INTRODUCING LINCH-PIN

HYBRID CLOUD PROVISIONING IN ANSIBLE

CLINT SAVAGE

Senior Systems Engineer

Continuous Infrastructure - Red Hat

PREVIOUSLY

THERE WAS PROVISIONER 1.0

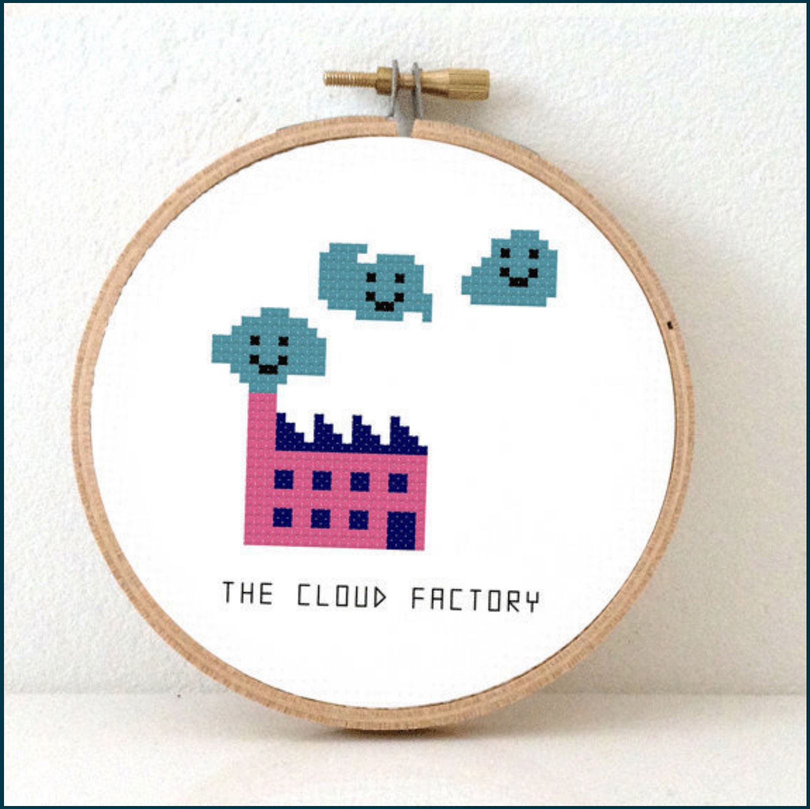
NOT SIMPLE TO USE

CI-FACTORY, CI-OPS-CENTRAL (AND OTHER DERIVATIVES)

POWERFUL!

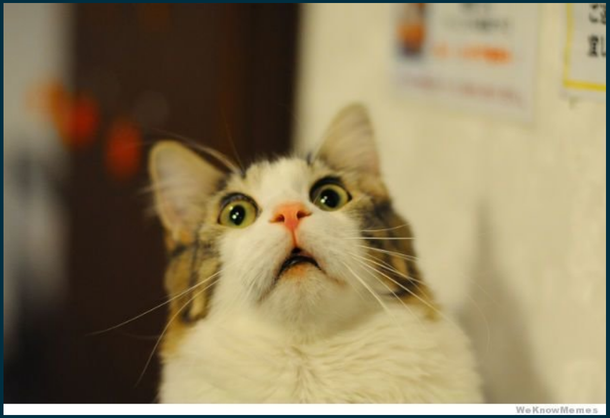
CUMBERSOME!

COMPLEX!



