# Observability Day

EUROPE

Observability Day
EUROPE

# OpenTelemetry Gateways: Enforce, Transform, Route

*Natalie Ujuk, IG Group*
*Juraci Paixão Kröhling, OllyGarden*

# Agenda

Observability Day EUROPE

1. How we got here
2. Collector Gateways
3. Demo
4. Q&A


IG
28,355
27,554
24,405
23,785
28,355
27,554
24,405
23,785

# Why O11y breaks at scale

Observability Day EUROPE

## Reality at Scale

Heterogeneous Environments

Inconsistent Telemetry

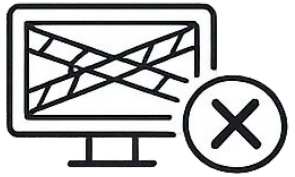
Cardinality/PII risk

Backend Coupling

VS

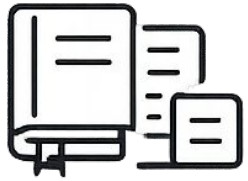
## Human Constraints

"We can't change the code"



Legacy Systems

Platform teams can't review everything



Docs ≠ Enforcement

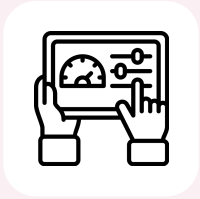
**You cannot scale observability by teaching every team to "*do it right.*"**

