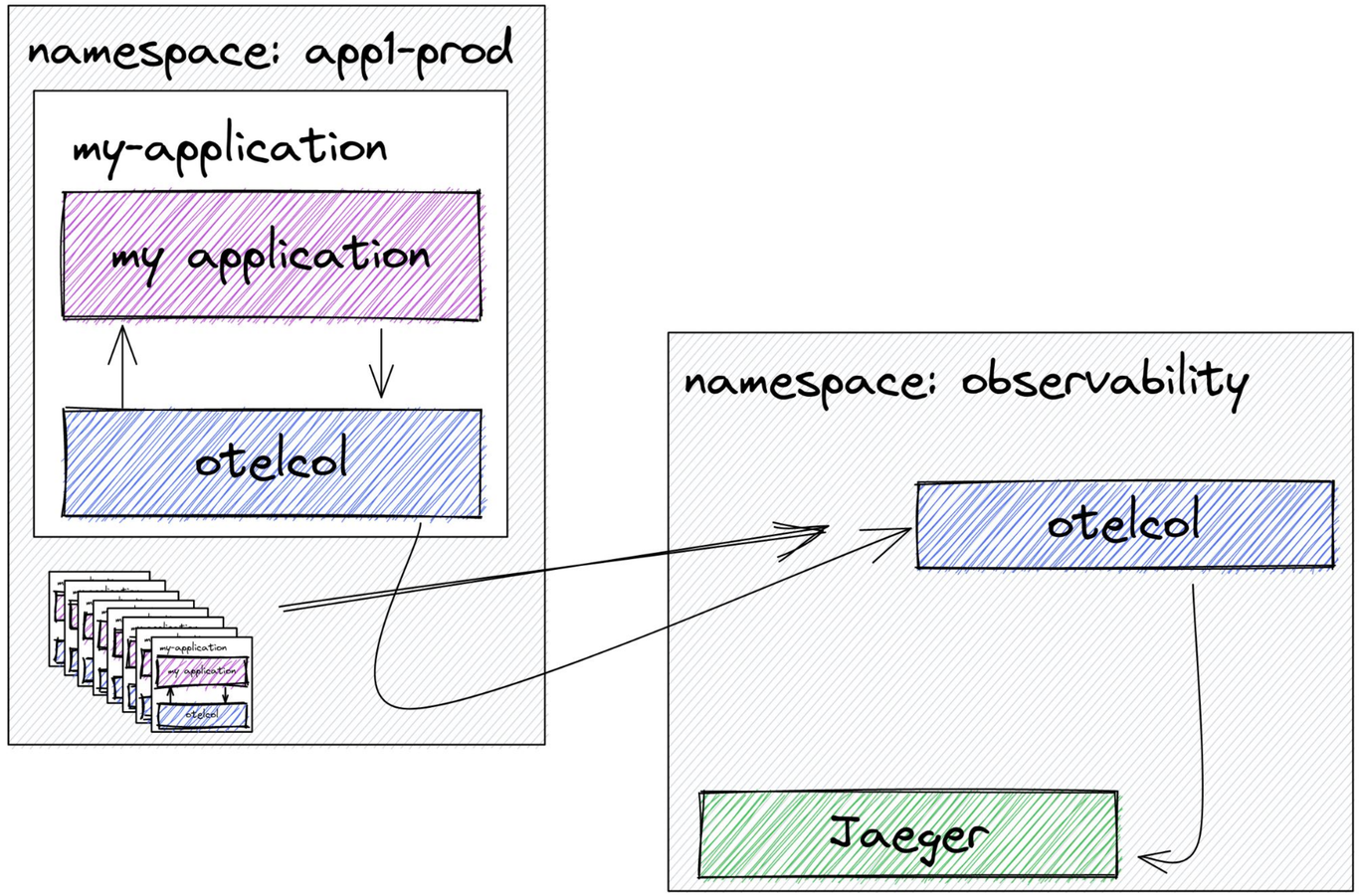
- ➔ Workload pod with collector sidecar
- ➔ Collector
- ➔ Backends



