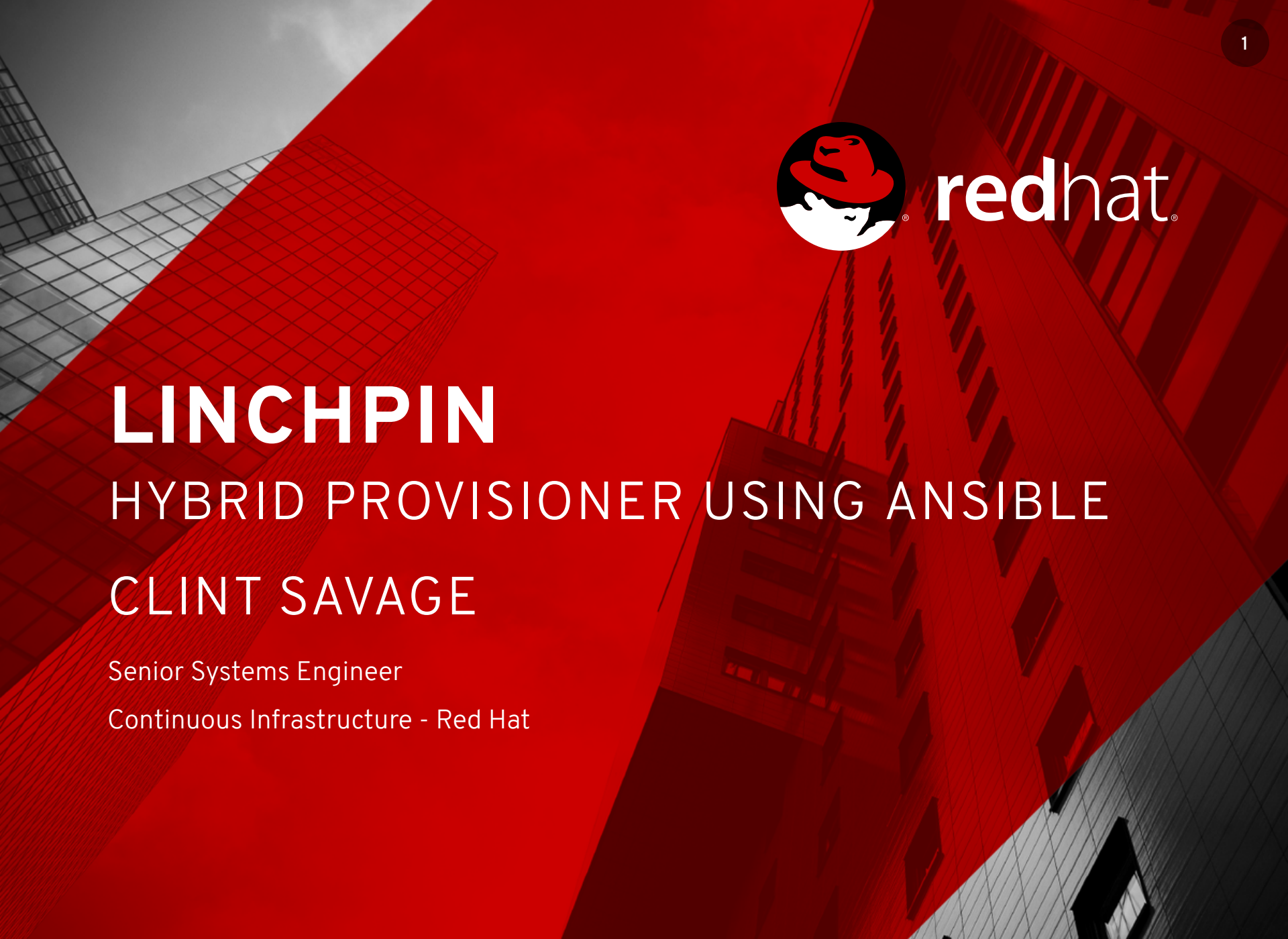


LINCHPIN

HYBRID PROVISIONER USING ANSIBLE

CLINT SAVAGE

Senior Systems Engineer

Continuous Infrastructure - Red Hat

AGENDA

- HISTORY
- LINCHPIN (PROVISIONER 2.0)
- BEYOND PROVISIONING
- DEMONSTRATION
- LINCHPIN NOW AND FUTURE
- Q&A

HISTORY

PROVISIONER 1.0

- **COMPLEX INSTALLATION**

```
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 $7}' | cut -d. -f
    export MINOR_VER=$(egrep ' 6| 7' /etc/redhat-release | awk '{print $7}')

    if [ "$MAJOR_VER" == "6" ]; then
        echo -n "Release and Optional Repos"
        export RHEL_RELEASE=http://download.eng.bos.redhat.com/released/RHEL-$MAJOR_VER/
        export RHEL_OPTIONAL=http://download.lab.bos.redhat.com/rel-eng/latest-RHEL-$MAJ
        export PKG_LIST='git python-unittest2 python-nose python-futures
        python-paramiko python-lxml python-six python-configobj python-pip
```

