CLUSTER AS CODE

FROM SOURCE CODE TO RUNNING MESOS CLUSTER

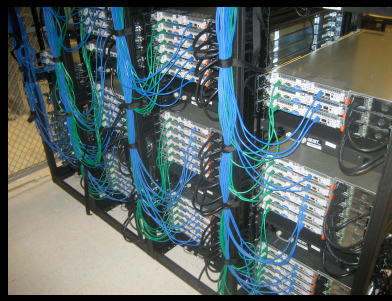
STEPHAN HOERMANN

COMMONWEALTH BANK OF AUSTRALIA

BACKGROUND

- Large financial institution in Australia
- 4 person team - Started early this year
- Bare metal hardware
- Hadoop workload and PaaS

What do we want?



WHAT DO WE REALLY WANT?

- Build and manage servers and clusters deterministically
- Immutable infrastructure
- All our configuration/changes in source control
- Ability to test our changes
- Abstractions to reason about clusters

HIGH LEVEL APPROACH

1. Create OS images with configuration for each role
2. Copy OS image on machine
3. Use Cloud Config to provide machine specific configuration
4. Change OS image and redeploy

WHY NOT …?

OUR STACK

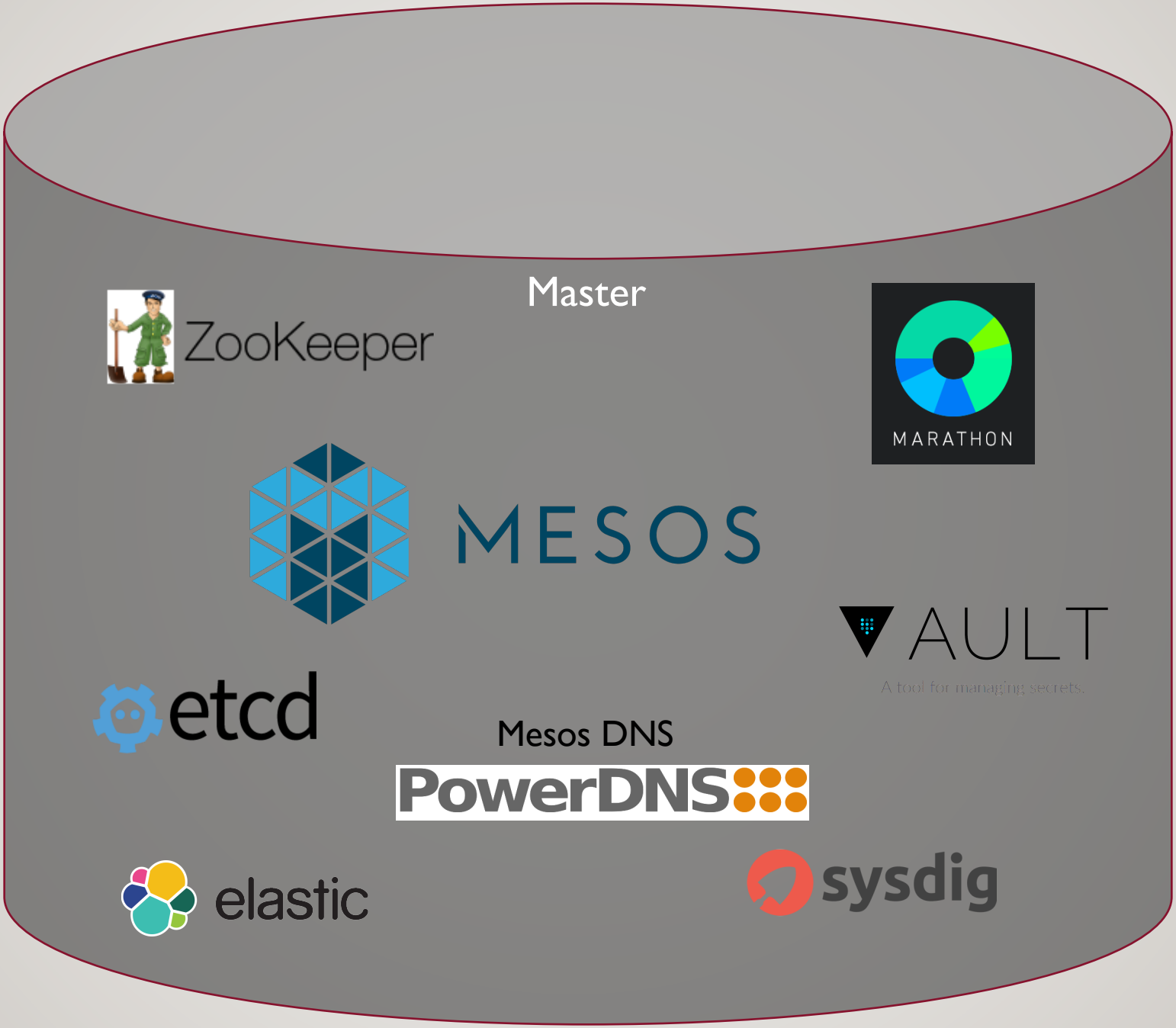
- Physical hardware
- Ubuntu
- Mesos
- Marathon
- Calico
- Docker
- Mesos DNS
- Power DNS
- Elastic stack
- Sysdig
- Vault
- Openstack Ironic

