Ansible …

“Half of it nobody understands anyway.”

Dr. Bernhard Hopfenmüller

16.10.2018
Bernhard Hopfenmüller
IT Consultant @ ATIX AG

IRC: Fobhep
github.com/Fobhep
The Linux & Open Source Company
Garching @ München

over 15 years
datacenter automation, Linux Consulting, Engineering, Support, Training
Ender’s Game

imdb.com
Ansible in a nutshell

Ansible …

- … is “radically simple”
- … requires: 1 Unixnode, SSH, Python
- … needs no agent
- … describes the desired state
- … is written in YAML
- … satisfies idempotency
Run Ansible

- User: Write Statements
- Ansible: Create program
- Ansible: Copy program to Run-Node
- Ansible: Connect to Run-Node (local, SSH, PowerShell)
- Ansible: Run Program
- Ansible: Communication via JSON
- Ansible: Delete local Program
- Ansible: Give Feedback to User
Ansible dict

- **Inventory** - infrastructure as code
- **Adhoc mode** - one task + terminal
- **Playbook** - collection of tasks/roles
- **Role** - one collection of tasks
- **Task** - one Ansible command: `apt-get install nano`
- **Jinja 2 templating** - e.g. for variables
- **Modules** - core-components: yum module, ping module,... communication via JSON
Inventory

- defining the infrastructure
- ini or yaml syntax
- combine servers into groups
- define specific vars

[dbserver]
db1.server.com
db2.server.com

[webserver]
web1.server.com ansible_host=4.8.15.16
web2.server.com ansible_host=4.8.23.42

[appserver]
app1.server.com default_port=23
app2.server.com default_port=69105
Ad-hoc mode

- run only one task quickly
- either one-timer or quicker than playbook

```
$ ansible webserver -m patch \\
  -a \\
  "src=/src/critical.patch \\
  dest=/app/bin/base"

$ ansible webserver -a "sbin/reboot"

$ ansible webserver -m ping
```
- playbooks
  - hosts: webserver # where?
    become: true # run as root
  roles:
    - bake_server # role1
    - gitlab # role2

- combination of roles
- written in YAML (or JSON)-Syntax
- ideally in combination with VCS
Role

- combination of tasks
- featuring idempotency
- ideally in combination with VCS

---

- name: Install GitLab dependencies
  yum:
    name: "{{ item }}"
    state: latest
  loop:
    - openssh-server
    - postfix
    - curl
    - openssl

- name: Install GitLab
  yum:
    name: "{{ gitlab-version }}"
    state: present

- name: Copy config file
  template:
    src: gitlab.rb.j2
    dest: /etc/gitlab/gitlab.rb
    mode: 0600
  notify: restart gitlab

...
Questions - so far?

https://imgflip.com/
Some RL- Stories

unrelated cat-content
YAML - everything easy, or not?!

My variable is not working!!”
YAML - everything easy, or not?!
--- # don't forget this!
a_dict:
  foo: "fine!"
  foo42: "meh- but ok"
---
  # don't forget this!
  a_dict:
    foo: "fine!"
    foo42: "meh- but ok"
    foo-two: "no!"
--- 
# don't forget this!
a_dict:
  foo: "fine!"
  foo42: "meh- but ok"
  foo-two: "no!"
  foo.bar: "oh god, noo!"
---  # don't forget this!
a_dict:
  foo: "fine!"
  foo42: "meh- but ok"
  foo-two: "no!"
  foo.bar: "oh god, noo!"
  foo bar: "what??!"
---

# don't forget this!

```yaml
a_dict:
  foo: "fine!"
  foo42: "meh - but ok"
  foo-two: "no!"
  foo.bar: "oh god, noo!"
  foo bar: "what??!"
  12: "... o0"
```

---
---  # don't forget this!

```
a_dict:  
  foo: "fine!"  
  foo42: "meh- but ok"  
  foo-two: "no!"  
  foo.bar: "oh god, noo!"  
  foo.bar: "what??!"  
  12: "... o0"  
  foo_12: "yes!!"
```
YAML - everything easy, or not?!

