The Great “k8s.gcr.io” Vanity Domain Flip

Linus Arver (Google)
Stephen Augustus (VMware)

Who We Are

Linus Arver

Container Image Promoter Maintainer
WG K8s Infra contributor
Software Engineer, Google

Stephen Augustus

SIG Release Chair
Kubernetes Release Manager
Senior OSS Engineer, VMware

The Great “k8s.gcr.io” VDF

Overview

- Historical context & rationale
- Infrastructural changes
  - How the promoter works
  - How it’s tested
- Lessons learned

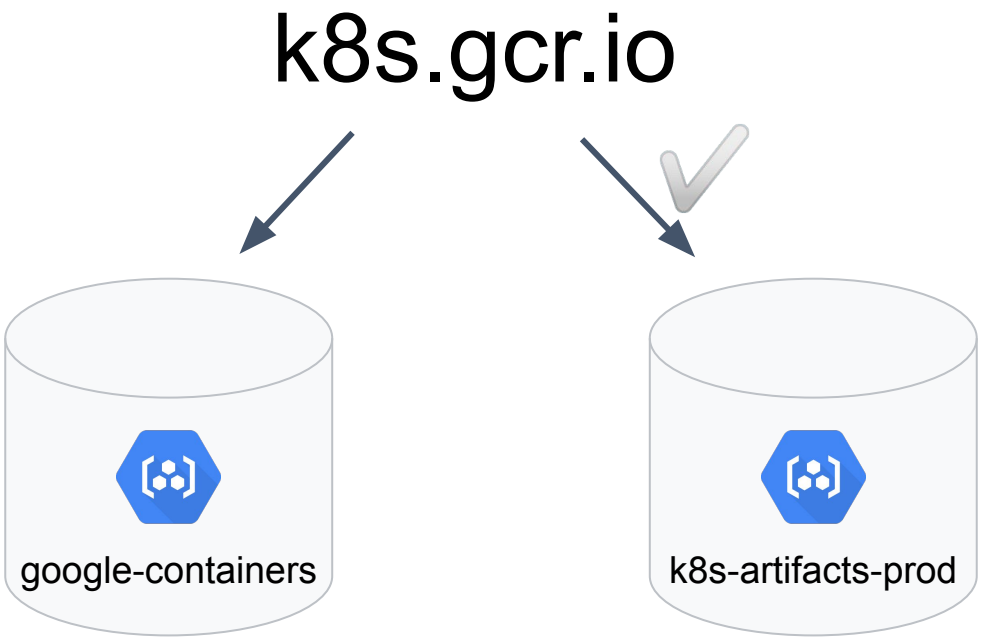
The Great “k8s.gcr.io” VDF

What is “k8s.gcr.io”?

- Vanity domain: essentially, it’s a **human-friendly name** to a **folder** that contains container (Docker) images.
  - **google-containers** (old)
  - **k8s-artifacts-prod** (new)
- The “**k8s.gcr.io**” name is widely used throughout the Kubernetes codebase and configuration logic.
- The **flip** that happened this July made the name point from **google-containers** to **k8s-artifacts-prod.**

The Great “k8s.gcr.io” VDF

Summary of the vanity domain flip



The Great “k8s.gcr.io” VDF

Is that it?

- Infrastructure improvements (e.g., the promoter) were not trivial
- This gave us an opportunity to improve production (and testing!) hygiene, such as:
  - Security
  - Auditability
  - Backups

The Great “k8s.gcr.io” VDF

Historical timeline

- 2018: Internal image promoter released for Googlers
- Early 2020: OSS Container Image Promoter is an improved version of the internal promoter
  - Same idea as internal promoter, but with added **infrastructure**
  - **Auditing + backups!**
- July 24, 2020: Domain flipped to use **k8s-artifacts-prod**!