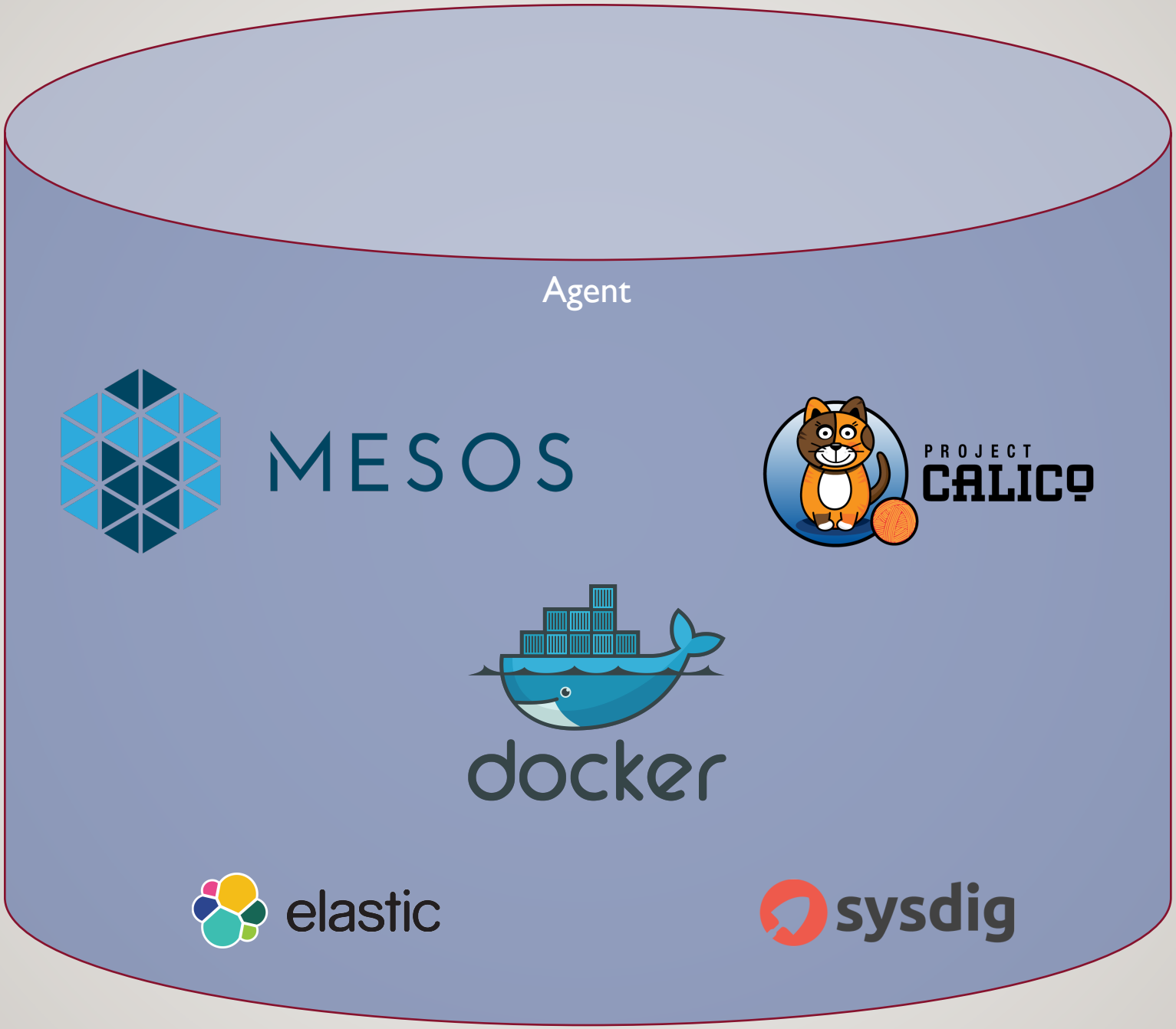
HOW DO WE DO IT?

