Advanced Ansible Workshop

Jens Kubieziel, Andreas Scherbaum

12. März 2023

Agenda

1. Einleitung
2. Einführung in Ansible und in den Workshop
3. Erste Schritte mit Ansible
   - Vorarbeiten
4. Inventory und Variablen
   - Hostvariablen
5. Rollen

Organisatorisches

- Geplante Dauer: 3 Stunden
- Nach einer kurzen Einführung gibt es Übungen.
- Wir stellen euch AWS-Instanzen zur Verfügung.

Organisatorisches

- Geplante Dauer: 3 Stunden
- Nach einer kurzen Einführung gibt es Übungen.
- Wir stellen euch AWS-Instanzen zur Verfügung.

Ziel: Betrieb einer kleinen PHP-Anwendung mit Web- und Datenbankserver verwaltet über Ansible

Wir

Kurze Vorstellung

- Jens Kubieziel
- Andreas Scherbaum

Wir

Kurze Vorstellung – Jens Kubieziel

- Arbeitet für OctoPi.Consulting im Schwerpunkt Datenschutz und Informationssicherheit
- Ehrenamtlich für Projekte aus dem Tor-Umfeld und im Hackspace Jena e. V.
- erreichbar über verschiedene Wege: `https://kubieziel.de/blog/archives/1685-Kommunikationswerkzeuge-im-Jahr-2022.html`; am liebsten per Matrix: `@qbi:matrix.kraut.space`

Wir

Kurze Vorstellung – Andreas Scherbaum

- Arbeitet für Adjust GmbH
- Arbeitet mit vielen PostgreSQL Datenbanken
- Board of Directors: PostgreSQL Europe
- Organisiert eine Reihe Konferenzen: pgconf.de, pgconf.eu, FOSDEM PGDay
- Blog: `https://andreas.scherbaum.la/`

Agenda

Warum Ansible?

Was ist Ansible?

Details

- leichtgewichtiges Werkzeug zur Automatisierung von Administrationsaufgaben

Was ist Ansible?

Details

- leichtgewichtiges Werkzeug zur Automatisierung von Administrationsaufgaben
- Freie Software

Was ist Ansible?

Details

- leichtgewichtiges Werkzeug zur Automatisierung von Administrationsaufgaben
- Freie Software
- in Python entwickelt

Wie funktioniert Ansible?

Ihr beschreibt den gewünschten Zustand der Zielsysteme. Ansible loggt sich per SSH ein und führt ggf. notwendige Aktionen aus.

Wie funktioniert Ansible?

Voraussetzungen

- SSH
- Python (Version 2.6 bzw. Python 3)

Wie installiere ich Ansible?

- Über die Paketverwaltung deines GNU/Linux-Systems:
  - `apt install ansible`
  - Unter Ubuntu gibt es ein PPA: `apt-add-repository ppa:ansible/ansible`
  - `yum install ansible` (ggf. das EPEL-Repository aktivieren)
  - `emerge -av app-admin/ansible`

Wie installiere ich Ansible?

- Über die Paketverwaltung deines GNU/Linux-Systems:
  - `apt install ansible`
  - Unter Ubuntu gibt es ein PPA: `apt-add-repository ppa:ansible/ansible`
  - `yum install ansible` (ggf. das EPEL-Repository aktivieren)
  - `emerge -av app-admin/ansible`
- Aus den Quellen:
  - `pip install ansible`
  - `tar.gz` von `https://github.com/ansible/ansible/releases`

Agenda

Unser Repository

Für den Workshop haben wir ein Repository vorbereitet:

`https://github.com/andreasscherbaum/ansible-workshop-clt-2023`

Unser Repository

Für den Workshop haben wir ein Repository vorbereitet:

`https://github.com/andreasscherbaum/ansible-workshop-clt-2023`

Bitte clont dieses:

```
git clone https://github.com/andreasscherbaum/
        ansible-workshop-clt-2023.git
```

Unser Repository

Für den Workshop haben wir ein Repository vorbereitet:

`https://github.com/andreasscherbaum/ansible-workshop-clt-2023`

Bitte clont dieses:

```
git clone https://github.com/andreasscherbaum/
    ansible-workshop-clt-2023.git
```

Die Übungen befinden sich im Unterverzeichnis uebungen/.

