Role over, lay down

Testing Roles With Molecule

Ton Kersten

Ghent / Belgium / 2018

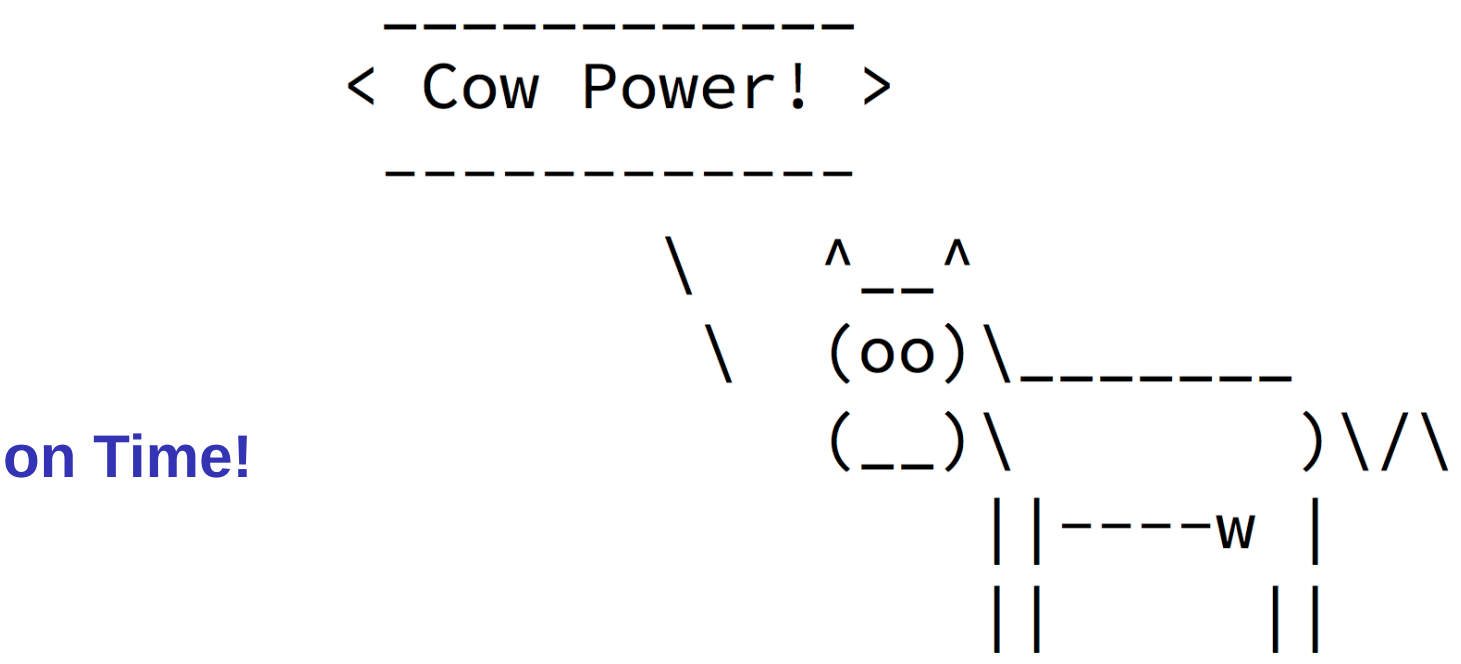
at computing
The one-stop-Linux-shop

Config Management Camp

ANSIBLE

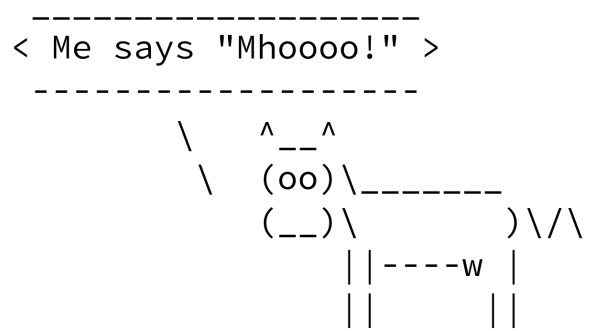
Agenda

1. **Introduction**
2. **Why**
3. **Question Time!**



$ who am i

- UNIX/Linux consultant and Trainer @ AT Computing
- UNIX Nerd (started in 1986 with SunOS 3)
- Linux Geek (started in 1992 with 0.96α)
- Scripting / programming Freak
- Configuration Management Addict
- Ansible user and contributor since 2012
- Red Hat Certified Specialist in Ansible Automation
- Member of the Ansible Organisation on Github
- Ansible Ambassador since 2015
- Free and Open Source Software Enthusiast
- HAM Operator (`pa1ton`)
- Plain Text Aficionado
- Big fan of things that just work
- . . .



