

Auf einen Blick

1	Einführung und Installation	23
2	Basiseinrichtung und erstes Inventory-Management	41
3	Ad-hoc-Kommandos und Patterns	57
4	Die Konfigurations- und Serialisierungssprache YAML	67
5	Playbooks und Tasks: die Grundlagen	77
6	Playbooks und Tasks: fortgeschrittene Methoden	101
7	Module und Collections verwenden	173
8	Modularisierung mit Rollen und Includes	197
9	Webinterfaces: AWX und mehr	221
10	Weitere Tools und Techniken	243
11	Ansible und Docker	275
12	Inventory-Management: fortgeschrittene Methoden	301
13	Ansible und die Cloud	315
14	Ansible als Orchestrierungswerkzeug	339
15	Ansible und Windows	355
16	Callback-Plugins	369
17	Eigene Collections und Module erstellen	375
18	Entwickeln und Testen mit Molecule	399
19	Kochrezepte, How-tos und Best Practices	411
20	Was könnte noch besser sein, bzw. was fehlt noch?	445

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	28
1.5	Ansible-Installation auf dem Control Host	33
1.6	Installation via PIP (+ Virtualenv)	35
1.7	Authentifizierung und Autorisierung auf den Target Hosts	36
1.8	Einrichten der SSH-Public-Key-Authentifizierung	38
1.9	Ein Ad-hoc-Test ohne jegliche Konfiguration	39
1.10	Noch ein Hinweis zur Migration von älteren Versionen	40

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	45
2.4	Konfigurationseinstellungen vs. Parameter vs. ...?	47
2.5	Inventory-Aliasse und Namensbereiche	48
2.6	Jenseits von Ping	50
2.7	Ein etwas komplexeres Beispiel	52
2.8	Alternative bzw. mehrere Inventorys	53

3	Ad-hoc-Kommandos und Patterns	57
<hr/>		
3.1	Ad-hoc-Kommandos	57
3.2	Use Cases jenseits von »command« und »shell«	59
3.3	Idempotenz	60
3.4	Interne Funktionsweise	61
3.4.1	Parallele Ausführung	61
3.4.2	Persistente Verbindungen	62
3.4.3	Was passiert beim Aufruf eines Moduls?	63
3.5	Die Ansible-Konsole	64
3.6	Patterns zum Adressieren von Hosts	65
4	Die Konfigurations- und Serialisierungssprache YAML	67
<hr/>		
4.1	Syntax und Struktur	67
4.2	YAML-Files editieren	68
4.3	Syntaktische Überprüfung	71
4.4	Listen und Maps	71
4.5	Verschachtelte Strukturen	72
4.6	Textpassagen und Block-Ausdrücke	73
4.7	Das Nichts in YAML	75
4.8	Anchor und References	75
5	Playbooks und Tasks: die Grundlagen	77
<hr/>		
5.1	Hallo Ansible – das allererste Playbook	77
5.2	Formulierung von Tasks	81
5.3	Beenden von Plays	83
5.4	Der problematische Doppelpunkt	84
5.5	Fehlerbehandlung, Retry-Files	85
5.6	Tags	87

5.7	Das Kommando »ansible-playbook«	89
5.8	Eine exemplarische Apache-Installation	90
5.8.1	Schritt für Schritt	90
5.8.2	Das komplette Playbook	92
5.8.3	»--start-at-task«, »--check«, »--diff«	93
5.9	Handler: Tasks nur bei Changes durchführen	94
5.9.1	Schritt für Schritt	94
5.9.2	Handler	96
5.9.3	Das komplette Playbook bis hierhin	99
6	Playbooks und Tasks: fortgeschrittene Methoden	101
6.1	Variablen	101
6.1.1	Play Vars	101
6.1.2	Extra Vars	102
6.1.3	Präzedenzen	103
6.1.4	»set_fact«	104
6.1.5	»group_vars«	104
6.1.6	»host_vars«	107
6.1.7	»vars_files«: Variablen in beliebigen externen Dateien	107
6.1.8	Prompting	107
6.1.9	Zugriffe auf komplexe Strukturen	108
6.1.10	»assert«-Tests	109
6.2	Registrierte Variablen	110
6.3	Facts und implizite Variablen	114
6.3.1	Facts	114
6.3.2	Cachen von Facts	116
6.3.3	Implizite Variablen	117
6.3.4	Ein Beispiel	118
6.3.5	Externe Informationsbeschaffer: »facter« und »ohai«	119
6.3.6	Noch nicht genug Fakten? »/etc/ansible/facts.d«!	119
6.4	Bedingte Ausführung mit »when«	120
6.5	Systemunterschiede ausgleichen – wie denn jetzt?	122
6.5.1	Die plumpe Methode	123
6.5.2	Die solide Methode	124
6.5.3	Die trickreiche Methode	125

