consol
Wir unternehmen IT

redhat
ADVANCED
BUSINESS PARTNER

# Ansible

## Einführung und Hands-on

**Michael Kraus**
Meetup „Ansible im Monitoring-Umfeld“ / München, 27. Juli 2016

Münchner Monitoring-Stammtisch
OK, WARNING, CRITICAL, UNKNOWN und Bier

## Was ist Ansible?

„Ansible is a extra-simple Python API for doing '***remote things***' over SSH. “

(Erster Commit auf Github)

## Einsatzmöglichkeiten

- Konfigurationsmanagement
- Softwareverteilung
- Orchestrierung
- Administration
- …

## Vorteile

- Konfiguration leicht zu erlernen
- Keine spezielle Software nötig
  - „control machine“
  - „managed node“
- Kommunikation über SSH
- Koexistenz möglich

## Voraussetzungen

- „control machine“: Unix (kein Windows)
  - Python 2.6 oder 2.7
- „managed node“: Unix (auch Windows)
  - Python 2.5
  - Python 2.4 mit python-simplejson
  - (libselinux-python)

## Windows-Unterstützung

- PowerShell remoting statt SSH
- Voraussetzungen
  - „managed node“: WinRM aktiviert
  - „control node“: python-winrm, (python-kerberos)
- Seit 2.1 nicht mehr „beta“

# Ad-hoc Kommandos [Demo]

# Ad-hoc Kommandos [Demo]

```
[~]$ ansible all -a ”hostname”
app1 | SUCCESS | rc=0 >>
app1

app2 | SUCCESS | rc=0 >>
app2

web1 | SUCCESS | rc=0 >>
web1

web2 | SUCCESS | rc=0 >>
web2
```

```
[~]$ ansible all -a "uptime"
web2 | SUCCESS | rc=0 >>
 13:11:42 up  3:49,  1 user,  load
average: 0,01, 0,03, 0,05

app1 | SUCCESS | rc=0 >>
 13:11:42 up  3:48,  1 user,  load
average: 0,00, 0,01, 0,05

app2 | SUCCESS | rc=0 >>
 13:11:42 up  3:48,  1 user,  load
average: 0,00, 0,01, 0,05
...
```

consol Wir unternehmen IT

redhat ADVANCED BUSINESS PARTNER

## Begriffe

- Inventory
- Module
- Task / Playbook / Role
- „Idempotenz“

# Begriff: Inventory

- Liste von Zielelementen (z.B. Hosts)
- Gruppierung, Variablenzuweisung

```
[webserver]
web01 ansible_user=someuser
web02 http_port=8080

[webserver:vars]
function=webserver

[appserver]
app[01:20]
```

## Begriff: Inventory

- Möglichkeiten der Erzeugung
  - Manuell
  - Programm-Export
  - Programm-Ausgabe (JSON-Format)
    - „Dynamic Inventory“

## Begriff: Module

- Module stellen Funktionalitäten bereit
- Eigene Module implementierbar

```yaml
- name: Install httpd
  yum:
    name: httpd
    state: latest
```

```yaml
- name: Start httpd
  service:
    name: httpd
    state: started
    enabled: yes
```

# Begriff: Module

**Module Index:**

All Modules
Cloud Modules
Clustering Modules
Commands Modules
Database Modules
Files Modules
Inventory Modules
Messaging Modules
Monitoring Modules
Network Modules
Notification Modules
Packaging Modules
Source Control Modules
System Modules
Utilities Modules
Web Infrastructure Modules
Windows Modules

**System Modules:**

alternatives (E) - Manages alternative programs for common commands
at (E) - Schedule the execution of a command or script file via the at command.
authorized_key - Adds or removes an SSH authorized key
capabilities (E) - Manage Linux capabilities
cron - Manage cron.d and crontab entries.
cronvar (E) - Manage variables in crontabs
crypttab (E) - Encrypted Linux block devices
debconf (E) - Configure a .deb package
facter (E) - Runs the discovery program *facter* on the remote system
filesystem (E) - Makes file system on block device
firewalld (E) - Manage arbitrary ports/services with firewalld
getent (E) - a wrapper to the unix getent utility
gluster_volume (E) - Manage GlusterFS volumes
group - Add or remove groups
hostname - Manage hostname
iptables (E) - Modify the systems iptables
kernel_blacklist (E) - Blacklist kernel modules
known_hosts (E) - Add or remove a host from the ``known_hosts`` file
locale_gen (E) - Creates or removes locales.
lvg (E) - Configure LVM volume groups
lvol (E) - Configure LVM logical volumes
modprobe (E) - Add or remove kernel modules
mount - Control active and configured mount points
ohai (E) - Returns inventory data from *Ohai*
open_iscsi (E) - Manage iscsi targets with open-iscsi
...

