From a pipeline to a government cloud

Toby Lorne
SRE @ GOV.UK Platform-as-a-Service

www.toby.codes
github.com/tlwr
github.com/alphagov

From a pipeline to a government cloud

How the UK government deploy a Platform-as-a-Service using Concourse, an open-source continuous thing-doer

From a pipeline to a government cloud

1. GOV.UK PaaS overview
2. Concourse overview
3. Pipeline walkthrough
4. Patterns and re-use

What is GOV.UK PaaS?

What is a Platform-as-a Service?

What are some challenges with digital services in government?

How does GOV.UK PaaS make things better?

What is a PaaS?

Run, manage, and maintain apps and backing services

Without having to buy, manage, and maintain infrastructure or needing specialist expertise

Here is my source code

Run it for me in the cloud

I do not care how

Deploy to production safer and faster

Reduce waste in the development process

Proprietary	Open source
Heroku	Cloud Foundry
Pivotal application service	DEIS
EngineYard	Openshift
Google App Engine	kf
AWS Elastic Beanstalk	Dokku
Tencent BlueKing	Rio

UK-based web hosting for government services

Focus on building your service, not managing infrastructure.

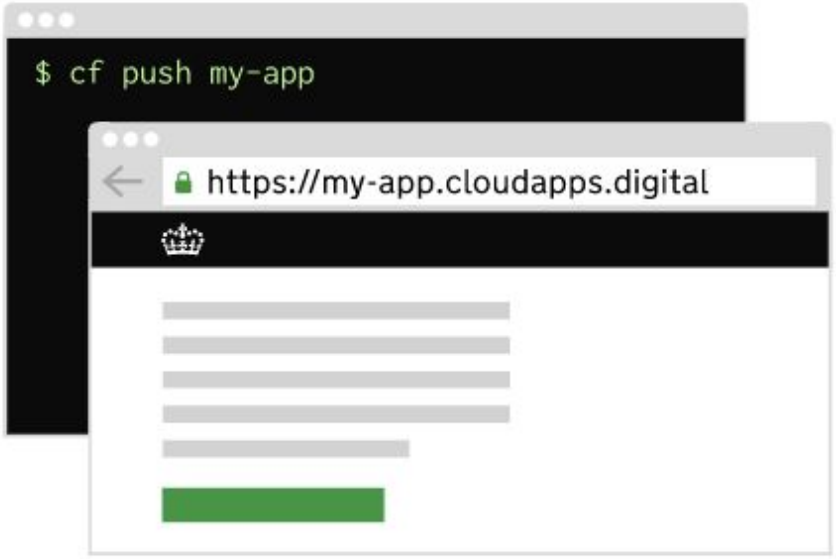
See how to get started

Deploy applications without infrastructure specialists

You can get something up on the Internet in minutes, without depending on a webops team.

The GOV.UK PaaS team manages the underlying platform, so you can:

- make the best use of your developers' time
- focus your budget on your applications



Why does government need a PaaS?

UK-based web hosting for government services

Government should focus on building useful services, not managing infrastructure

Enable teams to create services faster

Reduce the cost of procurement and maintenance

An opinionated platform promotes consistency

Communication within large bureaucracies can be slow

Diverse app workloads are impossible to reason about

Highly leveraged team requires trust and autonomy

Only able to do this because of open source software and communities

APPS	SERVICES	MANAGEMENT
API + CLI provided by Cloud Foundry	Service brokers OSB specification compliant	Operational metrics User management Billing

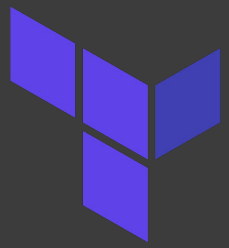
BOSH

Concourse

Grafana

Terraform

Prometheus



Terraform

terraform.io

Infrastructure as code, for provisioning arbitrary resources

Versatile tool for managing cloud infrastructure

BOSH

bosh.io

Release engineering, VM provisioning and lifecycling management

Very specific use-case, but very good at it

Steep learning curve, high reward

Prometheus

prometheus.io

Metric collection, storage, and query

Large open-source ecosystem

Multi-dimensional labels enable a rich query language

Grafana

grafana.com

Visualisation and dashboarding tool

Good for aggregating multiple data sources for display

What is Concourse?

Concourse is an open-source continuous thing-doer

“A thing which does things, sometimes continuously”

concourse-ci.org

A general approach to automation, with extensibility as the primary design goal


PIPELINE
RESOURCE
JOB
TASK

Pipelines

Directed acyclic graph, not just read left-to-right

Contain resources and jobs

Written in YAML

Automatically visualised in the web UI

Jobs

Can run in parallel, or in series

Composed of steps

Steps are compositions of running tasks, flow-control, and resource interactions

Tasks

Specific

Represent doing a thing
(unit of code execution)

Are stateless
(in the long run)

Code is executed inside an
ephemeral environment,
based on a container image

Resources

Generic

Defined by
resource types

Immutable, idempotent,
external source of truth

“a single object with a
linear version sequence”

Step flow control

`in_parallel` is a step for running other steps in parallel, e.g. clone many git repos concurrently