INSTALLATION

docs



```
mkdir -p <source code directory>; cd <source code directory>
git clone https://code.engineering.redhat.com/gerrit/ci-ops-central
cd ci-ops-central
sudo ./install.sh

#!/bin/bash

if grep -q 'Red Hat Enterprise Linux' /etc/redhat-release; then
    # Determine Version of RHEL
    export MAJOR_VER=$(egrep ' 6| 7' /etc/redhat-release | awk '{print $7}' | cu
    export MINOR_VER=$(egrep ' 6| 7' /etc/redhat-release | awk '{print $7}')

    if [ "$MAJOR_VER" == "6" ]; then
        echo -n "Release and Optional Repos"
        export RHEL_RELEASE=http://download.eng.bos.redhat.com/released/RHEL-$MA
        export RHEL_OPTIONAL=http://download.lab.bos.redhat.com/rel-eng/latest-R
        export PKG_LIST='git python-unittest2 python-nose python-futures
        python-paramiko python-lxml python-six python-configobj python-pip
        python-argparse python-glanceclient python-keystoneclient
        python-novaclient gcc compat-gcc-34.x86_64 libffi-devel python-devel
```

CLI

docs

```
ci-ops-central/bootstrap/provision_jslave.sh \
--site=ci-osp \
--project_defaults=/path/to/project_defaults \
--topology=ci-ops-central/project/config/aio_jslave \
--ssh_keyfile=/path/to/keyfile \
--jslavename=jslave-projex-slave \
--jslaveflavor=m1.xlarge \
--jslaveimage=rhel-7.1-server-x86_64-released \
--jslave_execs=10 --jslavecreate \
--resources_file=jslave-projex-slave.json
```

NOT OPEN SOURCE

NO THANK YOU!

SIMPLE IS BETTER

- CLEANER INSTALLATION
- SIMPLE TOPOLOGIES
- OPEN SOURCE
- SIMPLE COMMAND LINE (VAGRANT-LIKE)
- SIMPLE PROVISION/TEARDOWN
- EASILY EXTENSIBLE
- COMPLETE INVENTORIES
- AND MUCH, MUCH MORE!

