Konfigurationsmanagement über verschiedene Netze mit Ansible und AWX

Dr. Ottavia Balducci, ATIX AG

Chemnitzer Linux-Tage, 11. März 2023

Inhalt

Wer bin ich?

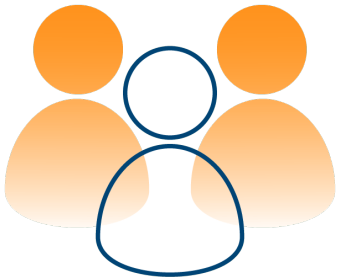
- Ottavia Balducci
- IT-Beraterin bei ATIX AG

Mein Fokus:

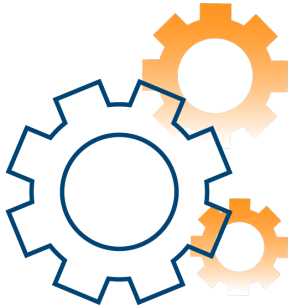
- Ansible
- AWX/AAP
- orcharhino/Foreman

- Email: `balducci@atix.de`

THE Linux & Open Source Company!



Consulting



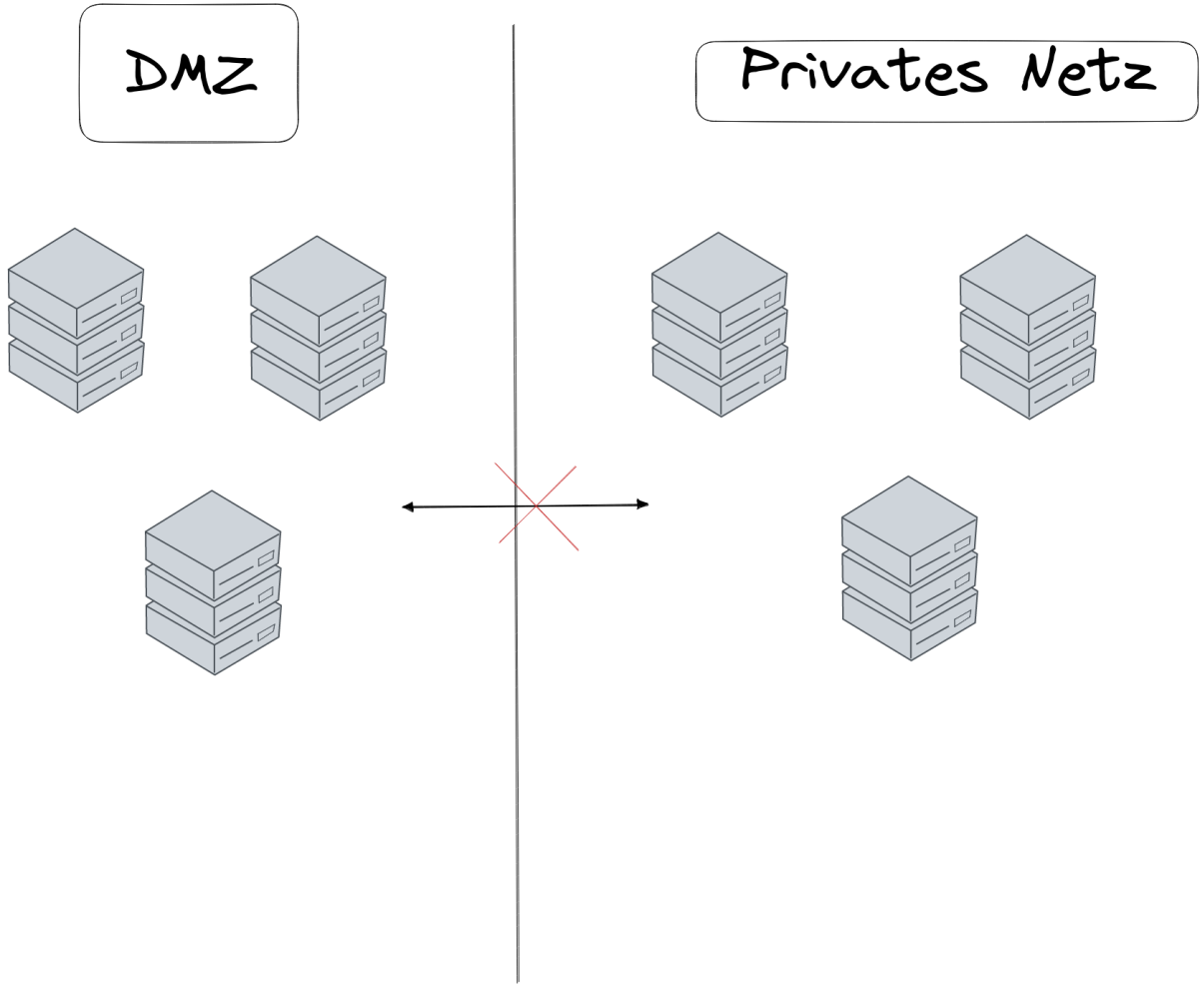
Engineering

Support

Training

Das Ziel

1. Wir möchten Hosts mit Ansible konfigurieren
2. Wir möchten die Gelegenheit haben, diese Hosts aus einer GUI zu konfigureren $\Rightarrow$ AWX
3. Wir möchten das auch über verschiedene Netze machen können


DMZ
Privates Netz

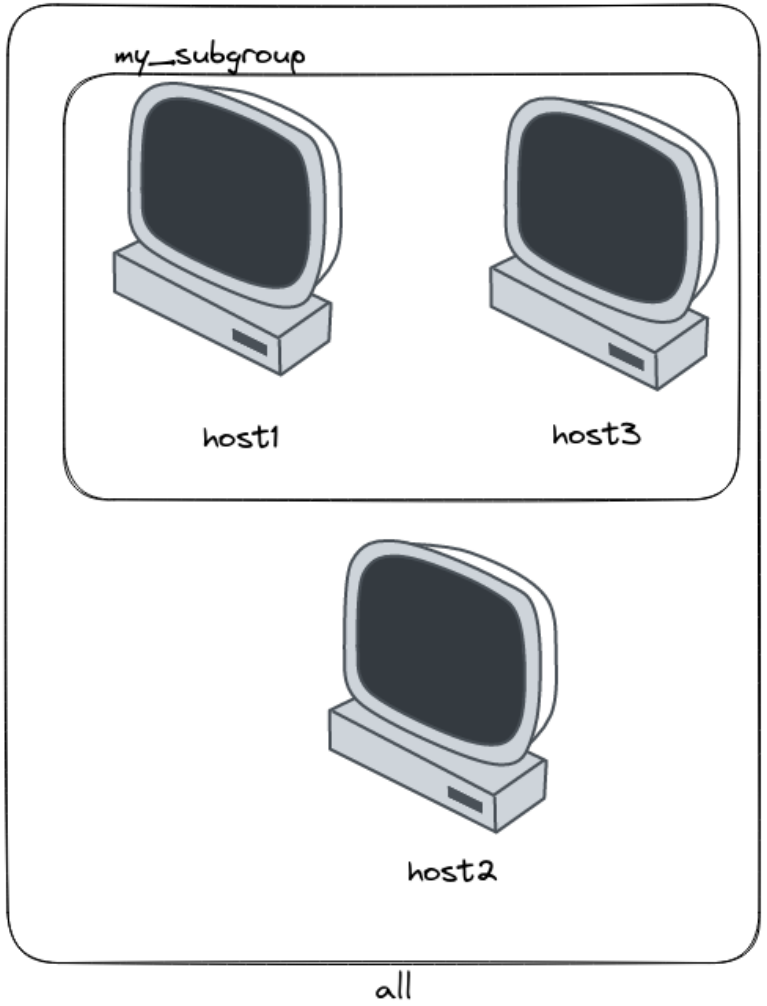
Ansible

Ansible

- Konfigurationsmanagement- und Automatisierungstool
- Ohne Clients und Daemons
- Idempotent: Nur Wunschzustand angeben
- Alle Hosts parallel verwalten
- Manche Tasks auf Hosts-Untergruppen

Grundkomponenten – Inventory

- Definiere Hosts und Hostgruppen
- YAML- oder INI-Datei
- Kann dynamisch erstellt werden, z.B.:
  - VMware
  - orcharhino/foreman
  - Cloud Providers
  - Docker
  - ...

