

CS 4530 Software Engineering

Lesson 8.3: Continuous Delivery

Jonathan Bell, Adeel Bhutta, Ferdinand Vesely, Mitch Wand

Khoury College of Computer Sciences
© 2022, released under [CC BY-SA](#)

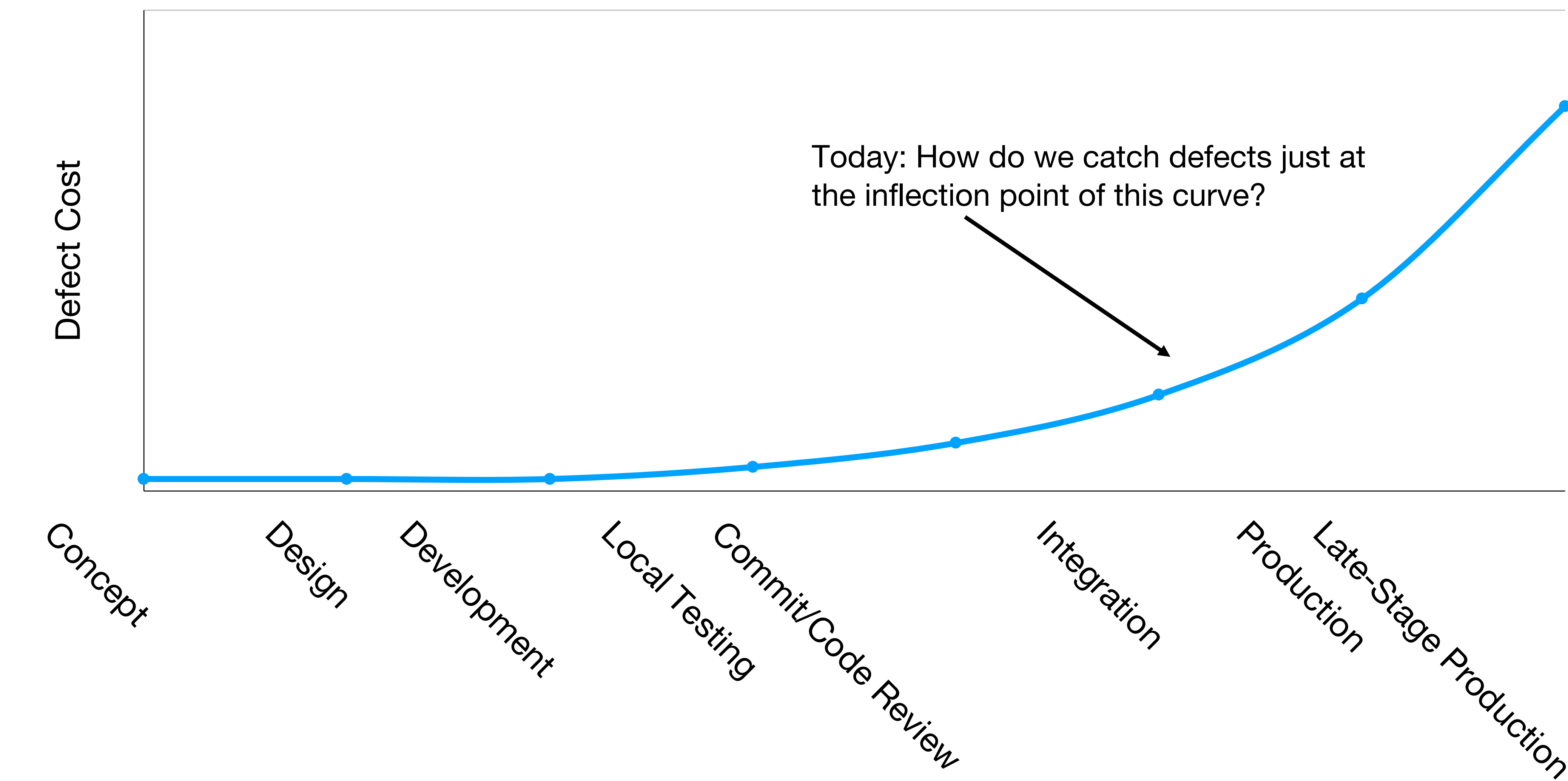
Learning Objectives for this Lesson

By the end of this lesson, you should be able to...

- Describe how continuous delivery helps to catch errors sooner in a product's lifecycle
- Describe the distinction between a DevOps and “traditional” developer/operator mentality
- Describe strategies for performing quality-assurance on software as and after it is delivered

Cost to Fix a Defect Over Time

Rough estimate



Deploying New Code

The best that we can hope for?



