Red Hat
Technical Account
Management

The TAM
your tech hour

GitOps für OpenShift Administratoren

Simon Krenger
Principal Technical Account Manager , Red Hat

The TAM your tech hour

Was wir heute besprechen

Einleitung zu typischen GitOps-Installationen

Deklarative GitOps-Konfiguration

OpenShift mit GitOps verwalten

Best Practices für Argo CD

Red Hat Technical Account Management

The TAM
your tech hour

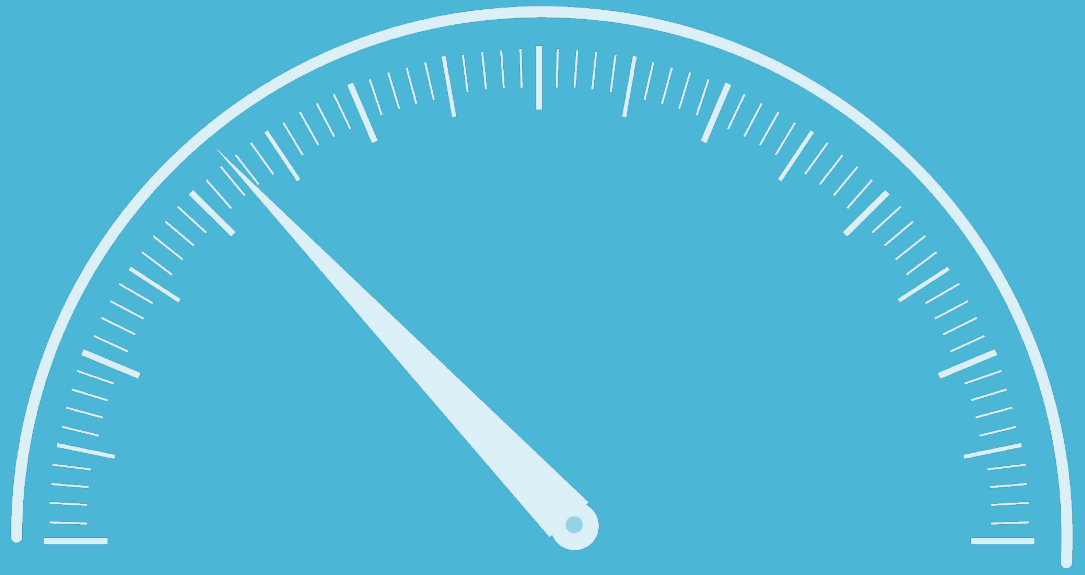
Simon Krenger

Principal Technical Account Manager
OpenShift Container Platform

Aus Bern, Schweiz

Red Hat
Technical Account
Management

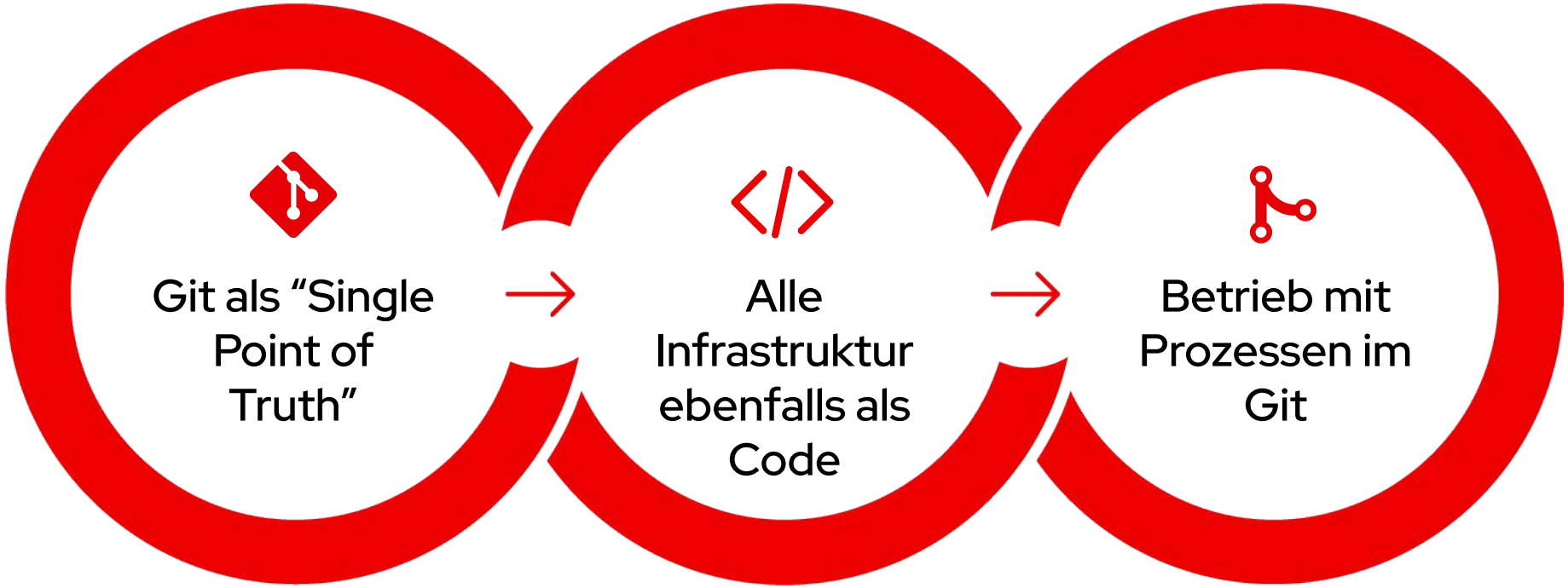
Typische GitOps-Installationen



The TAM your tech hour