# What O11y at scale requires

Observability Day EUROPE

## Standards

Enforced, not suggested



## Controls

Validation, Redaction, Filtering

## Simple Defaults

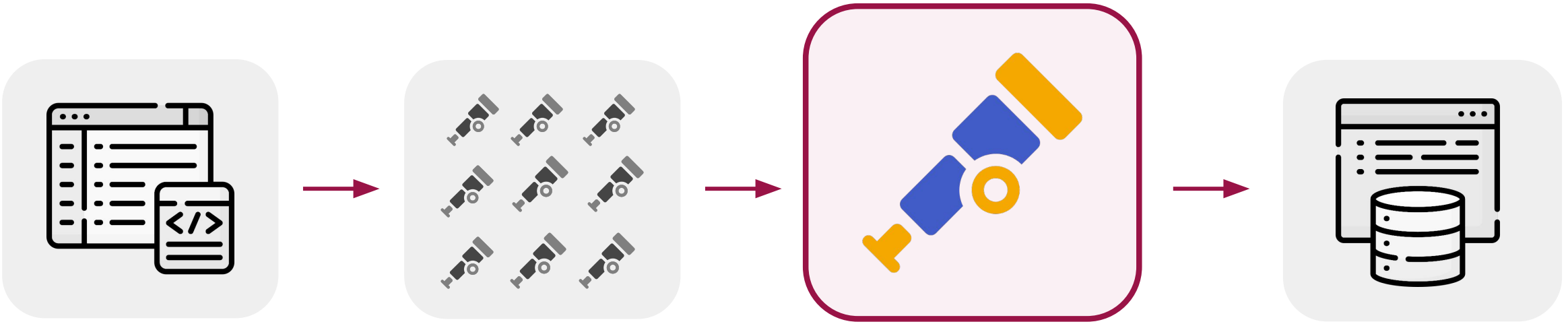
Reduce complexity for developers

## Flexibility

Change backends without touching code

# The Solution: Centralised Gateway



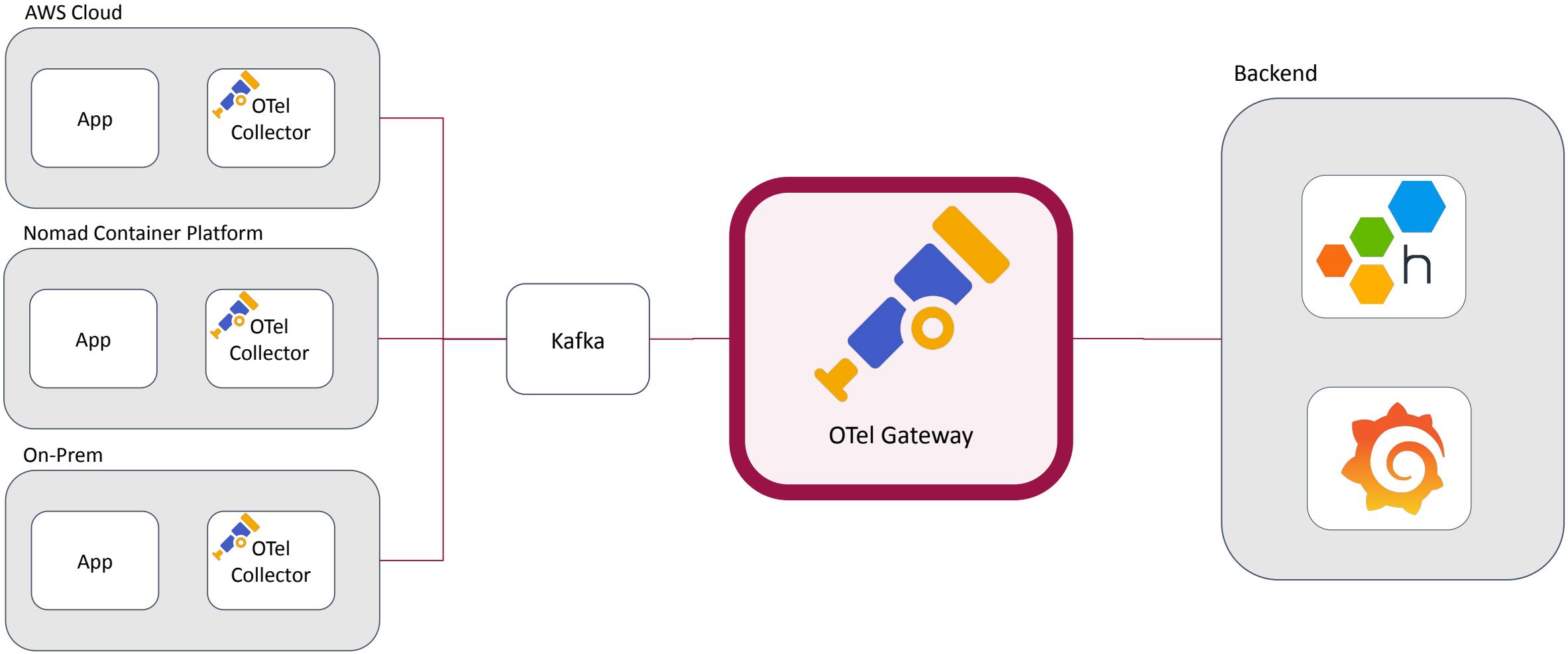



✓ CENTRALISED CONTROL

ENFORCE | TRANSFORM | ROUTE

# Our architecture





# Gateways - Today

Observability Day EUROPE

## The next slides show how gateways work today

In the future,

*OpenTelemetry Weaver could accomplish much of this*

# Gateways - Enforce

Observability Day EUROPE

**Purpose:** Validate data quality and enforce mandatory standards at the gateway

## What gateways can ENFORCE:

### Mandatory fields

- Reject telemetry missing required attributes `service.name` `service.namespace` `deployment.environment.name`

### Field value validation

- Validate `service.name` against a source of truth (CMDB)
- Block unknown or misconfigured services

### PII protection

- Multi-layer defense
- Sensitive data never reaches downstream
- Routing → Filtering → Redaction → Attribute removal

# Gateways - Enforce

Observability Day EUROPE

## Deep Dive: PII Protection

Prevent Sensitive Data from reaching backends

Pattern Redaction

```
redaction/pii_traces:
  allow_all_keys: true
  allowed_keys:
    - http.target
    - customer_attribute_name
  blocked_values:
    - '\b[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}\b'  # Email addresses
```

