Dynamically Testing Individual Microservice Releases In Production

Matt Turner

KubeCon + CloudNativeCon, Detroit MI | October 2022

@mt165 | mt165.co.uk

Outline

- Microservice topologies
- Continuous Deployment / Release
- Testing in production
- Implementation
- Prior art
- Istio
- Override-operator
- Demo
- Challenges and future work
- Conclusions

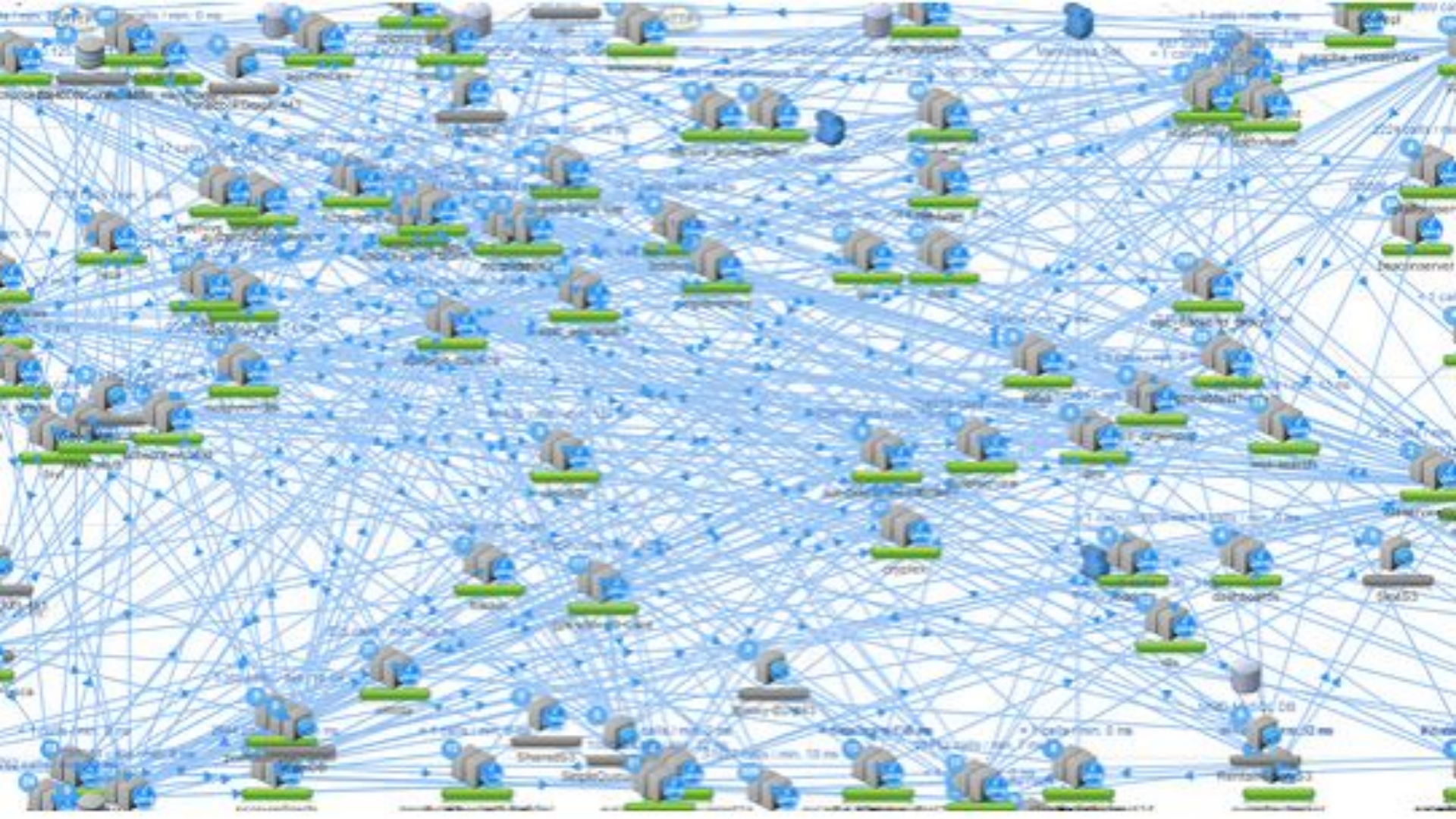
tetrate

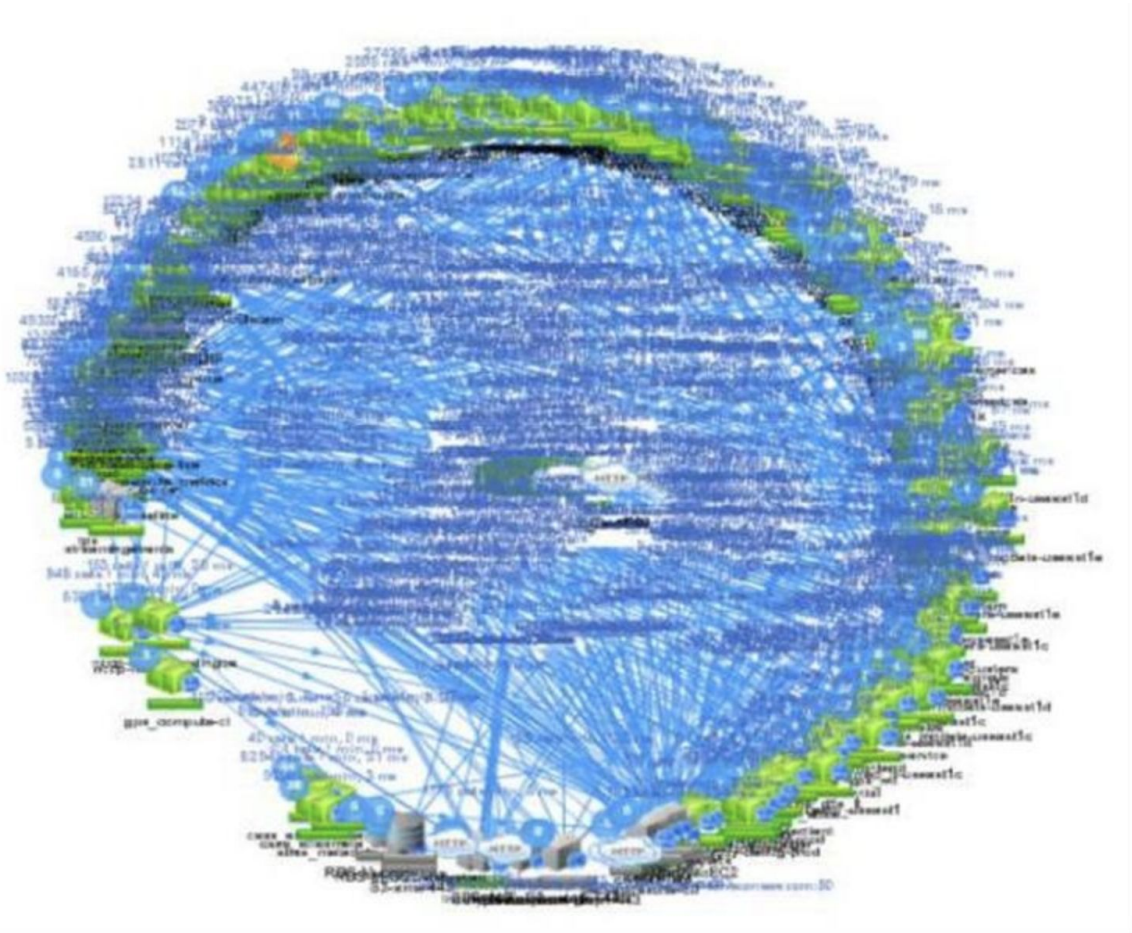
THE ENTERPRISE SERVICE MESH COMPANY

About you

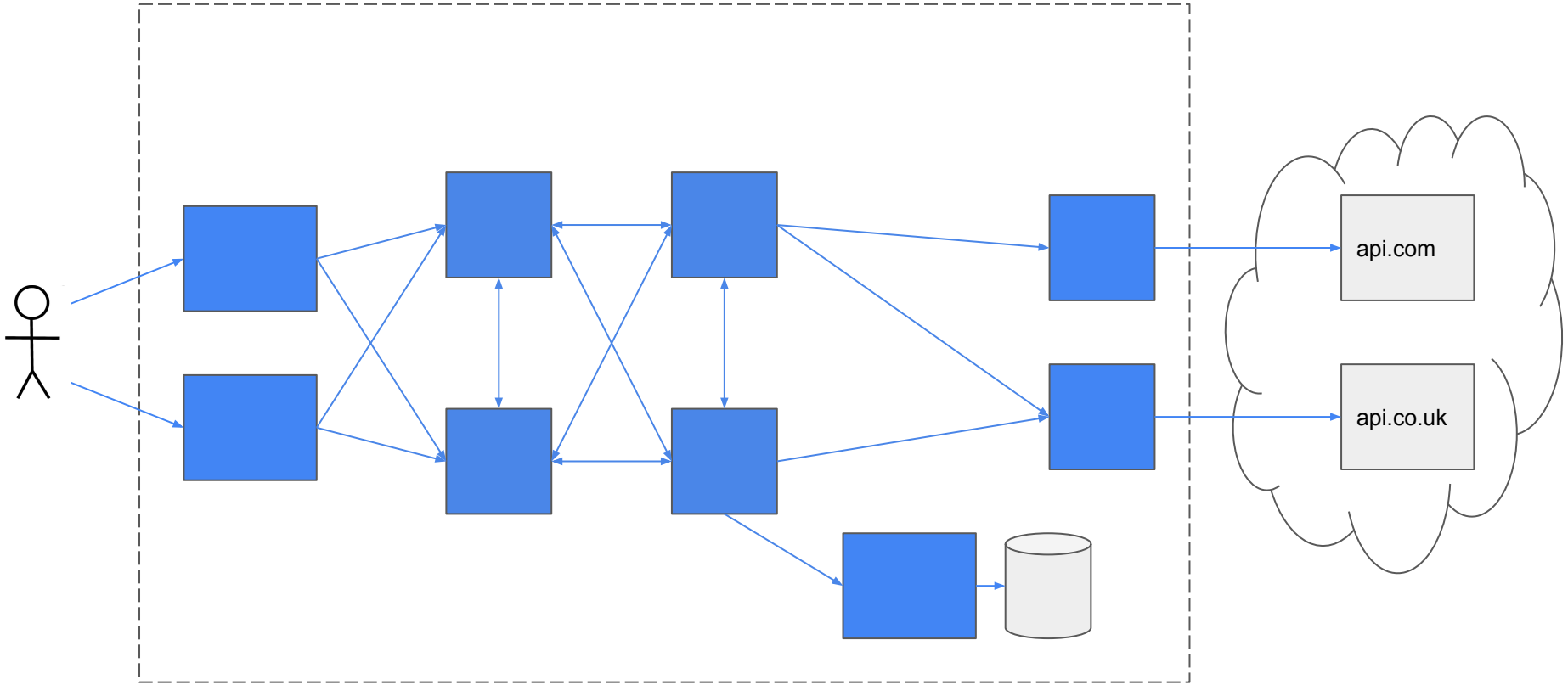
- Never used Istio
- Beginner
- Expert

Microservice Topologies

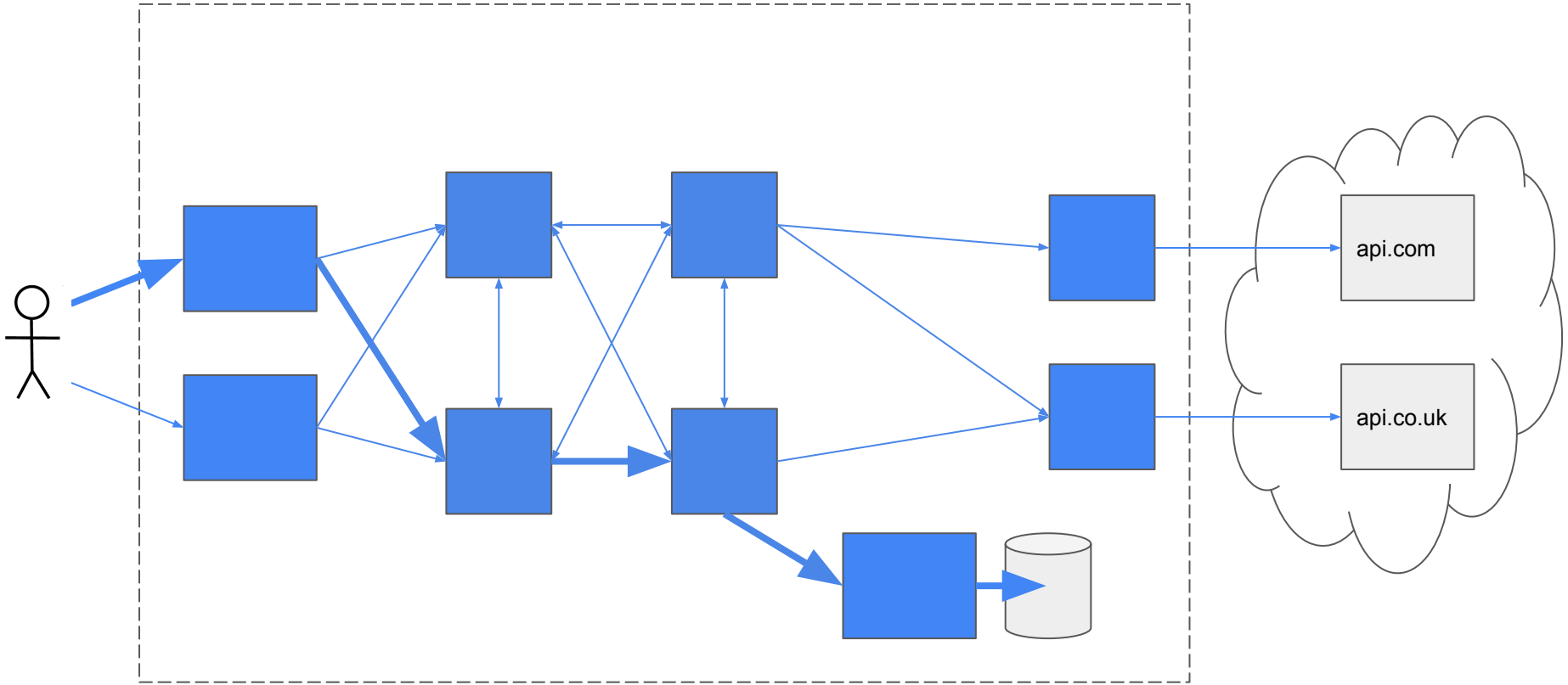