- 6.5.4 Die modulare Methode 127
 - 6.5.5 Das komplette Playbook bis hierhin 129
- 6.6 Jinja und Templates 131**
 - 6.6.1 Begriffsklärung: Templates und Template-Engines 131
 - 6.6.2 Eine individuelle Startseite für unsere Apache-Server 132
 - 6.6.3 Schnelles Testen von Jinja-Templates 134
 - 6.6.4 Jinja-Syntax: Ausgabeausdrücke, Anweisungen, Kommentare 136
 - 6.6.5 Filter 138
 - 6.6.6 Whitespace-Kontrolle 139
 - 6.6.7 Macros 141
- 6.7 Variablentests 142**
- 6.8 Lookup-Plugins 144**
- 6.9 Schleifen 146**
 - 6.9.1 Iteration über eine Liste mit »with_items« oder »with_list« 147
 - 6.9.2 Iteration über eine Map mit »with_dict« 150
 - 6.9.3 Iteration über eine generierte Folge mit »with_sequence« 151
 - 6.9.4 Schleife über die Kombination zweier Listen mit »with_nested« 151
 - 6.9.5 Schleife über zwei parallele Listen mit »with_together« 152
 - 6.9.6 Verschachtelte Schleife mit »with_subelements« 153
 - 6.9.7 Tasks wiederholen mit »until« 154
 - 6.9.8 Mehr Kontrolle mit »loop_control« 155
 - 6.9.9 »register« + Schleife 158
- 6.10 Fehlerbehandlung mit »failed_when« und »ignore_errors« 159**
- 6.11 Blöcke 161**
- 6.12 Timeouts und asynchrone Ausführung 162**
- 6.13 Lokale Tasks 166**
- 6.14 Umgebungsvariablen 168**

7 Module und Collections verwenden 173

- 7.1 Collections 173**
 - 7.1.1 Eine Minimalumgebung mit »ansible-core« 174
 - 7.1.2 Collections managen 174
 - 7.1.3 Der FQCN (Fully Qualified Collection Name) 176
 - 7.1.4 Zwischenfazit 177
- 7.2 Module 177**

7.3	Module zur Kommandoausführung	178
7.4	Module zur Paketverwaltung	180
7.5	Module zur Verwaltung von Dateien und Dateiinhalten	182
7.6	Module für weitere typische Verwaltungsaufgaben	187
7.7	Module zur Interaktion mit Netzwerk-Services	190
7.8	Spezialmodule (Kontrollflusssteuerung etc.)	191

8	Modularisierung mit Rollen und Includes	197
----------	--	------------

8.1	Erstellung und Verwendung von Rollen	197
8.1.1	Das Rollenkonzept in Ansible	197
8.1.2	Ein einfaches Beispiel für eine Rolle	199
8.1.3	Rollen in einem Playbook verwenden	199
8.1.4	Plays mit Rollen und Tasks, »pre_tasks« und »post_tasks«	201
8.1.5	Abhängigkeiten zwischen Rollen	202
8.1.6	Wählen anderer Startdateien	203
8.1.7	Erstellen neuer Rollen mit »ansible-galaxy«	203
8.2	Das Online-Repository Ansible Galaxy	204
8.3	Verwendung von Imports/Includes	205
8.3.1	»import_tasks« und »include_tasks«	205
8.3.2	»include_tasks« und Tags	206
8.3.3	Dynamisches Laden von Variablen mit »include_vars«	208
8.3.4	»import_playbook«	209
8.4	Noch mal Apache	209
8.5	Dokumentation (und Konvention)	214
8.5.1	»defaults/main.yml« als Konvention	214
8.5.2	»README.md«	215
8.6	Wiederverwendung von Rollen	217

9	Webinterfaces: AWX und mehr	221
----------	--	------------

9.1	Installation von Python-Paketen auf aktuellen Debian/Ubuntu-Systemen	221
9.2	Ansible Configuration Management Database (ansible-cmdb)	222

9.3	Vorbereitungen zum Betrieb anspruchsvollerer Anwendungen	224
9.4	Der Git-Server Gitea	228
9.4.1	Inbetriebnahme und erste Anmeldung	228
9.4.2	Einchecken unseres initialen Projekts	228
9.4.3	»README.md« hinzufügen und nützliche Git-Kommandos	230
9.5	AWX	232
9.5.1	Inbetriebnahme und erste Anmeldung	232
9.5.2	Exemplarische Verwendung	233
9.5.3	Execution Environments	235
9.5.4	Fazit	237
9.6	ARA	238
9.6.1	Test-Setup	238
9.6.2	Weitere Möglichkeiten	239
9.7	Weitere, hier nicht näher betrachtete Anwendungen	240
9.7.1	Semaphore	240
9.7.2	Polemarch	241
9.7.3	Jenkins	241
9.7.4	Rundeck	242
9.8	Nicht mehr benötigte Anwendungen beenden oder löschen	242