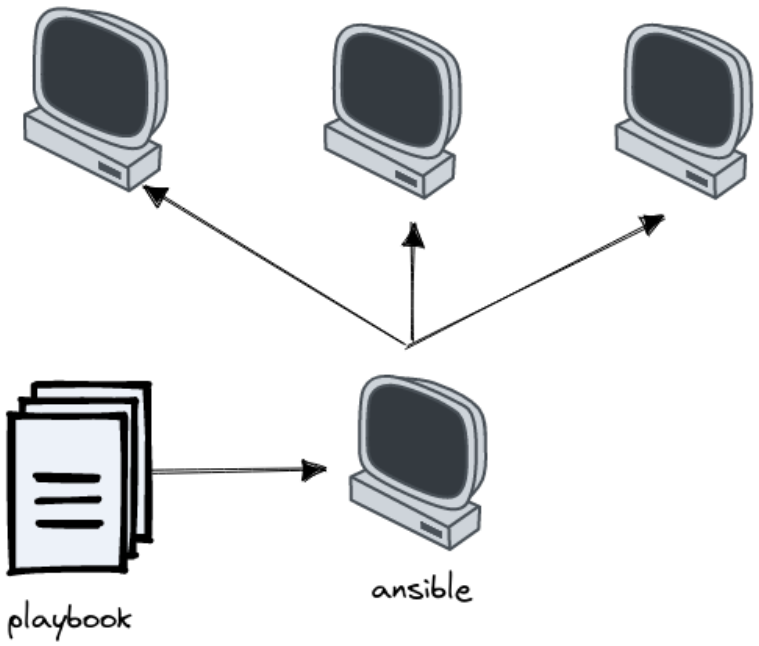


Grundkomponenten – Inventory

```yaml
---
all:
  hosts:
    192.168.121.1:
    192.168.121.2:
    192.168.121.3:
  vars:
    ansible_user: my_user
    ansible_password: my_password
  children:
    my_subgroup:
      hosts:
        192.168.121.1:
        192.168.121.3:
...
```

Grundkomponenten – Playbooks

- YAML-Datei
- Reihenfolge von Befehlen (Tasks)
- Beschreibt die Schritte für eine größere Aufgabe



Grundkomponenten – Playbooks

```yaml
---
- hosts: all
  tasks:
    - name: Install Apache packages and deps
      become: true
      ansible.builtin.yum:
        name: "{{ packages }}"
        state: present
      vars:
        packages:
          - httpd
          - httpd-devel
    - name: Copy configuration files.
      ansible.builtin.copy:
        src: "{{ item.src }}"
        dest: "{{ item.dest }}"
        mode: 0644
      loop:
        - src: "/path/to/config/httpd.conf"
          dest: "/var/httpd/conf/httpd.conf"
        - src: "/path/to/config/httpd-vhosts.conf"
          dest: "/var/httpd/conf/httpd-vhosts.conf"
    - name: Make sure Apache is started on boot.
      become: true
      ansible.builtin.service:
        name: httpd
        state: started
        enabled: true
...
```

