O'REILLY®
Velocity

# Containers can actually improve your security story(!)

Maya Kaczorowski, Google Cloud
June 12 2019

velocityconf.com/ca
#VelocityConf

# Maya Kaczorowski

Security PM, Google Cloud

@MayaKaczorowski

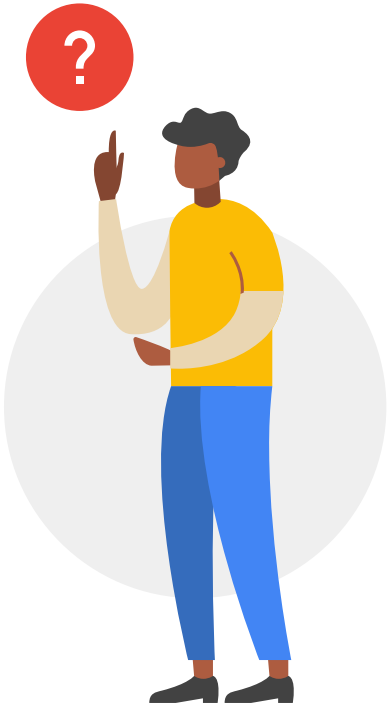
Google Cloud

## Objection:

“My security team is opposed to containers and Kubernetes”

# Security people like to complain about containers and Kubernetes

- What's a koober net ease
- I can't use my IDS, firewall, ...
- Containers don't contain
- I am stuck with it, help me



“70 percent of change programs fail to achieve their goals, largely due to employee resistance and lack of management support.”

Changing change management,
McKinsey & Co.

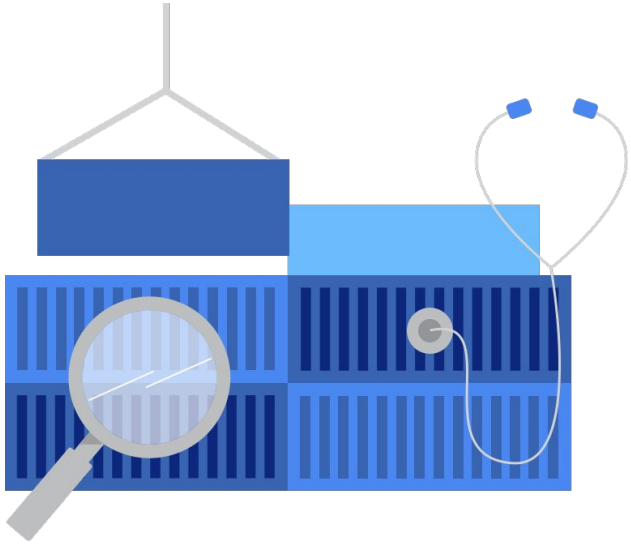
# Agenda

1. How container security is different
2. Traditional software supply chain and patch management
3. Ideal software supply chain and best practices in image maintenance, patching, and validation

# How container security is different

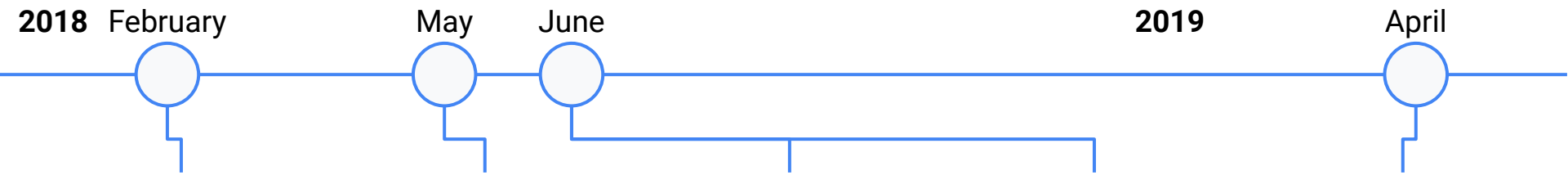

Google Cloud

# … container security isn't that different from any other security



Google Cloud

# Threats seen in the wild




**Tesla**

Unsecured Kubernetes dashboard with cloud account credentials

**Used to mine cryptocurrency**

**Shopify**

Researcher could access and replay kubelet credentials

**Not exploited**

**Weight Watchers**

Unsecured Kubernetes dashboard with sensitive data, including credentials

**Not exploited**

**Docker Hub**

Public images with embedded cryptocurrency mining malware

**Used to mine cryptocurrency**

**Docker Hub**

Database with 190k+ Docker Hub accounts exposed

**Unknown impact**

Google Cloud

# Container security threats & risks

## INFRASTRUCTURE SECURITY

- Privilege escalation
- Credential compromise
- Kubernetes API compromise
- Over-privileged users