“Ansible is not working!”
---

ver_1: 2.4.2
ver_2: 2.5

- name: Diff versions
debug:
  msg: "Version 1 is smaller!"
  when: ver1 < ver2  # Nope!
---

ver_1: "2.4.2"
ver_2: "2.5"

- name: Diff versions
d  debug:
    msg: "Yeah!"
    when: ver1 < ver2  # YipYip!
ooor...be smart ;)

Filters ftw!!

```python
{{ ver2 | version_compare('ver1', '>=' ) }}
```
Easy life with filters

“We need this, but it’s impossible in Ansible!”
Add a new entry to a dict? Make use of implemented filters!

---

vars:
  dict:
    foo: "bar"
    foo2: "bar2"

tasks:
  - set_fact:
    dict: "{{ dict | combine( { 'foo3': 'bar3' } ) }}"

...
Easy life with filters

Real life example: PUT- Request to REST - API

```json
body = {
    "id":"01189998819991197253",
    "label":"plugin",
    "skip_production": true # we don't want this!
}
```
Easy life with filters

a while later...

imgflip.com
- set_fact:
    body: "{{ {} |combine({{item.key: item.value}})}}"
    when: "{{item.key not in ['skip_production']}}"
    with_dict: "{{ body }}"
...
Easy life with filters

---

- set_fact:
  
  body: "{{ {} |combine({item.key: item.value})}}"

  when: "{{item.key not in ['skip_production']}}"

  with_dict: "{{ body }}"