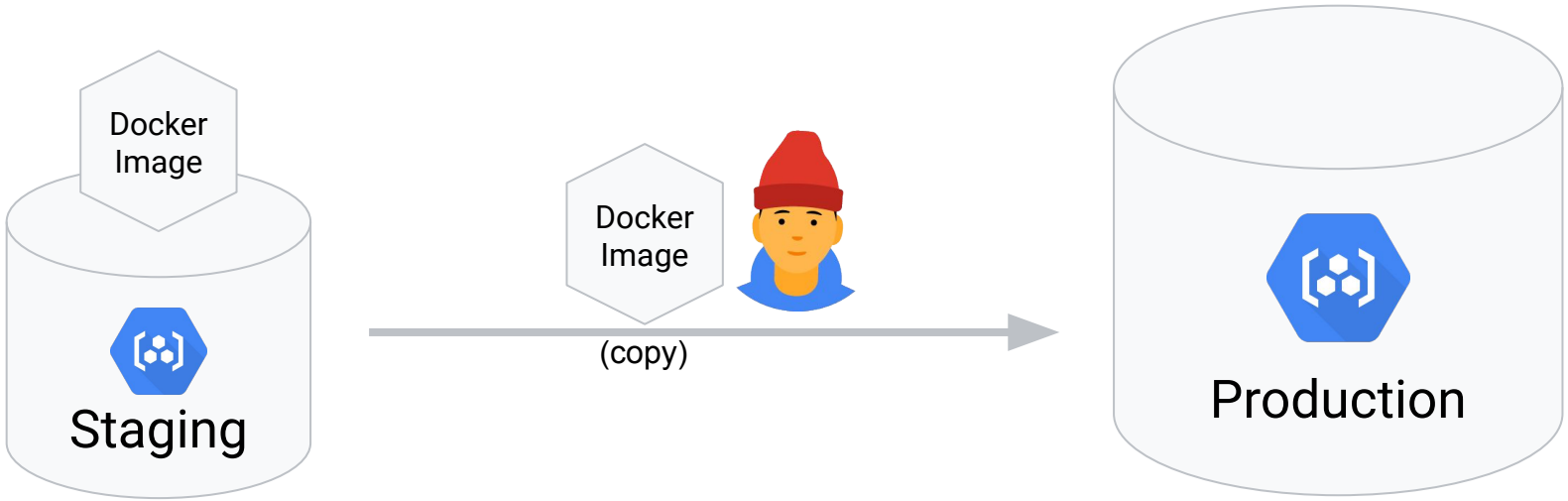
The Great “k8s.gcr.io” VDF

Why did Google need an internal promoter?

- Tighten security posture
- Reduce risk of human error
- Make production changes auditable

The Great “k8s.gcr.io” VDF

A long time ago (circa 2018), Googlers manually copied images to production (for the community)



The Great “k8s.gcr.io” VDF

Why the old (manual way) was problematic…!