## SOFTWARE SUPPLY CHAIN

- Unpatched vulnerability
- Supply chain vulnerability
- Zero day exploit on common library

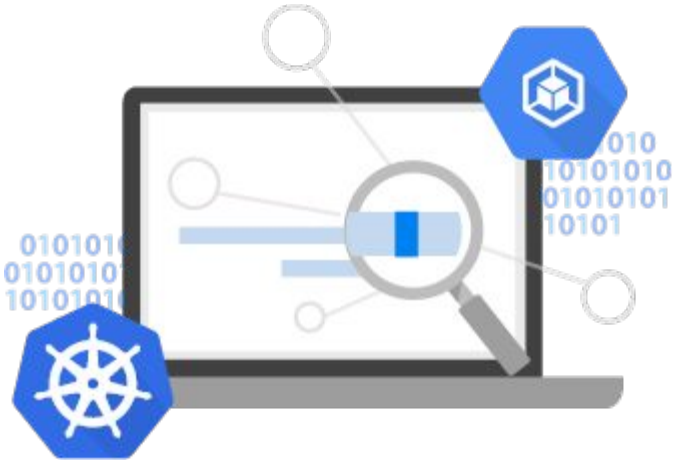
## RUNTIME SECURITY

- DDoS
- Node compromise and exploit
- Container escape
- Flood event pipeline

# INFRASTRUCTURE SECURITY

Is my infrastructure
**secure for developing containers?**

- How can I use Kubernetes security features to protect my identities, secrets, and network?
- How can I use native GCP functionality, like IAM, audit logging, and networking?




Google Cloud

# SOFTWARE SUPPLY CHAIN

Is my container image
**secure to build and deploy?**

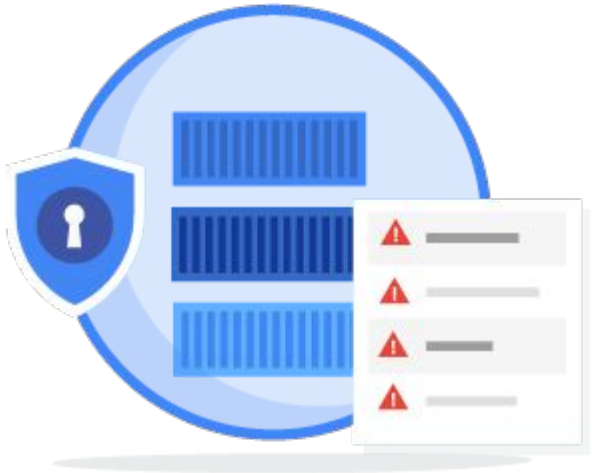
- How can I make sure my container images are vulnerability-free?
- How can I make sure the images I built aren't modified before they are deployed?


Google Cloud

# RUNTIME SECURITY

Is my container
**secure to run?**

- How can I identify a container acting maliciously in production?
- How can I take action to protect and isolate my workload?
- How can I securely scale my containers deployment?