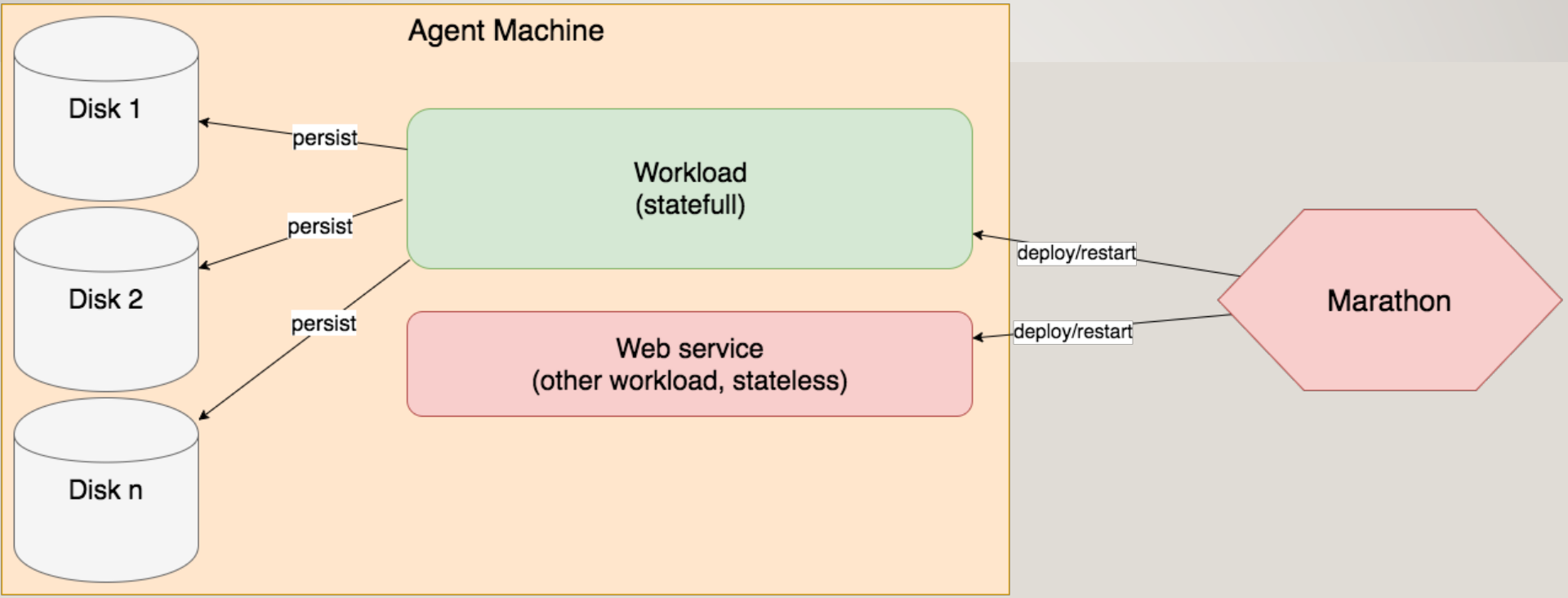
Two key parts:

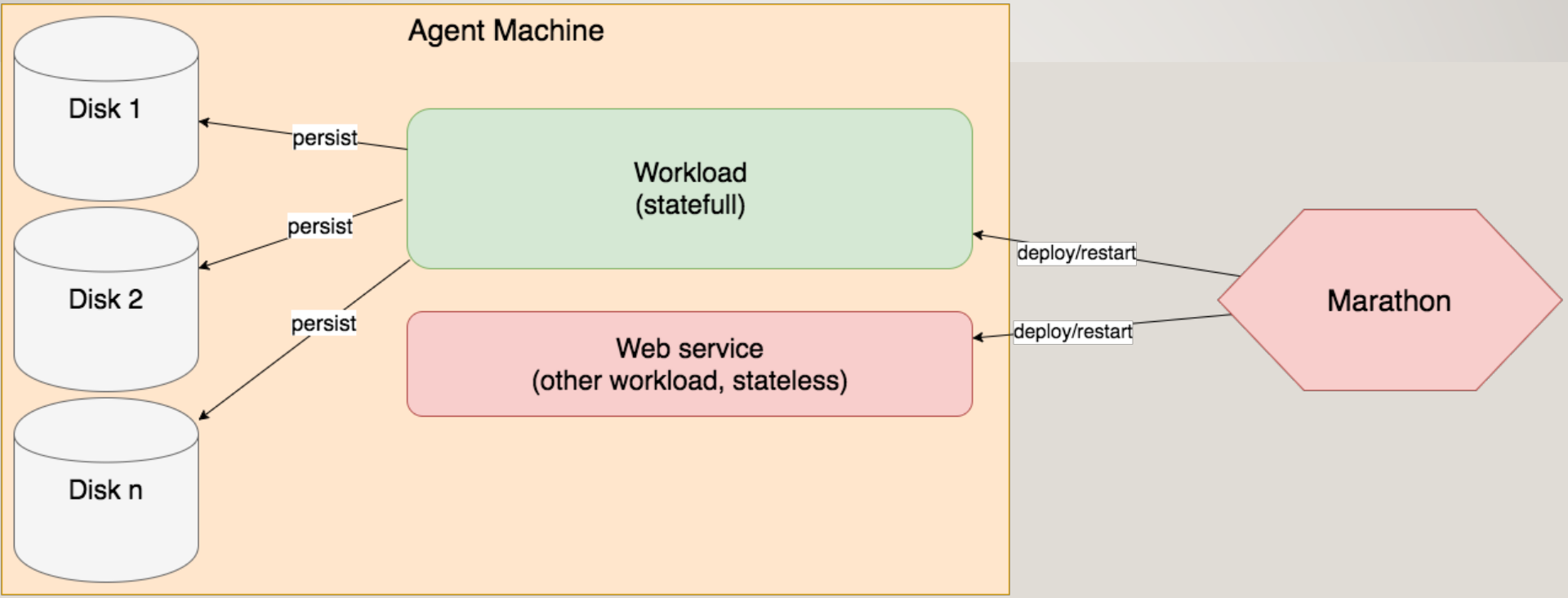
- Build OS images
- Deploy/Orchestrate clusters