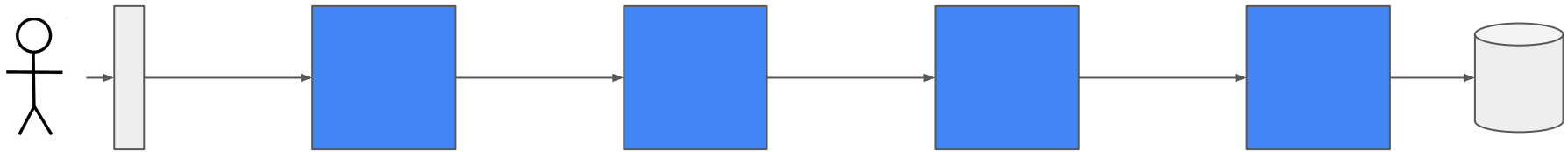
Service Chains



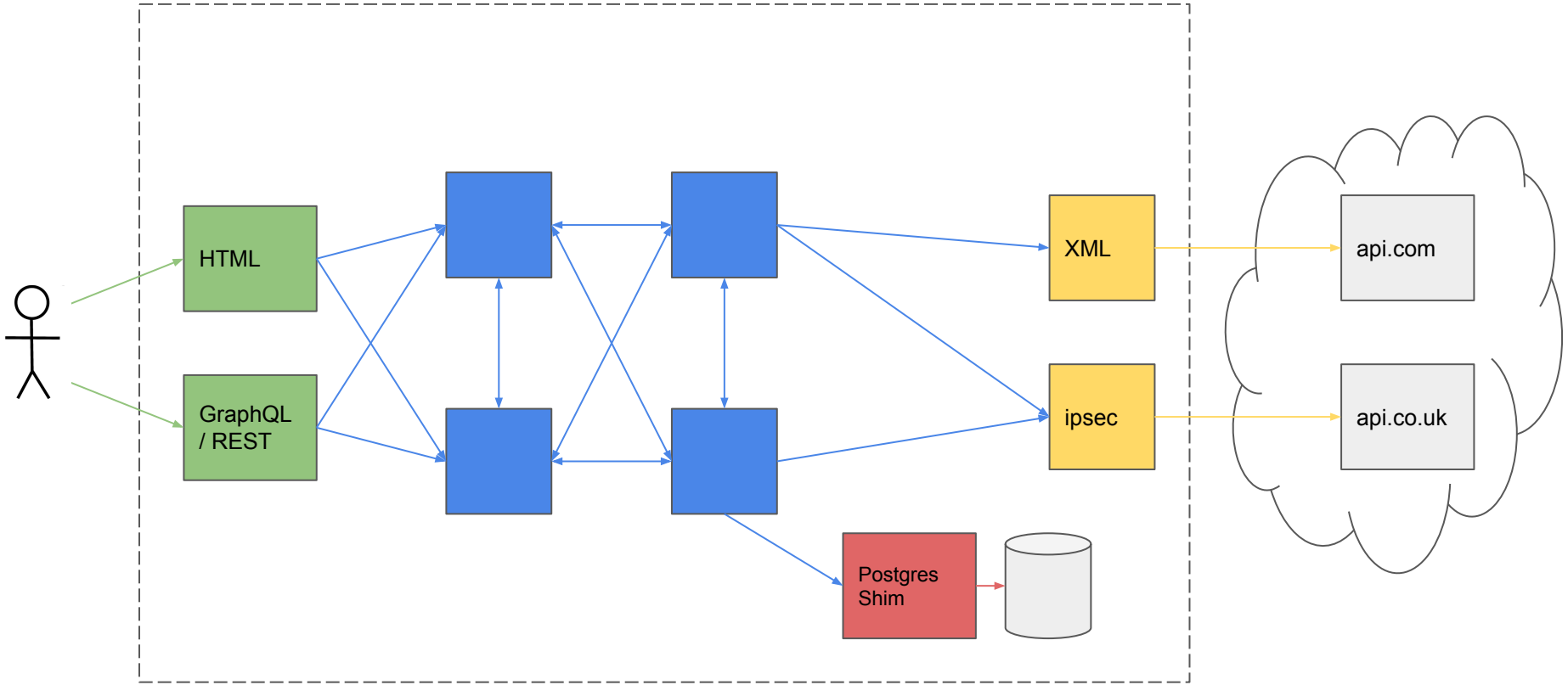
Service Chains



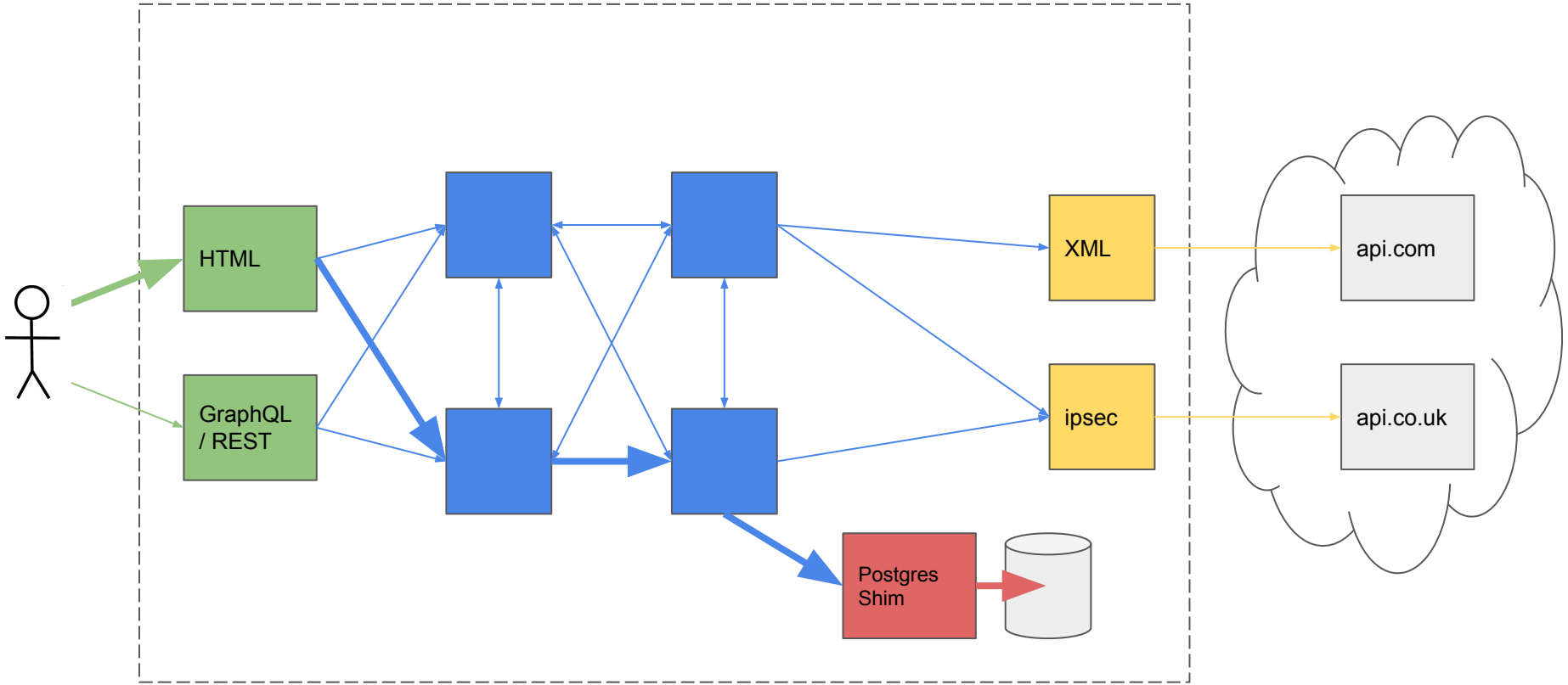
Service Chains



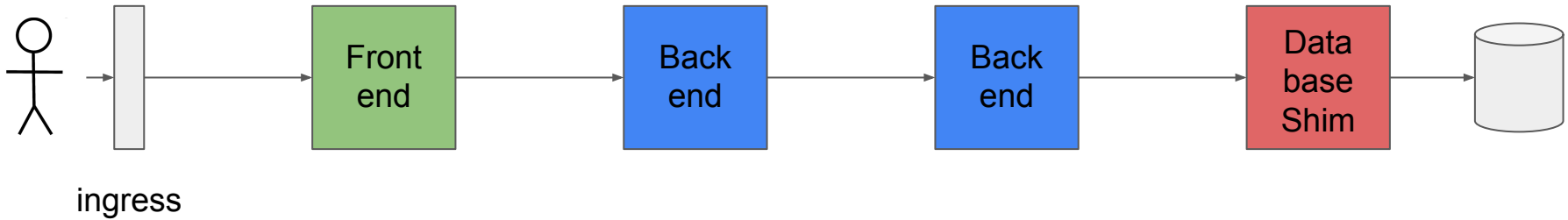
µService Taxonomy



µService Taxonomy

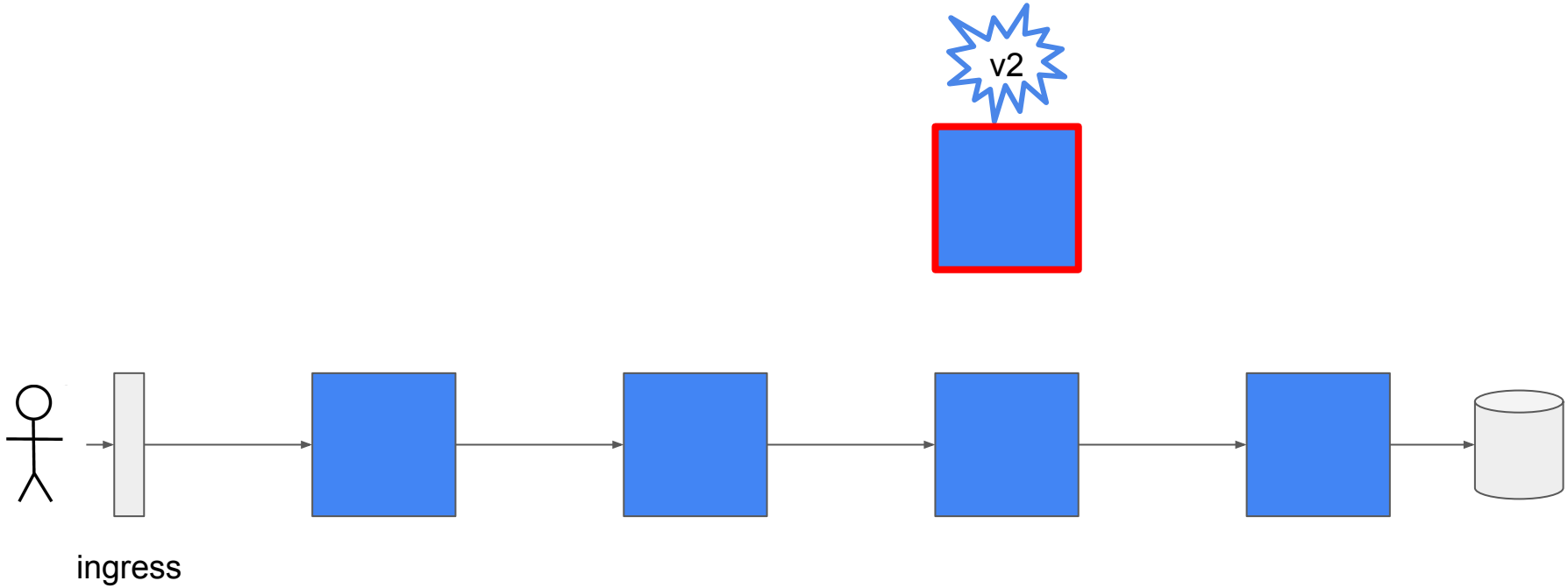