“If stuff blows up it affects a very small percentage of people”



Instagram cofounder and CTO Mike Krieger

Continuous Delivery

“Faster is safer”: Key values of continuous delivery

- Release frequently, in small batches
- Maintain key performance indicators to evaluate the impact of updates
- Phase roll-outs
- Evaluate business impact of new features

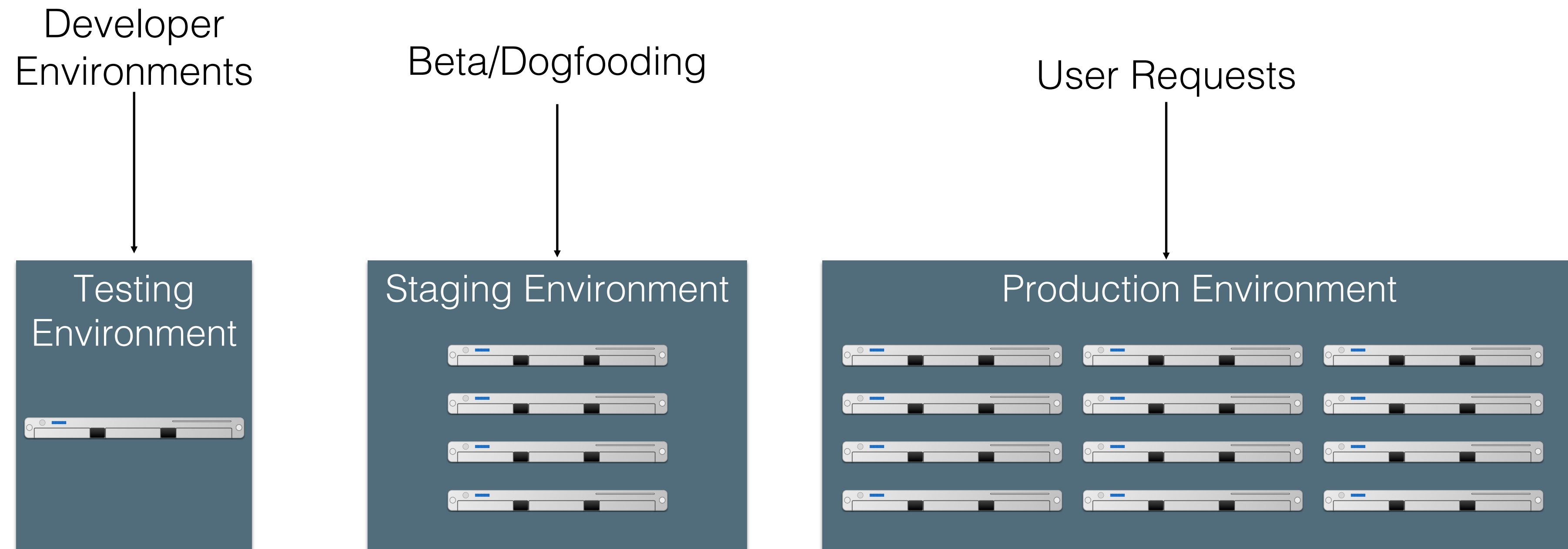
Staging Environments

Enabling Continuous Delivery

- As software gets more complex with more dependencies, it's impossible to simulate the whole thing when testing
- Idea: Deploy to a complete production-like environment, but don't have everyone use it
 - Examples:
 - “Eat your own dogfood”
 - Beta/Alpha testers
- Lower risk if a problem occurs in staging than in production

Test-Stage-Production

Continuous Delivery in Action



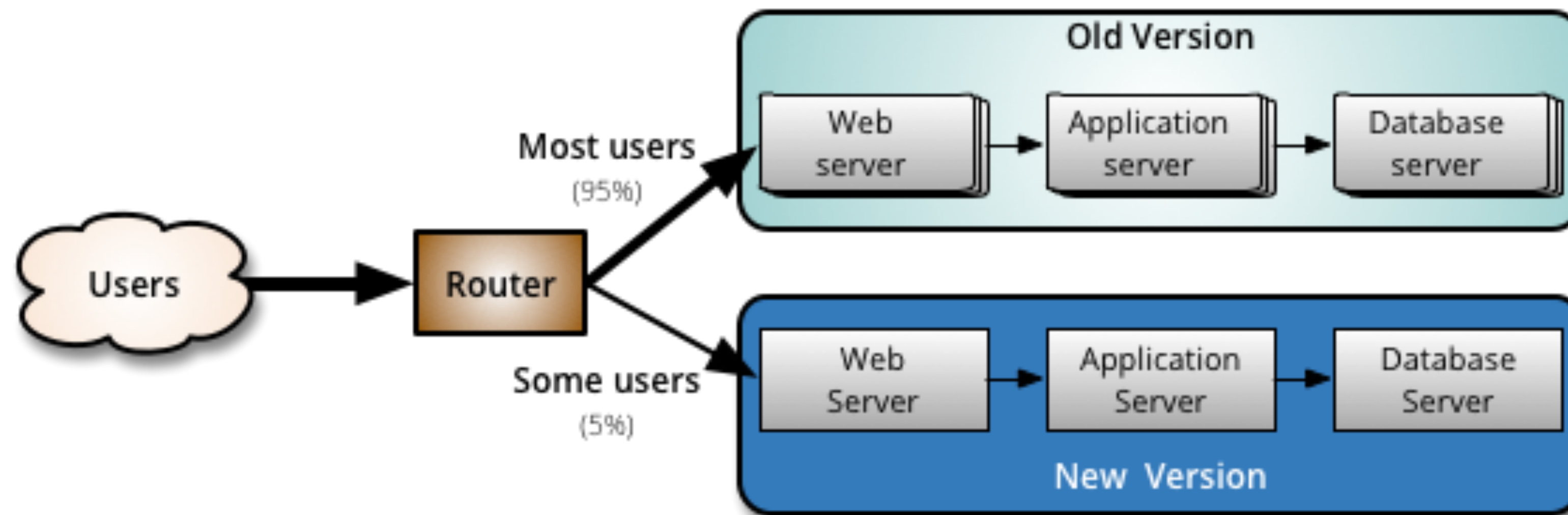
Revisions are “promoted” towards production



Q/A takes place in each stage (including production!)