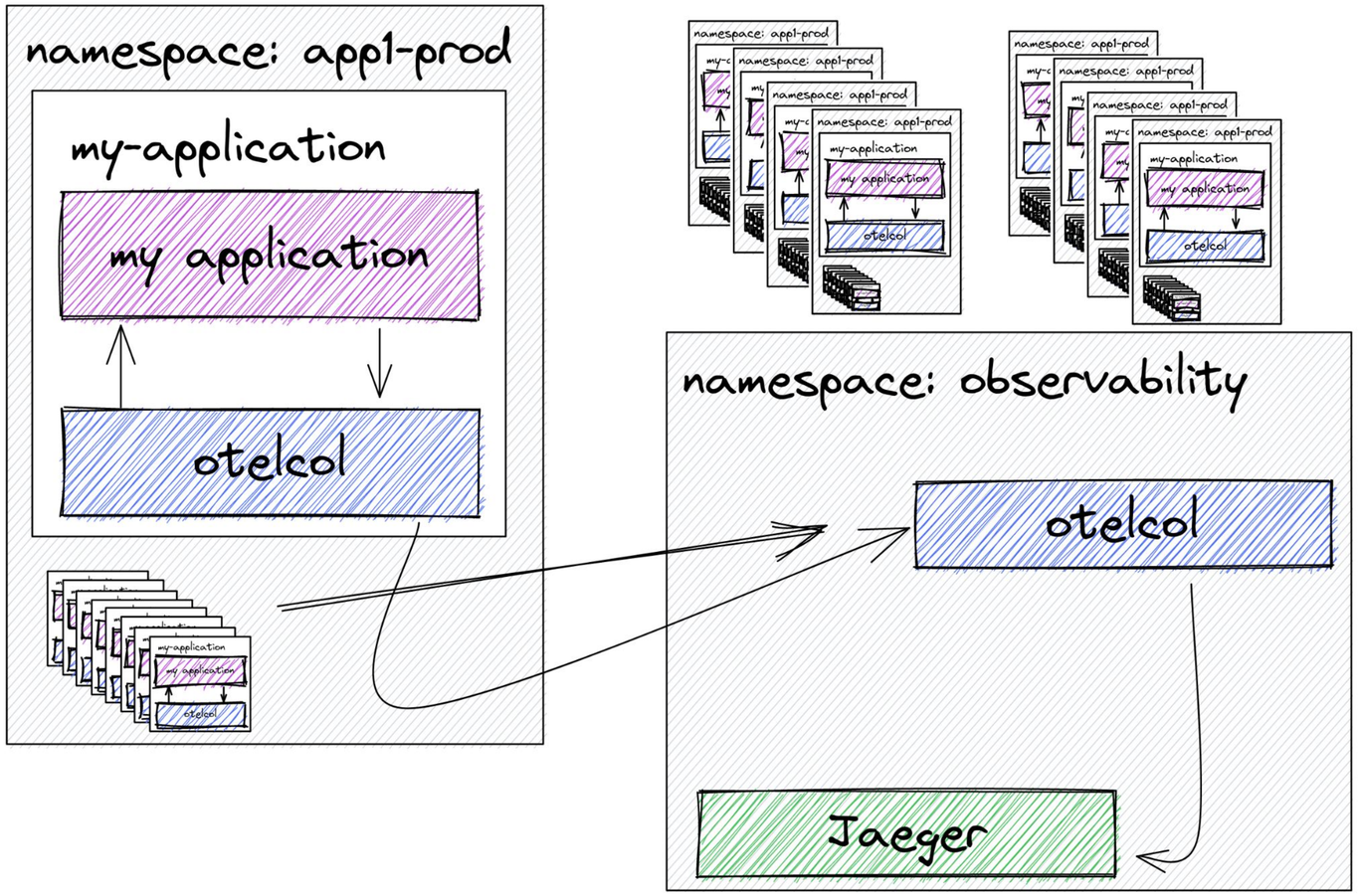
Pattern #3 - On Kubernetes I



Pattern #3 - On Kubernetes I

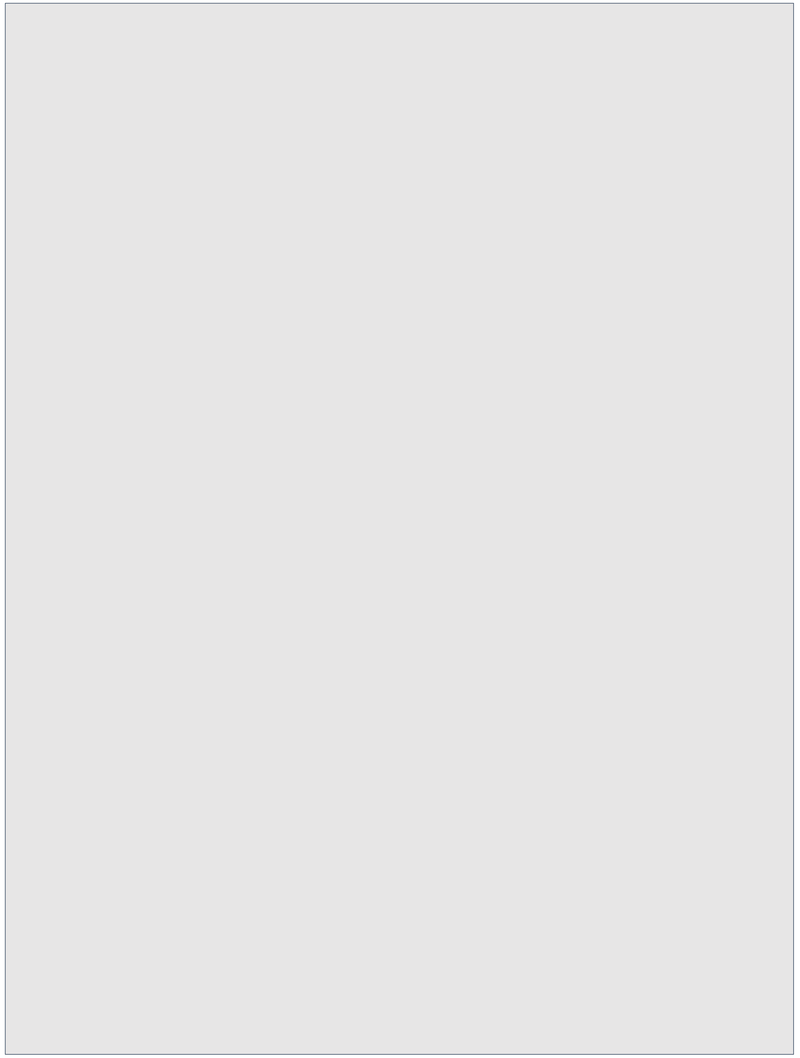


Pattern #3 - On Kubernetes I