Google Cloud

# How is securing a container different?

## Surface of Attack

**Minimalist host OS** and limits the surface of an attack.

## Resource Isolation

Host resources are **separated using namespaces and cgroups.**

## Permissions

**Access controls are** for app privileges and shared resources.

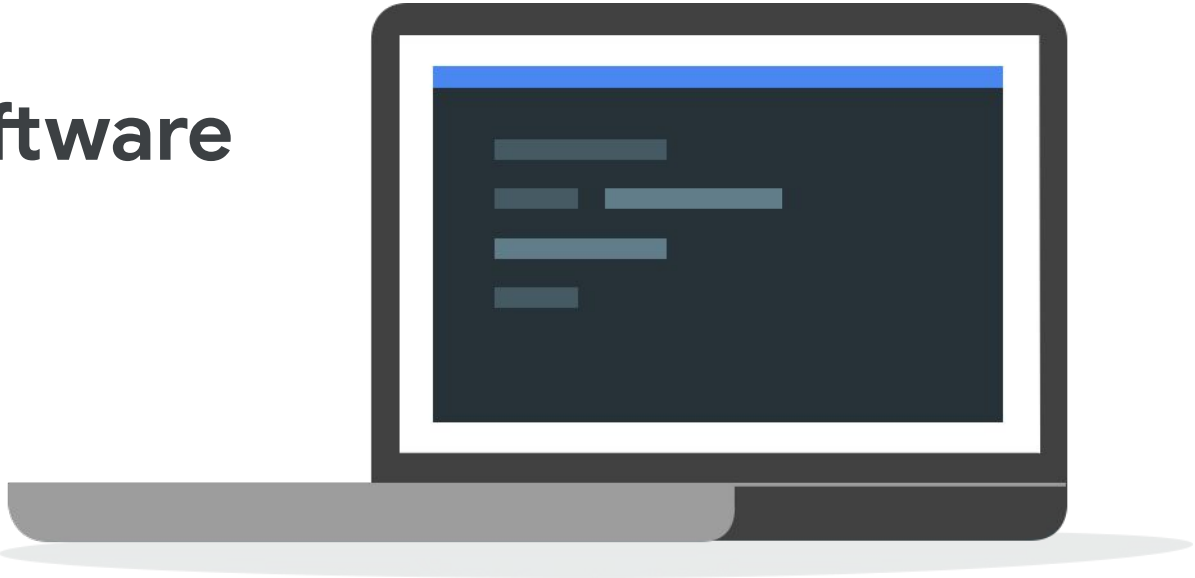
## Lifecycle

Containers have a **shorter, better defined lifecycle.**

# Traditional software supply chain and patch management


Google Cloud

# Traditional software supply chain



Google Cloud

# Traditional patch management

01

### Get patch

From the distributor, some random mailing list, a vendor. Not always sent to the security team.

02

### Take down server n=1 and apply patch

Test the patch in prod! Take some unimportant workload down to make sure nothing goes too bad.

03

### Repeat for n servers, where n is unknown

It worked! Now do it again, for everything you think is affected. Miss a bunch of it.

# Problems with traditional patch management

- Spreadsheet-driven management
- Down time
- 0days are scary
- Unclear what's running in your infrastructure / what's running where / if you even need a patch

# Ideal software supply chain


Google Cloud

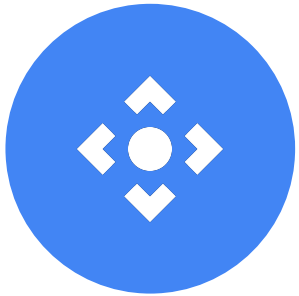
Containers are meant to be
short-lived
frequently redeployed
immutable
and help you ‘shift left’

Google Cloud

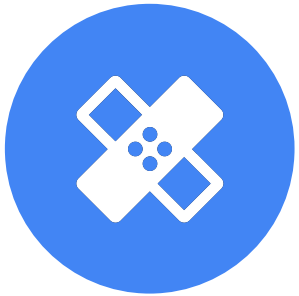
# DevSecOps?!?

Google Cloud

# Running containers allows you to adopt a fundamentally different security model



Containers give you a **software supply chain**



Containers let you **patch continuously**, automatically

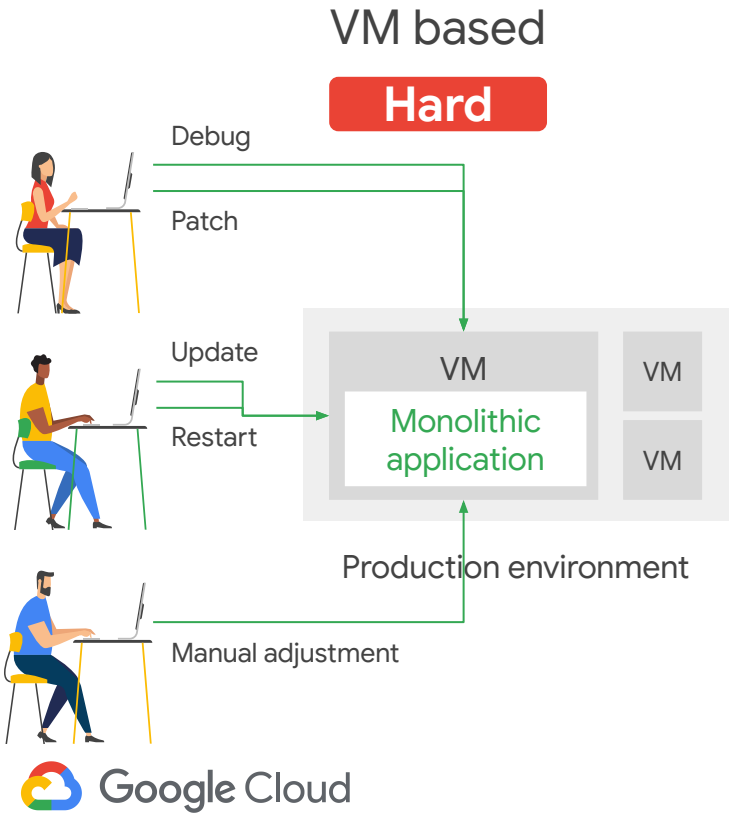
Containers mean you can actually **tell if you're affected** by a new vulnerability

