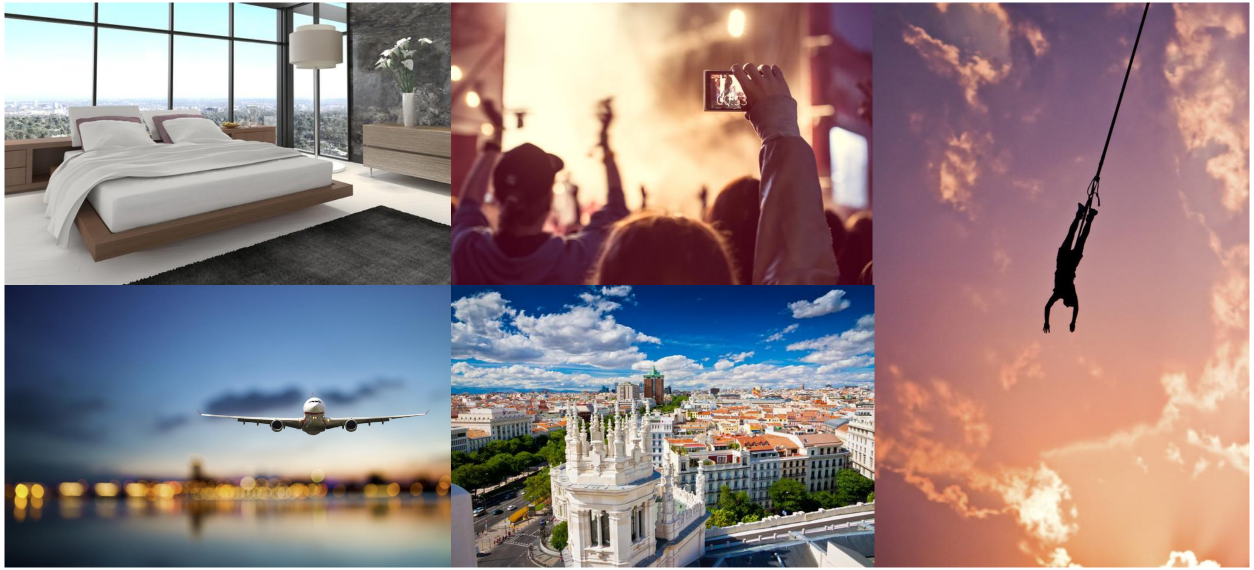
KubeCon - Berlin, 29-30 March 2017

Tales from Lastminute.com machine room: our journey towards a full on-premise kubernetes architecture in production

michele.orsi@lastminute.com
manuel.ranieri@lastminute.com

An inspiring travel company ..



Jetcost lastminute.com volagratis

A tech company to the core

Tech department: **300+ people**

Applications: **~100**

Database: **4 TB of data**

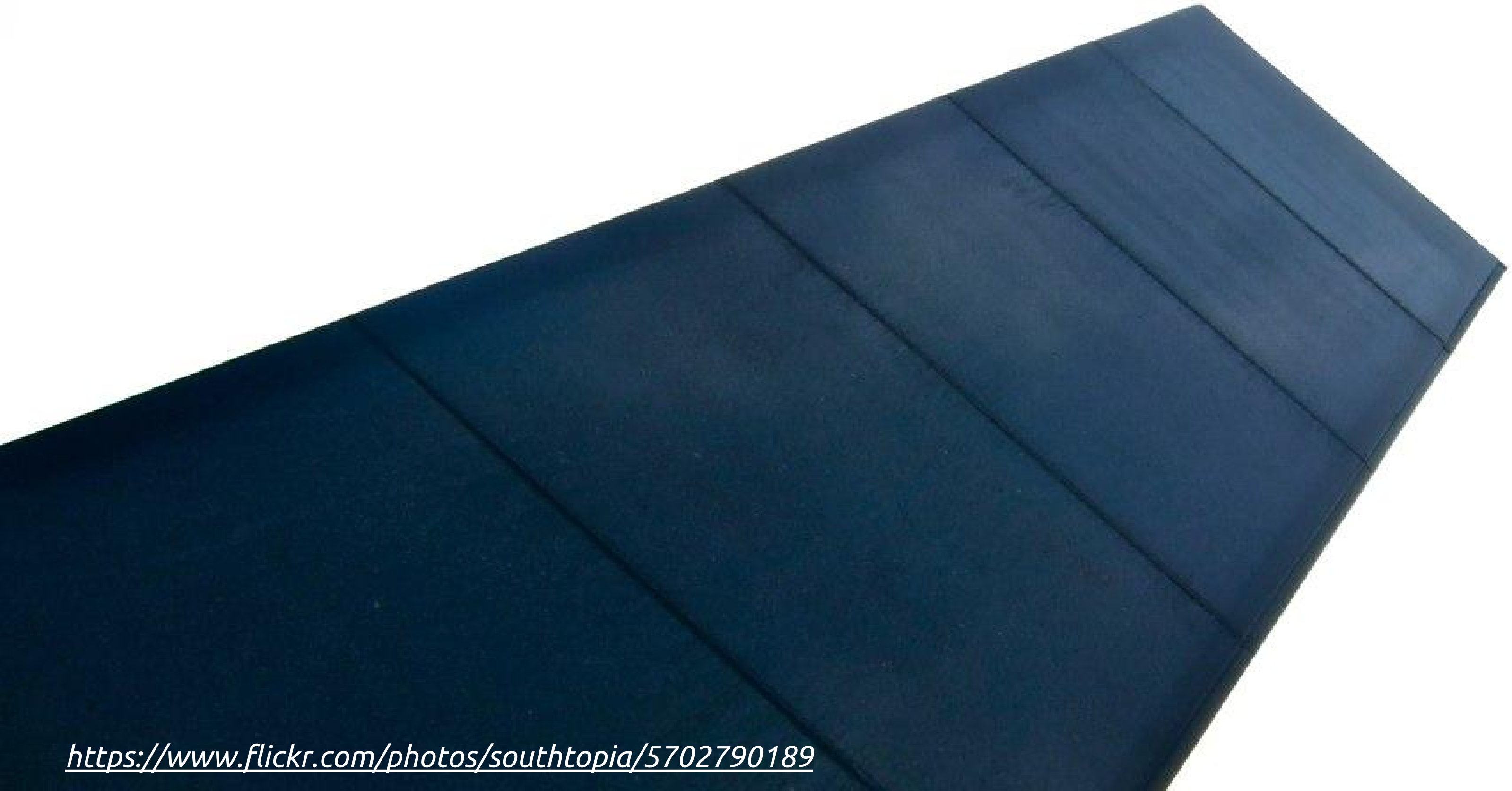
Servers: **1400 VMs, 300 physical machines**

Locations: **Chiasso, Milan, Madrid, London, Bengaluru**

Business: "technology is slow"

https://www.pexels.com/photo/turtle-walking-on-sand-132936/

Technology: "the monolith is the problem"



https://www.flickr.com/photos/southtopia/5702790189

"... let’s break into microservices!"

https://www.pexels.com/photo/gray-pebbles-with-green-grass-51168/

A lot of issues

- **LONG** provisioning time
- **LACK OF** alignment across environments
- **LACK OF** alignment across applications
- **LACK OF** awareness about *ops*

