Tracing Adventures from Pull Request to Production

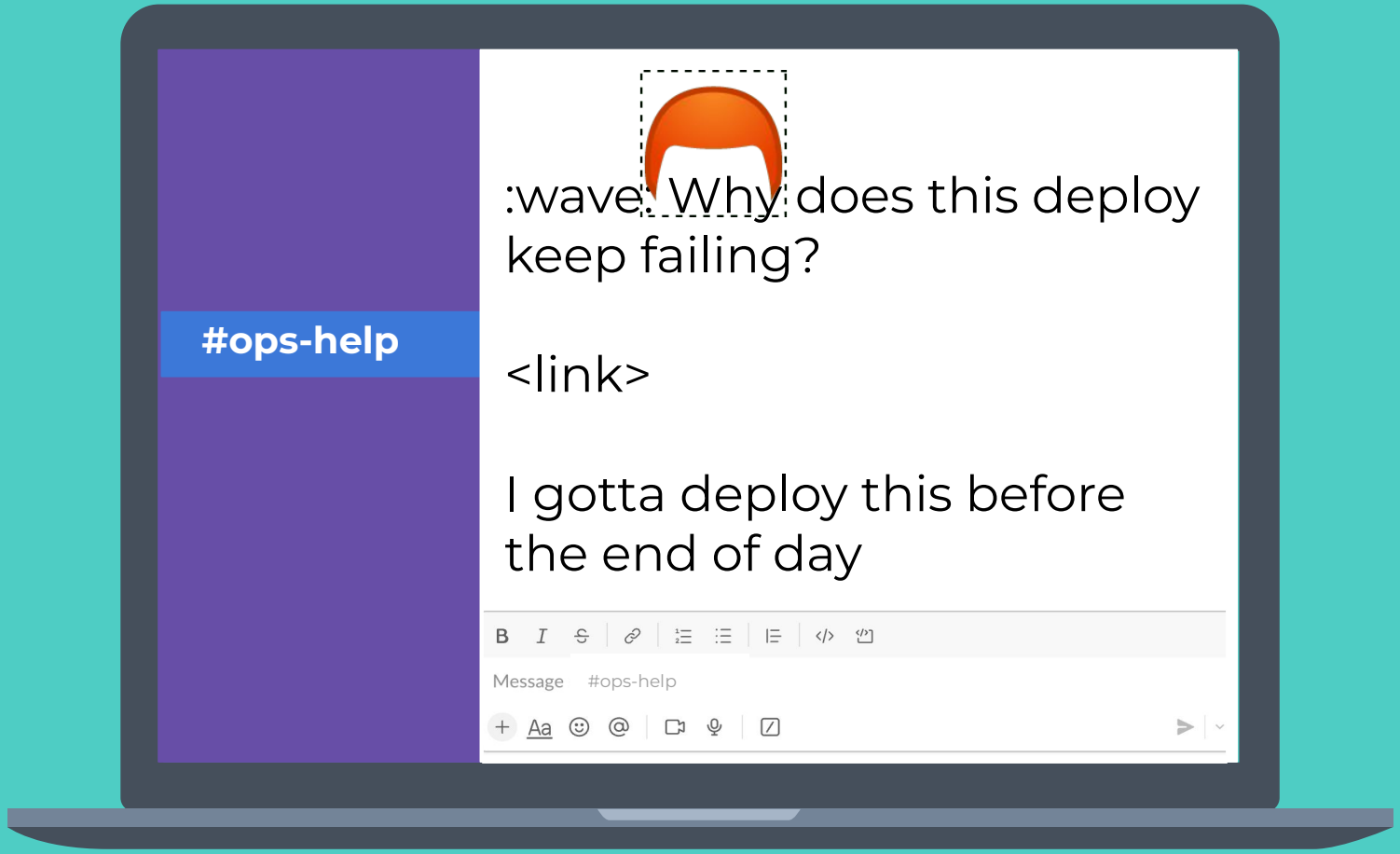
@paigerduty

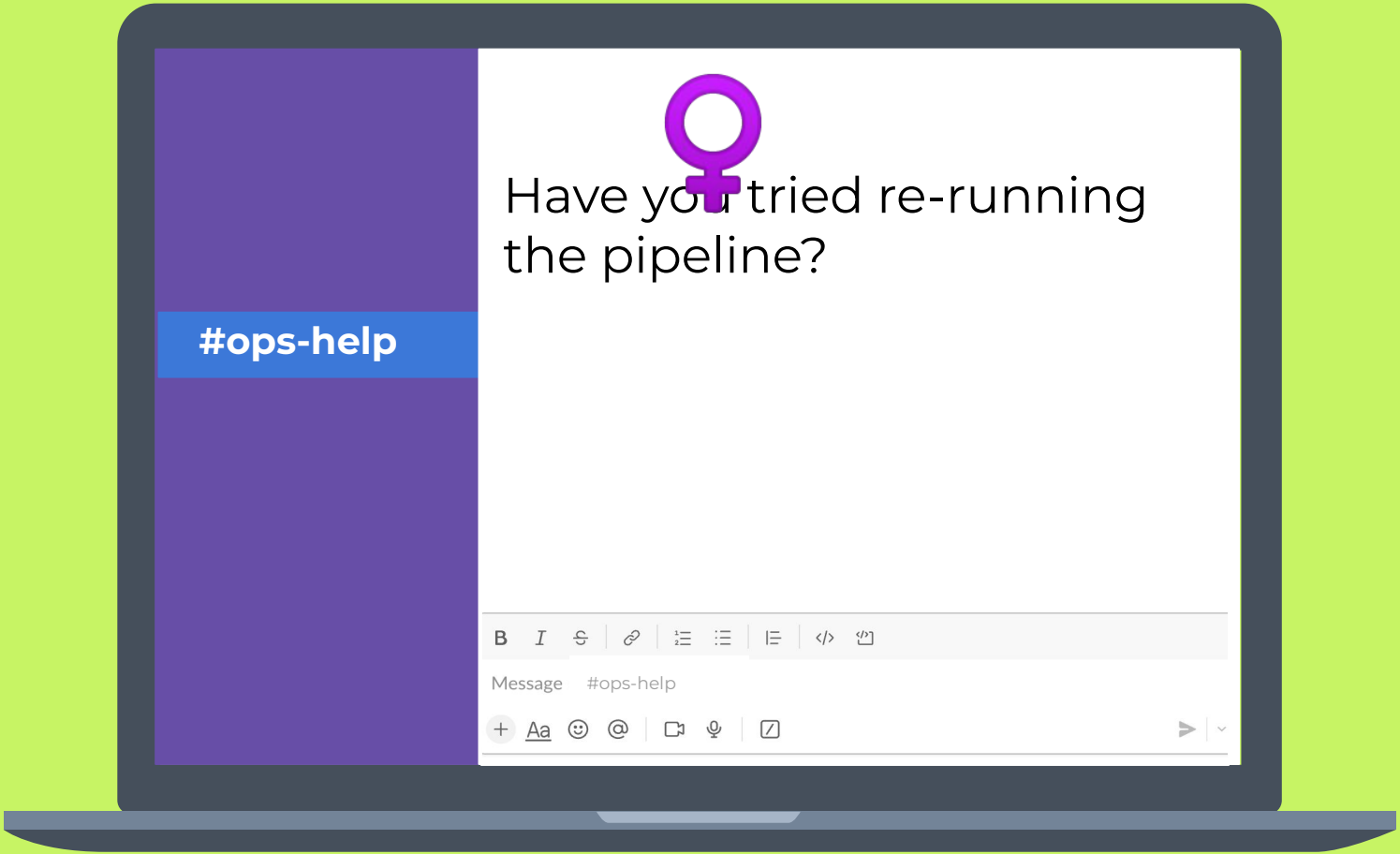
chronosphere


Can I Deploy?!
Weds, 4:30 pm

UGH why is pipeline failing?