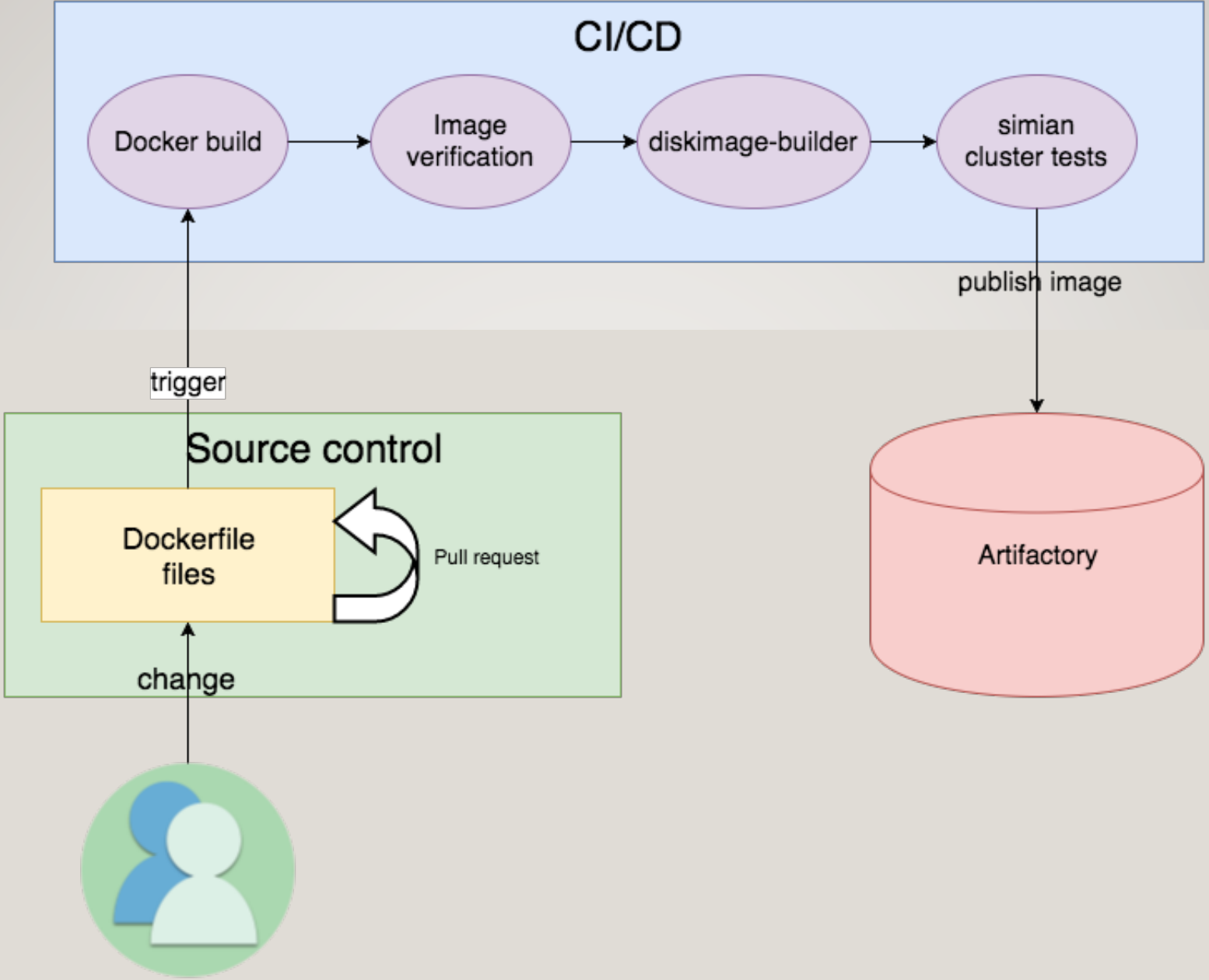
BUILD OS IMAGES


Master
ZooKeeper
MARATHON
MESOS
AULT
A tool for managing secrets.
etcd
Mesos DNS
PowerDNS
elastic
sysdig


Agent
MESOS
PROJECT
CALICO
docker
elastic
sysdig


Agent Machine
Disk 1
Disk 2
Disk n
persist
persist
persist
Workload
(statefull)
Web service
(other workload, stateless)
deploy/restart
deploy/restart
Marathon