ENTER LINCH-PIN

EXTENSIBLE,

MULTI-CLOUD,

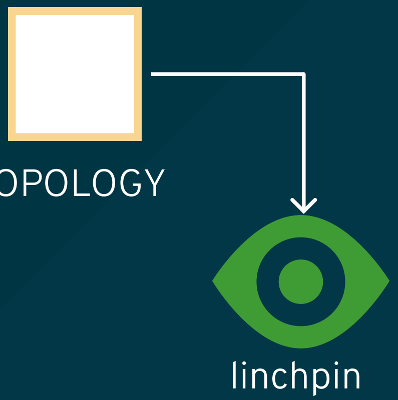
HYBRID PROVISIONER

WRITTEN IN ANSIBLE

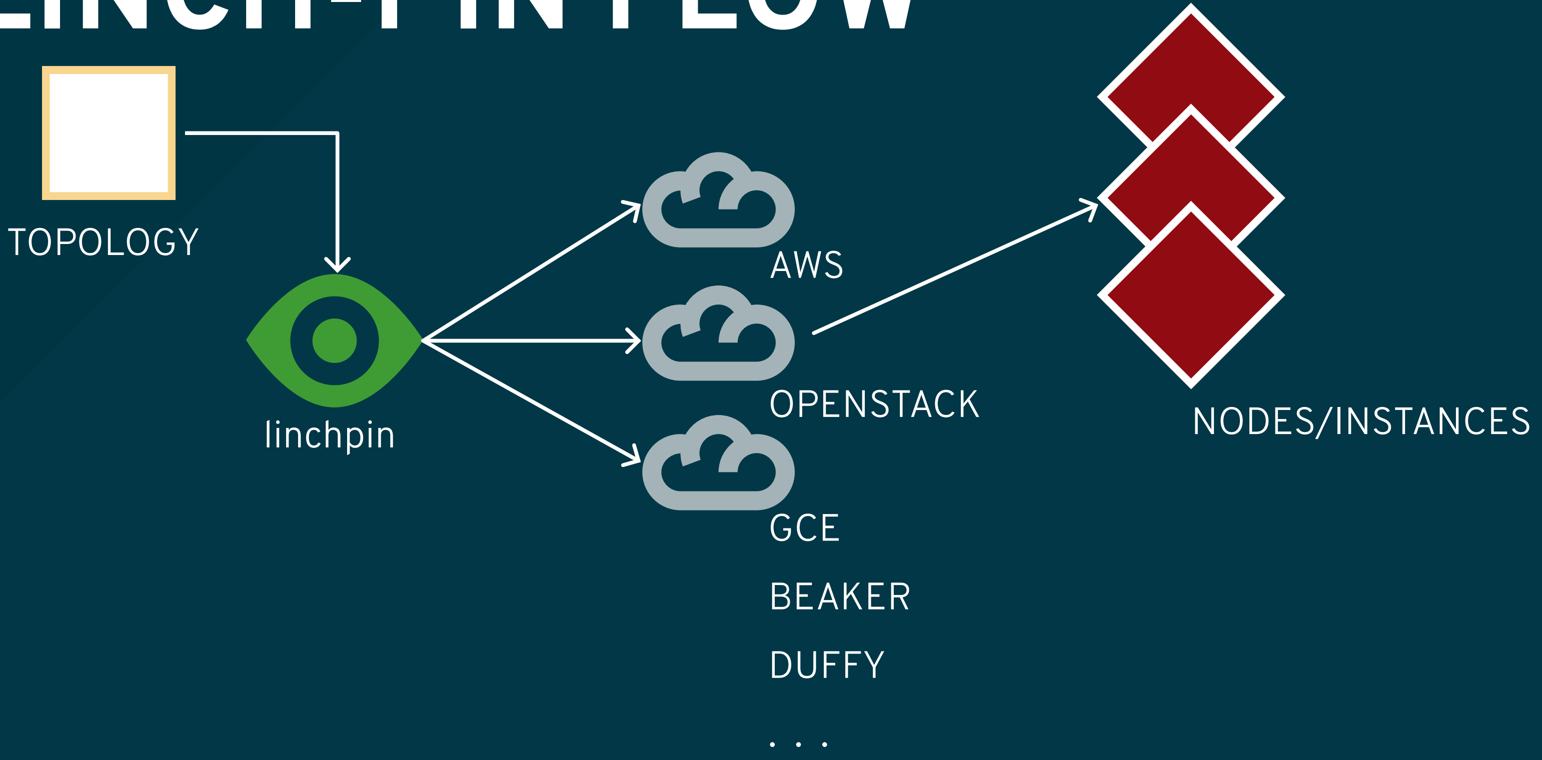
WHY ANSIBLE?

- Provides 90% of everything needed
- Community is Amazing!
- Cloud Modules
- Asynchronous
- Good docs