...better ping Ops


#ops-help
:wave: Why does this deploy keep failing?
<link>
I gotta deploy this before the end of day
Message
#ops-help


#ops-help
Have you tried re-running the pipeline?
Message
#ops-help

#ops-help

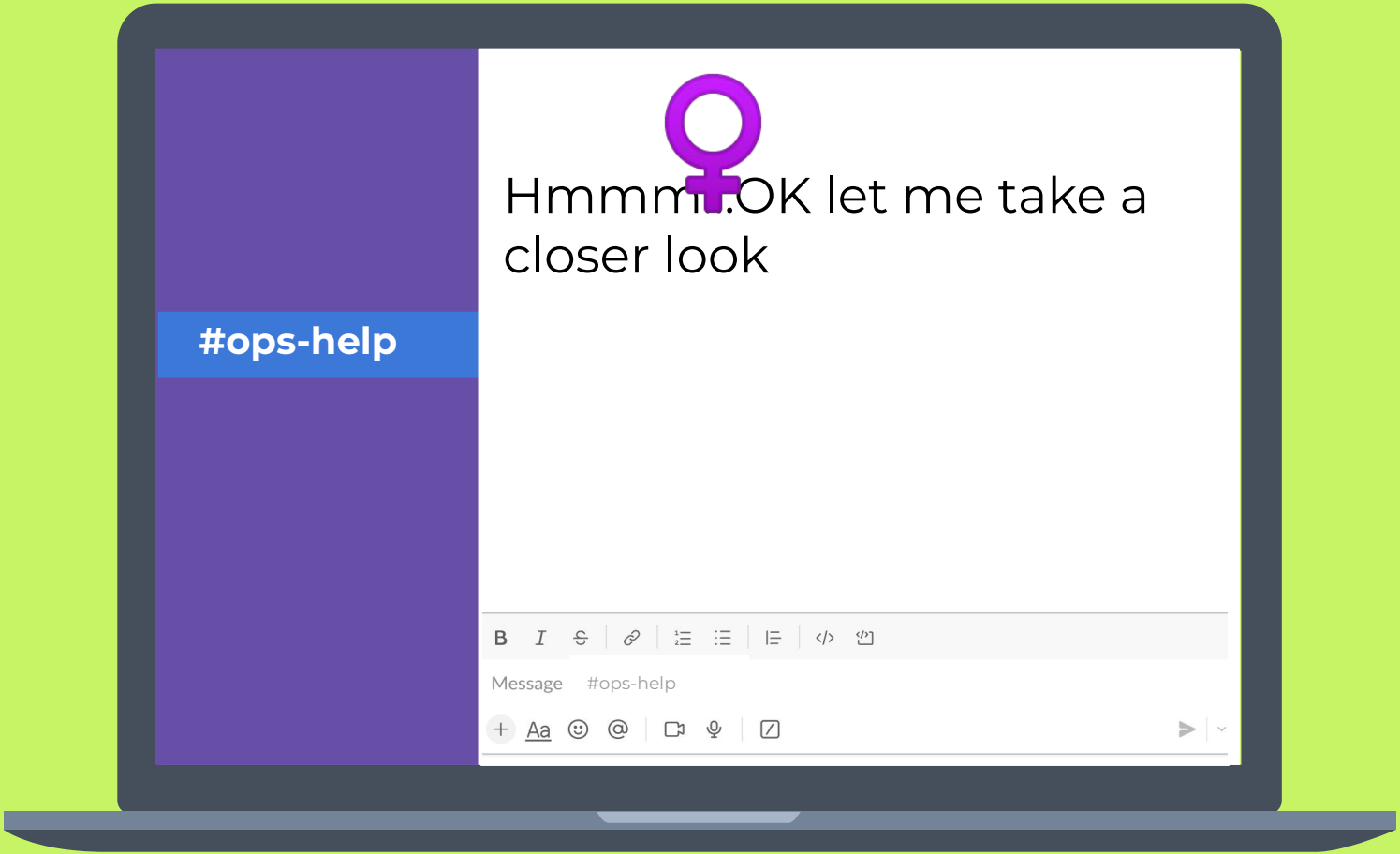
Yep. Tried re-running it 3x before posting here.

Every time it keeps hanging and spits out this $BIG_ERROR

Message #ops-help

OF COURSE there's an urgent issue to solve with only 30m left in the day

...I can't wait to log off


#ops-help
Hmmm...OK let me take a closer look
Message #ops-help

@paigerduty

chronosphere

1.

Dev Ops

@paigerduty

Dev and Ops have distinct perspectives and data on CI/CD platform its not always straightforward to know who is responsible for an issue

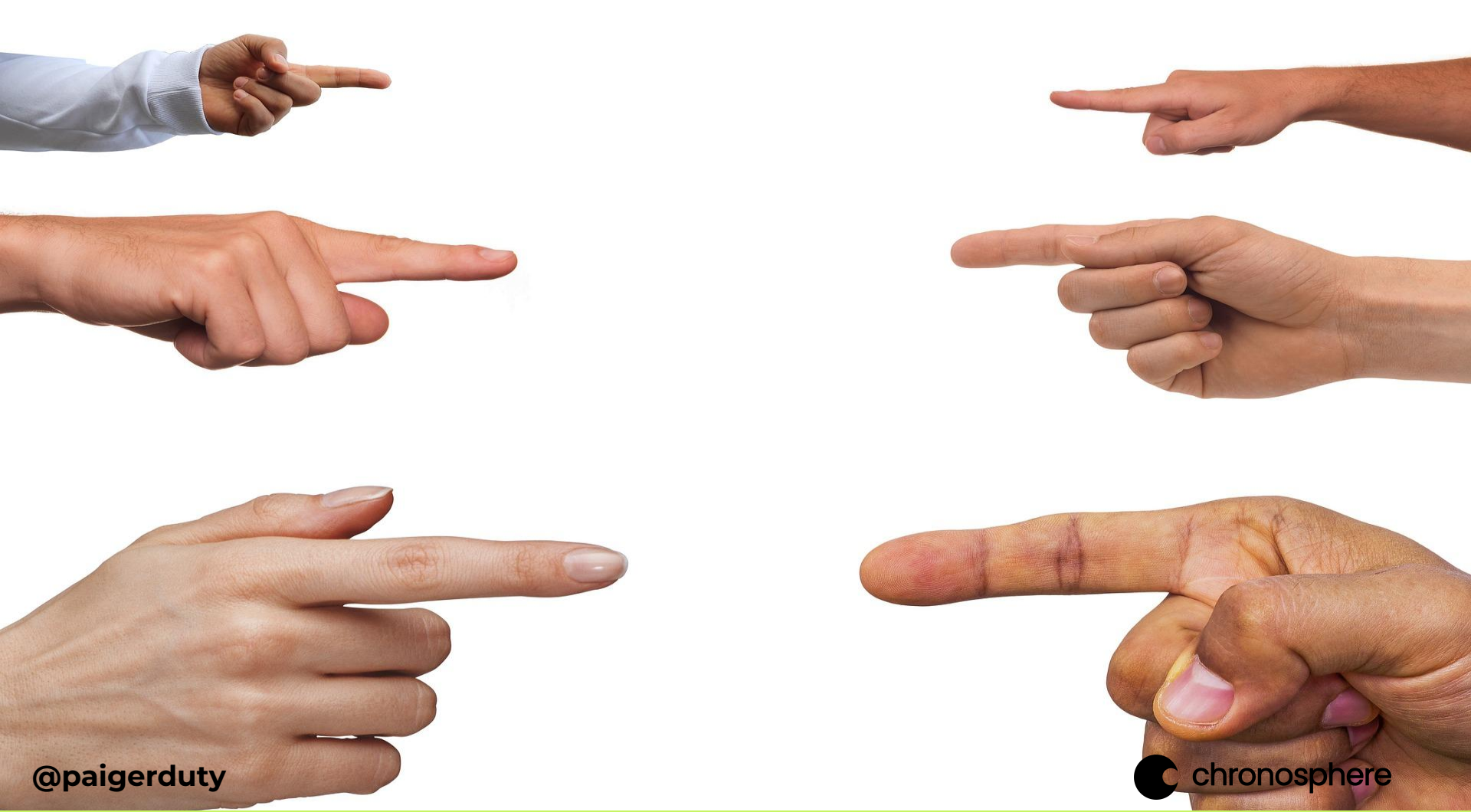
@paigerduty

Dev

Ops

@paigerduty

chronosphere



@paigerduty

chronosphere

It's the platform!

@paigerduty

chronosphere

It's the platform!

It's your change!