Vorarbeiten

Mit den folgenden Übungen sollen die Maschinen in eine Art Anfangszustand versetzt werden.

- Übung 1 Könnt ihr euch überhaupt einloggen?
- Übung 2 Funktionieren Ad-Hoc-Befehle?
- Übung 3 Web- und Datenbankserver installieren und konfigurieren.

Übung 1

Bevor wir mit Ansible loslegen, wollen wir wissen, ob ihr euch auf den Maschinen einloggen könnt.

Führt die Übung 1 im Verzeichnis uebungen/01-ssh aus.

Übung 2

Mit dem ersten Ansible-Kommando wollen wir die Maschinen anpingen. Bekommt ihr alle Kontakt zu den Maschinen?

Ad-Hoc-Modus zur Erinnerung

Mit dem Aufruf von Ansible auf der Kommandozeile lassen sich verschiedene Befehle mitgeben. Diese landen eventuell in der Shellhistory, sind aber ansonsten nirgendwo hinterlegt. Diese Art von Aufruf wird als *Ad-Hoc-Modus* bezeichnet.

Ad-Hoc-Modus zur Erinnerung

Mit dem Aufruf von Ansible auf der Kommandozeile lassen sich verschiedene Befehle mitgeben. Diese landen eventuell in der Shellhistory, sind aber ansonsten nirgendwo hinterlegt. Diese Art von Aufruf wird als *Ad-Hoc-Modus* bezeichnet.
Der Aufruf enthält die betreffenden Hosts sowie Optionen:

- `-i` bezeichnet den Ort des Inventorys
- `-m` Modul, welches ausgeführt werden soll (z. B. shell)
- `-a` Argumente zum obigen Modul bzw. Shell-Kommando (command-Modul)
- `-u` Benutzername (Standard: aktueller Benutzername)
- `-b` Aktionen werden mit den Rechten des angegebenen Benutzers ausgeführt

Ad-Hoc-Modus zur Erinnerung

Mit dem Aufruf von Ansible auf der Kommandozeile lassen sich verschiedene Befehle mitgeben. Diese landen eventuell in der Shellhistory, sind aber ansonsten nirgendwo hinterlegt. Diese Art von Aufruf wird als *Ad-Hoc-Modus* bezeichnet.
Der Aufruf enthält die betreffenden Hosts sowie Optionen:

- `-i` bezeichnet den Ort des Inventorys
- `-m` Modul, welches ausgeführt werden soll (z. B. shell)
- `-a` Argumente zum obigen Modul bzw. Shell-Kommando (command-Modul)
- `-u` Benutzername (Standard: aktueller Benutzername)
- `-b` Aktionen werden mit den Rechten des angegebenen Benutzers ausgeführt

Hello World

```
ansible all -i hosts -a '/bin/echo Hello World'
```

Übung 2

Für Ansible gibt es das Modul ping, welches einen oder mehrere Hosts kontaktiert und das Ergebnis zurückmeldet.

Führt die Übung 2 im Verzeichnis uebungen/`02-ping` aus.

Übung 3

Nun werden Web- und Datenbankserver installiert und konfiguriert. Diese Übung schließt an die Workshops der Vorjahre an und nutzt dies als Anfangszustand.

Übung 3

Nun werden Web- und Datenbankserver installiert und konfiguriert. Diese Übung schließt an die Workshops der Vorjahre an und nutzt dies als Anfangszustand.
Ihr bekommt eine Ausgabe der Form:

Die URL lautet: http://xxx.xxx.xxx.xxx/index.php

Damit könnt ihr eine kleine Infoseite sehen.