Questionable security

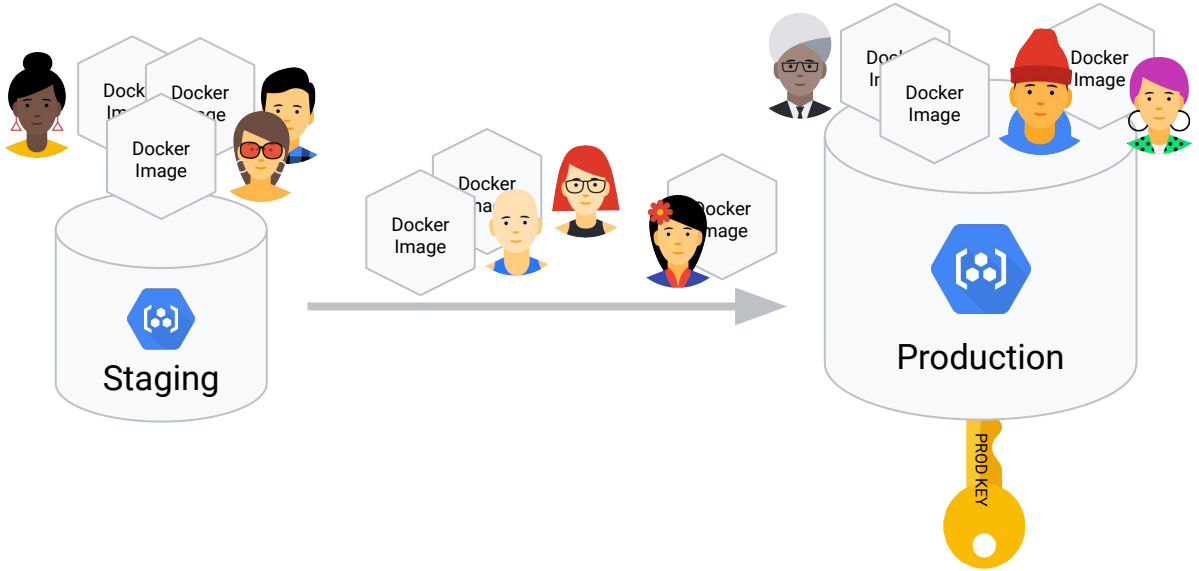
overbroad production access

No history

cannot tell who promoted which image

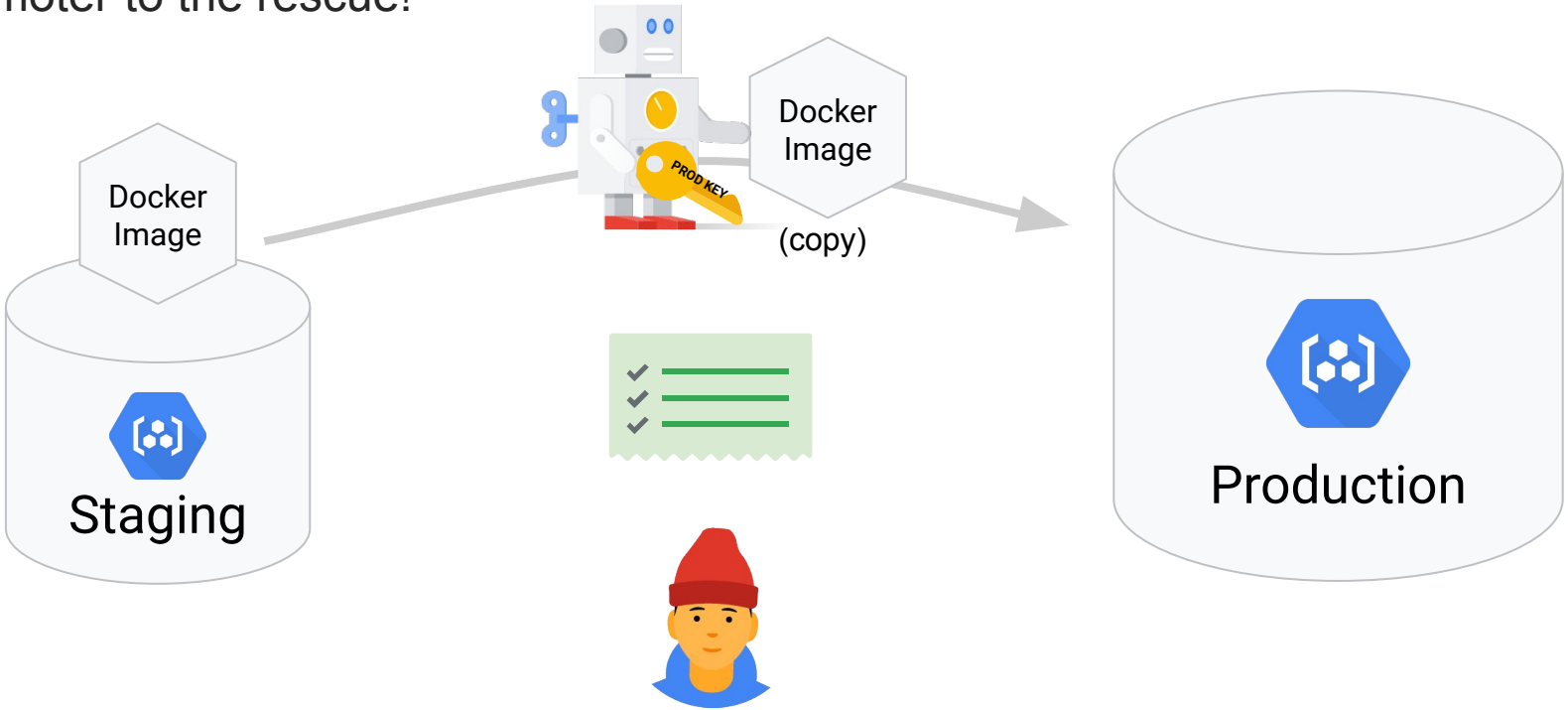
Manual

Incurs human toil for pushing images



The Great “k8s.gcr.io” VDF

Promoter to the rescue!



The Great “k8s.gcr.io” VDF

Promoter to the rescue!