10	Weitere Tools und Techniken	243
-----------	--	------------

10.1	Ansible Vault	243
10.1.1	Vor aller Technik	243
10.1.2	Erste Schritte	245
10.1.3	Bedeutung der Vault-ID	246
10.1.4	Weitere Vault-Kommandos	247
10.1.5	Ein Trick zum Wiederfinden von Variablen	248
10.1.6	Verschlüsseln einzelner Variablen	248
10.1.7	Mehr Bequemlichkeit bzw. Automatisierbarkeit	250
10.1.8	Bequem und relativ sicher mit einem Passwort-Client-Skript	251
10.1.9	Bequem und (möglichst) sicher mit GPG + pass	252
10.2	Debugging und Troubleshooting	253
10.2.1	Debug-Mode und Verbosity-Level	253
10.2.2	Die Lesbarkeit von Ausgaben verbessern	255
10.2.3	Gathering Facts dauert zu lange	257
10.2.4	Der Playbook-Debugger	258
10.2.5	Statische Codeanalyse mit »ansible-lint«	261

10.2.6	Check-Mode und Diff-Mode	263
10.2.7	Last, but not least: das »debug«-Modul	266
10.3	Playbooks beschleunigen mit Pipelining	267
10.4	Die sprechende Kuh	268
10.5	Ansible im Pull-Mode	269
10.5.1	»ansible-pull«: Technik und Voraussetzungen	270
10.5.2	Erste Schritte	271
10.5.3	Die ganze Lösung	272
10.5.4	Was fehlt eventuell noch?	273

11

Ansible und Docker

275

11.1	Installation von Docker	275
11.2	Docker-Module	277
11.2.1	Vorbereitungen und Vorüberlegungen	277
11.2.2	Ein erstes einfaches Beispiel	278
11.2.3	Überblick	279
11.3	Eine Beispielanwendung	285
11.4	Ansible und Docker Compose	289
11.5	Das »docker«-Connection-Plugin	293
11.6	Erstellen von Images	294
11.6.1	Erstellen von Images mit »docker build«	295
11.6.2	»ansible-bender«	296
11.6.3	Erstellen von Images mit »ansible-bender«	297
11.6.4	Fazit	300

12

**Inventory-Management:
fortgeschrittene Methoden**

301

12.1	Das Kommando »ansible-inventory«	301
12.2	Verschachtelte Gruppen	302
12.3	Statische Inventorys im YAML-Format	303
12.4	»On the fly«-Inventorys erstellen mit »add_host«	306
12.5	Dynamische Gruppen mit »group_by«	307

12.6	Dynamische bzw. externe Inventorys	311
12.6.1	Inventory-Skripte	311
12.6.2	Verwenden von Inventory-Plugins	314

13	Ansible und die Cloud	315
-----------	------------------------------------	------------

13.1	Versionsprobleme und Virtualenv	316
13.2	Wohin mit Keys, Tokens, Secrets etc.?	316
13.3	Hetzner Cloud	317
13.3.1	Vorbereitungen auf dem Control Host	317
13.3.2	Vorbereitungen in der Cloud	318
13.3.3	Verwenden von Cloud-Modulen	319
13.3.4	Provisionieren von Cloud-Servern	321
13.3.5	Inventarisieren von Cloud-Servern	322
13.3.6	Weitere Möglichkeiten des Inventory-Plugins	323
13.4	AWS EC2	324
13.4.1	Vorbereitungen auf dem Control Host	324
13.4.2	Vorbereitungen in der Cloud	325
13.4.3	Verwenden von Cloud-Modulen	326
13.4.4	Provisionieren von Cloud-Servern	327
13.4.5	Inventarisieren von Cloud-Servern	329
13.4.6	Weitere Möglichkeiten des Inventory-Plugins	330
13.5	Proxmox VE	331
13.5.1	Vorbereitungen auf dem Control Host	332
13.5.2	Vorbereitungen in Proxmox VE	332
13.5.3	Ein erster Test	333
13.5.4	Provisionieren von VMs	334
13.5.5	Ausblick	337

