

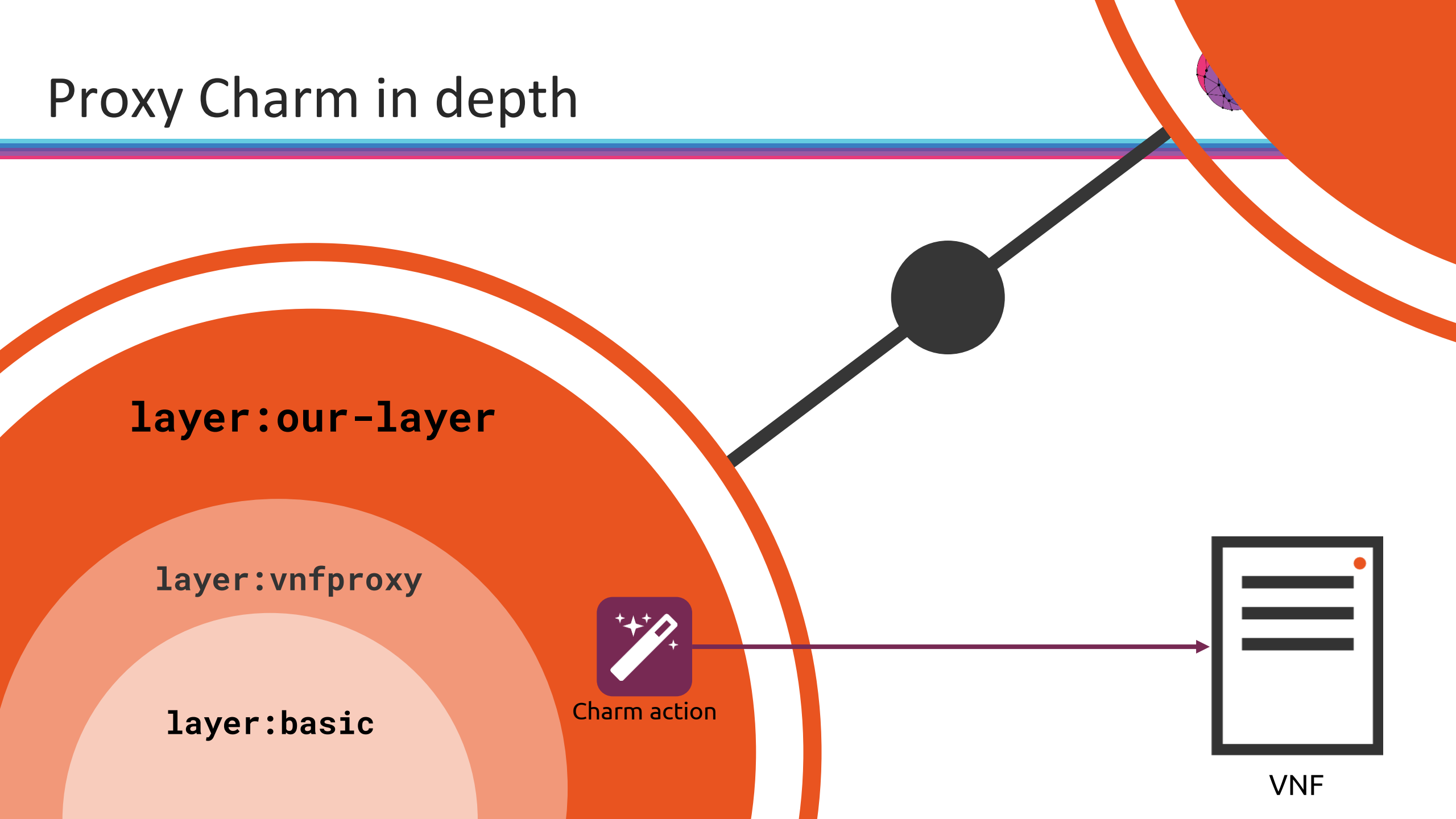
# Open Source MANO

## OSM Hackfest – Session 6 Building a Proxy Charm

Dominik Fleischmann, David Garcia (Canonical)

1. Proxy Charm in depth
2. Creating a Proxy charm layer
3. Basic configuration
  - a. metadata.yaml
  - b. layer.yaml
  - c. actions.yaml
  - d. reactive/simple.py
4. Actions
  - a. actions/touch
  - b. Implementing the action
5. Building a Proxy charm layer

# Proxy Charm in depth



# Creating a Proxy charm layer

```
$ cd JUJU_REPOSITORY/layers
$ charm create simple
$ cd simple/
$
```

→

```
$JUJU_REPOSITORY/layers
|-- simple
|  |-- README.ex
|  |-- config.yaml
|  |-- icon.svg
|  |-- layer.yaml
|  |-- metadata.yaml
|  |-- reactive
|  |  |-- simple.py
|-- tests
|  |-- 00-setup
|  |-- 10-deploy
```



Open Source  
MANO

# Basic Configuration

# Metadata.yaml

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

```
name: simple
summary: A simple VNF proxy charm
maintainer: Name <user@domain.tld>
subordinate: false
series: ['xenial', 'bionic']
```

metadata.yaml

# Layer.yaml

Layer.yaml states all the layers on which our layer is based.

```
includes: ['layer:basic', 'layer:vnfproxy']
options:
  basic:
    use_venv: false
```

layer.yaml

# Actions.yaml

Actions.yaml contains the high level description of the actions that will be implemented in our charm.

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

actions.yaml



# Reactive/simple.py

Reactive/simple.py contains the actual code of our Proxy charm.

```
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.sshproxy
```

```
@when('sshproxy.configured')
@when_not('simple.installed')
def install_simple_proxy_charm():
    """Set the status to active when ssh configured."""
    set_flag('simple.installed')
    status_set('active', 'Ready!')
```

reactive/simple.py



Open Source  
**MANO**

# Actions

# Actions/touch

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

```
#!/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/touch

# Implementing the action

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

```
...  
  
@when('actions.touch')  
def touch():  
    """Touch a file."""  
    err = ''  
    try:  
        filename = action_get('filename')  
        cmd = ['touch {}'.format(filename)]  
        result, err = charms.sshproxy._run(cmd)  
    except:  
        action_fail('command failed: {}'.format(err))  
    else:  
        action_set({'output': result})  
    finally:  
        clear_flag('actions.touch')
```

reactive/simple.py

# Building a Proxy charm layer

```
$ charm build
```

```
$ ls $JUJU_REPOSITORY/builds/simple
```

```
actions          bin              copyright        hooks            layer.yaml      Makefile
reactive         README.md       simple           tox.ini          actions.yaml    config.yaml
deps             icon.svg        lib              README.ex        metadata.yaml   test
requirements.txt wheelhouse
```

# CANONICAL