`do` is a step for running steps in series

`try` is a step which will not fail a job if it does not succeed

`set_pipeline` will update a pipeline's config

Resource interactions

`get`ting a resource pulls external state from the source of truth

`put`ting *a resource* step pushes local state to the source of truth

Periodically resources are `check`ed for new *versions*

Task examples

Build a container image

Compile release artefacts

Run automated tests

Generate release notes

Resource types

Git/Image repository

File in object storage

Semantic version

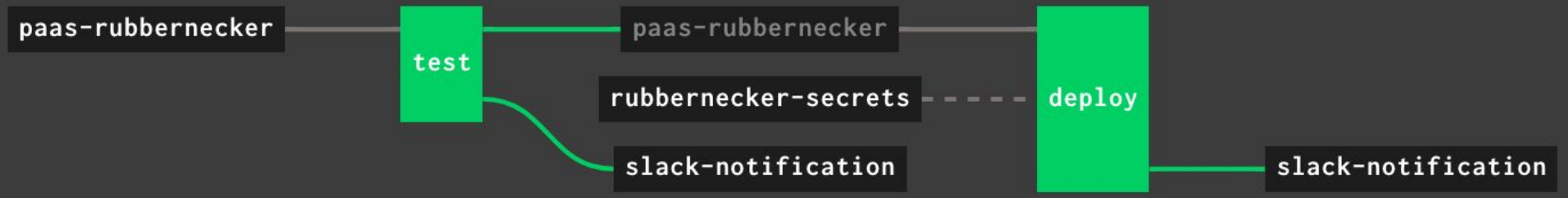
Distributed lock/pool

GitHub release

Terraform deployment

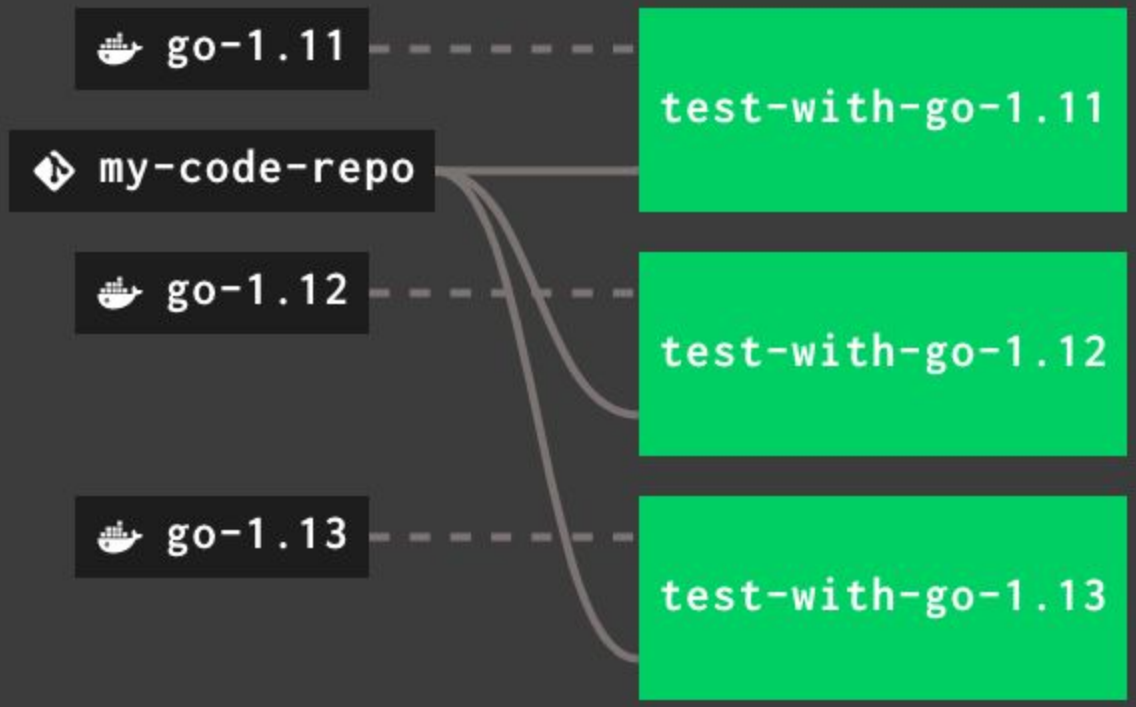
Cloud Foundry app

Simple continuous deployment

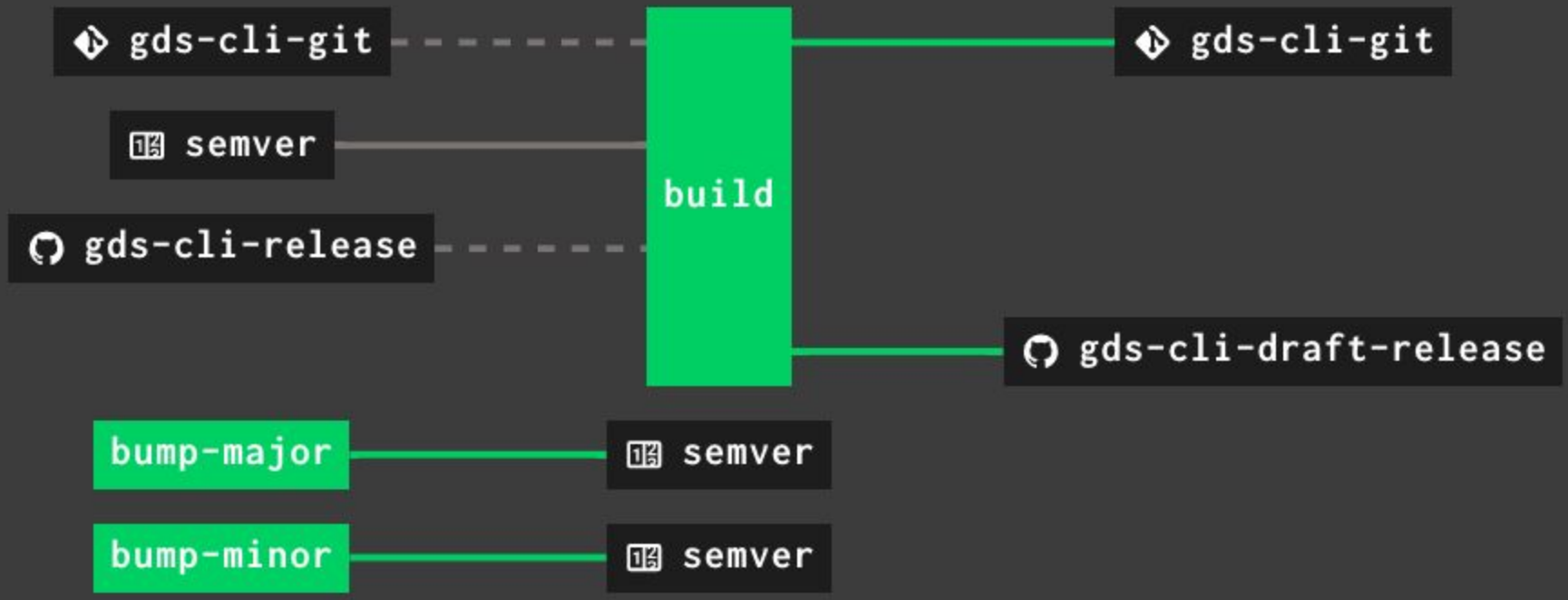


Multi-environment continuous deployment

A branching pipeline



“Autonomate” a manual release process

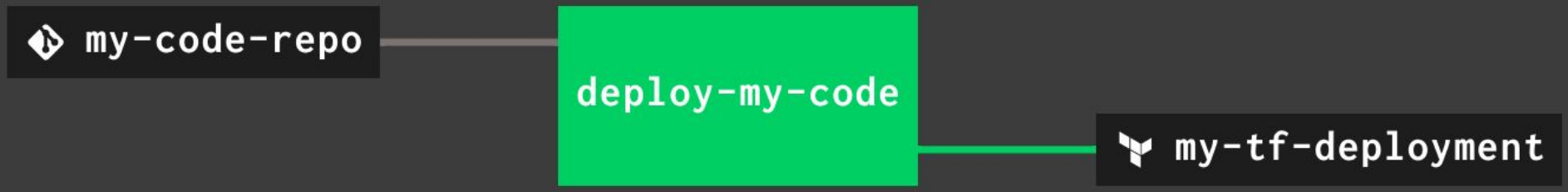


“Show me the YAML”

Example:

Continuously deploy terraform

Continuously deploy terraform



```
resources:
- name: my-code-repo
  …
- name: my-tf-deployment
  …

jobs:
- name: deploy-my-code
  …
```

```
resources:
- name: my-code-repo
  type: git
  icon: git
  source:
    branch: develop
    uri: https://github.com/x/y.git
- name: my-tf-deployment
  …

jobs: …
```

```
resources:
- name: my-code-repo
  …
- name: my-tf-deployment
  type: terraform
  icon: terraform
  source: …

jobs:
- name: deploy-my-code
  …
```

```
resources: …

jobs:
- name: deploy-my-code
  serial: true
  plan:
    - get: my-code-repo
      trigger: true
    - put: my-tf-deployment
```

This pipeline will deploy terraform whenever the develop branch changes

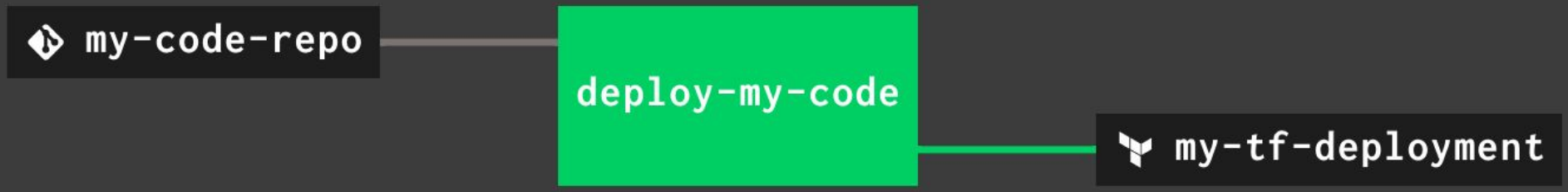
((secrets)) are retrieved from a credentials provider when they are needed

Credential providers:
- Credhub
- AWS SSM
- Kubernetes
- Hashicorp Vault

```
resources:
- name: my-code-repo
  type: git
  icon: git
  source:
    branch: develop
    uri: https://github.com/x/y.git

- name: my-tf-deployment
  type: terraform
  icon: terraform
  source:
    backend_type: s3
    backend_config:
      bucket: my-prod-bucket
      key: tfstate/my-deployment.tfstate
      region: eu-west-2
      access_key: ((aws_access_key_id))
      secret_key: ((aws_secret_access_key))

jobs:
- name: deploy-my-code
  serial: true
  plan:
    - get: my-code-repo
      trigger: true
    - put: my-tf-deployment
```

```
fly login \
    --target my-concourse \
    --open-browser

fly set-pipeline \
    --pipeline deployment \
    --config cd-tf.yml
```

Continuously deploy terraform



Continuously deploy terraform (oh no)

```
resources:
- name: my-code-repo
  …
- name: my-tf-deployment
  …
- name: project-slack-channel
  type: slack
  icon: slack
  source: …

jobs: …
```

```
…
put: my-tf-deployment
on_failure:
  put: project-slack-channel
    params:
      channel: '#develop'
      icon_emoji: ':airplane:'
      text: |
        Build $BUILD_NAME failed.
        Check it out at: …
```

Continuously deploy terraform with failure notifications

Extending Concourse

Build your own resource

An OCI compatible image, hosted somewhere Concourse can access.

Which should contain up to three executables:
- /opt/resource/check
- /opt/resource/in
- /opt/resource/out

Resource interactions

`check`
is executed periodically

`in`
is executed for a `get` step

`out`
is executed for a `put` step

A git repo flies

Through a concourse pipeline

It becomes a cloud

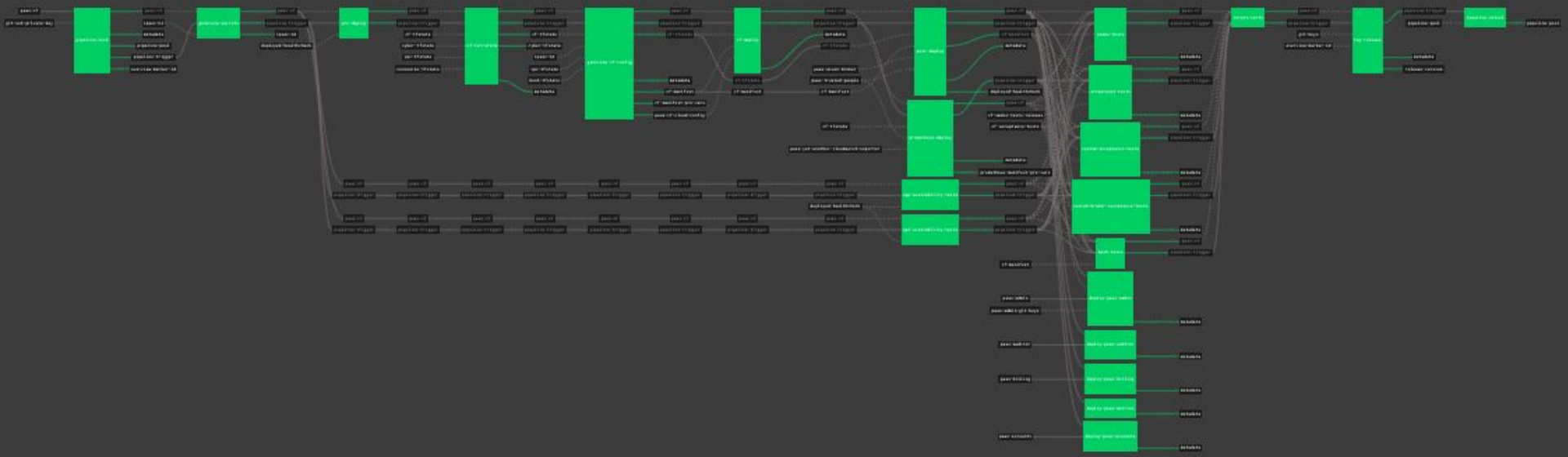
What do we care about?