Grundkomponenten – Playbooks

Ausgeführt mit

```
ansible-playbook -i inventory.yaml playbook.yaml
```

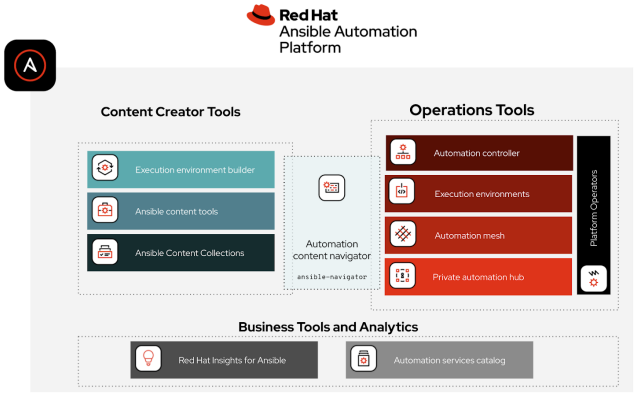
Grundkomponenten – Playbooks

Outputbeispiel:

```
PLAY [rocky] ***********************************************************************

TASK [Gathering Facts] *************************************************************
ok: [node1]

TASK [install chrony on rocky hosts] ***********************************************
ok: [node1]

TASK [set timezone] ****************************************************************
changed: [node1]

TASK [copy chrony config file] *****************************************************
changed: [node1]

TASK [start chrony] ****************************************************************
changed: [node1]

PLAY RECAP *************************************************************************
node1: ok=5    changed=3    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

Ansible Automation Engines

- AWX/ Ansible Automation Platform
- Auf Ansible aufgebaut
- Zum Planen und Verwalten von Ansible Jobs
- Angenehme Benutzeroberfläche

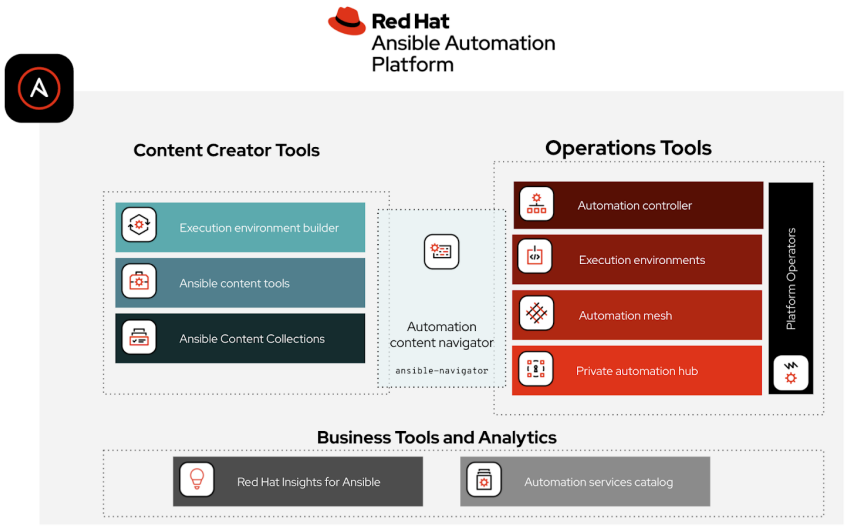


AWX

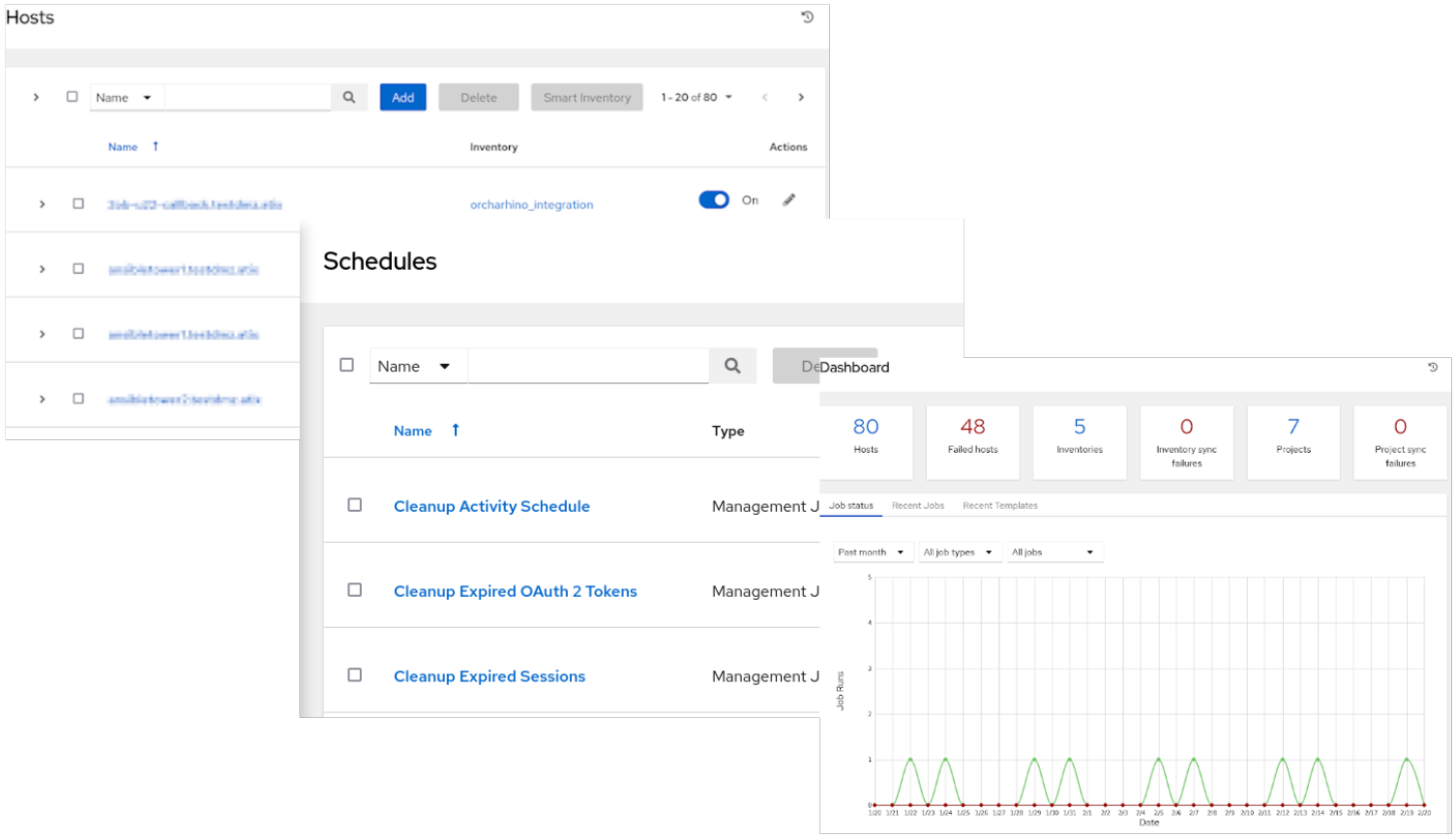
- Open Source
- vollkommen anpassbar
- Web-UI
- REST API
- RBAC
- CI/CD

Ansible Automation Platform

- Enterprise-Produkt
- stabil und zuverlässig
- Private Automation Hub
- Enterprise-Inhalte



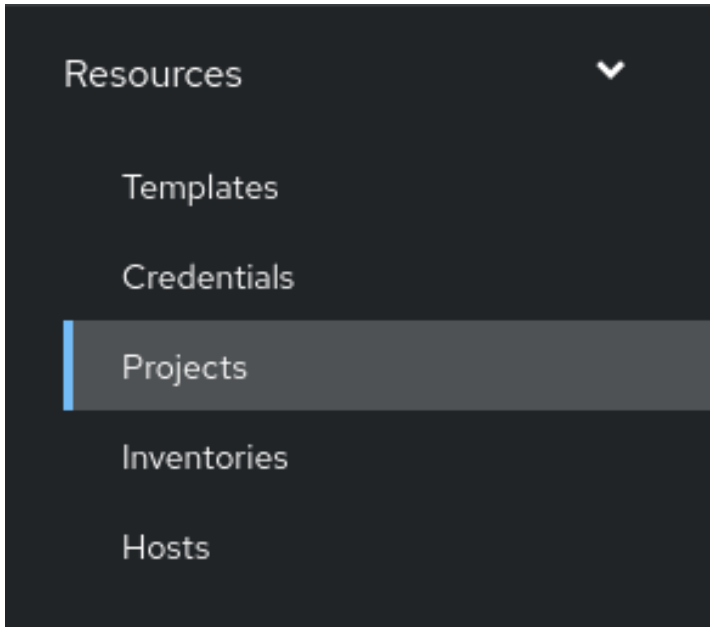
Benutzeroberfläche



AWX-Grundkomponenten: Projekte

Projekt:

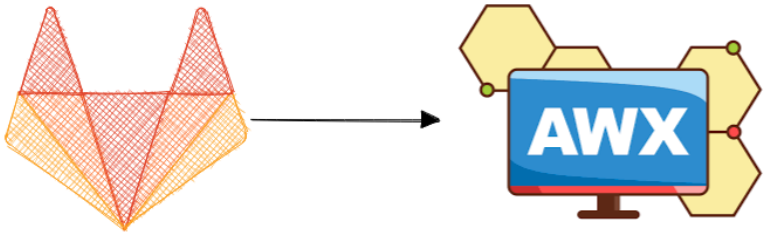
- Logische Sammlung von Ansible Playbooks
- Wird mit einem Versionsverwaltungstool synchronisiert
- Daraus werden Job Templates abgeleitet



AWX-Grundkomponenten: Projekte

Mögliche Quellen:

- Manual
- Git/Subversion
- RedHat Insights
- Remote Archive



AWX-Grundkomponenten: Job Templates

Job Template:

- Wird aus einem Projekt erstellt
- Muss ein Playbook aus dem Projekt auswählen
- Es können zusätzliche Variablen gesetzt werden
- Man kann ein Inventory zuweisen
- und mehr ...

AWX-Grundkomponenten: Job Templates

- Spezifisches Output für jeden Job unter dem Jobs-Tab
- Man kann nach fehlgeschlagenen Hosts, übersprungenen Tasks, usw. filtern
- Auch verfügbar: JSON-Ouput für jeden Task
- Ausführlichkeit kann ausgewählt werden

Jobs > 4209 - demo_my_collection

Output

Back to Jobs | Details | Output

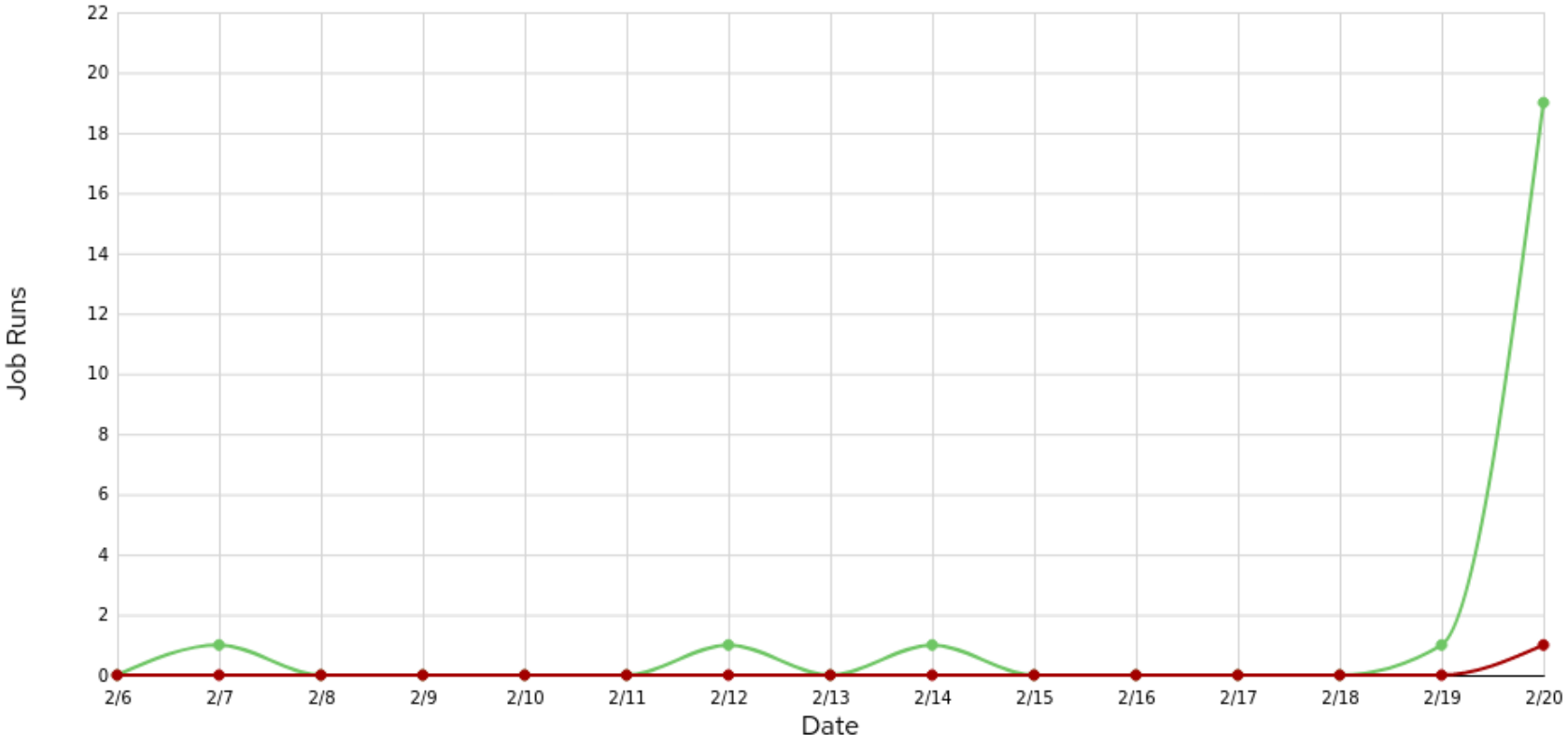
demo_my_collection Successful | Plays 1 | Tasks 2 | Hosts 1 | Elapsed 00:00:11

Stdout