## Begriff: Task

- Rahmenbedingungen für Funktionsaufrufe
- Strukturierung

```
- name: Install software
  yum:
    name: “{{ item }}”
    state: latest
  with_items:
    - httpd
    - mysql
  when: is_webserver
  tag:
    - mytag
```

```
- name: Include OS specific
  include: tasks/RedHat.yml
  when: ansible_os_family == “RedHat”
```

# Begriff: Playbook

- Sammlung von Tasks und/oder Roles

```
---
- hosts: all
  become: yes
  gather_facts: no

  tasks:
  - name: Install packages
    yum:
      name: "{{ item }}"
      state: latest
    with_items:
      - rsync
```

```
---
- hosts: all
  gather_facts: yes

  roles:
    - nagiosconfig
    - ...
    - ...
```

## Begriff: Role

- Wiederverwendbare Komponenten
- Tasks, Variablen, Templates, ...

```
roles/
└── nagiosconfig
    ├── defaults
    │   └── main.yml
    ├── files
    ├── handlers
    │   └── main.yml
    ├── meta
    │   └── main.yml
    ├── README.md
    ├── tasks
    │   └── main.yml
    ├── templates
    ├── tests
    │   ├── inventory
    │   └── test.yml
    └── vars
        └── main.yml
```

## Begriff: Facts [Demo]

- Informationen über den Ziel-Host

## Begriff: Facts [Demo]

```
[~]$ ansible web1 -m setup
web1 | SUCCESS => {
    "ansible_facts": {
        "ansible_all_ipv4_addresses": [
            "10.0.2.15",
            "10.0.15.21"
        ],
        "ansible_all_ipv6_addresses": [
            "fe80::a00:27ff:fef6:b007",
            "fe80::a00:27ff:fe91:afac"
        ],
        "ansible_architecture": "x86_64",
...
```

## Begriff: „Idempotenz“

„In der Kommunikation zwischen Mensch und Maschine ist dann *Idempotenz* gegeben, wenn das mehrmalige Drücken eines Knopfes den gleichen Effekt hat wie das einmalige Drücken.“

(Wikipedia)

- Wiederholbare Ausführung

## Demo

- Demo
  - Verteilung von SSH-Keys
  - Generierung einer Nagios-Konfiguration anhand der gefundenen Ansible-Facts

## Demo

- Demo-Umgebung
  - **https://github.com/m-kraus/ansible-demo**

## Neu in 2.0 - I

- Blocks: Gruppierung, Fehlerbehandlung

```
tasks:
 - block:
     - debug: msg='I execute normally'
     - do something ...
   rescue:
     - debug: msg='I caught an error'
     - undo something ...
   always:
     - debug: msg='This always executes‘
   when: some_condition
```

## Neu in 2.0 - II

- Ausführung
  - Linear:
    Warten auf Abschluss eines Tasks für alle Hosts
  - Frei:
    Ausführung pro Host so schnell wie möglich

## Neu in 2.0 - III

- „Dynamic includes“

```
# Vor v2.0

 - include: RedHat.yml
   when: ansible_os_family == "RedHat"
 - include: Debian.yml
   when: ansible_os_family == "Debian"

# Seit v2.0

 - include: "{{ ansible_os_family }}“.yml
```

## Neu in 2.1

- Netzwerk-Komponenten
  - Cisco, HP, Juniper, Arista, Cumulus
- Windows-Untersützung nicht mehr „beta“
- Erweiterte Docker-Unterstützung

ANSIBLE

## Danke!

## ConSol Software GmbH

Franziskanerstraße 38
D-81669 München

Tel:   +49-89-45841-100
Fax: +49-89-45841-111

info@consol.de
www.consol.de