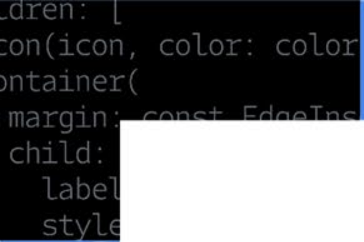


```
dren: [
con(icon, color: color
ontainer(
 margin: const EdgeIns
 child:
   label
   style
```

Our Never-Ending Journey of GitOps Transformation with Flux CD

Or, democratizing infrastructure

Kwong Tung Nan
DevOps Engineer, TalkHub
tungnan5636@gmail.com

About Me

Kwong Tung Nan | 邝东南

DevOps Engineer @ TalkHub

- Graduated Year 2022 @ Universiti Teknikal Malaysia Melaka (UTeM)
  BaCompSc. in Artificial Intelligence
- Frequent FOSS Contributor
- Mainly develops in Django & Angular
- Codes the Malaysia Land Public Transport Fans (MLPTF) community site @ community.mlptf.org.my

https://github.com/kwongtn

THE LINUX FOUNDATION CERTIFIED IT ASSOCIATE

kubernetes AND CLOUD NATIVE ASSOCIATE

Disclaimer

- I **do not** represent / work at Weaveworks / ControlPlane, nor am a Flux maintainer. I contributed a few lines of docs, but that’s all 😉
- All views represented here are of **mine** and don’t represent any other entities.
- I’m **still a novice**, stuff presented are accurate to the best of my knowledge. If there are any discrepancies, I stand corrected 🥺

Kubernetes in a Nutshell

The typical way of thinking things

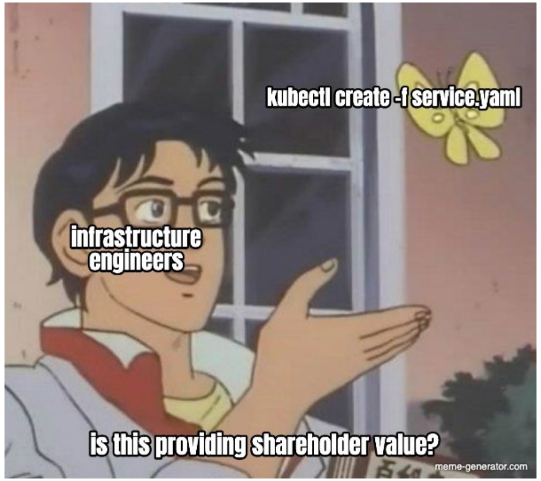
“I don’t care, as long as”

1. **I don’t care** where the storage is, **as long as** I have 20GB.
2. **I don’t care** how you schedule, **as long as** I have a pod that has X CPU, Y memory.
3. **I don’t care** which traffic goes where, **as long as** 10% goes to the new version.

The traditional way of doing things

“kubectl --apply-and-forget”

1. Create a yaml file, lint it (do you?)
2. Run kubectl apply
3. Pray
4. Probably chuck the yaml file somewhere, **hopefully** in version control


kubectl create -f service.yaml
infrastructure engineers
is this providing shareholder value?
meme-generator.com

As time passes, changes pile up

- You start to ask yourself (and teammates around)
- “**Did I** apply this configuration?”
- “**Where** did I apply it? **When**?”
- “Isn’t this deleted already?”
- “Does this change reflect the cluster state?”
- Or spend time `kubectl get` the cluster to validate problems above

STILL BUSY?
YEAH, I'M TRYING TO FIX THE PROBLEMS I CREATED WHEN I TRIED TO FIX THE PROBLEMS I CREATED WHEN I TRIED TO FIX THE PROBLEMS I CREATED WHEN...
Illustrator: iujrm

The Problem

- We are putting too much trust on people
- We say: “You should commit in source control after you do the change”, but
- “Haiya I’ll commit the change later, lemme fix prod first”, then later
- “Eh I did X, Y and Z….. Right?”

