Stefan Freilinger, Pritz Sebastian

Projektdefinition:

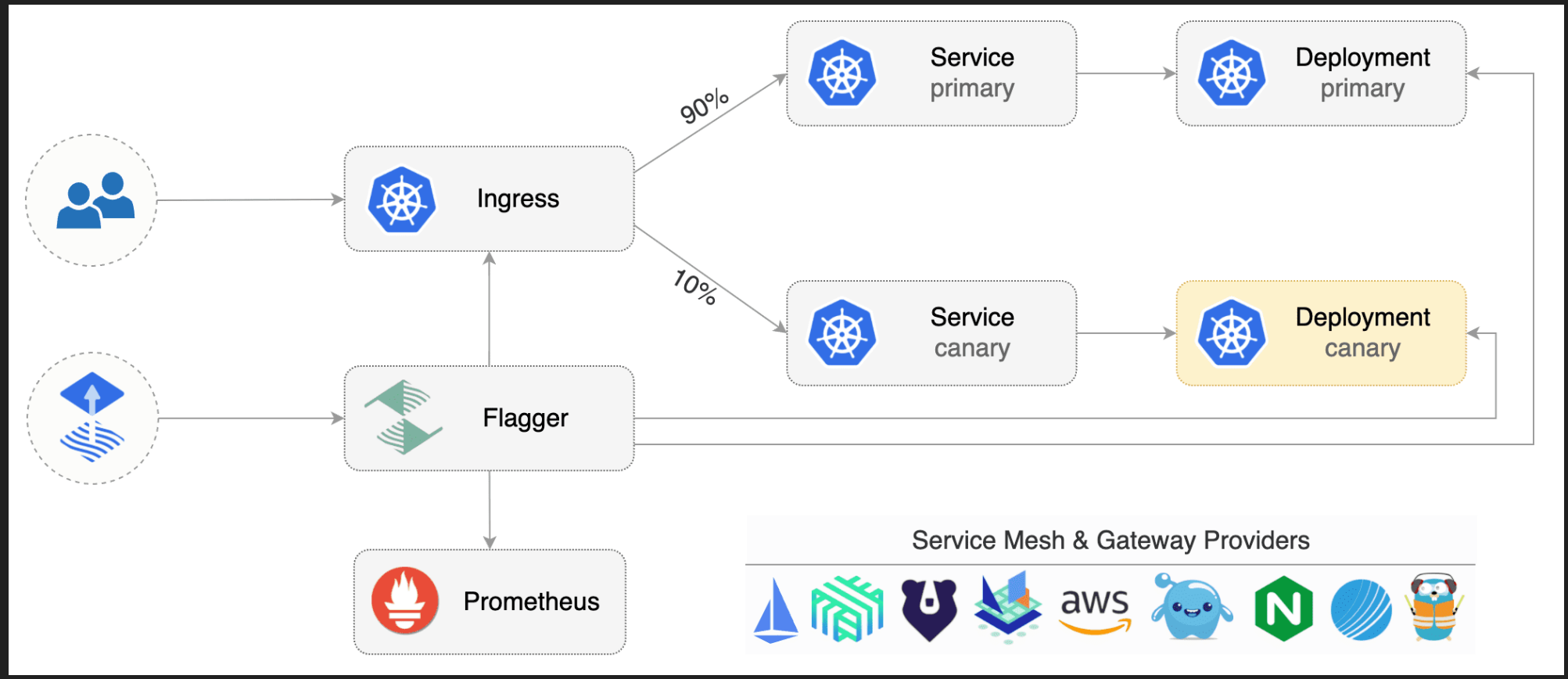
- Canary Releases mit Flagger
- Dabei folgende Testsituation schaffen:
  - Einen sehr simplen Webservice deployen (V1)
  - Neue Version releasen (V2)
  - Flagger sollte automatisch den Canary Release automatisch vornehmen
  - V2 ist aber absichtlich fehlerhaft – ein automatischer Rollback soll erfolgen!
  - Alerts einrichten

Was ist Flagger?