```
    "msg": "192.168.165.111"
}

TASK [test_namespace.test_collection2.test_role : dummy task] ****************** 15:44:44
ok: [localhost] => {
    "msg": "dummy task"
}

PLAY RECAP ********************************************************************* 15:44:44
localhost : ok=2    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

AWX-Grundkomponenten: Job Templates

Dashboard:



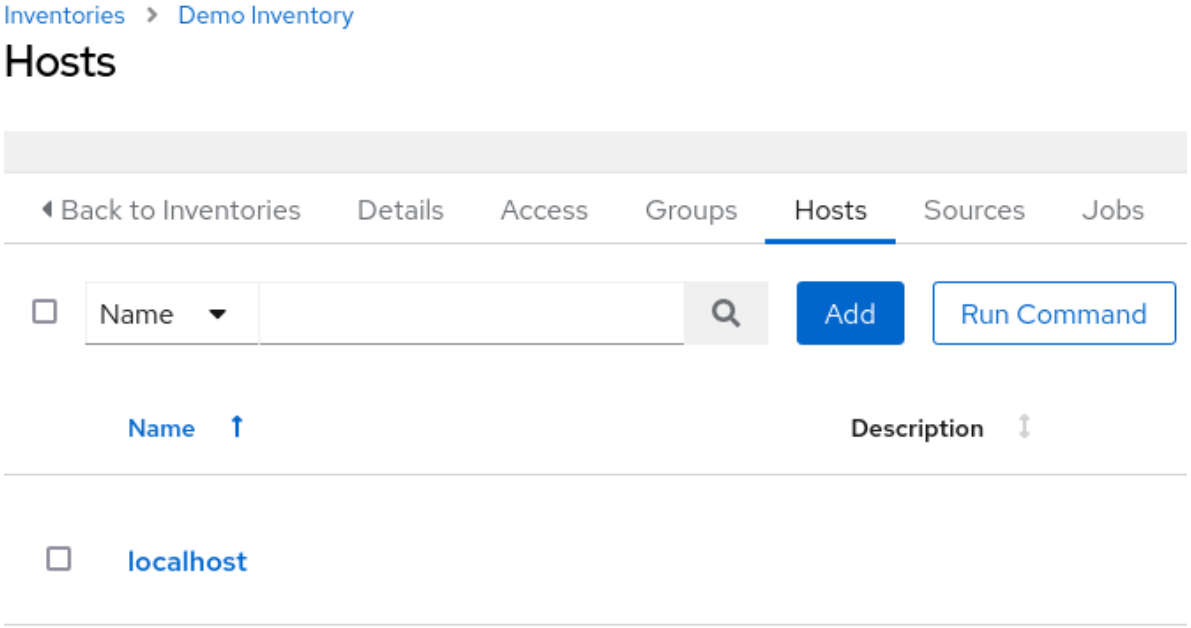
AWX-Grundkomponenten: Inventories

- Sammlung von Target-Hosts (analog zu Inventory-Dateien)
- Eigene Resource in der Weboberfläche
- Gruppen und Untergruppen möglich
- Zusatzvariablen möglich
- Werden den Jobs zugewiesen

AWX-Grundkomponenten: Inventories

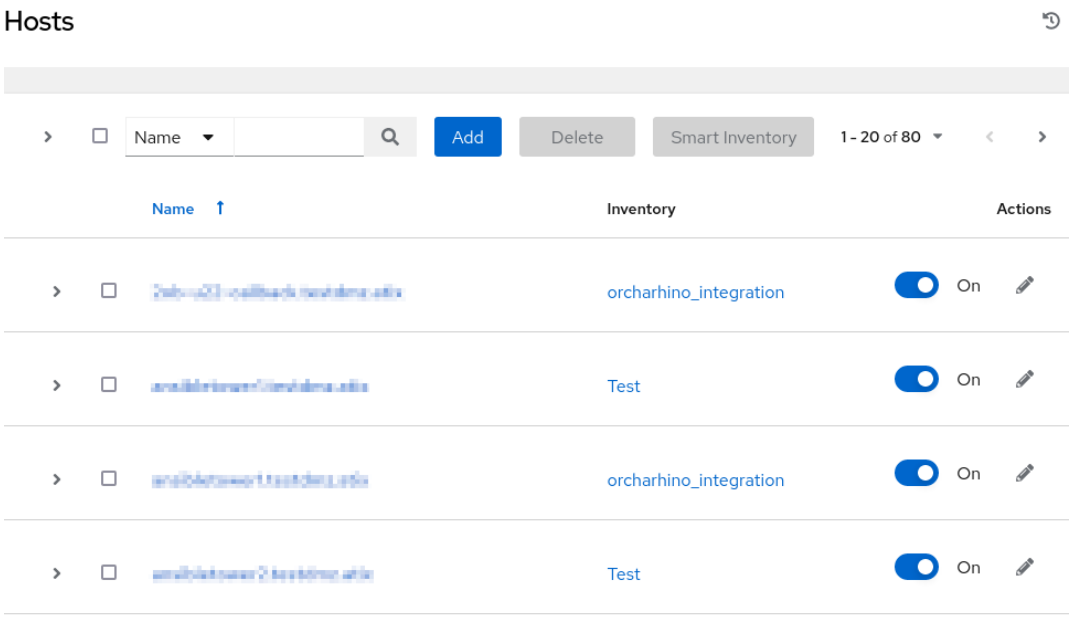
Standard-Inventories:

- Hosts per Hand ausgewählt



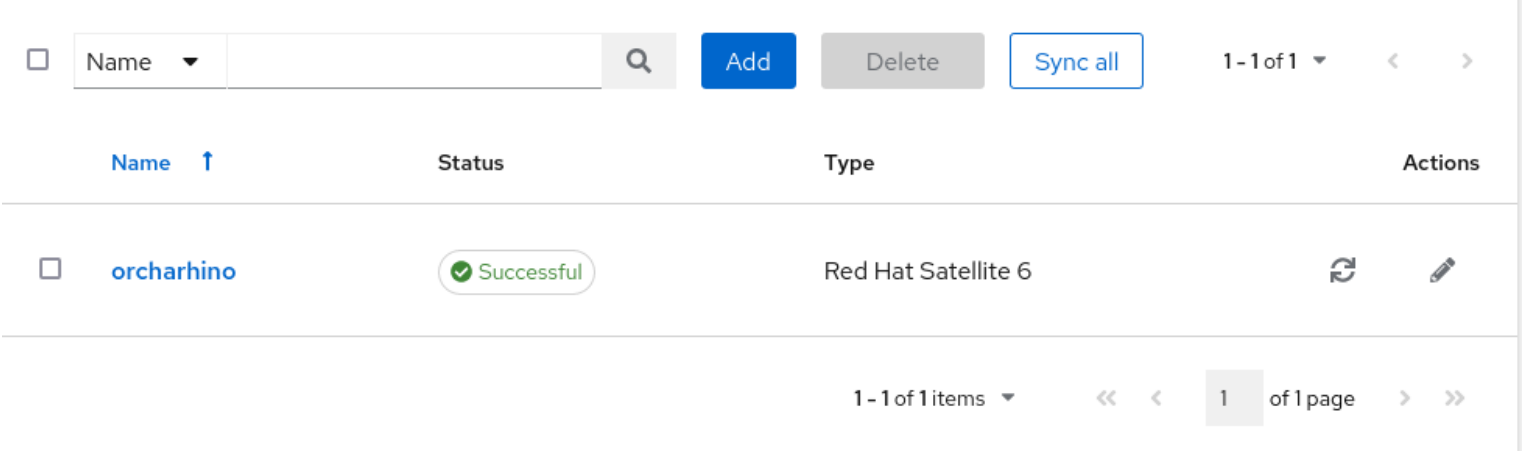
AWX-Grundkomponenten: Hosts

- eigene Resource
- eigene Variablen
- kann deaktiviert werden



AWX-Grundkomponenten: Dynamische Inventories

- Hosts werden aus einer Quelle gezogen
- Verschiedene Quellen möglich
- Automatische Gruppen hängen von der Quelle ab