A/B Deployments with Canaries

Mitigating risk in continuous delivery



Monitor both:
But minimize impact of problems in new version

Operations Responsibility

DevOps in a slide

- Once we **deploy**, someone has to monitor software, make sure it's running OK, no bugs, etc
- Assume 3 environments:
 - Test, Staging, Production
- Whose job is it?

	Developers			Operators		
Waterfall				Test	Staging	Production
Agile	Test			Staging	Production	
DevOps	Test	Staging	Production			Production

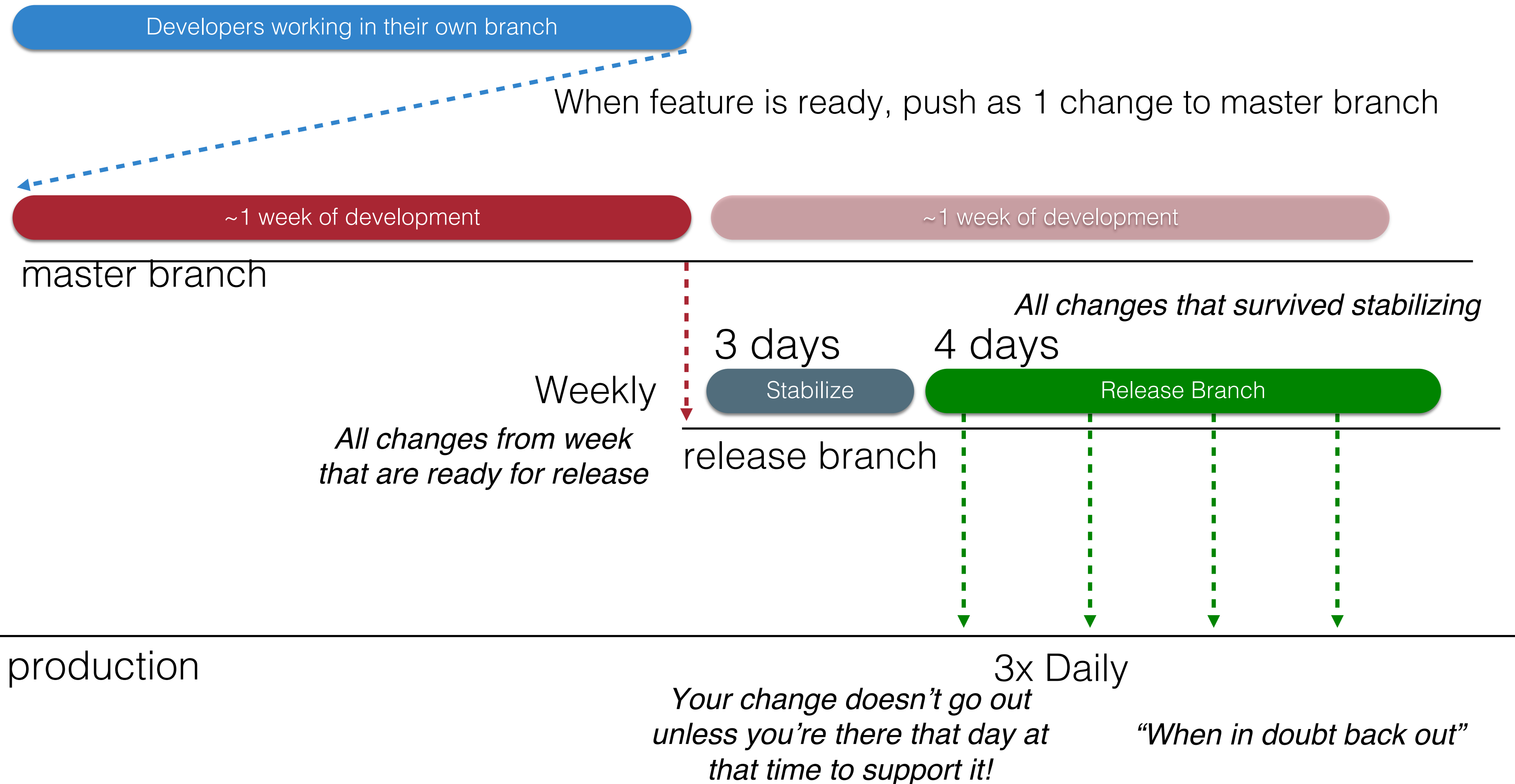
Release Pipelines

How quickly is my change deployed?

