

Cloud-native Computing

Software Engineering von Diensten und
Applikationen für die Cloud

DAS INHALTS- VERZEICHNIS

» Hier geht's
direkt
zum Buch

Inhalt

| | |
|---|-------------|
| Vorwort | XIII |
| 1 Einleitung | 1 |
| 1.1 An wen sich dieses Buch richtet | 2 |
| 1.2 Was dieses Buch behandelt | 3 |
| 1.3 Sprachliche Konventionen | 5 |
| 1.4 Notationskonventionen | 6 |
| 1.5 Ergänzende Materialien | 7 |
| Teil I: Grundlagen | 9 |
| 2 Cloud Computing | 11 |
| 2.1 Service-Modelle | 12 |
| 2.1.1 Infrastructure as a Service (IaaS) | 15 |
| 2.1.2 Platform as a Service (PaaS) | 15 |
| 2.1.3 Software as a Service (SaaS) | 16 |
| 2.2 Cloud-Ökonomie | 19 |
| 2.2.1 Eignung von unterschiedlichen Arten von Workloads | 19 |
| 2.2.2 Effekt von Zuteilungsdauer und Ressourcengröße | 22 |
| 2.3 Entwicklung der letzten Jahre | 24 |
| 3 DevOps | 27 |
| 3.1 Prinzipien des Flow | 29 |
| 3.1.1 Prinzip 1: Arbeit sichtbar machen | 29 |
| 3.1.2 Prinzip 2: Work in Progress beschränken | 30 |
| 3.1.3 Prinzip 3: Flaschenhalse minimieren | 31 |
| 3.2 Prinzipien des Feedbacks | 32 |
| 3.2.1 Prinzip 4: Probleme früh erkennen | 32 |
| 3.2.2 Prinzip 5: Probleme sofort lösen | 32 |
| 3.2.3 Prinzip 6: Probleme professionell verantworten | 33 |
| 3.3 DevOps-geeignete Architekturen | 33 |
| 3.3.1 Randbedingungen für die Entwicklung | 34 |

| | | |
|----------|--|-----------|
| 3.3.2 | Nutzung von Orchestrierungsplattformen..... | 34 |
| 3.3.3 | Randbedingungen im Betrieb | 35 |
| 4 | Cloud-native | 37 |
| 4.1 | Definitionen in Industrie und Forschung..... | 39 |
| 4.2 | Die Cloud-native-Definition dieses Buchs..... | 40 |
| 4.3 | Zusammenfassung und Ausblick auf Teil II bis IV | 41 |
| | Teil II: Everything as Code..... | 45 |
| 5 | Einleitung zu Teil II..... | 47 |
| 6 | Deployment-Pipelines | 49 |
| 6.1 | Deployment-Pipelines as Code..... | 50 |
| 6.1.1 | Phasen-Pipelines..... | 51 |
| 6.1.2 | Gerichtete Pipelines | 52 |
| 6.1.3 | Hierarchische Pipelines..... | 53 |
| 6.1.4 | Steuerung von Pipelines | 54 |
| 6.2 | DevOps-geeignete Branching-Strategien..... | 56 |
| 6.2.1 | Git-Flow | 57 |
| 6.2.2 | GitHub-Flow | 58 |
| 6.2.3 | Trunk-basierte Entwicklung | 59 |
| 6.3 | Zusammenfassung | 60 |
| 7 | Infrastructure as Code..... | 63 |
| 7.1 | Virtualisierung | 65 |
| 7.1.1 | Virtualisierung von Hardware-Infrastruktur..... | 65 |
| 7.1.2 | Virtualisierung von Software-Infrastruktur..... | 66 |
| 7.2 | Provisionierung..... | 68 |
| 7.2.1 | Immutable Infrastructure | 68 |
| 7.2.2 | IaC-Ansätze | 69 |
| 7.2.3 | Provisionierung von lokalen Umgebungen | 72 |
| 7.2.4 | Provisionierung von Multi-Host-Umgebungen | 74 |
| 7.3 | Zusammenfassung | 77 |
| 8 | Standardisierung von Deployment Units (Container) | 79 |
| 8.1 | Hintergrund (PaaS)..... | 79 |
| 8.2 | Betriebssystem-Virtualisierung..... | 82 |
| 8.3 | Container Runtime Environments..... | 83 |
| 8.3.1 | Kernel-Namespaces | 84 |
| 8.3.2 | Process Capabilities | 85 |
| 8.3.3 | Control Groups | 86 |