Agenda

Inventory

Das Inventory sammelt die diversen Systeme und besteht aus einer oder mehreren Dateien:

hosts

```
192.168.23.42
clt.20.example.org
[webservers]
192.168.17.189
clt.20.example.org
```

Inventory

Format

Das Inventory kann im INI-Format vorliegen

Inventory

Format

Das Inventory kann im INI-Format vorliegen

hosts als INI

```
192.168.23.42
clt.20.example.org
[webservers]
192.168.17.189
clt.20.example.org
```

Inventory

Format

Das Inventory kann im INI-Format vorliegen oder als YAML-Datei:

hosts als YAML

```
all:
  hosts:
    192.168.23.42
    clt.20.example.org
  children:
    webservers:
      clt.20.example.org:
```

Standardgruppen

Ansible legt standardmäßig zwei Gruppen an:

- `all`

Standardgruppen

Ansible legt standardmäßig zwei Gruppen an:

- `all`
- `ungrouped`

Standardgruppen

Ansible legt standardmäßig zwei Gruppen an:

- `all`
- `ungrouped`

Standardgruppen

Ansible legt standardmäßig zwei Gruppen an:

- `all`
- `ungrouped`

Weitere Gruppen könnt ihr selbst festlegen und die Hosts dort oder in Kindgruppen einsortieren.

Dynamische Inventorys

Die Inventorys sind nicht unbedingt statisch. Neue Host kommen hinzu, alte werden aus dem System genommen. Je nach Umgebung kann dies sehr dynamisch passieren.

Dynamische Inventorys

Die Inventorys sind nicht unbedingt statisch. Neue Host kommen hinzu, alte werden aus dem System genommen. Je nach Umgebung kann dies sehr dynamisch passieren.
Ansible setzt hier mittels Inventoryplugins oder -skripten an: `ansible-doc -t inventory -l`.
Damit werden die Inventorys dynamisch aufgebaut und mit Ansible benutzt.

Variablen

Zwischen den Einzelsystemen kann es Unterschiede geben. Ansible nutzt hier *Variablen*, um diese Unterschiede zu verwalten.

Variablen

Zwischen den Einzelsystemen kann es Unterschiede geben. Ansible nutzt hier *Variablen*, um diese Unterschiede zu verwalten.
Diese Variablen können

- auf der Kommandozeile übergeben,
- in einer Datei (Playbook, Inventory etc.) gespeichert oder
- als Rückgabewert ausgewertet werden.

Einfache Variablen

Ein Variablenname kann Buchstaben, Zahlen und Unterstriche enthalten und kann mittels YAML folgendermaßen festgelegt werden:

```
http_port: 80
```

Einfache Variablen

Ein Variablenname kann Buchstaben, Zahlen und Unterstriche enthalten und kann mittels YAML folgendermaßen festgelegt werden:

```
http_port: 80
```

Unter anderem im Playbook kann der Zugriff erfolgen:

```
port: '{{ http_port }}'
```

Inventoryvariablen

Die Variablen können auch für Gruppen innerhalb von Repositorys verwendet werden.

hosts

```
[webservers]
192.168.17.189
clt.20.example.org
[webservers:vars]
http_port=8080
```

Übung 4

In der Übung 4 erweitert ihr das Inventory um einen Eintrag und legt eine Variable namens year an. Diese wird für die PHP-Datei genutzt.

Gruppenvariablen

Innerhalb des Playbooks kann es ein Verzeichnis namens `group_vars` geben. Dort werden in Dateien Variablen für eine Gruppe von Hosts in YAML-Syntax hinterlegt.

Übung 5

In der Übung 5 legen wir das richtige Verzeichnis an und speichern dort eine Datei namens `webservers.yml`. In der Datei ist u. a. wieder die Variable year festgelegt.

Hostvariablen

Im Inventory können pro Host Variablen festgelegt werden. Diese können später wieder im Playbook verwendet werden.

hosts

```
[webservers]
192.168.17.189 http_port=80
clt.20.example.org http_port=8080
```

Übung 6

In der Übung 6 erweitern wir das Inventory um die Variable year.

Extravariablen

Schließlich könnt ihr auch auf der Kommandozeile Variablen übergeben: `ansible-playbook -e "foo=bar"` oder `ansible-playbook --extra-vars "foo=bar"`

Übung 7

In der Übung 7 übergeben wir die Variable year über die Kommandozeile.