- Even if you are deploying every day, you still have some latency
- A new feature I develop today won't be released today
- But, a new feature I develop today can begin the **release pipeline** today (minimizes risk)
- **Release Engineer**: gatekeeper who decides when something is ready to go out, oversees the actual deployment process

Deployment Example: Facebook.com

Pre-2016



Deployment Example: Facebook.com

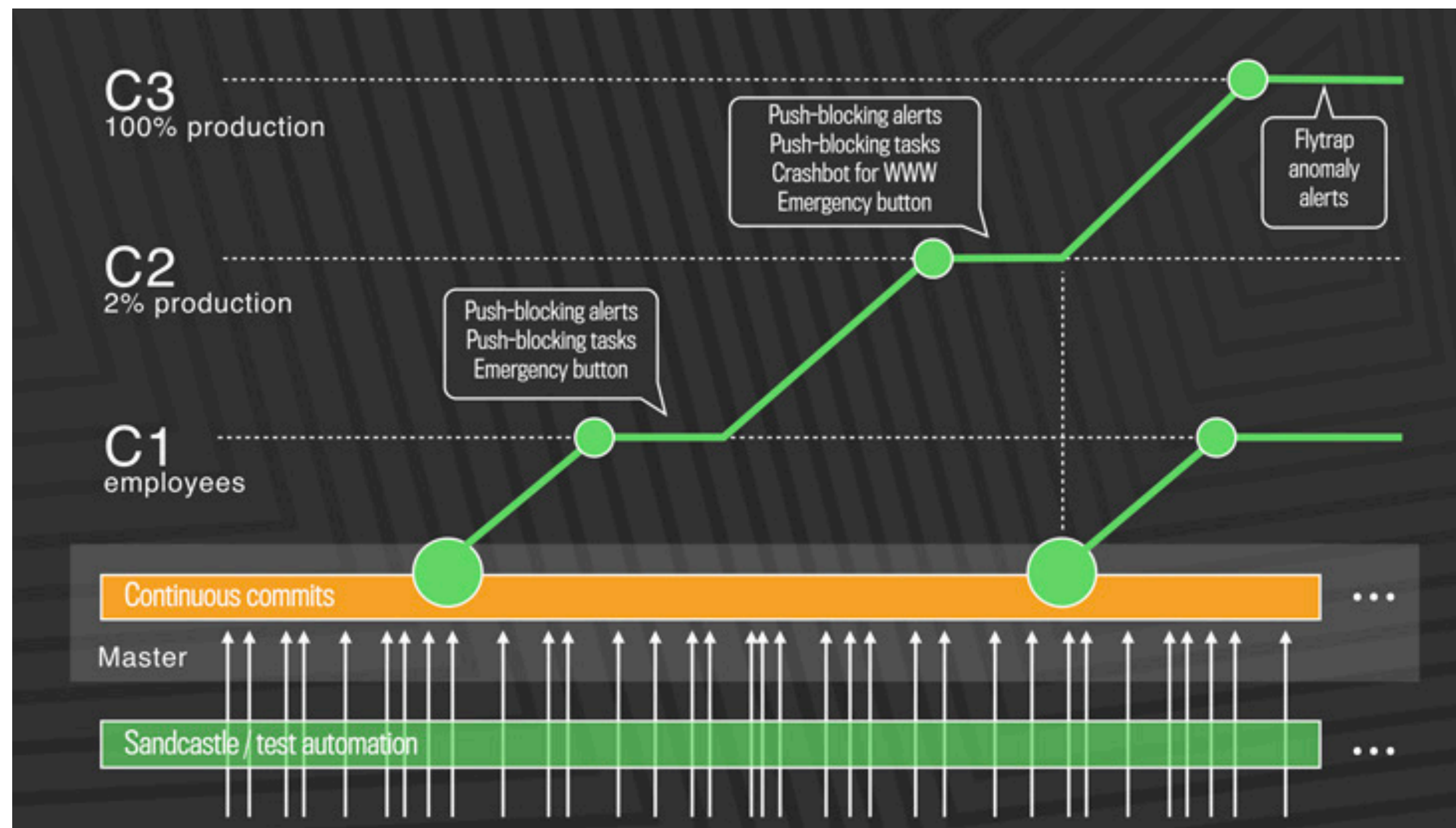
Chuck Rossi, Director Software Infrastructure & Release Engineering @ Facebook



“Our main goal was to make sure that the new system made people’s experience better — or at the very least, didn’t make it worse. After almost exactly a year of planning and development, over the course of three days in April 2017 **we enabled 100 percent of our production web servers to run code deployed directly from master.**”