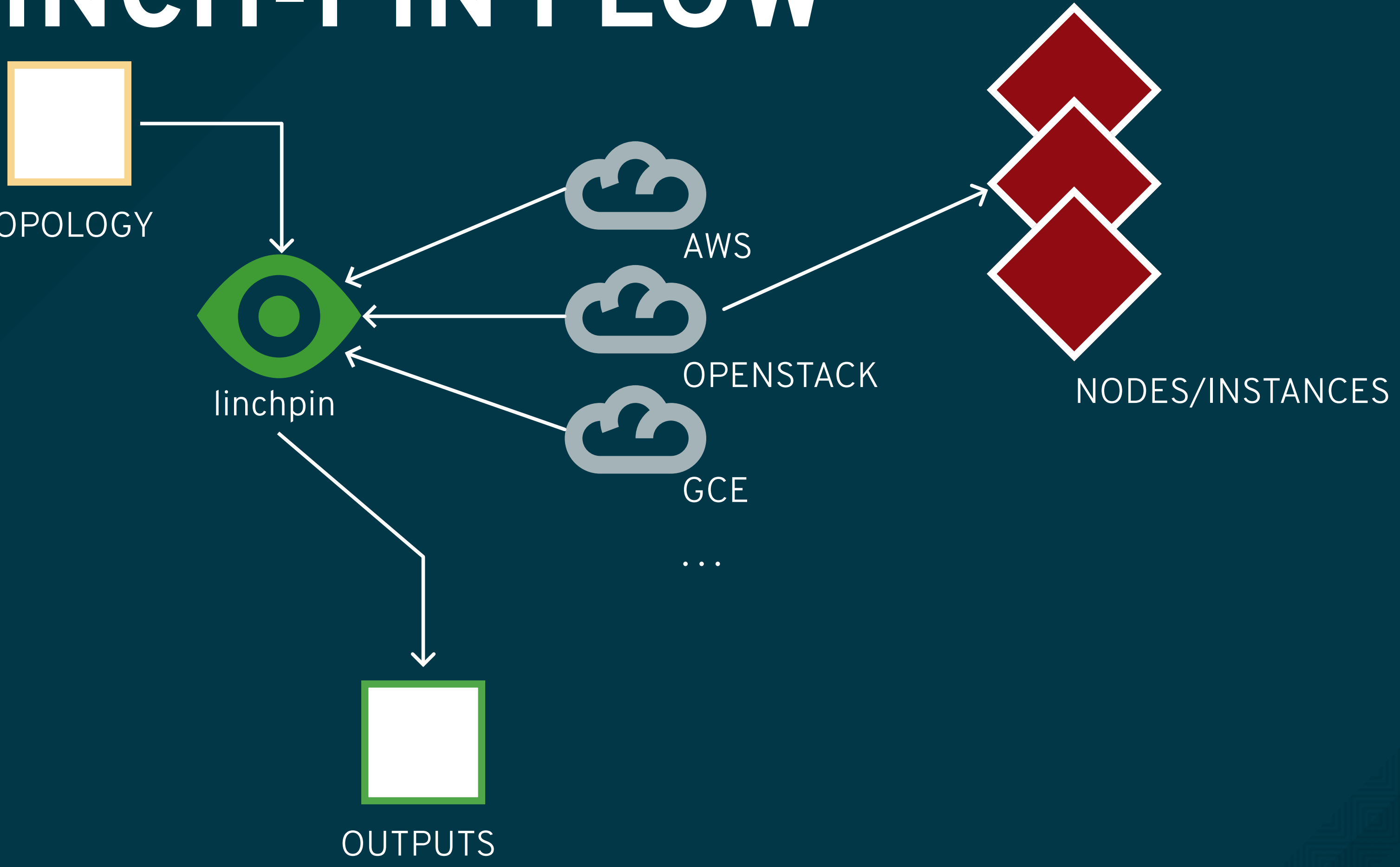
LINCH-PIN FLOW



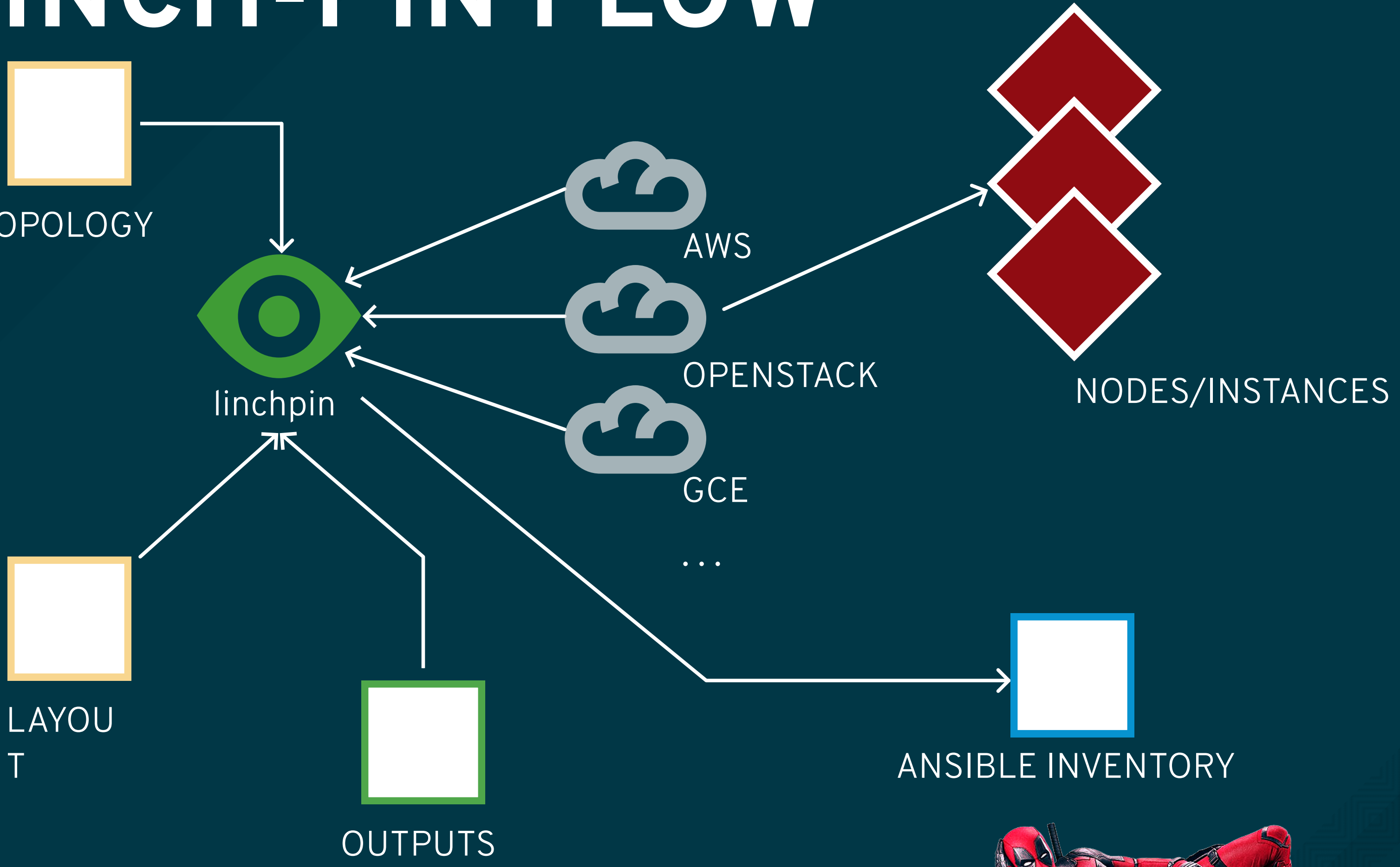