More secure
only the promoter has access to prod key

Full history
promoter manifest kept in version control

Automatic
runs as a postsubmit

The Great “k8s.gcr.io” VDF

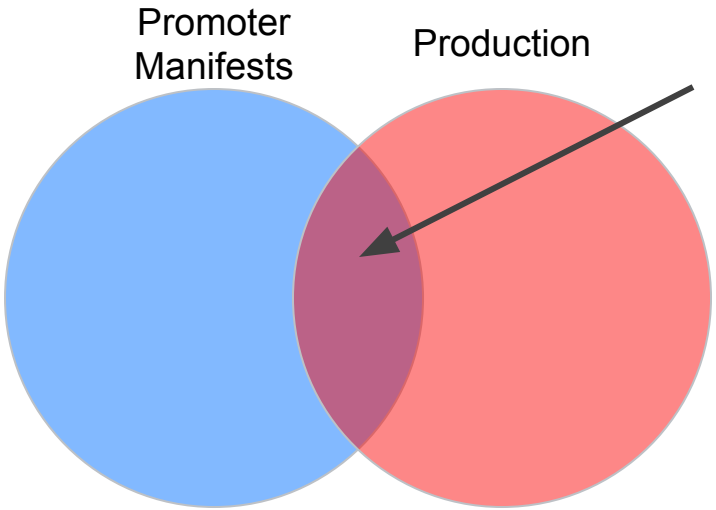
The OSS Promoter

- It’s the open-source rewrite (Golang) of the internal promoter (Python)
- Performance-oriented
  - Promoter manifests specify ~30K images to promote, or 90K if you consider the 3 target regions
  - We take ~30 seconds to read all 90K (30K * 3 regions) image metadata from GCR
- “Edge” data structure for simplicity and correctness

The Great “k8s.gcr.io” VDF

The OSS Promoter: Performance optimizations

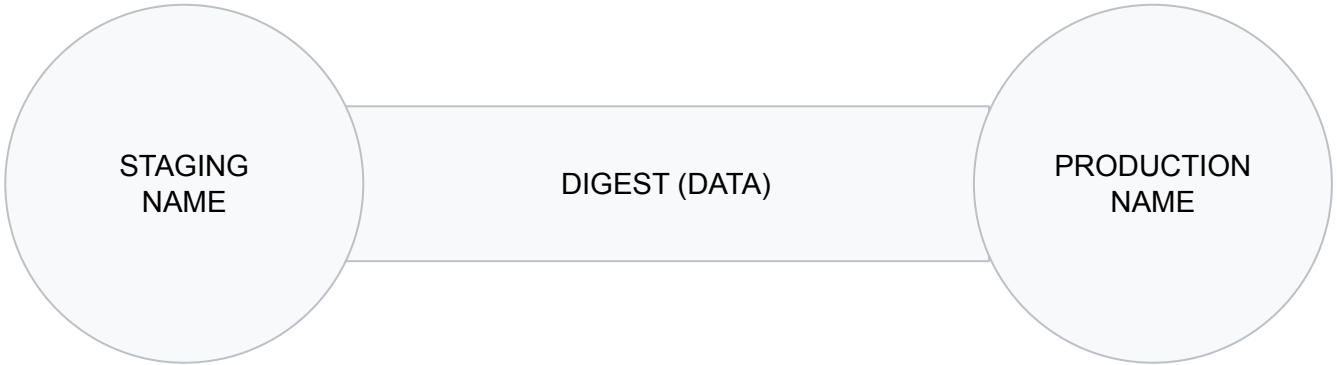
- Read images in promoter manifests
- Read images in production GCR
- Remove unnecessary promotions (purple)
- Only promote what's left (blue)



The Great “k8s.gcr.io” VDF

The OSS Promoter: “Edge” data structure

- A promotion “edge” represents the idea of a “copy”, but without the notion of time
- "Edge" has 3 parts = staging GCR "vertex", digest "edge", production GCR "vertex"



The Great “k8s.gcr.io” VDF

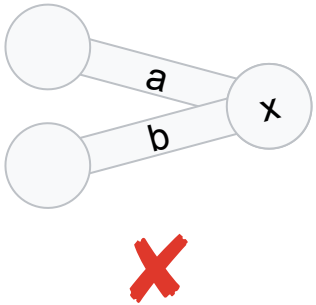
The OSS Promoter: “Edge” data structure

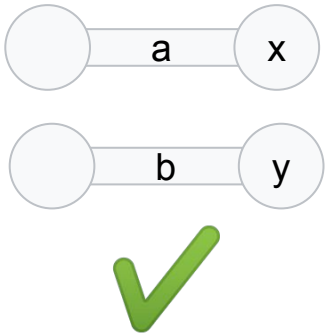
> Promote gcr.io/staging/foo@sha256:0xabc to gcr.io/production/path/to/foo:1.0

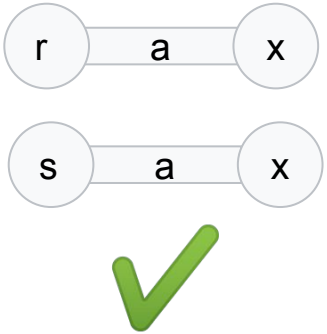
The Great “k8s.gcr.io” VDF

The OSS Promoter: “Edge” data structure

- Checking against tag overwrites, where "overwrite" means putting a different image into a production name
- However, copying the same image from multiple locations into the same production endpoint is OK (redundancy!)







