

Inhaltsverzeichnis

Vorwort	xix
Einleitung	xxi
1 Revolution in der Cloud	1
1.1 Die Entstehung der Cloud	2
1.1.1 Zeit einkaufen	3
1.1.2 Infrastructure as a Service	3
1.2 Der Aufstieg von DevOps	3
1.2.1 Keiner versteht DevOps	5
1.2.2 Der Geschäftsvorteil	6
1.2.3 Infrastruktur als Code	6
1.2.4 Gemeinsames Lernen	7
1.3 Das Aufkommen von Containern	7
1.3.1 State of the Art	8
1.3.2 Innerhalb der Box denken	8
1.3.3 Software in Containern verpacken	9
1.3.4 Plug-and-Play-Anwendungen	10
1.4 Das Container-Orchester dirigieren	11
1.5 Kubernetes	12
1.5.1 Von Borg zu Kubernetes	12
1.5.2 Was macht Kubernetes so wertvoll?	13
1.5.3 Wird Kubernetes wieder verschwinden?	14
1.5.4 Kubernetes kann nicht alles	15
1.6 Cloud Native	16
1.7 Die Zukunft von Operations	19
1.7.1 Verteiltes DevOps	19
1.7.2 Manches wird zentralisiert bleiben	19
1.7.3 Developer Productivity Engineering	20
1.7.4 Sie sind die Zukunft	21
1.8 Zusammenfassung	21

2	Erste Schritte mit Kubernetes	23
2.1	Starten Sie Ihren ersten Container	23
2.1.1	Docker Desktop installieren	24
2.1.2	Was ist Docker?	24
2.1.3	Ein Container-Image starten	25
2.2	Die Demo-Anwendung	25
2.2.1	Den Quellcode anschauen	26
2.2.2	Go?	26
2.2.3	Wie die Demo-Anwendung funktioniert	27
2.3	Einen Container bauen	27
2.3.1	Dockerfiles verstehen	28
2.3.2	Minimale Container-Images	28
2.3.3	docker image build ausführen	29
2.3.4	Ihrem Image einen Namen geben	29
2.3.5	Port Forwarding	30
2.4	Container-Registries	30
2.4.1	Sich an der Registry authentifizieren	31
2.4.2	Ihr Image benennen und pushen	31
2.4.3	Ihr Image ausführen	31
2.5	Hallo Kubernetes	32
2.5.1	Die Demo-App starten	32
2.5.2	Wenn der Container nicht startet	33
2.6	Minikube	34
2.7	Zusammenfassung	34
3	Kubernetes-Installationen	35
3.1	Cluster-Architektur	36
3.1.1	Die Steuerungsebene	36
3.1.2	Komponenten auf den Knoten	37
3.1.3	Hochverfügbarkeit	38
3.2	Die Kosten eines selbst gehosteten Kubernetes	39
3.2.1	Es ist mehr, als Sie denken	40
3.2.2	Es geht nicht nur um das initiale Setup	41
3.2.3	Tools nehmen Ihnen nicht die ganze Arbeit ab	41
3.2.4	Kubernetes ist schwer	42
3.2.5	Administrativer Overhead	42
3.2.6	Beginnen Sie mit Managed Services	42
3.3	Managed Kubernetes Services	44
3.3.1	Google Kubernetes Engine (GKE)	44
3.3.2	Cluster Autoscaling	45

3.3.3	Amazon Elastic Container Service for Kubernetes (EKS) . . .	45
3.3.4	Azure Kubernetes Service (AKS)	46
3.3.5	OpenShift	46
3.3.6	IBM Cloud Kubernetes Service	46
3.3.7	Heptio Kubernetes Subscription (HKS)	46
3.4	Turnkey-Kubernetes-Lösungen	47
3.4.1	Stackpoint	47
3.4.2	Containership Kubernetes Engine (CKE)	47
3.5	Kubernetes-Installer	48
3.5.1	kops	48
3.5.2	Kubespray	48
3.5.3	TK8	49
3.5.4	Kubernetes The Hard Way	49
3.5.5	kubeadm	49
3.5.6	Tarmak	50
3.5.7	Rancher Kubernetes Engine (RKE)	50
3.5.8	Puppet Kubernetes Module	50
3.5.9	Kubeformation	50
3.6	Kaufen oder bauen: Unsere Empfehlungen	51
3.6.1	Run Less Software	51
3.6.2	Nutzen Sie Managed Kubernetes, wenn Sie können	52
3.6.3	Aber was ist mit dem Vendor Lock-In?	52
3.6.4	Nutzen Sie Self-Hosting-Standard-Tools für Kubernetes, wenn Sie müssen	53
3.6.5	Wenn Ihre Auswahl begrenzt ist	53
3.6.6	Bare-Metal und On-Prem	54
3.7	Clusterless Container Services	54
3.7.1	Amazon Fargate	55
3.7.2	Azure Container Instances (ACI)	55
3.8	Zusammenfassung	56
4	Mit Kubernetes-Objekten arbeiten	59
4.1	Deployments	59
4.1.1	Supervising und Scheduling	60
4.1.2	Container neu starten	60
4.1.3	Deployments abfragen	61
4.2	Pods	62
4.3	ReplicaSets	62
4.4	Den gewünschten Status warten	63
4.5	Der Kubernetes-Scheduler	64
4.6	Ressourcen-Manifeste im YAML-Format	65