Agent Machine
Disk 1
Disk 2
Disk n
persist
persist
persist
Workload
(statefull)
Web service
(other workload, stateless)
deploy/restart
deploy/restart
Marathon


CI/CD
Docker build
Image verification
diskimage-builder
simian cluster tests
publish image
trigger
Source control
Dockerfile files
Pull request
change
Artifactory

```
mesos-agent
├── Dockerfile
└── tree
    ├── data
    ├── datarw
    ├── etc
    |   ├── calico
    |   |   └── calico.env
    |   ├── mesos-slave
    |   |   ├── container_logger
    |   |   ├── containerizers
    |   |   └── modules
    |   ├── mesos-slave-modules.json
    |   ├── modules-load.d
    |   |   └── overlay.conf
    |   ├── systemd
    |   |   └── system
    |   |       ├── calico-libnetwork.service
    |   |       ├── calico-node.service
    |   |       ├── data.mount
    |   |       ├── datarw.mount
    |   |       ├── docker.service.d
    |   |       |   └── override.conf
    |   |       ├── etcd.service
    |   |       ├── mesos-slave.service.d
    |   |       |   └── override.conf
...
```

```
FROM zbi/host-os/base:latest

LABEL DOCKER_IMAGE="host-os/mesos-agent"
ARG REPO_SETTINGS=repo-settings

COPY $REPO_SETTINGS/apt/ubuntu/xenial/etc/apt/sources.list.d/docker.list /etc/apt/sources.list.d/docker.list
COPY $REPO_SETTINGS/apt/ubuntu/xenial/etc/apt/sources.list.d/mesosphere.list /etc/apt/sources.list.d/mesosphere.list

# Do this first so we can cache the yum installs.
COPY $REPO_SETTINGS/apt/public-keys/apt-dockerproject.asc /tmp/
COPY $REPO_SETTINGS/apt/public-keys/apt-mesosphere.asc /tmp/

RUN apt-key add /tmp/apt-dockerproject.asc && rm /tmp/apt-dockerproject.asc && \
    apt-key add /tmp/apt-mesosphere.asc && rm /tmp/apt-mesosphere.asc

ARG EXTRA_PKGS
```

```
# Install Packages - version fixed
RUN apt-get update && apt-get install --no-install-recommends -y $EXTRA_PKGS \
    mesos=1.1.0-0.0.268.pre.20160821gitbb047cd.ubuntu1604 \
    docker-engine=1.11.2-0~xenial \
    logrotate=3.8.7-2ubuntu2 \
    # Filesystem utilities for mesos disks
    e2fsprogs xfsprogs util-linux parted gdisk grub-common

ADD https://artifactory.br.zbi.cba/artifactory/static/calico/v0.21.0/calicoctl /usr/bin/
ADD https://artifactory.br.zbi.cba/artifactory/static/etcd/v3.0.6/etcdctl /usr/bin/
RUN chmod 755 /usr/bin/calicoctl /usr/bin/etcdctl

ADD https://artifactory.br.zbi.cba/artifactory/hashicorp-vault/0.6.2/vault_0.6.2_linux_amd64.zip /tmp
RUN cd /tmp; /usr/bin/unzip /tmp/vault_0.6.2_linux_amd64.zip; rm vault_0.6.2_linux_amd64.zip
RUN mv /tmp/vault /usr/bin/vault
RUN chmod 755 /usr/bin/vault
```

```
# Clear out the side-effectful default scripts
RUN rm -f /etc/default/mesos-slave && \
    rm -f /etc/default/mesos-master && \
    rm -f /etc/default/mesos

## Import our files
COPY tree/ /

## Symlink /etc/docker to /var/etc/docker
RUN ln -sf /var/etc/docker /etc

RUN systemctl enable data.mount

# explicitly disable master
RUN systemctl mask mesos-master.service && \
    systemctl enable mesos-slave.service && \
    systemctl enable calico-node.service && \
    systemctl enable calico-libnetwork.service && \
    systemctl enable docker.service && \
    systemctl enable etcd.service
```

```
# -------- DIB Required Environment Variables
# this allows us to add elements which will run ZBI tooling to add extra partitions
export ELEMENTS_PATH=${RELATIVE_PATH}/build/dib-elements
export DIB_IMAGE_SIZE=${diskSizeGB}
export DISTRO_NAME=ubuntu
export DIB_RELEASE=${DIB_RELEASE:-xenial}
export DIB_DOCKER_IMAGE=${DIB_DOCKER_IMAGE}
export DIB_BUILD_ELEMENTS="docker dpkg zbi-partition zbi-readonlyfs-prep bootloader"
export DIB_PARTED_DISK_ENTRIES=${zbiPartedDiskEntries}
export DIB_MOUNT_EXTRA=${zbiExtraMounts}
# ---------

echo ":: $0 Building disk image..."

executeCmd="disk-image-create -x -n -a amd64 --root-label ROOT -o ${OS_BASE_FNAME_QCOW} -t qcow2 ${DIB_BUILD_ELEMENTS}"
```

Search...