Wieso wollen wir GitOps nutzen?

GitOps ist ein Entwickler-zentrierter Ansatz zu Continuous Delivery und für den Betrieb



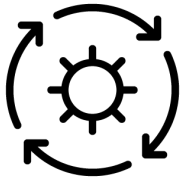
Red Hat Technical Account Management

The TAM your tech hour

GitOps Prinzipien

Das System wird deklarativ als Code beschrieben

Der gewünschte Zustand ist in Git versioniert



Bewilligte Veränderungen werden automatisch appliziert

Eine Instanz erkennt und korrigiert Abweichungen

The TAM your tech hour

Typische Fragen von TAM-Kunden

... und was wir heute anschauen

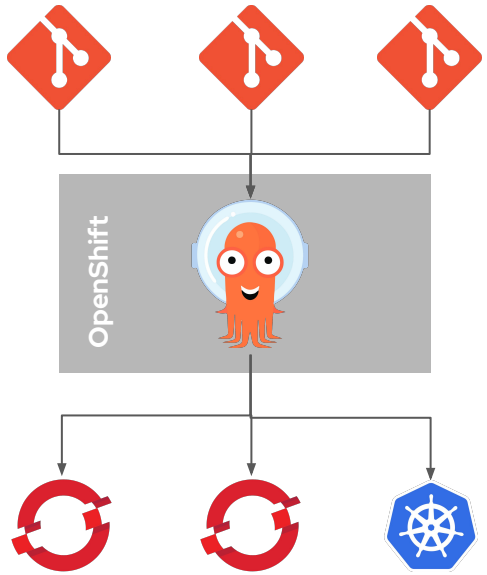
Wie kann ich als OpenShift-Administrator GitOps nutzen?

Was machen andere Red Hat Kunden?

Was sind typische Konfigurationen?