4.6.1	Ressourcen sind Daten	65
4.6.2	Deployment-Manifeste	66
4.6.3	kubectl apply verwenden	66
4.6.4	Service-Ressourcen	67
4.6.5	Das Cluster mit kubectl abfragen	69
4.6.6	Ressourcen noch leistungsfähiger machen	70
4.7	Helm: Ein Kubernetes-Paketmanager	71
4.7.1	Helm installieren	71
4.7.2	Einen Helm-Chart installieren	72
4.7.3	Charts, Repositories und Releases	73
4.7.4	Helm-Releases anzeigen	73
4.8	Zusammenfassung	74
5	Ressourcen managen	77
5.1	Ressourcen verstehen	77
5.1.1	Ressourcen-Einheiten	78
5.1.2	Ressourcen-Anforderungen	78
5.1.3	Ressourcen-Grenzen	79
5.1.4	Halten Sie Ihre Container klein	80
5.2	Den Lebenszyklus des Containers managen	81
5.2.1	Liveness-Probe	81
5.2.2	Verzögerung und Häufigkeit der Probe	82
5.2.3	Andere Arten von Proben	82
5.2.4	gRPC-Proben	83
5.2.5	Readiness-Proben	83
5.2.6	Dateibasierte Readiness-Proben	84
5.2.7	minReadySeconds	84
5.2.8	Pod Disruption Budgets	85
5.3	Namensräume verwenden	86
5.3.1	Mit Namensräumen arbeiten	87
5.3.2	Welchen Namensraum sollte ich verwenden?	87
5.3.3	Service-Adressen	88
5.3.4	Resource Quotas	89
5.3.5	Standards für Ressourcen-Anforderungen und -Grenzen	90
5.4	Die Kosten von Clustern optimieren	91
5.4.1	Deployments optimieren	91
5.4.2	Pods optimieren	92
5.4.3	Vertical Pod Autoscaler	93
5.4.4	Knoten optimieren	93
5.4.5	Storage optimieren	95
5.4.6	Ungenutzte Ressourcen aufräumen	96
5.4.7	Freie Kapazitäten prüfen	98

5.4.8	Reservierte Instanzen nutzen	98
5.4.9	Präemptive (Spot)-Instanzen verwenden	99
5.4.10	Sorgen Sie für eine ausgeglichene Verteilung Ihrer Workloads	101
5.5	Zusammenfassung	103
6	Cluster betreiben	105
6.1	Sizing und Skalieren des Clusters	105
6.1.1	Kapazitätsplanung	106
6.1.2	Knoten und Instanzen	109
6.1.3	Das Cluster skalieren	111
6.2	Konformitäts-Prüfungen	113
6.2.1	CNCF Certification	114
6.2.2	Konformitäts-Tests mit Sonobuoy	115
6.3	Validierung und Auditing	116
6.3.1	K8Guard	117
6.3.2	Copper	117
6.3.3	kube-bench	118
6.3.4	Kubernetes Audit Logging	118
6.4	Chaos Testing	118
6.4.1	Nur Produktion ist Produktion	119
6.4.2	chaoskube	120
6.4.3	kube-monkey	120
6.4.4	PowerfulSeal	121
6.5	Zusammenfassung	121
7	Power-Tools für Kubernetes	123
7.1	Die Arbeit mit kubectl	123
7.1.1	Shell-Aliasse	123
7.1.2	Kurze Flags verwenden	124
7.1.3	Ressourcen-Typen abkürzen	124
7.1.4	Autovervollständigen von kubectl-Befehlen	125
7.1.5	Hilfe erhalten	125
7.1.6	Hilfe zu Kubernetes-Ressourcen erhalten	126
7.1.7	Detailliertere Ausgaben nutzen	126
7.1.8	Mit JSON-Daten und jq arbeiten	127
7.1.9	Objekte beobachten	128
7.1.10	Objekte beschreiben	128
7.2	Mit Ressourcen arbeiten	128
7.2.1	Imperative kubectl-Befehle	128
7.2.2	Wann Sie keine imperativen Befehle verwenden sollten	129