Google Cloud

# Containers give you a software supply chain

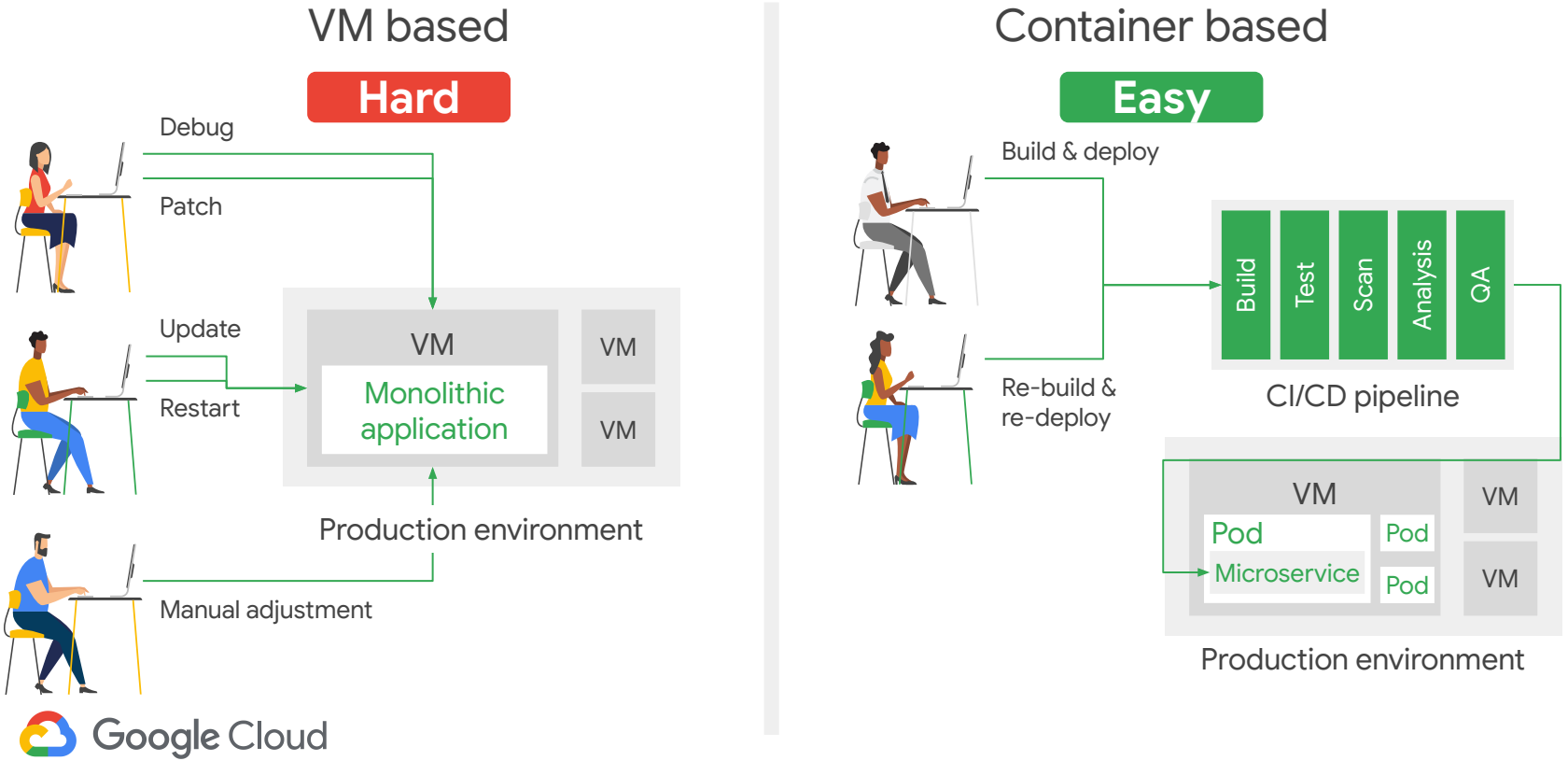
Google Cloud

# What's different about supply chains with containers



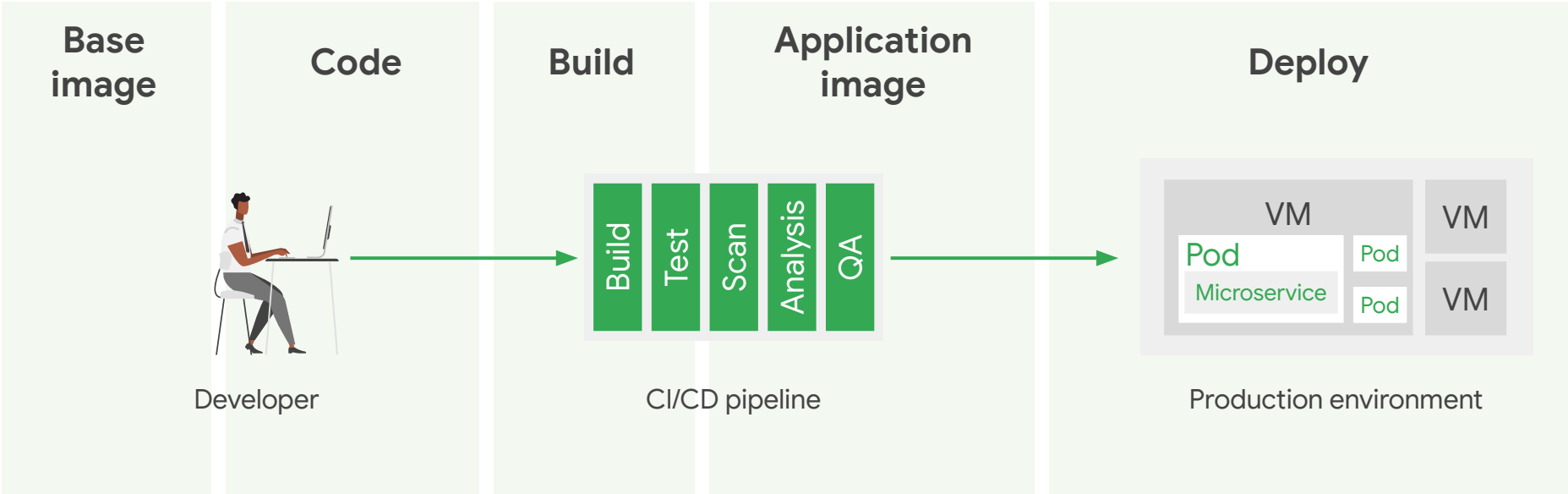

Google Cloud

# What's different about supply chains with containers




Google Cloud

# Containers let you enforce a software supply chain




Google Cloud

Containers let you patch continuously, automatically

Google Cloud

# Constantly patch your registry... and roll out as normal

01

**Patch the image in your registry**

Figure out what's affected, and apply the patch everywhere you need it.