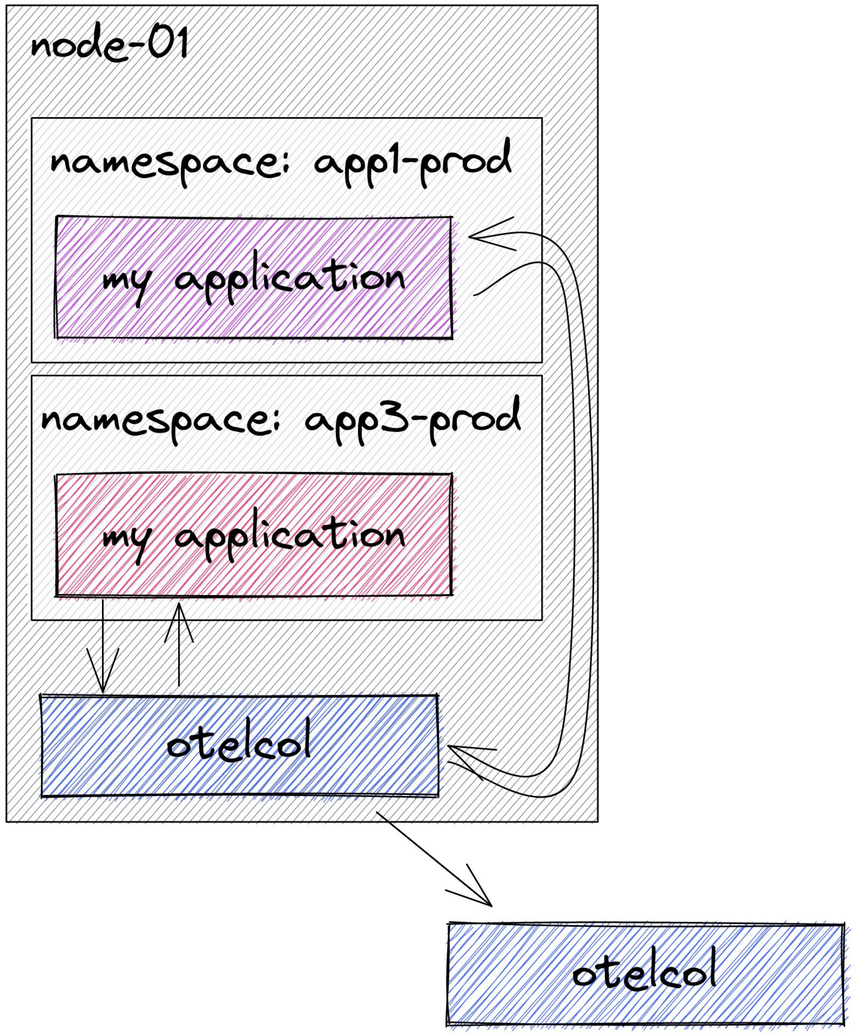


Pattern #3 - On Kubernetes II

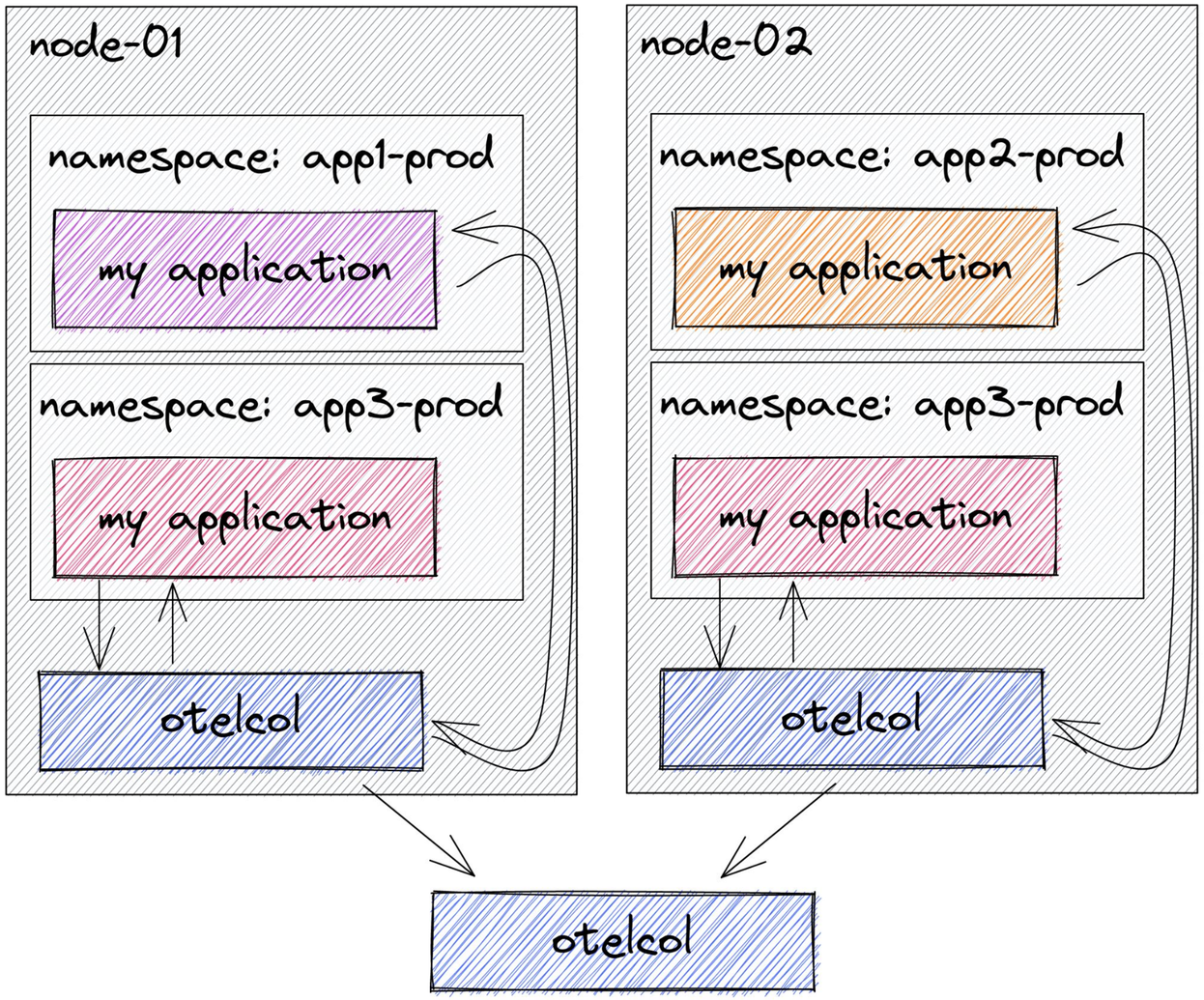
- ➔ Collector as DaemonSet
- ➔ Collector
- ➔ Backends