Gruppierung

Ansible erlaubt es, das Inventory zu strukturieren. Sinnvoll ist es, sich Gedanken um

Was	Anwendungen, Microservices etc.
Wo	Raum, Gebäude, Rechenzentrum etc.
Wann	Test, Staging, Produktion etc.

Gruppierung

Im Inventory könnt ihr Gruppen anlegen und diese dann über Metagruppen zusammenfassen. So könntet ihr die Gruppen `dbservers` und `webservers` haben. Diese könnten dann beispielsweise in die Metagruppe `rechenzentrum` zusammengefasst werden:

```
[rechenzentrum:children]
webservers
dbservers
```

Übung 8

In der Übung 8 praktizieren wir das mal, in dem wir eine neue Gruppe namens workshop anlegen und dort unsere Server mit reinpacken.

Agenda

1. Einleitung
2. Einführung in Ansible und in den Workshop
3. Erste Schritte mit Ansible
   - Vorarbeiten
4. Inventory und Variablen
   - Hostvariablen
5. Rollen

Rollen

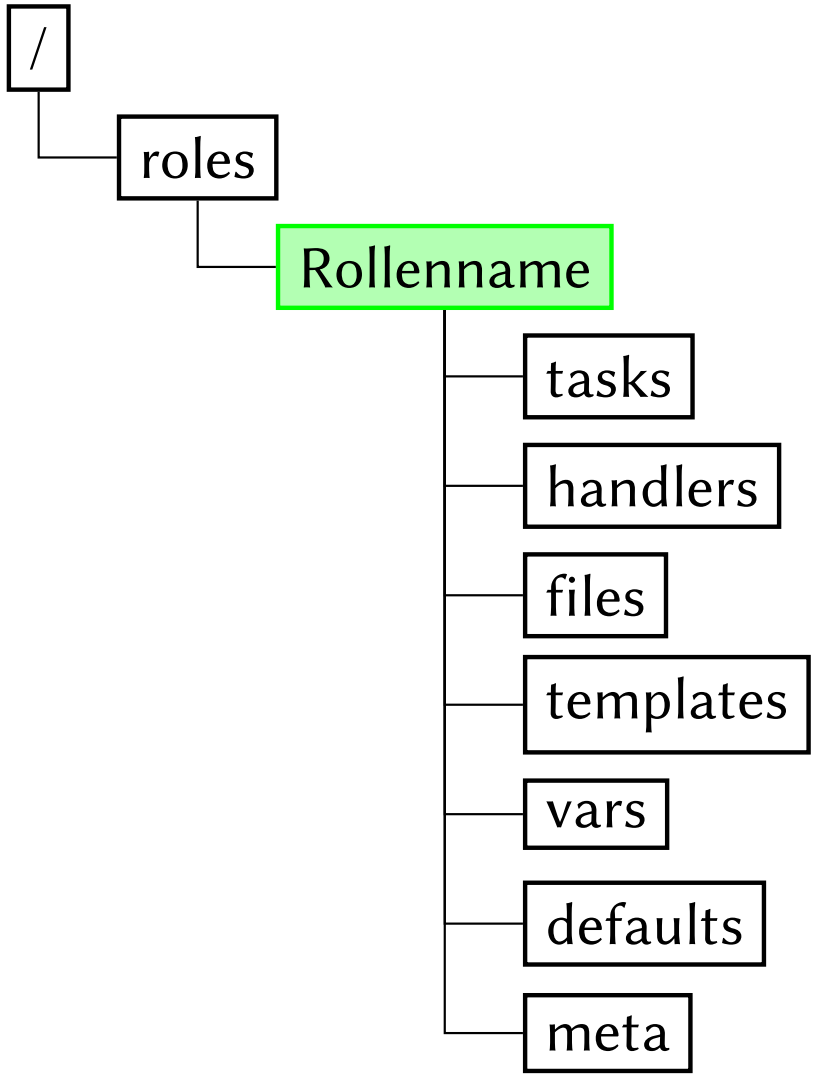
Einführung

Irgendwann kommt der Punkt, wo die Arbeit besser organisiert werden soll. Denn in der Regel sollen viele kleine Aufgaben ausgeführt werden statt einer großen.
Seit Ansible 2.4 gibt es die Möglichkeit, Inhalte anderer Dateien einzubinden (`import` und `include`).
Rollen sind ein älteres Mittel. Diese greifen auf eine vordefinierte Verzeichnisstruktur zurück und können Tasks ausführen, auf Variablen zugreifen etc.