ZBI / host-os.mesos › 170

Merge pull request #75 from ZBI/merge-base-ubuntu-repo Merge the base ubuntu repository and update to the new JSON based provisioning format.

stephanh authored a day ago to mesoscon

RUNNING

started a few seconds ago

CANCEL

```
[info] Pulling image gliderlabs/alpine:3.1
```

drone-runv.zbi.dev.marathon.dyn-green.zbi.cba/ZBI/host-os.mesos/170/1

```ruby
require 'spec_helper'

packages = [
  { name: 'ca-certificates-java' },
  { name: 'docker-engine', version: '1.11.2-0~xenial' },
  { name: 'logrotate', version: '3.8.7-2ubuntu2' },
  { name: 'mesos', version: '1.1.0-0.0.268.pre.20160821gitbb047cd.ubuntu1604' },
  { name: 'net-tools' }, # contains netstat
  { name: 'openjdk-9-jre-headless' }
]

describe 'Packages' do

  packages.each do |p|
    describe package(p[:name]) do
      if p[:version].nil?
        it { should be_installed }
      else
        it { should be_installed.with_version(p[:version]) }
      end
    end
  end
end
```

```
zcmd = "sudo docker run --rm -it --entrypoint /opt/zookeeper/bin/zkCli.sh #{ZBI::PARAMS.zookeeper.image} -server #{masters.join(',')}"

# generate a random value so we can rerun tests in dev
random = rand(1000000000000000000)
# check that we can create a new k/v pair
describe command("#{zcmd} create /bedrock-#{random} \"rocks\"") do
  its(:stdout) { should match /^Created \/bedrock-#{random}$/ }
end
# check that we can read the new k/v pair
describe command("#{zcmd} get /bedrock-#{random}") do
  its(:stdout) { should match /^rocks$/ }
end

# ----------------------------------------
# CHECK THAT WE HAVE A LEADER
# ----------------------------------------
cmd = ['rm -rf /tmp/zk-membership']

ZBI::PARAMS.zookeeper.masters.each do |server|
  cmd << "echo mntr | nc #{server} #{ZBI::PARAMS.zookeeper.ports.client} >> /tmp/zk-membership"
end

cmd << 'cat /tmp/zk-membership'

# dump the stats from all the nodes to a file
describe command(cmd.join('; ')) do
  its(:exit_status) { should eq 0 }

  if masters.size == 1
    its(:stdout) { should match /^zk_server_state\sstandalone$/ }
  else
    # we expect to have exactly one leader
    its(:stdout) { expect(subject.stdout.scan(/^zk_server_state\sleader$/m).size).to eq 1 }
  end
end
```

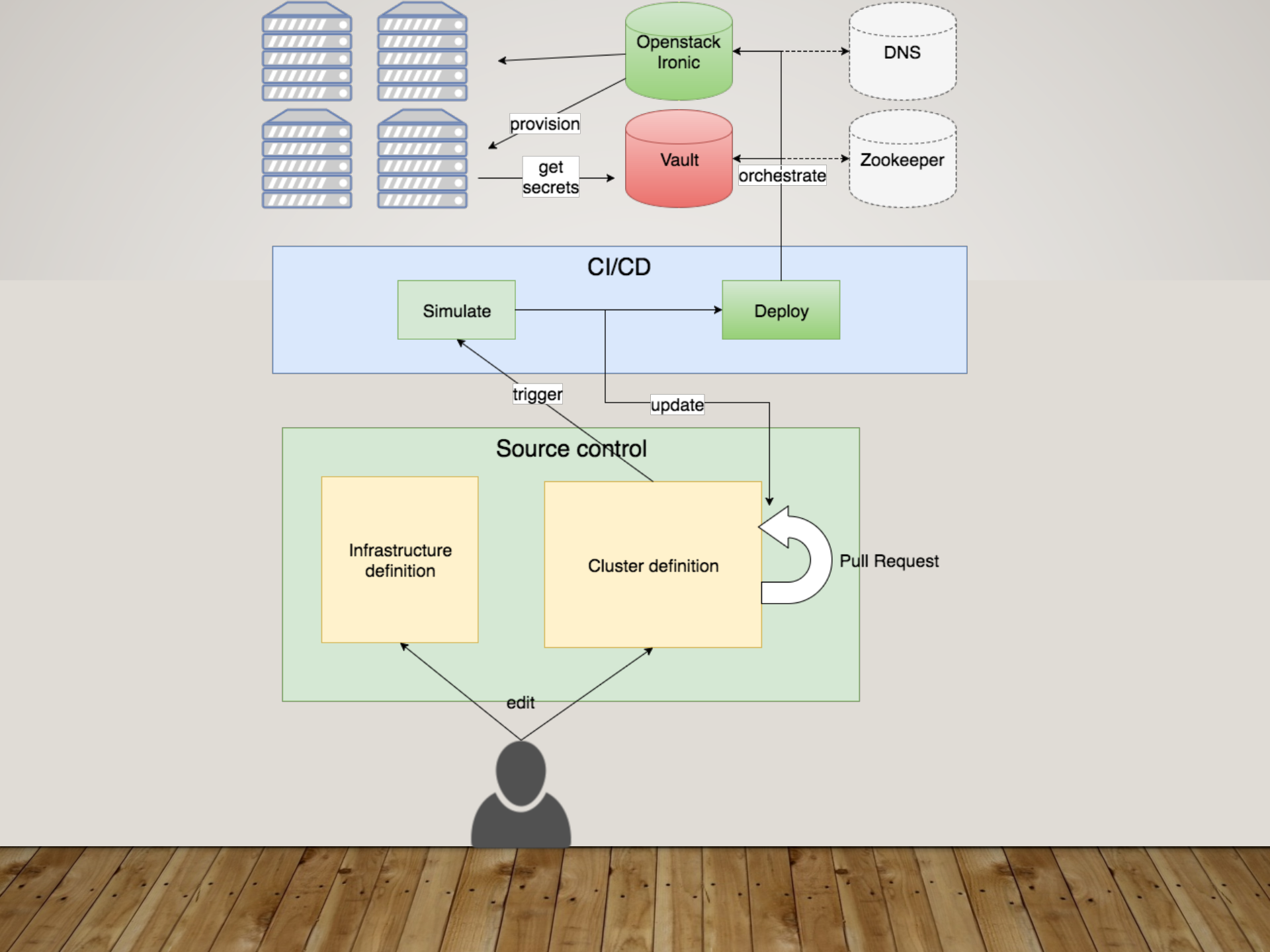
WHAT WORKED WELL?