Pattern #3 - On Kubernetes II

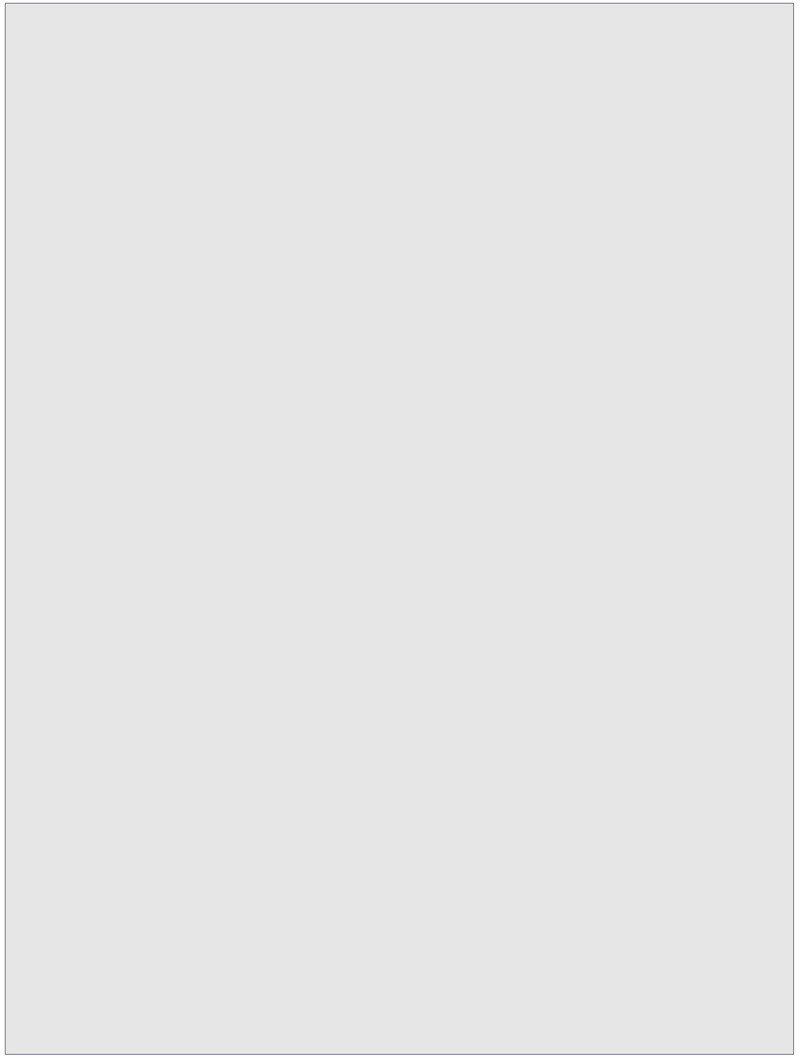


Pattern #3 - On Kubernetes II

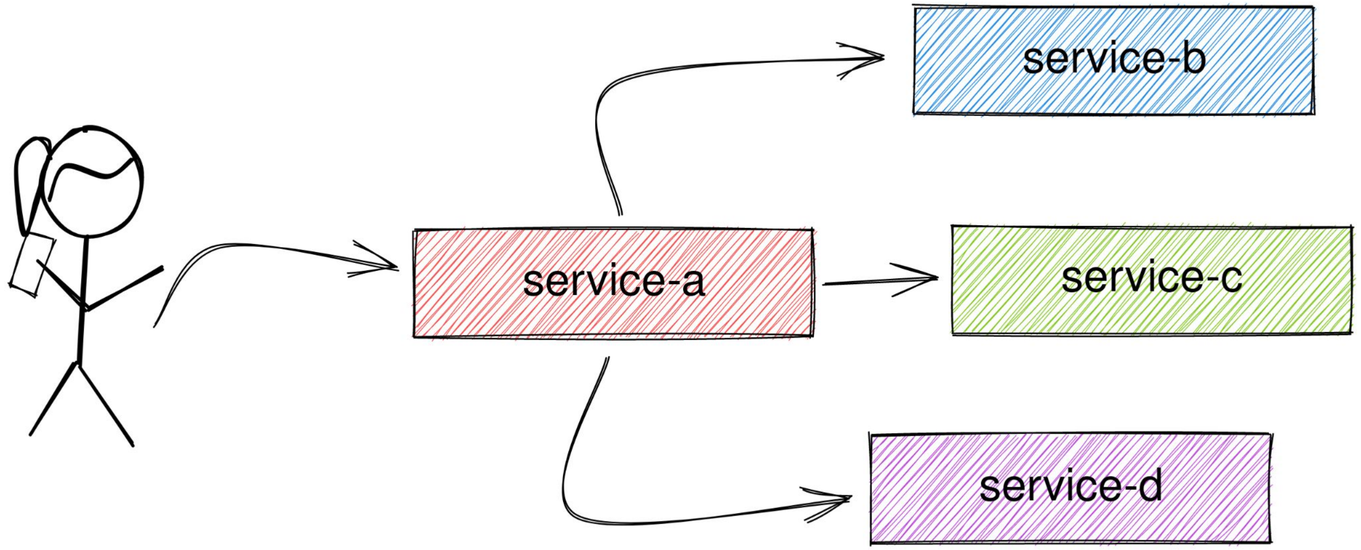


Pattern #4 - Load balancer

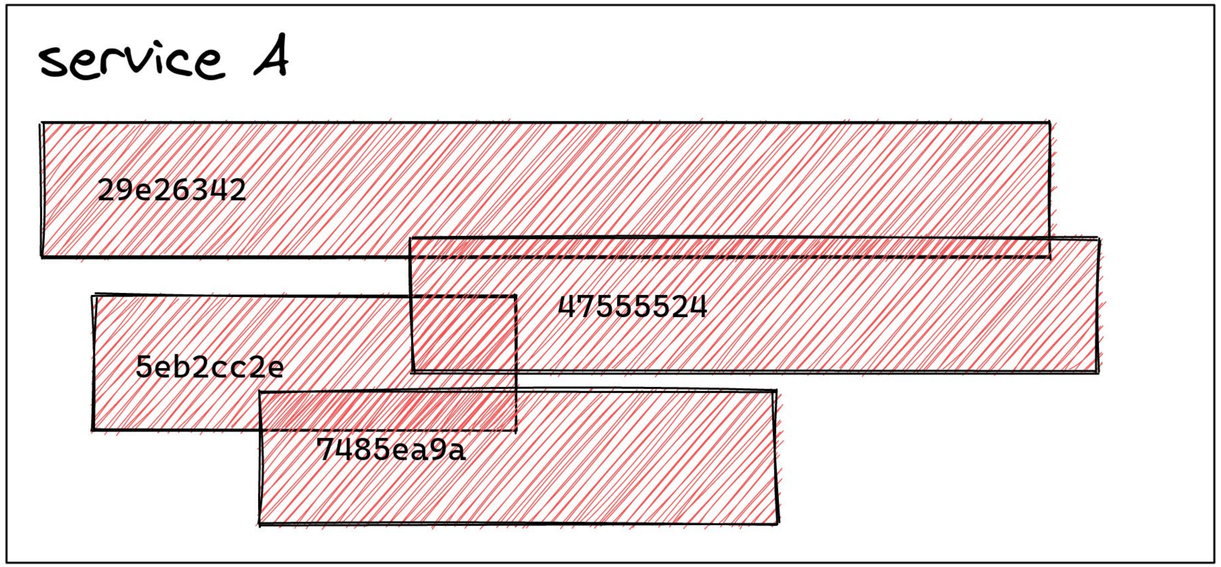
- ➔ Multiple services sending data
- ➔ Trace ID aware load balancer
- ➔ Scalable tail-based sampling!
- ➔ Backends