| | | |
|----------|---|-----------|
| 8.3.4 | Union Filesystem | 86 |
| 8.3.5 | High-Level- und Low-Level-Container-Laufzeitumgebungen | 87 |
| 8.4 | Bau und Bereitstellung von Container-Images | 88 |
| 8.5 | Faktoren gut betreibbarer Container | 90 |
| 8.5.1 | Codebase | 91 |
| 8.5.2 | Abhängigkeiten und Konfigurationen | 91 |
| 8.5.3 | Unterstützende Services und Port Binding | 92 |
| 8.5.4 | Build-, Release- und Run-Phase | 93 |
| 8.5.5 | Horizontale Skalierung über Prozesse | 94 |
| 8.5.6 | Umgebungen, Logs und Betrieb | 95 |
| 8.6 | Zusammenfassung | 96 |
| 9 | Container-Plattformen | 99 |
| 9.1 | Scheduling | 100 |
| 9.1.1 | Heterogenität von Workloads | 101 |
| 9.1.2 | Scheduling-Algorithmen | 102 |
| 9.1.2.1 | Einfache Scheduling-Algorithmen | 102 |
| 9.1.2.2 | Multidimensionale Scheduling-Algorithmen | 103 |
| 9.1.2.3 | Kapazitätsbasierte Scheduling-Algorithmen | 103 |
| 9.1.3 | Scheduling-Architekturen | 104 |
| 9.1.3.1 | Monolithischer Scheduler | 105 |
| 9.1.3.2 | 2-Level-Scheduler | 105 |
| 9.1.3.3 | Shared-State Scheduler | 106 |
| 9.2 | Orchestrierung | 107 |
| 9.2.1 | Definition von Betriebszuständen | 107 |
| 9.2.2 | Regelkreis: Desired versus Current State | 108 |
| 9.3 | Inside Kubernetes | 109 |
| 9.3.1 | Kubernetes-Architektur | 110 |
| 9.3.2 | Verwaltete Ressourcen und Basis-Blueprint | 112 |
| 9.3.3 | Schedulbare Workloads | 114 |
| 9.3.3.1 | Deployments | 114 |
| 9.3.3.2 | (Cron-)Jobs | 116 |
| 9.3.3.3 | Daemon-Sets | 117 |
| 9.3.3.4 | Stateful-Sets | 118 |
| 9.3.4 | Scheduling Constraints | 121 |
| 9.3.4.1 | Angabe des Ressourcenbedarfs mittels Requests und Limits | 121 |
| 9.3.4.2 | Knoten-Selektoren | 122 |
| 9.3.4.3 | Knotenaffinitäten | 123 |
| 9.3.4.4 | Pod-(Anti-)Affinitäten | 124 |
| 9.3.5 | Automatische Skalierung von Workloads | 125 |
| 9.3.6 | Exponieren von Workloads als interne und externe Services | 126 |
| 9.3.7 | Health Checking | 129 |
| 9.3.8 | Persistenz | 132 |

| | | |
|---|--|------------|
| 9.3.9 | Isolation von Workloads | 133 |
| 9.3.9.1 | Namespaces und Role-based Access Model (Multi-Tenancy) | 133 |
| 9.3.9.2 | Quotas und Limit Ranges | 134 |
| 9.3.9.3 | Network Policys | 135 |
| 9.4 | Zusammenfassung | 137 |
| 10 | Function as a Service | 141 |
| 10.1 | FaaS-Plattformen | 143 |
| 10.1.1 | Das FaaS-Programmiermodell | 145 |
| 10.1.2 | Zu berücksichtigende Randbedingungen | 146 |
| 10.1.3 | Veranschaulichung des FaaS-Programmiermodells | 147 |
| 10.2 | Plattformagnostische FaaS-Frameworks | 149 |
| 10.3 | Ereignisbasierte Autoskalierung | 152 |
| 10.4 | Zusammenfassung | 155 |
| Teil III: Cloud-native Architekturen | | 157 |
| 11 | Einleitung zu Teil III | 159 |
| 12 | Microservice und Serverless-Architekturen | 161 |
| 12.1 | Eigenschaften von Microservices | 162 |
| 12.2 | Integrationsmuster für Microservices | 166 |
| 12.2.1 | Datenbankbasierte Integration | 167 |
| 12.2.2 | (g)RPC-basierte Interprozesskommunikation | 167 |
| 12.2.3 | Representational State Transfer (REST) | 170 |
| 12.2.4 | Ereignisbasierte Integration (asynchron) | 173 |
| 12.2.5 | API-Versioning | 175 |
| 12.3 | Architekturelle Sicherheit | 178 |
| 12.3.1 | Circuit-Breaker | 178 |
| 12.3.2 | Bulkhead | 179 |
| 12.3.3 | Idempotente API-Operationen | 180 |
| 12.4 | Skalierung von Microservices | 180 |
| 12.4.1 | Load Balancing | 181 |
| 12.4.2 | Messaging | 181 |
| 12.4.3 | Skalierung zustandsbehafteter Komponenten | 183 |
| 12.4.3.1 | Scaling for Reads | 184 |
| 12.4.3.2 | Scaling for Writes (Sharding) | 184 |
| 12.4.3.3 | Command Query Responsibility Segregation (CQRS) | 185 |
| 12.4.4 | Caching | 186 |
| 12.5 | Prinzipien zur Entwicklung von Microservices | 187 |
| 12.5.1 | Prinzip 1: Bilde Modelle um Geschäftskonzepte | 187 |
| 12.5.2 | Prinzip 2: Erschaffe eine Kultur der Automatisierung | 187 |