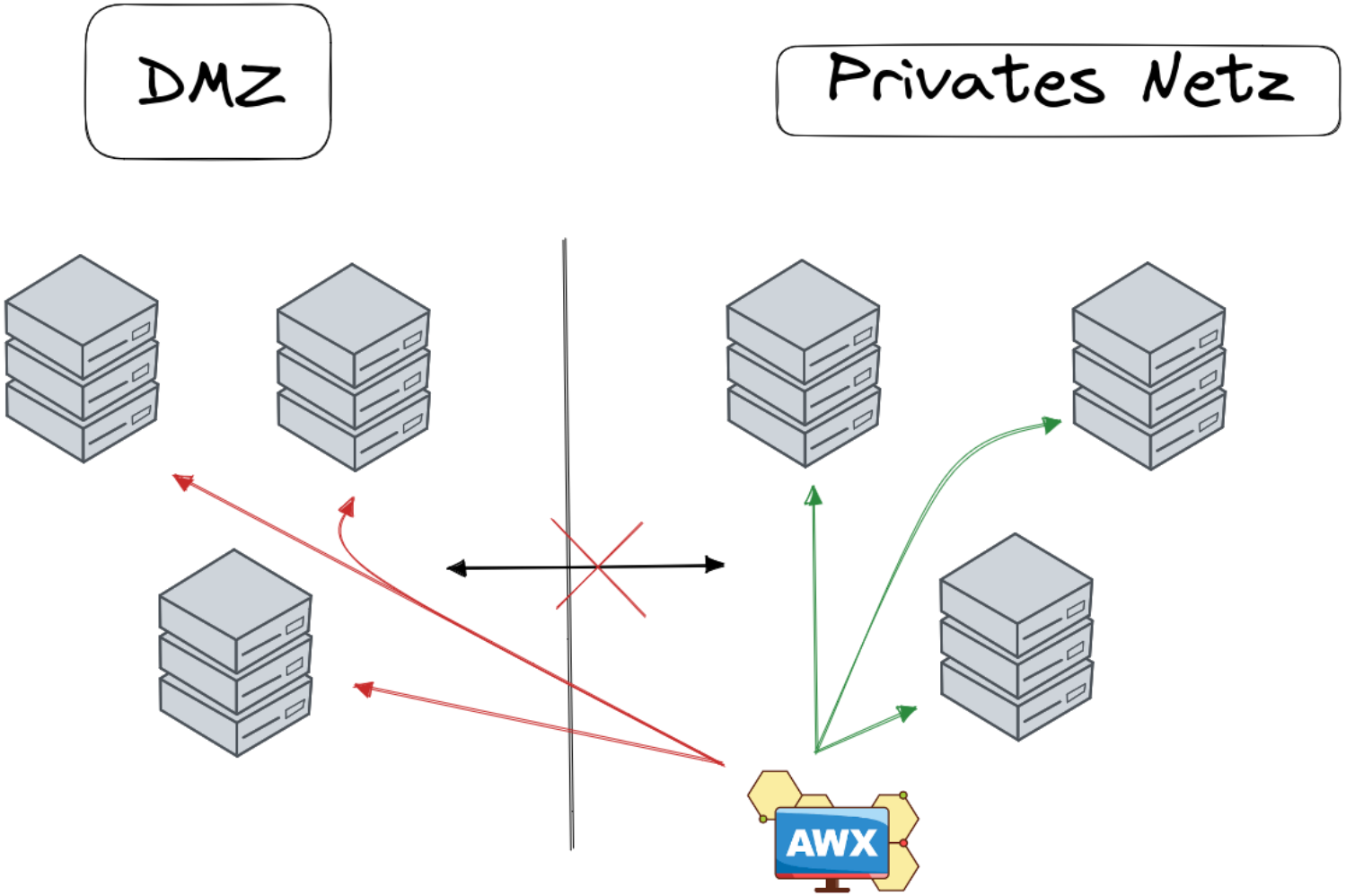
AWX-Grundlagen: Zusammenfassung

- Projekte
- Job Templates
- Inventories

Damit kann man schon alle Hosts bedienen, die von AWX erreichbar sind!

Unser Problem - 1

Die Hosts liegen in verschiedenen Netzen und können nicht alle gleichzeitig von AWX erreicht werden!



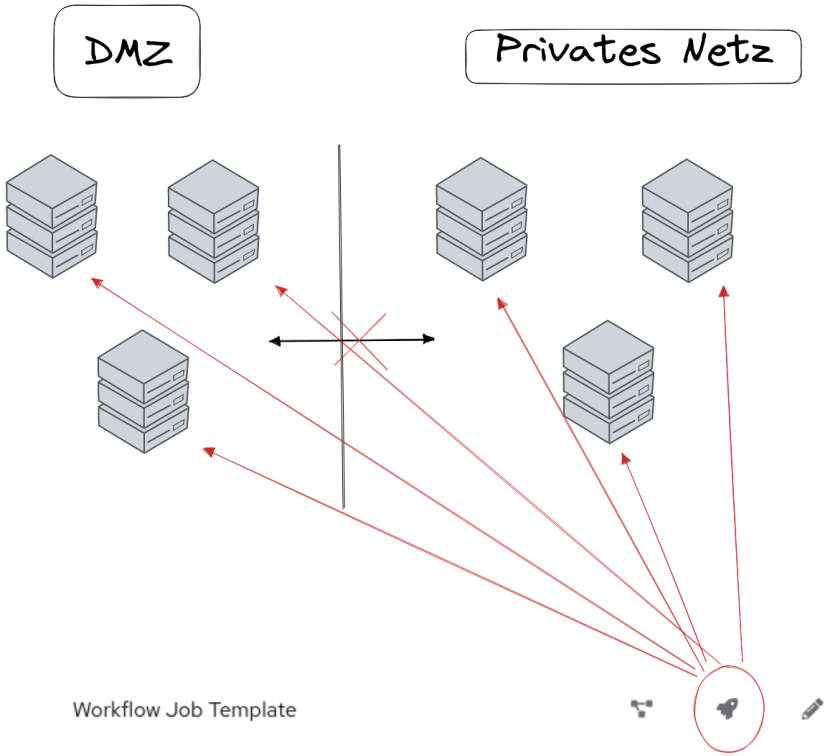
Unser Problem - 2

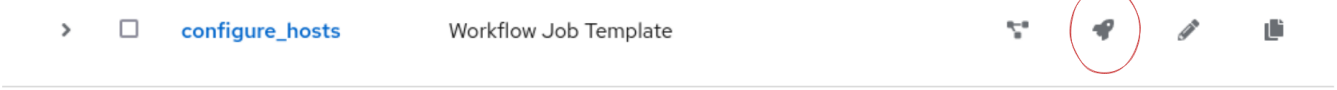
Außerdem haben wir nur ein einziges flaches Inventory:

- dynamisch erstellt
- keine geeigneten Gruppen

```yaml
---
all:
  hosts:
    host-1-dmz:
    host-2-dmz:
    host-3-privat:
    host-4-dmz:
    host-5-privat:
    host-6-privat:
    # ...
...
```

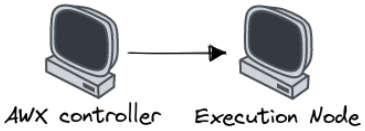
⇒ Wie können wir trotzdem alle Hosts auf einmal mit einem Knopfdruck konfigurieren?

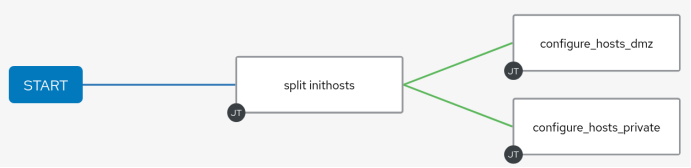




Spoiler Alert!