HISTORY

PROVISIONER 1.0

- **COMPLEX INSTALLATION**
- **CUMBERSOME CLI**

```
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
```

HISTORY

PROVISIONER 1.0

- **COMPLEX INSTALLATION**
- **CUMBERSOME CLI**
- **FRAGMENTED TOOLING**

```
$ ls ci-factory/bootstrap

ansible_exec.sh          provision_jmaster.sh    run_janitor.sh
ci-ops-tests.sh          provision_jslave.sh     teardown_jslave.sh
manage_jenkins_jobs.sh   provision_resources.sh  teardown_resources.sh

$ ci-factory/bootstrap/provision_jslave.sh
Error: --project_defaults not specified exiting...

$ ci-factory/bootstrap/provision_resources.sh \
--project_defaults=/path/to/project_defaults
Error: --topology not specified exiting...
```

PROVISIONING GOALS

IS THE GOAL CLOUD ORCHESTRATION?

- IF NOT, WHAT IS THE GOAL?

WHAT'S OUT THERE ALREADY?

- WHAT IS THE PROJECT'S FOCUS?
- DOES IT MEET OUR REQUIREMENTS?
- IS THERE ALREADY A COMMUNITY AROUND IT?

WHAT DOES SOMETHING NEW COST?

- MAYBE NOT IN MONEY, SO MUCH AS IN TIME

MOVING ON UP

PROVISIONER 1.0

- **COMPLEX INSTALLATION**
- **CUMBERSOME CLI**
- **FRAGMENTED TOOLING**

PROVISIONER 2.0
(AKA LinchPin)

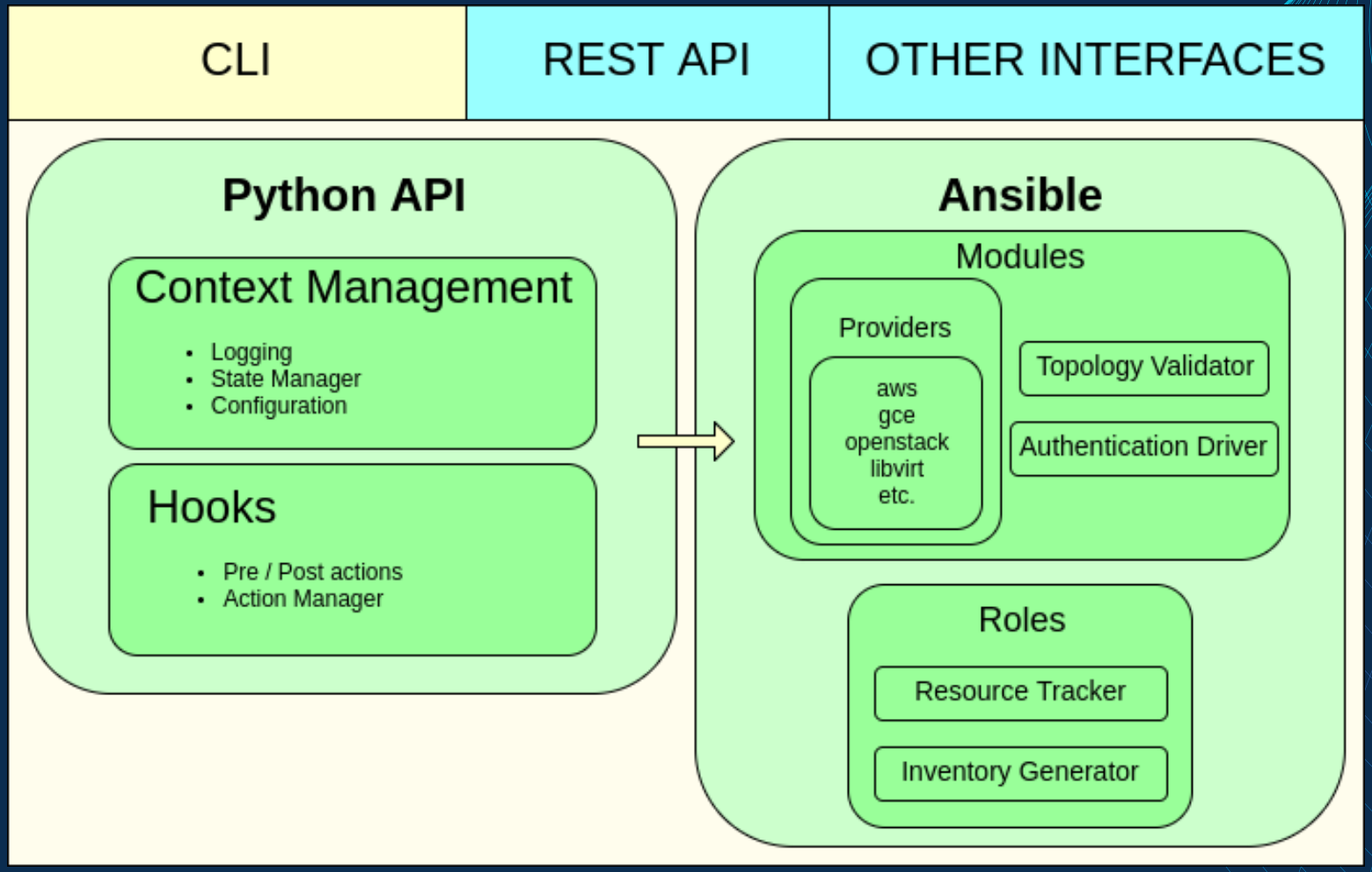
- **POWERFUL**
- **SIMPLE**
- **EXTENSIBLE**