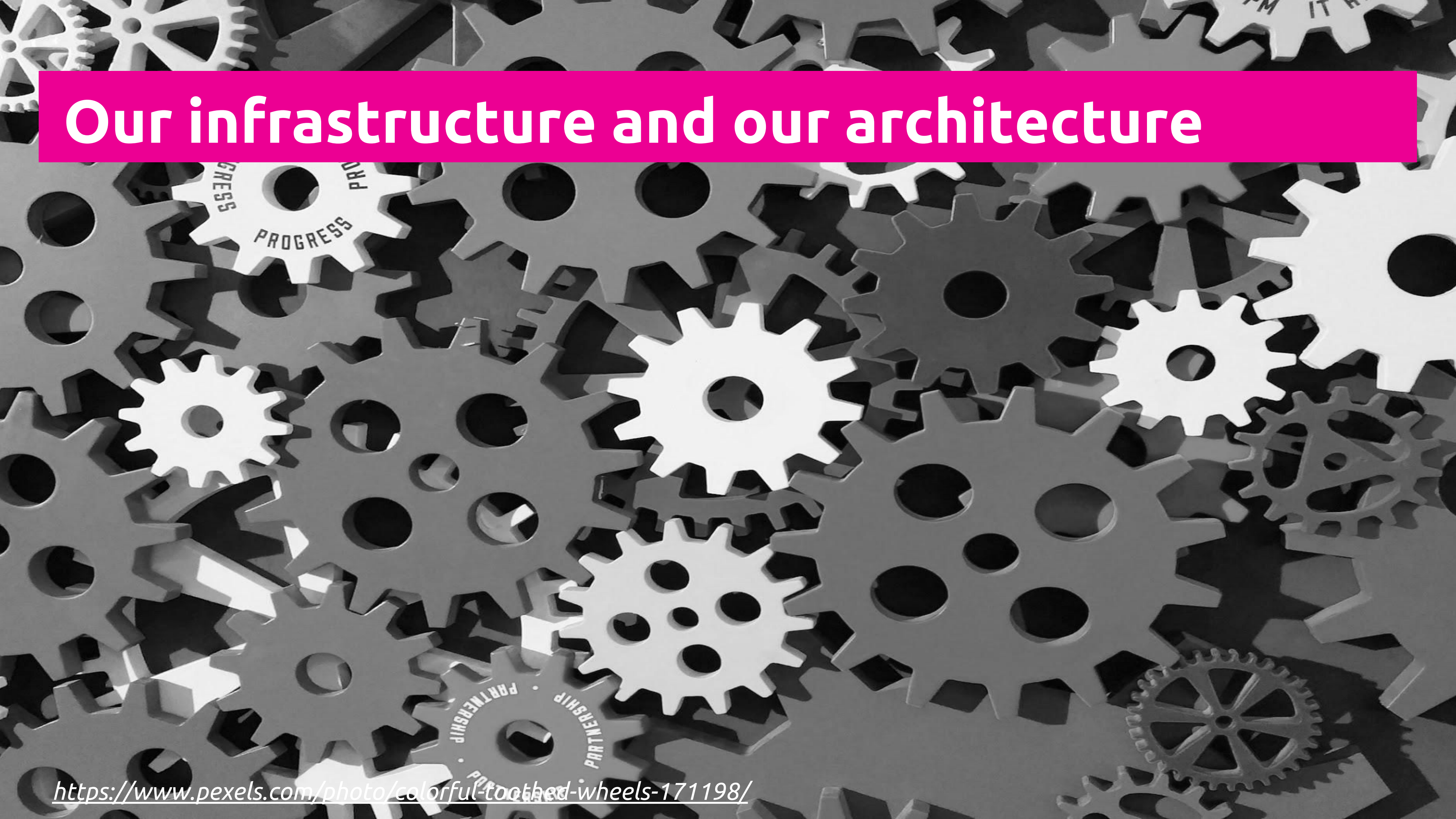
A year-long endeavour

- build a **new, modern infrastructure**
- migrate the **search (flight/hotel) product** there