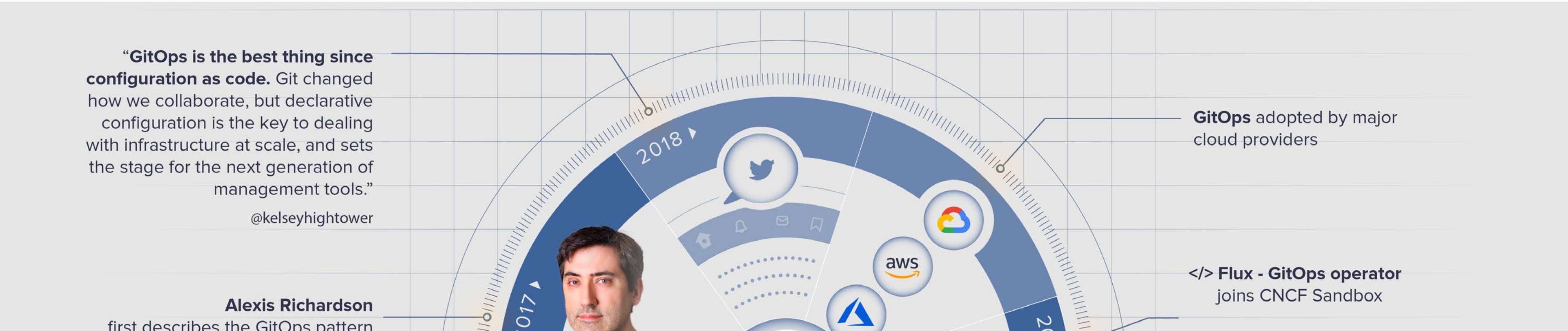
A ~~DevOps~~ yaml Engineer…

- Maintains infrastructure
- Be first line of defense when shit hits the fan
- Most importantly, edit yaml files

- Hence it is very important that our files & cluster is what we say it is

Meet Weaveworks

- Year 2015, Weaveworks launched Weave Cloud to help manage Kubernetes clusters.
- Year 2016, they destroyed their entire production cluster, but managed to restore it in 40 minutes, due to good practices.
- Year 2017, the **GitOps** term was coined.


"GitOps is the best thing since configuration as code. Git changed how we collaborate, but declarative configuration is the key to dealing with infrastructure at scale, and sets the stage for the next generation of management tools."
@kelseyhightower
2018
2017
aws
GitOps adopted by major cloud providers
</> Flux - GitOps operator joins CNCF Sandbox
Alexis Richardson
first describes the GitOps pattern

GitOps Core Principles

1. **Declarative**

   The entire system has to be described declaratively.

2. **Versioned and immutable**

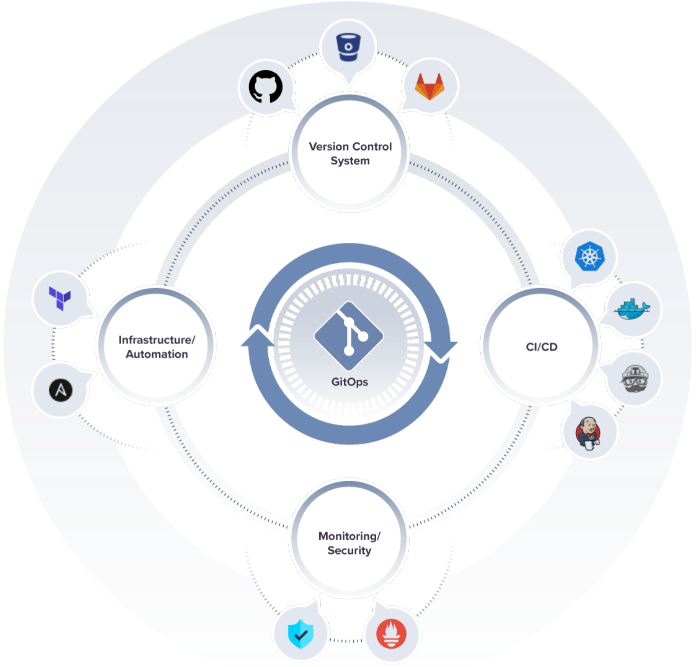
   The canonical desired system state is versioned in Git.

3. **Pulled automatically**

   Approved changes are automatically applied to the system.

4. **Continuously reconciled**

   Software agents to ensure correctness and alert on divergence.