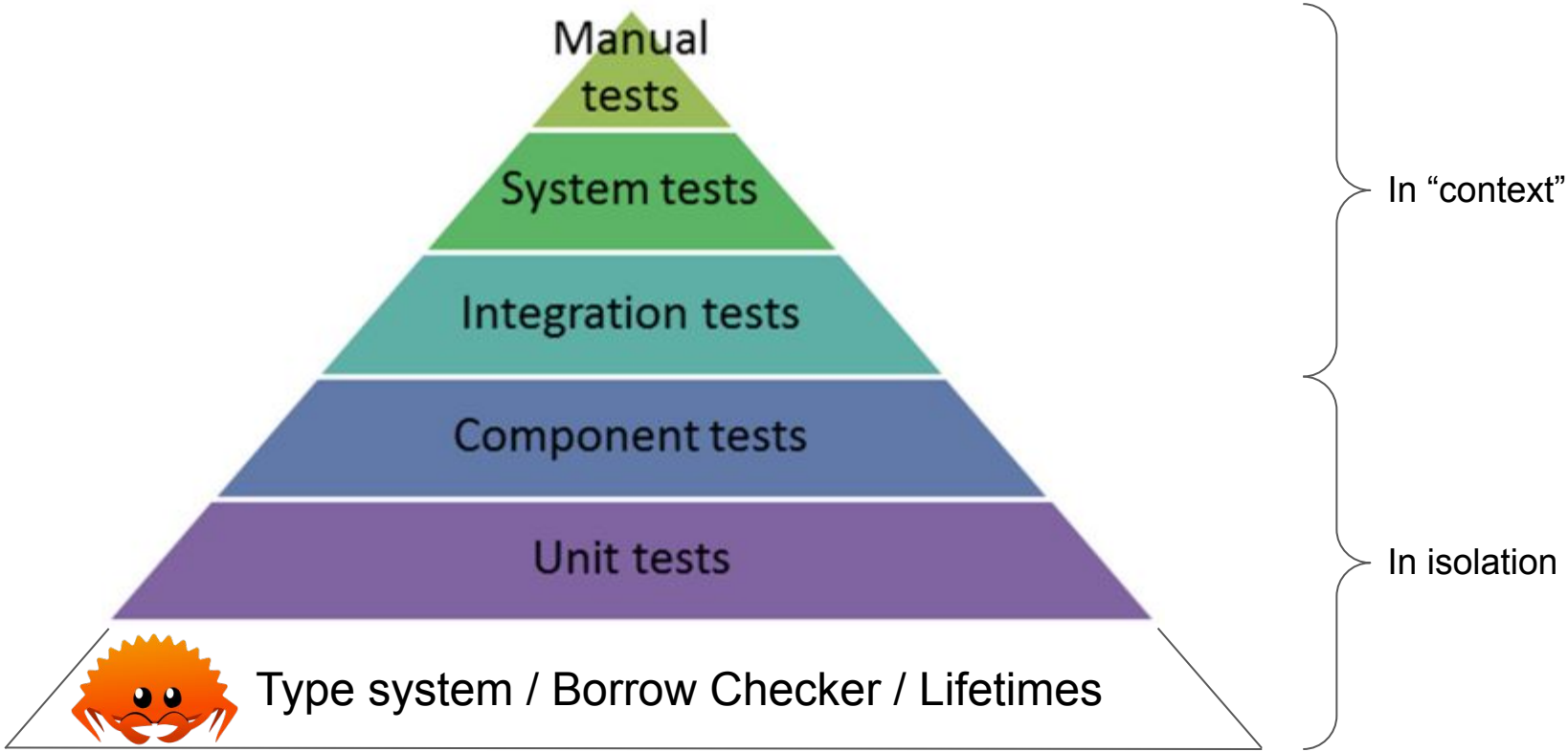
µService Taxonomy



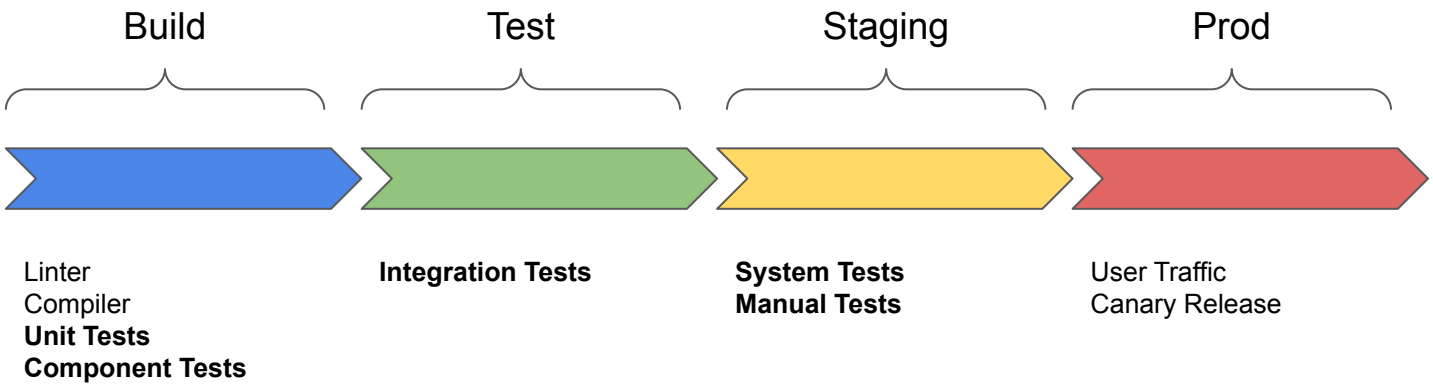
Continuous Deployment / Release

Testing new versions

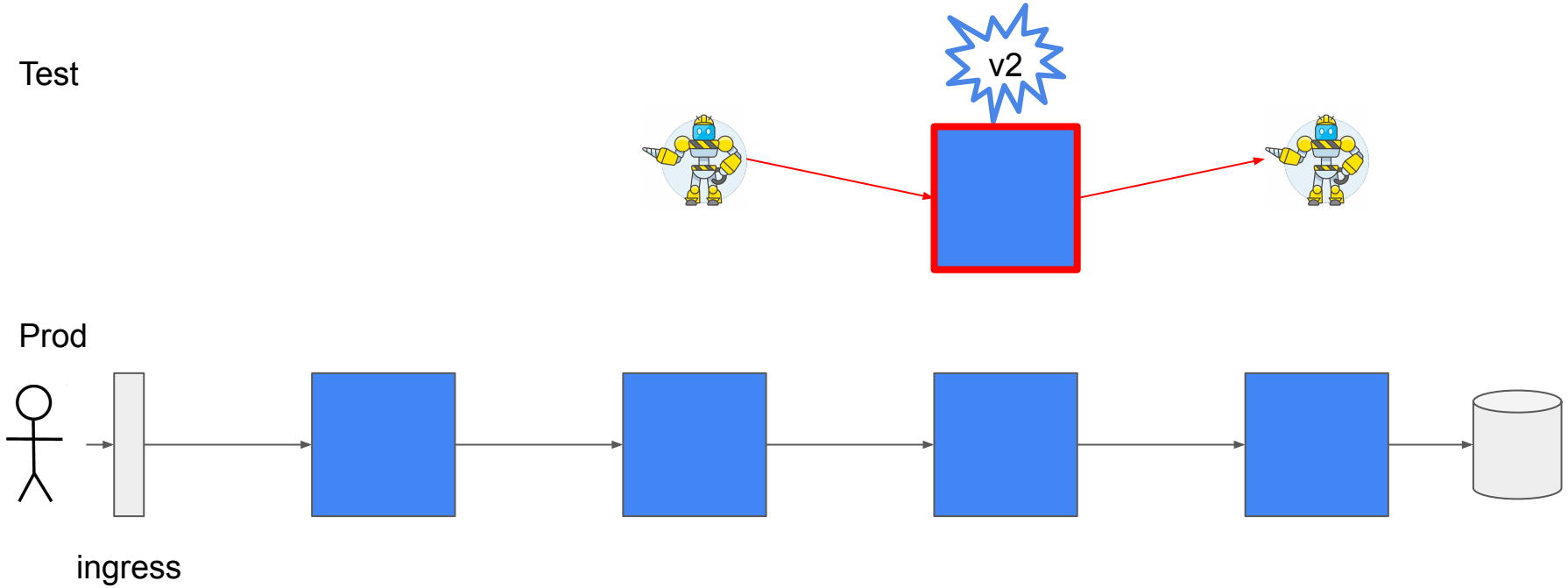



Manual tests
System tests
Integration tests
Component tests
Unit tests
Type system / Borrow Checker / Lifetimes
In “context”
In isolation