... *without:*

- impacting the business
- throwing away our whole datacenter

Our infrastructure and our architecture



https://www.pexels.com/photo/colorful-toothed-wheels-171198/

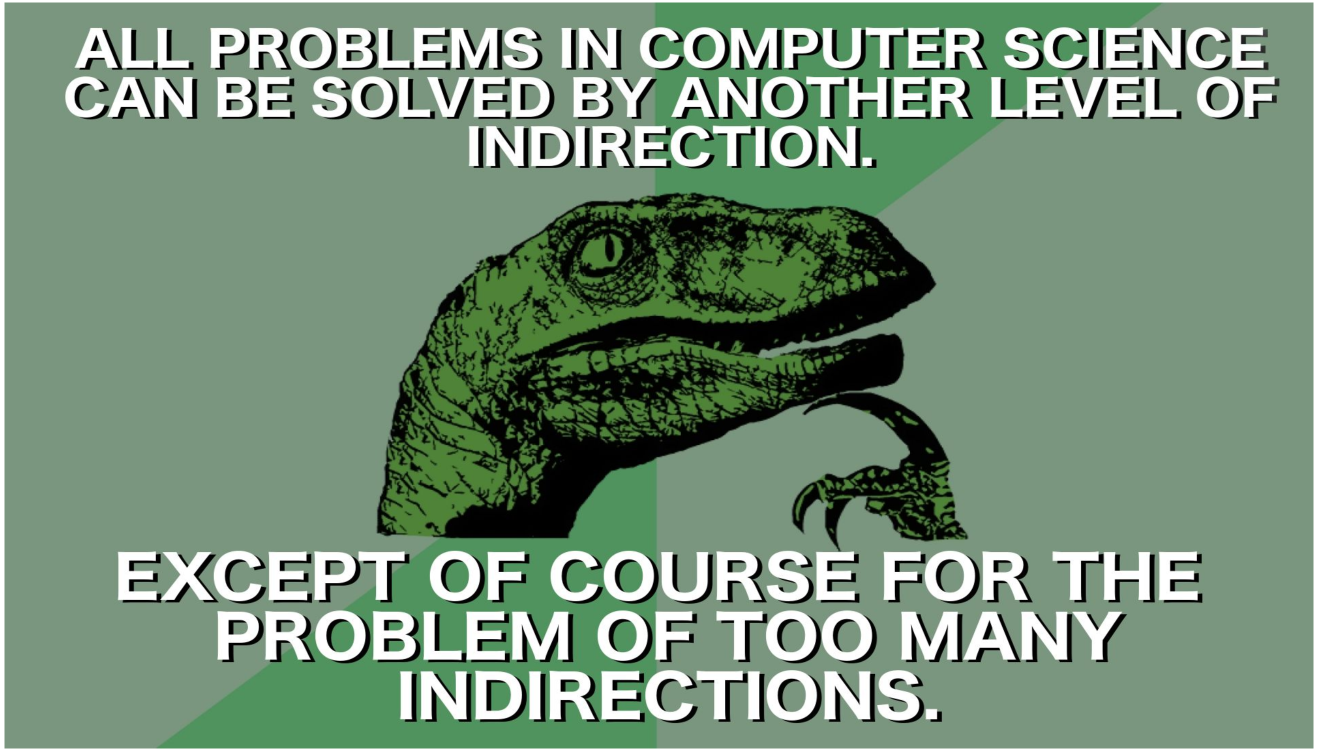
Virtualization platform

TONS

OF

VIRTUAL MACHINES

Virtualization platform

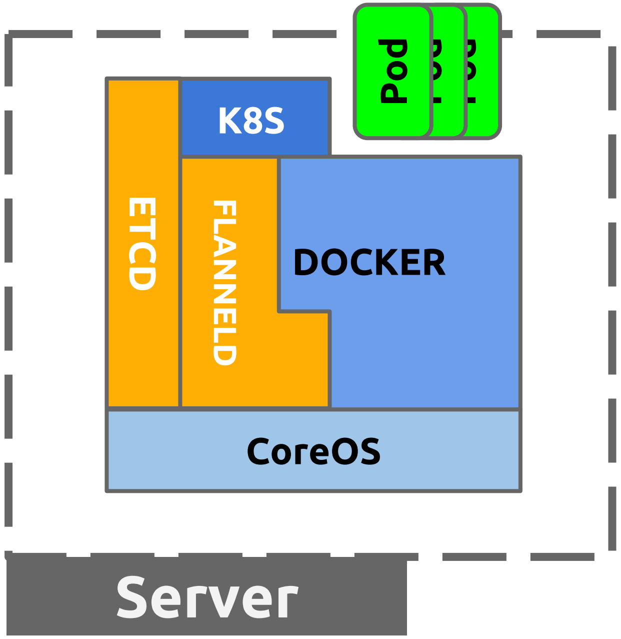

ALL PROBLEMS IN COMPUTER SCIENCE CAN BE SOLVED BY ANOTHER LEVEL OF INDIRECTION.
EXCEPT OF COURSE FOR THE PROBLEM OF TOO MANY INDIRECTIONS.

Engage

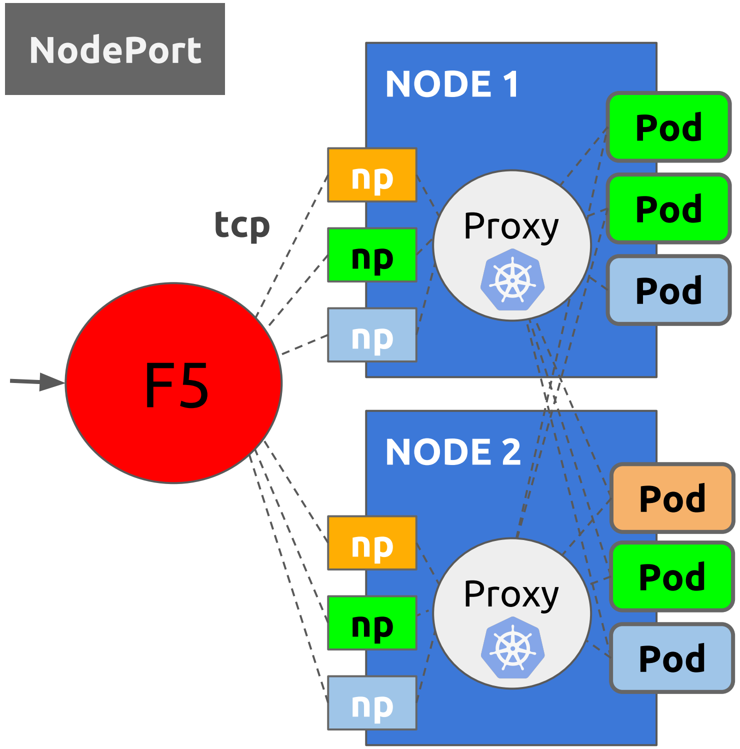
- CoreOS, the all-in-one choice
  - Cloudconfig configuration
  - Automatable in a shot
  - Really simple patch management

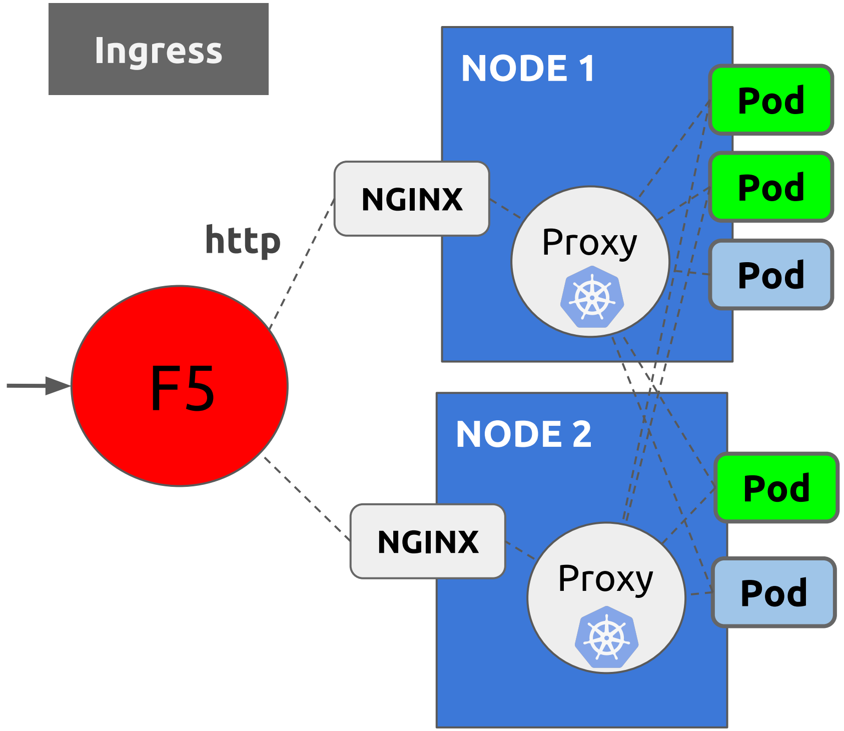
Our Kubernetes on CoreOS architecture is born

- The stack
  - ETCD
  - FLANNELD
  - DOCKER
- KUBERNETES (Google!)



How to talk with pods





In the name of service

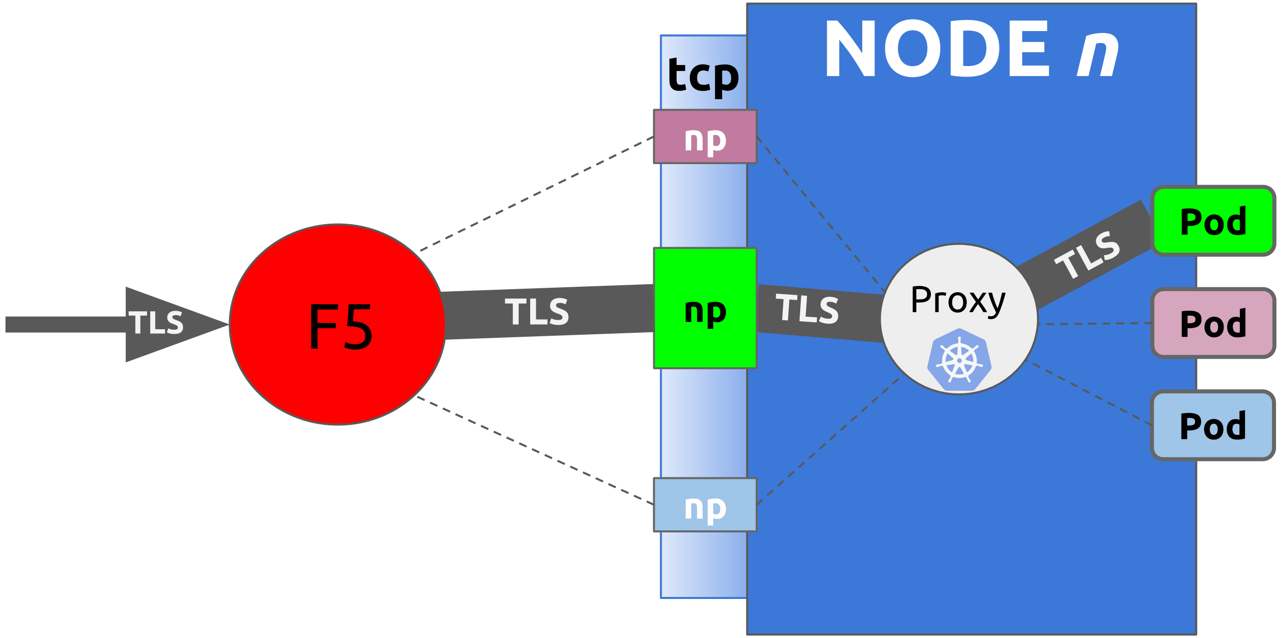
awesomeservice-ingress.yaml

```
- host: awesomeservice.prd.mykubecluster.intra
  http:
    paths:
    - path: /
      backend:
        serviceName: awesomeservice
        servicePort: 8081
```