...
Easy life with filters

another while later…
```python
#!/usr/bin/python

class FilterModule(object):
    def filters(self):
        return {
            'remove_key': self.remove_key
        }

def remove_key(self, old_dict, key):
    old_dict.pop(key, None)
    return old_dict

# atix # osad2018
```
#filter_plugins/remove_key.py

```python
#!/usr/bin/python

class FilterModule(object):
    def filters(self):
        return {
            'remove_key': self.remove_key
        }

    def remove_key(self, old_dict, key):
        old_dict.pop(key, None)
        return old_dict
```

```yaml
- set_fact:
  body: "{{ body | remove_key('skip_production') }}"
...
---

- set_fact:
  body: "{{ body | remove_key('skip_production') }}"

...
- set_fact:
  body: "{{ body | remove_key('skip_production') }}"

accurate face expression
“Wait? There’s more than ansible and ansible-playbook?”
Ansible-CLI

ansible-galaxy

- search public roles
- create a template for roles
- ...

#atix #osad2018
Ansible-CLI

ansible-doc
  ▶ offline documentation
  ▶ any plugin type
  ▶ ...

#atix #osad2018
ansible-console

- Ansible REPL
- test stuff quickly, e.g. jinja...
- ...

#atix #osad2018
ansible-pull

- automate your automation
- reset system automatically
- speed up!
- ...

#atix #osad2018
It’s slow...

ANSIBLE

Y U NO BE FASTER?!
What are you doing Ansible

▶ scan used module for python deps
What are you doing Ansible

- scan used module for python deps
- add create script and add it and deps to ZIP-file
What are you doing Ansible

- scan used module for python deps
- add create script and add it and deps to ZIP-file
- base64 encoding of ZIP-file
What are you doing Ansible

- scan used module for python deps
- add create script and add it and deps to ZIP-file
- base64 encoding of ZIP-file
- write content to local filesystem for sftp
What are you doing Ansible

- scan used module for python deps
- add create script and add it and deps to ZIP-file
- base64 encoding of ZIP-file
- write content to local filesystem for sftp
- find target user home dir
What are you doing Ansible

- scan used module for python deps
- add create script and add it and deps to ZIP-file
- base64 encoding of ZIP-file
- write content to local filesystem for sftp
- find target user home dir
- create temp dir in user home
What are you doing Ansible

- scan used module for python deps
- add create script and add it and deps to ZIP-file
- base64 encoding of ZIP-file
- write content to local filesystem for sftp
- find target user home dir
- create temp dir in user home
- sftp ZIP
What are you doing Ansible

- scan used module for python deps
- add create script and add it and deps to ZIP-file
- base64 encoding of ZIP-file
- write content to local filesystem for sftp
- find target user home dir
- create temp dir in user home
- sftp ZIP
- run script on target (16 quote layering! 10 steps including file operations)

#atix #osad2018
What are you doing Ansible

- scan used module for python deps
- add create script and add it and deps to ZIP-file
- base64 encoding of ZIP-file
- write content to local filesystem for sftp
- find target user home dir
- create temp dir in user home
- sftp ZIP
- run script on target (16 quote layering! 10 steps including file operations)
- rm temp dir on target
What are you doing Ansible

- scan used module for python deps
- add create script and add it and deps to ZIP-file
- base64 encoding of ZIP-file
- write content to local filesystem for sftp
- find target user home dir
- create temp dir in user home
- sftp ZIP
- run script on target (16 quote layering! 10 steps including file operations)
- rm temp dir on target
What are you doing Ansible

➤ scan used module for python deps
➤ add create script and add it and deps to ZIP-file
➤ base64 encoding of ZIP-file
➤ write content to local filesystem for sftp
➤ find target user home dir
➤ create temp dir in user home
➤ sftp ZIP
➤ run script on target (16 quote layering! 10 steps including file operations)
➤ rm temp dir on target
forks = 5 # default
# parallel Ansible processes i.e. number of hosts
- name: I will work while the rest goes on
  pip:
    requirements: requirements.txt
    async: 1000  # choose this high enough!
    poll: 0       # fire and forget!
    register: pip_install

- name: More tasks....
  [....]

- name: Deploy a python app.. I might crash :(
Async

- name: I will work while the rest goes on
  pip:
    requirements: requirements.txt
    async: 1000  # choose this high enough!
    poll: 0  # fire and forget!
    register: pip_install

- name: More tasks....
  [....]

- name: PIP result check
  async_status:
    jid: "{{ pip_install.ansible_job_id }}"
    register: job_result
    until: job_result.finished
    retries: 30

- name: Deploy a python app.. I am happy now :)

#atix #osad2018
SSH Pipelining

[ssh_connection]
  pipelining = True

Careful: This needs to disable requiretty' in /etc/sudoers on all managed hosts
Mitogen

▶ Python library
Mitogen

- Python library
- self reproducing programs (cells: Mitosis)

https://mitogen.readthedocs.io/en/latest/_images/cell_division.png
Mitogen

- Python library
- self reproducing programs (cells: Mitosis)
- "network-capable fork() on steroids"

https://mitogen.readthedocs.io/en/latest/_images/cell_division.png
Mitogen

- Python library
- self reproducing programs (cells: Mitosis)
- “network-capable fork() on steroids”
- “lazily-loaded duplicates on remote hosts, without requiring any upfront remote disk writes”

https://mitogen.readthedocs.io/en/latest/_images/cell_division.png
Mitogen

- tl;dr: similar to SSH-Pipeline but no "require-tty problem"

https://mitogen.readthedocs.io/en/latest/_images/cell_division.png
tl;dr: similar to SSH-Pipeline but no “require-tty problem”

facts only: “1.25x - 7x speedup + CPU usage reduction >= 2x”

https://mitogen.readthedocs.io/en/latest/_images/cell_division.png
More questions - your story?
Ansible Training

atix.de/training/ansible-training/

- 2 or 3 days
- Hands-on based
- topics include
  - Ansible basics
  - From Paper to Playbook
  - Jinja Templating
  - Ansible + VCS
  ...