Rollen

Verzeichnisstruktur

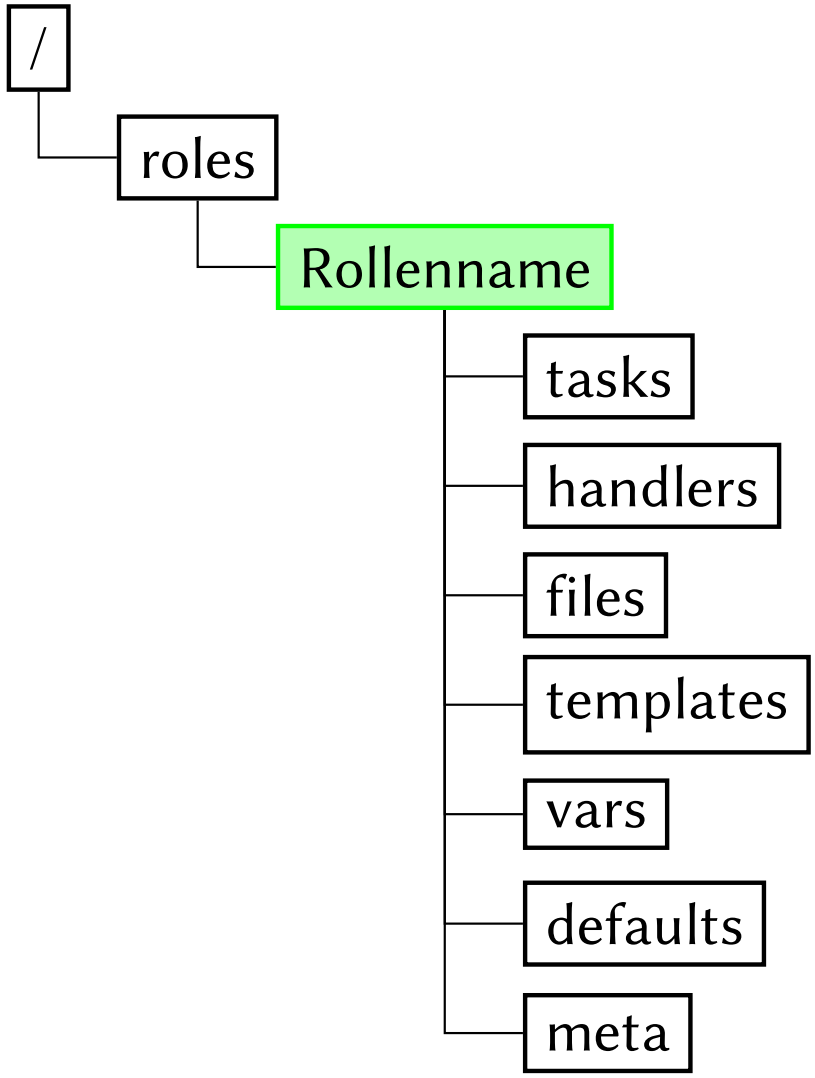
- Mindestens eines der Verzeichnisse muss existieren



Rollen

Verzeichnisstruktur

- Mindestens eines der Verzeichnisse muss existieren
- Die existierenden Verzeichnisse müssen eine Datei mit dem Namen `main.yml` enthalten.