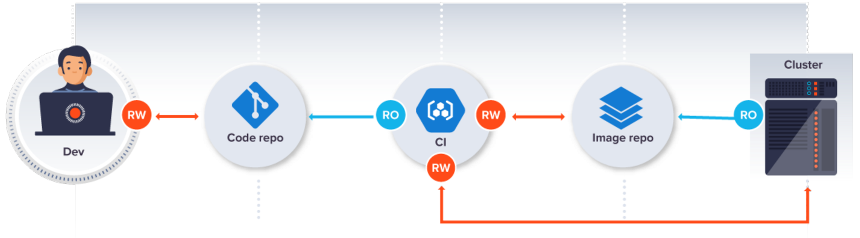


Meet FluxCD

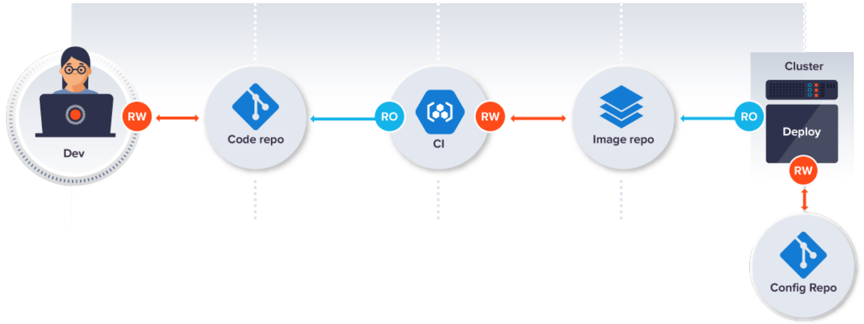
- Flux is a set of continuous and progressive delivery solutions for Kubernetes that are open and extensible.
- Runs using the GitOps Toolkit (gotk)
- A pull-based pipeline
- It is a CNCF Graduated project.
- Released by Weaveworks in 2019, now managed by ControlPlane
- Now on v2

Push vs Pull Pipelines

- Push-based pipelines **potentially expose credentials** outside of cluster.
- Hence it is a **commonly-known attack vector** for production.
- Pull-based pipelines on the other hand **works within the trust-domain of the cluster.**

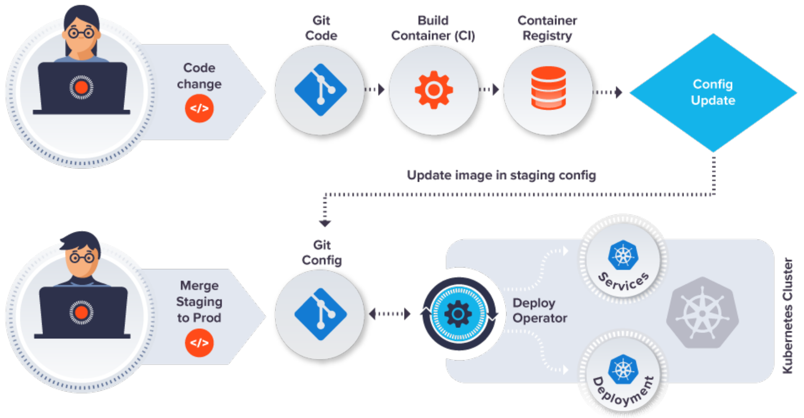


A typical push-based pipeline



A typical pull-based pipeline

The Weave GitOps Pull Pipeline



- The Deployment Automator watches the image registry for changes.
- The Deployment Synchronizer watches the cluster to maintain its state.

The GitOps Toolkit (gotk)

- <u>**Source Controller**</u>

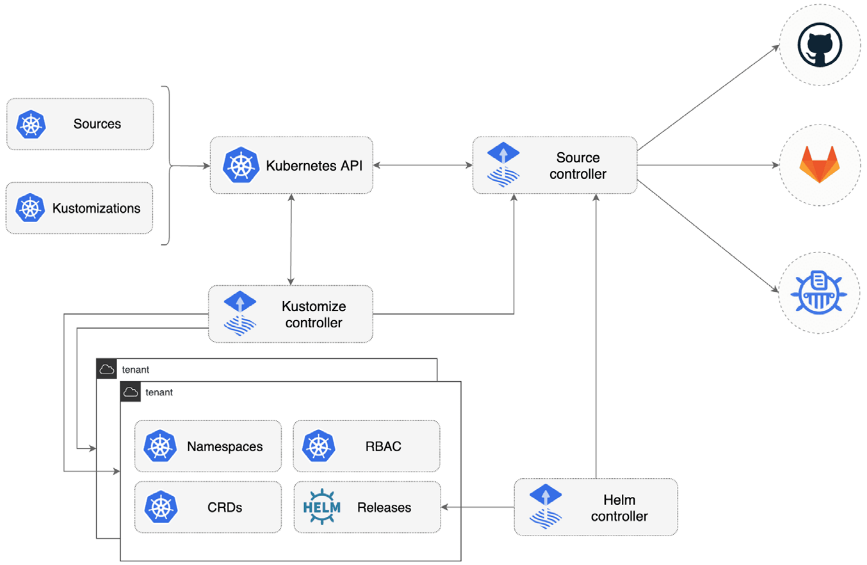
  Provide a common interface for **artifacts acquisition**.
- <u>**Kustomize Controller**</u>