@paigerduty

chronosphere

Dev

Ops

Dev

- Application
  - Code
  - Tests
- Deployment Pipeline
- Deploying changes

Dev

Ops

- Infrastructure
  - Nodes / Capacity
  - Network
- CI/CD Platform
  - Availability / Reliability
  - Cost Efficiency

Dev

- Application
  - Code
  - Tests
- Deployment Pipeline
- Deploying changes

Ops

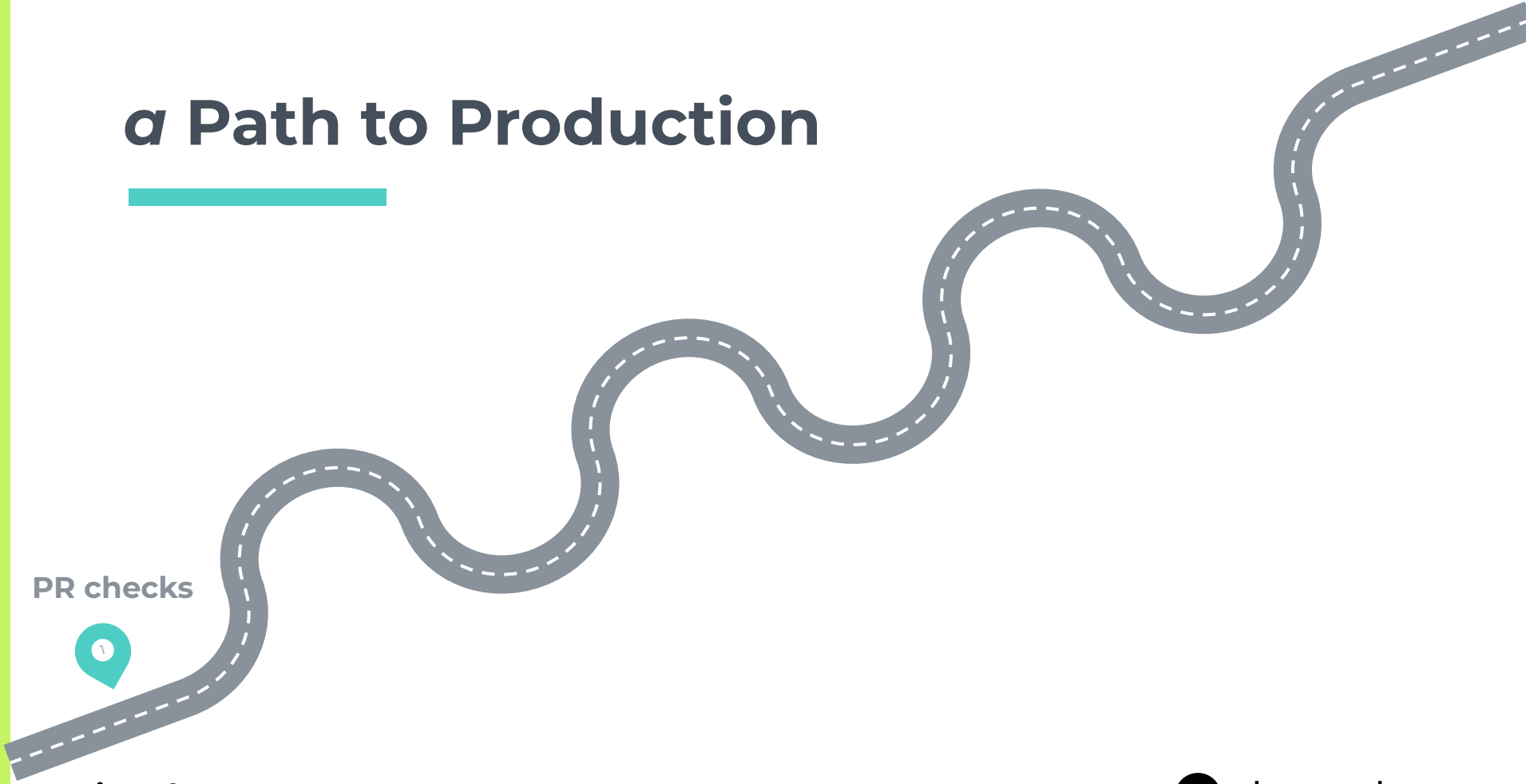
- Infrastructure
  - Nodes / Capacity
  - Network
- CI/CD Platform
  - Availability / Reliability
  - Cost Efficiency

2.

The Path to Production

Changes go on a long opaque journey from PR to Production. Tracing provides a vantage point both dev and ops can use