02

**Test, validate, and roll out**

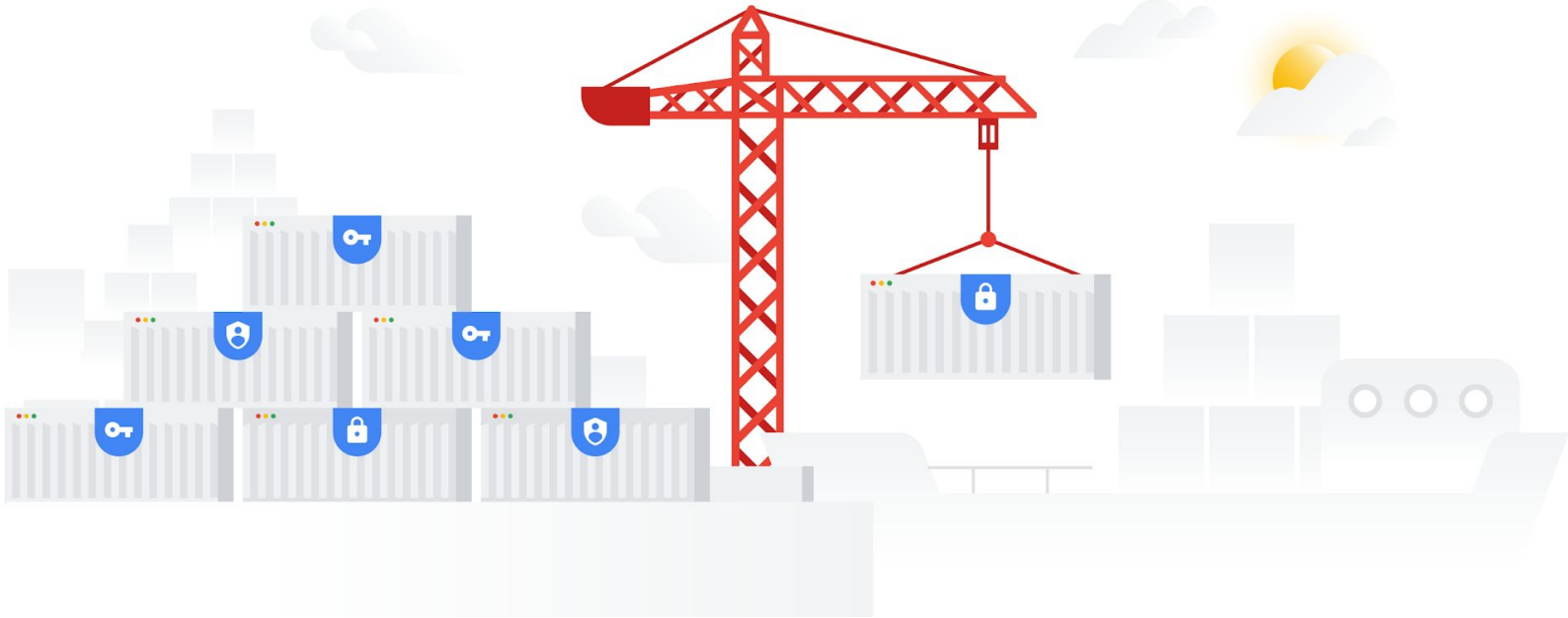
Roll out the patch like you would any other infrastructure change, going incrementally.

03

**Load balance traffic over**

When testing is successful, move traffic over to the new, patched workload, with no downtime.

# Containers enable passive patching



Google Cloud

not just uptime, but

# up-to-time

# Vulnerability mitigation strategies

## Update packages

apt-get update & upgrade gets you pretty far.  Do this daily.

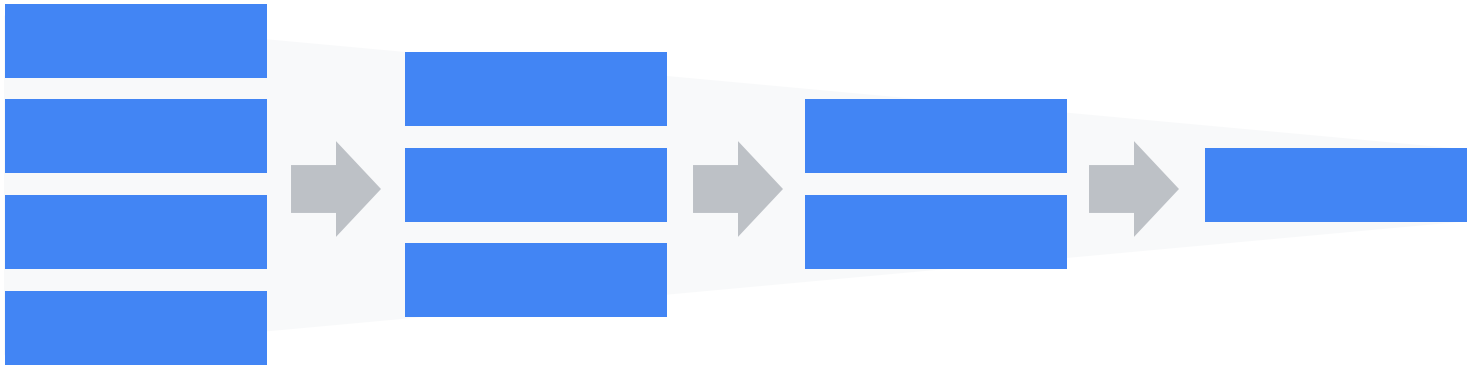
## Remove packages

Do you really need $6.022 \times 10^{23}$ debian packages installed on your production image?