LINCH-PIN FLOW



LINCH-PIN FLOW

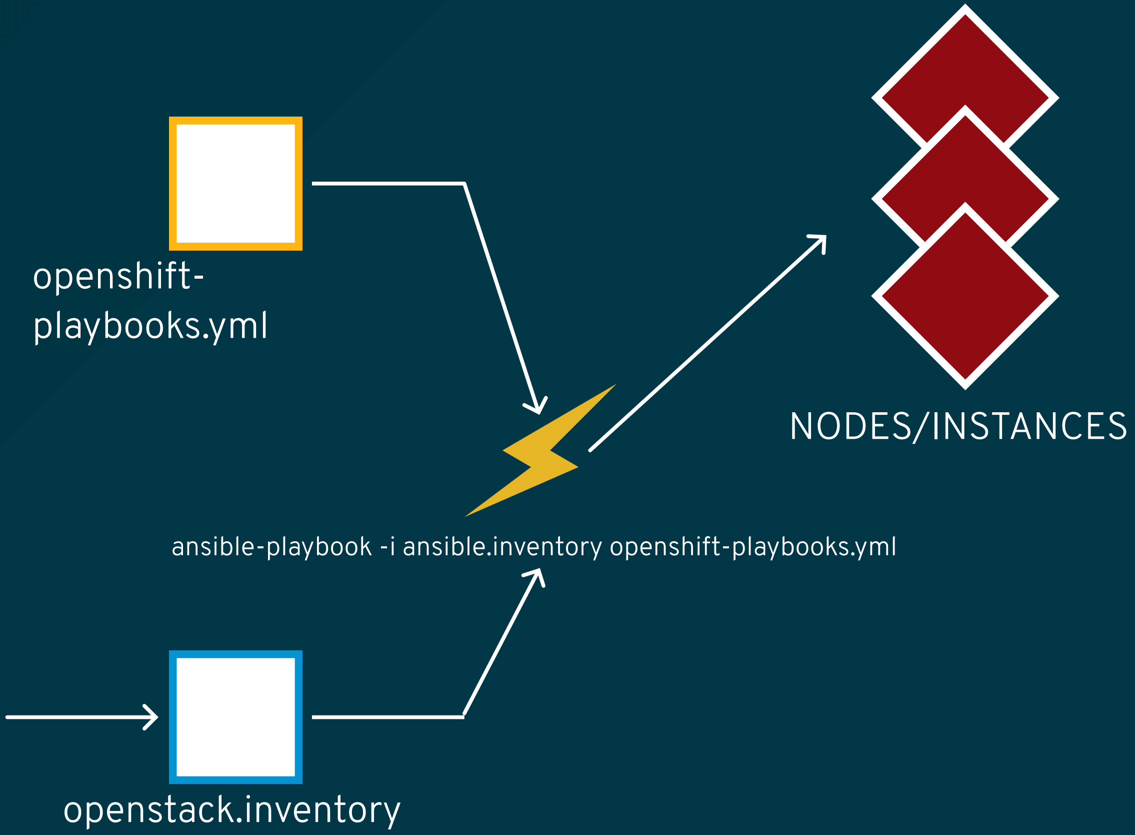