Scan for patterns

Attribute Removal

```
transform/remove_pii_attributes:
  statements:
    - delete_matching_keys(attributes, "^customer\\..*")
    - delete_matching_keys(attributes, "^details\\.data\\..*")
    - delete_key(attributes, "email") where attributes["email"] != nil
```

Use when attributes are known

# Gateways - Enforce

Observability Day EUROPE

## Deep Dive: PII Protection

Prevent Sensitive Data from reaching backends

Pattern Redaction

```
redaction/pii_traces:
  allow_all_keys: true
  allowed_keys:
    - http.target
    - customer attribute name
  blocked_values:
    - '\b[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}\b'  # Email addresses
```

Scan for patterns

Attribute Removal

```
transform/remove_pii_attributes:
  statements:
    - delete_matching_keys(attributes, "^customer\\..*")
    - delete_matching_keys(attributes, "^details\\.data\\..*")
    - delete_key(attributes, "email") where attributes["email"] != nil
```

Use when attributes are known

# Gateways - Transform

Observability Day EUROPE

**Purpose:** Modify telemetry in-flight to fix incomplete instrumentation and supplement with context applications can't provide.

**What gateways can TRANSFORM:**

### Fix incomplete instrumentation

- Fill missing required fields based on known context.

### Normalise conventions

- Correct misnamed, misplaced, or inconsistent attributes/values.

### Add system context

- Add additional attributes applications can’t know at application runtime (e.g. infrastructure metadata, environment context or ingest metadata).

### Enrich with geographic context

- Add city, region, country for client side observability

# Gateways - Transform

Observability Day EUROPE

## Deep Dive: Enrich with Geographic Context

Add information applications can't provide.

```
# Geographic enrichment for frontend telemetry
transform/normalize_to_client_address:
  statements:
    - set(attributes["frontend.client.ip"], body["client_address"])
      where IsMap(body) and resource.attributes["faro_frontend_observability"] == "true"

geoip:
  context: record
  providers:
    maxmind:
      database_path: /alloc/geoip/GeoLite2-City.mmdb
  attributes:
    - frontend.client.ip  # - adds city, region, country
```

**What this adds:** Geographic context for client side observability.

# Gateways - Transform

Observability Day EUROPE

## Deep Dive: Enrich with Geographic Context

Add information applications can't provide.

```
# Geographic enrichment for frontend telemetry
transform/normalize_to_client_address:
  statements:
    - set(attributes["frontend.client.ip"], body["client_address"])
      where IsMap(body) and resource.attributes["faro_frontend_observability"] == "true"

geoip:
  context: record
  providers:
    maxmind:
      database_path: /alloc/geoip/GeoLite2-City.mmdb
  attributes:
    - frontend.client.ip  # - adds city, region, country
```

**What this adds:** Geographic context for client side observability.

# Gateways - Route

Observability Day EUROPE

**Purpose:** Intelligently direct telemetry to multiple backends/pipelines without changing application code

## What gateways can ROUTE:

### To backends

- Send telemetry to different backends based on `deployment.environment.name`

### To pipelines

- Performance - Apply intensive processing only to telemetry that requires it.
- Cost Optimisation - Route high-cardinality metrics to aggregation pipeline

### Gateway to gateway

- Distribute and load balance traces by `traceID` to sampling gateways

# Gateways - Route

Observability Day EUROPE

## Deep Dive: Selective Processing - Performance

**Example 1:** Route only services requiring PII scanning and bypass others.

```
routing/pii_filtering_traces:
default_pipelines: [traces/passthrough]
table:
  - statement: route() where resource.attributes["service.name"] == "service_a"
    pipelines: [traces/pii-redaction]
  - statement: route() where resource.attributes["service.name"] == "service_b"
    pipelines: [traces/pii-redaction]
```

**Impact:** Only a small percentage of services may contain PII. Route them to scanning pipeline, others bypass. Major performance win.