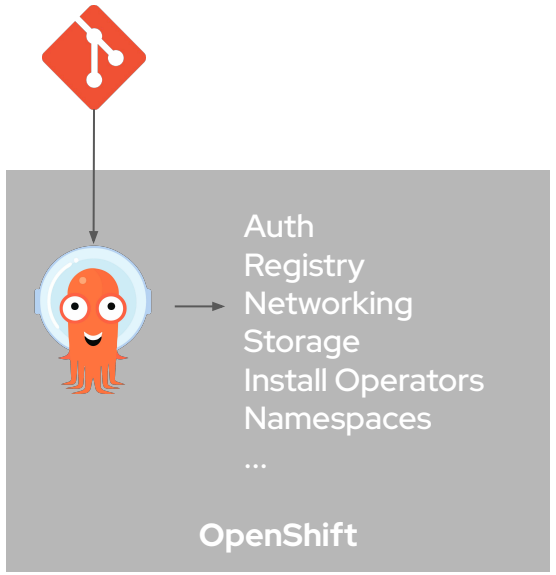
The TAM your tech hour

GitOps Architekturen



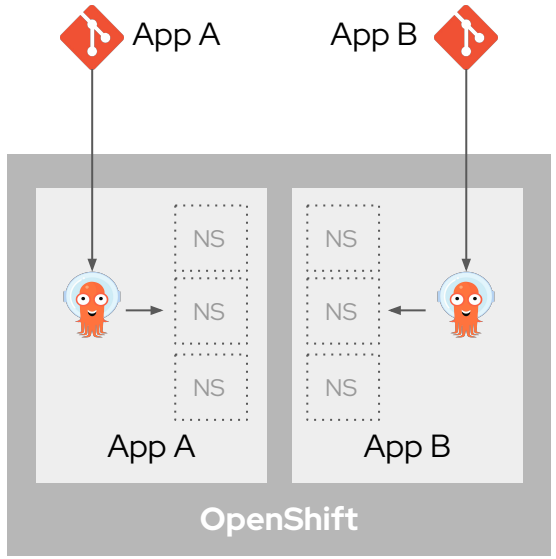
Central Hub (Push)

Eine zentrale Argo CD Instanz verwaltet mehrere Cluster



Cluster Scoped (Pull)

Pro Cluster gibt es eine Argo CD Instanz, welche die lokale Konfiguration übernimmt



Application Scoped (Pull)

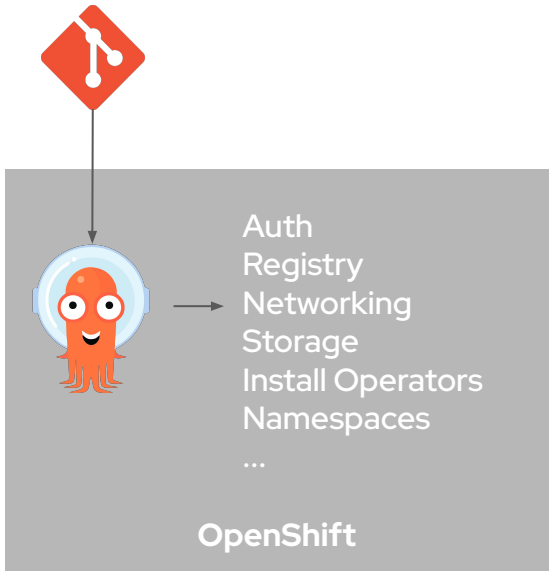
Pro Applikation oder pro Team gibt es eine Argo CD Instanz, welche die einzelnen Komponenten konfiguriert

Red Hat Technical Account Management

Wie nutzen Red Hat Kunden Argo CD?

Typische Konfiguration für OpenShift-Administratoren

- Pro Cluster eine Argo CD Instanz für die Administration
- Typischerweise wird die Standard-GitOps-Instanz genutzt, die Berechtigungen sind bereits konfiguriert
- Verschiedene `Applications` konfigurieren verschiedene Teile der Cluster-Konfiguration
- Administratoren nutzen die Instanz auch für Infrastruktur-Applikationen

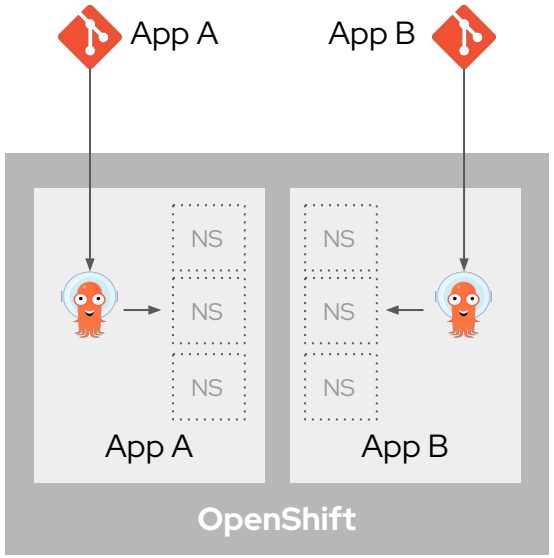


The TAM your tech hour

Wie nutzen Red Hat Kunden Argo CD?