Deployment Example: Facebook.com

Post-2016: Truly continuous releases from master branch



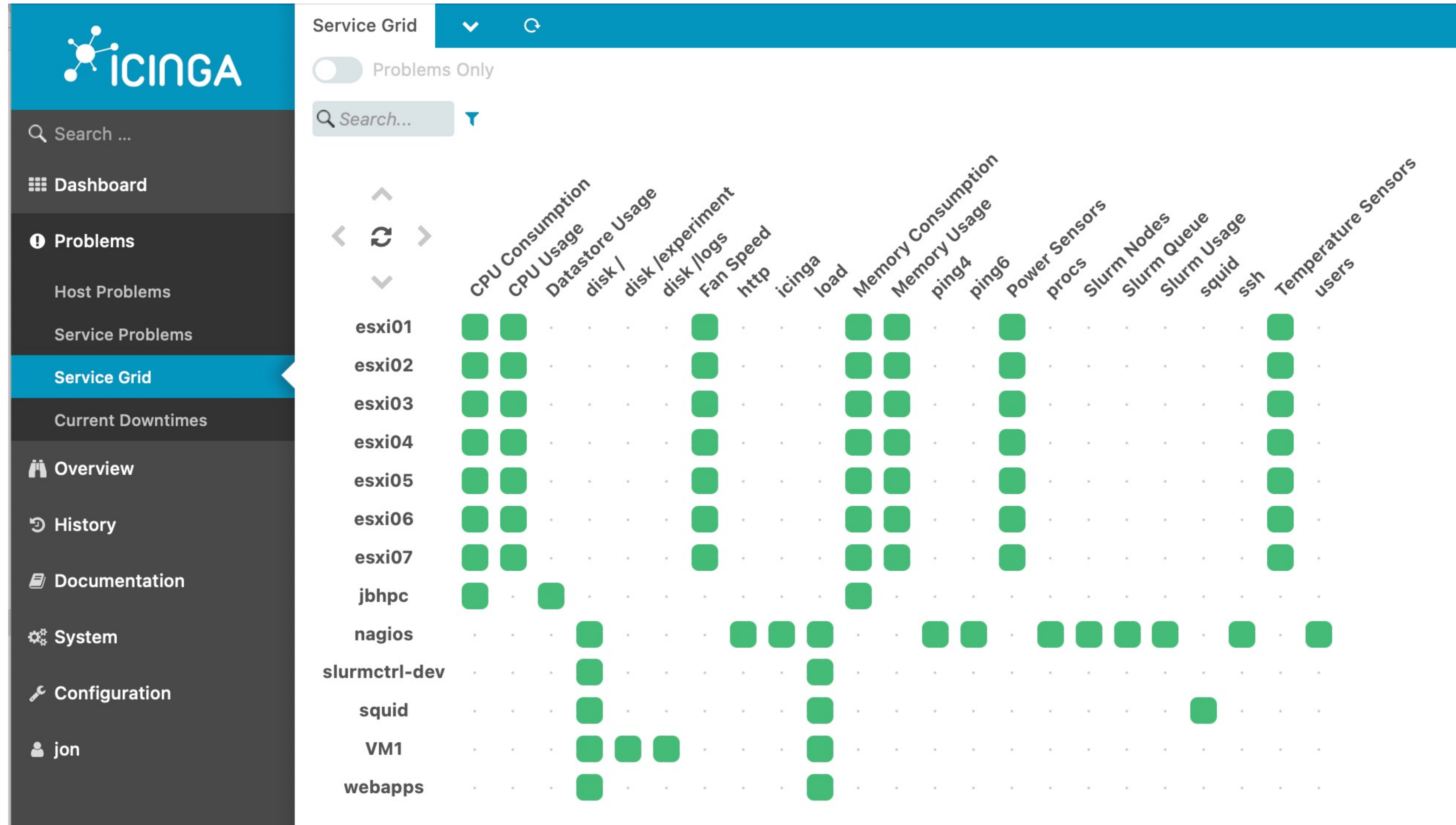
<https://engineering.fb.com/2017/08/31/web/rapid-release-at-massive-scale/>