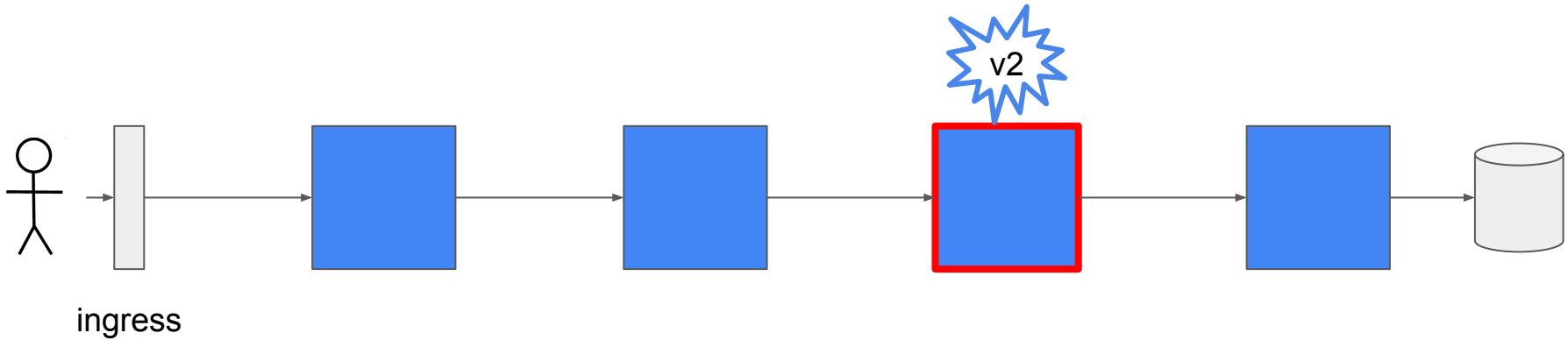
Deployment and release stages



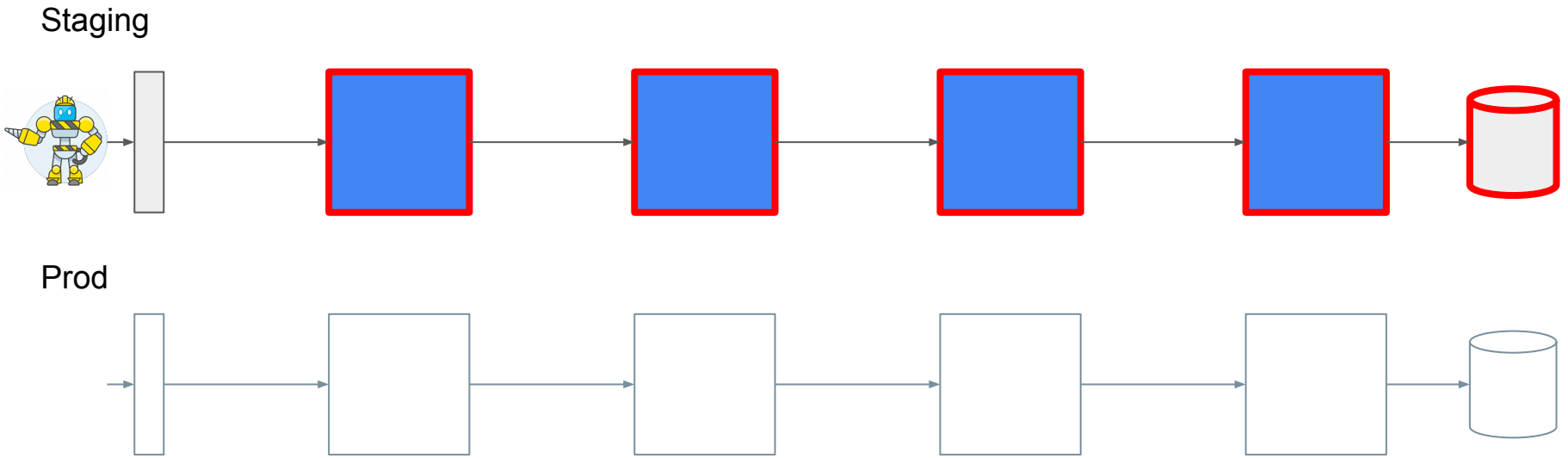
Integration tests



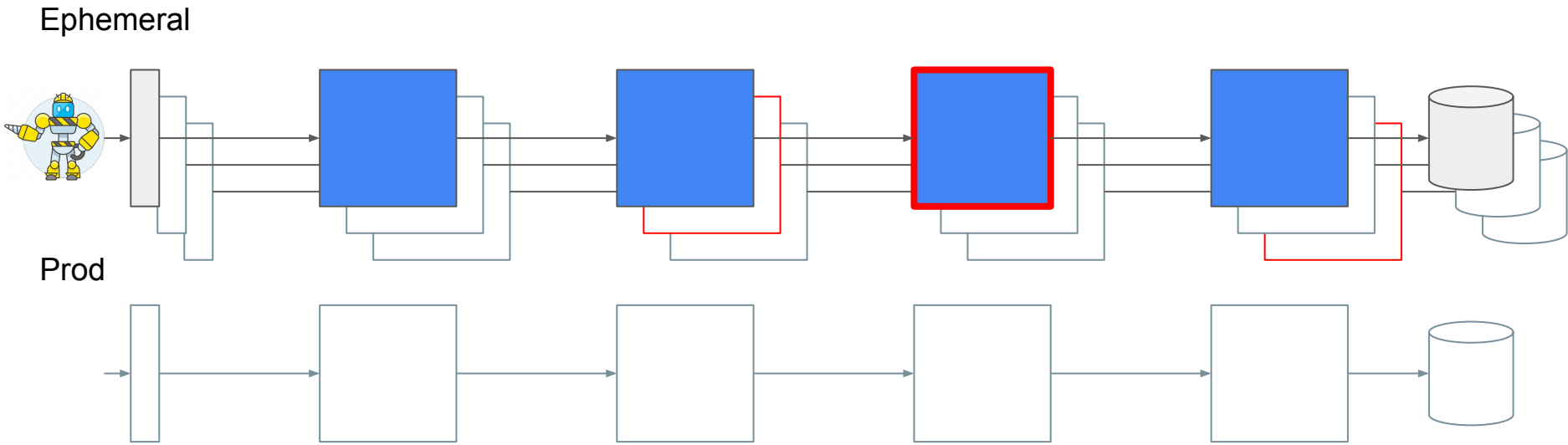
Testing in context



Staging environment

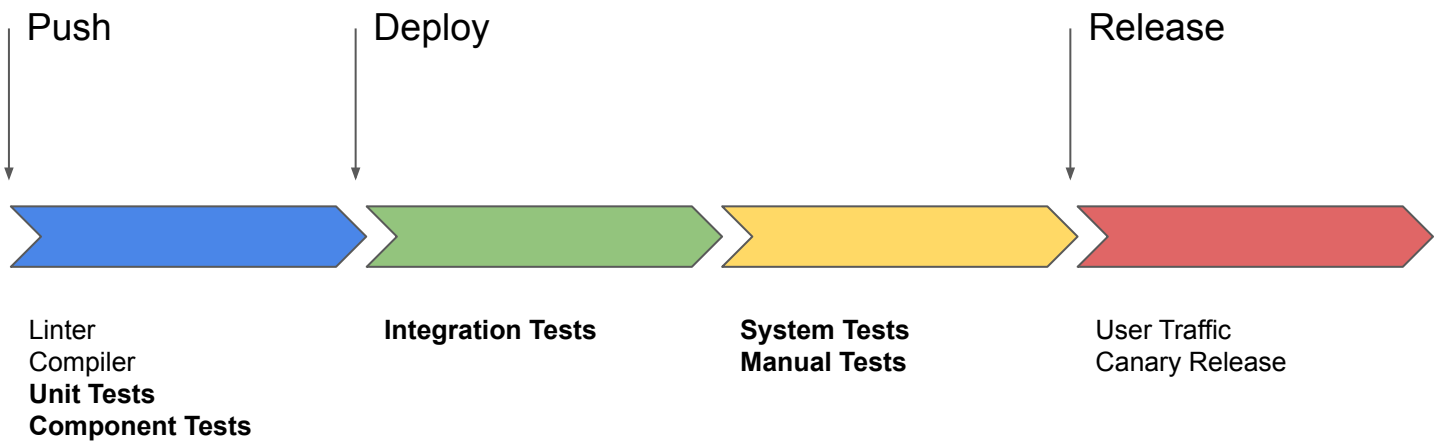


Ephemeral environments

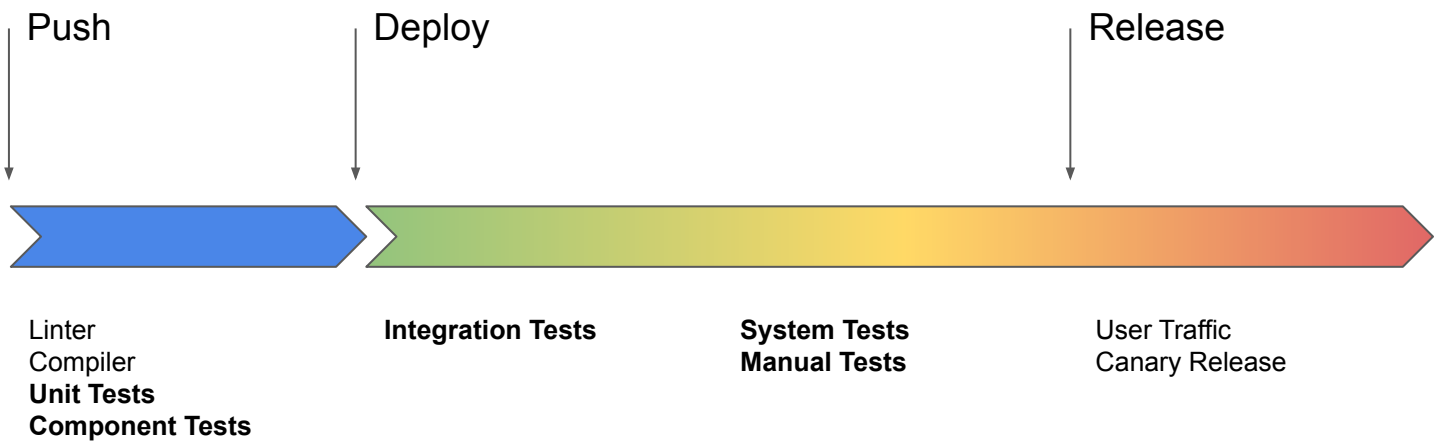


Testing in Production

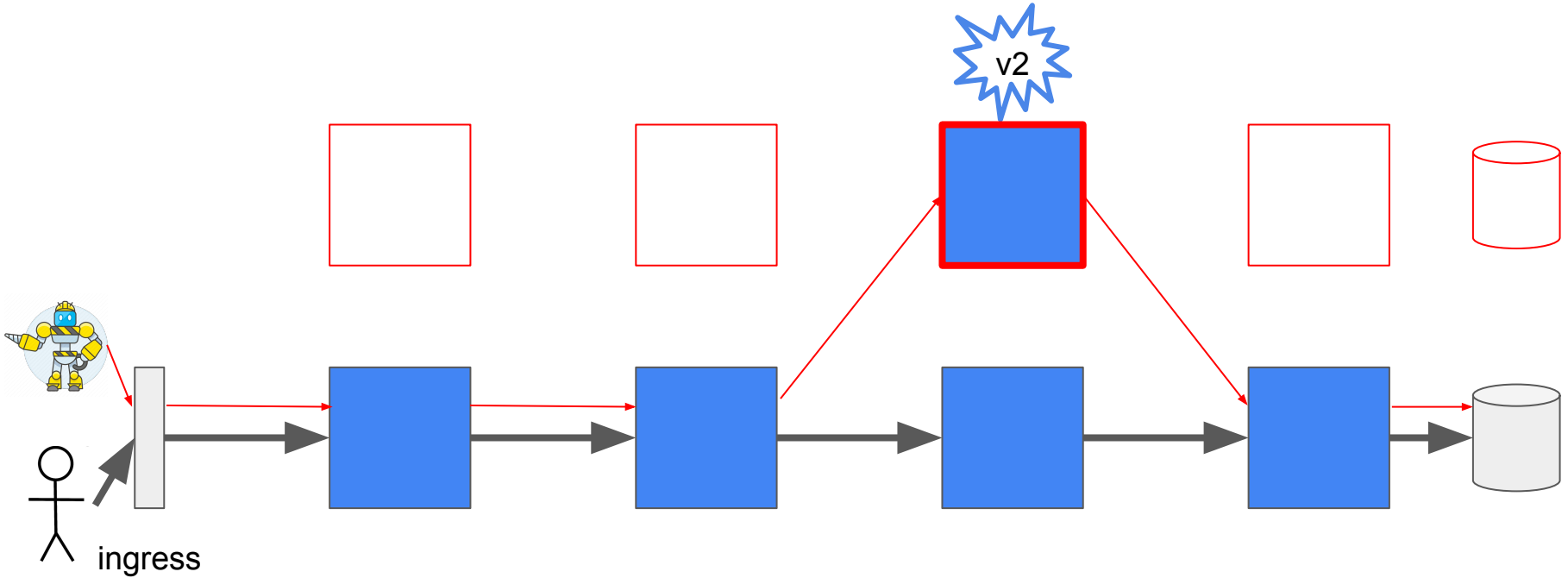
Do we really need *Test*?



Do we really need *Test*?