In the name of service

```
*.[prd|qa|dev].mykubecluster.intra.  IN CNAME  kubef5ingress
```

The return of NodePort

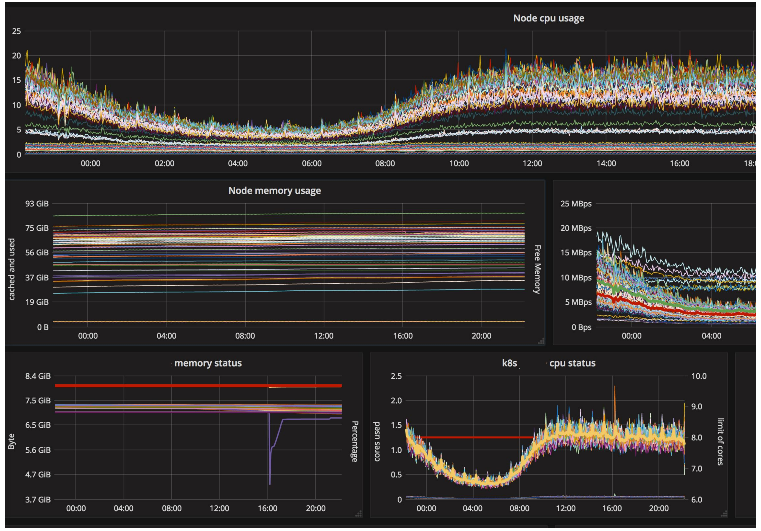


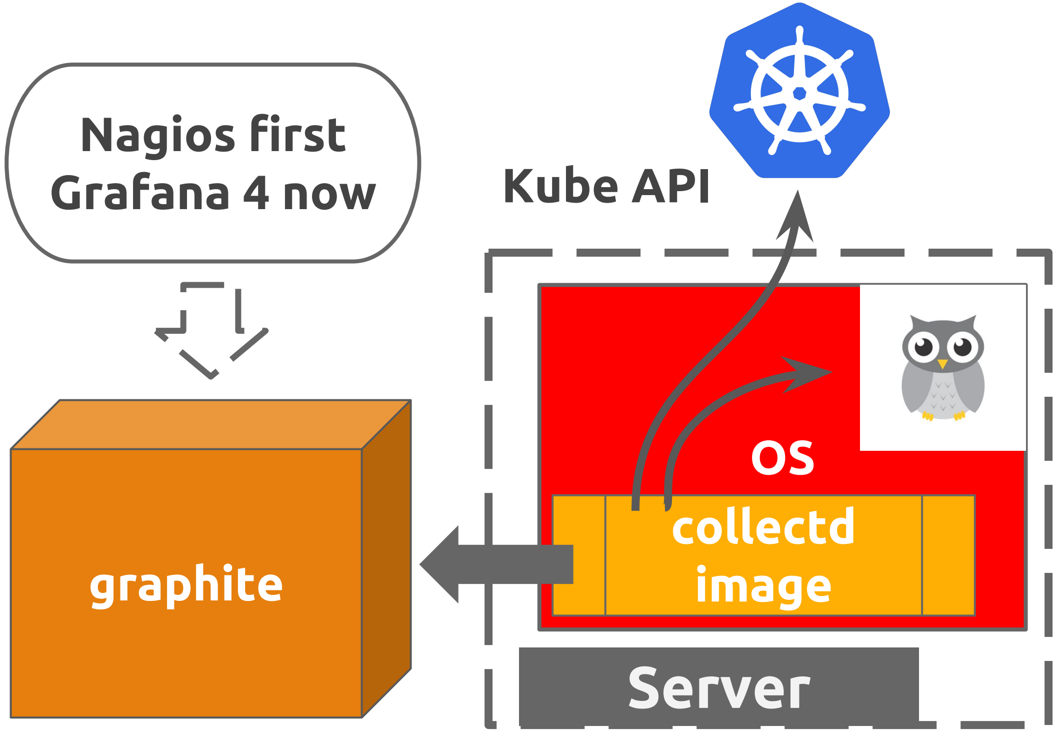
The registry brought another question…

?

Seriously?

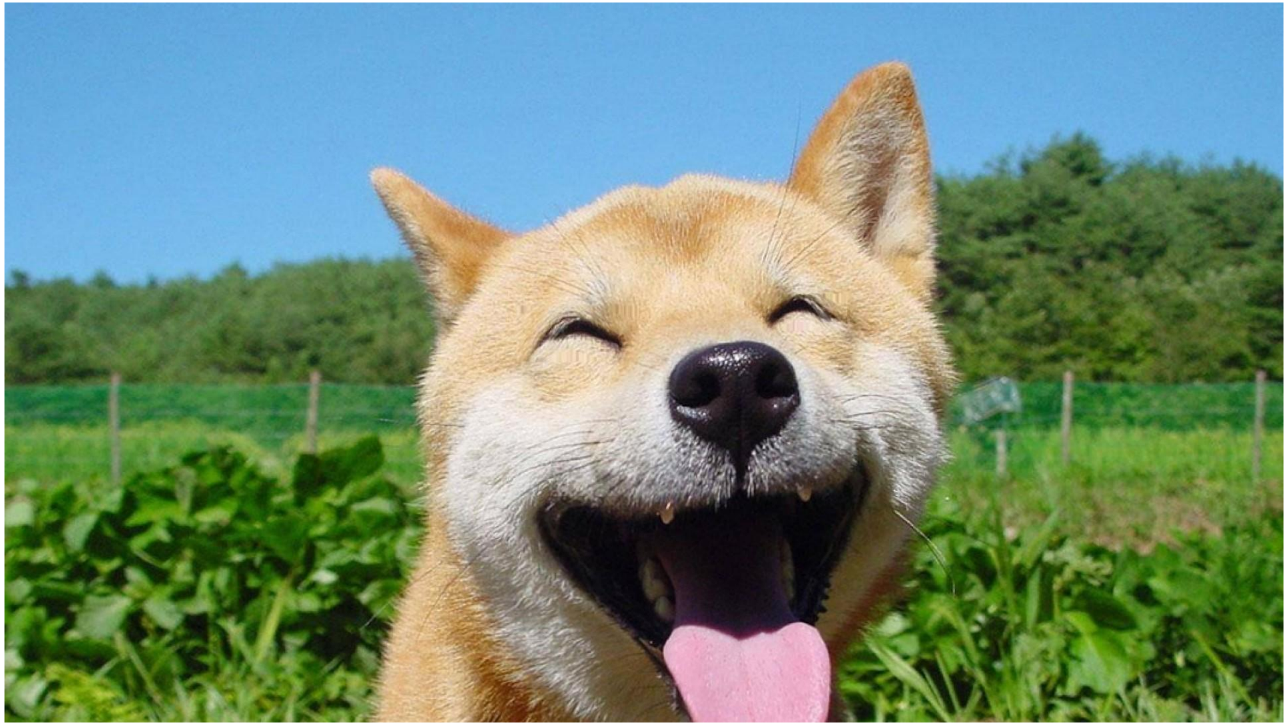
Rear window on kubernetes





icons from http://www.flaticon.com

We were happy!



lastminute.comgroup

Not happy anymore

Seriously?