Monitoring

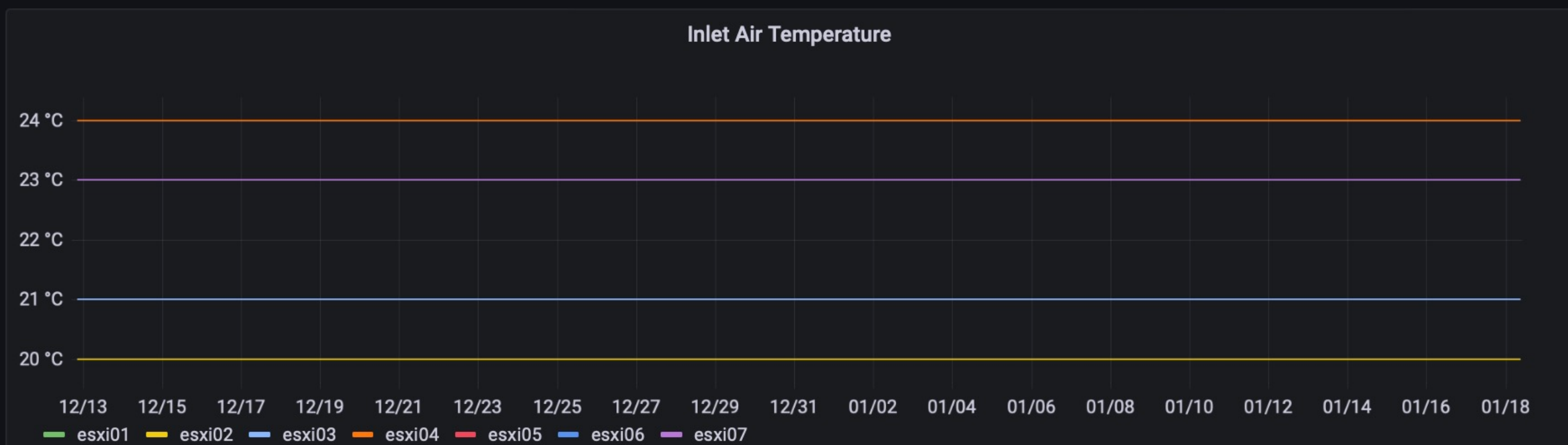
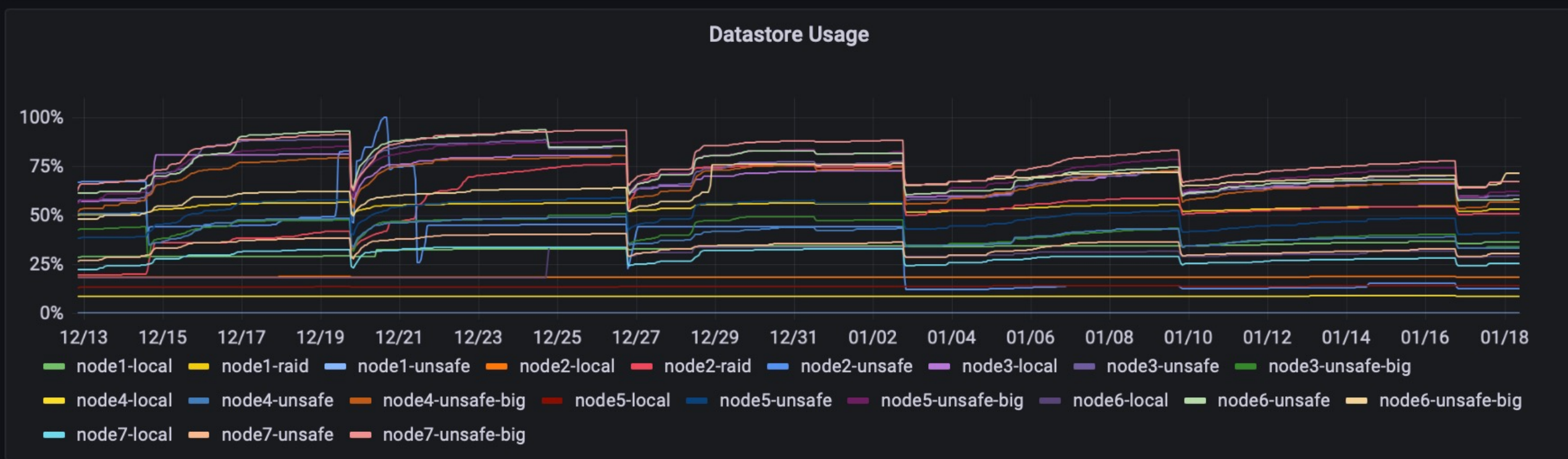
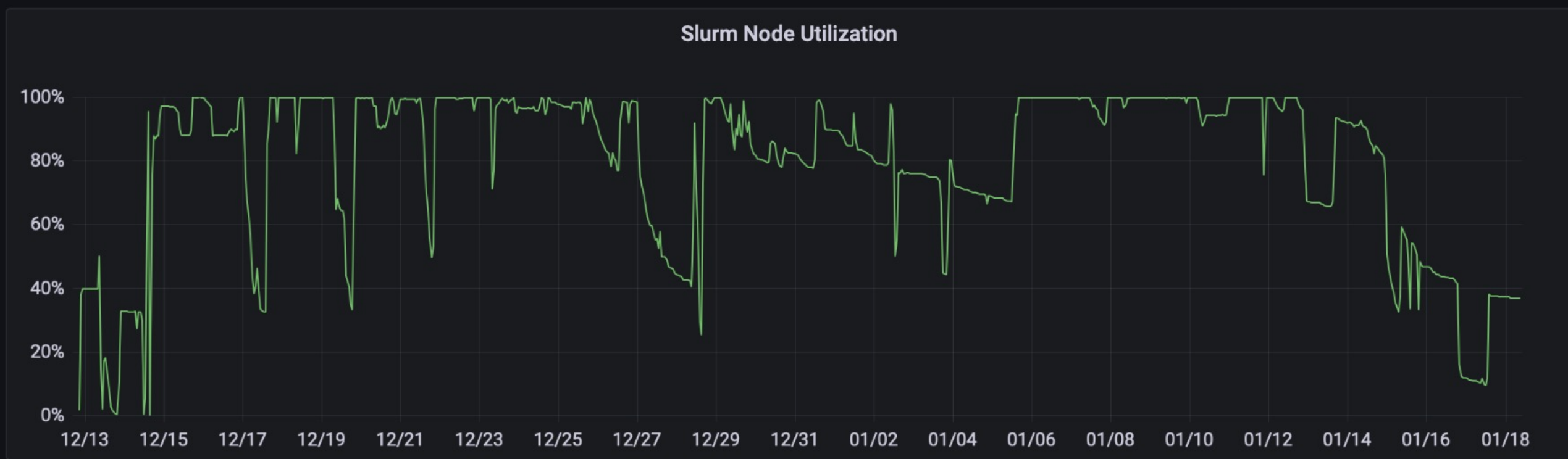
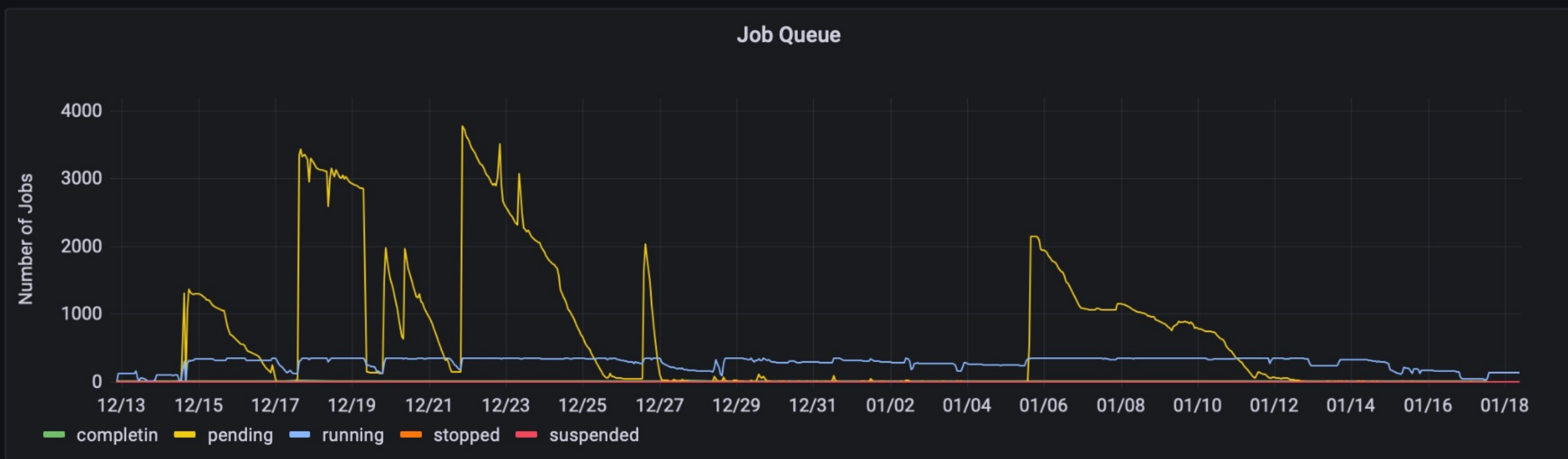