Lösung:





AWX + Execution Nodes + Workflow Template

Execution Nodes

Execution Nodes:

- Zusätzliche Nodes, die Ansible-Jobs ausführen
- Können keine Management-Jobs ausführen
- Können keine Web-Anfragen beantworten
- Kommunizieren über Receptor auf Port 27199

Topology

Topology:

- Netz aus Controller- und Execution-Nodes

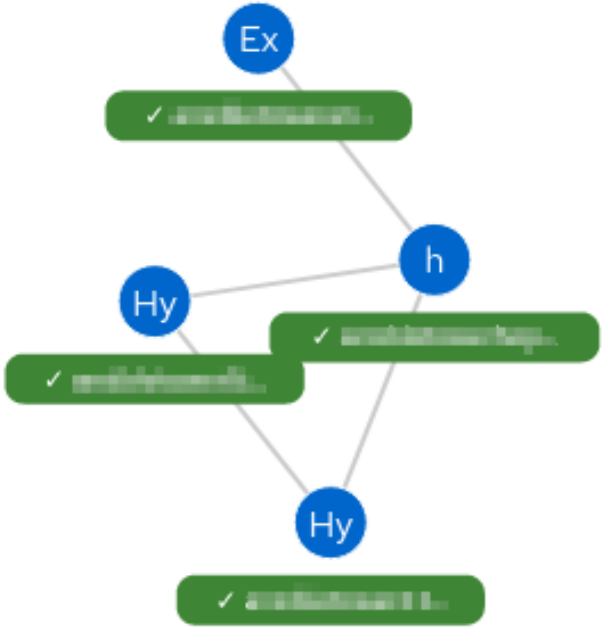
Legend

Node types

- C Control node
- Ex Execution node
- Hy Hybrid node
- h Hop node

Status types

- Healthy
- Error
- Disabled



Execution Nodes hinzufügen

Execution Nodes können in der AWX-GUI hinzugefügt werden!

Instances

Create new Instance

Host Name *

Description

Instance State

installed

Listener Port *

27199

Instance Type

execution

Options

Enable Instance

Save

Cancel

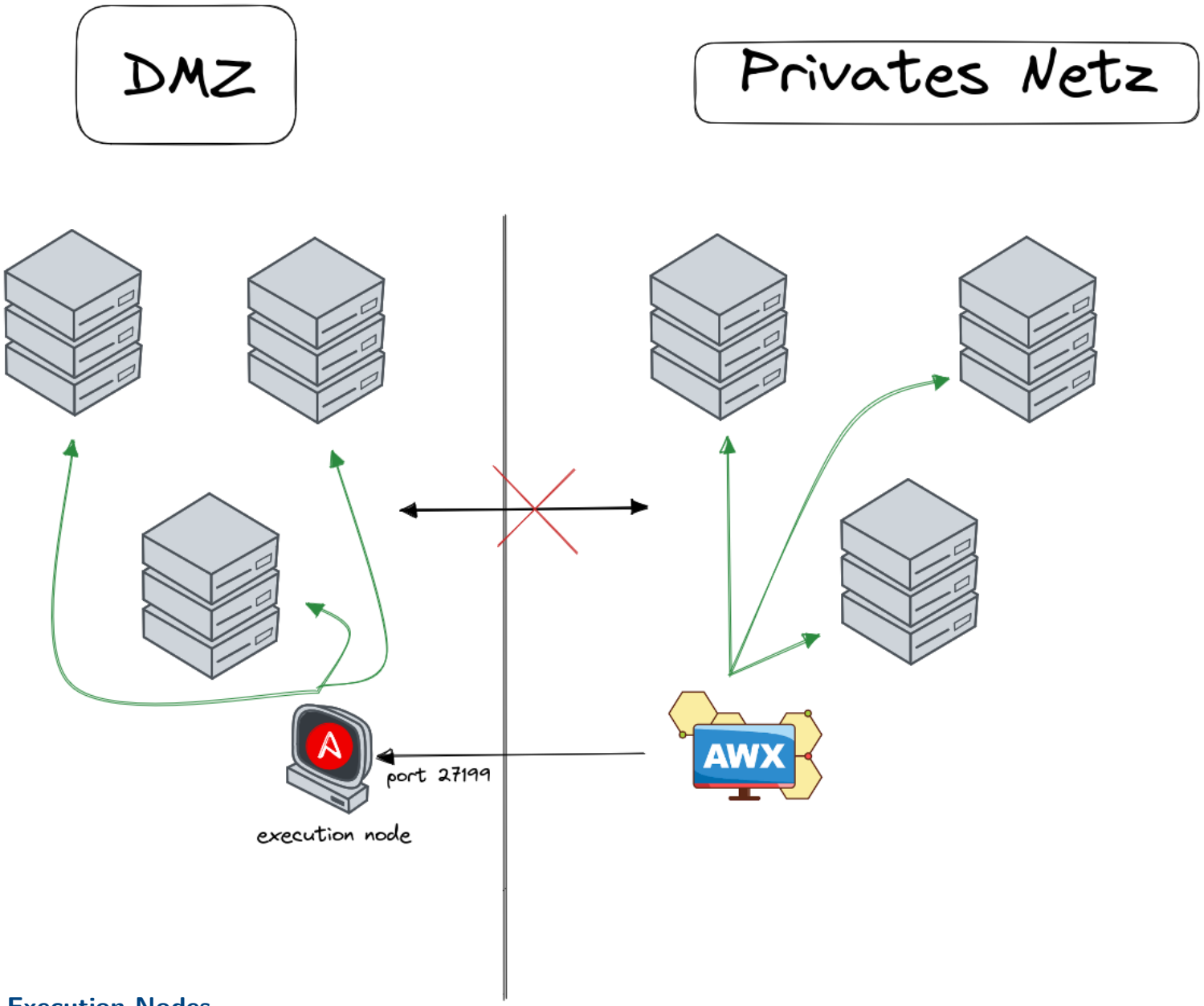
Neu seit Version 21.7.0.

Execution Nodes hinzufügen

Ansible Automation Platform:
Execution Nodes können im Inventory bei der Installation deklariert werden

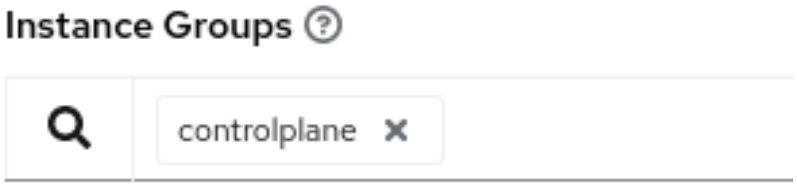
```
[automationcontroller]
mycontroller1.example
mycontroller2.example

[automationcontroller:vars]
peers=hop.example

[execution_nodes]
hop.example         node_type=hop
execution.example   node_type=execution
# this defaults to execution
execution2.example
```