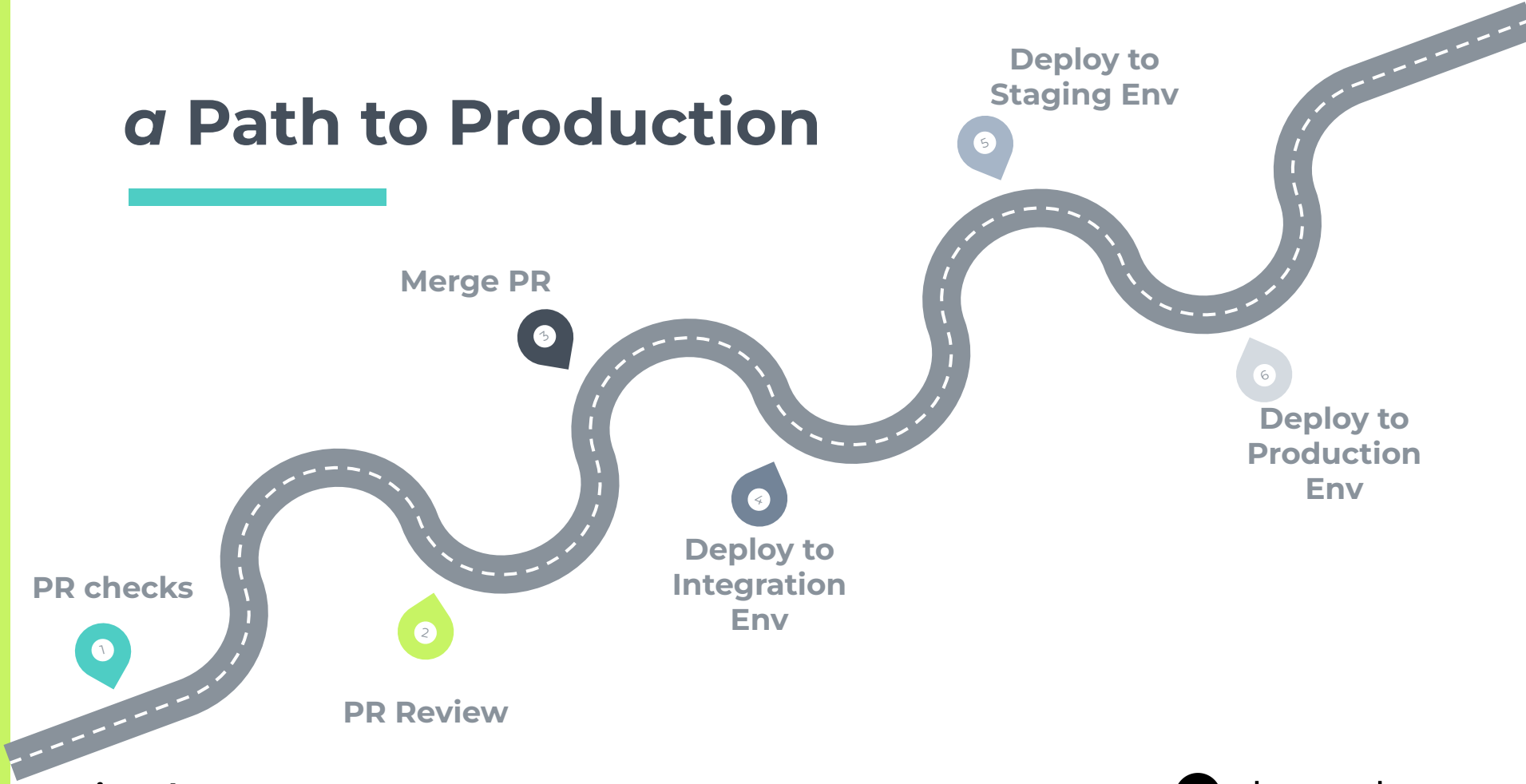
a Path to Production

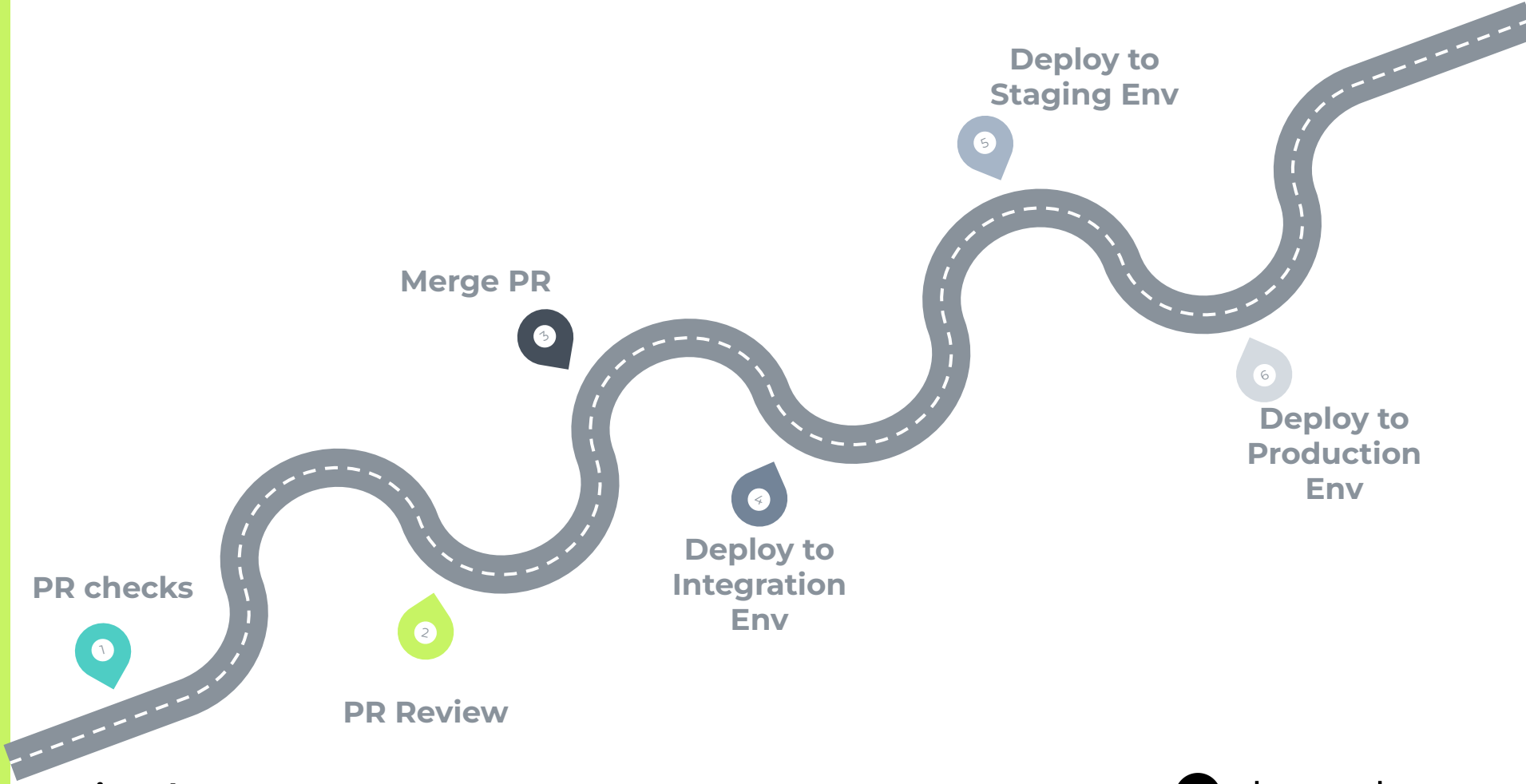


a Path to Production

a Path to Production

a Path to Production




PR checks
1
PR Review
2
Merge PR
3
Deploy to Integration Env
4
Deploy to Staging Env
5
Deploy to Production Env
6

Integration

Staging

Production

Local

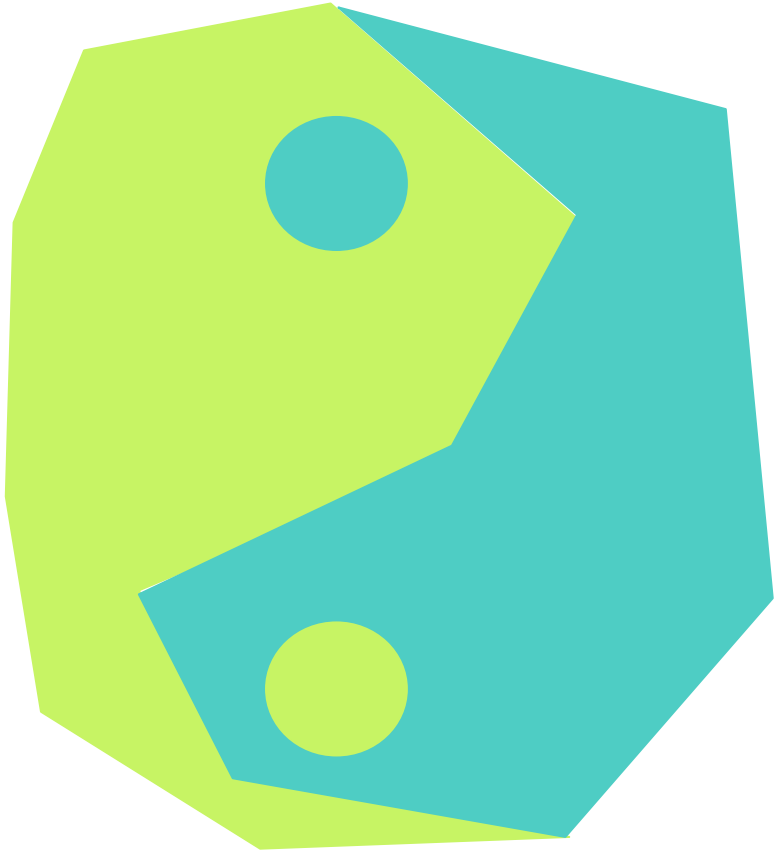
PR

Integration

Staging

Production

Dev & Ops



3.

Tracing Together

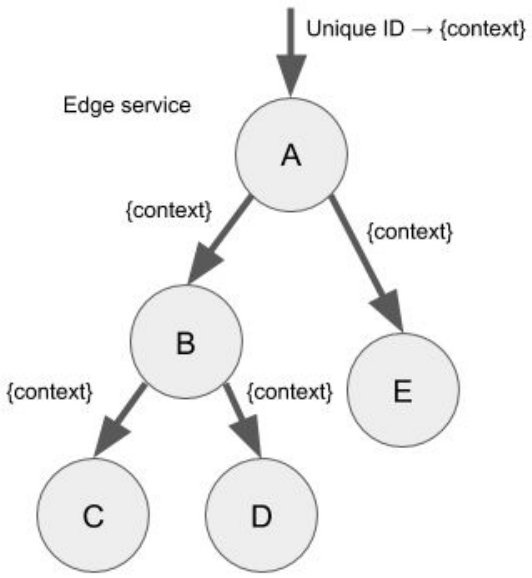
@paigerduty

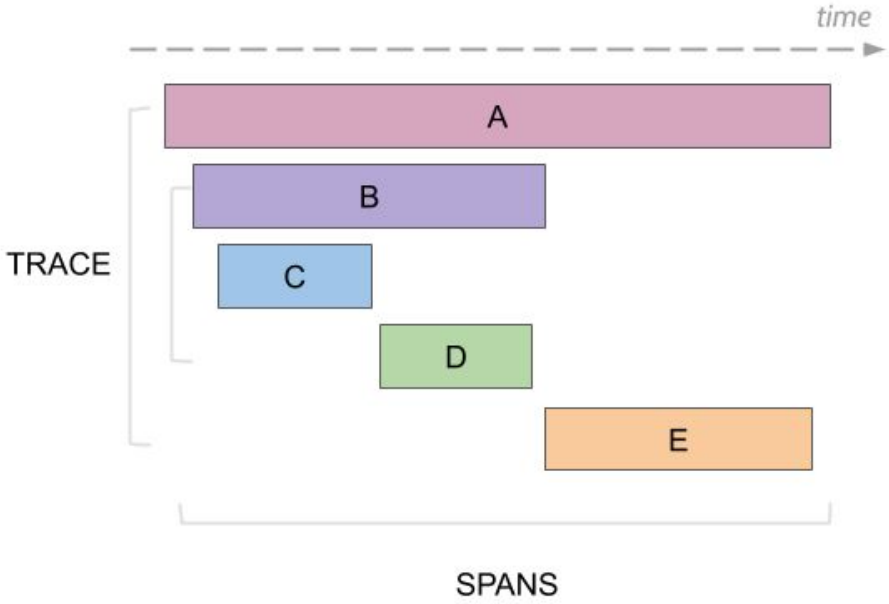
Traces bring people together, just as traces span service boundaries they span team/organization boundaries.

But we have metrics and logs...