LINCH-PIN FLOW



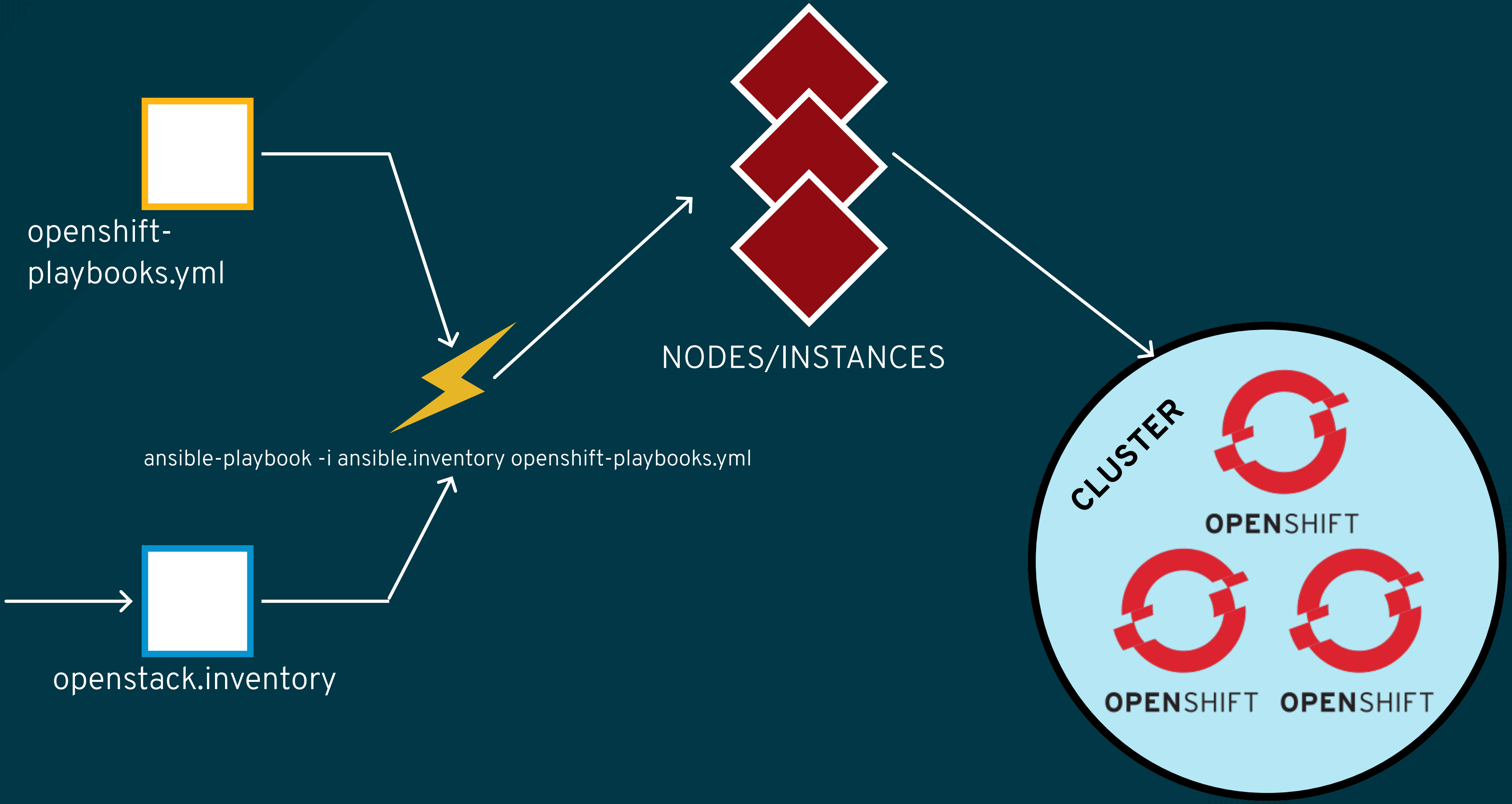
MAXIMUM EFFORT!!