  A specialized operator for running continuous delivery pipelines for infrastructure and workloads defined with Kubernetes manifests and **assembled with Kustomize**.
- <u>**Helm Controller**</u>

  Allows declaratively managing **Helm chart releases** with Kubernetes manifests.
- <u>**Notification Controller**</u>

  An operator specialized in handling **inbound and outbound events**.
- <u>**Image Automation Controller**</u>

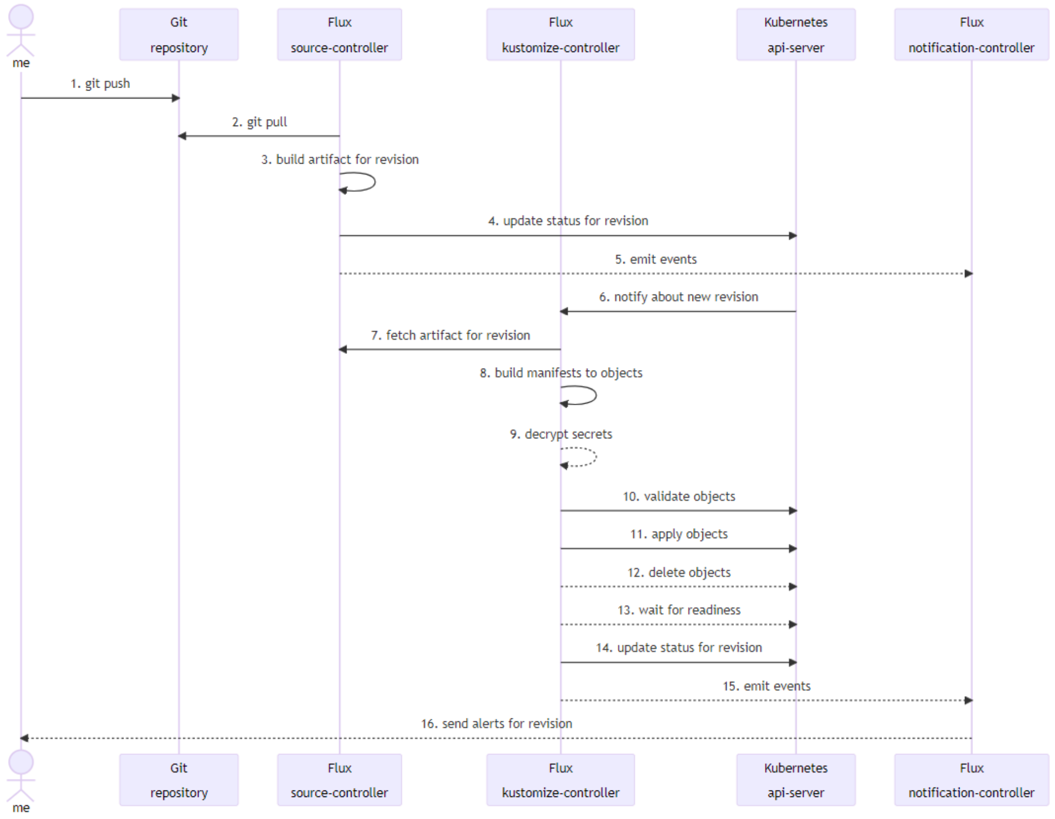
  To **update a Git repository** when new container images are available.