App availability (~99.99%)

API availability (~99.9%)

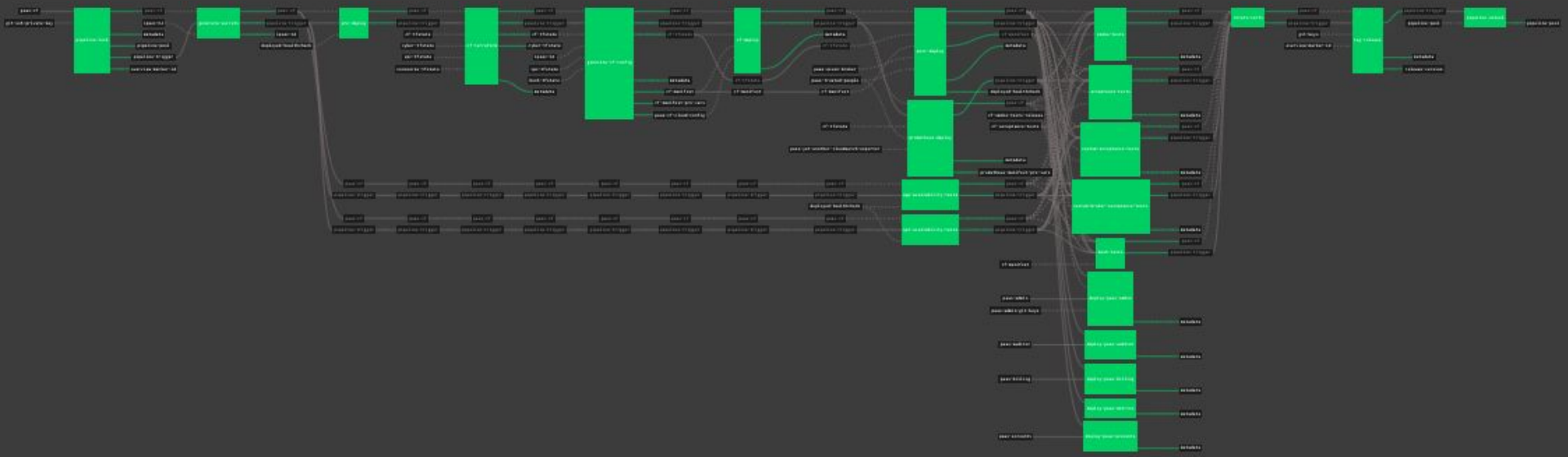
Safety and reproducibility are achieved through *autonomation*

