

Inhaltsverzeichnis

Teil I	Hintergrund und Grundlagen	1
1	Was Container sind und warum man sie nutzt	3
1.1	Container versus VMs	4
1.2	Docker und Container	6
1.3	Eine Geschichte von Docker	9
1.4	Plugins und Plumbing	10
1.5	64-Bit-Linux	11
2	Installation	13
2.1	Docker auf Linux installieren	13
2.1.1	SELinux im Permissive Mode ausführen	14
2.1.2	Ohne sudo starten	15
2.2	Docker auf Mac OS oder Windows installieren	15
2.3	Ein schneller Check	17
3	Erste Schritte	19
3.1	Ihr erstes Image ausführen	19
3.2	Die grundlegenden Befehle	20
3.3	Images aus Dockerfiles erstellen	24
3.4	Mit Registries arbeiten	28
3.4.1	Private Repositories	29
3.5	Das offizielle Redis-Image verwenden	30
3.6	Zusammenfassung	34
4	Grundlagen von Docker	35
4.1	Die Architektur von Docker	35
4.1.1	Zugrunde liegende Technologien	36
4.1.2	Zugehörige Technologien	37
4.1.3	Docker Hosting	40

4.2	Wie Images gebaut werden	40
4.2.1	Der Build Context	41
4.2.2	Imageschichten	42
4.2.3	Caching	44
4.2.4	Basis-Images	45
4.2.5	Anweisungen im Dockerfile	47
4.3	Container mit der Außenwelt verbinden	51
4.4	Container verlinken	52
4.5	Daten mit Volumes und Datencontainern verwalten	53
4.5.1	Daten gemeinsam nutzen	56
4.5.2	Datencontainer	56
4.6	Häufig eingesetzte Docker-Befehle	58
4.6.1	Der Befehl run	59
4.6.2	Container verwalten	62
4.6.3	Docker-Info	64
4.6.4	Container-Info	65
4.6.5	Arbeit mit Images	66
4.6.6	Die Registry verwenden	69
4.7	Zusammenfassung	70

Teil II	Der Software-Lebenszyklus mit Docker	71
----------------	---	-----------

5	Docker in der Entwicklung einsetzen	73
5.1	Sag »Hallo Welt!«	73
5.2	Mit Compose automatisieren	83
5.2.1	Der Compose-Workflow	84
5.3	Zusammenfassung	86
6	Eine einfache Webanwendung erstellen	87
6.1	Eine einfache Webseite erstellen	88
6.2	Auf vorhandene Images zurückgreifen	90
6.3	Caching ergänzen	95
6.4	Microservices	98
6.5	Zusammenfassung	99
7	Bereitstellen von Images	101
7.1	Namensgebung für Images und Repositories	101
7.2	Der Docker Hub	102

7.3	Automatisierte Builds	104
7.4	Private Distribution	106
7.4.1	Eine eigene Registry betreiben	106
7.4.2	Kommerzielle Registries	113
7.5	Die Imagegröße verringern	114
7.6	Herkunft eines Image	116
7.7	Zusammenfassung	117
8	Continuous Integration und Testen mit Docker	119
8.1	identidock mit Unit-Tests versehen	119
8.2	Einen Jenkins-Container erstellen	124
8.2.1	Builds triggern	132
8.3	Das Image pushen	132
8.3.1	Sinnvolles Taggen	133
8.3.2	Staging und Produktion	135
8.3.3	Image Sprawl	135
8.3.4	Jenkins Slaves durch Docker betreiben	136
8.4	Backups für Jenkins	136
8.5	Gehostete CI-Lösungen	136
8.6	Testen und Microservices	137
8.6.1	Im Produktivumfeld testen	139
8.7	Zusammenfassung	139
9	Container deployen	141
9.1	Ressourcen mit Docker Machine aufsetzen	142
9.2	Einen Proxy verwenden	145
9.3	Ausführungsoptionen	151
9.3.1	Shell-Skripten	152
9.3.2	Einen Process Manager einsetzen (oder systemd, sie alle zu knechten)	154
9.3.3	Ein Tool zum Configuration Management einsetzen	157
9.4	Host-Konfiguration	161
9.4.1	Ein Betriebssystem wählen	161
9.4.2	Einen Storage-Treiber wählen	162
9.5	Spezialisierte Hosting-Möglichkeiten	165
9.5.1	Triton	165
9.5.2	Google Container Engine	167
9.5.3	Amazon EC2 Container Service	167
9.5.4	Giant Swarm	170