How a commit cycle works

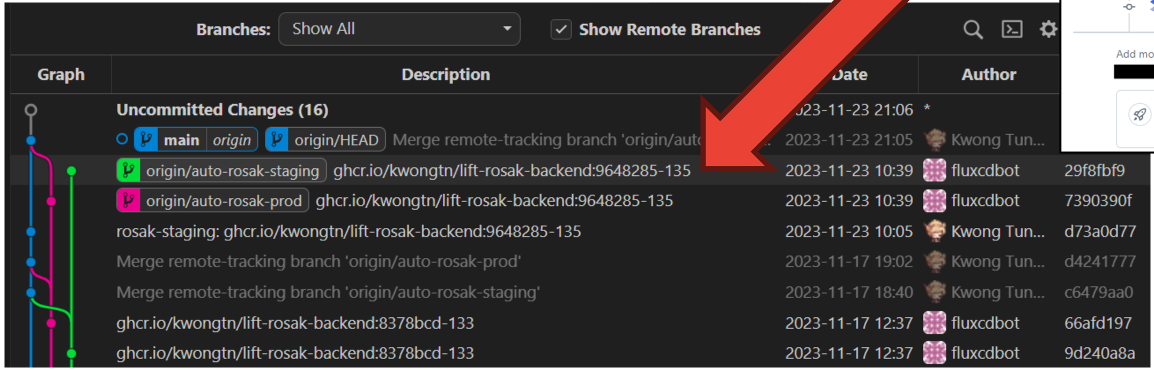
We assume that the cluster has been bootstrapped with flux.

- Take note that git is always the source of truth.
- A similar procedure also will be done on reconciliation.

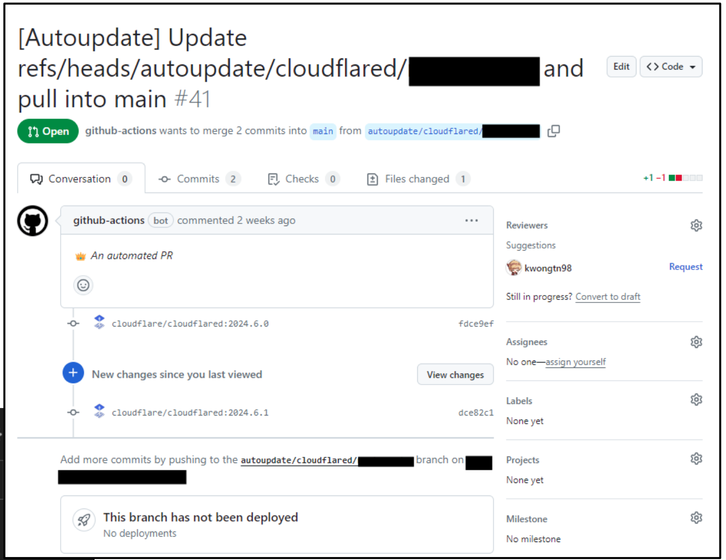

me
Git repository
Flux source-controller
Flux kustomize-controller
Kubernetes api-server
Flux notification-controller
1. git push
2. git pull
3. build artifact for revision
4. update status for revision
5. emit events
6. notify about new revision
7. fetch artifact for revision
8. build manifests to objects
9. decrypt secrets
10. validate objects
11. apply objects
12. delete objects
13. wait for readiness
14. update status for revision
15. emit events
16. send alerts for revision

Extras – Image Update Automation

- FluxCD automatically creates a branch with latest detected changes, ready for you to accept & incorporate into your cluster.



Graph	Description	Date	Author	
	Uncommitted Changes (16)	023-11-23 21:06	*	
	main origin origin/HEAD Merge remote-tracking branch 'origin/aut ...	2023-11-23 21:05	Kwong Tun...	
	origin/auto-rosak-staging ghcr.io/kwongtn/lift-rosak-backend:9648285-135	2023-11-23 10:39	fluxcdbot	29f8fbf9
	origin/auto-rosak-prod ghcr.io/kwongtn/lift-rosak-backend:9648285-135	2023-11-23 10:39	fluxcdbot	7390390f
	rosak-staging: ghcr.io/kwongtn/lift-rosak-backend:9648285-135	2023-11-23 10:05	Kwong Tun...	d73a0d77
	Merge remote-tracking branch 'origin/auto-rosak-prod'	2023-11-17 19:02	Kwong Tun...	d4241777
	Merge remote-tracking branch 'origin/auto-rosak-staging'	2023-11-17 18:40	Kwong Tun...	c6479aa0
	ghcr.io/kwongtn/lift-rosak-backend:8378bcd-133	2023-11-17 12:37	fluxcdbot	66afd197
	ghcr.io/kwongtn/lift-rosak-backend:8378bcd-133	2023-11-17 12:37	fluxcdbot	9d240a8a