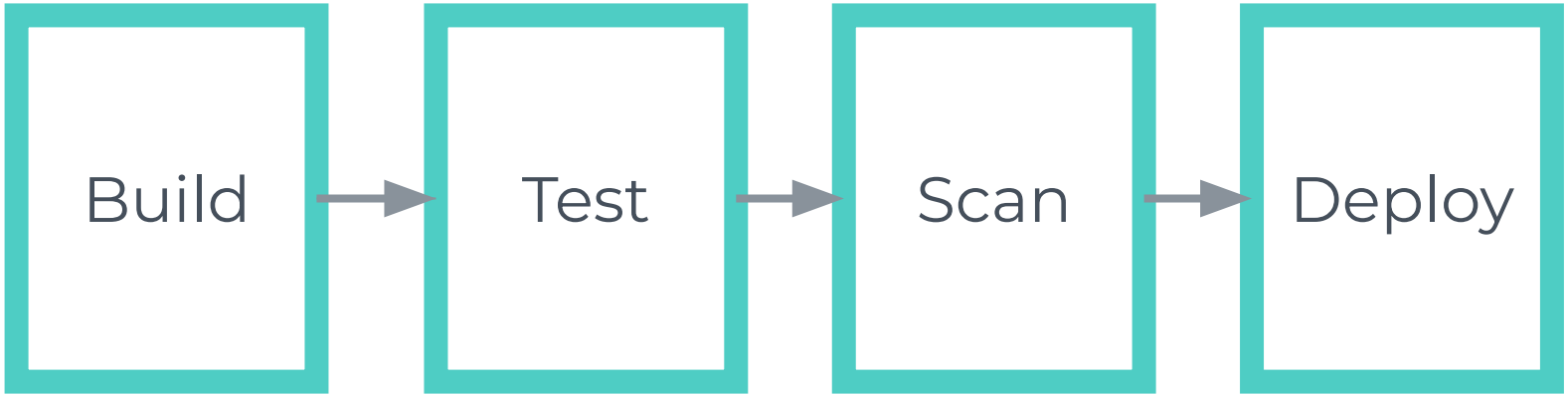
Why add tracing?

What's a Trace?

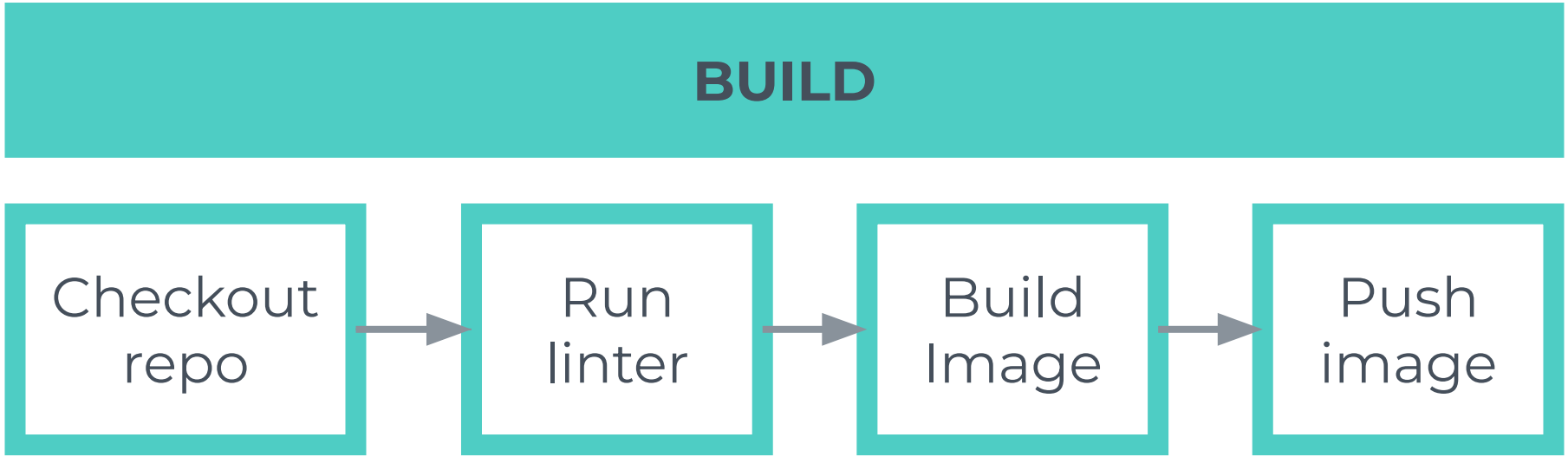




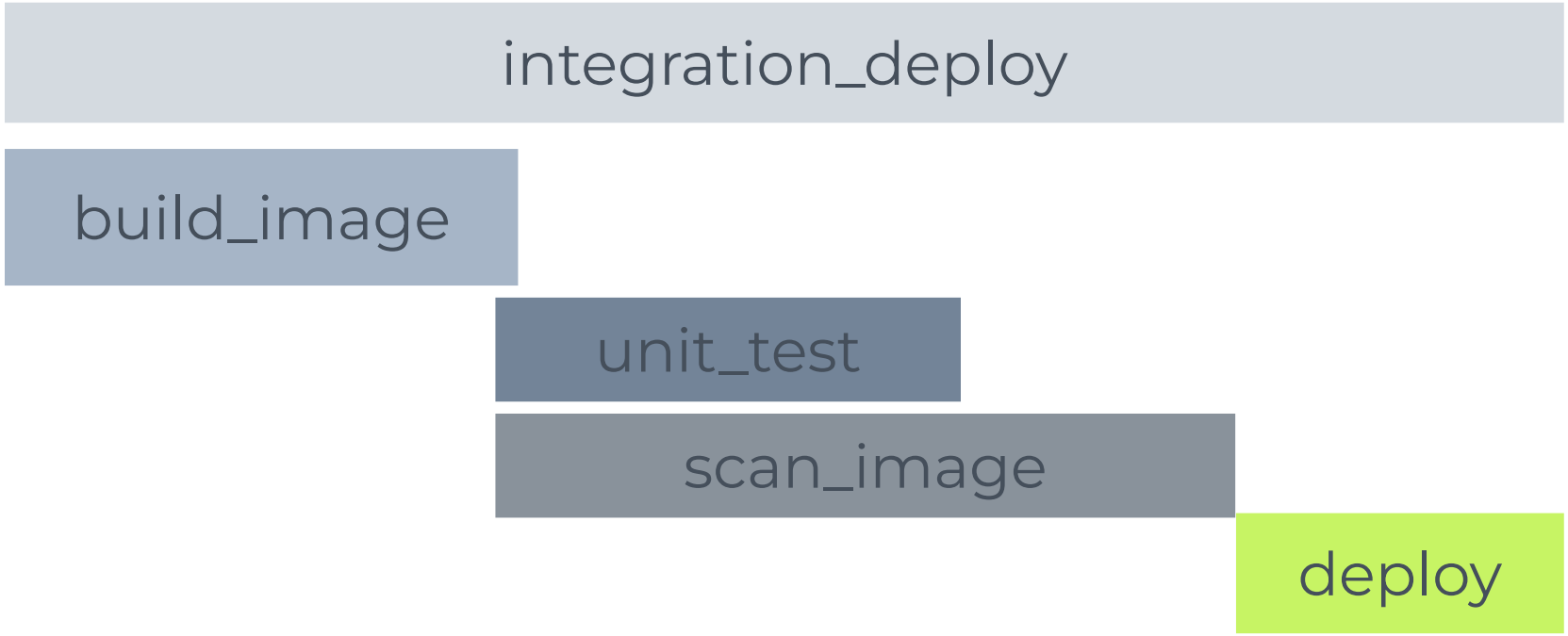
Pipeline Stages



Pipeline Steps



Pipeline as Trace Waterfall



JAEGER UI | Search | Compare | System Architecture | Lookup by Trace ID... | About Jaeger