GOV.UK PaaS deployment pipeline

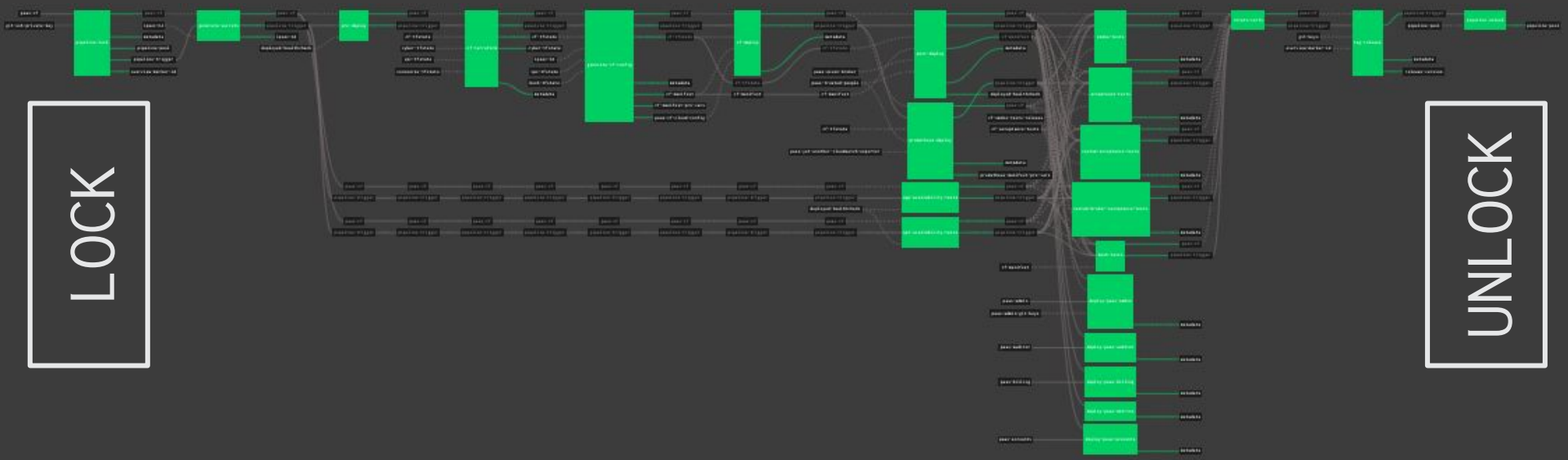


GOV.UK PaaS deployment pipeline

GOV.UK PaaS deployment pipeline

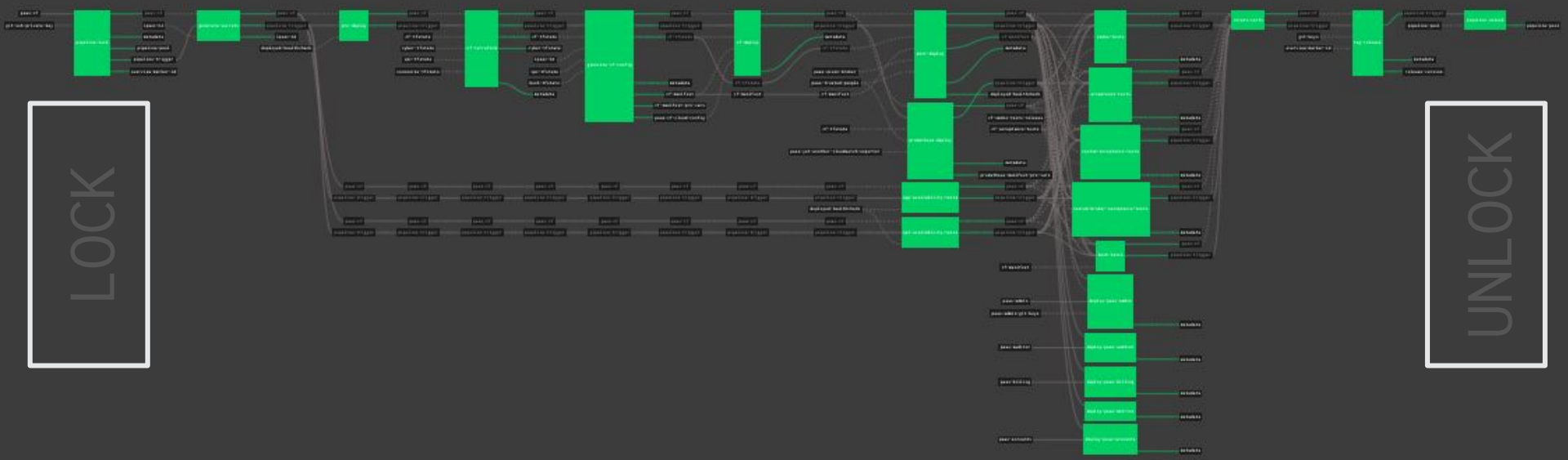


GOV.UK PaaS deployment pipeline



GOV.UK PaaS deployment pipeline

CONFIG



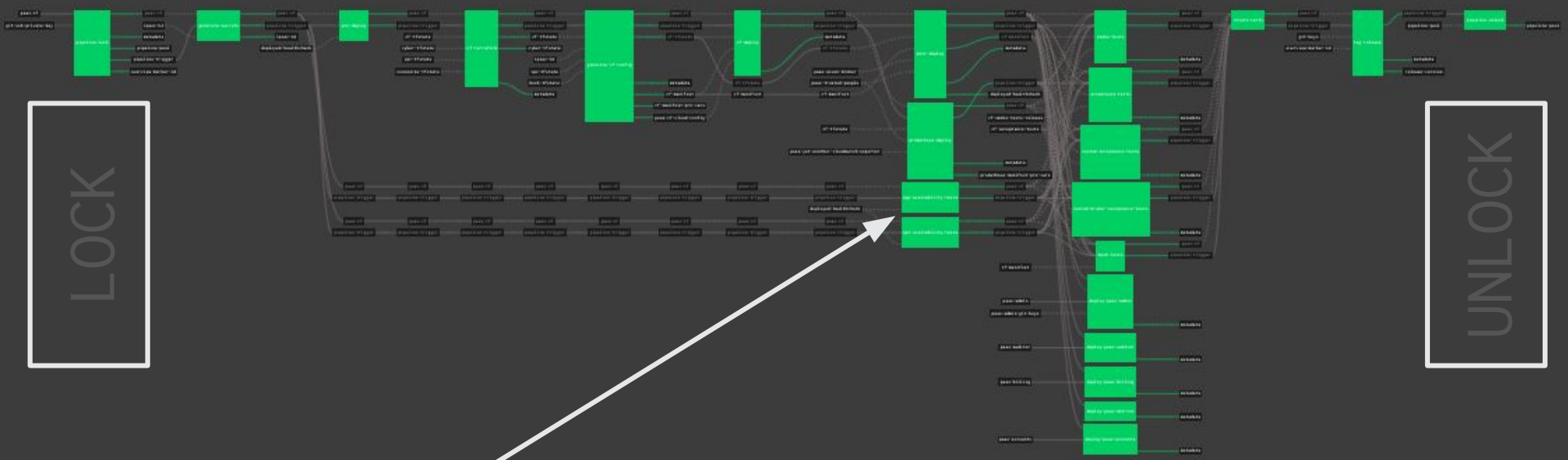
LOCK

UNLOCK

GOV.UK PaaS deployment pipeline

CONFIG

WAIT



LOCK

UNLOCK

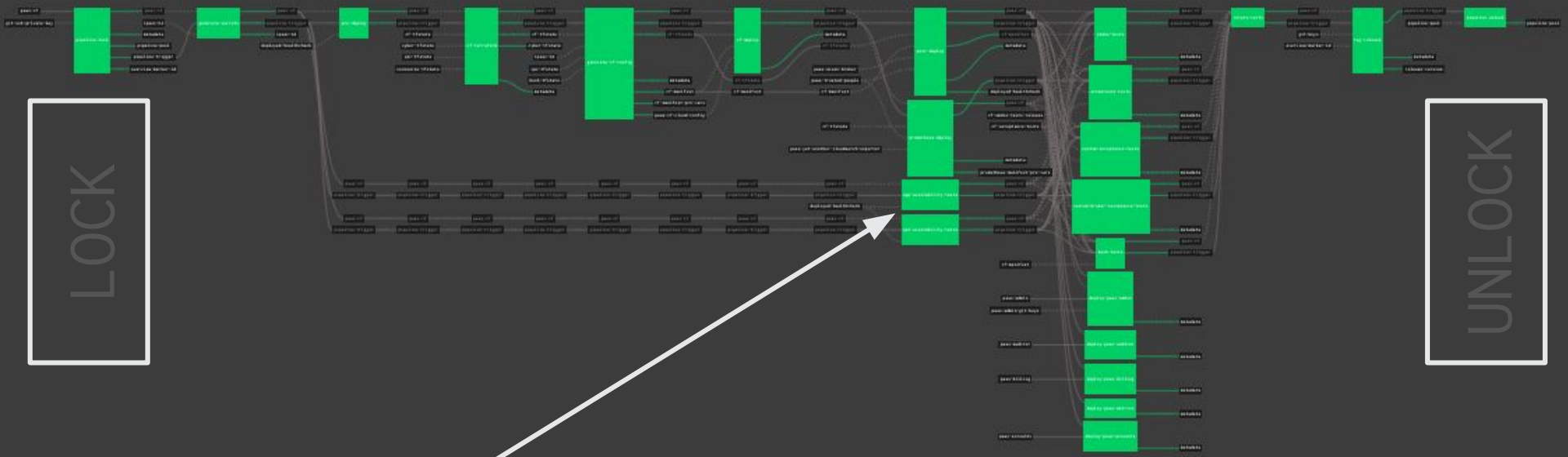
AVAILABILITY TESTS