9.6	Persistente Daten und Produktivcontainer	172
9.7	Gemeinsame Geheimnisse	172
9.7.1	Geheimnisse im Image ablegen	172
9.7.2	Geheimnisse in Umgebungsvariablen übergeben	173
9.7.3	Geheimnisse in Volumes übergeben	174
9.7.4	Einen Key/Value-Store einsetzen	174
9.8	Vernetzen	176
9.9	Produktiv-Registry	176
9.10	Continuous Deployment/Delivery	176
9.11	Zusammenfassung	177
10	Protokollieren und Überwachen	179
10.1	Protokollieren	180
10.1.1	Standard-Logging von Docker	180
10.1.2	Logs zusammenfassen	182
10.1.3	Mit ELK loggen	182
10.1.4	Docker-Logging mit syslog	193
10.1.5	Logs aus Dateien auslesen	199
10.2	Überwachen und Benachrichtigen	199
10.2.1	Mit den Docker-Tools überwachen	200
10.2.2	cAdvisor	201
10.2.3	Cluster-Lösungen	203
10.3	Kommerzielle Monitoring- und Logging-Lösungen	206
10.4	Zusammenfassung	206
Teil III	Tools und Techniken	209
11	Vernetzung und Service Discovery	211
11.1	Ambassadors	212
11.2	Service Discovery	216
11.2.1	etcd	217
11.2.2	SkyDNS	221
11.2.3	Consul	226
11.2.4	Registrieren	231
11.2.5	Andere Lösungen	232
11.3	Networking-Optionen	234
11.3.1	Bridge	234
11.3.2	Host	235

11.3.3	Container	235
11.3.4	None	236
11.4	Neues Docker-Networking	236
11.4.1	Netzwerktypen und Plugins	238
11.5	Vernetzungslösungen	238
11.5.1	Overlay	239
11.5.2	Weave	241
11.5.3	Flannel	245
11.5.4	Project Calico	251
11.6	Zusammenfassung	256
12	Orchestrieren, Clustering und Verwaltung	259
12.1	Clustering- und Orchestrierungstools	260
12.1.1	Swarm	261
12.1.2	fleet	268
12.1.3	Kubernetes	274
12.1.4	Mesos und Marathon	283
12.2	Container-Management-Plattformen	294
12.2.1	Rancher	295
12.2.2	Clocker	296
12.2.3	Tutum	298
12.3	Zusammenfassung	299
13	Container sichern und beschränken	301
13.1	Worüber Sie sich Gedanken machen sollten	302
13.2	Verteidigung in der Tiefe	304
13.2.1	Least Privilege	304
13.3	identidock absichern	305
13.4	Container nach Host trennen	307
13.5	Updates anwenden	308
13.5.1	Nicht unterstützte Treiber vermeiden	312
13.6	Imageherkunft	312
13.6.1	Docker Digests	313
13.6.2	Docker Content Trust	313
13.6.3	Reproduzierbare und vertrauenswürdige Dockerfiles	318
13.7	Sicherheitstipps	320
13.7.1	Einen Benutzer setzen	321
13.7.2	Netzwerkzugriffe von Containern beschränken	322
13.7.3	setuid/setgid-Binaries entfernen	324

13.7.4	Den Speicher begrenzen	325
13.7.5	Den CPU-Einsatz beschränken	326
13.7.6	Neustarts begrenzen	327
13.7.7	Zugriffe auf die Dateisysteme begrenzen	328
13.7.8	Capabilities einschränken	328
13.7.9	Ressourcenbeschränkungen (ulimits) anwenden	330
13.8	Einen gehärteten Kernel ausführen	332
13.9	Linux Security Modules	333
13.9.1	SELinux	333
13.9.2	AppArmor	336
13.10	Auditing	337
13.11	Reaktion auf Zwischenfälle	338
13.12	Zukünftige Features	339
13.13	Zusammenfassung	339

Index**341**