Why test roles

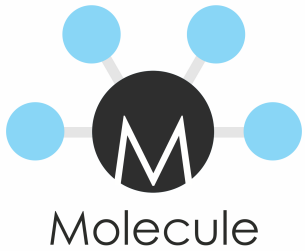
- Ansible Playbooks and roles are production code
- Check installation of wanted packages
- Process needed templates
- Check files for contents
- Run services
- Trigger handlers
- . . .

Return of investment

- Higher role quality
- Improved reliability $\Rightarrow$ Lesser interruptions
- Ensuring consistency
- Improve Robustness
- Automatic testing $\Rightarrow$ Continues Integration
- . . .

Molecule

- Open Source (of course)
- Designed to test Ansible roles
- Uses Ansible to test Ansible roles
- Supports different OS's / test scenarios
- Supports multiple virtualization providers



Setup The tools

- Molecule
- Testinfra
- Docker
- ansible-lint
- flake8

Install tools on Debian Stretch

```
# apt-get install docker-engine
# pip install ansible
# pip install ansible-lint
# pip install docker-py
# pip install molecule
# pip install yamllint
# pip install flake8
# pip install testinfra
```

Get started

Version 1

```
$ molecule init --role chrony --driver docker
```

Version 2

```
$ molecule init role                    \
   --role-name=chrony                   \
   --driver-name=docker                 \
   --verifier-name=testinfra
```

Configure `linter`

`molecule/default/molecule.yml` (*Part 1*)

```
lint:
  name: yamllint
  options:
    config-file: molecule/default/yamllint.cfg
```

`molecule/default/yamllint.cfg`

```
 extends: default
 rules:
   line-length:
     max: 132
     level: warning
     allow-non-breakable-words: true
     allow-non-breakable-inline-mappings: false
   truthy: disable
```

Configure provisioner

molecule/default/molecule.yml (*Part 2*)

```
#
# Ansible provisioner and variables
#
provisioner:
  name: ansible
  inventory:
    group_vars:                     # Mock data for Ansible group_vars
      all:
        name: Wile E. Coyote
        company: ACME
        domain: acme.net
    host_vars:                      # Mock data for Ansible host_vars
      centos7:
        foo: bar
  lint:
    name: ansible-lint
```

Configure verifier

molecule/default/molecule.yml (*Part 3*)

```
verifier:
  name: testinfra
  options:
    verbose: False
    debug: False
  lint:
    name: flake8
    options:
      ignore: "E201,E202,E203,E211,E223,E224"
```

Configure CentOS

molecule/default/molecule.yml (*Part 4*)

```
- name: centos7
  hostname: centos7
  image: couchbase/centos7-systemd
  image_version: latest
  privileged: True
  groups:
    - chrony
    - centos
  command: "/usr/sbin/init"
  # Mount cgroup volumes on Debian 9 host
  volumes:
    - "/sys/fs/cgroup:/sys/fs/cgroup:rw"
```

Run `/sbin/init` to startup `systemd`

Configure Ubuntu

molecule/default/molecule.yml (*Part 5*)

```
- name: ubuntu1604
  hostname: ubuntu1604
  image: solita/ubuntu-systemd
  image_version: latest
  privileged: True
  groups:
    - chrony
    - ubuntu
  command: "/sbin/init"
```

Run `/sbin/init` to startup `systemd`

Create tests 1

molecule/default/tests/test_default.py

```
import testinfra.utils.ansible_runner as ti

# Get all hosts from the Ansible inventory
# The inventory is generated from the molecule.yml file
inventory = '.molecule/ansible_inventory'
hosts = ti.AnsibleRunner(inventory).get_hosts('all')
```

Setup framework and get all Ansible hosts

Create tests 2

molecule/default/tests/test_default.py

```
def test_services(host):
    try:
        facts = host.ansible("setup")["ansible_facts"]
    except host.ansible.AnsibleException as exc:
        assert exc.result['failed'] is True
        assert exc.result['msg'] == 'Error getting facts'
    osfam = facts['ansible_os_family'].lower()

    if osfam == 'debian':
        present = [ host.service('chrony') ]
    elif osfam == 'redhat':
        present = [ host.service('chronyd') ]
    else:
        raise EnvironmentError("'%s' is an unsupported OS" % osfam)
```

Get all Ansible facts and check OS family

Create tests 3

molecule/default/tests/test_default.py

```
    if present:
        for this in present:
            assert this.is_running
            assert this.is_enabled
```

Check if all services are running

Problems

- Fragile environment
  - Versions of tools must match
  - Sudden, unexpected breaking of test line
- Large difference between version 1 and 2
- Documentation not always clear
- . . .

Question Time!

Questions??

Contact me

- Ton.Kersten@ATComputing.nl
- http://www.atcomputing.nl
- https://github.com/tonk
- https://speakerdeck.com/tonk
- @TonKersten on Twitter
- TKersten on IRC

Created with

- LaTeX Beamer
- Vim
- Poppler Tools
- ImageMagick
- Evince