

Inhalt

Geleitwort	XV
Vorwort der Autoren	XVII
Über dieses Buch	XXI
Teil 1: Erste Schritte	1
1 Einführung in Apache Kafka	3
1.1 Was ist Apache Kafka und wie löst es unsere Probleme?	3
1.2 Apache Kafka in Unternehmens-Architekturen	5
1.3 Architektonischer Überblick über Apache Kafka	8
1.4 Apache Kafka betreiben und nutzen	10
1.5 Unser Lernpfad	11
2 Erste Schritte mit Kafka	13
2.1 Einführung in unseren Anwendungsfall	13
2.2 Produzieren von Nachrichten	14
2.3 Konsumieren von Nachrichten	15
2.4 Paralleles Konsumieren und Produzieren von Nachrichten	16
2.5 Grafische Benutzeroberflächen für Kafka	19
Teil 2: Konzepte	23
3 Erkundung von Kafka-Topics und -Nachrichten	25
3.1 Topics	25

3.1.1	Anzeigen von Topics	26
3.1.2	Erstellen, Anpassen und Löschen von Topics	29
3.2	Nachrichten	32
3.2.1	Nachrichtentypen	33
3.2.2	Datenformate	35
3.2.3	Nachrichtenstruktur	36
4	Kafka als verteiltes Log	41
4.1	Logs	41
4.1.1	Was genau ist ein Log?	41
4.1.2	Grundlegende Eigenschaften eines Logs	43
4.1.3	Kafka als Log	44
4.2	Kafka als verteiltes System	46
4.2.1	Partitionierung und Keys	48
4.2.2	Consumer-Gruppen	53
4.2.3	Replikation	56
4.3	Komponenten von Kafka	60
4.3.1	Koordinationscluster	60
4.3.2	Broker	62
4.3.3	Clients	63
4.4	Kafka im geschäftlichen Einsatz	63
5	Zuverlässigkeit	67
5.1	Acknowledgements	68
5.1.1	Acknowledgement-Strategien in Kafka	69
5.1.2	Acknowledgements und ISR	71
5.1.3	Nachrichten-Zustellungsgarantien in Kafka	74
5.2	Transaktionen	77
5.2.1	Transaktionen in Datenbanken	78
5.2.2	Transaktionen in Kafka	79
5.2.3	Transaktionen und Consumer	81
5.3	Replikation und das Leader-Follower-Prinzip	83
6	Performance	91
6.1	Konfiguration von Topics für Performance	93
6.1.1	Skalierung und Lastverteilung	93

6.1.2	Wie viele Partitionen sollten wir haben?	95
6.1.3	Ändern der Anzahl von Partitionen	97
6.2	Producer-Performance	100
6.2.1	Producer-Konfiguration	100
6.2.2	Producer Performance Test	103
6.3	Broker-Konfiguration und Optimierung	106
6.3.1	Broker-Optimierung	107
6.3.2	Bestimmung der Broker-Anzahl und -Größe	107
6.4	Consumer-Performance	108
6.4.1	Consumer-Konfiguration	109
6.4.2	Consumer-Performance-Test	109
Teil 3: Kafka Deep Dive		113
7	Cluster-Management	115
7.1	KRaft Cluster-Management	116
7.2	ZooKeeper Cluster-Management	118
7.3	Migration von ZooKeeper zu KRaft	120
7.4	Verbindung zu Kafka	121
8	Produzieren und Persistieren von Nachrichten	125
8.1	Producer	125
8.1.1	Produzieren von Nachrichten	126
8.1.2	Was passiert während der Nachrichtenproduktion?	128
8.1.3	Producer und ACKs	129
8.2	Broker	130
8.2.1	Empfang und Persistenz von Nachrichten	130
8.2.2	Broker und ACKs	132
8.3	Daten- und Dateistrukturen	132
8.3.1	Metadaten, Checkpoints und Topics	133
8.3.2	Partitionsverzeichnis	134
8.3.3	Log-Daten und Indizes	138
8.3.4	Segmente	139
8.3.5	Gelöschte Topics	141
8.4	Replikation	142
8.4.1	In-Sync Replicas	142