| | | |
|-----------|--|------------|
| 12.5.3 | Prinzip 3: Blende interne Implementierungsdetails aus | 188 |
| 12.5.4 | Prinzip 4: Dezentralisiere | 188 |
| 12.5.5 | Prinzip 5: Definiere unabhängig aktualisierbare Einheiten | 188 |
| 12.5.6 | Prinzip 6: Isoliere Fehler | 189 |
| 12.5.7 | Prinzip 7: Baue gut beobachtbare Services | 189 |
| 12.6 | Serverless-Architekturen | 190 |
| 12.6.1 | Architekturelle Konsequenzen von Serverless-Limitierungen | 191 |
| 12.6.2 | Das API-Gateway-Pattern | 193 |
| 12.6.3 | Abgrenzung zu Microservices | 195 |
| 12.7 | Zusammenfassung | 196 |
| 13 | Beobachtbare Architekturen | 199 |
| 13.1 | Konsolidierung von Telemetriedaten | 200 |
| 13.2 | Instrumentierung von Systemen | 202 |
| 13.2.1 | Logging | 202 |
| 13.2.2 | Monitoring | 204 |
| 13.2.2.1 | Metrikarten | 206 |
| 13.2.2.2 | Empfehlungen für die Metrikinstrumentierung | 207 |
| 13.2.3 | Tracing | 207 |
| 13.2.3.1 | Empfehlungen für die Instrumentierung | 209 |
| 13.2.3.2 | Tracing-Instrumentierung und Erzeugung von Spans | 212 |
| 13.2.3.3 | Serverseitiges Tracing und Extraktion von Span-Kontexten | 213 |
| 13.2.3.4 | Clientseitiges Tracing und Weiterreichen von Span-Kontexten | 214 |
| 13.3 | Automatisierte Instrumentierung | 215 |
| 13.3.1 | Eigenschaften von Service-Meshs | 216 |
| 13.3.2 | Traffic-Management | 218 |
| 13.3.3 | Resilienz | 221 |
| 13.3.4 | Sicherheit | 223 |
| 13.3.5 | Management und Analyse von Verkehrstopologien | 226 |
| 13.4 | Zusammenfassung | 227 |
| 14 | Domain-driven Design | 229 |
| 14.1 | Fachlichkeit | 230 |
| 14.2 | Strategisches Design | 232 |
| 14.2.1 | Subdomänen | 233 |
| 14.2.1.1 | Kerndomäne (Core Subdomain) | 233 |
| 14.2.1.2 | Unterstützende Subdomäne (Supporting Subdomain) | 234 |
| 14.2.1.3 | Generische Subdomänen (Generic Subdomain) | 234 |
| 14.2.1.4 | Anmerkungen am Beispiel einer Fallstudie | 234 |
| 14.2.2 | Ubiquitous Language | 236 |
| 14.2.2.1 | Eine gemeinsame Sprache als Schlüssel zu einem gemeinsamen Verständnis | 237 |
| 14.2.2.2 | Mehrdeutige und synonyme Begriffe | 238 |