GOV.UK PaaS deployment pipeline

CONFIG

WAIT

TERRAFORM



LOCK

UNLOCK

AVAILABILITY TESTS

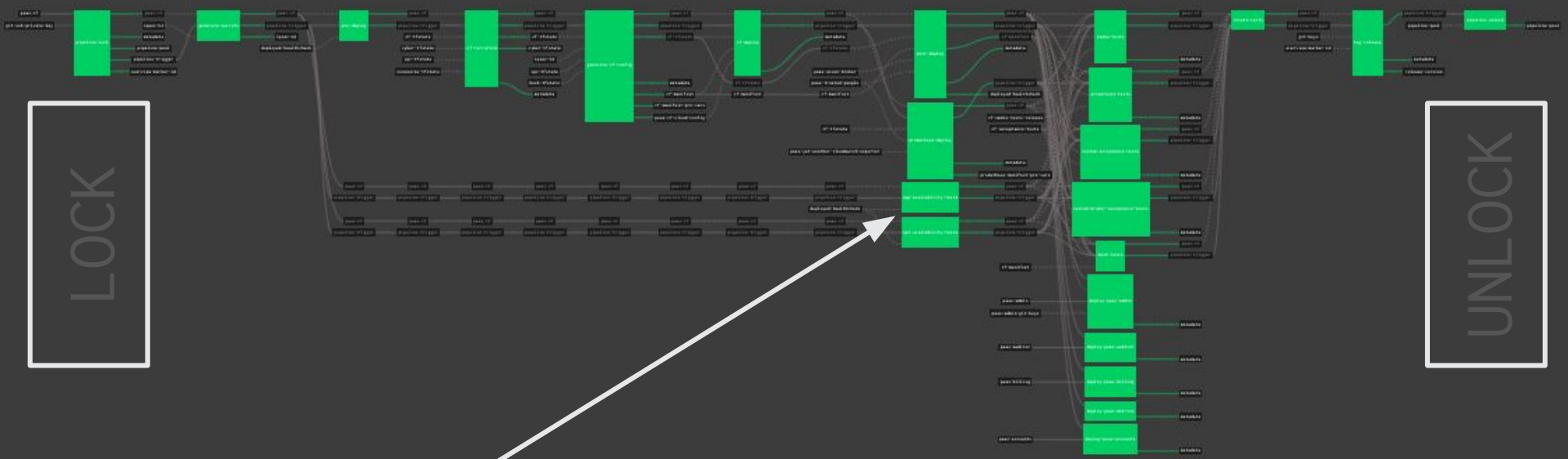
GOV.UK PaaS deployment pipeline

CONFIG

WAIT

TERRAFORM

DEPLOY CF



LOCK

UNLOCK

AVAILABILITY TESTS

GOV.UK PaaS deployment pipeline

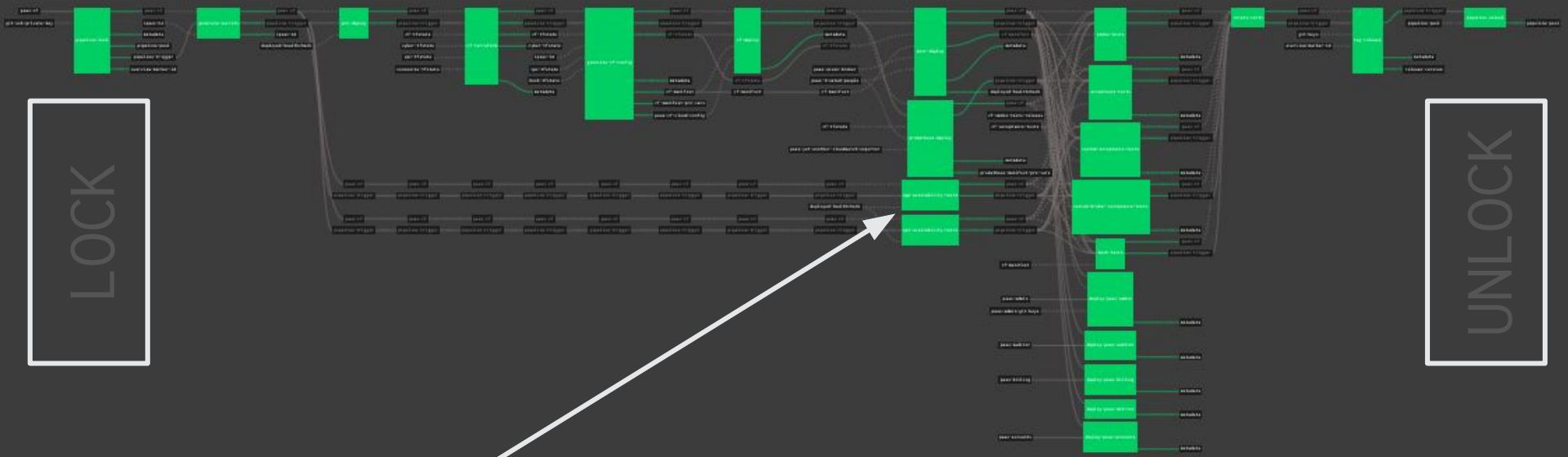
CONFIG

WAIT

TERRAFORM

DEPLOY CF

PROMETHEUS & BROKERS

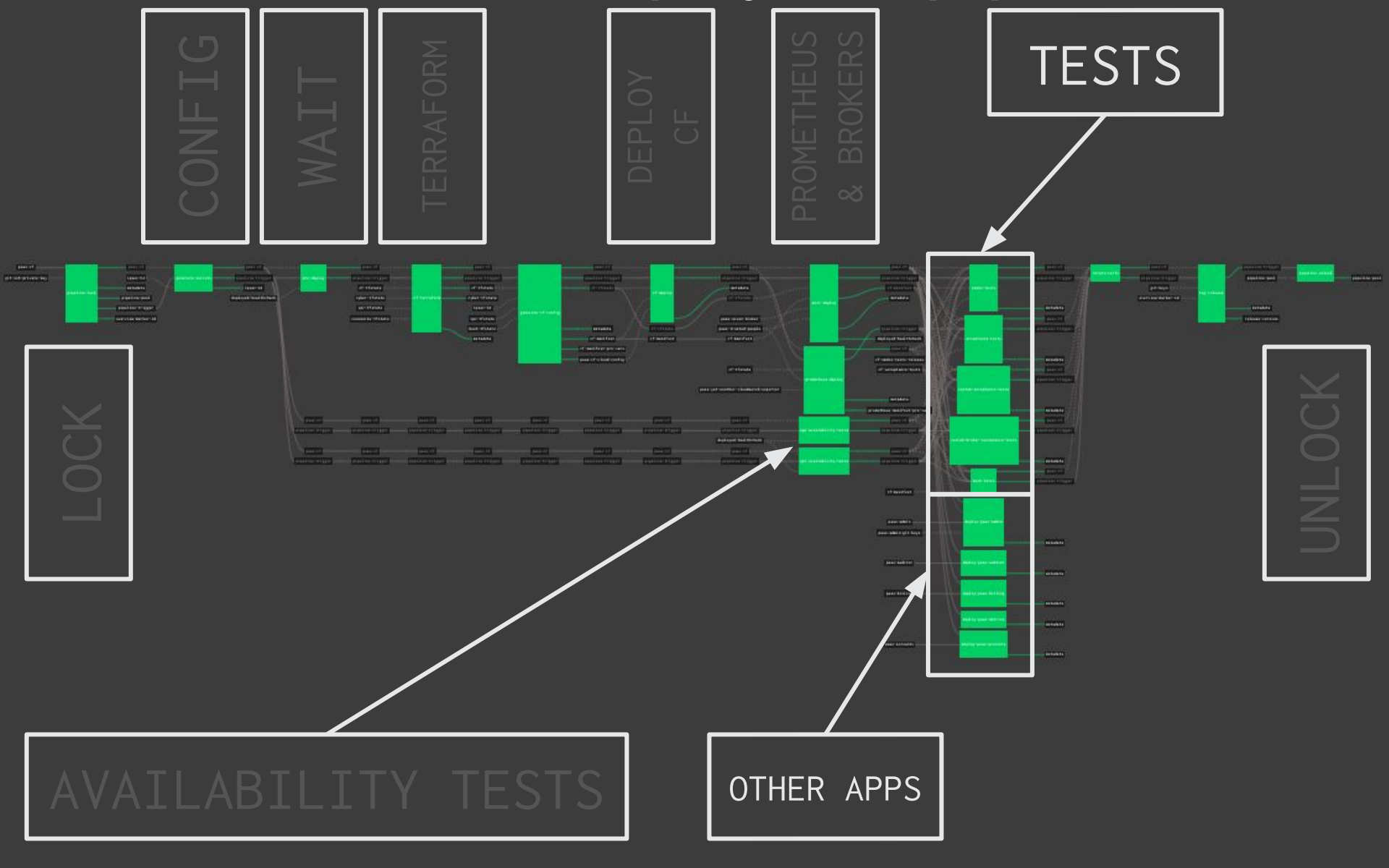


LOCK

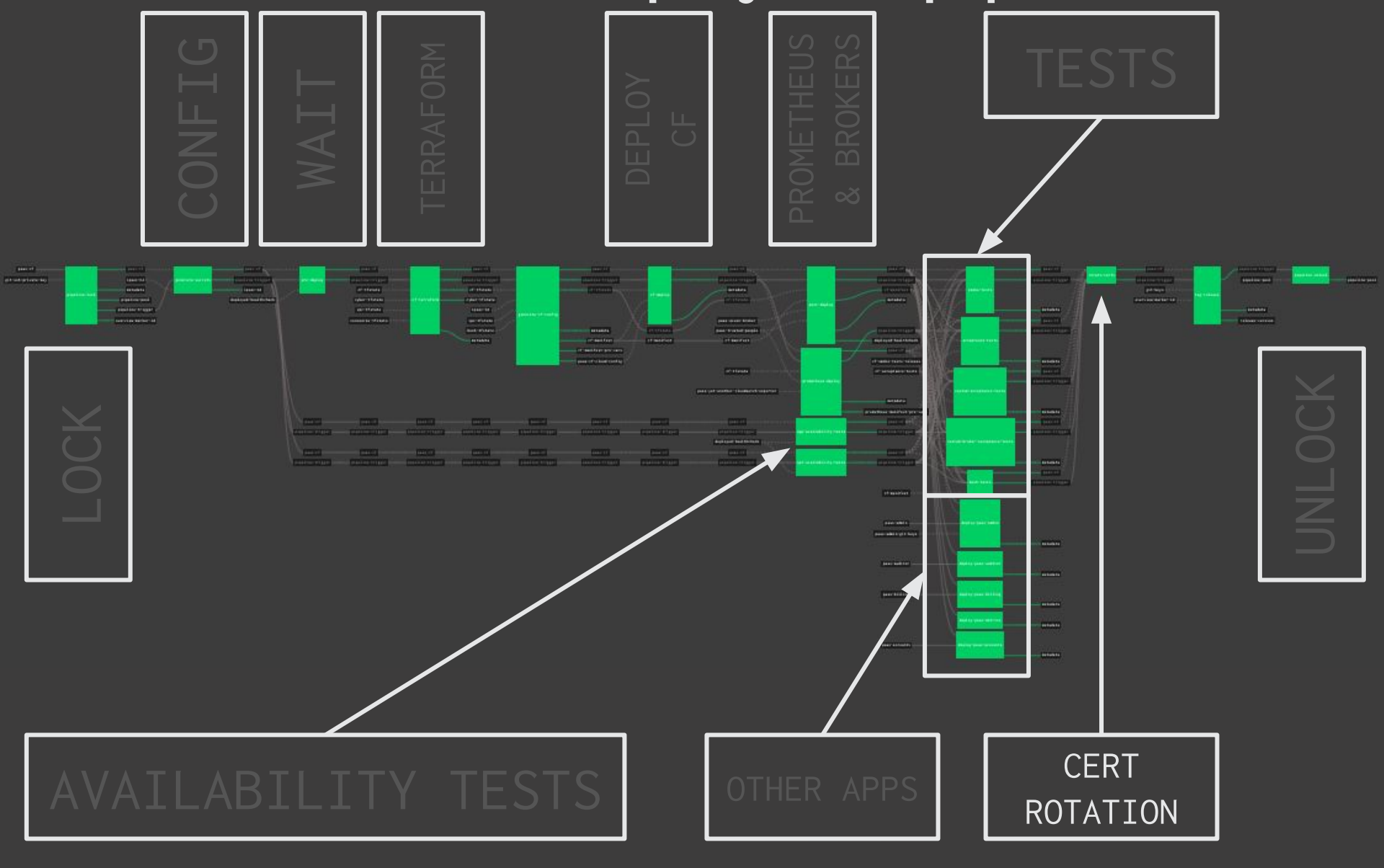