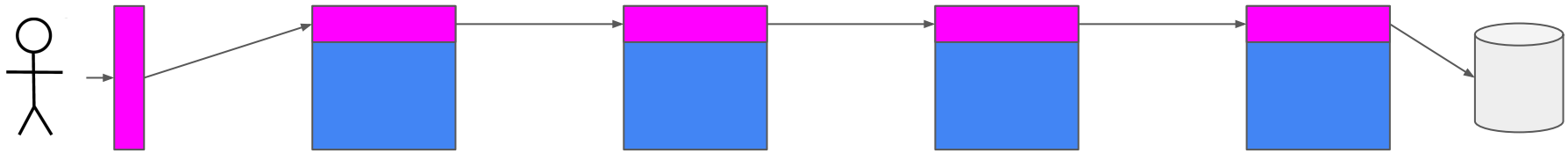


End-to-end tests

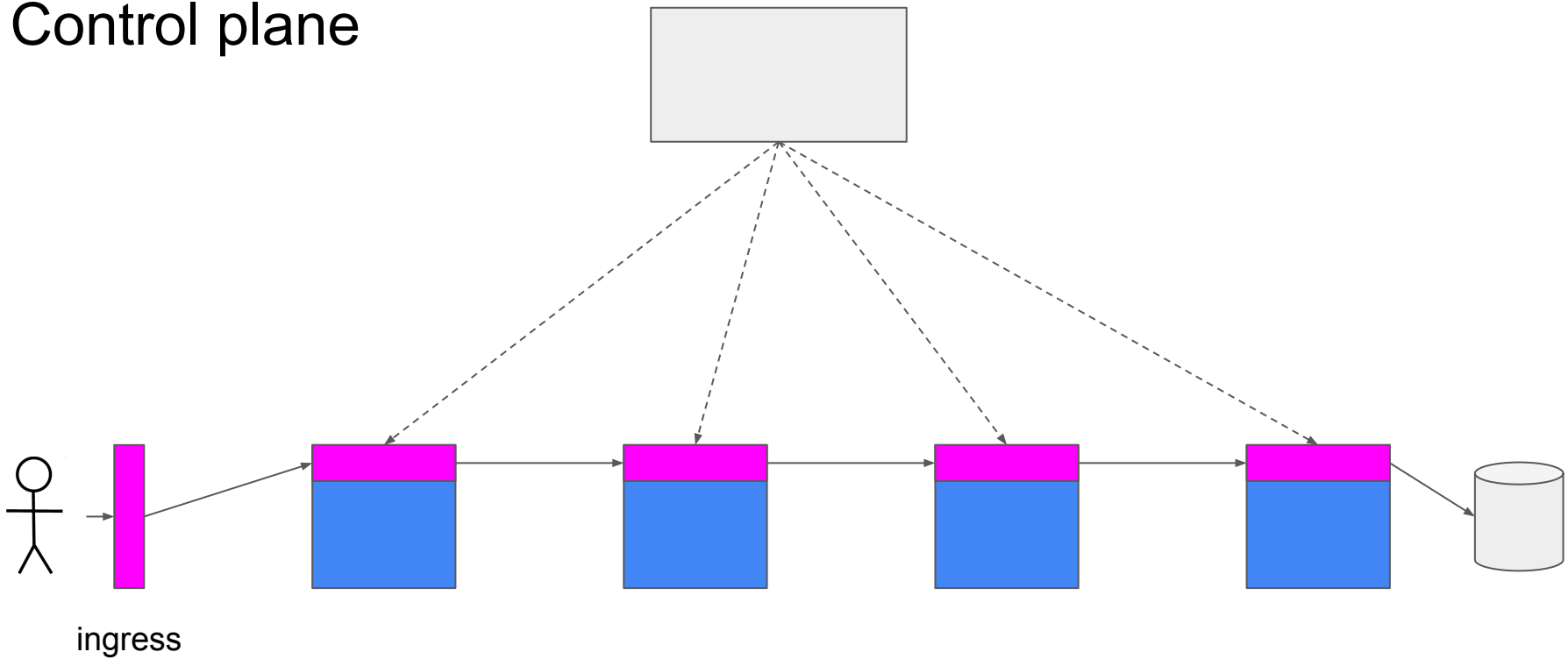


Implementation

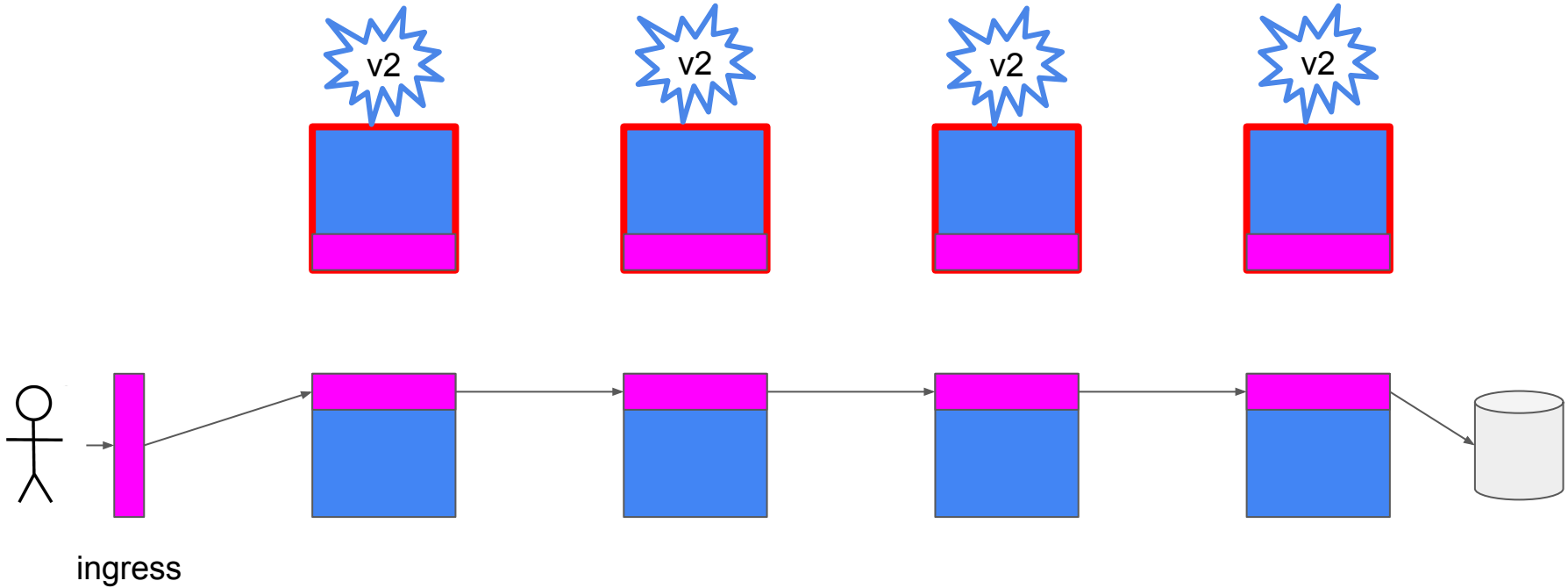
Sidecar proxies



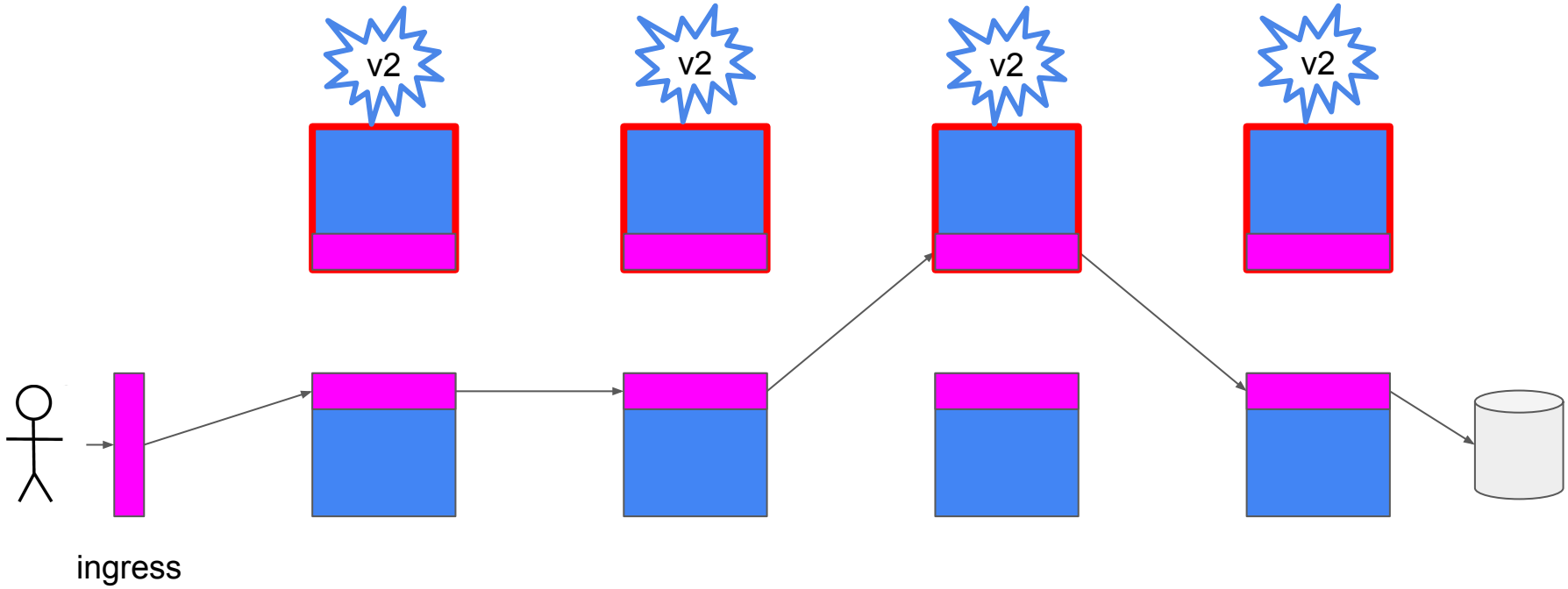
Control plane



Service mesh



Service mesh



Prior Art

“Breaking up Lyft’s Development Monolith”

- Jake Kaufman
- QCon London 2022

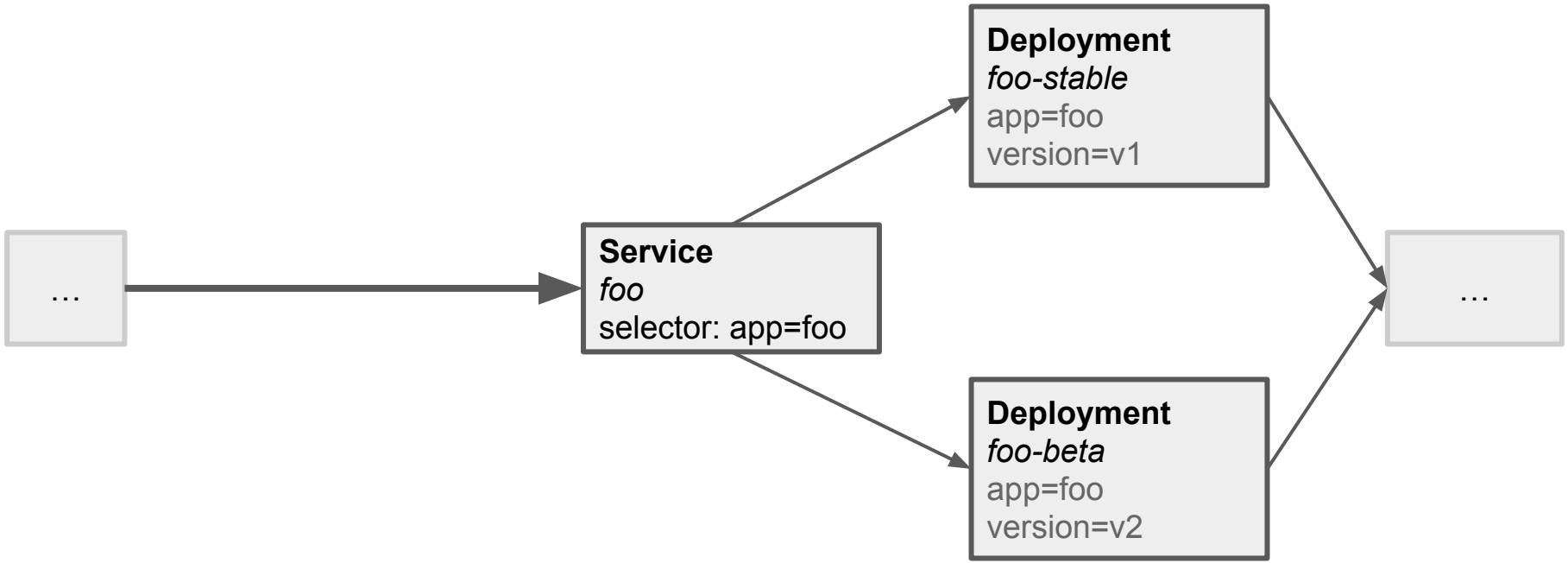
- Lyft’s proprietary control-plane
- Custom C++ code