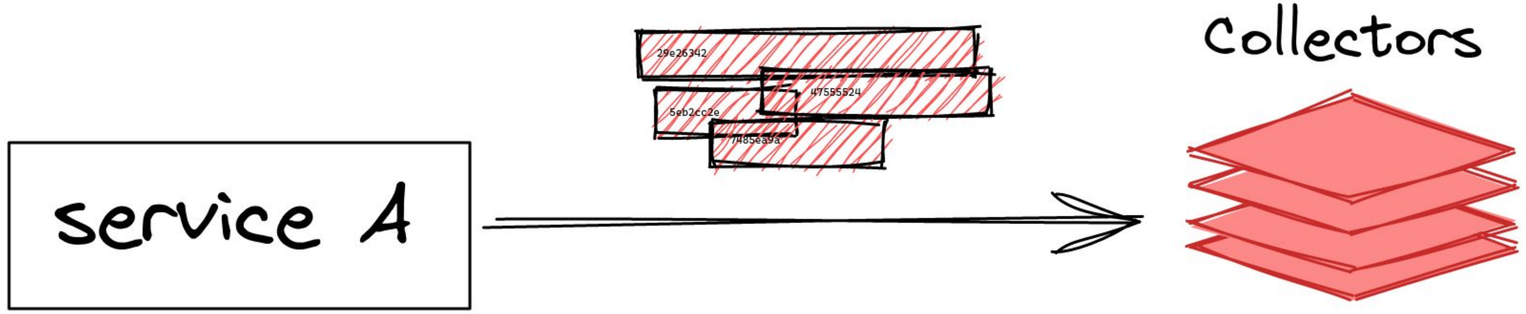
Pattern #4 - Load balancer



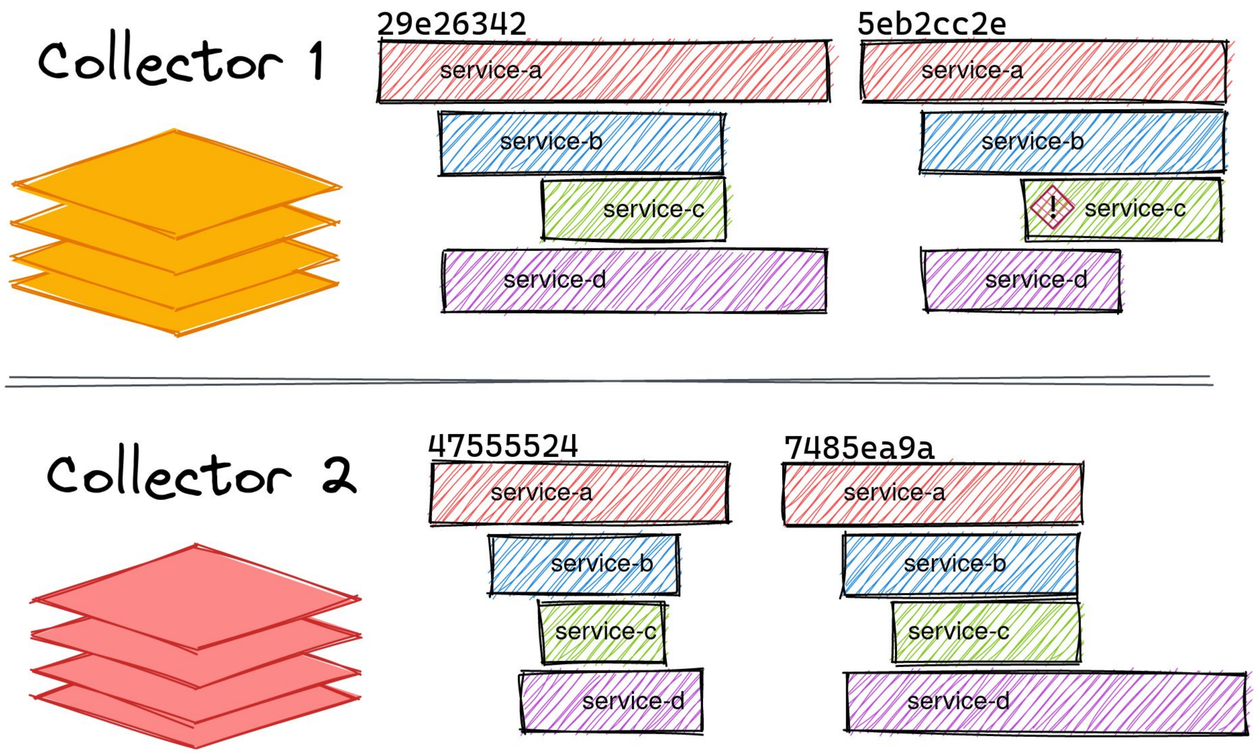
Pattern #4 - Load balancer



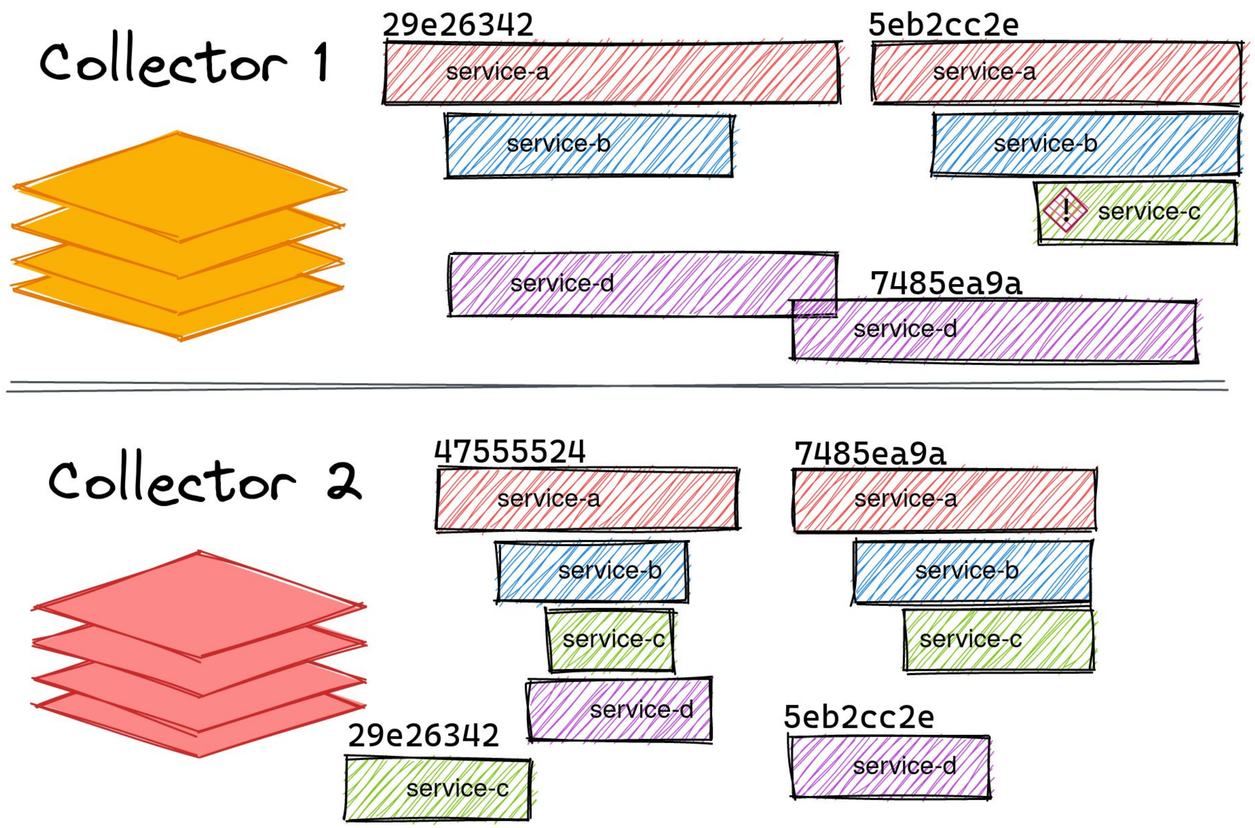
Pattern #4 - Load balancer



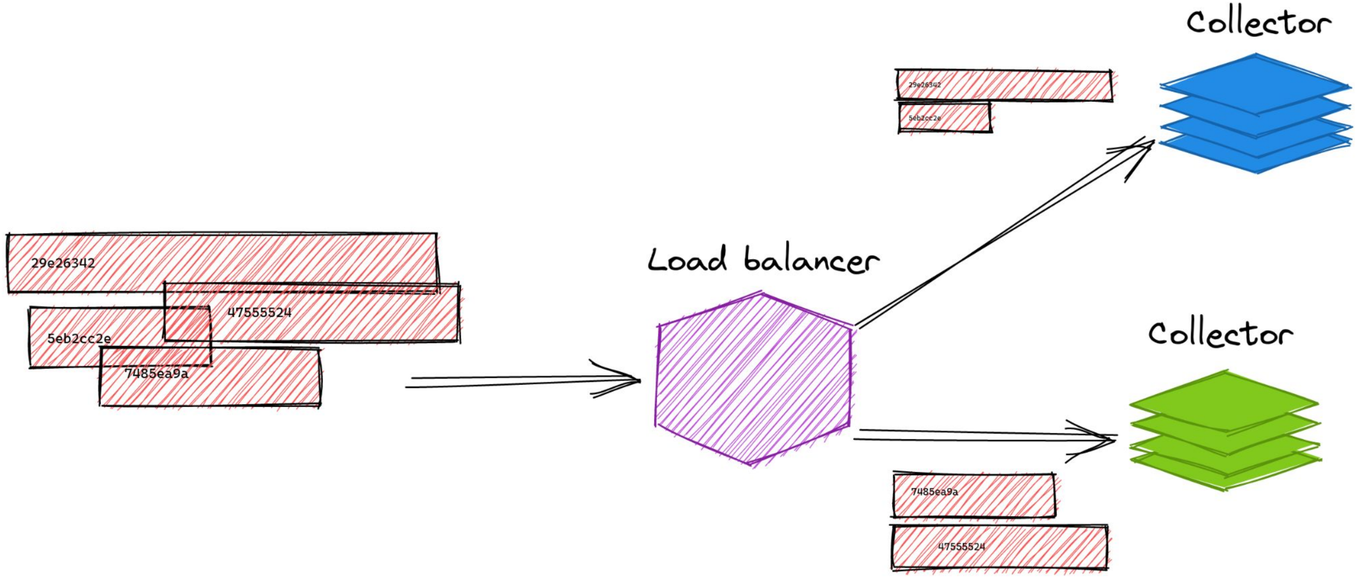
Pattern #4 - Load balancer



Pattern #4 - Load balancer