[Autoupdate] Update refs/heads/autoupdate/cloudflared/[redacted] and pull into main #41

Open github-actions wants to merge 2 commits into `main` from `autoupdate/cloudflared/`[redacted]

github-actions bot commented 2 weeks ago

An automated PR

`cloudflare/cloudflared:2024.6.0` fdce9ef

New changes since you last viewed — View changes

`cloudflare/cloudflared:2024.6.1` dce82c1

Add more commits by pushing to the **autoupdate/cloudflared/**[redacted] branch on [redacted] [redacted]

This branch has not been deployed
No deployments

Reviewers
Suggestions
kwongtn98 Request
Still in progress? Convert to draft

Assignees
No one—assign yourself

Labels
None yet

Projects
None yet

Milestone
No milestone

In our Prod...

```
lookup.KeyValue
f.constant(['em
=tf.constant([0
.lookup.StaticV

_buckets=5)
```

Our Scenario

- Our product, TALK)(UBs a CRM that serves customers mainly in SEA.
- For usual deployments, a SemaphoreCI pipeline is invoked & helm charts are applied.
- Most utility applications (redis, elasticsearch etc.) are already deployed using helm charts.
- We have 3 k8s clusters, production, staging & monitoring.
- A near 24/7 availability is required.

How did we migrate?

1. Add a new cluster, “test”. Everything is tested there first.
2. Then in staging, the Flux Controller & GitOps Toolkit were installed.
3. The SealedSecrets controller was added first, so we can start committing secrets into the GitOps repository.
4. Anything already managed by Helm were taken over by FluxCD.
5. Declarations were also added according to namespaces. Worth noting that flux ignores resources that are not declared / labeled.

Metrics?

>100x speed improvements on cluster recovery

- Since the introduction of GitOps**, cluster recovery reduced** from 3 days to 40 minutes, and
- We can be **pretty sure** the cluster is at the state we want it to be.

More Metrics?

>5x infra innovation

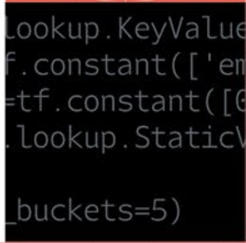
From 2-3 changes in a month, to 3-5 changes in a **week**

- Changes can now be queued and iteratively applied. We **do not need to wait** forever for maintenance window to do changes.
- We can now **“democratize” cluster management by involving juniors** without being scared that our cluster will randomly explode (or make untraceable changes).

Dev…Oops!

How I brought down production, twice.

And lessons learnt.

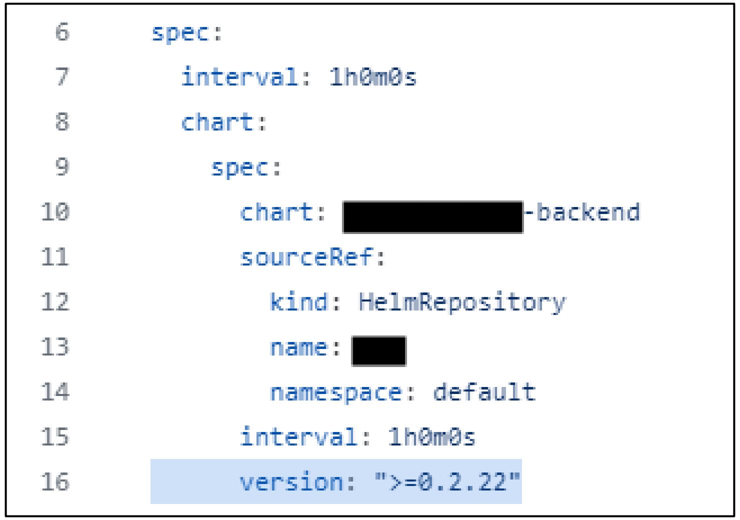


```
lookup.KeyValue
f.constant(['en
=tf.constant([0
.lookup.StaticV

_buckets=5)
```

Lesson #1: Always lock versions for prod

- During migration, configuration was **copied directly** from staging to prod. We missed the version line 🤦
- Later, when updates to the helm charts are done and merged (for testing), **prod automatically updated itself** too 😔
- Outage lasted for about 15 minutes. Boss was definitely **NOT** happy 🥹



```
spec:
  interval: 1h0m0s
  chart:
    spec:
      chart: [redacted]-backend
      sourceRef:
        kind: HelmRepository
        name: [redacted]
        namespace: default
      interval: 1h0m0s
      version: ">=0.2.22"
```