LINCHPIN

SIMPLICITY IS THE GOAL

- SOLID ARCHITECTURE
- DECLARATIVE INFRASTRUCTURE
- CUSTOMIZABLE INVENTORIES
- CLEAN INSTALLATION
- SIMPLE AND EFFECTIVE COMMAND-LINE
- EASILY EXTENSIBLE

SOLID ARCHITECTURE



TOPOLOGY

A linch-pin topology file makes cloud customization simple

```
---
topology_name: "dummy_cluster"
resource_groups:
  -
    resource_group_name: "dummy"
    resource_group_type: "dummy"
    resource_definitions:
      -
        name: "web"
        type: "dummy_node"
        count: 3
```

LAYOUT

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

```
---
inventory_layout:
  vars:
    hostname: __IP__
  hosts:
    example-node:
      count: 3
      host_groups:
        - example
```

PINFILE

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

```
---
dummy1:
  topology: dummy-cluster.yml
  layout: dummy-layout.yml
```

WORKSPACE

To manage these resources, a workspace is created. By default, it's the current directory, but can be customized.

```
[workspace]$ tree
.
├── layouts
|   └── dummy-layout.yml
├── PinFile
└── topologies
    └── dummy-cluster.yml
```

CLEAN INSTALLATION

$ pip install linchpin

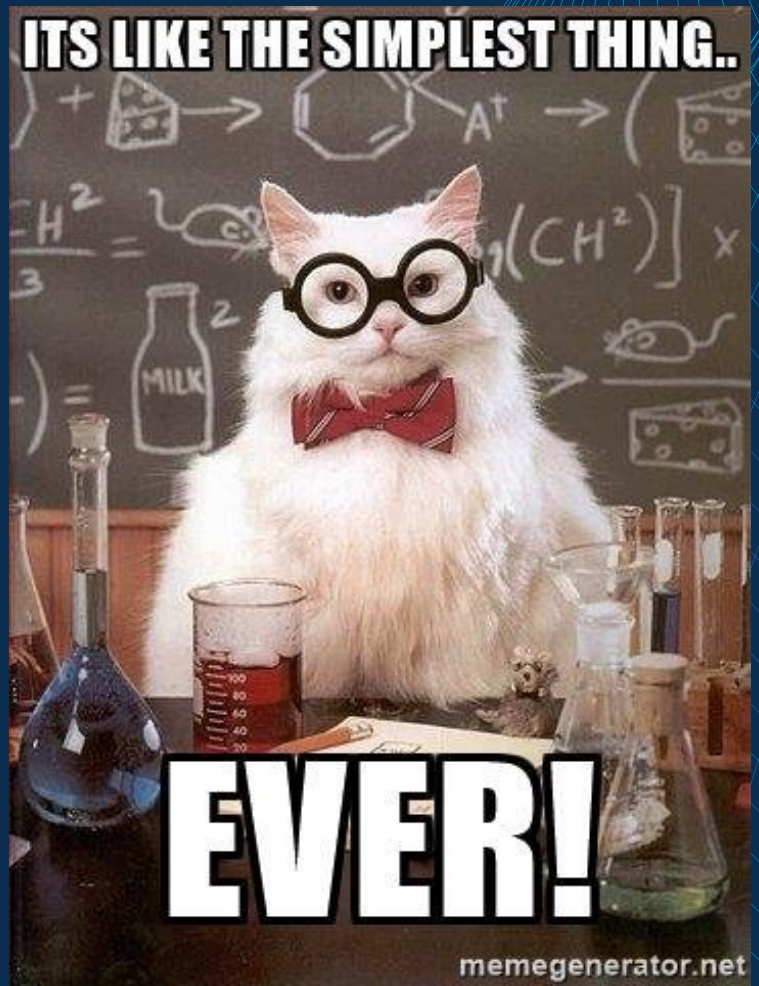
If you want to make it harder, add a virtualenv.

Even then, it's pretty simple.

$ mkvirtualenv venv

.. snip ..