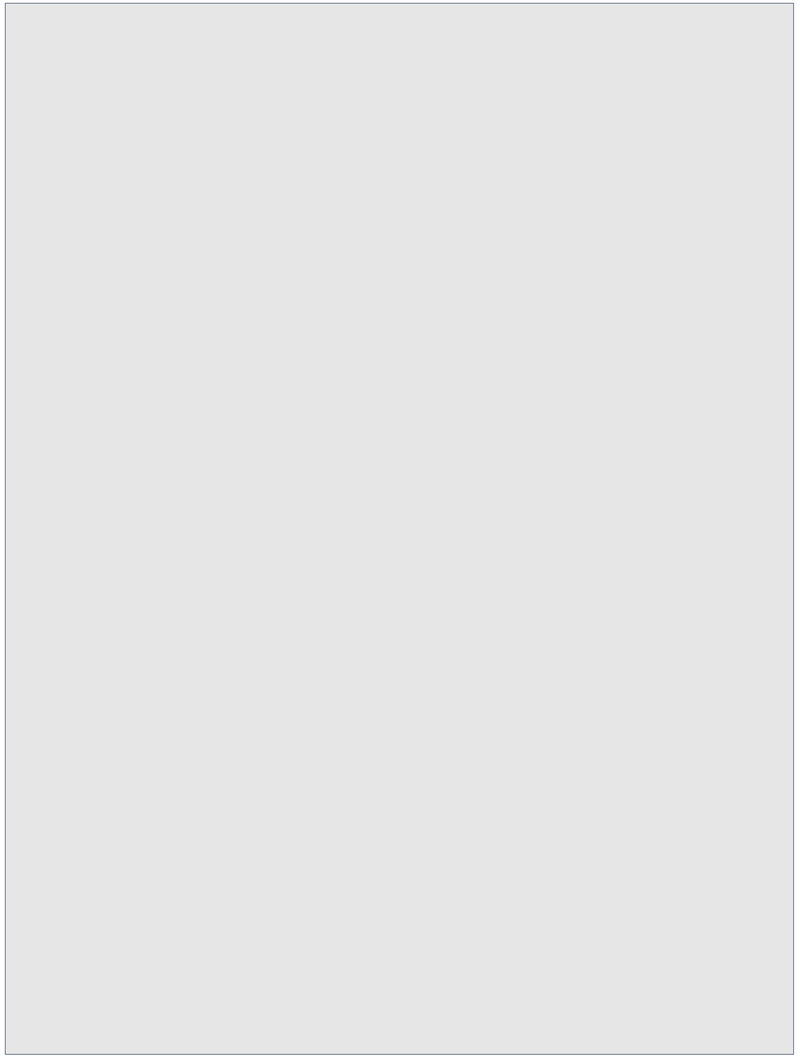


Pattern #4 - Load balancer

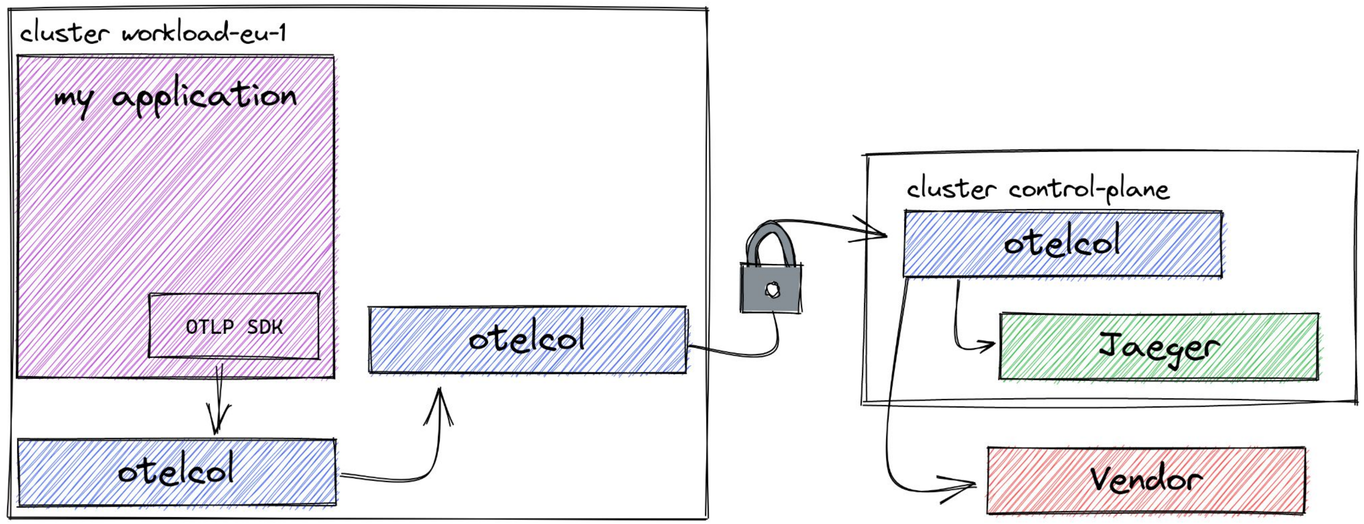


Pattern #5 - Multi-cluster

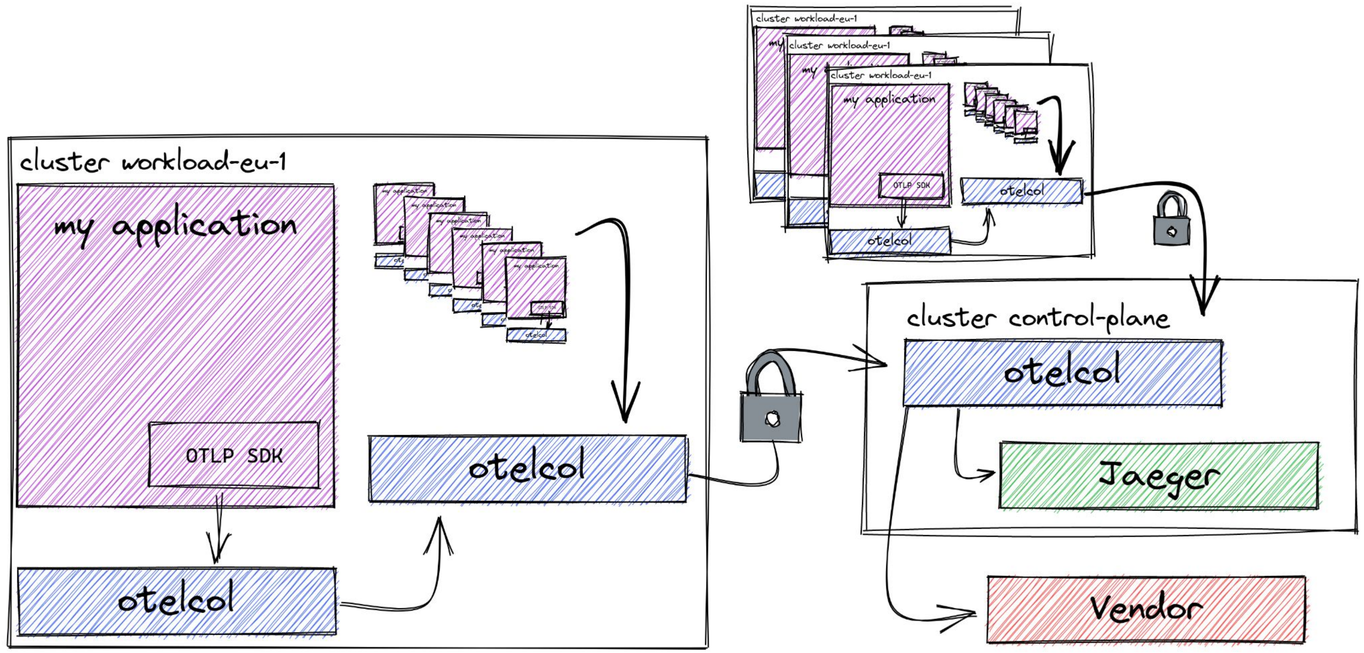
- ➔ Collector as agent
- ➔ Collector in the workload cluster
- ➔ Collector in the control-plane cluster
- ➔ Backends possibly elsewhere



