

Inhaltsverzeichnis

Vorwort	v
 Teil I Grundlagen	 1
1 Was ist GitOps?	3
1.1 CIOps vs. GitOps	4
1.2 Der Weg zu GitOps	9
1.2.1 Traditionelle Silos	9
1.2.2 DevOps	10
1.2.3 Infrastructure as Code	11
1.2.4 Kubernetes	14
1.2.5 OpenGitOps	14
1.3 Die vier Prinzipien	16
1.3.1 Prinzip 1: Deklarativ	16
1.3.2 Prinzip 2: Versioniert und unveränderlich	18
1.3.3 Prinzip 3: Automatisch bezogen	20
1.3.4 Prinzip 4: Kontinuierlich angeglichen	20
1.4 Fragen und Missverständnisse	23
 2 Welchen Unterschied macht GitOps?	 25
2.1 Indizien aus den DORA-Studien	25
2.2 Der Unterschied im Alltag: Geschichten eines Entwicklungsteams	28
2.3 Szenario	29
2.3.1 Repositories	29
2.3.2 Deployment-Fluss	31
2.3.3 Vergleichsszenario »Mit GitOps«	31
2.4 Kontinuierlich nach Dev deployen	34
2.5 Ressourcen aufräumen	35
2.6 Grenzfälle in CI vermeiden	36
2.7 Ressourcen wiederherstellen	39
2.8 Konfigurationsänderungen ausrollen	41

2.9	Incidents navigieren	43
2.10	Zielsysteme besser absichern	45
3	Wie fange ich mit GitOps an?	47
3.1	Agile Empfehlung: zügiger Durchstich	47
3.2	Fragen zur Orientierung	49
3.3	Eine Beispielimplementierung mit Argo CD	51
3.3.1	Zielsetzung	51
3.3.2	Voraussetzungen	54
3.4	Schritt 1: Einen Cluster starten	54
3.5	Schritt 2: Argo CD installieren	54
3.5.1	Ressourcentypen von Argo CD verstehen	56
3.6	Schritt 3: Eine eigene Application erzeugen	61
3.7	Schritt 4: Eine Änderung deployen	64
3.8	Fazit	65

Teil II Praxis 67

4	Argo CD oder Flux auswählen	69
4.1	Zahlen und Fakten	69
4.2	Bootstrapping	71
4.3	Linking	72
4.4	CLI und GUI	73
4.5	Komponenten und Ressourcenbedarf	75
4.6	Authentifizierung und Autorisierung	79
4.7	Templating	79
4.8	Configuration Management	80
4.9	Monitoring und Alerting	82
4.10	Ökosystem	85
4.11	Mandantentrennung	88
4.12	Multi-Cluster-Management	90
4.13	OCI statt Git	90
4.14	Hochverfügbarkeit und Lastverteilung	93
4.15	Reifegrad	93
4.16	Kommerzielle Angebote	94
4.17	Fazit und Tipps zur Entscheidungsfindung	95
5	Secrets sicher verwalten	97
5.1	Secrets lagern und verwalten	98
5.1.1	Secrets verschlüsselt im Repo speichern	98
5.1.2	Secrets extern verwalten	102

5.2	Secrets konsumieren	107
5.2.1	Secrets als native Kubernetes-Secrets konsumieren ..	108
5.2.2	Secrets über Sidecar-Container injizieren	111
5.2.3	Secrets über ein CSI-Volume konsumieren	113
5.3	Wir erweitern die Beispielimplementierung	115
5.3.1	Ziele	116
5.3.2	Datenfluss von HashiCorp Vault über ESO in den Cluster	117
5.3.3	Schritt 1: Das Config-Repo bootstrappen	119
5.3.4	Schritt 2: Anwendungen in neuen Namespace deployen	120
5.3.5	Schritt 3: ESO mit HashiCorp Vault verbinden	124
5.3.6	Schritt 4: Beispiel-Secret erstellen	126
5.3.7	Schritt 5: Das Secret integrieren	129
5.3.8	Schritt 6: Das Secret ändern	130
5.4	Fazit	131
6	Repositories und Ordner strukturieren	133
6.1	Design des GitOps-Prozesses	133
6.2	Kategorien von GitOps-Patterns	136
6.3	Operator Deployment Patterns	137
6.3.1	Instance per Cluster	138
6.3.2	Hub and Spoke	138
6.3.3	Instance per Namespace	139
6.4	Repository Patterns	139
6.4.1	Monorepo	140
6.4.2	Repo per Team	140
6.4.3	Repo per Application	141
6.4.4	Repo per Environment	149
6.5	Promotion Patterns	150
6.5.1	Branch oder Folder per Environment	151
6.5.2	Preview Environments	157
6.5.3	Global Environments oder Environment per Application	159
6.5.4	Configuration Management	160
6.5.5	Config Update	163
6.6	Verdrahtungs-Patterns	168
6.6.1	Bootstrapping	168
6.6.2	Linking	168
6.7	Beispiele für Config-Repos	169
6.7.1	Beispiel 1: Argo CD Autopilot	170
6.7.2	Beispiel 2: GitOps Playground	175
6.7.3	Beispiel 3: Flux Monorepo	180