$ pip install linchpin



SIMPLE AND EFFECTIVE CLI

Using the Click library provided a simple and powerful usage guide without having to read documentation.

```
$ linchpin
Usage: linchpin [OPTIONS] COMMAND [ARGS]...

  linchpin: hybrid cloud orchestration

Options:
  -c, --config PATH     Path to config file
  -w, --workspace PATH  Use the specified workspace if the familiar Jenkins
                        $WORKSPACE environment variable is not set
  -v, --verbose         Enable verbose output
  --version             Prints the version and exits
  --creds-path PATH     Use the specified credentials path if CREDS_PATH
                        environment variable is not set
  -h, --help            Show this message and exit.

Commands:
  init     Initializes a linchpin project.
  up       Provisions nodes from the given target(s) in...
  destroy  Destroys nodes from the given target(s) in...
```

EASILY EXTENSIBLE

Providers can be added via the schema, and ansible roles. From there, a just a topology is needed to provision the new provider.

CORE PROVIDERS

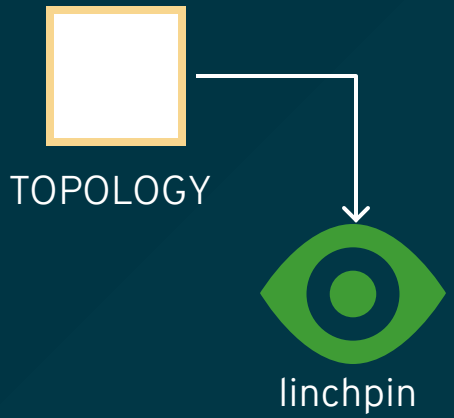
ADDITIONAL PROVIDERS

- Beaker
- Openshift
- Duffy
- Rackspace
- Dummy (testing tool)