Typische Konfiguration für OpenShift-Entwickler

- Für Entwicklerteams gibt es je nach Anforderungen verschiedene Möglichkeiten:
  - Eine zentrale Argo CD Instanz, welche verschiedene Namespaces für verschiedene Teams konfiguriert
  - Eine Argo CD Instanz pro Namespace oder pro Team



Red Hat Technical Account Management

The TAM your tech hour

Red Hat Advanced Cluster Management

RHACM und OpenShift GitOps

Red Hat Advanced Cluster Management for Kubernetes

ACM wurde für Multi-Cluster konzipiert

ACM erlaubt die Konfiguration von mehreren Clustern, viele Kunden verwenden deshalb einen zentralisierten ACM mit GitOps statt dem “einfachen” GitOps

Nutzt das ACM Placement API

Declarative Konfiguration von Applikationen über “predicate selection”, “taints/tolerations”, “scoring/prioritizer”, “spread” und “affinity”

Zusatzfunktionen

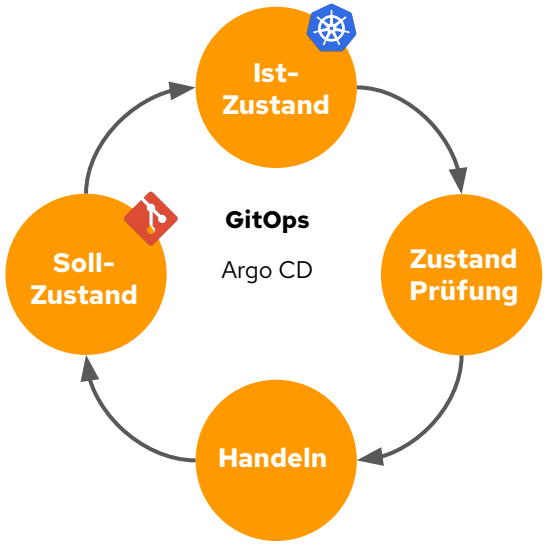
Übersicht für ApplicationSets, Unterstützung für weitere Anwendungsfälle (push / pull), Lebenszyklus-Management von Clustern

The TAM your tech hour

Was sollte ich sonst noch wissen?

GitOps Prinzipien sind nicht nur technischer Natur

- Um alle Vorteile von GitOps zu nutzen, sollten auch Prozesse im Betriebsteam angepasst werden, z.B.:
  - Pull Requests / Merge Requests mit Genehmigung von Änderungen
  - Alle Konfiguration nur in Git (keine Administrator-Accounts mit Schreibberechtigungen)
  - Konfiguration von “Auto-prune” und “Self-heal”



Red Hat Technical Account Management

The TAM
your tech hour

Deklarative GitOps-Konfiguration

The TAM
your tech hour

Deklarative GitOps-Konfiguration

oder: GitOps mit GitOps

The TAM your tech hour

Der OpenShift GitOps Operator

Standardmässige CustomResourceDefinitions für die Konfiguration von GitOps

Automatisch installierte CRDs:

- `ArgoCD`
- `Application`
- `ApplicationSet`
- `AppProject`
- `Rollout`
- …

The TAM your tech hour

Konfigurieren von Argo CD Instanzen

"ArgoCD" CustomResourceDefinition

- Konfiguriert eine Argo CD / GitOps Instanz
  - Separate Routes / RBAC / Konfiguration pro Instanz
  - Einfache Konfiguration von verschiedenen Instanzen für Teams
- Bei der Installation des OpenShift GitOps Operators wird die Cluster-weite Instanz automatisch erstellt

```
apiVersion: argoproj.io/v1beta1
kind: ArgoCD
metadata:
 name: openshift-gitops
 namespace: openshift-gitops
spec:
 server:
   autoscale:
     enabled: false
   grpc:
     ingress:
       enabled: false
   ingress:
     enabled: false
[..]
```