The Great “k8s.gcr.io” VDF

The OSS Promoter: How it really works

1. Gather set of promoter manifests
2. Convert desired promotions as promotion edges
3. Remove edges that are illegal (tag overwrite) or unnecessary (redundant)
4. "Actuate" each promotion edge with an image copy

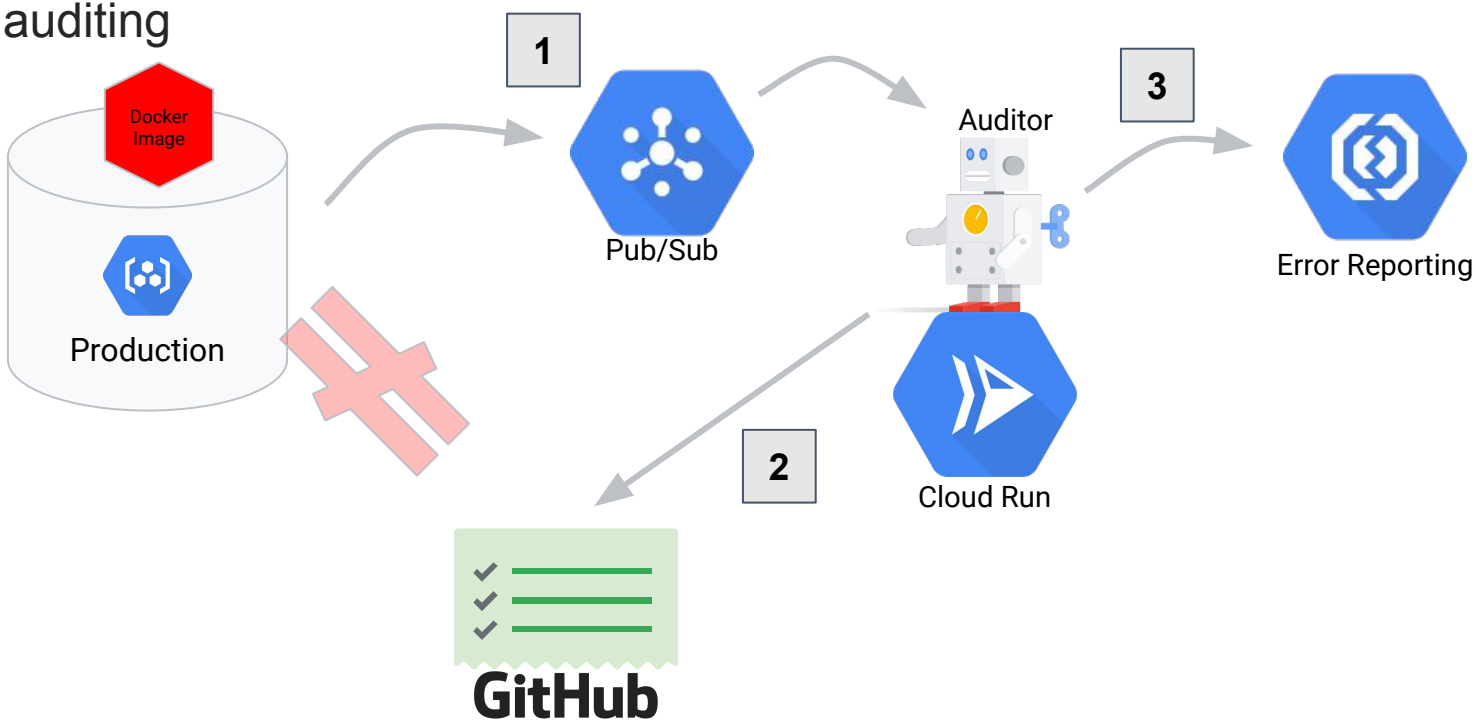
The Great “k8s.gcr.io” VDF

The OSS Promoter: Supporting Cast

- Additional infrastructure:
  - **Image auditing**
  - **Backups**

The Great “k8s.gcr.io” VDF

Image auditing



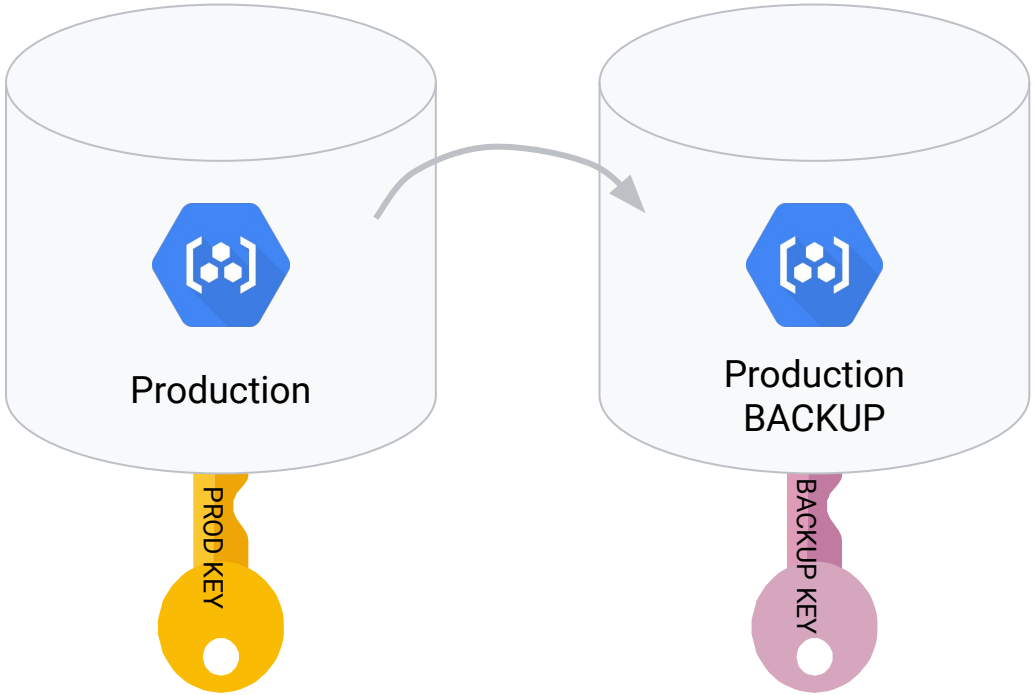
The Great “k8s.gcr.io” VDF

Backups!