7.2.3	Ressourcen-Manifeste generieren	130
7.2.4	Ressourcen exportieren	131
7.2.5	Ressourcen diffen	131
7.3	Mit Containern arbeiten	132
7.3.1	Die Logs eines Containers anzeigen	132
7.3.2	Sich mit einem Container verbinden	133
7.3.3	Kubernetes-Ressourcen mit kubespys beobachten	133
7.3.4	Einen Container-Port weiterleiten	134
7.3.5	Befehle in Containern ausführen	134
7.3.6	Container zur Fehlersuche ausführen	135
7.3.7	BusyBox-Befehle verwenden	136
7.3.8	BusyBox zu Ihren Containern hinzufügen	137
7.3.9	Programme auf einem Container installieren	138
7.3.10	Live Debugging mit kubescape	138
7.4	Kontexte und Namensräume	139
7.4.1	kubectx und kubens	140
7.4.2	kube-ps1	141
7.5	Shells und Tools für Kubernetes	142
7.5.1	kube-shell	142
7.5.2	Click	142
7.5.3	kubed-sh	143
7.5.4	Stern	143
7.6	Bauen Sie Ihre eigenen Kubernetes-Tools	144
7.7	Zusammenfassung	145
8	Container ausführen	147
8.1	Container und Pods	147
8.1.1	Was ist ein Container?	148
8.1.2	Was gehört in einen Container?	149
8.1.3	Was gehört in einen Pod?	150
8.2	Container-Manifeste	151
8.2.1	Image-Bezeichner	152
8.2.2	Das Tag »latest«	153
8.2.3	Container-Digests	153
8.2.4	Basis-Image-Tags	154
8.2.5	Ports	154
8.2.6	Ressourcen-Anforderungen und -Grenzen	154
8.2.7	Image-Pull-Richtlinie	155
8.2.8	Umgebungsvariablen	155
8.3	Sicherheit von Containern	156
8.3.1	Container als Nicht-Root-Benutzer ausführen	157
8.3.2	Root-Container blockieren	158

8.3.3	Ein Read-Only-Dateisystem nutzen	158
8.3.4	Privilege Escalation deaktivieren	159
8.3.5	Capabilities	159
8.3.6	Pod Security Contexts	161
8.3.7	Pod Security Policies	161
8.3.8	Pod Service-Accounts	162
8.4	Volumes	162
8.4.1	emptyDir-Volumes	163
8.4.2	Persistente Volumes	164
8.5	Neustart-Richtlinien	165
8.6	Image Pull Secrets	165
8.7	Zusammenfassung	166
9	Pods managen	167
9.1	Labels	167
9.1.1	Was sind Labels?	167
9.1.2	Selektoren	168
9.1.3	Komplexere Selektoren	169
9.1.4	Andere Einsatzzwecke für Labels	170
9.1.5	Labels und Anmerkungen	171
9.2	Node Affinities	171
9.2.1	Hard Affinities	172
9.2.2	Soft Affinities	173
9.3	Pod Affinities und Anti-Affinities	173
9.3.1	Pods zusammenhalten	174
9.3.2	Pods auseinanderhalten	175
9.3.3	Soft Anti-Affinities	175
9.3.4	Wann Sie Pod Affinities verwenden	176
9.4	Taints und Tolerations	176
9.5	Pod-Controller	178
9.5.1	DaemonSets	178
9.5.2	StatefulSets	180
9.5.3	Jobs	181
9.5.4	Cronjobs	182
9.5.5	Horizontal Pod Autoscaler	183
9.5.6	PodPresets	184
9.5.7	Operatoren und Custom Resource Definitions (CRDs)	186
9.6	Ingress-Ressourcen	187
9.6.1	Ingress-Regeln	187
9.6.2	TLS Termination mit Ingress	188
9.6.3	Ingress-Controller	189