# Gateways - Route

Observability Day EUROPE

## Deep Dive: Selective Processing - Performance

**Example 1:** Route only services requiring PII scanning and bypass others.

```
routing/pii_filtering_traces:
default_pipelines: [traces/passthrough]
table:
  - statement: route() where resource.attributes["service.name"] == "service_a"
    pipelines: [traces/pii-redaction]
  - statement: route() where resource.attributes["service.name"] == "service_b"
    pipelines: [traces/pii-redaction]
```

**Impact:** Only a small percentage of services may contain PII. Route them to scanning pipeline, others bypass. Major performance win.

# Gateways - Route

Observability Day EUROPE

## Deep Dive: Selective Processing - Cost

**Example 2:** Route high-cardinality data to aggregation, everything else direct.

```
routing/metrics_aggregation:
  default_pipelines: [metrics/export]
  table:
    - statement: route() where attributes["metric_name"] == "http_server_response_size_bytes_bucket"
      pipelines: [metrics/aggregation]
    - statement: route() where IsMatch(attributes["metric_name"], "^jvm_.*")
      pipelines: [metrics/aggregation]
```

**Impact:** Histogram buckets and per-thread JVM metrics get aggregated. Reduces cardinality by 97.5% (304K → 7K time series).

# Gateways - Route

Observability Day EUROPE

## Deep Dive: Selective Processing - Cost

**Example 2:** Route high-cardinality data to aggregation, everything else direct.

```
routing/metrics_aggregation:
  default_pipelines: [metrics/export]
  table:
    - statement: route() where attributes["metric_name"] == "http_server_response_size_bytes_bucket"
      pipelines: [metrics/aggregation]
    - statement: route() where IsMatch(attributes["metric_name"], "^jvm_.*")
      pipelines: [metrics/aggregation]
```

**Impact:** Histogram buckets and per-thread JVM metrics get aggregated. Reduces cardinality by 97.5% (304K → 7K time series).

# Examples

Observability Day EUROPE

- Code available here:

https://github.com/jpkrohling/o11y-day-eu26-otelcol-gateways

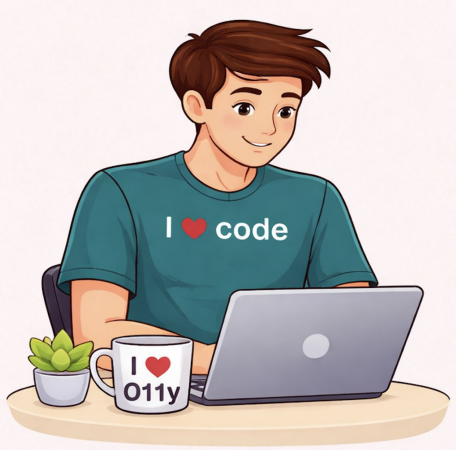
Observability Day
EUROPE

# Demo

Observability Day
EUROPE

# Why this architecture scales

Observability Day
EUROPE




## Developer

“Observability just happens for me”


## O11y Engineer

“I enforce, transform and route…from my repo”

**You don’t need every team to do it right.***

*Observability Engineering excluded

# Takeaways

Observability Day EUROPE

**The Problem:** You can't scale observability by teaching every team to "do it right."

**The Solution:** OpenTelemetry Collector Gateways - your centralised control point between applications and backends.

**What gateways enable:**

- **Enforce** standards after data leaves applications
- **Transform** in-flight telemetry to fix incomplete instrumentation and add context
- **Route** intelligently without changing application code

**Gotchas to consider**:

- Gateways become critical infrastructure - plan for high availability
- Start simple, add complexity as needed (don't over-engineer day one)
- Keep edge collectors vanilla - resist the urge to push logic there

**Code examples:** https://github.com/jpkrohling/o11y-day-eu26-otelcol-gateways

Observability Day
EUROPE

# Q&A

Observability Day
EUROPE

# Thank you! Obrigado!

Natalie Ujuk
Observability Lead

Juraci Paixão Kröhling
Making telemetry pipelines more efficient