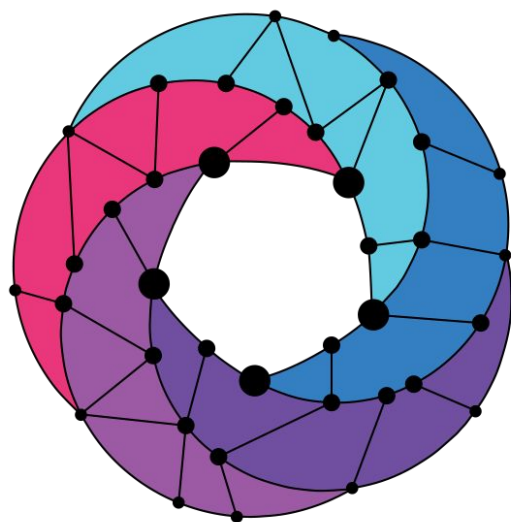


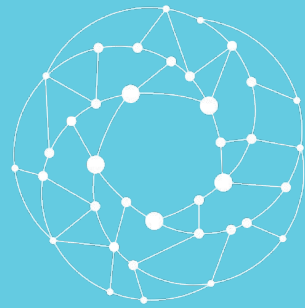


Open Source
MANO

OSM Hackfest – Session 7.2 Building a Proxy Charm

Mark Beierl (VMware)

1. Building on Proxy Charms
2. Actions
 - a. Ansible playbook
 - b. Writing playbooks
 - c. Passing variables
3. Building a Proxy charm layer



Open Source
MANO

Beyond Proxy Charms

A Charm Can Be Anything

This is a pattern, not a prescription

There are many ways to approach Charms

Use this as a start, not as the definitive solution



Metadata.yaml

Metadata.yaml includes all the high level information of our charm.

Note the name

```
name: ansible
summary: An anbsible playbook wrapper
maintainer: Name <user@domain.tld>
subordinate: false
series: ['xenial','bionic']
```

metadata.yaml

Layer.yaml

Layer.yaml: new layers apt and ansible.

```
includes:
- layer:apt
- layer:basic
- layer:vnfproxy
- layer:ansible-base
options:
  basic:
    use_venv: false
  packages:
    - ansible
    - sshpass
```

layer.yaml

Config.yaml

Config.yaml contains additional configuration

In this case, a PPA for a specific version of Ansible

```
options:  
  install_sources:  
    default: ppa:ansible/ansible-2.8
```

config.yaml

Actions.yaml

Actions.yaml: high level description actions

New action: playbook (arbitrary name, but must match reactive)

```
playbook:
  description: "Creates a file."
  params:
    filename:
      description: "The name of the file to touch."
      type: string
      default: ""
  required:
    - filename
```

actions.yaml

Reactive/ansible.py

Reactive/ansible.py - Name matches metadata.yaml

```
from charmhelpers.core.hookenv import action_get, action_fail, action_set, status_set
from charms.reactive import clear_flag, set_flag, when, when_not
import charms.libansible
```

```
@when('ansible.configured')
```

```
@when('actions.playbook')
```

```
def playbook():
```

```
    playbook_vars = {}
```

```
    playbook = action_get('script')
```

```
    result = charms.libansible.execute_playbook(playbook + '.yaml', playbook_vars)
```