Pattern #5 - Multi-cluster

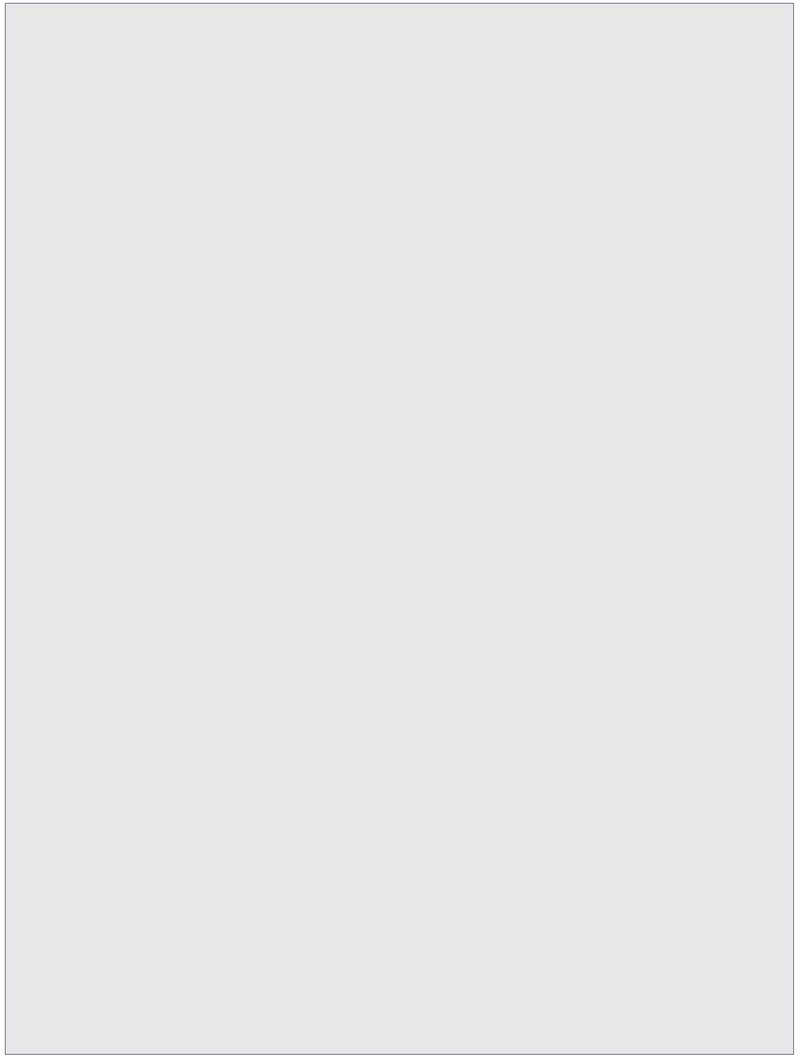


Pattern #5 - Multi-cluster

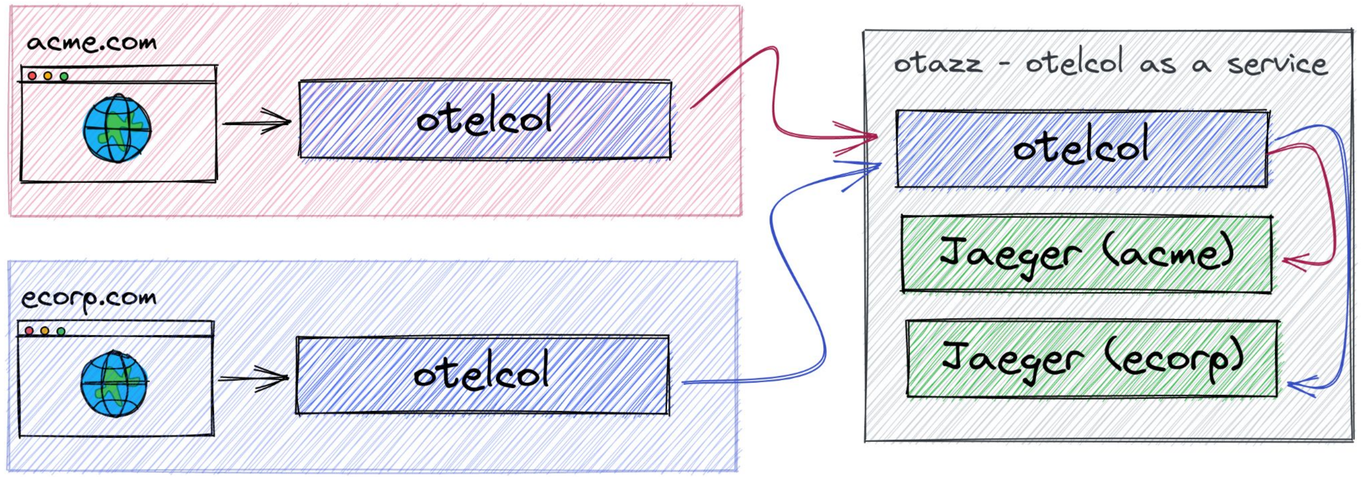


Pattern #6 - Multitenant

- ➔ Incoming data from different tenants
- ➔ Collector with the routing processor
- ➔ Multiple Jaeger instances