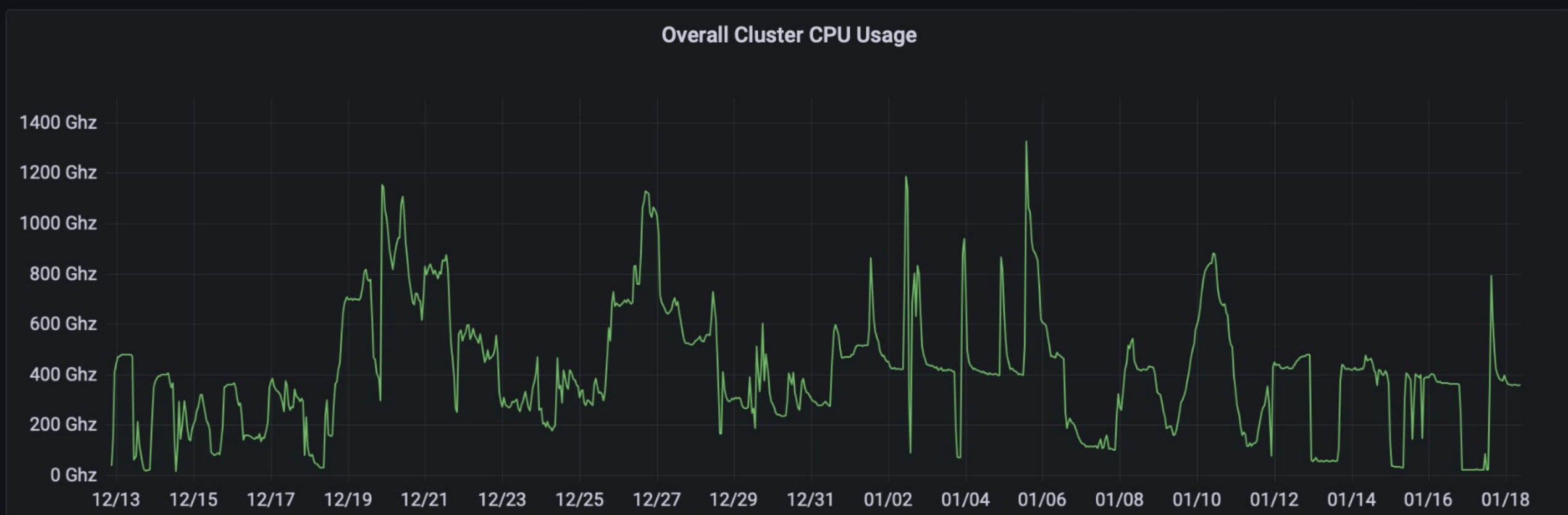
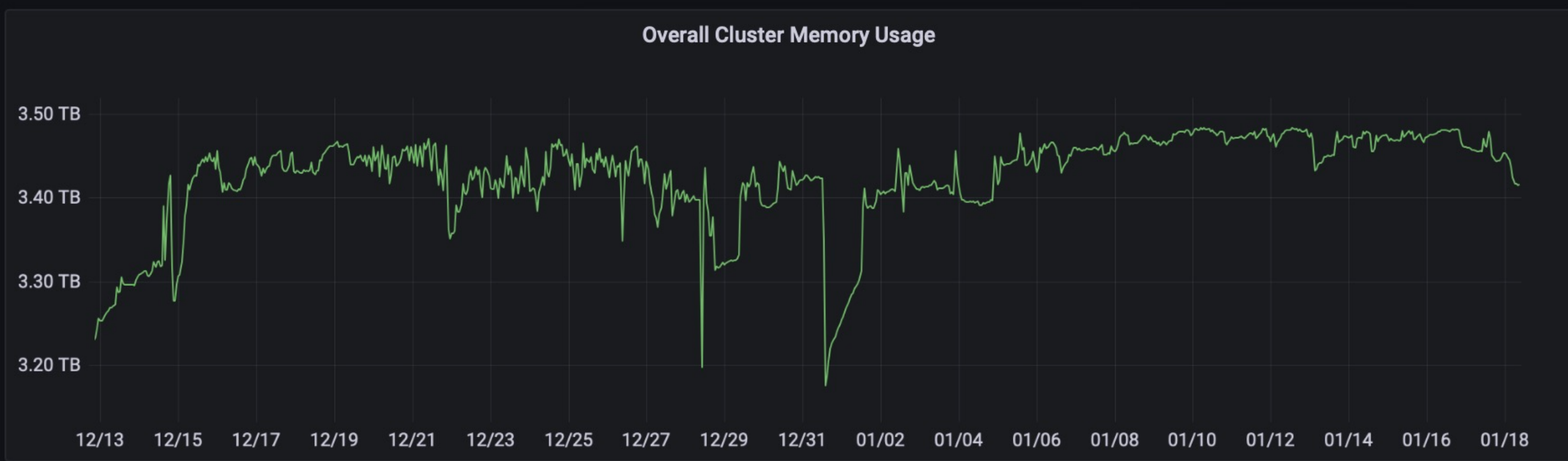
The last step in continuous deployment: track metrics