- 8.4.2 High Watermark 143
 - 8.4.3 Auswirkungen von Verzögerungen während der Replikation 145
- 9 Konsumieren von Nachrichten 149
 - 9.1 Abrufen von Nachrichten 150
 - 9.1.1 Fetch-Anfragen 150
 - 9.1.2 Fetch from the closest replica 151
 - 9.2 Broker-Handhabung von Consumer-Fetch-Anfragen 151
 - 9.3 Offsets und Consumer 153
 - 9.3.1 Offset-Verwaltung 153
 - 9.3.2 Offsets in Kafka verstehen 154
 - 9.4 Kafka Consumer Groups im Detail 157
 - 9.4.1 Verwaltung von Consumer Groups 157
 - 9.4.2 Verteilung der Partitionen auf die Consumer 160
 - 9.4.3 Static Memberships 164
- 10 Aufräumen von Nachrichten 167
 - 10.1 Warum müssen wir Nachrichten aufräumen? 167
 - 10.2 Kafkas Methoden zum Aufräumen 168
 - 10.3 Log Retention 169
 - 10.3.1 Wann wird ein Log durch Retention aufgeräumt? 170
 - 10.3.2 Offset Retention 173
 - 10.4 Log Compaction 173
 - 10.4.1 Wann wird ein Log durch Log Compaction aufgeräumt? 175
 - 10.4.2 Wie funktioniert der Log Cleaner? 178
 - 10.4.3 Tombstones 180
- Teil 4: Kafka im Unternehmenseinsatz 183
- 11 Integration externer Systeme mit Kafka Connect 185
 - 11.1 Was ist Kafka Connect? 185
 - 11.2 Kafka Connect Cluster 188
 - 11.2.1 Konfiguration eines Kafka Connect Clusters 188
 - 11.2.2 Erstellung eines Connectors 189
 - 11.2.3 Testen des Connectors 190
 - 11.3 Skalierbarkeit und Ausfallsicherheit von Kafka Connect 191

11.4	Worker-Konfiguration	193
11.5	Die REST-API von Kafka Connect	196
11.5.1	Status eines Connect-Clusters	196
11.5.2	Erstellen, Ändern und Löschen von Connectoren	199
11.6	Konfiguration von Connectoren	201
11.6.1	Allgemeine Konfiguration eines Connectors	201
11.6.2	Fehlerbehandlung in Kafka Connect	202
11.7	Single Message Transformations	204
11.8	Connect-Beispiel: JDBC Source Connector	206
11.8.1	Vorbereitung des JDBC Source Connectors	206
11.8.2	Konfiguration des JDBC Source Connectors	207
11.8.3	Testen des JDBC Source Connectors	208
11.9	Connect-Beispiel: Change Data Capture Connector	210
11.9.1	Vorbereitung des Debezium-Connectors für PostgreSQL	211
11.9.2	Konfiguration des Debezium Connectors für PostgreSQL	212
11.9.3	Testen des Debezium Connectors für PostgreSQL	213
12	Stream-Processing	217
12.1	Überblick über Stream-Processing	218
12.1.1	Stream-Processing-Bibliotheken	219
12.1.2	Datenverarbeitung	220
12.2	Stream-Prozessoren	221
12.2.1	Prozessortypen	222
12.2.2	Prozessor-Topologien	225
12.3	Stream-Verarbeitung mit SQL	227
12.4	Stream States	229
12.4.1	Streams und Tabellen	230
12.4.2	Aggregationen	232
12.4.3	Streaming Joins	234
12.4.4	Anwendungsfall: Benachrichtigungen	237
12.5	Streaming und Zeit	240
12.5.1	Zeit ist relativ	240
12.5.2	Zeitfenster	241
12.5.3	Anwendungsfall: Betrugserkennung	243
12.6	Skalierung von Kafka Streams	245

- 13 Governance 251**
 - 13.1 Schema-Management 252
 - 13.1.1 Warum brauchen wir Schemas? 253
 - 13.1.2 Kompatibilitätsstufen 254
 - 13.1.3 Schema Registries 258
 - 13.1.4 Avro 260
 - 13.2 Security 262
 - 13.2.1 Transportverschlüsselung 263
 - 13.2.2 Authentifizierung 265
 - 13.2.3 Autorisierung 266
 - 13.2.4 Verschlüsselung in Ruhe 268
 - 13.2.5 Ende-zu-Ende-Verschlüsselung 269
 - 13.2.6 ZooKeeper-Sicherheit 270
 - 13.2.7 Absicherung eines unsicheren Kafka-Clusters 270
 - 13.3 Quotas in Kafka: Schutz des Clusters vor Überlastung 272
- 14 Kafka-Referenzarchitektur 277**
 - 14.1 Nützliche Komponenten und Werkzeuge 279
 - 14.1.1 kcat 279
 - 14.1.2 Grafische Benutzeroberflächen 280
 - 14.1.3 Verwaltung von Kafka-Ressourcen 281
 - 14.1.4 Cruise Control 282
 - 14.2 Deployment-Umgebungen 284
 - 14.2.1 Kafka auf eigener Hardware 284
 - 14.2.2 Kafka in virtualisierten Umgebungen 285
 - 14.2.3 Kafka in Kubernetes: Strimzi 285
 - 14.2.4 Kafka in der Public Cloud betreiben 287
 - 14.3 Hardwareanforderungen 288
 - 14.3.1 Broker 289
 - 14.3.2 Koordinationscluster 291
- 15 Kafka Monitoring und Alerting 293**
 - 15.1 Infrastruktur-Metriken 294
 - 15.2 Broker-Metriken 295
 - 15.2.1 Kafka-Server-Metriken 295
 - 15.2.2 Kafka-Log-Metriken 297