Quellen:
https://docs.openshift.com/gitops/1.12/argocd_instance/argo-cd-cr-component-properties.html

The TAM your tech hour

AppProjects

"AppProject" CustomResourceDefinition

- Repräsentiert ein Argo CD Projekt (nicht verwechseln mit OpenShift Projekt)
  - Wird verwendet um Source, Destination, Namespaces einzuschränken
- Wird typischerweise verwendet, wenn die Mandantenfähigkeit von Argo CD verwendet wird

```yaml
apiVersion: argoproj.io/v1alpha1
kind: AppProject
metadata:
 name: my-project
 namespace: openshift-gitops
spec:
 destinations:
   - name: '*'
     namespace: '*'
     server: '*'
 sourceRepos:
   - '*'
status: {}
```


Quellen:
https://argo-cd.readthedocs.io/en/stable/operator-manual/declarative-setup/#projects

The TAM your tech hour

Repositories

Ein “Secret” mit dem “secret-type: repository” Label

- Repräsentiert ein Quellcode-Repository
  - Wird jeweils benötigt, wenn für das Repository Zugangsdaten benötigt werden
- Kein eigenes CRD, Repositories sind “Secrets” mit einem `argocd.argoproj.io/secret-type: repository` Label

```yaml
kind: Secret
apiVersion: v1
metadata:
 name: repo-2105479437
 namespace: openshift-gitops
 labels:
   argocd.argoproj.io/secret-type: repository
stringData:
 project: default
 type: git
 sshPrivateKey: <REDACTED>
 url: https://github.com/simonkrenger/gitops-basics-demos.git
type: Opaque
```


Quellen:
https://argo-cd.readthedocs.io/en/stable/operator-manual/declarative-setup/#repositories

Clusters

Ein “Secret” mit dem “secret-type: cluster” Label

- Repräsentiert einen Ziel-Cluster
- Kein CRD, Clusters werden als “Secret” mit dem Label `argocd.argoproj.io/secret-type: cluster` gespeicher
- Wird für die Speicherung von Zugangsdaten zu einem anderen Cluster (Tokens) verwendet

```yaml
kind: Secret
apiVersion: v1
metadata:
 name: mycluster-example
 namespace: openshift-gitops
 labels:
   argocd.argoproj.io/secret-type: cluster
 annotations:
stringData:
 config: |
  {"bearerToken": "<authentication token>", [..]}
 name: mycluster.example.com
 server: https://mycluster.example.com
type: Opaque
```


Quellen:
https://argo-cd.readthedocs.io/en/stable/operator-manual/declarative-setup/#clusters

Demo:
Deklarative GitOps-Konfiguration

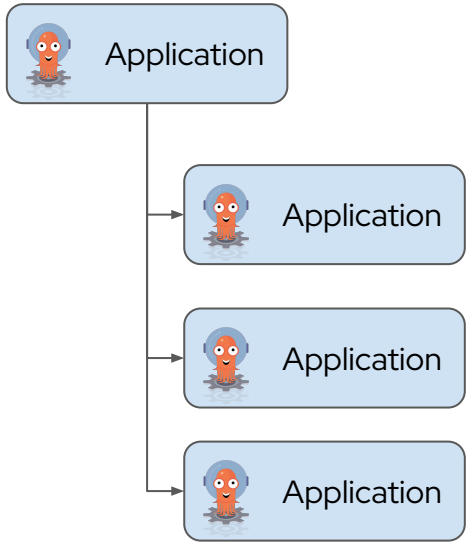
The TAM
your tech hour

OpenShift mit GitOps verwalten

App-of-apps

Applications mit einer eigenen Application verwalten

- Die “App-of-Apps” ist ein verbreitetes Modell, bei welchem eine Argo CD `Application` auf ein Repository zeigt, welches andere `Applications` beinhält
- Eignet sich gut, um zusammengehörige Applikationen zu erstellen und gemeinsam zu verwalten.
- Das Muster existiert bereits lange, Red Hat sieht das Muster bei einigen Kunden im Einsatz