Lösung für unser Problem

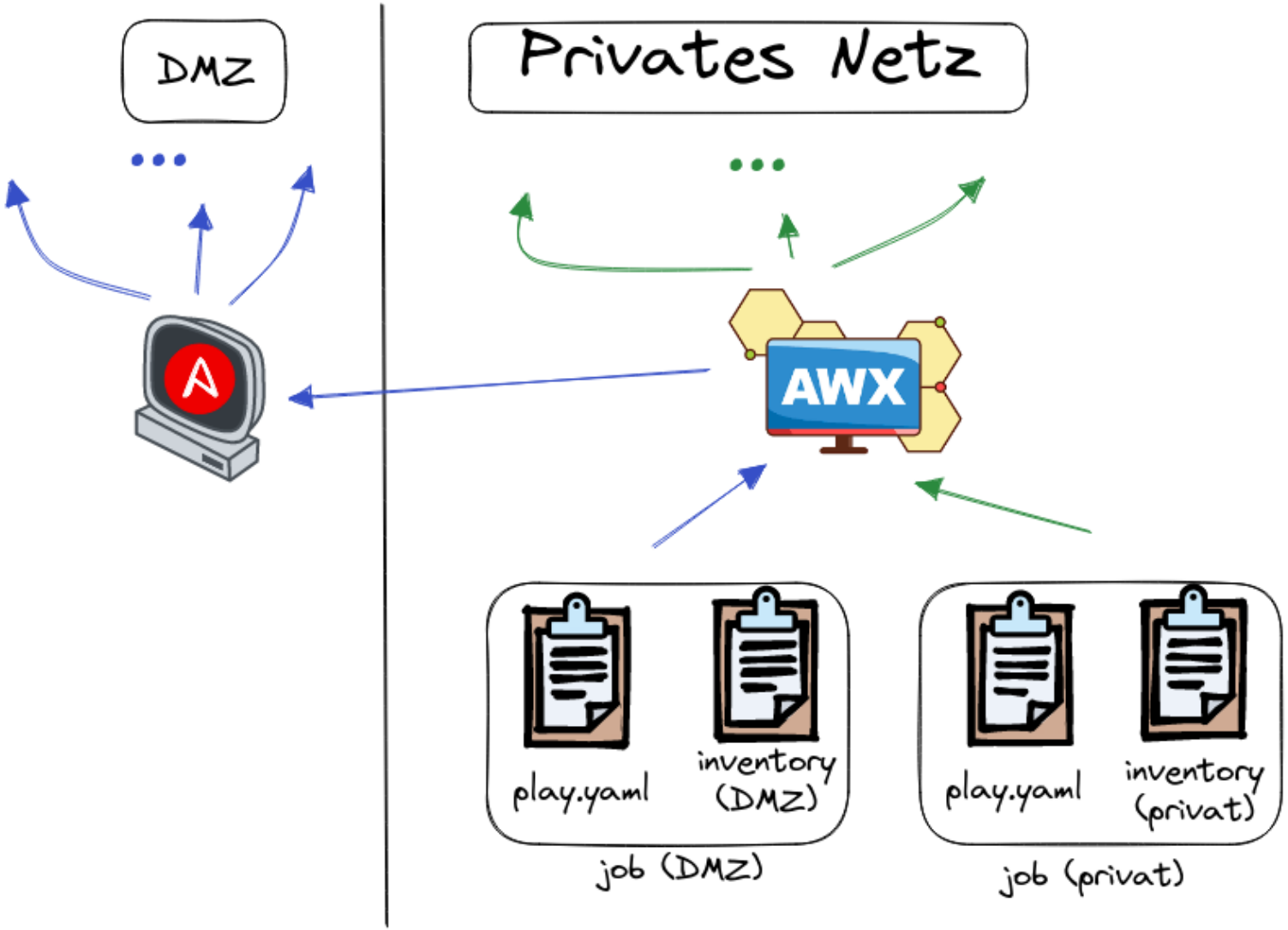


Instance Groups

- Gruppe von Nodes, die einen Job ausführen können
- Policies für die Gruppe möglich (z.B. Anzahl der Jobs)
- Kann in jedem Job Template ausgewählt werden



Lösung für unser Problem



Unser Problem

- Wir haben nur ein einziges Inventory
- Das Inventory ist dynamisch erstellt
- Es ist flach, d.h. es gibt keine geeigneten Gruppen

```yaml
---
all:
  hosts:
    host-1-dmz:
    host-2-dmz:
    host-3-privat:
    host-4-dmz:
    host-5-privat:
    host-6-privat:
    # ...
...
```

Lösungsidee

Erstelle einen neuen Job:

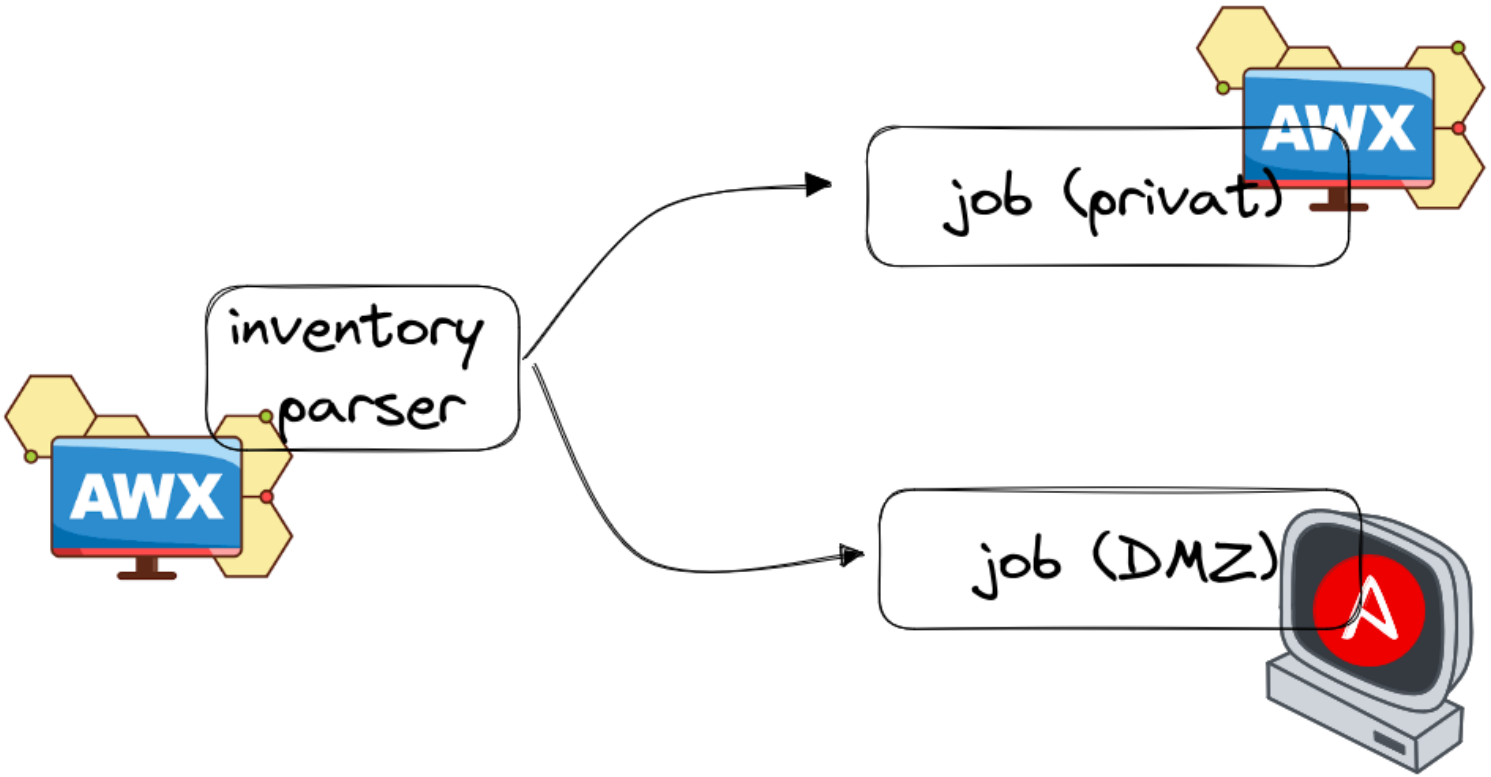
- Soll als Input das flache Inventory haben
- Soll als Output zwei Inventories zurückgeben
- Trennung z.B. mittels regex anhand der Hostnamen
- Keine Verbindung zu den Hosts nötig, weil keine Facts gebraucht werden
- Soll auf localhost laufen

Inventory parser

```yaml
---
- hosts: "{{ inithosts }}"
  gather_facts: false
  tasks:
    - name: "Add hosts to dmz"
      ansible.builtin.add_host:
        name: "{{ item }}"
        groups: dmz
      loop: "{{ ansible_play_hosts }}"
      when: '"dmz" in item'
      delegate_to: localhost
    - name: "Add hosts to private"
      ansible.builtin.add_host:
        name: "{{ item }}"
        groups: private
      loop: "{{ ansible_play_hosts }}"
      when: '"privat" in item'
      delegate_to: localhost
    - name: "Pass the new groups to the next jobs"
      ansible.builtin.set_stats:
        data:
          dmz_hosts: "{{ groups.dmz | default('localhost') }}"
          private_hosts: "{{ groups.private | default ('localhost')}}"
        aggregate: false
      delegate_to: localhost
```

