Ansible-Playbooks für Debian-Systeme

Rollenfindung

Hartmut Goebel

Konfiguriert der Administrator seine Systeme mit Ansible, benötigt er passende Rollen für alle Fälle. Das Projekt DebOps stellt sie für Debian-Linux bereit.

Wer Server nicht individuell einrichten und pflegen will, automatisiert diese Arbeit mit einem Konfigurationsmanagement-Tool wie Ansible [1], Puppet oder Chef. Allerdings findet man sich schnell mit der Frage konfrontiert, woher man Rezepte (oder Rollen, wie die Konfigurationsdateien bei Ansible heißen) bekommt, die zusammenpassen. Will der Administrator eine Rolle etwa von der Plattform Ansible Galaxy übernehmen, muss er meist einiges ändern und die Templates anpassen. Das ist nicht nur umständlich, sondern erschwert zudem Updates – falls es für die Rolle überhaupt jemals welche gibt.

Administratoren benötigen jedoch mächtige und flexible Rollen, die kombiniert werden können. Damit lässt sich etwa ein Webserver mit TLS-Verschlüsselung, ownCloud und dem Online-Editor Etherpad lite nach Anpassen einiger Konfigurationsvariablen in 10 Minuten einrichten, inklusive Mail-Forwarding und Zeiteinstellung per NTP.

Das Projekt DebOps (debops.org) bietet eine Sammlung von inzwischen mehr als 90 aufeinander abgestimmten Rollen für Debian-Linux und verwandte Distributionen. Um beispielsweise Gitlab auf einem DepOps-Server zu installieren, genügt es, ihn im Inventory in die Gruppe *debops_gitlab* aufzunehmen:

```
[debops_gitlab]
my-server42
```

Anschließend ruft man das Kommando *debops* auf. Übergibt man kein eigenes Playbook als Konfigurationsskript, verwendet DebOps ein Standard-Playbook, das solche Gruppenzugehörigkeiten für Rollen auswertet.

Alternativ definiert man die Rollen in einem eigenen Playbook und übergibt dort die gewünschten Einstellungen. Das folgende Playbook *owncloud.yml* beispielsweise installiert eine ownCloud-Instanz mit SQLite-Datenbank, die unter den Namen cloud.heise.de und cloud.ix.de erreichbar ist:

```
- name: Manage my-server23
  hosts: my-server23
  sudo: True
  roles:
  - role: debops.owncloud
    tags: owncloud
    owncloud_domain:
    - cloud.heise.de
    - cloud.ix.de
    owncloud_data_path: /var/www/owncloud/data
    owncloud_release: 8.1
    owncloud_database: sqlite
    owncloud_autosetup: True
```

Wichtige Rollen

Anwendungen: *boxbackup, dokuwiki, etherpad lite*, Gitlab, Gitlab CI, *IPXE, LibreNMS, Mailman, ownCloud, phpIPAM*, Debian Preseeding, *rstudio_server, sks* (GPG key-server)

Hardware: *gru, hwraid*

Netzdienste: *dhcpd, dnsmasq, radvd, stunnel, subnetwork, tinc*

Services: *ferm* (*iptables*-Firewall), *ntp, rsyslog, sshd, tcpwrappers*

Serverdienste: *mariadb, postgresql, fail2ban, nginx, salt, tgt, docker_gen, samba, postfix, dovecot, memcached, monkeysphere, nodejs, smstools, elasticsearch, monit, redis, reprepro, snmpd, tftpd*

System: *nfs, iscsi, slapd, lvm, swapfile*, Backup per *rsnapshot*

Virtualisierung: *docker, libvirt, lxc, openvz*

Der Befehl *debops owncloud.yml* richtet alles Nötige dafür ein.

Rollen und Parameter bei DebOps sind durchweg gut dokumentiert. Für alles gibt es sinnvolle Default-Werte, die sich im Playbook oder im Inventory ändern lassen. Passwörter, private Schlüssel und TLS-Zertifikate werden bei Bedarf automatisch erzeugt und in einem Container des verschlüsselten Dateisystems EncFS auf dem Administrationssystem gespeichert.

Ein Webserver wird standardmäßig für TLS und automatisches Weiterleiten von HTTP auf HTTPS konfiguriert. Die Kryptoparameter entsprechen den aktuellen Empfehlungen (starke Algorithmen, Perfect Forward Secrecy). Öffentliche SSH-Schlüssel werden ebenso verteilt wie Hostkeys. Außerdem verwaltet DebOps, ob einem Benutzer etwa nur *git* oder *sftp* erlaubt ist. Die benötigten Schlüssel erzeugt das System bei Bedarf.

Das DebOps-Projekt legt einige Beschränkungen fest, um die Rollensammlung mit vertretbarem Aufwand zu pflegen. So werden ausschließlich Debian-basierte Distributionen unterstützt, als Mailserver ist Postfix mit Dovecot und als Webserver nginx vorgegeben. Bei den Datenbanken hingegen besteht die Wahl zwischen MariaDB, PostgreSQL und SQLite.

Wer Rollen probieren, jedoch keine virtuelle Umgebung einrichten will, dem bietet DebOps Beispiel-Setups für virtuelle Maschinen und Docker-Container. Die Setups sind mit Vagrant zu verwalten, gut dokumentiert und darauf ausgelegt, Playbooks des Administrators auszuführen.

Für das automatisierte Testen von Rollen existiert das Tool *rolespec*, das auch in den CI-Tests (Continuous Integration) des DebOps-Projekts genutzt wird. Ein Tipp für Puppet-Benutzer: Der Schweizer Non-Profit-Provider immerda.ch veröffentlicht eine Sammlung von Puppet-Rezepten bei GitHub, die Ähnliches wie DebOps für Puppet leisten dürfte (tiw)

Hartmut Goebel

ist CISSP, CSSLP, selbstständig und berät zum Management von IT-Sicherheit.

Literatur

[1] Sebastian Meyer, André Nähring; Systemverwaltung; Gut abgemischt; Konfigurationsmanagement ohne Client-Installation; *iX* 10/2014, S. 72

Alle Links: www.ix.de/ix1601144