ApplicationSets

Ein standardisierter Weg, viele Applications zu deployen

- Der ApplicationSet Controller generiert automatisch Argo CD `Applications` basierend auf der `ApplicationSet` Custom Resource (CR).
- Beispiele für verfügbare Generatoren:
  - List generator
  - Cluster generator
  - Git generator
  - Matrix generator

```
apiVersion: argoproj.io/v1alpha1
kind: ApplicationSet
metadata:
 name: guestbook
 namespace: openshift-gitops
Spec:
[..]
 generators:
 - list:
     elements:
     - cluster: dev
       url: https://1.2.3.4
[..]
  template:
    metadata:
      name: '{{.cluster}}-guestbook'
[..]
```

Quellen: https://argo-cd.readthedocs.io/en/stable/user-guide/application-set/

The TAM your tech hour

App-of-apps / ApplicationSets

Wann sollte man was verwenden

App-of-apps:

- Einfacher einzuführen, wenn es bereits `Applications` gibt
- Funktioniert mit allen Argo CD Versionen
- Skalierung (>100 Applications) kann problematisch werden
  - Pflege von vielen `Application` YAMLs

ApplicationSets:

- Setzt eine gewisse Standardisierung von Applications / YAMLs voraus
- Flexibilität durch Generatoren
  - Dynamische Erstellung von `Applications` (git generator)
  - Kann kompliziert werden (matrix generators)

The TAM your tech hour

Secrets Management

Zugangsdaten sollten nie im Klartext in Git abgelegt werden

- Ein Nachteil bei GitOps ist, dass auch Zugangsdaten in Git hinterlegt sind. Ein typisches Problem für unsere Kunden
- Red Hat hat aktuell keine Produkte für Secrets Management
- Support für External Secrets Operator ist geplant in der Zukunft

Red Hat bietet keinen Support oder Empfehlungen für spezifische Software von Drittanbietern. In der Praxis sehen wir aber folgende Software bei unseren Kunden:

- Bitnami Sealed Secrets
- Hashicorp Vault
- Mozilla SOPS

The TAM your tech hour

Grössere GitOps Installationen

Überlegungen für grössere Argo CD Instanzen

- Argo CD hat architektonische Einschränkungen bei grossen Installationen mit vielen Namespaces
- Das betrifft vor allem “namespace-scoped” Argo CD Instanzen
- Diese Einschränkungen sollten berücksichtigt werden, wenn mit OpenShift GitOps gestartet wird

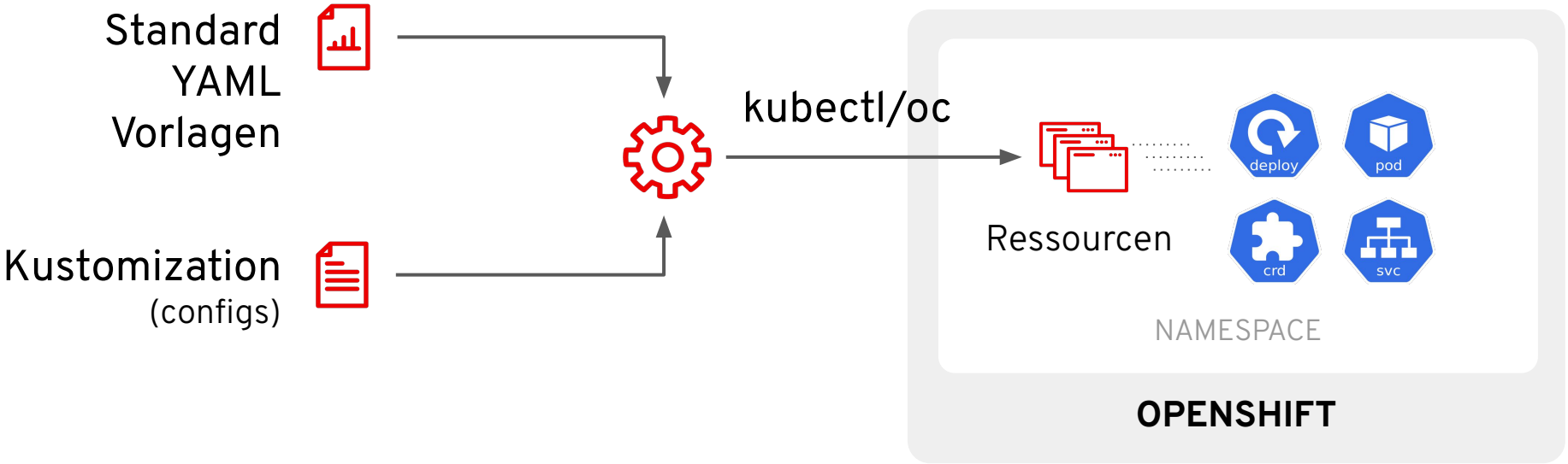
- Solution 7006291 beschreibt Möglichkeiten zur Konfiguration
  - QPS, Processors, cluster-scope, resource exclusions
- Werden mehrere Cluster mit Argo CD verwaltet, so kann sogar “dynamic sharding” eingesetzt werden