USE INVENTORY



```
ansible-playbook -i ansible.inventory openshift-playbooks.yml
```

OPENSHIFT CLUSTER



SIMPLE INSTALLATION

```
$ virtualenv venv && source venv/bin/activate
$ pip install linchpin
```

* RPM packages are on the roadmap, installation will be simpler in the future

TOPOLOGY

A linch-pin topology file makes cloud customization simple

```
---
topology_name: "duffy_3node_cluster" # topology name
resource_groups:
  -
    resource_group_name: "duffy_3node_cluster"
    res_group_type: "duffy"
    res_defs:
      -
        res_name: "duffy_nodes"
        res_type: "duffy"
        version: 7
        arch: "x86_64"
        count: 3
    assoc_creds: "duffy_creds"
```

INVENTORY LAYOUT

Adding an inventory layout (or just layout) file compliments the topology by generating an ansible inventory with desired values

```
---
inventory_layout:
  vars:
    openshift_hostname: __IP__
    openshift_public_hostname: __IP__
  hosts:
    openshift-master:
      host_groups:
        - masters
        - nodes
        - OSEv3
    openshift-node:
      count: 1
      host_groups:
        - nodes
        - OSEv3
    openshift-repo-host:
      host_groups:
        - nodes
        - OSEv3
        - repo_host
  host_groups:
    OSEv3:
      vars:
        openshift_docker_additional_registries: |
            "registry.example.com"
```

PINFILE

The PinFile provides a simple map to both the topology and layout.

```
$ cat PinFile
---
ae2e-test:
  topology: simple-ae2e-cluster.yml
  layout: openshift-3node-cluster.yml

e2e:
  topology: simple-e2e-os-cluster.yml
  layout: openshift-3node-cluster.yml

$ tree
├── layouts
|    └── openshift-3node-cluster.yml
├── PinFile
└── topologies
    ├── duffy-3node-cluster.yml
    ├── simple-ae2e-cluster.yml
    └── simple-e2e-os-cluster.yml
```

LINCHPIN CLI

The linchpin command line interface simplifies the management of clouds

- linchpin init

  Generate PinFile

- linchpin {rise|drop}

  Provision / Teardown (topology, outputs required)

- linchpin {config|validate}

  Generate a linchpin_config.yml (useful for developing linchpin)

  Validate PinFile, Topology, and Layout files (in development)