UNLOCK

AVAILABILITY TESTS

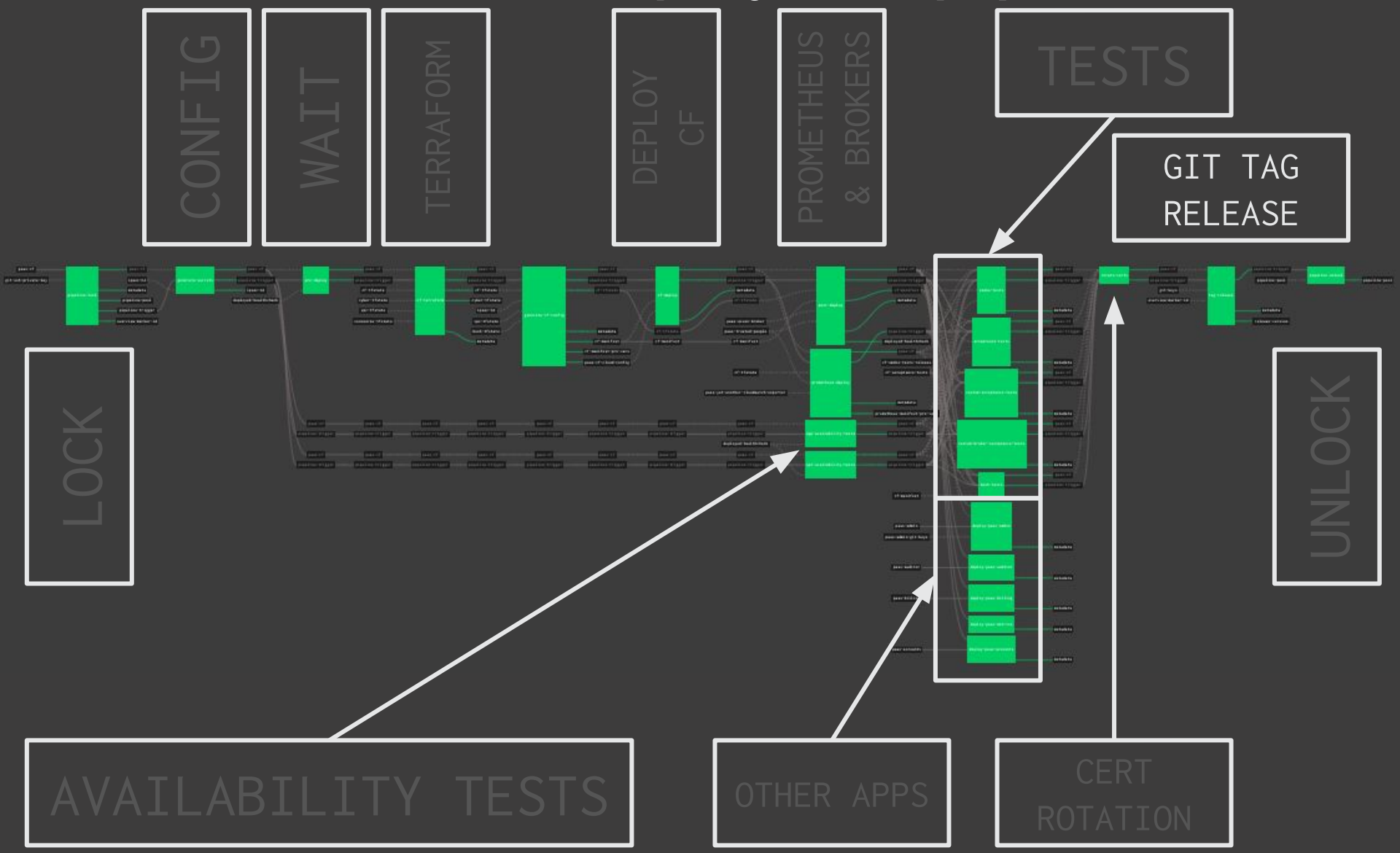
GOV.UK PaaS deployment pipeline



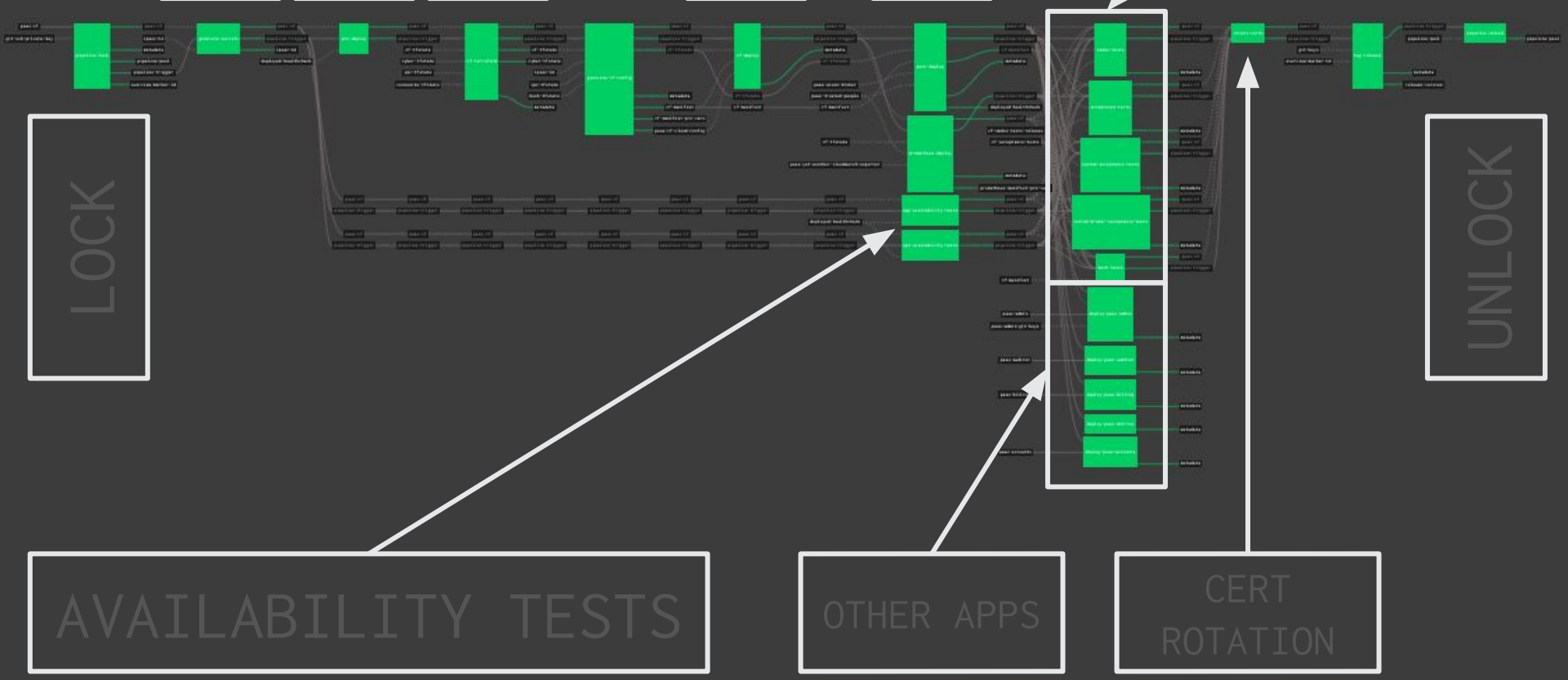
GOV.UK PaaS deployment pipeline



GOV.UK PaaS deployment pipeline



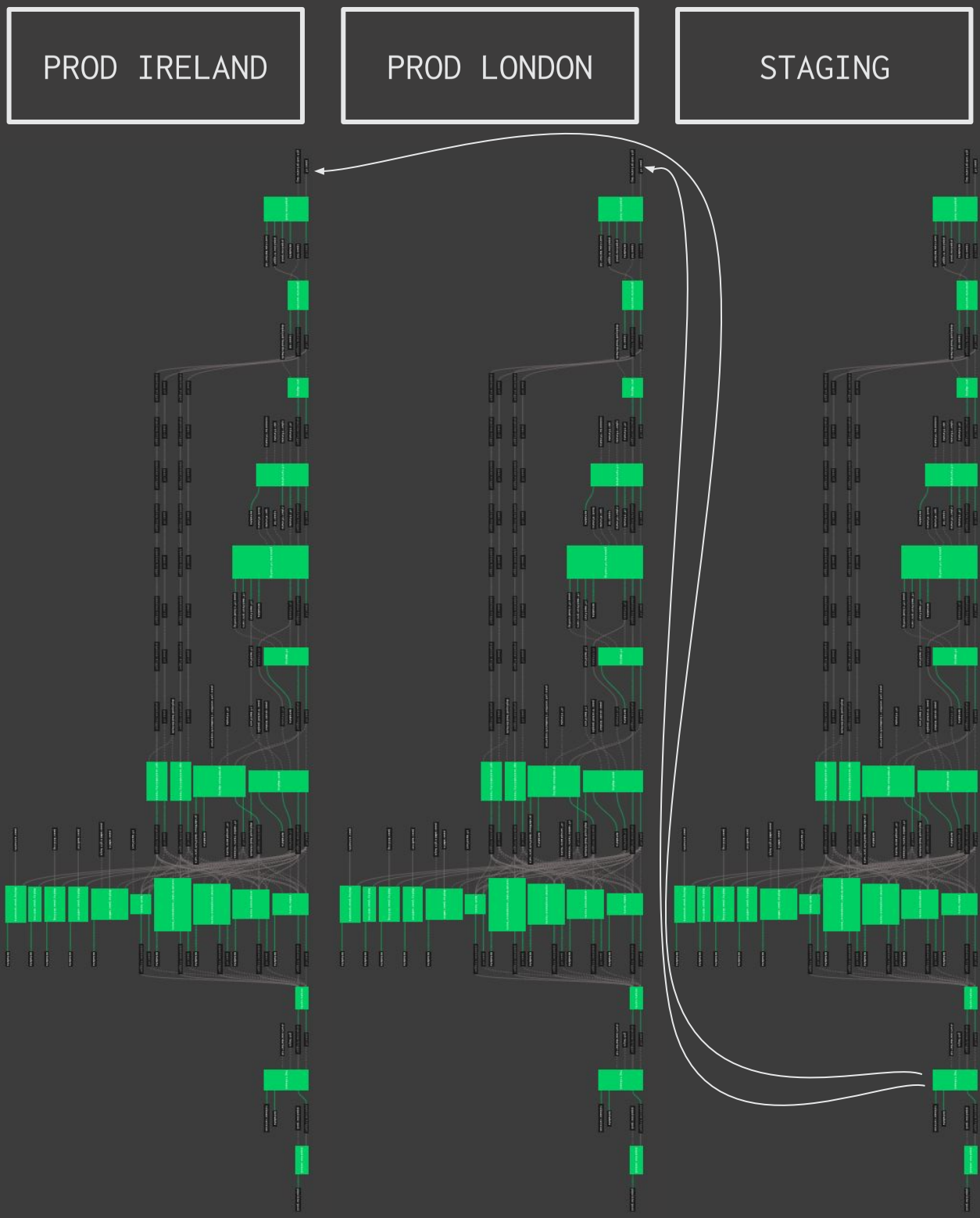
GOV.UK PaaS deployment pipeline



Now do it all again!

```
git merge --gpg-sign
→ Deploy staging
   → git tag
      → Deploy prod London
      → Deploy prod Dublin
```