Quellen:
https://access.redhat.com/solutions/7006291
https://developers.redhat.com/articles/2023/09/26/dynamically-scale-argo-cd-application-controller-openshift-gitops-110

Kustomize / Helm

Kustomize / Helm ist in Argo CD eingebaut

Erlaubt es, Templating mit YAML zu machen



Demo:
OpenShift mit Argo CD

The TAM
your tech hour

Best Practices für Argo CD

The TAM your tech hour

Argo CD Best Practices

Aus der Argo CD “Best Practices” Dokumentation

Argo CD Dokumentation

Die Best Practices documentation Seite des Argo CD Projekts hält einige Tipps bereit

Konfiguration und Quellcode trennen

Es sollten verschiedene Repositories jeweils für den Quellcode und die Konfiguration (YAMLs) erstellt werden

Nicht alles muss konkret definiert werden

Es müssen nicht unbedingt alle Werte im YAML definiert werden (ein gutes Beispiel ist das `replicas` Feld, wenn HPA genutzt wird)

Referenzen sollten unveränderlich sein

Tags wie “latest” sollten bei Container-Images vermieden werden, Quellcode sollte mit SHA-Wert oder mit Tags referenziert werden

Quellen:
https://argo-cd.readthedocs.io/en/stable/user-guide/best_practices/

The TAM your tech hour

Tipps zum einfacheren Einsatz von GitOps

Tipps vom OpenShift GitOps Team

“annotation tracking” statt Labels

Annotationen in Kubernetes haben nicht die gleichen Einschränkungen wie Labels (.z.B. Länge), “annotation tracking” kann als Alternative verwendet werden

Mit kubectl-neat zu minimalen YAMLs

Das `neat` plugin für `kubectl` und `oc` erlaubt es, einfach YAMLs für GitOps zu exportieren

App of Apps einfach ausschalten

Mittels `ignoreDifferences` können bei App-of-Apps einfach notwendige manuelle Änderungen in `Applications` ignoriert werden

Globale AppProjects

Werden `AppProjects` verwendet, so kann mit einem globalen Projekt viel Konfiguration zentral für alle Teams verwaltet werden

Quellen:
https://www.redhat.com/en/blog/gitops-quality-life-tips
https://github.com/itaysk/kubectl-neat

The TAM your tech hour

Best Practices für unsere Kunden

Von und für Red Hat Kunden

Doppelt hält nicht immer besser

Ressourcen mit einer schnell inkrementierenden `resourceVersion` werden eventuell durch eine andere Komponente geändert, dies belastet das API

Argo CD health checks manuell ausführen

Mit dem `argocd admin` Befehl können Health Checks von Argo CD manuell ausgeführt werden