The change… It’s a kind of magic

KEEP

CALM

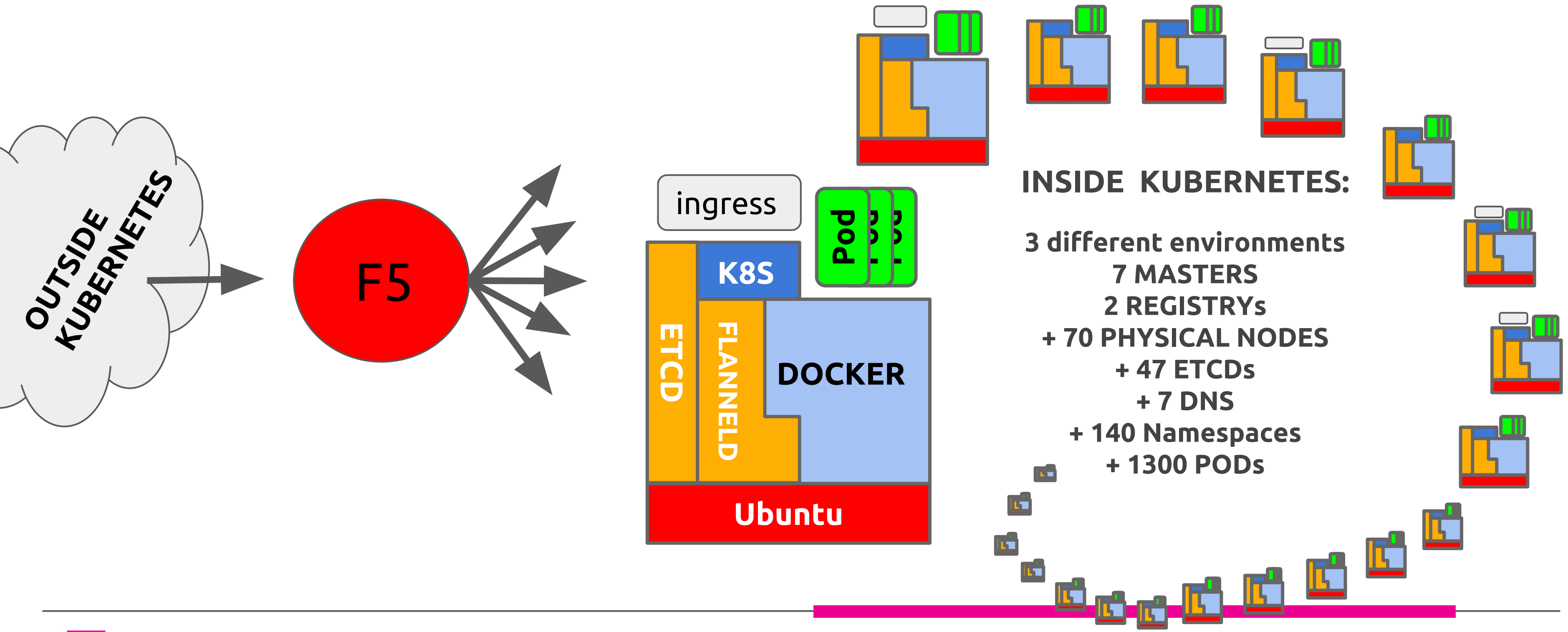
and

TRUST KUBERNETES

What we learned

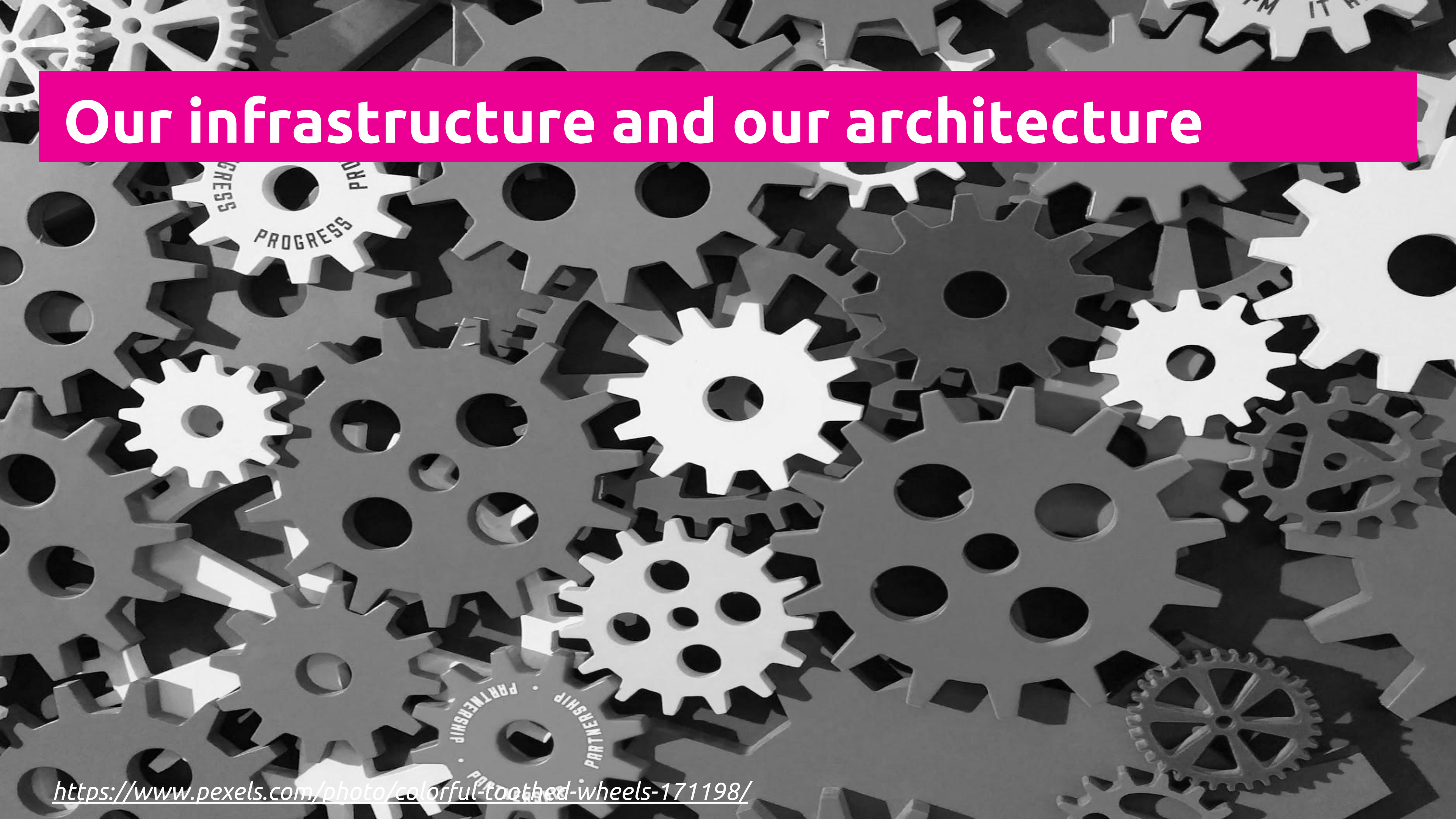
Lots of things!

The final architecture (so far...)