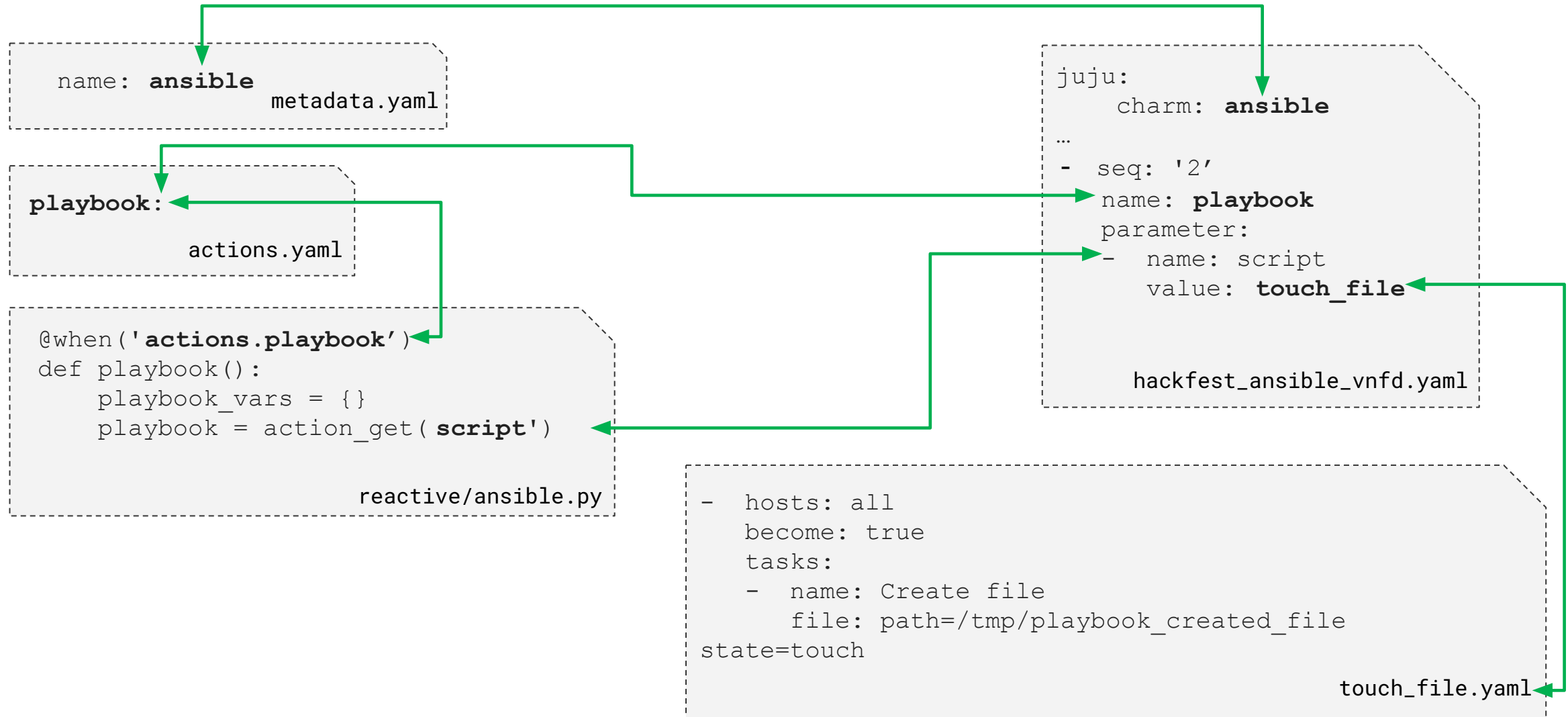
reactive/ansible.py

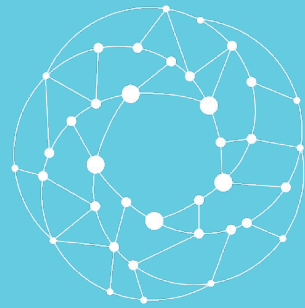
Juju: charm: name matches metadata.yaml

```
vdu-configuration:
  config-access:
    ssh-access:
      required: true
  juju:
    charm: ansible
initial-config-primitive:
  - seq: '2'
    name: playbook
    parameter:
      - name: script
        value: touch_file
      - name: filename
        value: <filename>
```

hackfest_ansible_vnfd.yaml

Piecing it Together





Open Source
MANO

Playbooks

Actions/playbook

```
$ mkdir actions
$ nano actions/playbook
$ chmod +x actions/playbook
```

```
#!/usr/bin/env python3
import sys
sys.path.append('lib')
from charms.reactive import main, set_state
from charmhelpers.core.hookenv import action_fail, action_name

set_state('actions.{}'.format(action_name()))

try:
    main()
except Exception as e:
    action_fail(repr(e))
```

actions/playbook

Note: We saw this before in the proxy charm session. Every Proxy charm action uses the same code.

Implementing the action

Append the implementation of the action to reactive/ansible.py

```
@when('ansible.configured')
@when('actions.playbook')
def playbook():
    try:
        playbook_vars = {}
        for name in ['filename']:
            playbook_vars[name] = action_get(name)
        playbook = action_get('script')
        result = charms.libansible.execute_playbook(playbook + '.yaml', playbook_vars)
    except:
        exc_type, exc_value, exc_traceback = sys.exc_info()
        err = traceback.format_exception(exc_type, exc_value, exc_traceback)
        action_fail('playbook failed: ' + str(err))
    else:
        action_set({'output': result})
    finally:
        remove_flag('actions.playbook')
```