9.7	Istio	190
9.8	Envoy	191
9.9	Zusammenfassung	192
10	Konfiguration und Secrets	195
10.1	ConfigMaps	195
10.1.1	ConfigMaps erstellen	196
10.1.2	Umgebungsvariablen aus ConfigMaps setzen	197
10.1.3	Die gesamte Umgebung aus einer ConfigMap erstellen ...	199
10.1.4	Umgebungsvariablen in Argumenten für Befehle einsetzen .	200
10.1.5	Konfigurationsdateien aus ConfigMaps erstellen	201
10.1.6	Pods bei einer Konfigurationsänderung aktualisieren	203
10.2	Kubernetes-Secrets	203
10.2.1	Secrets als Umgebungsvariablen verwenden	204
10.2.2	Secrets in Dateien schreiben	205
10.2.3	Secrets lesen	205
10.2.4	Zugriff auf Secrets	206
10.2.5	Encryption at Rest	207
10.2.6	Secrets behalten	207
10.3	Strategien zum Verwalten von Secrets	207
10.3.1	Secrets in der Versionsverwaltung verschlüsseln	208
10.3.2	Secrets remote ablegen	209
10.3.3	Ein dediziertes Secrets-Management-Tool einsetzen	209
10.3.4	Empfehlungen	210
10.4	Secrets mit Sops verschlüsseln	211
10.4.1	Einführung in Sops	211
10.4.2	Eine Datei mit Sops verschlüsseln	212
10.4.3	Ein KMS-Backend verwenden	214
10.5	Zusammenfassung	214
11	Sicherheit und Backups	217
11.1	Zugriffskontrolle und Berechtigungen	217
11.1.1	Den Zugriff vom Cluster abhängig machen	217
11.1.2	Role-Based Access Control (RBAC)	218
11.1.3	Rollen	219
11.1.4	Rollen mit Benutzern verbinden	220
11.1.5	Welche Rolle benötige ich?	220
11.1.6	Den Zugriff auf cluster-admin beschränken	221
11.1.7	Anwendungen und Deployment	221
11.1.8	Fehlersuche mit RBAC	222

11.2	Security Scanning	223
11.2.1	Clair	223
11.2.2	Aqua	223
11.2.3	Anchore Engine	224
11.3	Backups	224
11.3.1	Muss ich ein Backup von Kubernetes machen?	225
11.3.2	Backup von etcd	225
11.3.3	Backup des Ressourcen-Status	226
11.3.4	Backup des Cluster-Status	226
11.3.5	Große und kleine Katastrophen	226
11.3.6	Velero	227
11.4	Den Cluster-Status monitoren	230
11.4.1	kubectl	230
11.4.2	CPU- und Speicherauslastung	232
11.4.3	Konsole des Cloud-Providers	233
11.4.4	Kubernetes Dashboard	234
11.4.5	Weave Scope	235
11.4.6	kube-ops-view	235
11.4.7	node-problem-detector	236
11.5	Weitere Informationen	237
11.6	Zusammenfassung	237
12	Kubernetes-Anwendungen deployen	239
12.1	Manifeste mit Helm bauen	239
12.1.1	Was befindet sich in einem Helm-Chart?	240
12.1.2	Helm-Templates	241
12.1.3	Variablen interpolieren	242
12.1.4	Template-Werte mit Anführungszeichen versehen	243
12.1.5	Abhängigkeiten festlegen	243
12.2	Helm-Charts deployen	244
12.2.1	Variablen setzen	244
12.2.2	Werte in einem Helm-Release angeben	245
12.2.3	Eine Anwendung mit Helm aktualisieren	245
12.2.4	Zu früheren Versionen zurückrollen	246
12.2.5	Ein Helm-Chart-Repo erstellen	246
12.2.6	Helm-Chart-Secrets mit Sops managen	247
12.3	Mehrere Charts mit Helmfile managen	249
12.3.1	Was befindet sich in einem Helmfile?	249
12.3.2	Chart-Metadaten	250
12.3.3	Das Helmfile anwenden	251

