

Diese Leseprobe haben Sie beim
 edv-buchversand.de heruntergeladen.
Das Buch können Sie online in unserem
Shop bestellen.

[Hier zum Shop](#)

Auf einen Blick

1	Einführung und Installation	23
2	Basiseinrichtung und erstes Inventory-Management	41
3	Ad-hoc-Kommandos und Patterns	55
4	Die Konfigurations- und Serialisierungssprache YAML	65
5	Playbooks und Tasks: die Grundlagen	73
6	Playbooks und Tasks: fortgeschrittene Methoden	95
7	Module und Collections verwenden	155
8	Modularisierung mit Rollen und Includes	175
9	Webinterfaces	199
10	Weitere Tools und Techniken	223
11	Ansible und Docker	255
12	Inventory-Management: fortgeschrittene Methoden	281
13	Ansible und die Cloud	293
14	Ansible als Orchestrierungswerkzeug	307
15	Ansible und Windows	323
16	Callback-Plugins	337
17	Eigene Collections und Module erstellen	345
18	Entwickeln und Testen mit Molecule	367
19	Kochrezepte, Howtos und Best Practices	379
20	Was könnte noch besser sein bzw. was fehlt noch?	401

Inhalt

Vorwort	17
Über dieses Buch	19

1 Einführung und Installation 23

1.1 Was ist Ansible?	23
1.2 Was ist Ansible nicht?	25
1.3 Geschichte und Versionen	26
1.4 Setup/Laborumgebung	27
1.5 Ansible-Installation auf dem Control Host	31
1.6 Authentifizierung und Autorisierung auf den Target Hosts	35
1.7 Einrichten der SSH-Public-Key-Authentifizierung	36
1.8 Ein Ad-hoc-Test ohne jegliche Konfiguration	37
1.9 Noch ein Hinweis zur Migration von älteren Versionen	39

2 Basiseinrichtung und erstes Inventory-Management 41

2.1 Verzeichnisstruktur einrichten	41
2.2 Grundkonfiguration (»ansible.cfg«)	42
2.3 Erstellen und Verwalten eines statischen Inventorys	43
2.4 Inventory-Aliasse und Namensbereiche	46
2.5 Jenseits von Ping	47
2.6 Ein etwas komplexeres Beispiel	50
2.7 Alternative bzw. mehrere Inventorys	51

3	Ad-hoc-Kommandos und Patterns	55
3.1	Ad-hoc-Kommandos	55
3.2	Use Cases jenseits von »command« und »shell«	57
3.3	Idempotenz	57
3.4	Interne Funktionsweise	59
3.4.1	Parallele Ausführung	59
3.4.2	Persistente Verbindungen	60
3.4.3	Was passiert beim Aufruf eines Moduls?	61
3.5	Die Ansible-Konsole	62
3.6	Patterns zum Adressieren von Hosts	63
4	Die Konfigurations- und Serialisierungssprache YAML	65
4.1	Syntax und Struktur	65
4.2	YAML-Files editieren	66
4.3	Listen und Maps	68
4.4	Verschachtelte Strukturen	69
4.5	Textpassagen und Block-Ausdrücke	70
4.6	Das Nichts in YAML	71
4.7	Anchors und References	72
5	Playbooks und Tasks: die Grundlagen	73
5.1	Hallo Ansible – das allererste Playbook	73
5.2	Formulierung von Tasks	77
5.3	Beenden von Plays	79
5.4	Der problematische Doppelpunkt	79
5.5	Fehlerbehandlung, Retry-Files	80
5.6	Tags	82
5.7	Das Kommando »ansible-playbook«	83