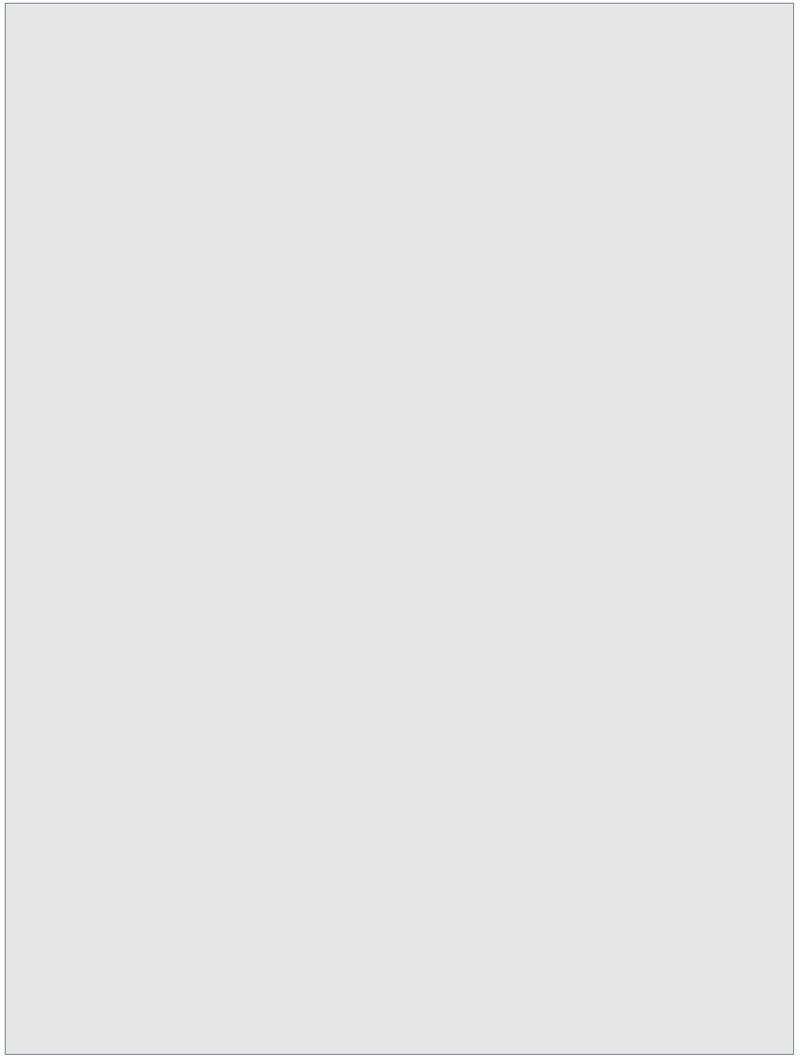


Pattern #6 - Multitenant

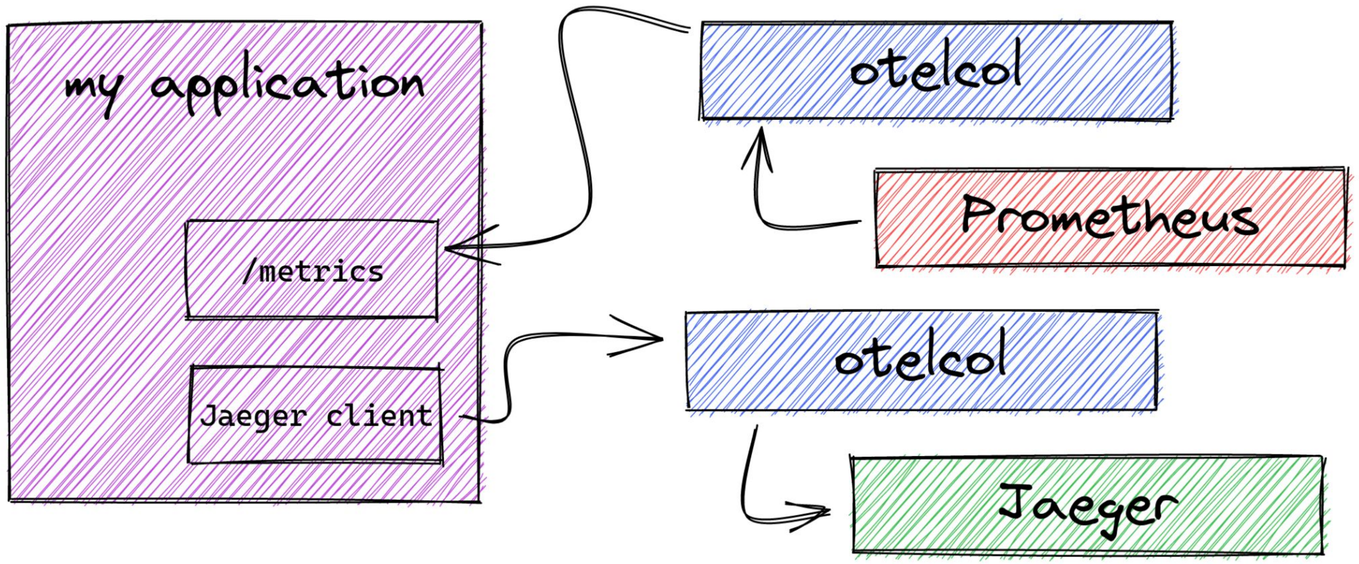


Bonus Pattern - Per signal

- ➔ Client emits different signals
- ➔ One collector for each signal
- ➔ Different backends

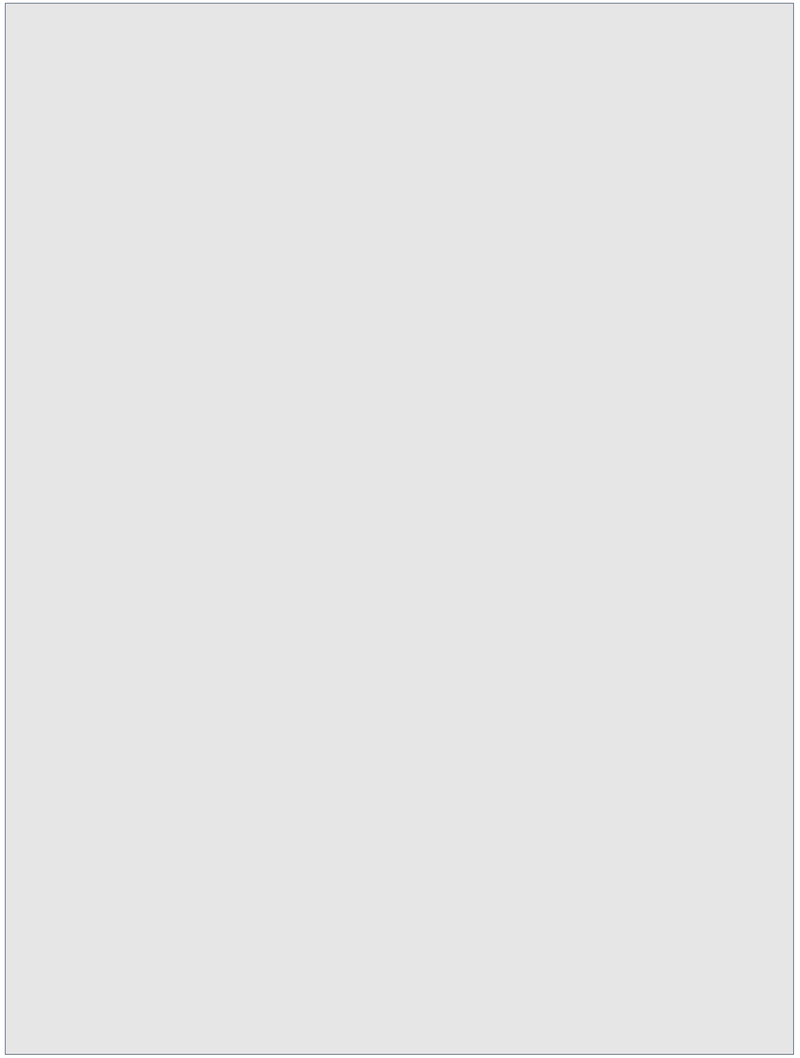


Bonus Pattern - Per signal



Key takeaways

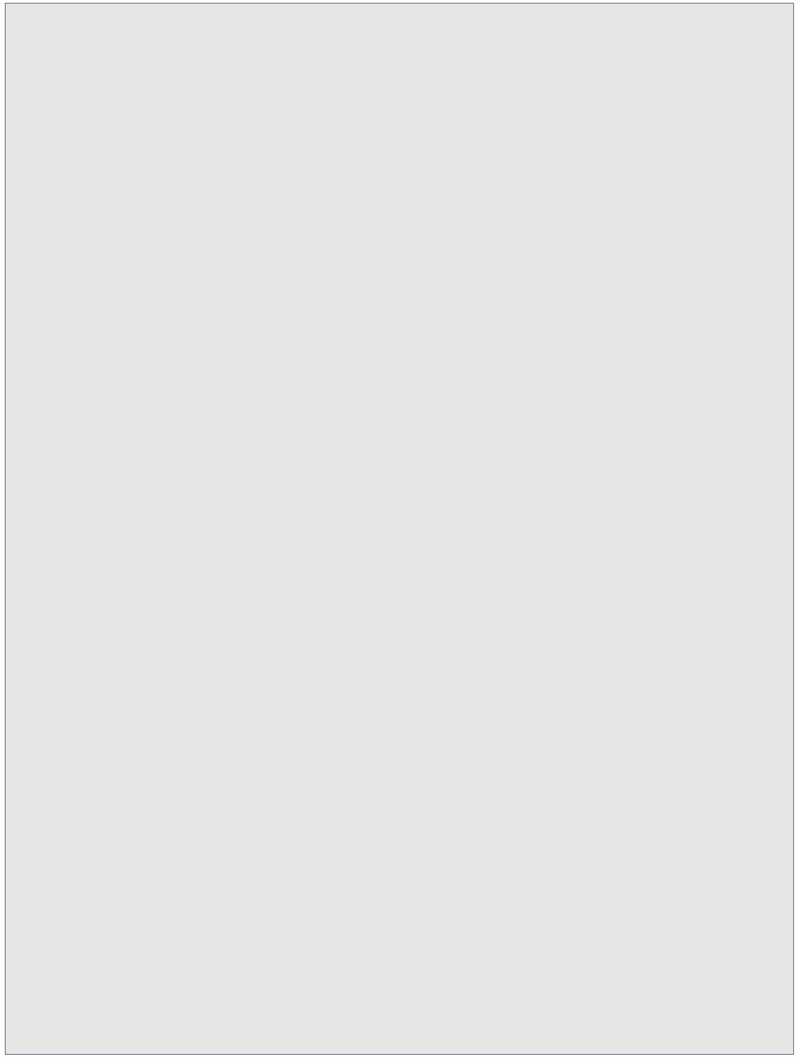
- ➔ The OpenTelemetry Collector is very versatile
- ➔ Get to know the existing components
- ➔ Chain collectors together
- ➔ Mix and match



Resources

- OpenTelemetry
- OpenTelemetry Collector
- OpenTelemetry Collector Contrib
- OpenTelemetry Collector Builder
- Patterns from this presentation

Contribute with your own patterns!



Thank you!