6.7.4	Beispiel 4: Flux Repo per Team	183
6.7.5	Beispiel 5: The Path to GitOps	185
6.7.6	Beispiel 6: Environment-Varianten	187
6.8	Mandantentrennung	189
6.8.1	Rolle der GitOps-Operatoren	189
6.8.2	Rolle der Repo-Struktur	190
6.8.3	Rolle der Cluster	191
6.8.4	Teams und Environments trennen	192
6.9	Fazit	194
7	Asynchron deployen	197
7.1	Deployment-Flüsse	197
7.1.1	Schritte	197
7.1.2	Kombinationen von Zuständigkeiten	199
7.1.3	Deployment-Fluss 1: CIOps	200
7.1.4	Deployment-Fluss 2: Schmal-GitOps	201
7.1.5	Deployment-Fluss 3: CI-Klammer	202
7.1.6	Deployment-Fluss 4: Übergabe	203
7.1.7	Deployment-Fluss 5: GitOps-geführt	204
7.2	Rollout durch den GitOps-Operator	205
7.2.1	Rollout-Schritte im GitOps-Operator	205
7.2.2	Timing und Koordination der Rollout-Schritte	208
7.2.3	Intervalle der Rollout-Schritte	211
7.2.4	Rollout im GitOps-Operator aktiv triggern	212
7.3	Config Update	215
7.3.1	Argo CD Image Updater	216
7.3.2	Flux Image Automation	216
7.3.3	Dependency-Bot	217
7.4	Prüfen	218
7.4.1	Prüfen via CI-Server	218
7.4.2	Prüfen via GitOps-Operator	219
7.4.3	Progressive Delivery	221
7.5	CI-Server mit GitOps verwalten	224
7.6	Fazit	226
8	Alerting integrieren	229
8.1	Gesundheitszustand feststellen	230
8.1.1	Kubernetes-nativ mit kstatus	230
8.1.2	Helm-Hooks	235
8.1.3	Flux	235
8.1.4	Argo CD	238

8.2	Benachrichtigungen verschicken	239
8.2.1	Flux	240
8.2.2	Argo CD	242
8.3	Ganzheitliche Herangehensweise	245
9	Imperativ eingreifen	247
9.1	Eindeutig ausgeschlossene Aktionen	247
9.2	Risiken und Chancen	248
9.3	Einen Debug-Pod starten	250
9.4	Ein Backup wiederherstellen	251
9.5	Ein Deployment neu starten	255
9.6	Ressourcen neu erzeugen	256
9.7	Ein Deployment skalieren	258
9.8	Fazit	260

Teil III Weiterführendes 261

10	Mehrere Cluster verwalten	263
10.1	Single-Cluster vs. Multi-Cluster	264
10.2	Cluster API	265
10.3	Konzept von Cluster API	266
10.4	Definition der Cluster API-Ressourcen	266
10.5	Installation des Management-Clusters	269
10.5.1	Provider konfigurieren und temporären Management-Cluster bauen	270
10.5.2	Worker Cluster anlegen	272
10.5.3	Installation des Cluster API Operators	275
10.5.4	Deklarative Installation des Management-Clusters ...	276
10.6	Verwaltung von Workload-Clustern mit Argo CD	284
10.7	Fazit	289
11	Infrastruktur verwalten	291
11.1	Terraform	292
11.1.1	Ein kurzes Tutorial zum Einstieg	292
11.1.2	Grundbausteine von Terraform	295
11.1.3	Terraform vs. OpenTofu	298
11.1.4	Terraform und GitOps	299
11.1.5	Eine EC2-Instanz verwalten mit TF-Controller und Flamingo	301

11.2	Pulumi	309
11.2.1	Grundbausteine von Pulumi	310
11.2.2	Eine EC2-Instanz verwalten mit Pulumi Kubernetes Operator	314
11.3	Crossplane	321
11.3.1	Grundbausteine von Crossplane	321
11.3.2	Eine EC2-Instanz verwalten mit Crossplane	323
12	GitOps außerhalb von Kubernetes	333
12.1	Aus den GitOps-Prinzipien folgende Verantwortlichkeiten ...	334
12.2	Infrastructure-as-Code-Formate	335
12.3	Weitere GitOps-Operatoren	338
12.4	Features von GitOps-Operatoren	338
12.5	Einen eigenen GitOps-Operator bauen	340
12.6	Eigene GitOps-Operatoren aus der Praxis	341
12.6.1	Docker Swarm und Ansible	341
12.6.2	Helmfile	348
12.6.3	Lektionen	350

Teil IV Anhang **351**

Nachwort	353
-----------------------	------------

Index	357
--------------------	------------