5.8	Eine exemplarische Apache-Installation	85
5.8.1	Schritt für Schritt	85
5.8.2	Das komplette Playbook	86
5.8.3	»--start-at-task«, »--check«, »--diff«	88
5.9	Handler: Tasks nur bei Changes durchführen	89
5.9.1	Schritt für Schritt	89
5.9.2	Handler	90
5.9.3	Das komplette Playbook bis hierhin	93
6	Playbooks und Tasks: fortgeschrittene Methoden	95
6.1	Variablen	95
6.1.1	Play Vars	95
6.1.2	Extra Vars	96
6.1.3	Präzedenzen	96
6.1.4	»set_fact«	97
6.1.5	»group_vars«	98
6.1.6	»host_vars«	101
6.1.7	»vars_files«: Variablen in beliebigen externen Dateien	101
6.1.8	Prompting	101
6.1.9	Zugriffe auf komplexe Strukturen	102
6.1.10	Seid ihr alle da?	103
6.2	Registrierte Variablen	104
6.3	Facts und implizite Variablen	108
6.3.1	Facts	108
6.3.2	Cachen von Facts	110
6.3.3	Implizite Variablen	111
6.3.4	Ein Beispiel	111
6.3.5	Externe Informationsbeschaffer: »facter« und »ohai«	113
6.3.6	Noch nicht genug Fakten? »/etc/ansible/facts.d«!	113
6.4	Bedingte Ausführung mit »when«	114
6.5	Systemunterschiede ausgleichen – wie denn jetzt?	115
6.5.1	Die plumpe Methode	116
6.5.2	Die solide Methode	116
6.5.3	Die trickreiche Methode	118
6.5.4	Die modulare Methode	119
6.5.5	Das komplette Playbook bis hierhin	120

6.6	Jinja und Templates	123
6.6.1	Begriffsklärung: Templates und Template-Engines	123
6.6.2	Eine individuelle Startseite für unsere Apache-Server	124
6.6.3	Schnelles Testen von Jinja-Templates	126
6.6.4	Jinja-Syntax: Ausgabeausdrücke, Anweisungen, Kommentare	128
6.6.5	Filter	129
6.7	Variablentests	130
6.8	Schleifen	132
6.8.1	Iteration über eine Liste mit »with_items« oder »with_list«	132
6.8.2	Iteration über eine Map mit »with_dict«	135
6.8.3	Iteration über eine generierte Folge mit »with_sequence«	136
6.8.4	Verschachtelte Schleife mit »with_subelements«	136
6.8.5	Tasks wiederholen mit »until«	137
6.8.6	Mehr Kontrolle mit »loop_control«	138
6.8.7	»register« + Schleife	141
6.9	Fehlerbehandlung mit »failed_when« und »ignore_errors«	143
6.10	Blocks (und noch mal Fehlerbehandlung)	144
6.11	Asynchrone Ausführung	146
6.12	Lokale Tasks	148
6.13	Lookup-Plugins	150
6.14	Umgebungsvariablen setzen	152
7	Module und Collections verwenden	155
7.1	Collections	155
7.1.1	Eine Minimalumgebung mit »ansible-core«	155
7.1.2	Der FQCN (Fully Qualified Collection Name)	157
7.1.3	Zwischenfazit	159
7.2	Module	159
7.3	Module zur Kommandoausführung	160
7.4	Module zur Paketverwaltung	161
7.5	Module zur Verwaltung von Dateien und Dateiinhalten	163
7.6	Module für weitere typische Verwaltungsaufgaben	168

7.7	Module zur Interaktion mit Netzwerk-Services	170
7.8	Spezialmodule (Kontrollflusssteuerung etc.)	171
8	Modularisierung mit Rollen und Includes	175
8.1	Erstellung und Verwendung von Rollen	175
8.1.1	Das Rollenkonzept in Ansible	175
8.1.2	Ein einfaches Beispiel für eine Rolle	177
8.1.3	Rollen in einem Playbook verwenden	177
8.1.4	Abhängigkeiten zwischen Rollen	180
8.1.5	Erstellen neuer Rollen mit »ansible-galaxy«	181
8.2	Das Online-Repository Ansible Galaxy	182
8.3	Verwendung von Imports/Includes	182
8.3.1	»import_tasks« und »include_tasks«	183
8.3.2	»include_tasks« und Tags	183
8.3.3	Dynamisches Laden von Variablen mit »include_vars«	185
8.4	Noch mal Apache	186
8.5	Dokumentation	190
8.5.1	Code-Konventionen	190
8.5.2	»README.md«	191
8.6	Wiederverwendung von Rollen	194
9	Webinterfaces	199
9.1	Vorbereitungen zum Betrieb	199
9.2	Ansible Configuration Management Database (ansible-cmdb)	202
9.3	Traefik und Gitea	204
9.3.1	Der Reverse-Proxy Traefik	204
9.3.2	Der Git-Server Gitea	205
9.3.3	IP-Adressen im Inventory	207
9.3.4	Einchecken unseres initialen Projekts	207
9.3.5	»README.md« hinzufügen und nützliche Git-Kommandos	209
9.4	Ansible AWX	210
9.4.1	Test-Setup und erste Anmeldung	210