jenkins: ecommerce-antifraud/main 7cf84d3

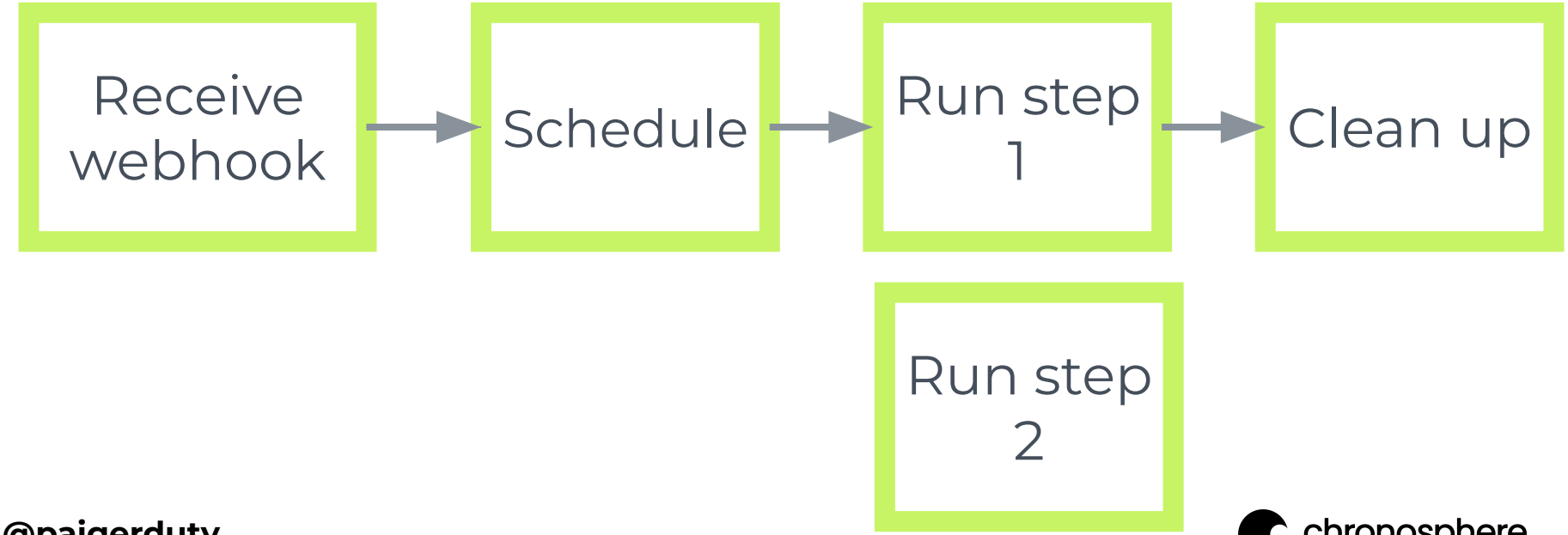
Find... | Trace Timeline

Service & Operation	0µs	19.14s	38.28s	57.42s	1m 17s
jenkins ecommerce-antifraud/main					
jenkins Phase: Start	129.85ms				
jenkins Phase: Run					
jenkins Agent: linux	1m 4s				
jenkins Agent Allocation: linux	367.17ms				
jenkins Stage: Declarative: Checkout SCM	2.51s				
jenkins → git checkout: github.com/cyrille-leclerc/ecomm...	2.19s				
jenkins Stage: Build	58.65s				
jenkins mvn deploy spring-boot:build-image	56.52s				
maven Build: com.mycompany.ecommerce:anti-fraud:0.0...	54.07s				
maven com.mycompany.ecommerce:anti-fraud	54.07s				
maven spring-boot:build-info (build-info) @ anti-...	760.89ms				
maven resources:resources (default-resources) ...	246.25ms				
maven compiler:compile (default-compile) @ ant...	234.74ms				
maven resources:testResources (default-testRe...	7.32ms				
maven compiler:testCompile (default-testCompi...	13.12ms				
maven surefire:test (default-test) @ anti-fraud	6.58s				
maven → snyk.io snyk:test (snyk-test) @ a...	7.04s				
maven jar:jar (default-jar) @ anti-fraud	489.76ms				
maven spring-boot:repackage (repackage) @ an...	521.66ms				
maven spring-boot:repackage (default) @ anti-fr...	17.11ms				
maven spring-boot:build-info (build-info) @ anti-...	2.17ms				
maven resources:resources (default-resources) ...	37.54ms				
maven compiler:compile (default-compile) @ ant...	8.71ms				
maven resources:testResources (default-testRe...	5.54ms				
maven compiler:testCompile (default-testCompi...	12.84ms				
maven surefire:test (default-test) @ anti-fraud	8.62ms				
maven → snyk.io snyk:test (snyk-test) @ a...	7.52s				
maven jar:jar (default-jar) @ anti-fraud	44.61ms				
maven spring-boot:repackage (repackage) @ an...	182.16ms				
maven spring-boot:repackage (default) @ anti-fr...	17.73ms				
maven → docker.io spring-boot:build-ima...	21.67s				
maven install:install (default-install) @ anti-fraud	282.29ms				
maven → snyk.io snyk:monitor (snyk-mon...	5.96s				
maven → maven.example.com deplo...	1.97s				
maven Build: com.mycompany.ecommerce:anti-fraud:0....	1.53s				
maven Build: com.mycompany.ecommerce:anti-fraud:0....	2.29s				
maven Build: com.mycompany.ecommerce:anti-fraud:0....					
jenkins archiveArtifacts					387.73ms