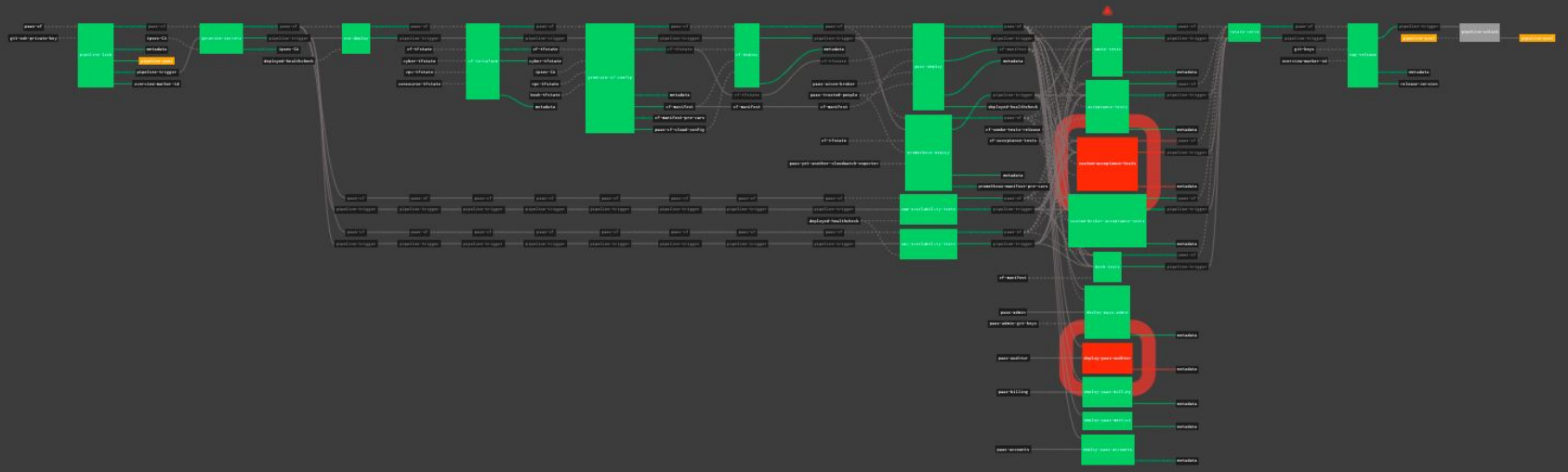
This process happens ~2.5x per day


PROD IRELAND
PROD LONDON
STAGING

Normal deployments are fully automated, so deploys are small, and occur often

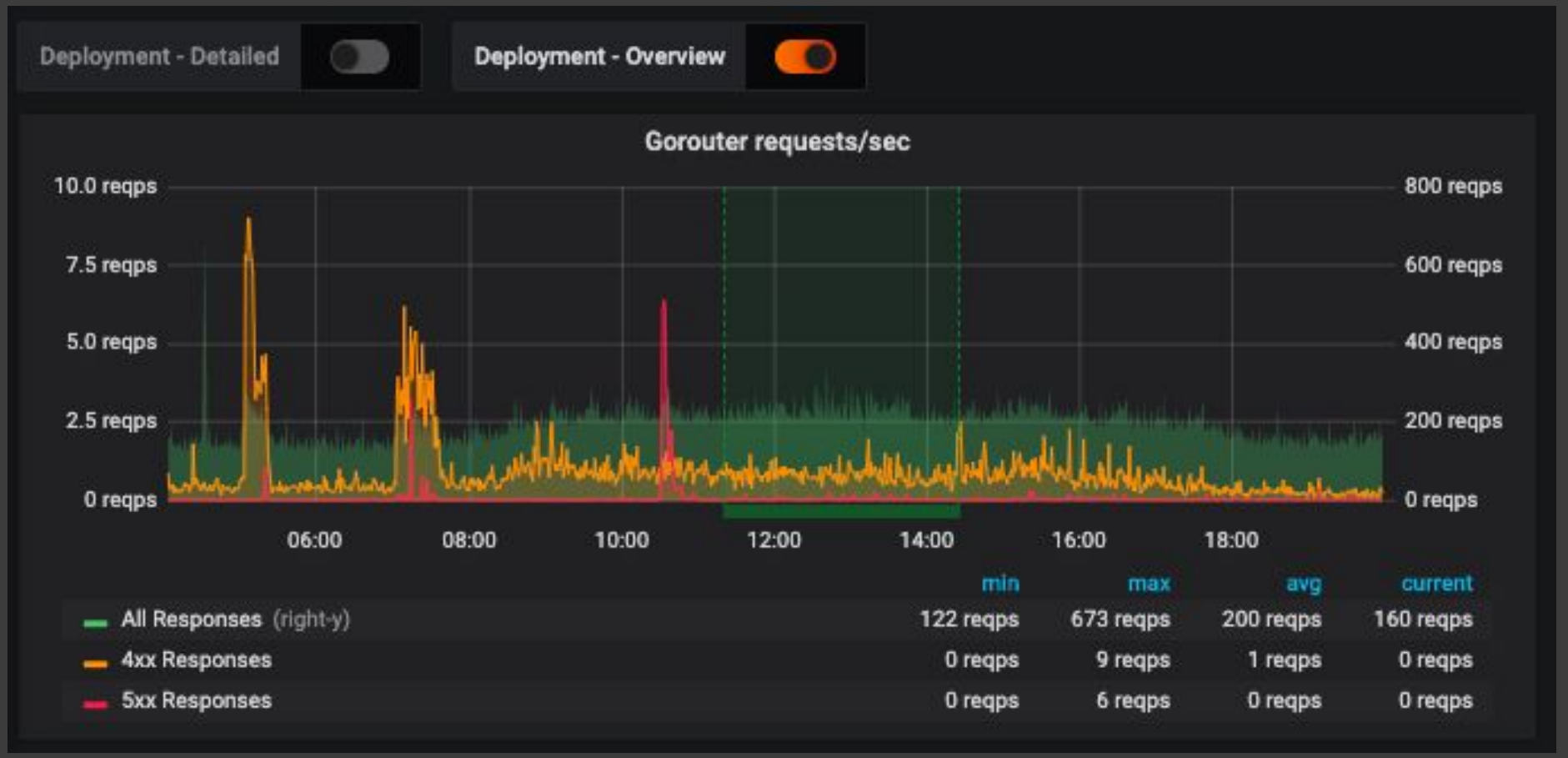
Deployments fail safely, due to locking, tests, *and BOSH*

The UI is “anger optimised” - *@vito*

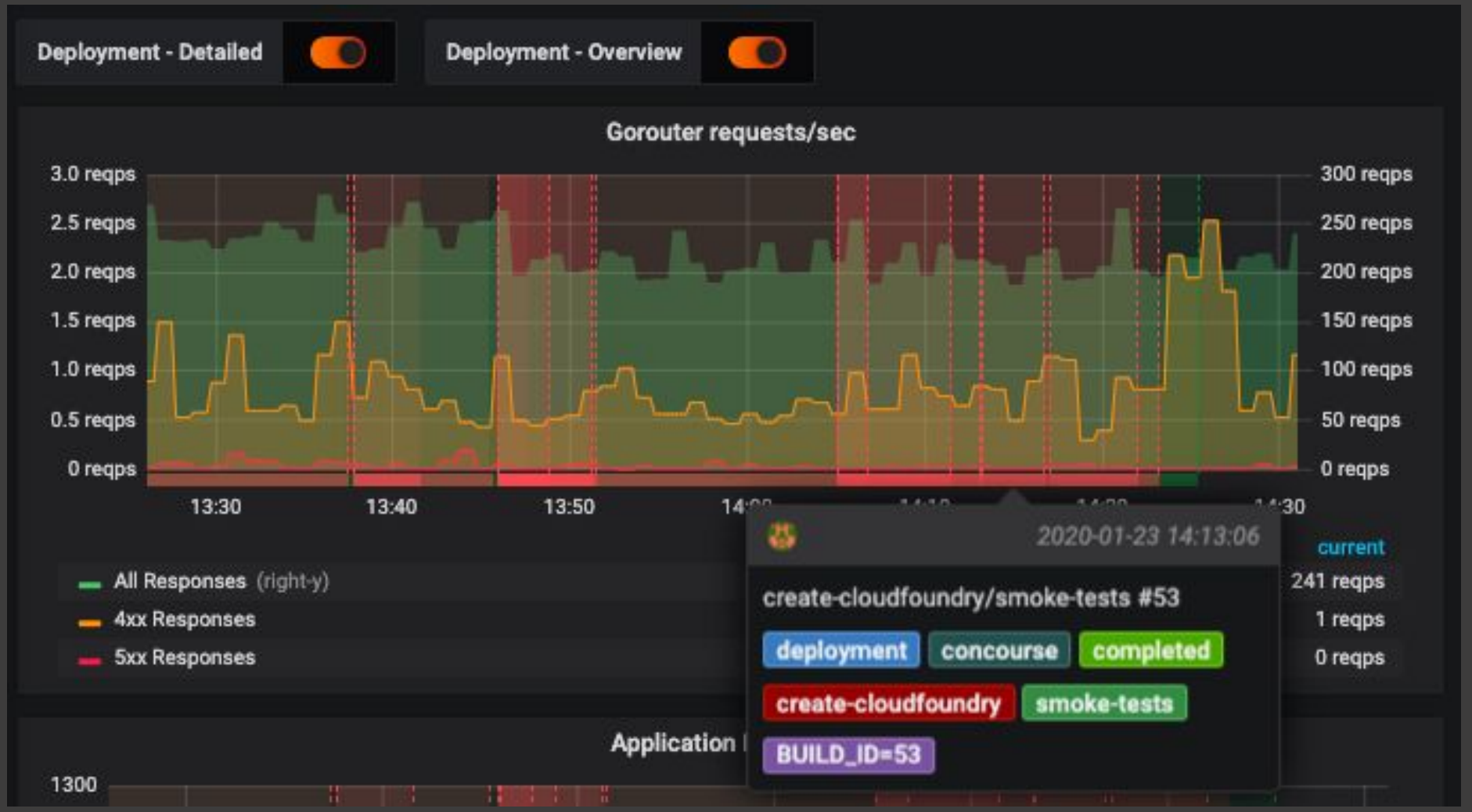


It is visually obvious* what state a pipeline is in, and if it is broken

Concourse and Grafana deployment overview annotations



Concourse and Grafana deployment overview details



Someone else's code

Is running in production

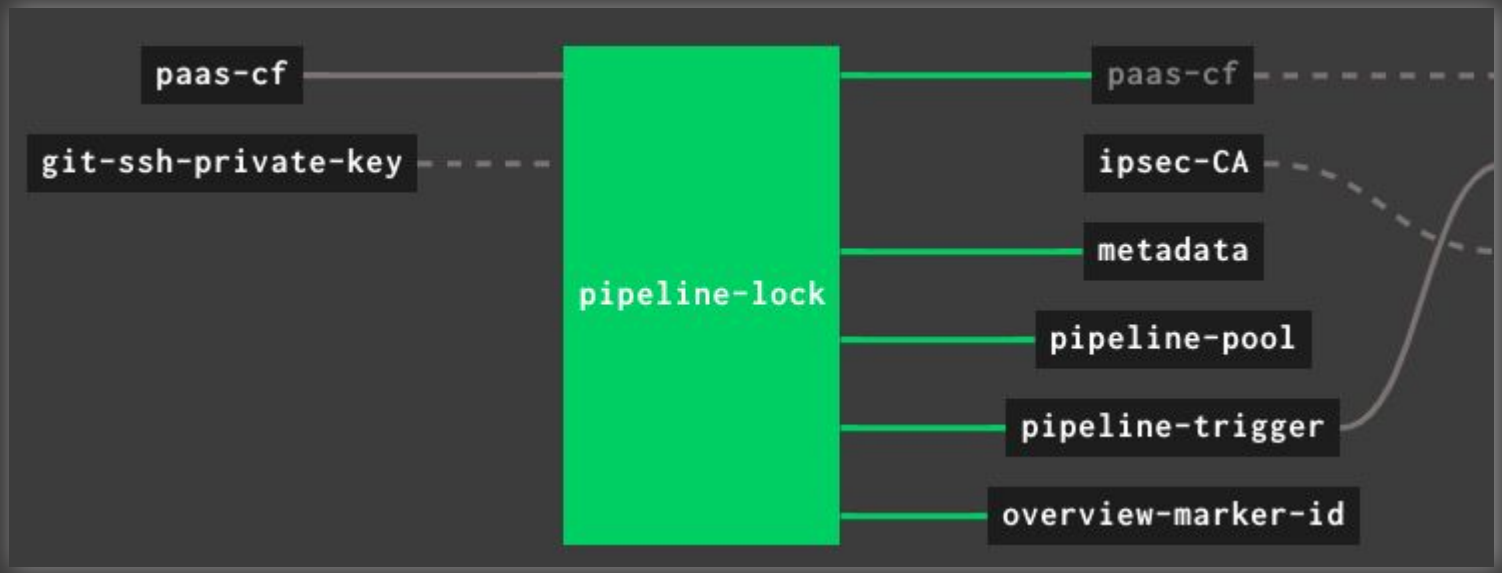
Can I re-use this?