Lesson #2: Use Ctrl + F extensively

- Humans make errors.
- I accidentally modified prod image targets instead of staging.
- Luckily with GitOps, we just roll back the commit. But routing changes were already partially done.
- This outage lasted about 5 minutes.
- Senior was extremely **NOT happy** 🥺

WHEN YOU TEST DIRECTLY ON PRODUCTION
EMERGENCY EXIT
(chuckles) I'm in danger.
imgflip.com

More Lessons Learnt

- We should have **started way, way earlier** – the earlier you incorporate GitOps, the better.
- When convincing your boss, try **getting into the “So What” conclusion** as soon as possible, to let them have a higher-level overview.
- Dig into existing config **as much as possible** to reduce discrepancies between declared & actual. (E.g. Secret targets like envFrom, or manually passed ones)
- Don’t worry if it takes 20 deployment cycles to fully roll it out – **be right on the first try**.
- Install a GUI dashboard / tool **right from the start** to help sort out deployment errors during migration.

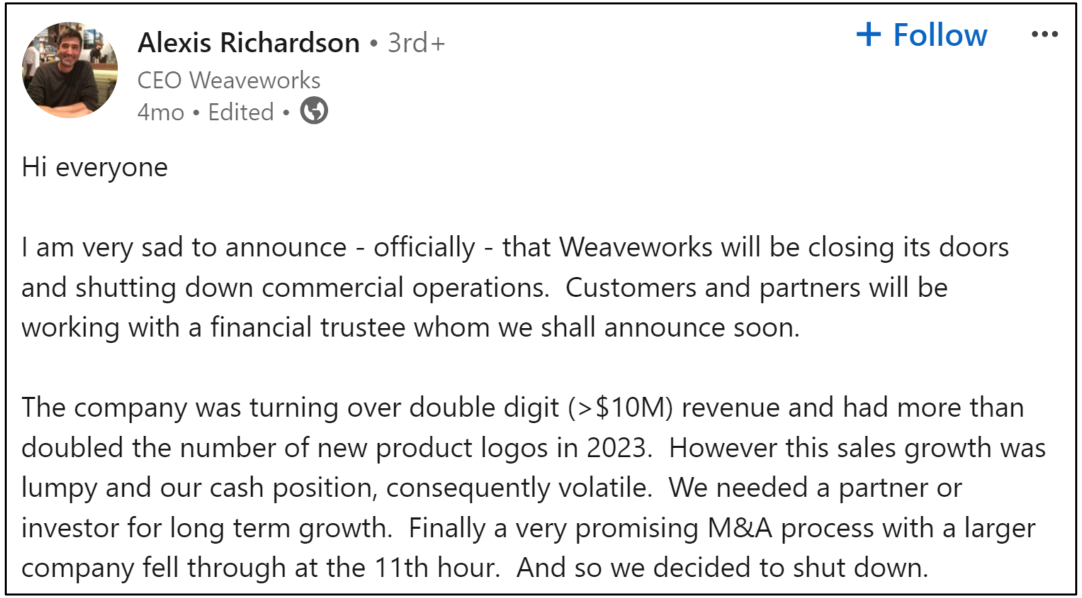
Also, things to consider

```
lookup.KeyValue
f.constant(['en
=tf.constant([0
.lookup.StaticV

_buckets=5)
```

Things to Consider

- Weaveworks went out of funds late 2023 and shut down subsequently.