14	Ansible als Orchestrierungswerkzeug	339
-----------	--	------------

14.1	Administrierst du noch, oder orchestrierst du schon?	339
14.2	Viele Target Hosts zum Testen	340
14.3	Die Abarbeitungsreihenfolge beeinflussen	341
14.3.1	»throttle« und »order«	343
14.3.2	»serial«	344

	Inhalt
14.3.3 Fehlerhafte Hosts im »serial«-Betrieb	345
14.3.4 Strategy-Plugins	347
14.4 Delegierung	350
15 Ansible und Windows	355
15.1 Ein Control Host auf Windows-Basis	355
15.1.1 Das Windows-Subsystem für Linux (WSL)	356
15.1.2 Cygwin	357
15.2 Windows-Targets und WinRM	359
15.3 Vorbereitungen auf dem Control Host	360
15.4 Voraussetzungen auf der Windows-Seite und WinRM-Setup	361
15.5 WinRM-Troubleshooting	362
15.6 Setup mit Active Directory/Kerberos	363
15.7 Windows-Module	365
16 Callback-Plugins	369
16.1 Stdout Callback Plugins	369
16.2 Aggregate und Notification Callback Plugins	371
17 Eigene Collections und Module erstellen	375
17.1 Namespaces, Namen und Einrichtung eines Collection-Projekts	375
17.2 Playbooks in Collections	377
17.3 Rollen in Collections	378
17.4 Module in Collections	379
17.4.1 Erste Schritte	380
17.4.2 Modulparameter	382
17.4.3 Module mit Python – exemplarische Problemstellung	385
17.4.4 Eine exemplarische Lösung	386
17.4.5 Erklärungen und weitere Möglichkeiten	388

Inhalt	
17.4.6	Eingebettete Dokumentation 391
17.4.7	Ausblick 392
17.5	Plugins in Collections 393
17.5.1	Ein exemplarisches Callback-Plugin 393
17.5.2	Ausblick 395
17.6	Collections deponieren und installieren 395
18	Entwickeln und Testen mit Molecule 399
18.1	Vorbereitungen und Einrichtung 399
18.2	Erste Schritte 401
18.3	Entwickeln 403
18.4	Testen mit dem Ansible-Verifier 405
18.5	Testen mit dem Testinfra-Verifier 407
18.6	Der komplette Testzyklus 409
18.7	Ausblick und Fazit 409
19	Kochrezepte, How-tos und Best Practices 411
19.1	Neue Projekte 411
19.1.1	Eine empfehlenswerte »ansible.cfg« 411
19.1.2	Eine Vorlage für ein neues Projekt 412
19.2	Administratives 413
19.2.1	Einfache Installer bauen 413
19.2.2	IP-Adresse eines Target Hosts bestimmen 415
19.2.3	»Firewalld« managen 418
19.2.4	Linux-Software-Updates einspielen 419
19.2.5	Initiales Verteilen von SSH-Keys 421
19.2.6	Passwörter auf der Kommandozeile übergeben 422
19.2.7	Ansible über einen Gateway- bzw. Jumphost 424
19.3	Jinja-Magie 424
19.3.1	Erweiterung von Maps oder Listen während der Laufzeit 424
19.3.2	Die Elemente einer Liste modifizieren und verbinden 426
19.3.3	In einer Liste von Maps suchen 427
19.3.4	Ein Attribut aus einer Liste von Maps filtern 428

	Inhalt
19.3.5	Aus Zielsystemfakten einen Report generieren 429
19.3.6	Passwörter und Passwort-Hashes generieren 430
19.4	Tasks und Kontrollfluss 432
19.4.1	Einen Task in Abhängigkeit von einem vorhergehenden Task ausführen 432
19.4.2	Einen Task ausführen, wenn der Host in einer bestimmten Gruppe ist 433
19.4.3	Redundante Modulparameter vermeiden mit »module_defaults« ... 433
19.4.4	Play-Hosts dynamisch festlegen 435
19.4.5	Lesen von Konfigurationsdateien 436
19.5	Sonstiges 439
19.5.1	Funktionen simulieren 439
19.5.2	Host-spezifische Ressourcen verwalten 441
20	Was könnte noch besser sein, bzw. was fehlt noch? 445
20.1	Lange laufende Tasks verfolgen 445
20.2	Abarbeitung einer Rolle beenden 446
20.3	Schleifen über Blöcke 448
20.4	Locking bei konkurrierenden Playbook-Aufrufen 449
20.5	Fazit 450
Anhang	453
A	Projektspezifische Umgebungsvariablen mit »direnv« 453
B	Der Passwortmanager »pass« 457
C	SSH (Secure Shell) 461
D	Reguläre Ausdrücke 479
E	»vim« und »nano«: Tipps und Tricks 487
Index 491