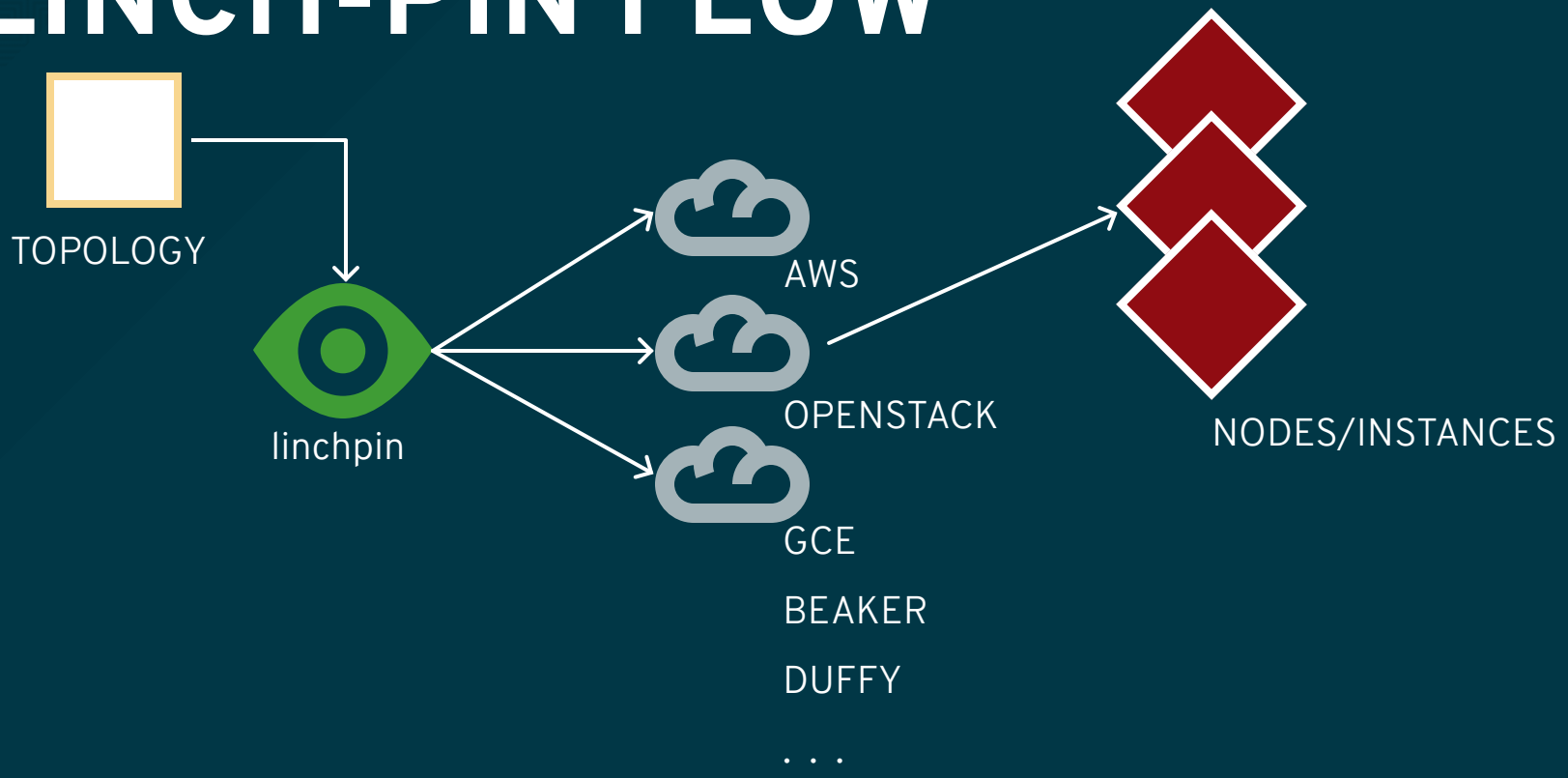
PLANNED PROVIDERS

- Azure
- Digital Ocean
- Docker / Atomic Host
- RHEV/oVirt

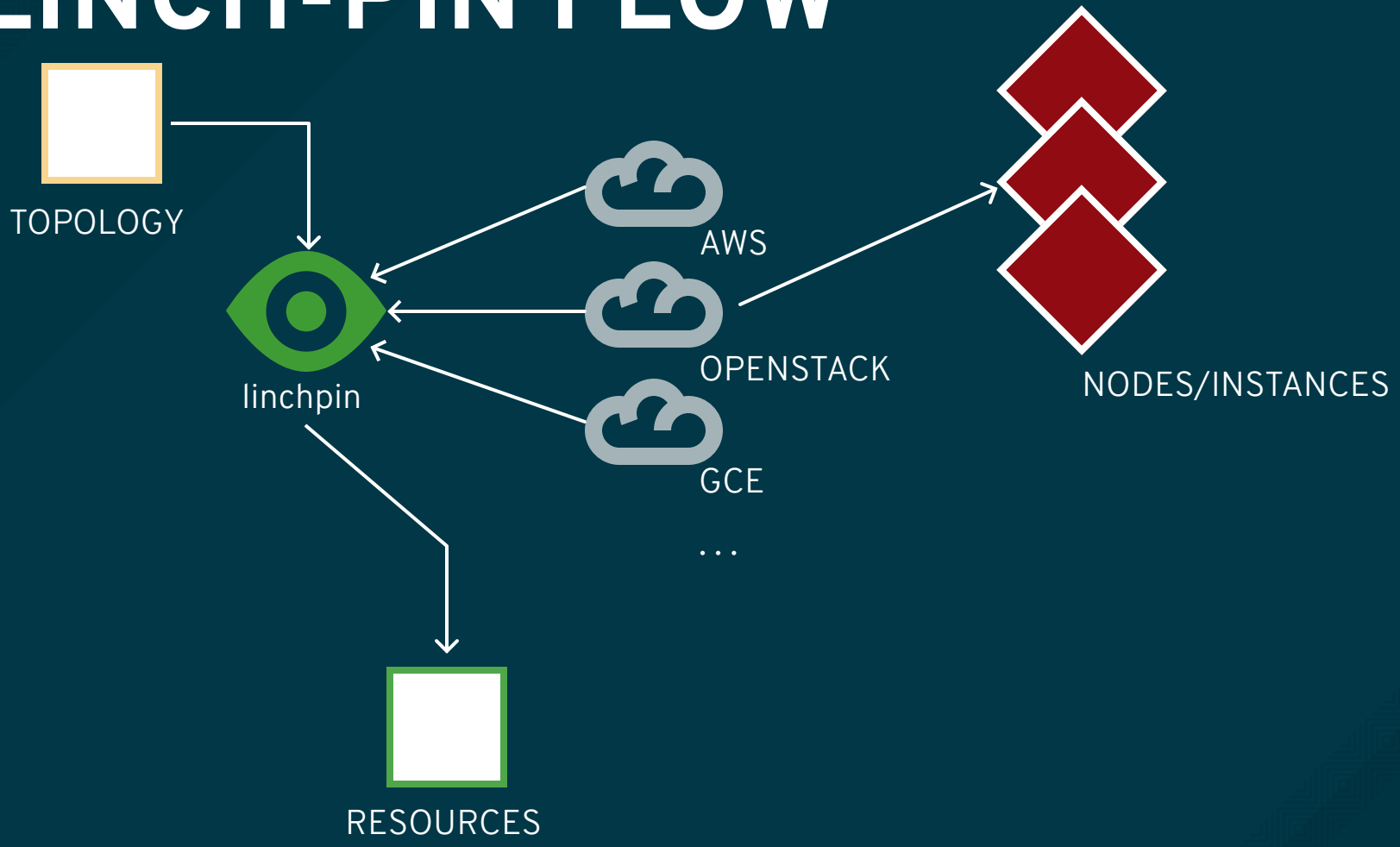
LINCH-PIN FLOW



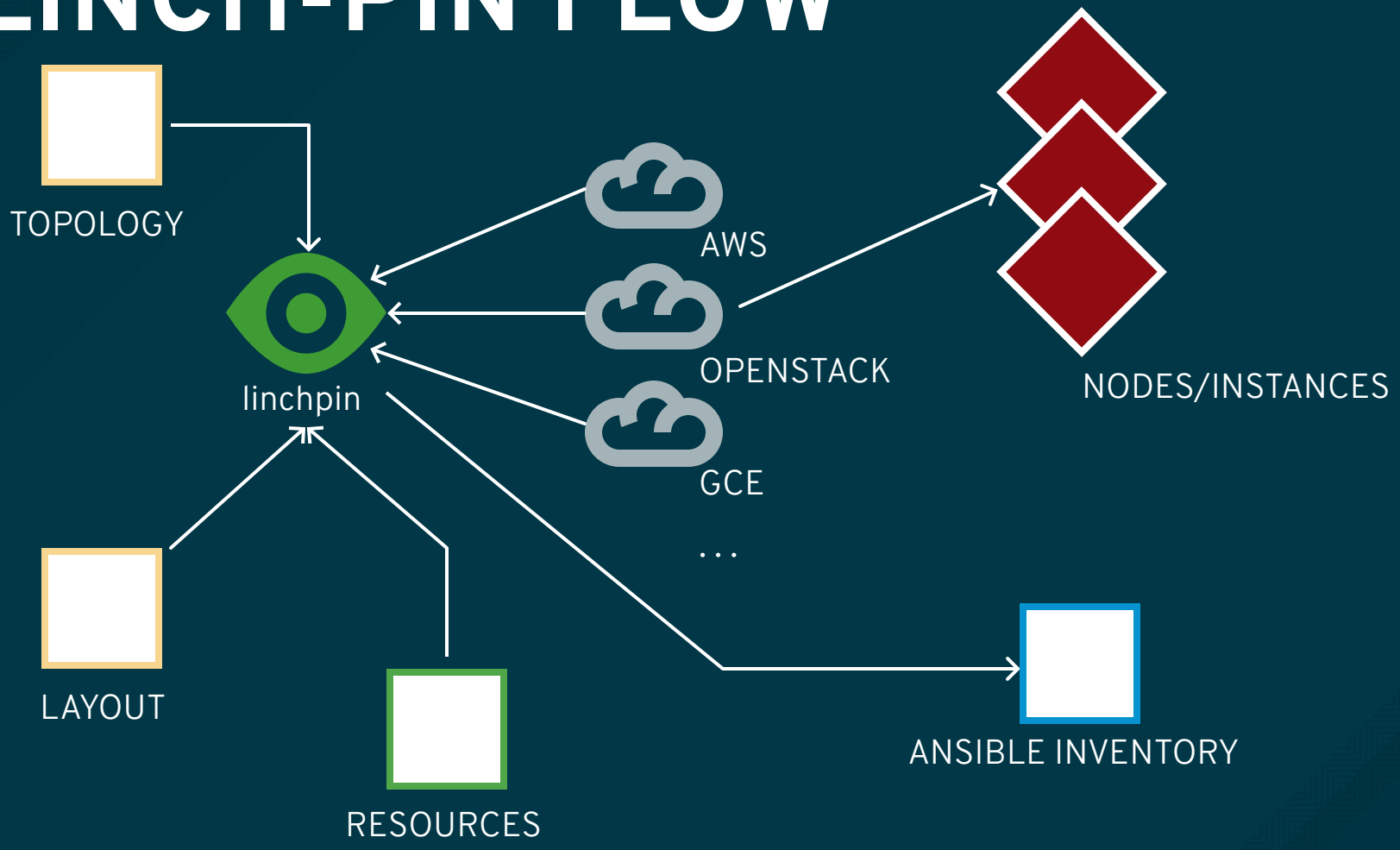
LINCH-PIN FLOW



LINCH-PIN FLOW



LINCH-PIN FLOW



BEYOND PROVISIONING

DO MORE WITH LINCHPIN!

AUTH DRIVER

The Authentication Driver enables external authentication to be applied to certain providers.

HOOKS

Hooks enable additional configuration options before, and after certain states in LinchPin.

ADDING CREDENTIALS

Adding credentials requires extending the topology

```
---
topology_name: os-test
resource_groups:
  -
    resource_group_name: lp-test
    resource_group_type: openstack
    resource_definitions:
      - name: resource
        type: os_server
        flavor: m1.small
        image: rhel-7.2-server-x86_64-released
        count: 10
        keypair: ci-factory
        networks:
          - atomic-e2e-jenkins-test
        fip_pool: 10.8.172.0/22
    credentials:
      filename: clouds2.yaml
      profile: ci-osp
```

HOOKS

Adding Hooks requires extending the PinFile

```
---
openstack-test:
  topology: openstack-cluster.yml
  layout: openstack-test.yml
  hooks:
    postup:
      - name: openshift
        type: ansible
        context: True
        actions:
          - playbook: make-cluster.yaml
```