Alexis Richardson • 3rd+
CEO Weaveworks
4mo • Edited •

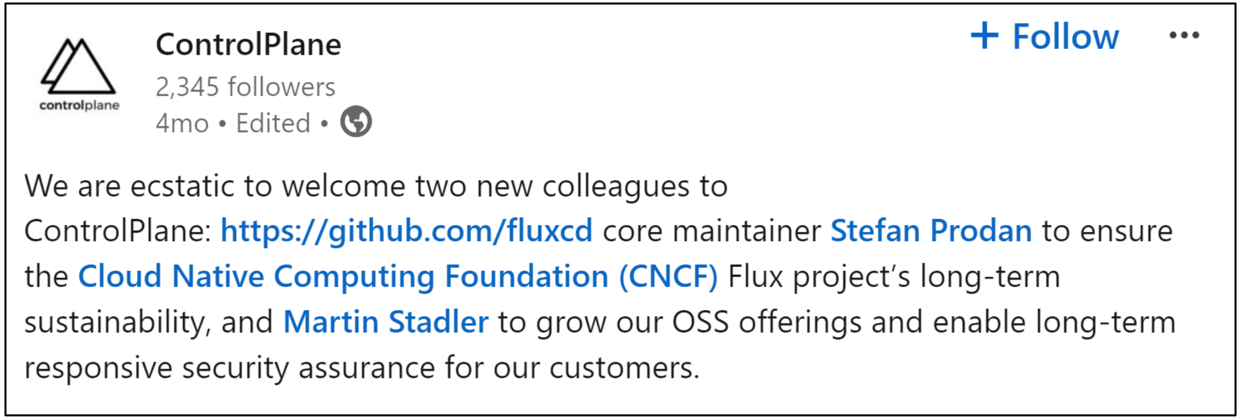
Hi everyone

I am very sad to announce - officially - that Weaveworks will be closing its doors and shutting down commercial operations. Customers and partners will be working with a financial trustee whom we shall announce soon.

The company was turning over double digit (>$10M) revenue and had more than doubled the number of new product logos in 2023. However this sales growth was lumpy and our cash position, consequently volatile. We needed a partner or investor for long term growth. Finally a very promising M&A process with a larger company fell through at the 11th hour. And so we decided to shut down.

Things to Consider

- However, ControlPlane took over the project (employing Flux maintainers in the process) to continue its development.
- Currently flux is growing at a higher velocity than before!


ControlPlane
2,345 followers
4mo • Edited •

+ Follow

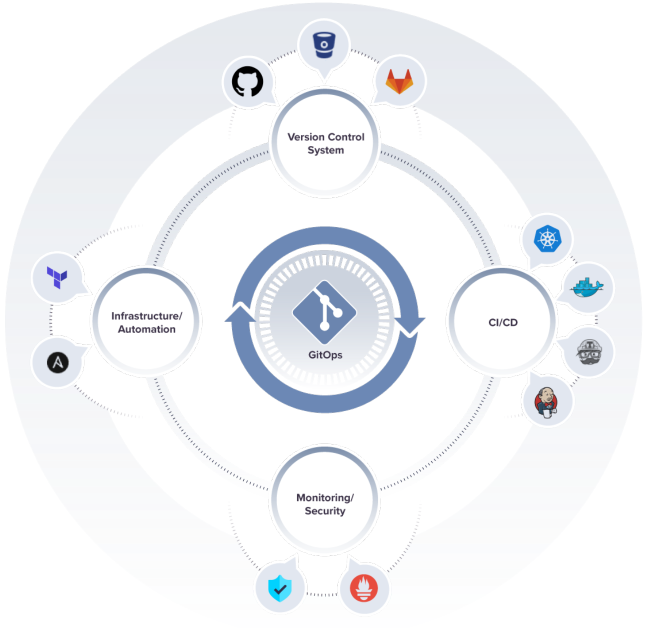
We are ecstatic to welcome two new colleagues to ControlPlane: https://github.com/fluxcd core maintainer Stefan Prodan to ensure the Cloud Native Computing Foundation (CNCF) Flux project's long-term sustainability, and Martin Stadler to grow our OSS offerings and enable long-term responsive security assurance for our customers.

Why FluxCD and not XYZ (Argo)?

- We are using flagger from the start already. Hence I figured that since both are from FluxCD project they should work well together.
- FluxCD is wayy more barebones than ArgoCD, simpler = better (?)
- Hence, easier dev onboarding from the yaml side.

In a nutshell, why GitOps?

1. Increased Productivity, Auditability, Reliability
2. Enhanced Developer Experience / Democratization of development
3. Stronger security guardrails
4. Faster development, better ops
5. Easier Compliance & Auditing

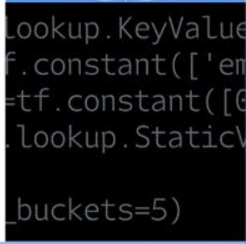


Thank you

Connect with me!

https://github.com/kwongtn

https://www.linkedin.com/in/kwongtn/



```
lookup.KeyValue
f.constant(['en
=tf.constant([
.lookup.StaticV

_buckets=5)
```