15.2.3	Kafka-Netzwerk-Metriken	298
15.2.4	Kafka-Controller-Metriken	298
15.3	Client-Metriken	300
15.3.1	Allgemeine Client-Metriken	301
15.3.2	Producer-Metriken	303
15.3.3	Consumer-Metriken	305
15.3.4	Kafka Connect und Kafka Streams-Metriken	308
15.4	Alerting	310
15.4.1	Von Metriken zu Alerts	310
15.4.2	Von Alerts zur Problemlösung	311
15.5	Kafka-Deployment-Umgebungen und deren Monitoring-Herausforderungen	312
15.5.1	Kafka auf eigener Hardware	312
15.5.2	Kafka auf virtuellen Maschinen	313
15.5.3	Kafka in der Public Cloud	313
15.5.4	Kafka in Kubernetes	313
15.5.5	Kafka als Managed Service	314
15.5.6	Sicherheitsaspekte in verschiedenen Umgebungen	314
16	Desaster Management	317
16.1	Was kann schon schief gehen?	318
16.1.1	Netzwerkfehler	318
16.1.2	Compute-Fehler	320
16.1.3	Speicherfehler	321
16.1.4	Ausfälle von Rechenzentren	322
16.2	Backups in Kafka	324
16.3	Spiegeln des Kafka-Cluster mit MirrorMaker	326
16.3.1	Active-Passive-Topologie	327
16.3.2	Active-Active-Topologie	328
16.3.3	Hub-and-Spoke-Topologie	329
17	Vergleich mit anderen Technologien	333
17.1	Data on the Outside vs. Data on the Inside	334
17.2	Klassische Messaging-Systeme vs. Kafka	335
17.2.1	Kafka ist agnostisch	336
17.2.2	Betriebliche Komplexität in klassischen Messaging-Systemen	337

- 17.2.3 Governance klassischer Messaging-Systeme 338
- 17.3 REST vs. Kafka 341
 - 17.3.1 Herausforderungen der synchronen Kommunikation 341
 - 17.3.2 Alternative Kommunikationsstrategien 342
- 17.4 Relationale Datenbanken vs. Kafka 343
 - 17.4.1 Stärken und Schwächen relationaler Datenbanken 343
 - 17.4.2 Komplementäre Rollen von Kafka und relationalen Datenbanken in modernen Datenarchitekturen 345
- 17.5 Kafka als Kern einer Streaming-Plattform 346
- 18 Kafkas Rolle in modernen Unternehmensarchitekturen 351**
 - 18.1 Kafka als Kern eines Data Mesh 352
 - 18.1.1 Die Herausforderungen des traditionellen Datenmanagements .. 352
 - 18.1.2 Die Prinzipien eines Data Mesh 352
 - 18.1.3 Data Mesh vs. traditionelle Ansätze 354
 - 18.1.4 Kafkas Rolle und Verantwortung bei der Implementierung eines Data Mesh 355
 - 18.2 Daten aus Kernsystemen mit Kafka befreien 357
 - 18.3 Kafka für Big Data 359
 - 18.4 Kafka für das industrielle Internet der Dinge 360
 - 18.4.1 Anwendungsfälle für Kafka im industriellen Internet der Dinge 361
 - 18.4.2 Herausforderungen bei der Datenspeicherung und -aufbewahrung 362
 - 18.4.3 Datenintegration und Zugriffsmanagement 362
 - 18.4.4 Wann sollten wir mehrere Kafka-Cluster verwenden? 364
 - 18.5 Was Kafka nicht ist 365
 - 18.5.1 Kafka ist keine relationale Datenbank 365
 - 18.5.2 Kafka ist keine synchrone Kommunikationsschnittstelle 366
 - 18.5.3 Kafka ist keine Dateiaustauschplattform 367
 - 18.5.4 Kafka für kleine Anwendungen ist fraglich 368
 - 18.5.5 Kafka ist kein Ersatz für gute Architektur 369
- Teil 5: Anhang 371**
 - 19 Anhang A – Einrichten einer Kafka-Testumgebung 373**
 - 20 Anhang B – Monitoring Setup 379**
 - Stichwortverzeichnis 389**