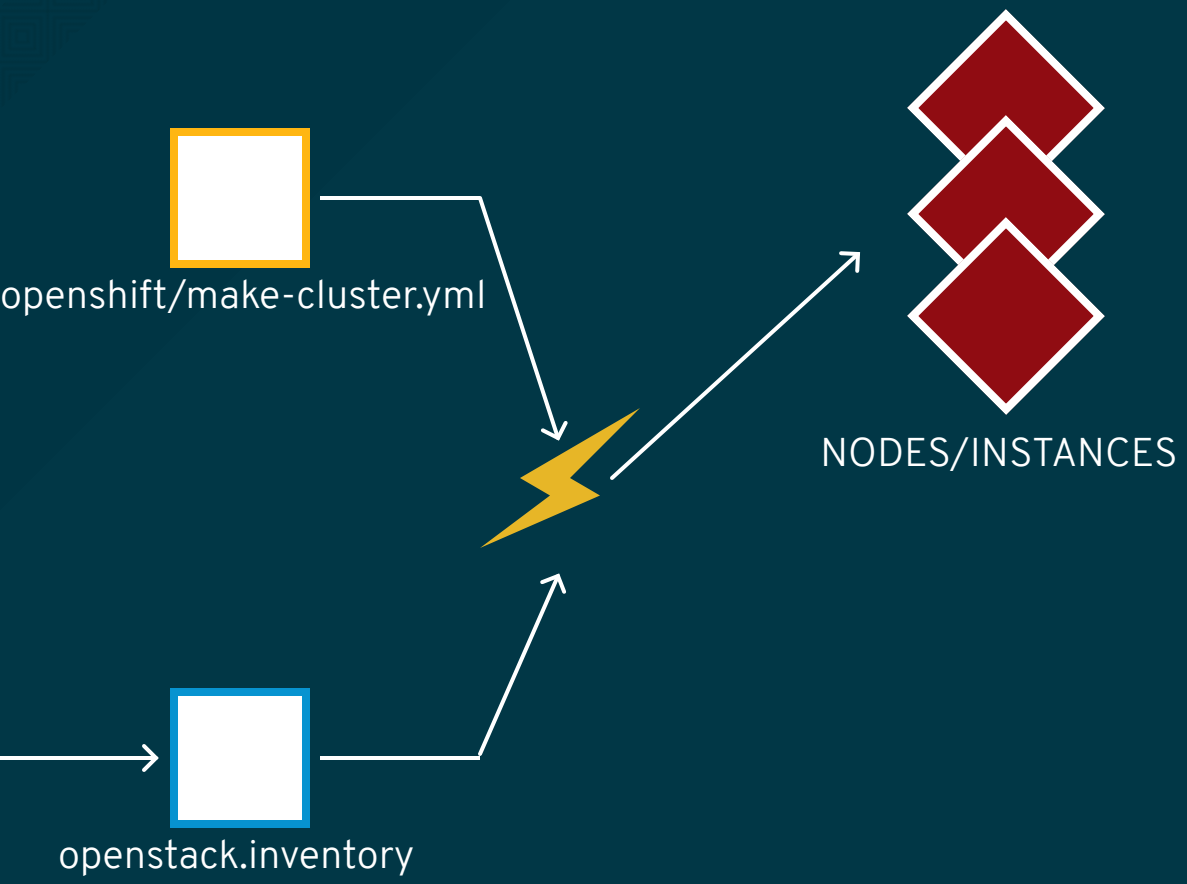
Hooks provides many different types, including custom types. The built-in types are: shell, ansible, python, ruby, and nodejs.