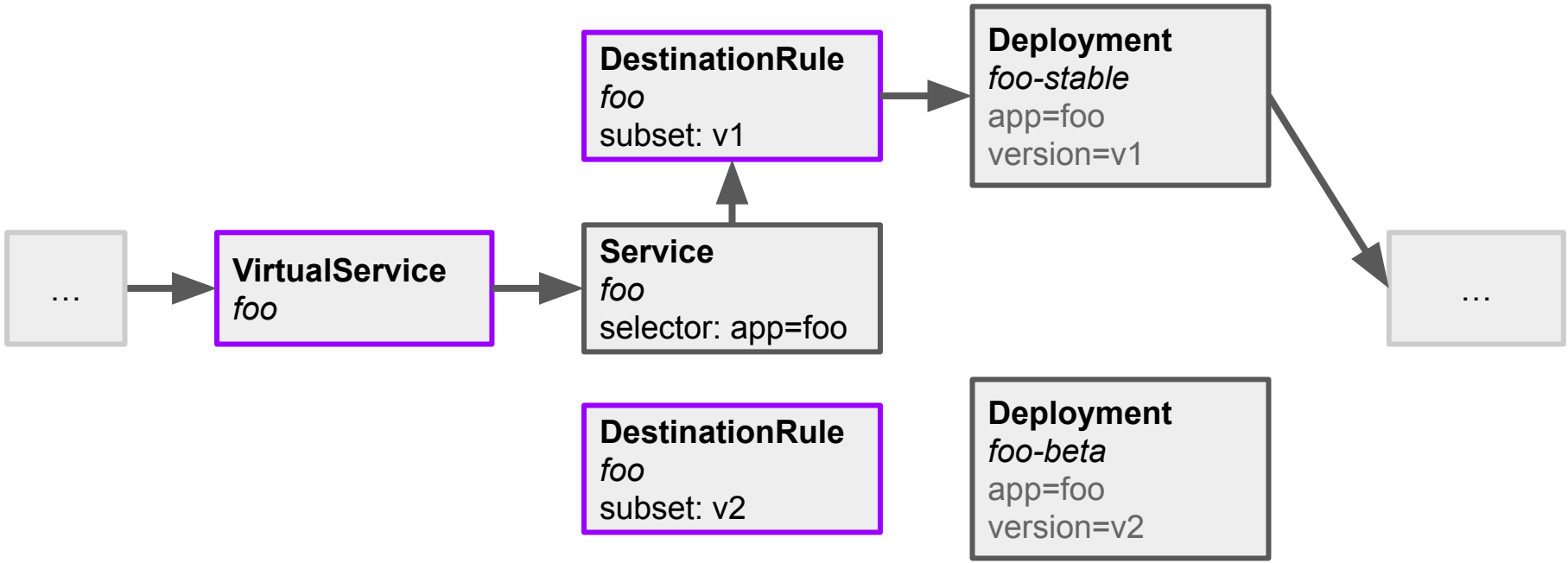
Istio

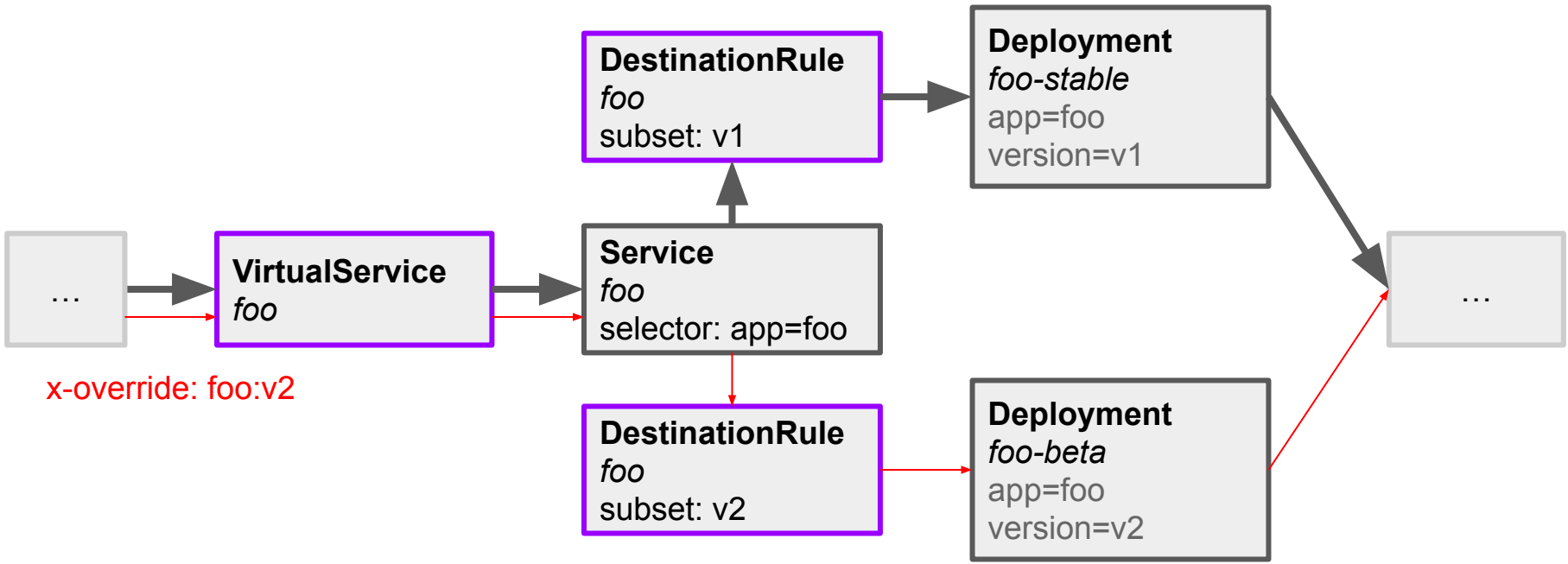
Istio Config Recap

- `VirtualService` - “Given a request for a named service, where do I route it?”
  - Select workloads based on headers
  - Select a subset of available workloads, based on a attributes like labels
- `DestinationRule` - “How do I talk to the workloads that have been selected?”
  - Load balancing strategy across them
  - TLS stance
  - Subsets to group them into

…
Service
foo
selector: app=foo
Deployment
foo-stable
app=foo
version=v1
…


…
Service
foo
selector: app=foo
Deployment
foo-stable
app=foo
version=v1
Deployment
foo-beta
app=foo
version=v2
…


…
VirtualService
foo
Service
foo
selector: app=foo
DestinationRule
foo
subset: v1
DestinationRule
foo
subset: v2
Deployment
foo-stable
app=foo
version=v1
Deployment
foo-beta
app=foo
version=v2
…

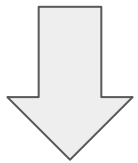

…
VirtualService
foo
x-override: foo:v2
DestinationRule
foo
subset: v1
Service
foo
selector: app=foo
DestinationRule
foo
subset: v2
Deployment
foo-stable
app=foo
version=v1
Deployment
foo-beta
app=foo
version=v2
…

DestinationRule

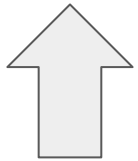
```yaml
apiVersion: networking.istio.io/v1beta1
kind: DestinationRule
metadata:
  name: foo
spec:
  host: foo
  subsets:
    - name: v1
      labels: { version: v1 }
    - name: v2
      labels: { version: v2 }
```

VirtualService

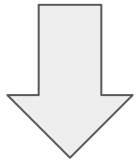
```
apiVersion: networking.istio.io/v1beta1
kind: VirtualService
metadata:
  name: foo-overrides
spec:
  hosts: [foo]
http:
```



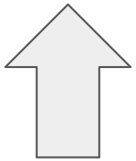
VirtualService



```
- match:
    - headers:
        x-override: “foo:v1”
  route:
    - destination: { host: foo, subset: v1 }
- match:
    - headers:
        x-override: “foo:v2”
  route:
    - destination: { host: foo, subset: v2 }
```

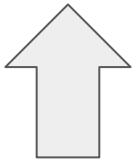


VirtualService

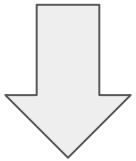


```
- route:
    - destination: { host: foo, subset: v1 }
```