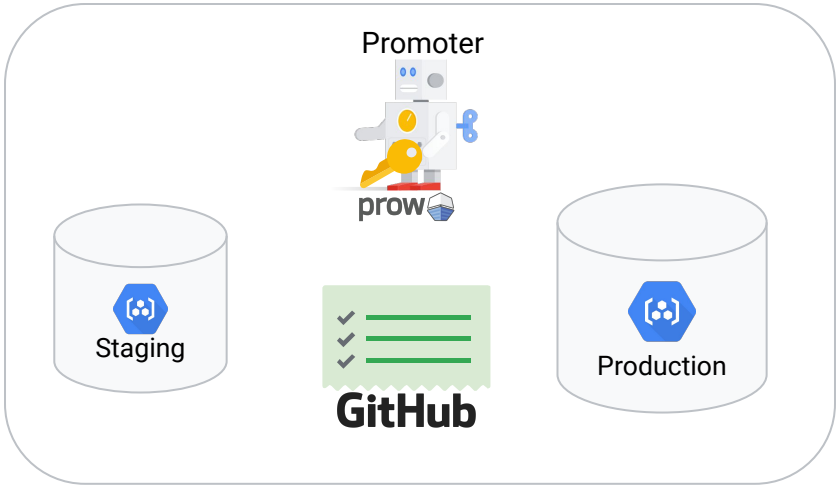
Regular
runs every 12 hours, full copy

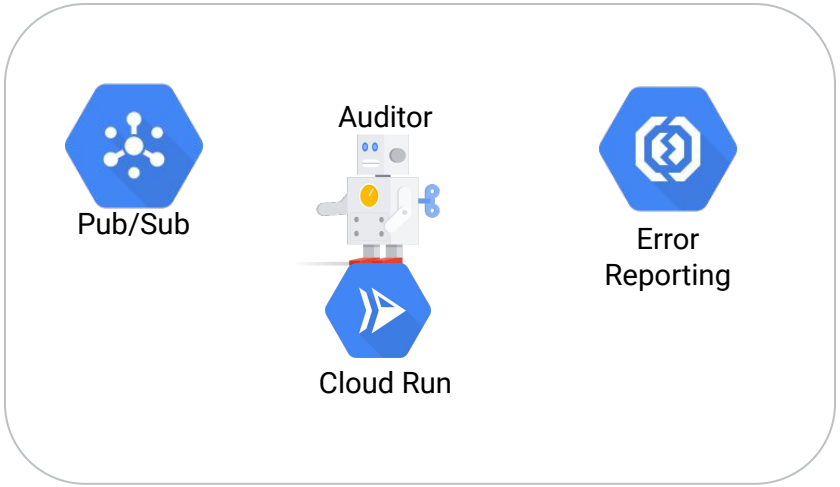
Simple
Single job

Secure
uses a different key

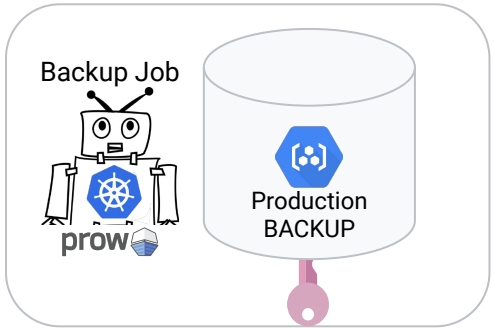


The Great “k8s.gcr.io” VDF





Infrastructure review



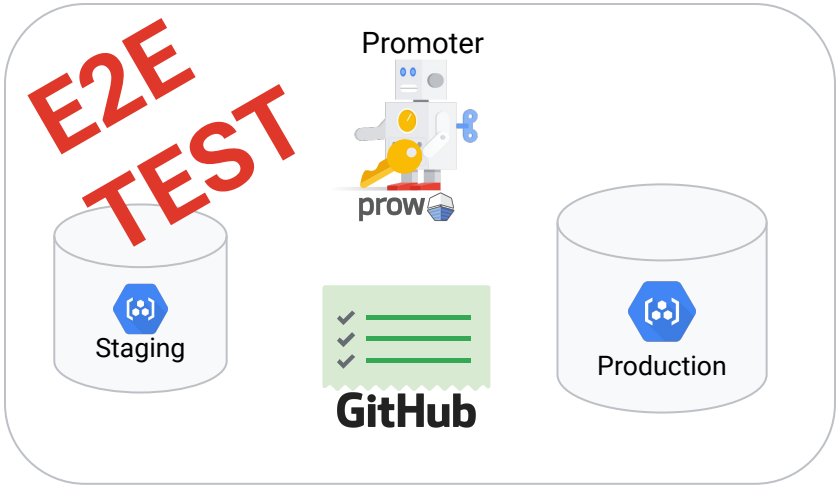
...but what about tests?

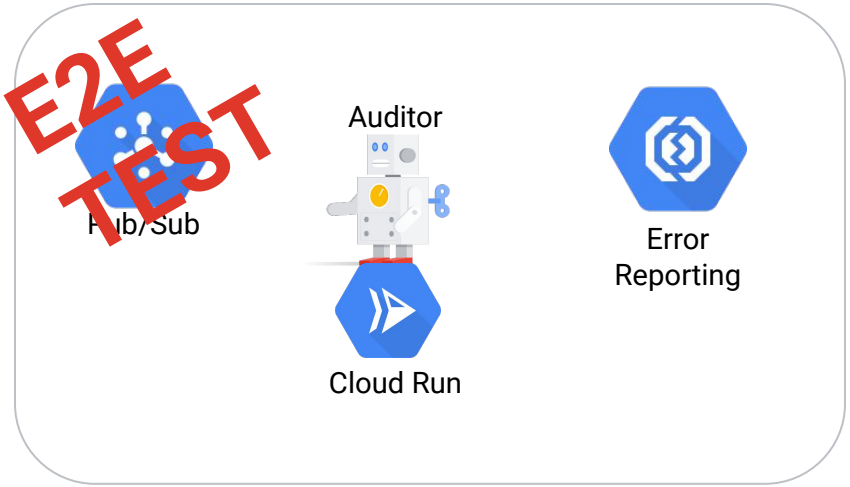
The Great “k8s.gcr.io” VDF

The OSS Promoter: Tests

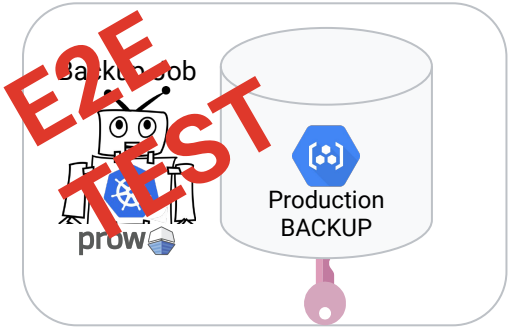
- Standard unit tests (no extra sauce, just the standard “**testing**” package)
- Custom E2E test framework
  - Fully replicated promotion stack against real GCR endpoints
  - Fully replicated auditing stack with Pub/Sub, Cloud Run, Error Reporting
  - Fully replicated backup stack

The Great “k8s.gcr.io” VDF





E2E tests have fully replicated environments!