12.4	Fortgeschrittene Tools zur Manifest-Verwaltung	252
12.4.1	ksonnet	252
12.4.2	Kapitan	253
12.4.3	kustomize	254
12.4.4	kompose	254
12.4.5	Ansible	255
12.4.6	kubeval	255
12.5	Zusammenfassung	256
13	Entwicklungs-Workflow	259
13.1	Entwicklungs-Tools	259
13.1.1	Skaffold	259
13.1.2	Draft	260
13.1.3	Telepresence	260
13.1.4	Knative	261
13.2	Deployment-Strategien	261
13.2.1	Rollierende Updates	262
13.2.2	Recreate	262
13.2.3	maxSurge und maxUnavailable	263
13.2.4	Blue/Green-Deployments	264
13.2.5	Rainbow-Deployments	265
13.2.6	Canary-Deployments	265
13.3	Migration mit Helm	265
13.3.1	Helm Hooks	266
13.3.2	Umgang mit fehlgeschlagenen Hooks	267
13.3.3	Andere Hooks	267
13.3.4	Hooks verketteten	267
13.4	Zusammenfassung	268
14	Continuous Deployment in Kubernetes	271
14.1	Was ist Continuous Deployment?	271
14.2	Welches CD-Tool soll ich verwenden?	272
14.2.1	Jenkins	272
14.2.2	Drone	273
14.2.3	Google Cloud Build	273
14.2.4	Concourse	273
14.2.5	Spinnaker	273
14.2.6	GitLab CI	274
14.2.7	Codefresh	274
14.2.8	Azure Pipelines	274

14.3	CD-Komponenten	274
14.3.1	Docker Hub	274
14.3.2	Gitkube	275
14.3.3	Flux	275
14.3.4	Keel	275
14.4	Eine CD-Pipeline mit Cloud Build	275
14.4.1	Google Cloud und GKE einrichten	276
14.4.2	Das Demo-Repository forken	276
14.4.3	Erste Schritte in Cloud Build	276
14.4.4	Den Test-Container bauen	277
14.4.5	Die Tests ausführen	277
14.4.6	Den Anwendungs-Container bauen	278
14.4.7	Die Kubernetes-Manifeste überprüfen	278
14.4.8	Das Image veröffentlichen	279
14.4.9	Git-SHA-Tags	279
14.4.10	Den ersten Build-Trigger erstellen	279
14.4.11	Den Trigger testen	280
14.4.12	Von einer CD-Pipeline deployen	281
14.4.13	Einen Deploy-Trigger erstellen	283
14.4.14	Die Build-Pipeline optimieren	284
14.4.15	Die Beispiel-Pipeline anpassen	284
14.5	Zusammenfassung	285
15	Observability und Monitoring	287
15.1	Was ist Observability?	287
15.1.1	Was ist Monitoring?	287
15.1.2	Black-Box Monitoring	288
15.1.3	Was heißt »Up«?	289
15.1.4	Logging	291
15.1.5	Metriken	292
15.1.6	Tracing	294
15.1.7	Observability	295
15.2	Die Observability-Pipeline	296
15.3	Monitoring in Kubernetes	297
15.3.1	Externe Black-Box Checks	298
15.3.2	Interne Health-Checks	299
15.4	Zusammenfassung	301

16	Metriken in Kubernetes	303
16.1	Was sind Metriken wirklich?	303
16.1.1	Zeitreihen-Daten	304
16.1.2	Zähler und Maße	304
16.1.3	Was können uns Metriken sagen?	305
16.2	Die Wahl guter Metriken	305
16.2.1	Services: Das RED-Muster	306
16.2.2	Ressourcen: Das USE-Muster	307
16.2.3	Business-Metriken	308
16.2.4	Kubernetes-Metriken	309
16.3	Metriken analysieren	313
16.3.1	Was ist an einem einfachen Durchschnitt falsch?	313
16.3.2	Mittelwert, Median und Ausreißer	314
16.3.3	Perzentile	315
16.3.4	Perzentile auf Metrikdaten anwenden	315
16.3.5	Meist wollen wir das Schlimmste wissen	317
16.3.6	Mehr als Perzentile	317
16.4	Metriken in Dashboards visualisieren	318
16.4.1	Ein Standard-Layout für alle Services verwenden	318
16.4.2	Einen Information Radiator mit Master Dashboards bauen	319
16.4.3	Dashboards mit Fehlerursachen	321
16.5	Alarme durch Metriken	321
16.5.1	Was ist falsch an Alarmen?	322
16.5.2	Bereitschaftsdienst sollte nicht die Hölle sein	323
16.5.3	Dringende, wichtige und umsetzbare Alarme	324
16.5.4	Dokumentieren Sie Warnungen und Alarme außerhalb der Bürozeiten und in der Nacht	325
16.6	Tools und Services für Metriken	325
16.6.1	Prometheus	326
16.6.2	Google Stackdriver	328
16.6.3	AWS Cloudwatch	328
16.6.4	Azure Monitor	329
16.6.5	Datadog	329
16.6.6	New Relic	330
16.7	Zusammenfassung	331
17	Nachwort	333
17.1	Wohin als Nächstes?	333
17.2	Willkommen an Bord	334
	Index	335