VirtualService



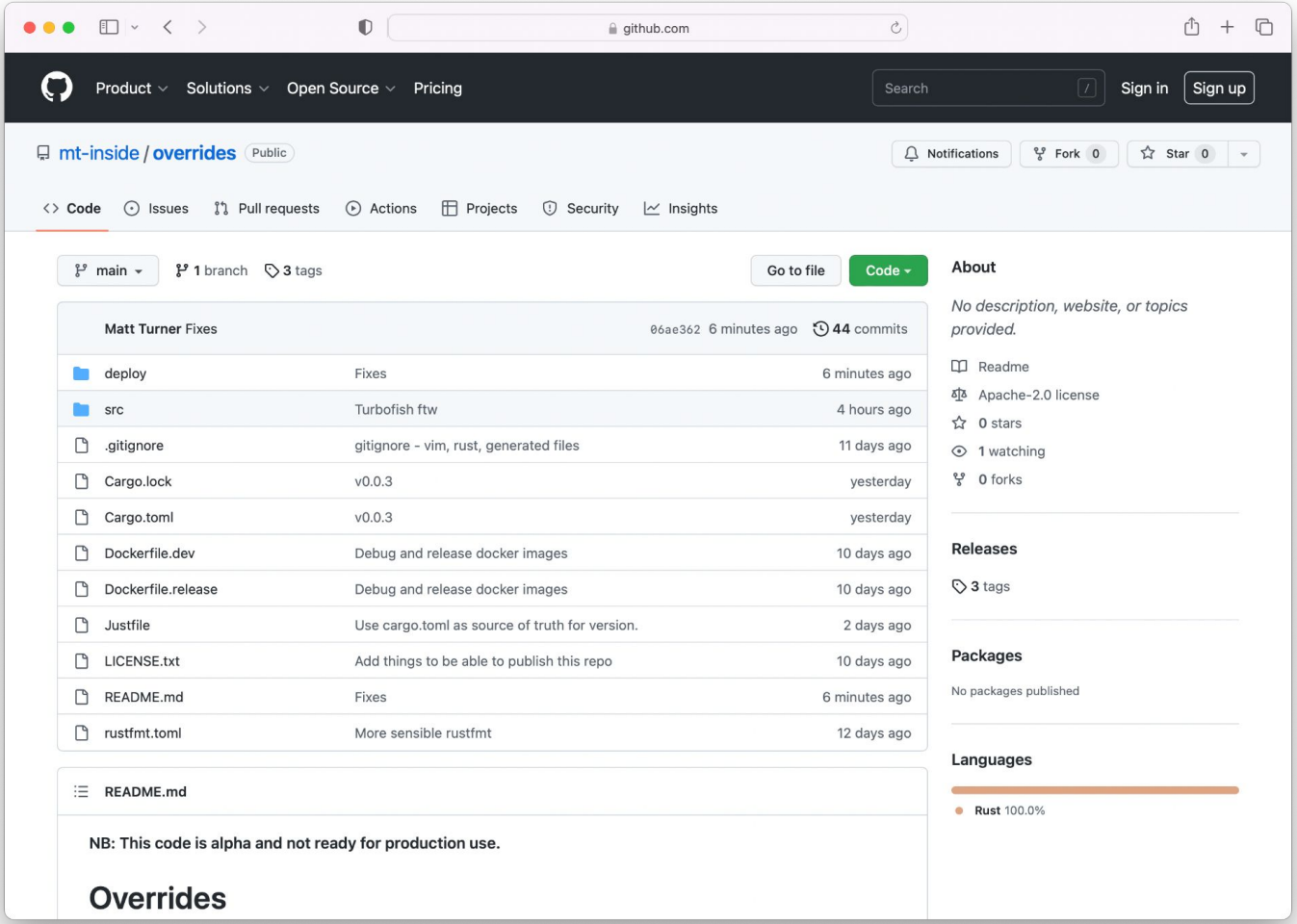
```
- match:
    - headers:
        x-override: { regex: “(.*,|^)foo:v1(,.*|$)” }
  route:
    - destination: { host: foo, subset: v1 }
- match:
    - headers:
        x-override: { regex: “(.*,|^)foo:v2(,.*|$)” }
  route:
    - destination: { host: foo, subset: v2 }
```



Repeated header key

- Curl puts them on the wire as multiple keys
  - This used to be contrary to the spec [RFC 2616 “HTTP/1.1” §4.2]
  - But is now ok [RFC 7230 “HTTP/1.1: Message Syntax and Routing” §3.2.2]
- And golang forwards them as multiple keys
- Envoy presents you a single combined string: “foo,bar,baz”
  - => Have to match a substring
- Envoy gives you: exact, prefix, suffix, contains, safe_regex
- Istio exposes: exact, prefix, regex (compiles to safe_regex)
- Regex it is then
  - Google RE2 syntax
  - Whole-string match
- `(.*,|^)foo:v2(,.*|$)` is the tightest regex I can get

Override-[generator,operator]


github.com
Product
Solutions
Open Source
Pricing
Search
Sign in
Sign up
mt-inside / overrides
Public
Notifications
Fork 0
Star 0
Code
Issues
Pull requests
Actions
Projects
Security
Insights
main
1 branch
3 tags
Go to file
Code
Matt Turner Fixes
06ae362 6 minutes ago
44 commits
deploy Fixes 6 minutes ago
src Turbofish ftw 4 hours ago
.gitignore gitignore - vim, rust, generated files 11 days ago
Cargo.lock v0.0.3 yesterday
Cargo.toml v0.0.3 yesterday
Dockerfile.dev Debug and release docker images 10 days ago
Dockerfile.release Debug and release docker images 10 days ago
Justfile Use cargo.toml as source of truth for version. 2 days ago
LICENSE.txt Add things to be able to publish this repo 10 days ago
README.md Fixes 6 minutes ago
rustfmt.toml More sensible rustfmt 12 days ago
README.md
NB: This code is alpha and not ready for production use.
Overrides
About
No description, website, or topics provided.
Readme
Apache-2.0 license
0 stars
1 watching
0 forks
Releases
3 tags
Packages
No packages published
Languages
Rust 100.0%

Demo! 😬

Caveats

- All Services in the namespace will be processed
  - The `kubernetes` Service is ignored, but all others must meet the requirements
  - Every Pod selected by each Service must have a version label (this is used to build the DR's subsets)
  - At least one of those Pods must have version: v1 (a default route is added to the v1 subset for traffic with no x-override header)
- There are no other `VirtualServices` or `DestinationRules` for the `Service`'s hostname
- Services need to forward the `x-override` header.

Challenges and Future Work

Trusting the Header

- Definitely needs filtering out at the edge
- Could let a compromised workload move laterally
- Lyft’s version stuck it in the JWT
- Also means it doesn’t have to be forwarded

GitOps integration

- Naïvely looks like transient state
- But a well-solved problem
- CLI - for integration with deployment pipelines
  - Needs to read YAML

Interaction with existing Istio resources

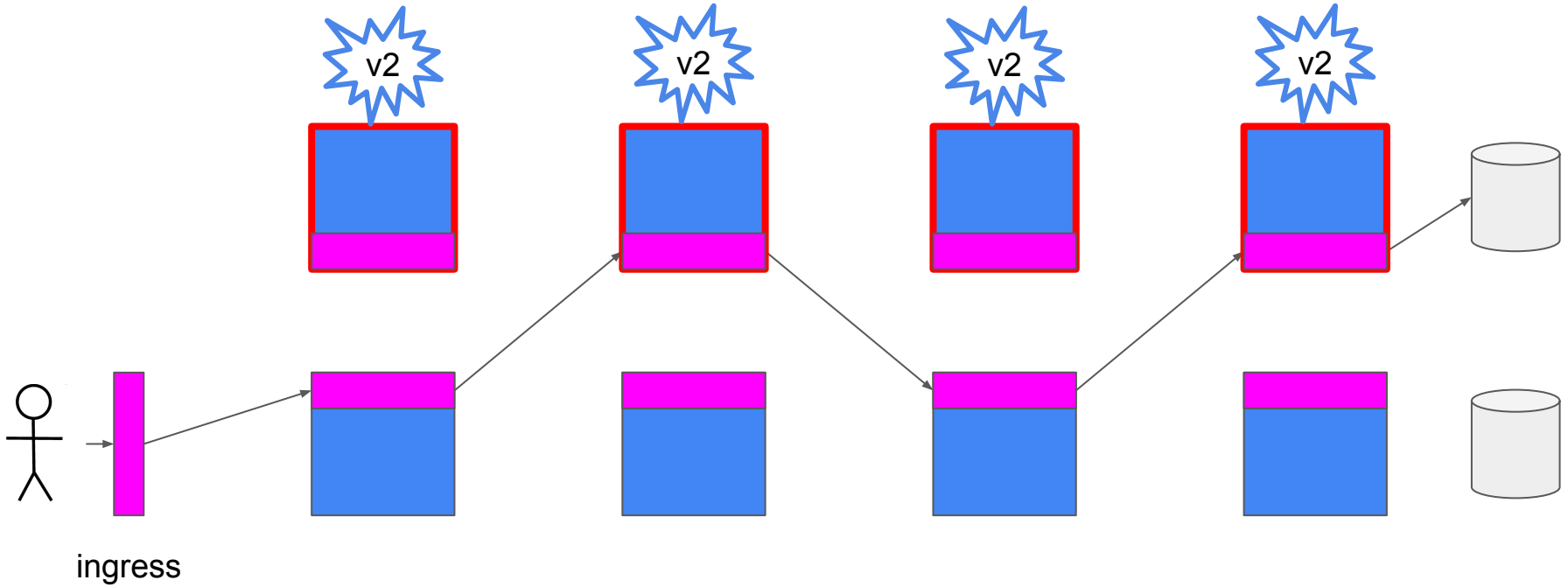
- Currently needs the namespace “to itself”
- Doesn’t work for the first hop

- Various options including delegated VirtualServices
- Operator uses server-side apply

“Sink to test”

- `db-shim` **`GET`** => v1
- `db-shim` **`POST`** => v2

V2 database shim



Conditional Routing

Pick `service-3:v2` *given* the caller is Foo, but not Bar

- Useful when the service chain isn't linear
- Or a service is part of more than one chain
- Coming soon with authz

Conclusions

	Basic	**Sophisticated**
Unit testing		
Testing (in prod)	Staging / ephemeral envs	Overrides
(Canary) Release	K8s rolling-update / DNS “load balancing”	Flagger

Thanks!

Questions	@mt165
Slides Videos Demo code	mt165.co.uk