- Hardware
 - Voltages, temperatures, fan speeds, component health
- OS
 - Memory usage, swap usage, disk space, CPU load
- Middleware
 - Memory, thread/db connection pools, connections, response time
- Applications
 - Business transactions, conversion rate, status of 3rd party components


Monitoring Services Aggregate System Status



Monitoring Dashboards Help Gather Insights



Monitoring Services Take Automated Actions



Search ...

Dashboard

Problems

Overview

History

Event Grid

Event Overview

Notifications

Timeline

Documentation

System

Configuration

jon

Notifications

« 1 2 3 4 5 6 7 ... 24 25 » # 25

Sort by Notification Start

Search...

OK

2022-02-18 08:49:05

Slurm Nodes on nagios

OK - 0 nodes unreachable, 332 reachable

Sent to jon

OK

2022-02-18 08:49:05

Slurm Nodes on nagios

OK - 0 nodes unreachable, 332 reachable

Sent to icingaadmin

WARNING

2022-02-18 08:45:05

Slurm Nodes on nagios

WARNING - 7 nodes unreachable, 326 reachable

Sent to jon

WARNING

2022-02-18 08:45:05

Slurm Nodes on nagios

WARNING - 7 nodes unreachable, 326 reachable

Sent to icingaadmin

CRITICAL

2022-02-18 08:42:05

Slurm Nodes on nagios

CRITICAL - 65 nodes unreachable, 161 reachable

Sent to icingaadmin

CRITICAL

2022-02-18 08:42:05

Slurm Nodes on nagios

CRITICAL - 65 nodes unreachable, 161 reachable

Sent to jon

WARNING

2022-02-18 08:40:05

Slurm Nodes on nagios

WARNING - 12 nodes unreachable, 205 reachable

Sent to icingaadmin

WARNING

2022-02-18 08:40:05

Slurm Nodes on nagios

WARNING - 12 nodes unreachable, 205 reachable

Sent to jon

CRITICAL

2022-02-18 08:34:07

Slurm Nodes on nagios

CRITICAL - 204 nodes unreachable, 145 reachable

Sent to icingaadmin

Notification

Current Service State

UP since 2021-11

nagios ::1 127.0.0.1

OK for 1m 52s

Service: Slurm Nodes

Event Details

Type

Notification

Start time

2022-02-18 08:42:05

End time

2022-02-18 08:42:05

Reason

Normal notification

State

CRITICAL

Escalated

No

Contacts notified

2

Output

CRITICAL - 65 nodes unreachable, 161 reachable

Monitoring Services Take Automated Actions

Automatically detecting irregular behavior at Netflix