INSIDE KUBERNETES:

3 different environments
7 MASTERS
2 REGISTRYs
+ 70 PHYSICAL NODES
+ 47 ETCDs
+ 7 DNS
+ 140 Namespaces
+ 1300 PODs



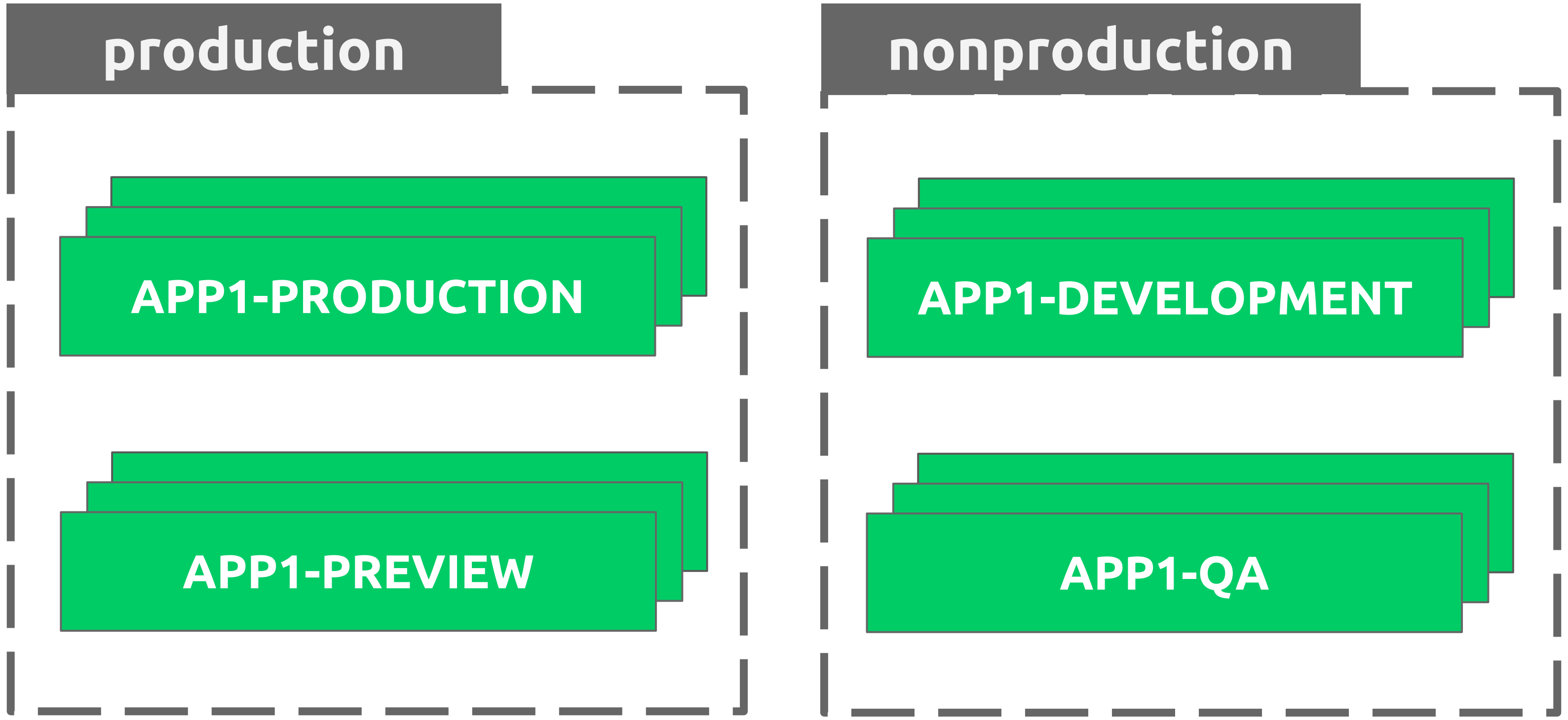
Our infrastructure and our architecture

https://www.pexels.com/photo/colorful-toothed-wheels-171198/

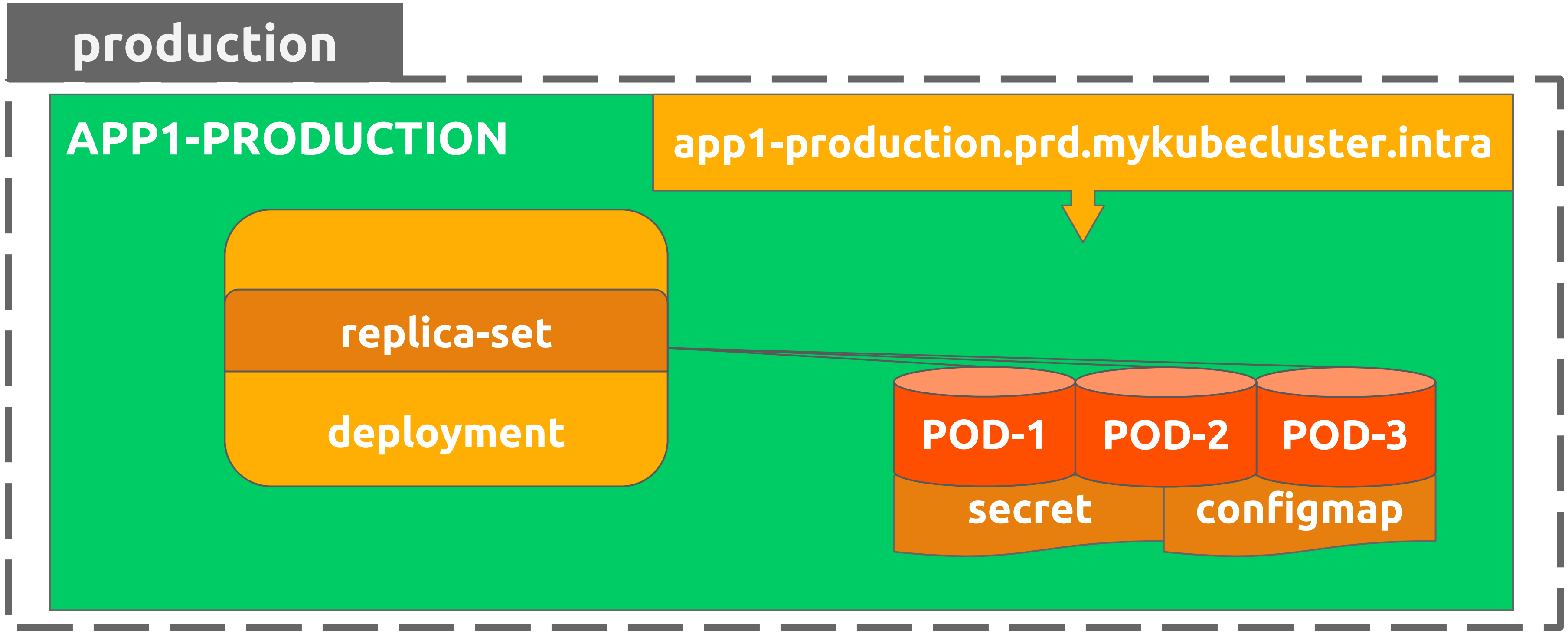
Our core axioms

- **same architecture** across environments
- a **common framework** to align software
- **centralized monitoring/logging**, with alerts
- **zero downtime** deployment
- **automation** everywhere