9.4.2	Exemplarische Verwendung	211
9.4.3	Versionen, Versionen, Versionen	213
9.5	Polemarch	214
9.5.1	Test-Setup und erste Anmeldung	214
9.5.2	Exemplarische Verwendung	215
9.5.3	Fazit	216
9.6	Jenkins	216
9.6.1	Test-Setup	217
9.6.2	Exemplarische Verwendung	217
9.6.3	Fazit	219
9.7	ARA	219
9.7.1	Test-Setup	219
9.7.2	Weitere Möglichkeiten	220
9.8	Weitere, hier nicht näher betrachtete Möglichkeiten	220
9.9	Laborumgebung: nicht mehr benötigte Anwendungen beenden	221
10	Weitere Tools und Techniken	223
10.1	Ansible Vault	223
10.1.1	Vor aller Technik	223
10.1.2	Erste Schritte	225
10.1.3	Mehrere Vault-Passwörter und weitere Vault-Kommandos	227
10.1.4	Ein Trick zum Wiederfinden von Variablen	227
10.1.5	Verschlüsseln einzelner Variablen	228
10.1.6	Mehr Bequemlichkeit bzw. Automatisierbarkeit	229
10.1.7	Bequem und (möglichst) sicher mit GPG und »pass«	230
10.2	Debugging und Troubleshooting	233
10.2.1	Debug-Mode und Verbosity-Level	233
10.2.2	Die Lesbarkeit von Ausgaben verbessern	234
10.2.3	Gathering Facts dauert zu lange	237
10.2.4	Der Playbook-Debugger	238
10.2.5	Statische Code-Analyse mit »ansible-lint«	240
10.2.6	Überwachen von Dateiänderungen mit »--check« und »--diff« ...	243
10.2.7	Last, but not least: das »debug«-Modul	245
10.3	Untersuchen von Konfigurationseinstellungen	247
10.4	Playbooks beschleunigen mit Pipelining	247

10.5	Die sprechende Kuh	248
10.6	Ansible im Pull-Mode	249
10.6.1	»ansible-pull«: Technik und Voraussetzungen	249
10.6.2	Erste Schritte	250
10.6.3	Die ganze Lösung	252
10.6.4	Was fehlt noch?	253
11	Ansible und Docker	255
11.1	Installation von Docker	255
11.2	Docker-Module	257
11.2.1	Vorbereitungen und Vorüberlegungen	257
11.2.2	Überblick	259
11.3	Eine Beispielanwendung	264
11.4	Ansible und Docker Compose	268
11.5	Das »docker«-Connection-Plugin	272
11.6	Erstellen von Images	273
11.6.1	Erstellen von Images mit »docker build«	273
11.6.2	»ansible-bender«	275
11.6.3	Erstellen von Images mit »ansible-bender«	276
11.6.4	Fazit	278
12	Inventory-Management: fortgeschrittene Methoden	281
12.1	Das Kommando »ansible-inventory«	281
12.2	Verschachtelte Gruppen	282
12.3	»On the fly«-Inventorys erstellen mit »add_host«	283
12.4	Dynamische Gruppen mit »group_by«	284
12.5	Dynamische bzw. externe Inventorys	287
12.5.1	Beispiel: ein Inventory-Skript in Perl	288
12.5.2	Verwenden von Inventory-Plugins	291

13 Ansible und die Cloud	293
13.1 Hetzner Cloud	294
13.1.1 Vorbereitungen auf dem Control Host	294
13.1.2 Vorbereitungen in der Cloud	295
13.1.3 Provisionieren von Cloud-Servern	296
13.1.4 Inventarisieren von Cloud-Servern	297
13.1.5 Weitere Möglichkeiten des Inventory-Plugins	298
13.2 AWS EC2	299
13.2.1 Vorbereitungen auf dem Control Host	299
13.2.2 Vorbereitungen in der Cloud	300
13.2.3 Provisionieren von Cloud-Servern	301
13.2.4 Inventarisieren von Cloud-Servern	303
13.2.5 Weitere Möglichkeiten des Inventory-Plugins	304
14 Ansible als Orchestrierungswerkzeug	307
14.1 Administrierst du noch oder orchestrierst du schon?	307
14.2 Viele Target Hosts zum Testen	308
14.3 Die Abarbeitungsreihenfolge beeinflussen	310
14.3.1 »throttle« und »order«	312
14.3.2 »serial«	313
14.3.3 Fehlerhafte Hosts im »serial«-Betrieb	314
14.3.4 Strategy-Plugins	315
14.4 Delegierung	318
15 Ansible und Windows	323
15.1 Ein Control Host auf Windows-Basis	323
15.1.1 Das Windows-Subsystem für Linux (WSL)	324
15.1.2 Cygwin	326
15.2 WinRM	327
15.3 Vorbereitungen auf dem Control Host	328