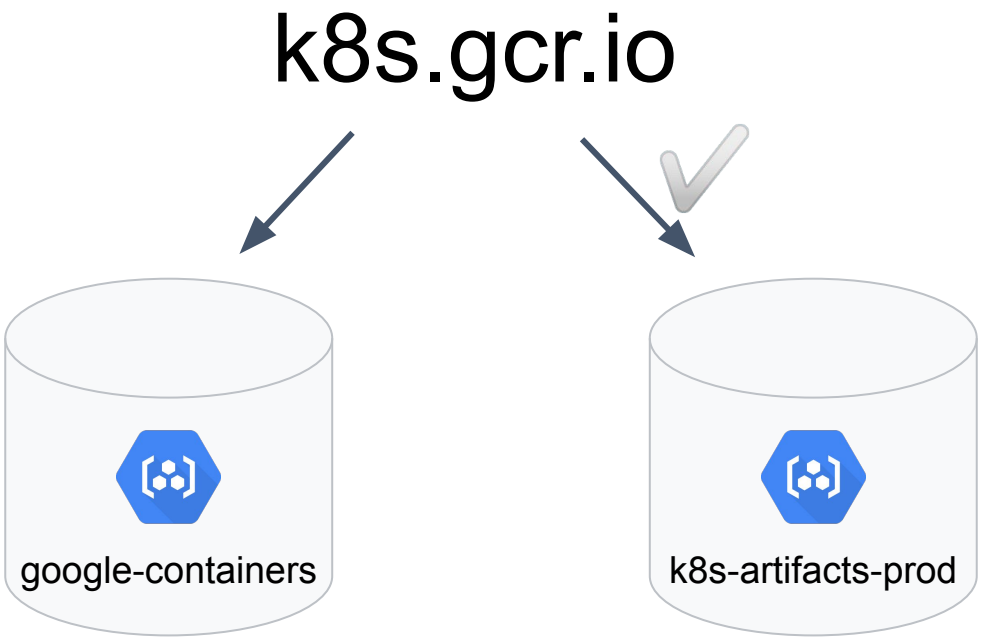
The Great “k8s.gcr.io” VDF

History of flip attempts

- 1st attempt: April 1, 2020
  - Rolled back due to a Google configuration issue
- 2nd attempt: June 22, 2020
  - Rolled back due to billing error
- 3rd attempt: July 24, 2020
  - ...it worked!

The Great “k8s.gcr.io” VDF

Summary of the vanity domain flip



The Great “k8s.gcr.io” VDF

Is that it? Nope!

- Develop tooling for creating staging projects
- Enable image pushing to staging projects via GCB and GitHub postsubmit
- Lots of bash script cleanup
- Migrating portions of the Kubernetes release process to Community infra

The Great “k8s.gcr.io” VDF

What’s Next?

- Create a tool that does both image and file promotion
- Deduplicate common Release Engineering libraries
- Support Google Cloud **Artifact Registry** (the next generation of Container Registry)
- Grow the set of promotion tool maintainers (help us!)
- Improve vulnerability scanning for images

The Great “k8s.gcr.io” VDF

Lessons learned

- Infrastructural changes (esp. changing legacy code) takes time, but the rewards are worth it
- “If it is not tested, it is broken” -- Tim Hockin
- It takes a village

The Great “k8s.gcr.io” VDF

Getting involved!

The container image promoter (and other artifact promotion tools) are maintained by SIG Release’s Release Engineering subproject and WG K8s Infra.

- Promotion tooling:
  - https://sigs.k8s.io/k8s-container-image-promoter
  - https://git.k8s.io/release
- SIG Release: https://git.k8s.io/community/sig-release
- WG K8s Infra: https://git.k8s.io/community/wg-k8s-infra
- SIG Release repo: https://git.k8s.io/sig-release
- Promoter manifests location: https://git.k8s.io/k8s.io


KEEP CLOUD NATIVE
EVERYWHERE
KubeCon
CloudNativeCon
North America 2020
Virtual
HELM
KV