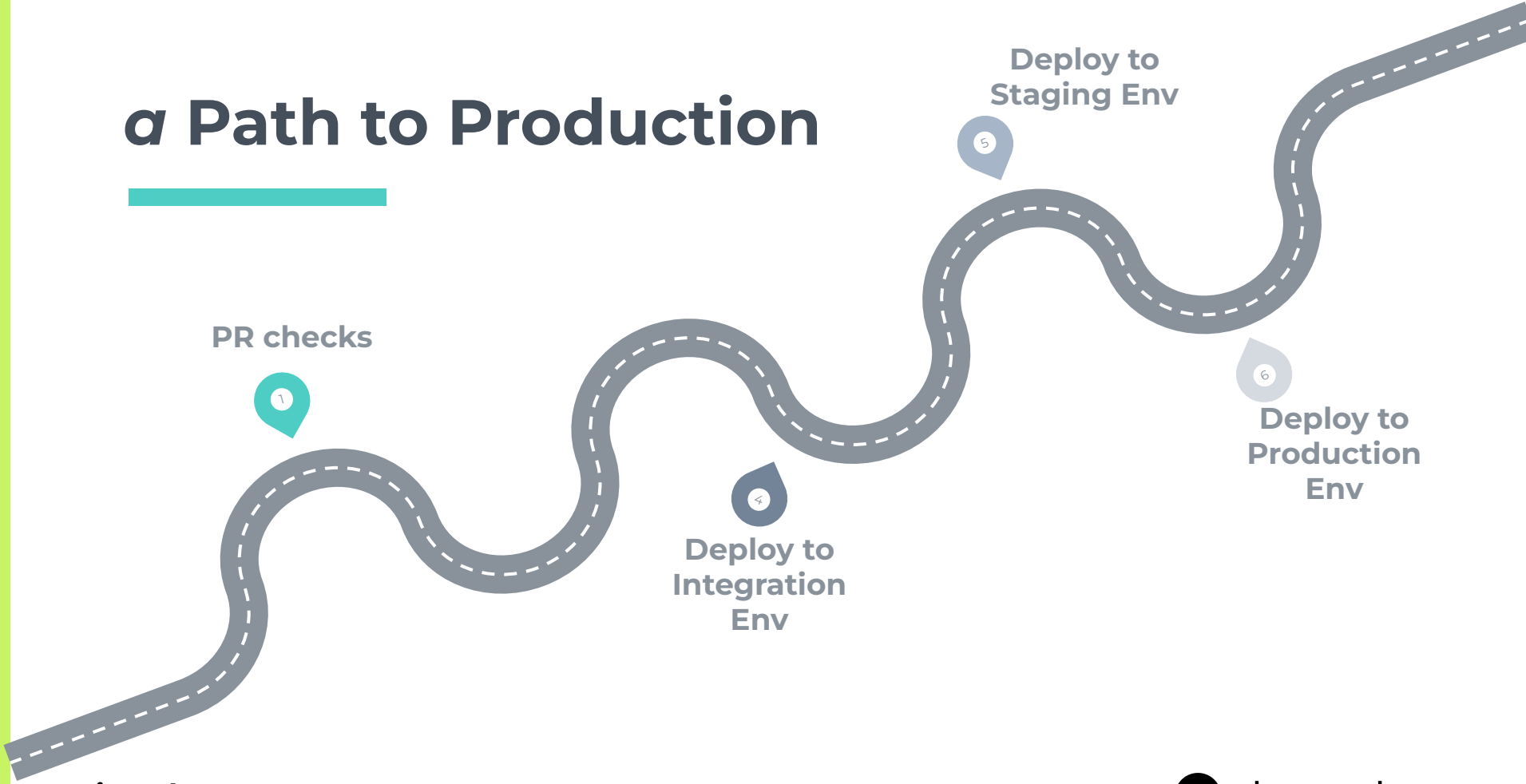
@paigerduty

chronosphere

Pipeline Steps - Platform POV



a Path to Production



@paigerduty

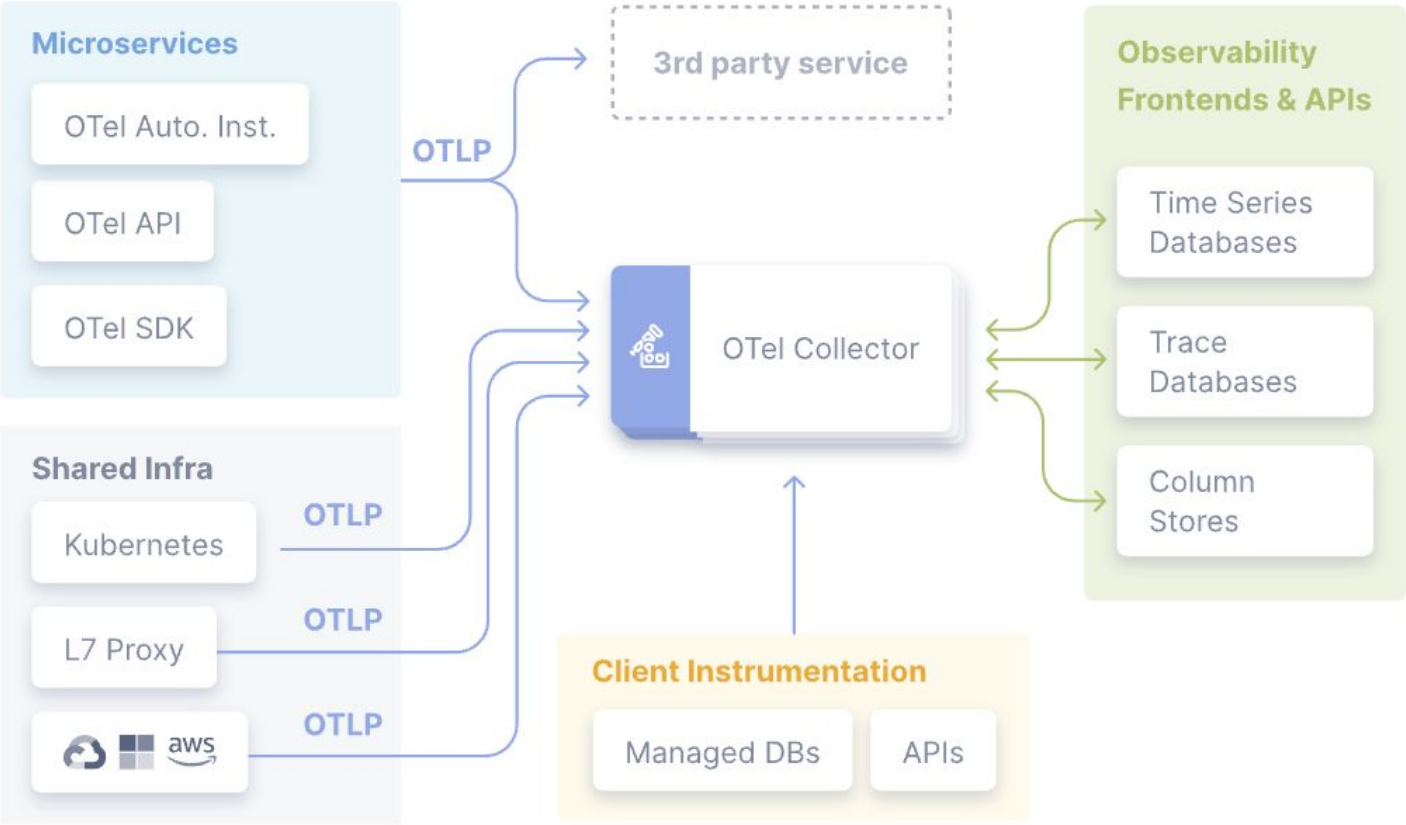
chronosphere


ASK

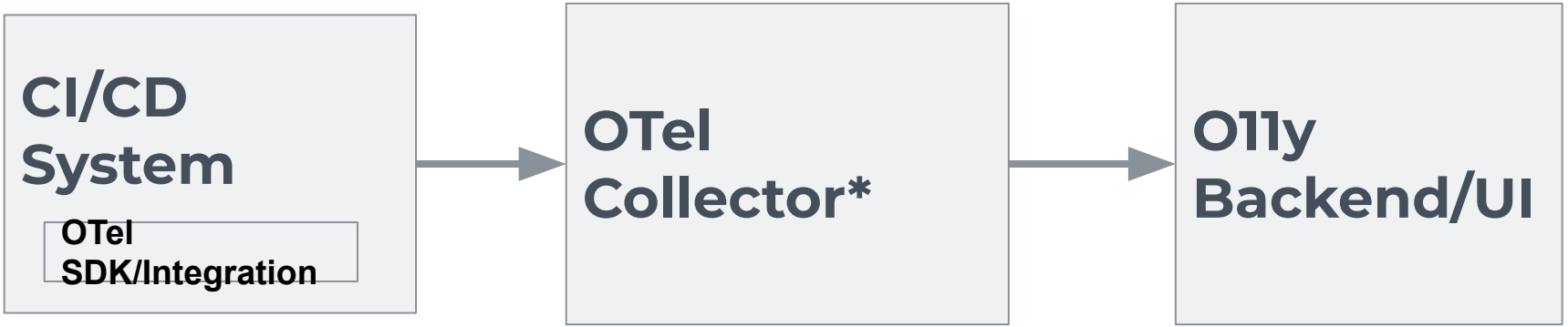
4.

Getting Started


OpenTelemetry


Microservices
OTel Auto. Inst.
OTel API
OTel SDK
OTLP
3rd party service
Observability Frontends & APIs
Time Series Databases
Trace Databases
Column Stores
OTel Collector
Shared Infra
Kubernetes
OTLP
L7 Proxy
OTLP
aws
OTLP
Client Instrumentation
Managed DBs
APIs

Data Flow

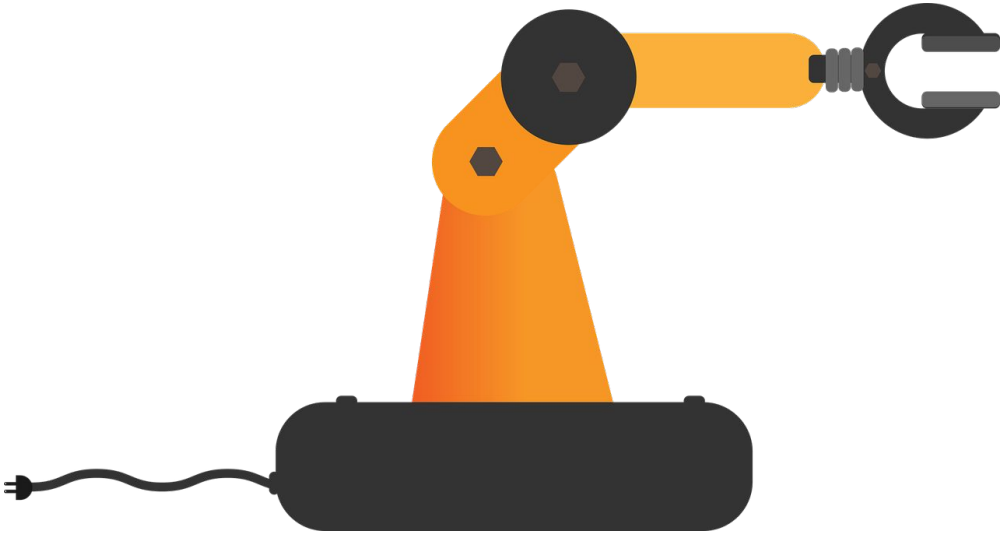


Generate & send

Process & route

Store & visualize

DIY or automatic?