- linchpin {topology|layout} {get|list}

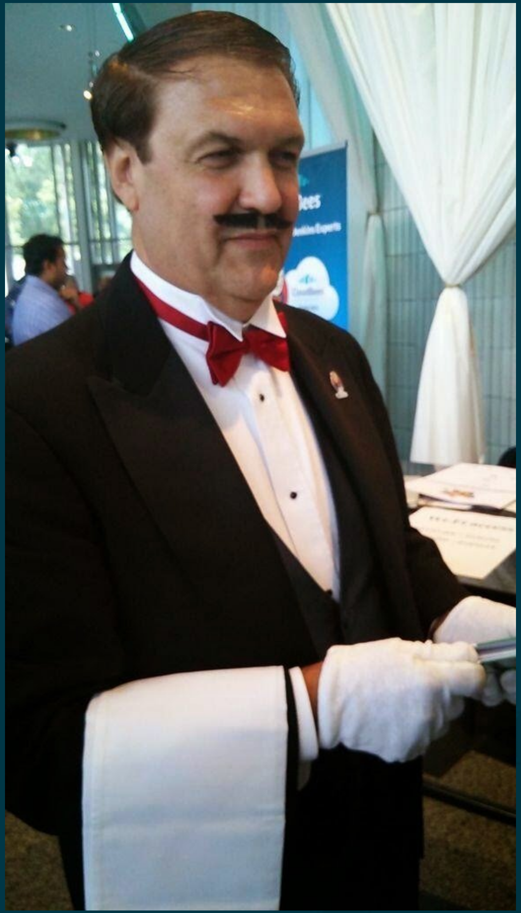
  Get topologies or layouts from upstream servers (in development)

  List local topologies or layouts

DEMONSTRATION

LEVERAGING LINCH-PIN IS A CINCH

JENKINS SLAVE CONFIGURATION FOR LINCH-PIN



WHAT IS CINCH?

- Linch-Pin provisioning + automated jenkins slave configuration
- Open source, hosted on GitHub for easier use for upstream
- Optimized for the Central CI use case
- All using Ansible!

LEVERAGING LAYOUTS

Cinch extends this concept

```
---
.. layout on previous slide ..

    certificate_authority:
      vars:
        certificate_authority_urls:
          - "https://password.corp.redhat.com/legacy.crt"
          - "https://password.corp.redhat.com/RH-IT-Root-CA.crt"
          - "https://engineering.redhat.com/Eng-CA.crt"
     repositories:
      vars:
        rhel_base: "http://pulp.dist.prod.ext.phx2.redhat.com/content/dist/rhel/server/7/7Server"
```

CURRENT & FUTURE

CURRENT

PIP & RPM Packages available (RPM before Feb 2017)

Asynchronous Provisioning

Simple CLI

Linch-Pin Python API

FUTURE

Satellite

OpenShift Provisioning

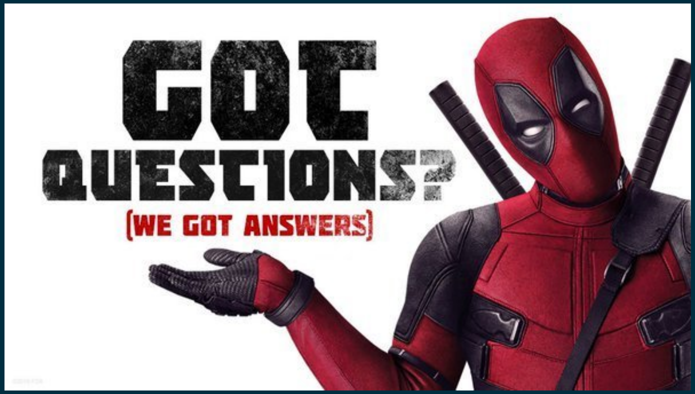
Vagrant Plugin

Hooks

Node up-scaling

Cloud Bursting

QUESTIONS? COMMENTS?



LINCH-PIN

github.com/CentOS-PaaS-SIG/linch-pin.git

linch-pin.rtfd.io

CINCH

github.com/RedHatQE/cinch.git

redhatqe-cinch.rtfd.io

IRC CHANNELS AND MAILING LISTS COMING SOON!