## Smaller distro

In many cases, you can get away with a smaller distro like Alpine or Debian Slim.

# Moving to a smaller base

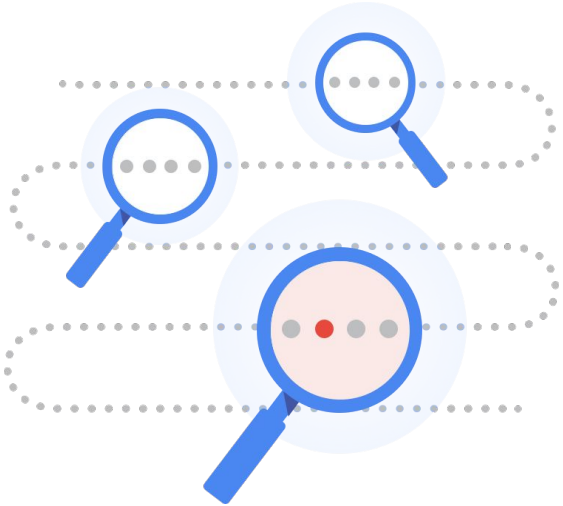



Google Cloud

Containers mean you can actually tell if you're affected by a new vulnerability

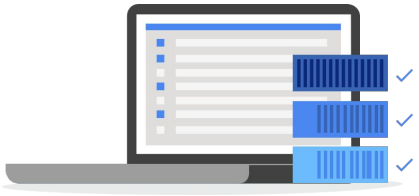
Google Cloud

# Check your registry and compare to what you deployed



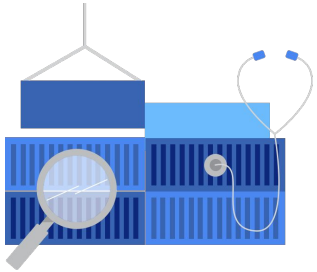
Google Cloud

# Figure out what's in production



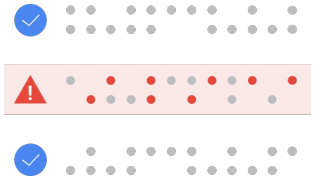
## Find all the containers in prod

kubectl get pods resolve everything to a digest



## Find out what is in those containers

Package manifests, application dependencies



## Find out what vulnz are in those packages

Cross reference BOM with CVE databases

# Instead container security should be

- Centralize and lock down release pipeline
- Build images from trusted sources
- Streamline image scanning and analysis
- Deploy only trusted images
- Monitor continuously

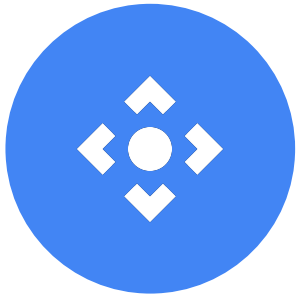
# Start here

You have a container registry
> Scan for vulnerabilities

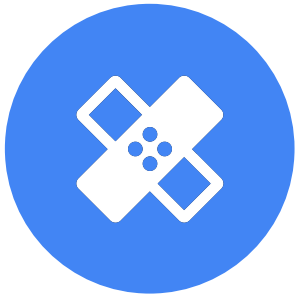
You have a mandated base image
> Make it minimal

You have a centralized CI/CD pipeline
> Enforce what's deployed

# Running containers allows you to adopt a fundamentally different security model



Containers give you a **software supply chain**



Containers let you **patch continuously**, automatically

Containers mean you can actually **tell if you're affected** by a new vulnerability

Google Cloud

# Learn more

Blog post: goo.gl/Ew6hYa

cloud.google.com/containers/security

Google Cloud

# Q&A


Google Cloud

# That’s a wrap.

**Learn more:**
cloud.google.com/containers/security

Google Cloud