DIY Instrumentation

- Otel-cli
- Tracepusher

- OTel Registry find your stack :)

OpenTelemetry Collector

Tracepusher

```yaml
image:
  name: gardnera/tracepusher:v0.5.0-ci

# Set your OTEL collector endpoint address here...
variables:
  OTEL_COLLECTOR_ENDPOINT: "http://1.2.3.4:4318"
```

```
default:
  before_script:
    - echo subspan_start_time=$(date +%s) >> /tmp/vars.env
    - echo subspan_id=$(hexdump -vn8 -e'4/4 "%08X" 1 "\n"' /dev/urandom) >> /tmp/vars.env
  after_script:
    - source /tmp/vars.env . # reconsitute env vars from before_script
    - subspan_end_time=$(date +%s)
    - duration=$(( subspan_end_time - subspan_start_time ))
    - python3 /app/tracepusher.py
        --endpoint=$OTEL_COLLECTOR_ENDPOINT
        --service-name=$CI_PROJECT_NAME
        --span-name=$CI_JOB_NAME
        --duration=$duration
        --trace-id=$main_trace_id
        --span-id=$subspan_id
        --parent-span-id=$main_span_id
        --time-shift=True
```

```yaml
generate-trace-id:
  stage: preparation
  before_script: []
  after_script: []
  script:
    - echo "main_trace_id=$(hexdump -vn16 -e'4/4 "%08X" 1 "\n"' /dev/urandom)" >> variables.env
    - echo "main_span_id=$(hexdump -vn8 -e'4/4 "%08X" 1 "\n"' /dev/urandom)" >> variables.env
    - echo "main_trace_start_time=$(date +%s)" >> variables.env
  artifacts:
    reports:
      dotenv: variables.env              # Use artifacts:reports:dotenv to expose the variables to other jobs
```

```yaml
default:
  before_script:
    - echo subspan_start_time=$(date +%s) >> /tmp/vars.env
    - echo subspan_id=$(hexdump -vn8 -e'4/4 "%08X" 1 "\n"' /dev/urandom) >> /tmp/vars.env
  after_script:
    - source /tmp/vars.env . # reconsitute env vars from before_script
    - subspan_end_time=$(date +%s)
    - duration=$(( subspan_end_time - subspan_start_time ))
    - python3 /app/tracepusher.py
        --endpoint=$OTEL_COLLECTOR_ENDPOINT
        --service-name=$CI_PROJECT_NAME
        --span-name=$CI_JOB_NAME
        --duration=$duration
        --trace-id=$main_trace_id
        --span-id=$subspan_id
        --parent-span-id=$main_span_id
        --time-shift=True
```

```yaml
build-job:
  stage: build
  script:
    # START: Do your real work here
    - echo "in build-job"
    - sleep 2
    # END: Do your real work here

second-build-job:
  stage: build
  script:
    # START: Do your real work here
    - echo "in second-build-job"
    - sleep 1
    # END: Do your real work here

verify-build:
    stage: verify
    script:
      # START: Do your real work here
      - echo "in verify-build"
      - sleep 3
      # END: Do your real work here
```

```yaml
push-main-trace:
  stage: .post
  after_script: []
  script:
    - main_trace_end_time=$(date +%s)
    - duration=$(( main_trace_end_time - main_trace_start_time ))
    - python3 /app/tracepusher.py
        --endpoint=$OTEL_COLLECTOR_ENDPOINT
        --service-name=$CI_PROJECT_NAME
        --span-name=$CI_PROJECT_NAME-$CI_PIPELINE_ID
        --duration=$duration
        --trace-id=$main_trace_id
        --span-id=$main_span_id
        --time-shift=True
```

Integrations

- Ansible Tracing Callback
- Maven Extension
- Pytest

Search | Submit | Reset | Language | Instrumentation

Any Language
Collector
C++
Crystal
Dart
.NET
Elixir
Erlang
Go
Haskell
Java
JavaScript
Julia
Kotlin
Lua
Ocaml
Perl
PHP
Python
Ruby
Rust
Scala
Swift

httpd (Apache) Instrumentation
httpd (Apache) OpenTelemetry module to add OpenTelemetry distributed tracing support to httpd.

Nginx Instrumentation
Nginx OpenTelemetry module to add OpenTelemetry distributed tracing support to nginx.

OpenTelemetry Webserver Module
The OTEL webserver module comprises of both Apache and Nginx instrumentation.

ASP.NET Instrumentation
This is an Instrumentation Library, which instruments ASP.NET and collect metrics and traces about incoming web requests.

ASP.NET Telemetry HttpModule for OpenTelemetry
The ASP.NET Telemetry HttpModule enables distributed tracing of incoming ASP.NET requests using the OpenTelemetry API.

ASP.NET Core Instrumentation for OpenTelemetry .NET
This is an Instrumentation Library, which instruments ASP.NET Core and collect metrics and traces about incoming web requests.

AWS SDK client instrumentation
This package provides AWS SDK client instrumentation

AWS .NET SDK for Lambda
This repo contains SDK to instrument Lambda handler to create incoming span.

Azure SDK Instrumentation
Instrumentation for Azure SDK for .NET (Track 2 libraries).

Cassandra Instrumentation for OpenTelemetry
This is an Instrumentation Library, which instruments CassandraCSharpDriver

Elasticsearch Client Instrumentation for OpenTelemetry .NET
Automatically instruments events emitted by the NEST/Elasticsearch.Net client library.

EntityFrameworkCore Instrumentation for OpenTelemetry .NET
Automatically instruments the outgoing database requests from Microsoft.EntityFrameworkCore.

EventCounters Instrumentation for OpenTelemetry .NET
This is an Instrumentation Library , which republishes EventCounters using OpenTelemetry Metrics API.

dotnet instrumentation

Native Instrumentation

- Jenkins Plugin
- GitHub Actions Step
- Concourse CI (experimental)
- Zuul OTel Proposal

open-telemetry / oteps Public

Watch 81 | Fork 143 | Star 263

Code | Issues 19 | Pull requests 28 | Actions | Projects | Security | Insights

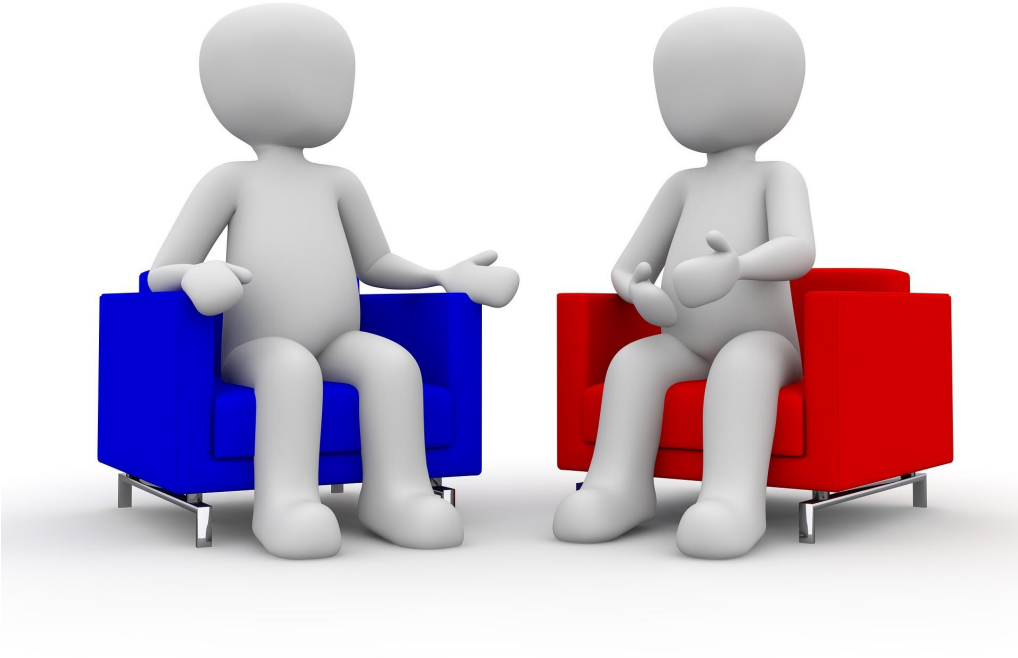
OpenTelemetry Proposal: Introduce semantic conventions for CI/CD observability #223

Code

Open **horovits** wants to merge 5 commits into `open-telemetry:main` from `horovits:main`

Conversation 35 | Commits 5 | Checks 1 | Files changed 1

+75 −0



OpenTelemetry

@paigerduty

chronosphere

Thanks!

You can find me at

- paigerduty@hachyderm.io
- paigerduty@chronosphere.io

Slides from SlidesCarnival, pics from Pixabay

BONUS

Ship It! – Episode #27

OpenTelemetry in your CI/CD

with Cyrille Le Clerc & Oleg Nenashev

Cyrille

Oleg

Gerhard

Paige's tale of how a 1-line config change caused a SEV-1....