Kustomize für Inhalte von ConfigMaps

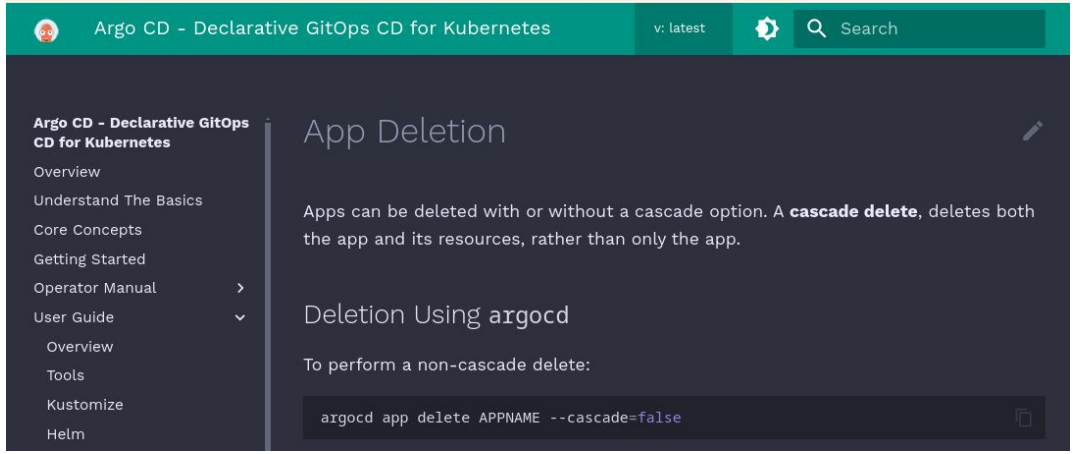
YAML in YAML ist oftmals kompliziert und fehleranfällig. Mit kustomize können `configMapGenerator` verwendet werden, um `ConfigMaps` / `Secrets` zu erzeugen.

Quellen:
https://www.redhat.com/en/blog/gitops-quality-life-tips

Achtung Gefahr: Löschen von Apps

Ein Schauermärchen

- Wenn Applications gelöscht werden, aufpassen dass nicht kaskadierend gelöscht wird - Argo CD löscht sonst auch alle Objekte der Application
- Ein Kunde hat aus Versehen beim Löschen einer Argo CD `App` alle `MachineSets` gelöscht
- Mit den Sync Options `Prune=false` oder `Delete=false` können solche Fehler bei wichtigen Objekten vermieden werden

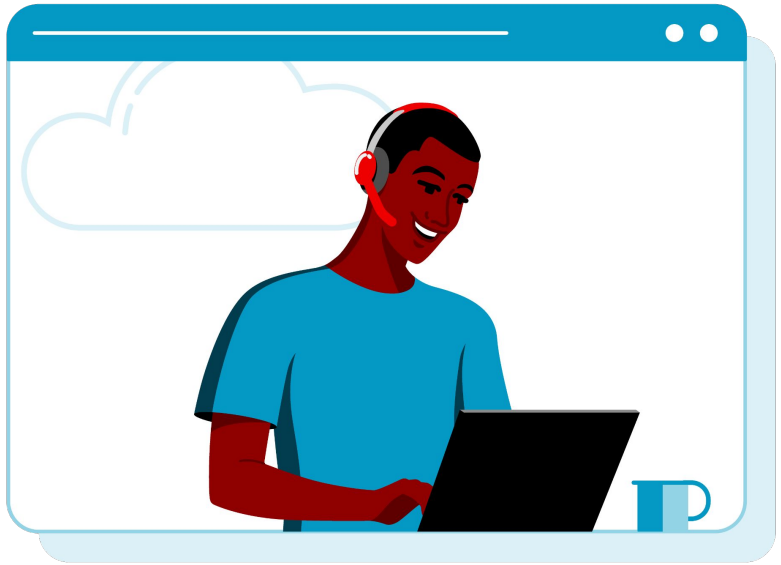


Demo:
Best Practices

Zusammenfassung

- **Typische GitOps-Installationen**
  Instanzen für Administratoren, Instanzen für Entwickler
- **Deklarative Konfiguration**
  GitOps mittels GitOps CRDs verwalten
- **OpenShift mit GitOps**
  App-of-apps, Secrets Management und grosse Argos
- **Best Practices**
  Diverse Tips für den Betrieb und aus der Praxis

Danke !



http://linkedin.com/company/Red-Hat

http://youtube.com/user/RedHatVideos

http://facebook.com/RedHatinc

https://twitter.com/RedHat

Red Hat
Technical Account Management