HOOKS

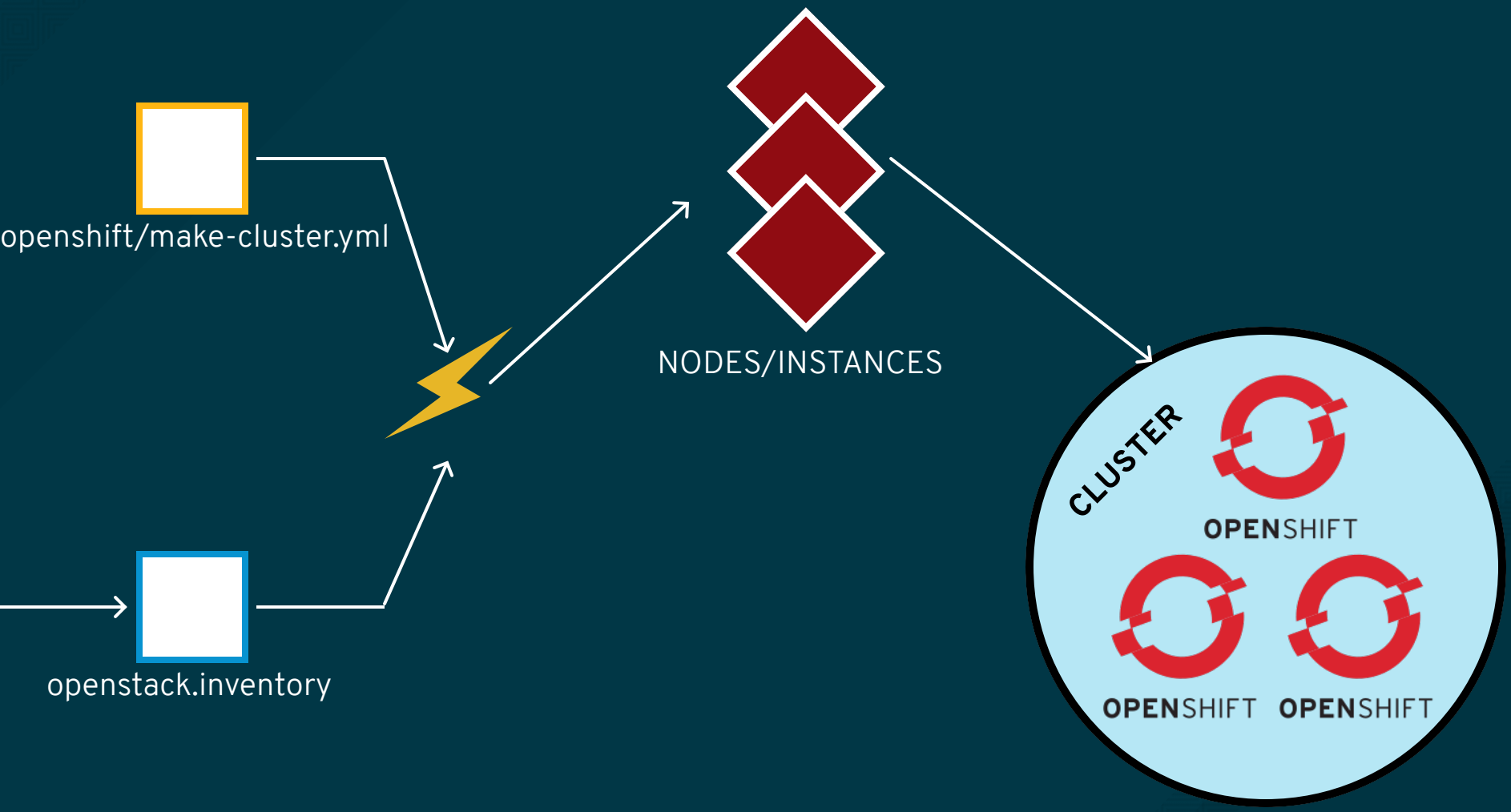
Hooks live in the workspace

```
$ tree
.
├── hooks
│   └── ansible
│       └── openshift
│           └── make-cluster.yaml
├── layouts
│   ├── dummy-layout.yml
│   └── openstack-test.yml
├── PinFile
└── topologies
    ├── dummy-cluster.yml
    └── openstack-cluster.yml
```

USE INVENTORY



OPENSHIFT CLUSTER



DEMONSTRATION

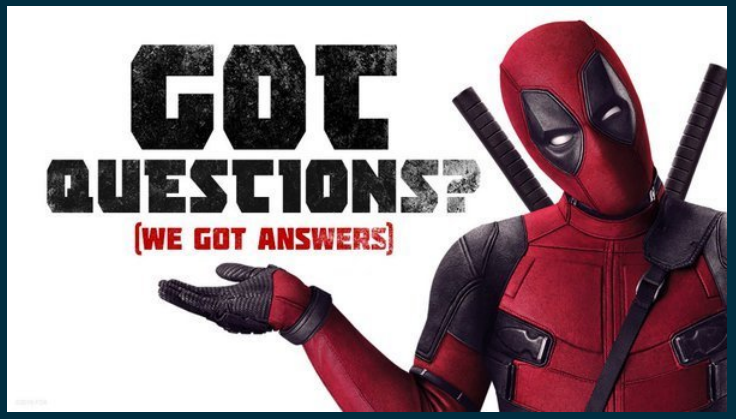
GOALS FOR 1.0

TARGET RELEASE DATE:
31 MAY 2017

ACTUAL RELEASE DATE:
01 JUNE 2017

FEATURE	COMPLETED
SATELLITE	NO
VAGRANT PLUGIN	NO
HOOKS	YES!
NODE UP-SCALING	YES!
CLOUD BURSTING	NO
ENABLE CROSS-CLOUD CLUSTERING	YES!
TOPOLOGY GENERATION	YES!
CUSTOMIZABLE $WORKSPACE OPTION	YES!
CLEANER PYTHON INSTALLATION PROCESS	YES!

QUESTIONS?
COMMENTS?



github.com/CentOS-PaaS-SIG/linchpin.git

linchpin.rtfd.io

@herlo

#linchpin on freenode