- Tool, das den Release neuer Version unterstützt
- Automatisiert den Prozess
- Verwaltet dabei den Zugriff -> lässt nur einen Teil auf die neue Version zugreifen
- Kann dabei auch Metriken benutzen um einen Rollback durchführen
- Kann dabei folgende Strategien anwenden:
  - A/B testing
  - Blue/Green mirroring
  - Canary Releases

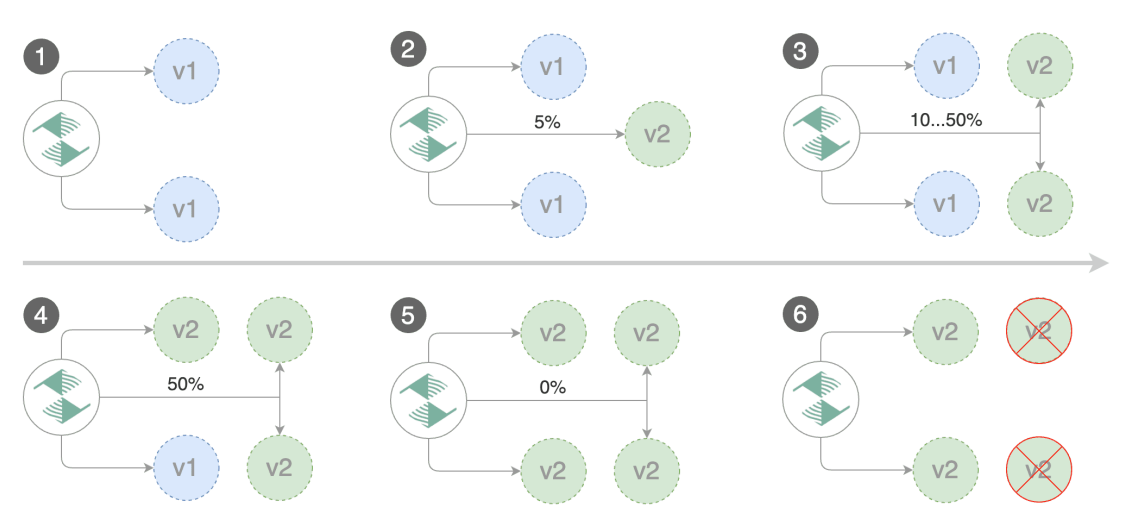
Was ist Flagger?

Quelle: https://docs.flagger.app/



Canary Release

- Schrittweiser Release einer neuen Version
- Startet mit kleinen Teilmenge
- Wird inkrementell erhöht -> kein direkter Switch!
- Sollte keine Probleme auftreten wird mit der Zeit ein voller Release durchgeführt!



Quelle:
https://docs.flagger.app/tutorials/nginx-progressive-delivery

Tools: Package Management (Helm)

- Helm = Package Manager
- "Helm is the best way to find, share, and use software built for Kubernetes." (https://helm.sh/)
- Arbeitet mit Charts – a package that contains all resources for one service

Tools: IngressController (NGINX)

- Ingress Controller managen den Zugriff auf die jeweiligen Services
- Dient als als point-of-entry
- Basierend auf Regeln die den Zugriff auf die Services bestimmen

Tools: Analysis (Prometheus)

- Monitoring Tool
- Erstellt/trackt Metriken
- Abfragen sind Query Based

Live Demo

Quelle: https://suretyit.com.au/blog/what-is-cloud-computing-and-how-does-it-support-business-objectives/

Lessons-Learned (1): Namespaces

- Aufpassen wegen Default-Namespace!
- Wenn man mit mehr als einem Namespace arbeitet, nicht den Überlick verlieren wo was in welches Ressource läuft – immer gut Ressourcen aufräumen
- --> Ingress in zwei Namespaces definiert, immer falschen erwischt
- Namespace Management: Wann ist eine Trennung notwendig?

Lessons Learned (2): Erst alles durchlesen

- Wir dachten ein DNS-Entry sei für den Ingress notwendig
- "Backend" Ressource wird eingerichtet (ist im Endeffekt nur Service)
- --> Verweist auf den Service, der erst später durch die Canary Releases angelegt wird
- --> Es ist nicht notwendig einen eigenen Service/Cluster-IP anzulegen!

Lessons-Learned (3): Misc

- Deployments teilweise buggy – Pods werden nicht eingerichtet
  - Delete Deployment + neues Apply = Pods auf einmal da, trotz gleicher Konfiguration
- Unterscheidung: Primary vs Canary
  - Automatisch erstellte Services und Deployments

Research-Guide: NGINX Canary Deployments

Folgender Guide wurde aus der Flagger Seite entnommen:

- https://docs.flagger.app/install/flagger-install-on-kubernetes#install-flagger-with-helm
- https://docs.flagger.app/usage/alerting
- https://docs.flagger.app/tutorials/nginx-progressive-delivery
- https://support.discord.com/hc/en-us/articles/228383668-Intro-to-Webhooks

Danke für eure Aufmerksamkeit,

und nicht vergessen:

Stop `giving up after 1 attempt(s)`