Rollen

Inhalt der Verzeichnisse

tasks enthalten die Liste an Tasks, die durch die Rolle ausgeführt wird

handlers Handler, die durch die Rolle benutzt werden

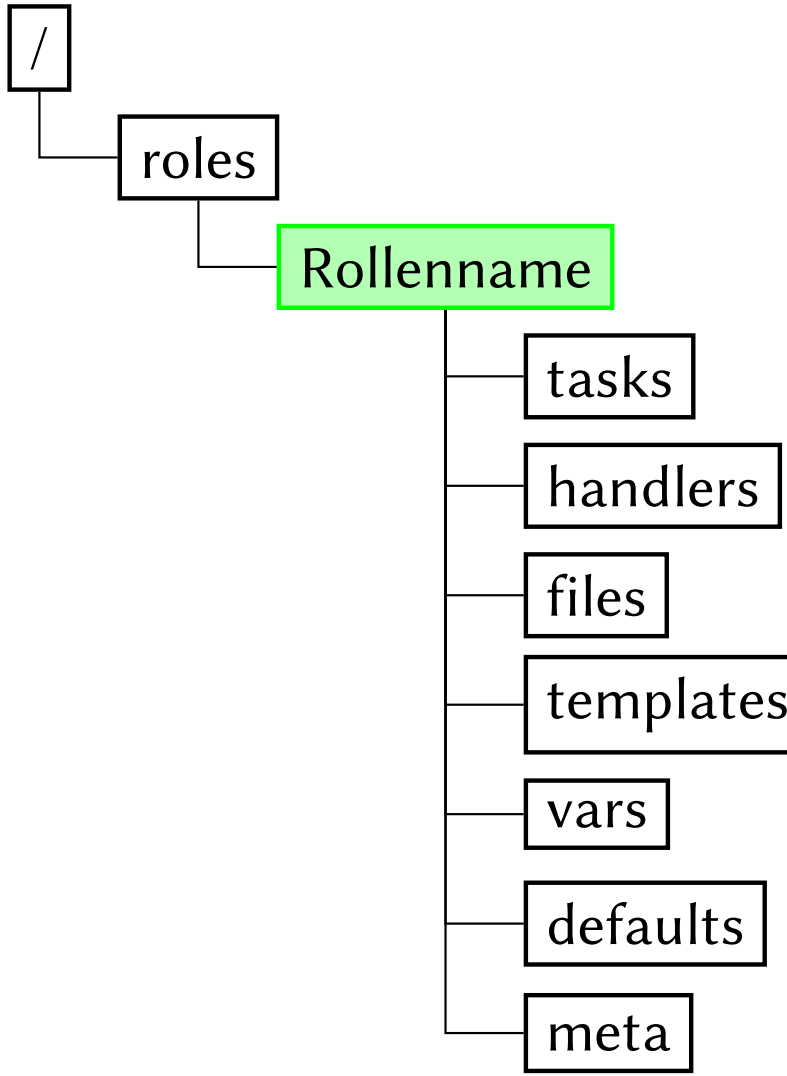
files Dateien, die von dieser Rolle benutzt werden

templates Templates, die dann deployt werden

vars Variablen für die Rolle

defaults Standardwerte für Variablen

meta Abhängigkeiten der Rolle



Agenda

1. Einleitung
2. Einführung in Ansible und in den Workshop
3. Erste Schritte mit Ansible
   - Vorarbeiten
4. Inventory und Variablen
   - Hostvariablen
5. Rollen

Templates

Neben dem Kopieren einfacher Dateien können wir auch Templates anfertigen. Ansible baut daraus die korrekte Datei und lädt diese ins Zielsystem. Die Basis für die Templates ist Jinja2.

Aufruf mit Template-Modul

```
- name: Konfiguration
  template: src=config.j2 dest=/etc/programm/tor.{{item.
      host}}.conf
  with_items: {{ ipadressen }}
```

Jinja2

Die Template-Sprache Jinja2 kommt aus dem Dunstkreis von Python und funktioniert mit aktuellen Version (2.6.x, 2.7.x, ab 3.3.x) der Sprache.

Jinja2

Variablen

Im Unterordner `vars` können Variablen in die Datei `main.yml` eingebaut werden. Auf diese greift die Template-Datei zu und fügt die Werte ein.

```
ipadressen:
  - host: Bridge1
    ip: 192.168.192.23
    port: 12345
  - host: Bridge2
    ip: 192.168.192.42
    port: 4521
```