Patterns and re-use, how?

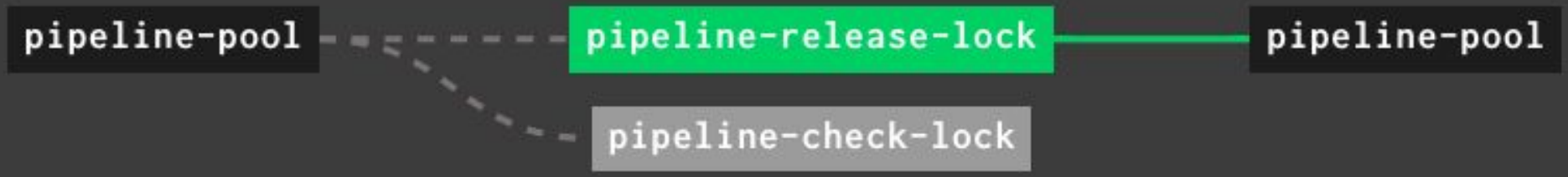
Concourse resource types available at `resource-types.concourse-ci.org`

Patterns
- Locks, pools, and counters
- Availability tests
- Metrics and annotations
- Releases and communications

Pools and locks



with controls for pipeline operators



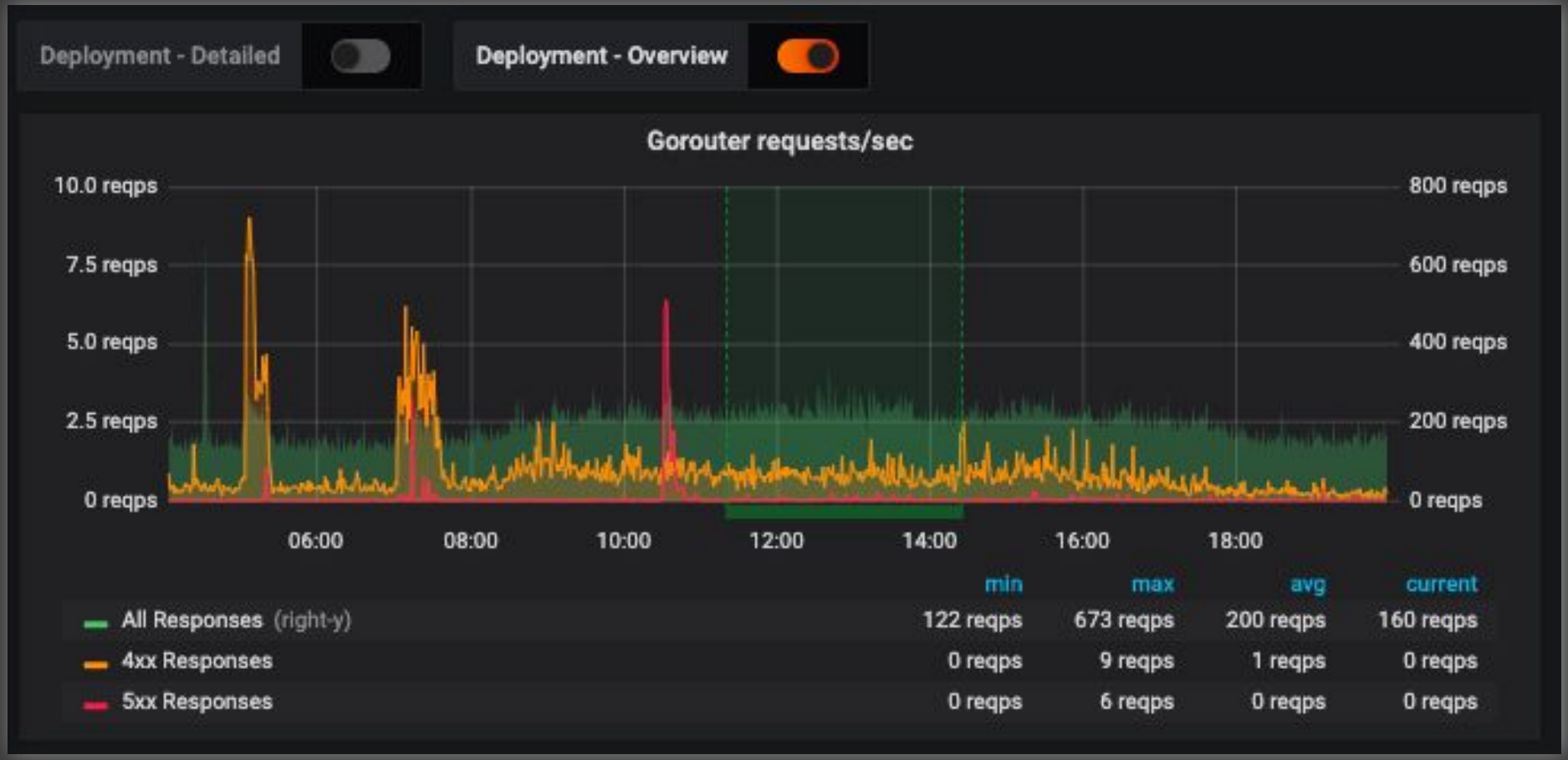
github.com/concourse/pool-resource

Availability tests

implemented as a task

github.com/tsenart/vegeta

Annotations



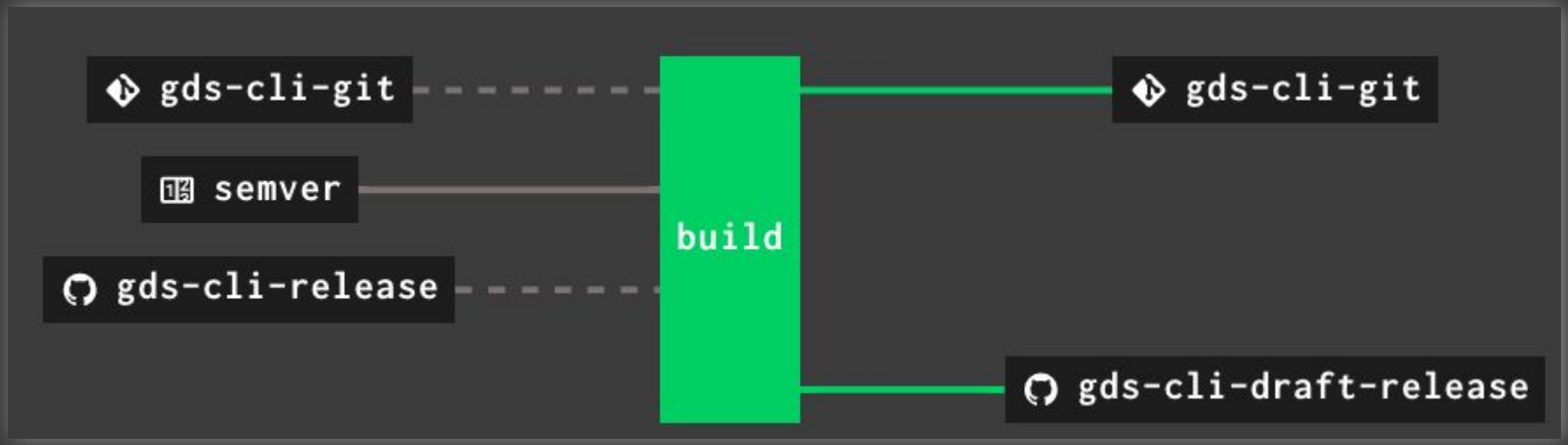
github.com/alphagov/paas-grafana-annotation-resource

Metrics

```
increase(
  concourse_builds_finished{
    exported_job="continuous-smoke-tests",
    status!="succeeded"
  }[30m]
) >= 1
```

concourse-ci.org/metrics.html

Release management



with controls for maintainers

github.com/concourse/github-release-resource
github.com/concourse/semver-resource

Communications

Please don’t rely on watching your pipelines

github.com/FidelityInternational/concourse-pagerduty-notification-resource
github.com/cloudfoundry-community/slack-notification-resource
github.com/hpcloud/hipchat-notification-resource
github.com/pivotal-cf/email-resource

That's Concourse!

Concourse is an open-source continuous thing-doer

"A thing which does things, sometimes continuously"

concourse-ci.org

From a pipeline to a government cloud

Toby Lorne
SRE @ GOV.UK Platform-as-a-Service

www.toby.codes
github.com/tlwr
github.com/alphagov