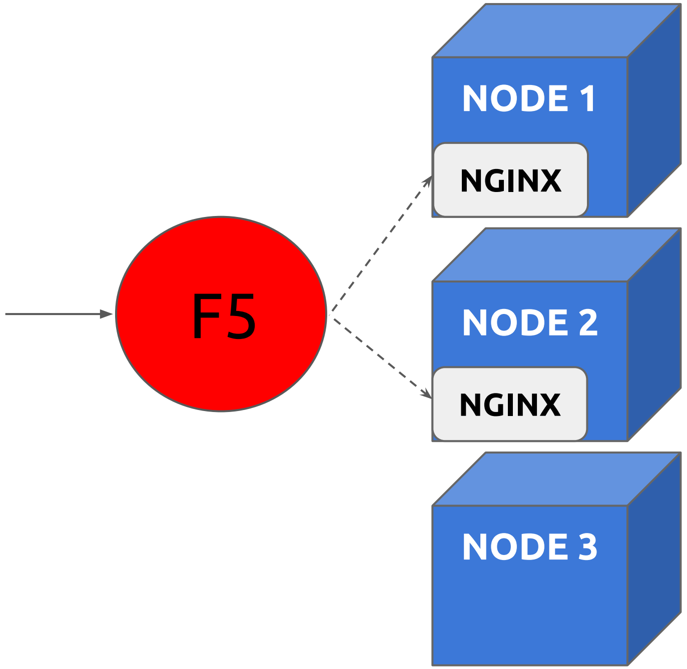
Kubernetes: our architecture



Kubernetes: our architecture and choices



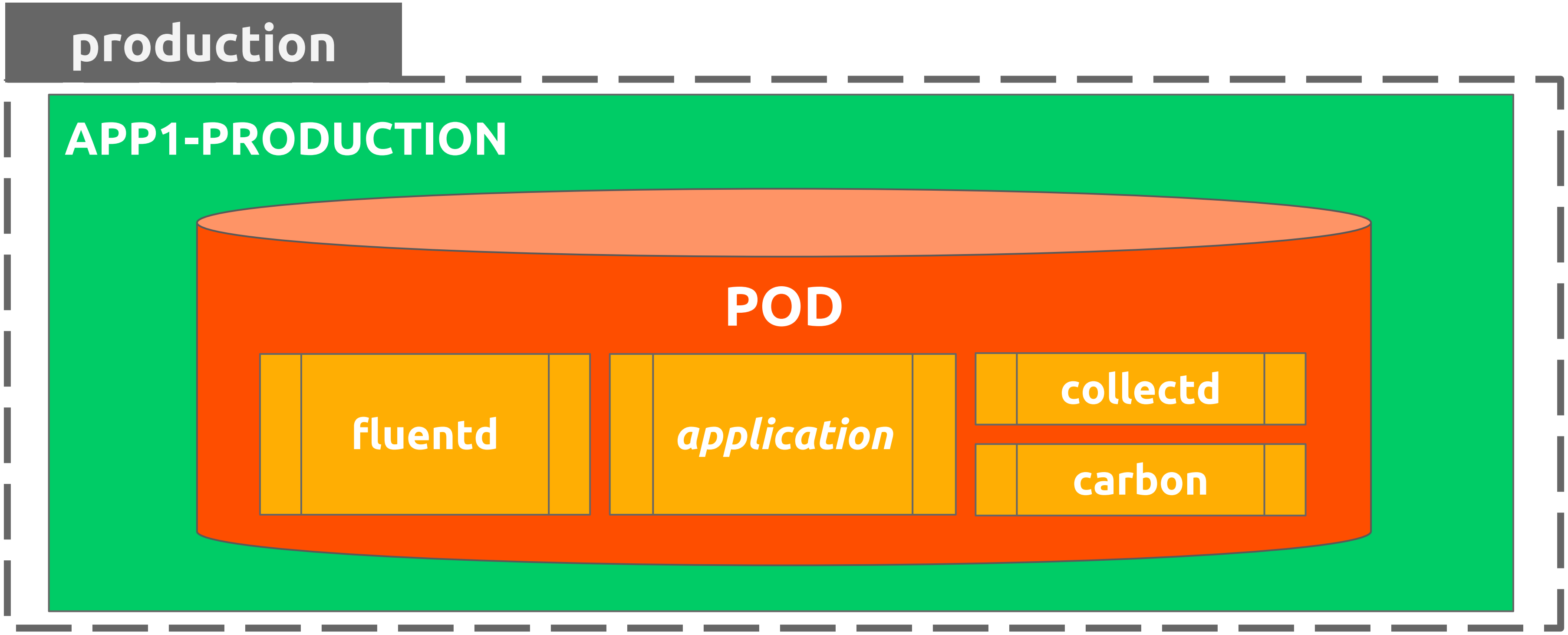
"To ingress or not to ingress? .."



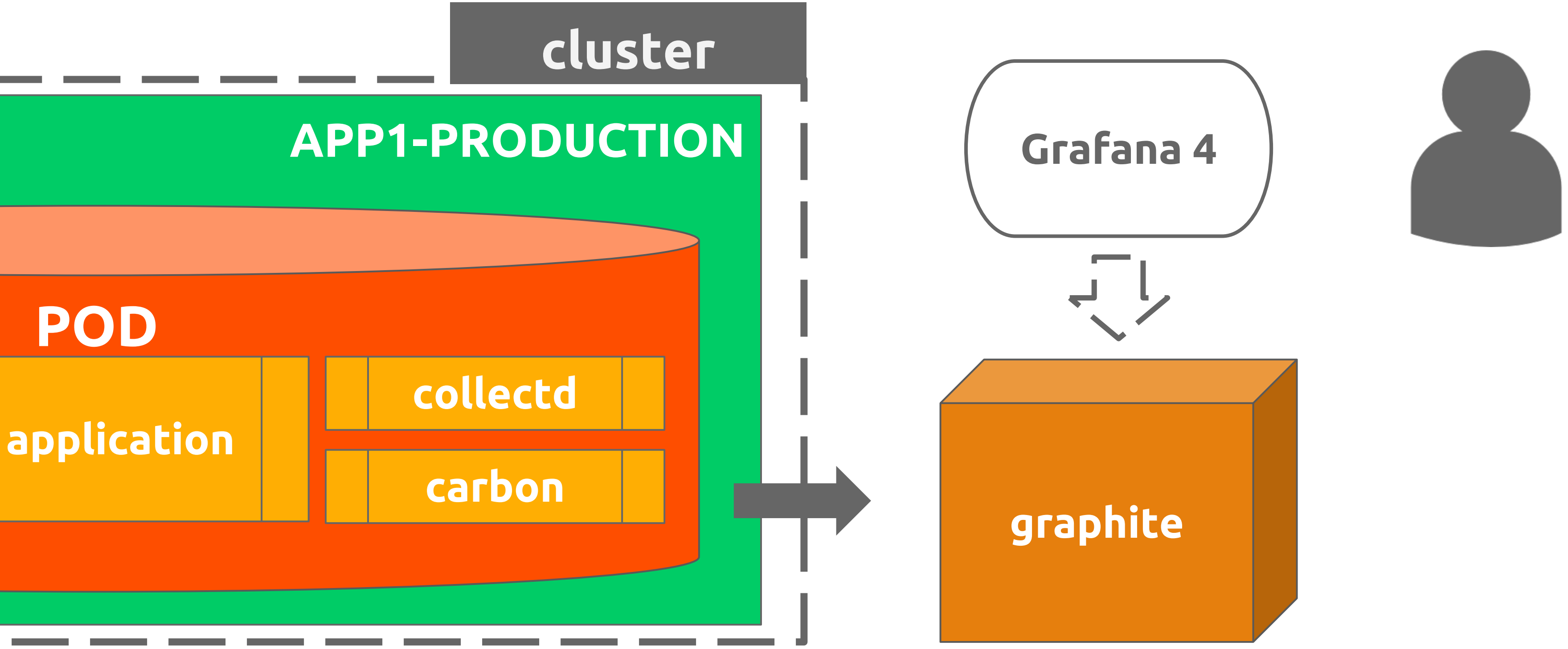
- easier DNS management
- customizable **proxy server**