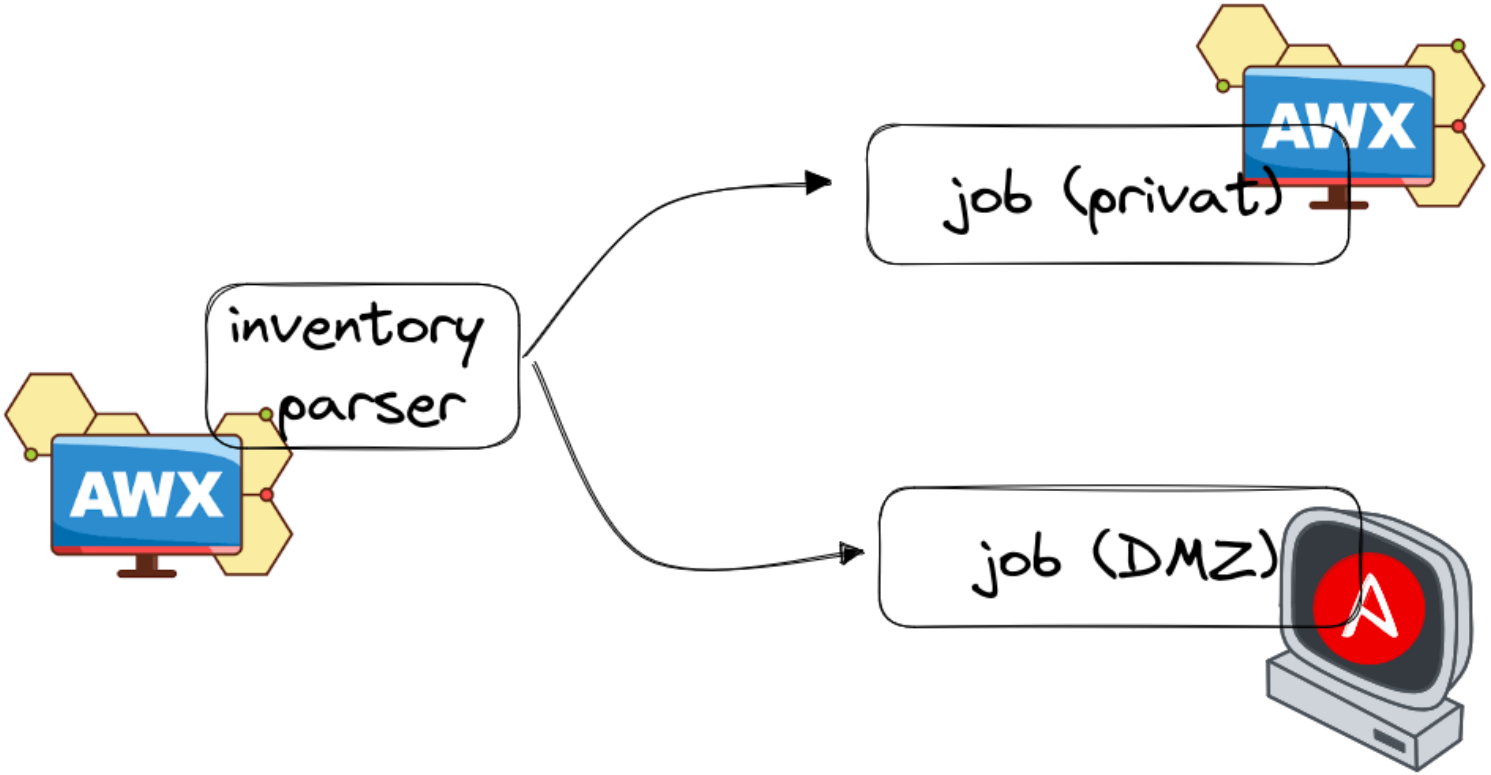
Wichtig!

- Die neuen Gruppen müssen an zwei verschiedene Jobs weitergegeben werden
- Sonst könnten sie nicht auf zwei verschiedenen Nodes ausgeführt werden



Workflow Templates

- Kette von Job Templates
- Zum Beispiel nützlich für:
  - Fehlerbehandlung
  - Genehmigung von Jobs
  - ... und unser Problem!



Workflow Templates

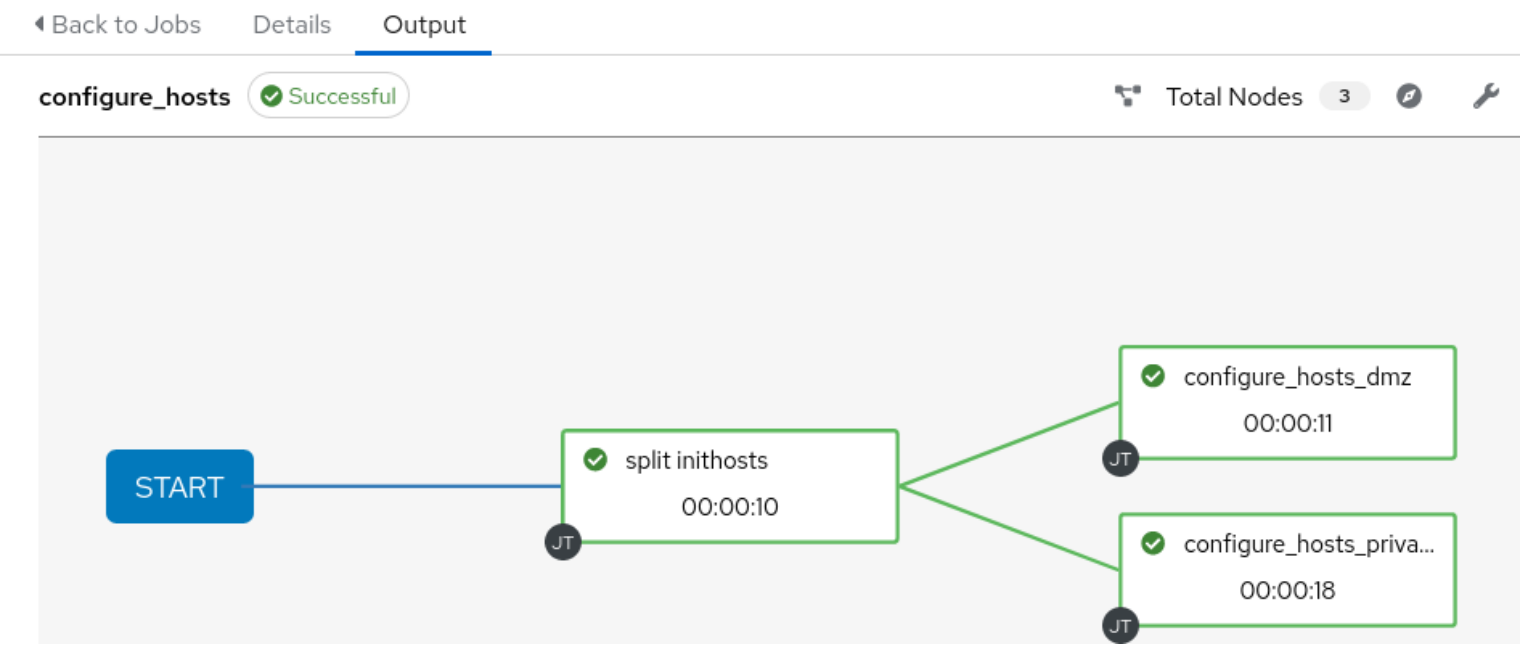
Verschiedene Arten von Nodes:

- Job Template
- Workflow Job Template
- Project Sync
- Approval
- Inventory Source Sync
- Management Job

Lösung

Ein Workflow Template welches drei Jobs ausführt:

1. Aufteilen der Hosts in die korrekten Inventories
2. Ausführen des Jobs im DMZ durch die Exection Node
3. Ausführen des Jobs im privaten Netz von AWX selbst



Zusammenfassung

- AWX als Tool für die Automatisierung und Verwaltung von Ansible-Projekten
- Komplexe Topologie-Szenarien dank Execution Nodes möglich
- Workflow Templates zur Abstimmung zwischen Jobs

Kurz: AWX ist cool!

Danke für eure Aufmerksamkeit!

```
 _________________
< Gibt es Fragen? >
 -----------------
        \   ^__^
         \  (oo)\_______
            (__)\       )\/\
                ||----w |
                ||     ||
```

- WebAssembly auf der Serverseite: Was ist WASI?, Dr. P. Fries
- Autoscaling in Kubernetes – From Zero to Hero, T. Manske und L. Paluch
- Das Chaos überblicken – Monitoring & Tracing in Kubernetes, S. Rauch und V. Welker