Matches
parameters from
actions.yaml

reactive/ansible.py

A Helper: libansible.py

```
def create_hosts(cfg, hosts):
    inventory_path = '/etc/ansible/hosts'
    with open(inventory_path, 'w') as f:
        f.write('[{}]\n'.format(hosts))
        h1 = '{0} ansible_connection=ssh ansible_ssh_user={1} ansible_ssh_pass={2} ' \
            'ansible_ssh_private_key_file=~/.ssh/id_juju_sshproxy ' \
            'ansible_python_interpreter=/usr/bin/python3\n'.format(cfg['ssh-hostname'], cfg['ssh-username'],
                                                                    cfg['ssh-password'])
        f.write(h1)

def create_ansible_cfg():
    ansible_config_path = '/etc/ansible/ansible.cfg'

    with open(ansible_config_path, 'w') as f:
        f.write('[defaults]\n')
        f.write('host_key_checking = False\n')
        f.write('log_path = /var/log/ansible.log\n')

        f.write('[ssh_connection]\n')
        f.write('control_path=%(directory)s/%%h-%%r\n')
        f.write('control_path_dir=~/.ansible/cp\n')
```

Lib/charms/libansible.py

A Helper: libansible.py

```
def execute_playbook(playbook_file, vars_dict=None):
    playbook_path = find(playbook_file, '/var/lib/juju/agents/')
    cfg = config()
    with open(playbook_path, 'r') as f:
        playbook_data = yaml.load(f, Loader=yaml.SafeLoader)

    hosts = 'all'
    if 'hosts' in playbook_data[0].keys() and playbook_data[0]['hosts']:
        hosts = playbook_data[0]['hosts']

    create_ansible_cfg()
    create_hosts(cfg, hosts)
    call = 'ansible-playbook -v %s ' % playbook_path

    if vars_dict and isinstance(vars_dict, dict) and len(vars_dict) > 0:
        call += '--extra-vars '
        string_var = ''
        for v in vars_dict.items():
            string_var += '%s=%s ' % v
        string_var = string_var.strip()
        call += '"%s"' % string_var

    call = call.strip()
    result = subprocess.check_output(call, shell=True)
    lastline = result.decode('utf-8').splitlines()[-2]
    return lastline
```

lib/charms/libansible.py

Finally – The Playbook

Variable substitution in Ansible: `{{ variable }}`

```
- hosts: all
  become: true
  tasks:
    - name: Create file
      file: path='{{ filename }}' state=touch
```

playbooks/touch_file.yaml

Matches
parameters from
actions.yaml

Building the Charm

A little build script

```
export JUJU_REPOSITORY="\`pwd\`/hackfest_ansible_vnfd/charm-sources"
export CHARM_LAYERS_DIR="$JUJU_REPOSITORY/layers"
export CHARM_BUILD_DIR="\`pwd\`/hackfest_ansible_vnfd/charms"

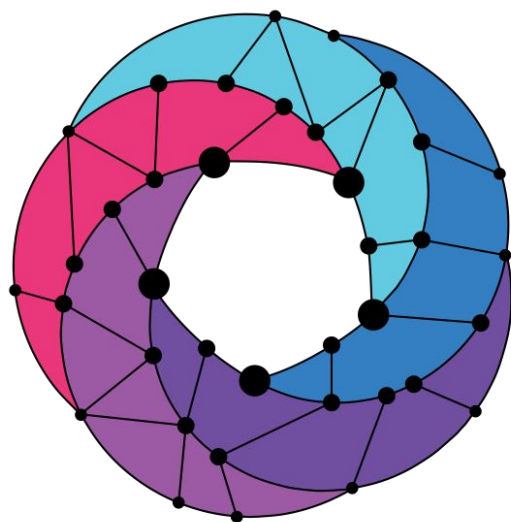
rm -rf "$CHARM_BUILD_DIR"

cd "$CHARM_LAYERS_DIR/ansible"
charm build
cd -

for file in hackfest_ansible_nsd hackfest_ansible_vnfd
do
    rm ${file}.tar.gz
    tar -czf ${file}.tar.gz ${file}/
done

osm nsd-delete hackfest_ansible_nsd
osm vnfd-delete hackfest_ansible_vnfd

osm vnfd-create hackfest_ansible_vnfd.tar.gz || exit 1
osm nsd-create hackfest_ansible_nsd.tar.gz || exit 1
```



Open Source MANO

Find us at:

osm.etsi.org
osm.etsi.org/wikipub