| | | |
|----------|---|-----|
| 14.2.3 | Bounded Contexts | 239 |
| 14.2.4 | Context Mapping | 241 |
| 14.2.4.1 | Partnerschaftliche Kooperationsmuster (Partners und Shared-Kernel) | 241 |
| 14.2.4.2 | Customer-Supplier-Kooperation | 243 |
| 14.2.4.3 | Separate Ways | 244 |
| 14.2.4.4 | Context Maps als Landkarte von Machtverhältnissen | 245 |
| 14.3 | Taktisches Design | 246 |
| 14.3.1 | Oft genutzte Pattern für Geschäftslogik | 246 |
| 14.3.1.1 | Das ETL-Pattern (primär Supporting Subdomains) | 246 |
| 14.3.1.2 | Das Active Record-Pattern (primär Supporting Subdomains) | 247 |
| 14.3.1.3 | Das Domain Model-Pattern (primär Core Subdomains) | 248 |
| 14.3.1.4 | Das Event-Sourcing-Pattern (primär Core Subdomains) | 250 |
| 14.3.2 | Oft genutzte Pattern für die Architektur | 251 |
| 14.3.2.1 | Die Ebenen-Architektur | 252 |
| 14.3.2.2 | Das Ports & Adapter-Pattern | 253 |
| 14.3.2.3 | Das CORS-Pattern | 253 |
| 14.4 | Zusammenfassung | 256 |

Teil IV: Sichere Cloud-native Anwendungen 259

15 Einleitung zu Teil IV 261

16 Härtung Cloud-nativer Anwendungen 263

| | | |
|----------|---|-----|
| 16.1 | Härtung (virtueller) Infrastrukturen | 267 |
| 16.1.1 | Tool-gestütztes System Hardening | 268 |
| 16.1.2 | Kontinuierliche Aktualisierung von virtuellen Maschinen | 270 |
| 16.1.3 | Sichere Authentifizierung mittels SSH | 271 |
| 16.1.4 | Kontinuierliche Überwachung virtueller Maschinen | 273 |
| 16.1.4.1 | Verhaltensbasierte Intrusion Detection mittels Auditing | 273 |
| 16.1.4.2 | Signaturbasierte Intrusion Detection | 274 |
| 16.1.4.3 | Log Forwarding | 276 |
| 16.1.5 | Einsatz von Sicherheitsgruppen und Firewalls | 277 |
| 16.2 | Härtung containerisierter Workloads | 279 |
| 16.2.1 | Absicherung von Public Endpoints mittels Ingresses | 280 |
| 16.2.2 | Namespace-basierte Netzwerkisolation | 286 |
| 16.2.3 | Pod Hardening | 289 |
| 16.2.4 | Erhöhung der Container Runtime Isolation | 299 |
| 16.2.5 | Volume Hardening | 300 |
| 16.2.6 | Workload Policing | 303 |
| 16.2.7 | Intrusion Detection | 307 |
| 16.2.8 | Sicherung der Supply Chain | 310 |

| | | |
|-----------|--|------------|
| 16.2.8.1 | Statische Software Composition Analysis (SCA) | 311 |
| 16.2.8.2 | Static Application Security Testing (SAST) | 314 |
| 16.2.8.3 | Kontinuierliches Schwachstellen-Scanning von Container- Plattformen | 316 |
| 16.3 | Zusammenfassung | 319 |
| 17 | Regulatorische Anforderungen | 321 |
| 17.1 | Cloud Compliance und Zertifizierungen | 322 |
| 17.1.1 | ISO 9001 | 324 |
| 17.1.2 | BSI-IT-Grundschutz und BSI-Standards | 325 |
| 17.1.3 | BSI-C5-Zertifizierung | 325 |
| 17.1.4 | ISO/IEC 27001 und 27017/27018 | 326 |
| 17.1.5 | CSA STAR | 327 |
| 17.1.6 | CISPE Code of Conduct | 328 |
| 17.1.7 | EU Cloud Code of Conduct | 329 |
| 17.1.8 | SOC 1-3 (Service Organization Control) | 330 |
| 17.1.9 | FedRAMP | 331 |
| 17.1.10 | HIPAA | 331 |
| 17.1.11 | PCI DSS | 332 |
| 17.1.12 | Zusammenfassung | 333 |
| 17.2 | Aus der DSGVO sich ergebende Anforderungen | 335 |
| 17.2.1 | Personenbezogene Daten | 335 |
| 17.2.2 | Grundsätze der Verarbeitung personenbezogener Daten | 336 |
| 17.2.3 | Auftragsverarbeitung | 338 |
| 17.2.4 | Datenschutz-Folgenabschätzungen | 340 |
| 17.2.5 | Internationale Datentransfers in Drittländer | 341 |
| 17.3 | Europäischer Datenschutz und Drittländer | 343 |
| 17.3.1 | Probleme am Beispiel des CLOUD Act | 344 |
| 17.3.2 | Lösungen trotz SCHREMS I + II | 345 |
| 17.4 | Zusammenfassung | 346 |
| 18 | Schlussbemerkungen | 349 |
| | Literaturverzeichnis | 359 |
| | Stichwortverzeichnis | 365 |