- 3rd party tool
- requires **external sync**
- **all requests** go through it
- reload risks

Kubernetes: our architecture and choices



Monitoring and alerting: grafana/graphite



lastminute.comgroup

icons from http://www.flaticon.com

Zero downtime (1): graceful shutdown

deployment.yaml

```
lifecycle:
  preStop:
    exec:
      command: ["/stop_helper.sh"]
```

stop_helper.sh

```
#!/bin/bash

wget http://localhost:8002/stop
```

Zero downtime (2): graceful startup

JobsExecutor.java

```
private CompletableFuture run(Stream<CompletableFuture> startupJobs)
 {

   return allOf(startupJobs.toArray(CompletableFuture[]::new))
     .thenAccept(this::raiseReadinessUp)
     .exceptionally(this::shutdown);

 }
```

Automate everything: pipeline DSL

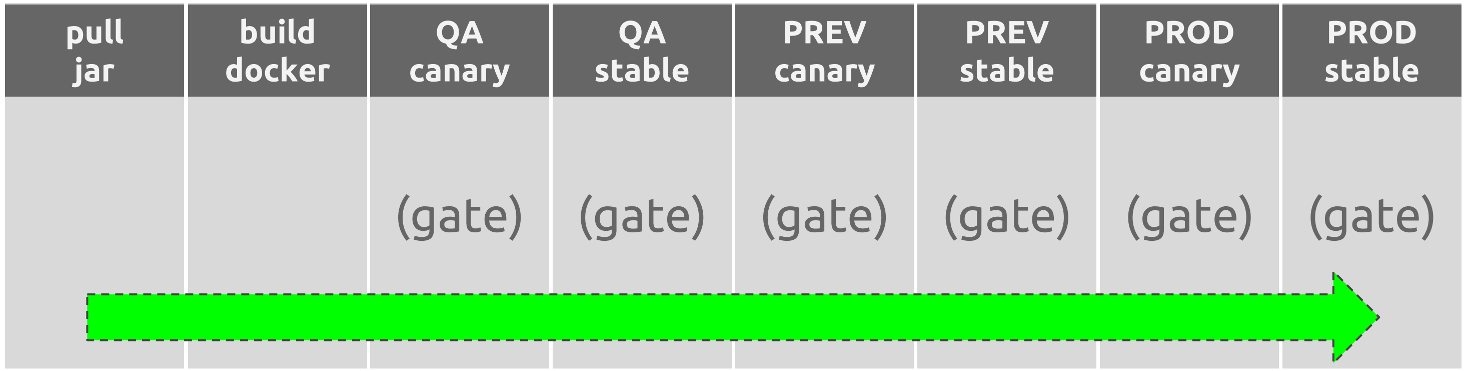
pipeline

```
microservice = factory.newDeployRequest()
        .withArtifact("com.lastminute.application1",2)
        .fromGitRepo("git.lastminute.com/team/application")

lmn_deployCanaryStrategy(microservice,"qa")
lmn_deployCanaryStrategy(microservice,"preview")
lmn_deployCanaryStrategy(microservice,"production")
```

Automate everything: pipeline

- git push
  - continuous integration
  - **continuous delivery**

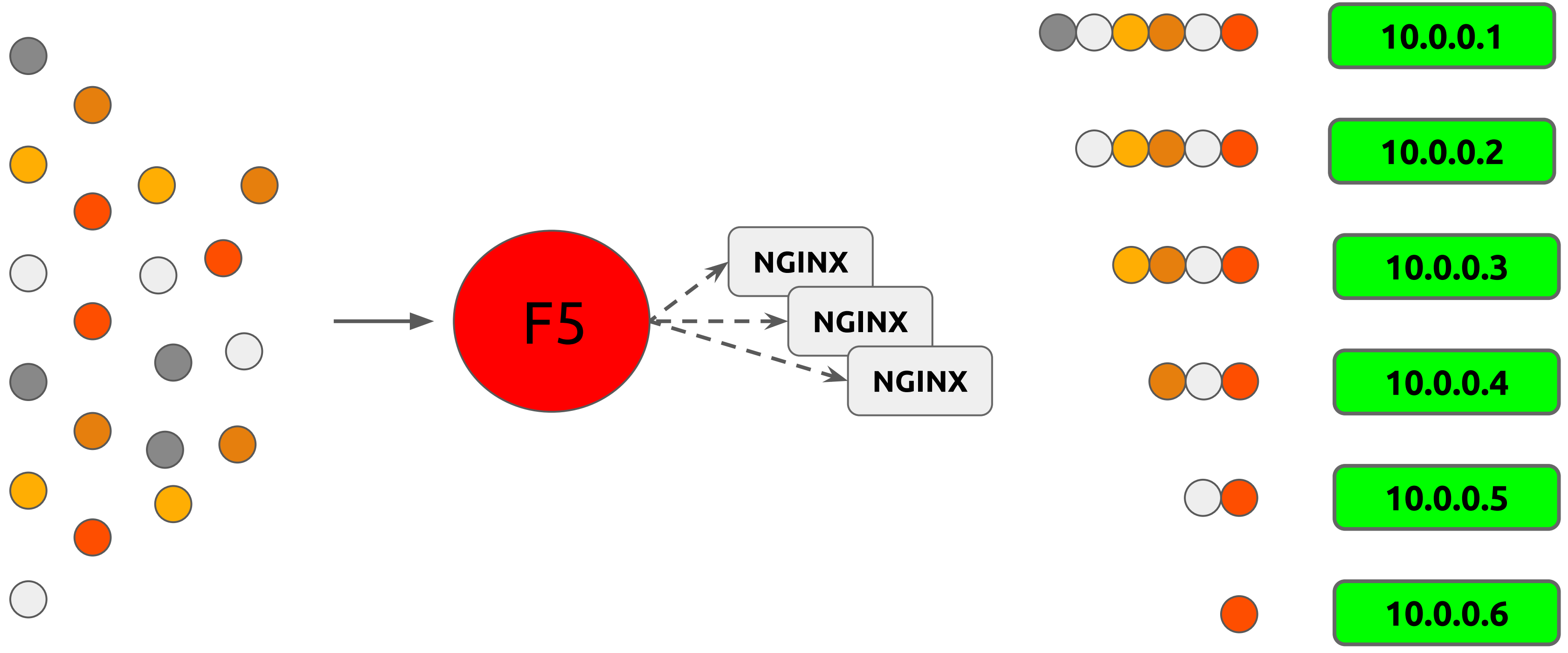


pull jar	build docker	QA canary	QA stable	PREV canary	PREV stable	PROD canary	PROD stable
		(gate)	(gate)	(gate)	(gate)	(gate)	(gate)

.. failure ..

https://www.flickr.com/photos/ghost_of_kuji/2763674926

nginx ingress controller problem



There’s light .. at the end

https://www.pexels.com/photo/grayscale-photography-of-person-at-the-end-of-tunnel-211816/

Give me the numbers .. again!

- **20K req/sec** in the new cluster
- **10 minutes** to create a new environment
- whole pipeline runs in **16 minutes**
  - **4 minutes** to release 100 instances of a new version
- **2M metrics/minute** flows

Yes, we're hiring!

THANKS

www.lastminutegroup.com