15.4 Voraussetzungen auf der Windows-Seite und WinRM-Setup	329
15.5 Setup mit Active Directory/Kerberos	330
15.6 WinRM-Troubleshooting	332
15.7 Windows-Module	333
16 Callback-Plugins	337
16.1 Ausgabe-Plugins	337
16.1.1 »default«	338
16.1.2 »yaml«	339
16.1.3 »json«	340
16.1.4 »unixy«	340
16.1.5 »dense«	341
16.1.6 »minimal«	341
16.1.7 »online«	341
16.1.8 »debug«	341
16.1.9 »selective«	342
16.1.10 »counter_enabled«	343
16.2 Sonstige Callback-Plugins	343
17 Eigene Collections und Module erstellen	345
17.1 Namespaces, Namen und Einrichtung eines Collection-Projektes	345
17.2 Playbooks in Collections	347
17.3 Rollen in Collections	348
17.4 Module in Collections	349
17.4.1 Erste Schritte	349
17.4.2 Modul-Parameter	352
17.4.3 Module mit Python – exemplarische Problemstellung	354
17.4.4 Eine exemplarische Lösung	355
17.4.5 Erklärungen und weitere Möglichkeiten	357
17.4.6 Eingebettete Dokumentation	360
17.4.7 Ausblick	361

17.5	Plugins in Collections	362
17.5.1	Ein exemplarisches Callback-Plugin	362
17.5.2	Ausblick	364
17.6	Collections deponieren und installieren	364
18	Entwickeln und Testen mit Molecule	367
18.1	Vorbereitungen und Einrichtung	367
18.2	Erste Schritte	369
18.3	Entwickeln	371
18.4	Testen mit dem Ansible-Verifier	373
18.5	Testen mit dem Testinfra-Verifier	375
18.6	Der komplette Testzyklus	376
18.7	Ausblick und Fazit	376
19	Kochrezepte, Howtos und Best Practices	379
19.1	Eine empfehlenswerte »ansible.cfg«	379
19.2	Ein neues Projekt beginnen	380
19.3	Einen Task in Abhängigkeit von einem vorhergehenden Task ausführen	380
19.4	Einen Task ausführen, wenn der Host in einer bestimmten Gruppe ist	382
19.5	In einer Liste von Maps suchen	382
19.6	Erweiterung von Maps oder Listen während der Laufzeit	383
19.7	Die Elemente einer Liste modifizieren und verbinden	385
19.8	Passwörter und Passwort-Hashes generieren	386
19.9	Einfache Installer bauen	387
19.10	IP-Adresse eines Target Hosts bestimmen	389
19.11	firewalld managen	392
19.12	Linux-Software-Updates einspielen	394
19.13	Ansible über einen Gateway- bzw. Jumphost	397
19.14	Host-spezifische Ressourcen verwalten	397

20	Was könnte noch besser sein bzw. was fehlt noch?	401
20.1	Lange laufende Tasks verfolgen	401
20.2	Abarbeitung einer Rolle beenden	402
20.3	Schleifen über Blöcke	404
20.4	Locking bei konkurrierenden Playbook-Aufrufen	405
20.5	Fazit	406
Anhang	407
A	Projektspezifische Umgebungsvariablen mit »direnv«	409
A.1	Installation und Shellintegration	409
A.2	Verwendung	410
A.3	Aliasse/Funktionen	411
A.4	Kommandoausführung beim Betreten	411
B	SSH (Secure Shell)	413
B.1	Serverseitige Voraussetzungen zur Nutzung von SSH	413
B.2	SSH-Client-Programme	414
B.3	Public-Key-Authentifizierung	415
B.4	SSH-Agenten (Linux-Client)	419
B.5	»PuTTY«, »PuTTYgen« und »Pageant«	420
B.6	WinSCP	422
B.7	SSH: Fortgeschrittene Konfiguration und Nutzung	422
C	Reguläre Ausdrücke	429
C.1	Motivation	429
C.2	Dialekte	430
C.3	Reguläre Ausdrücke – Basics	430
C.4	Reguläre Ausdrücke – erweiterte Möglichkeiten (PCRE)	432
C.5	Reguläre Ausdrücke – erweiterte Möglichkeiten (POSIX Extended RE)	434
Index	437