- Tests
- Dockerfile
- No mutation
- PR for changes

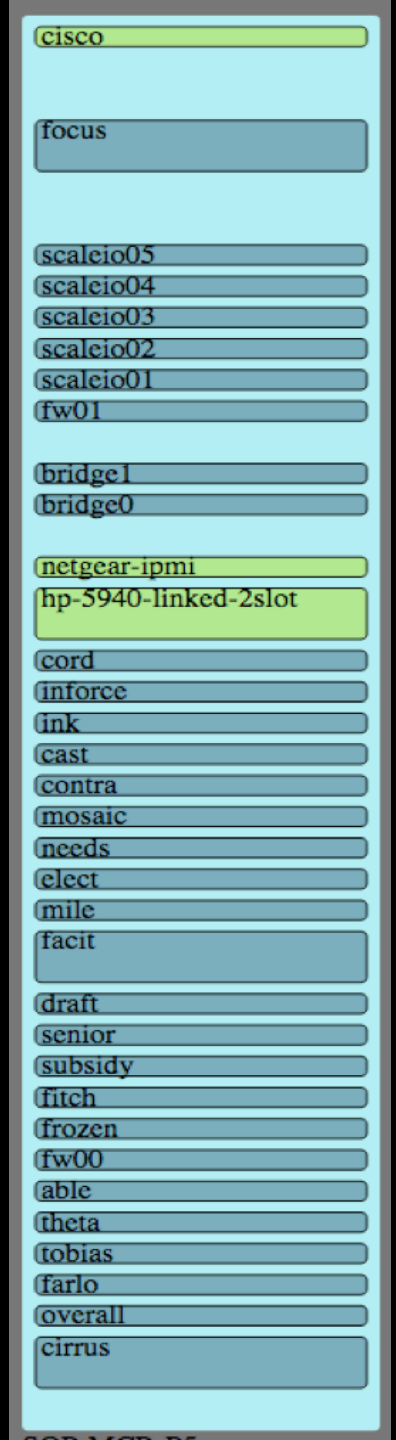
PAIN POINTS

- Converting docker images to qcow in docker
- Build cycle

DEPLOY/ORCHESTRATE CLUSTERS

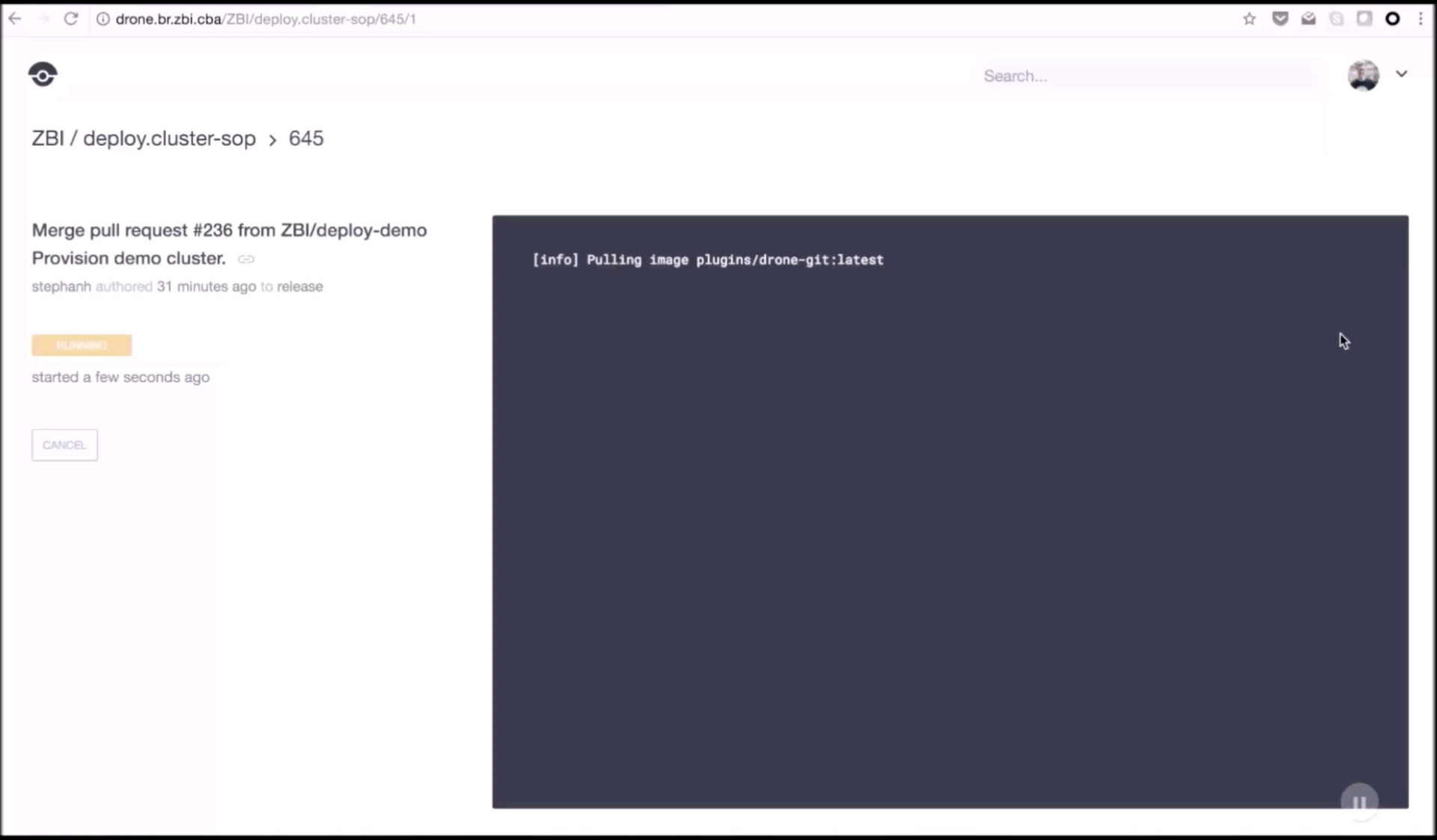

Openstack
Ironic
DNS
provision
Vault
Zookeeper
get
secrets
orchestrate
CI/CD
Simulate
Deploy
trigger
update
Source control
Infrastructure
definition
Cluster definition
Pull Request
edit

```yaml
rack:
  name: SOP-MCR-N5
  location: SOP-MCR
  size: 45
  pdus:
    - SOP-MCR-N5/PDU-2A-CB-43
    - SOP-MCR-N5/PDU-2B-CB-43
  devices:
    - server:
        name: murex
        usage: Rackspace
        additional_names: []
        missionplanner_managed: false
        ironic_uuid:
        serial: S174874264114 34
        hardware: *sm-g1
        date_added: "2016-09-09"
        rack_position:
          - 3
          - 4
        pdus:
          - SOP-MCR-N5/PDU-2B-CB-43
          - SOP-MCR-N5/PDU-2A-CB-43
        ipmi:
          mac:  0c:c4:7a:ae:d7:84
          switch: sop-mcr-n5-ipmi
          switch_port: 1
        network:
          - physical_slot: sfp-card/1
            mac_address: 0c:c4:7a:b6:f8:44
            switch: sop-mcr-n5-tor1
            switch_port: 1
          - physical_slot: sfp-card/2
            mac_address: 0c:c4:7a:b6:f8:45
            switch: sop-mcr-n5-tor2
            switch_port: 1
```



```yaml
clusters:
  green-cluster:
    dns:
      nameservers:
        - 10.11.11.1
      data_domain: nodes.zbi.cba
    masters:
      able:
        provision_id: 1
        lan:
          -
            mac: 00:00:b9:ab:19:43
            ip:  10.11.11.151/24
            vlan: 11
            gateway: 10.11.11.1
        ironic_id: a7af76ad-6583-4209-ba5f-cf1477b6405e
        flavor: ramish-baremetal-flavor2
        image: *mesos-master-green
      ...
    agents:
      earner:
        provision_id: 4
        lan:
          -
            mac: 00:00:b9:ab:19:44
            ip:  10.11.11.203/24
            vlan: 11
            gateway: 10.11.11.1
        ironic_id: 8065aa70-b658-4101-a176-fd4da69a3d39
        flavor: ramish-baremetal-flavor2
        image: *mesos-agent-persistent-storage
      ...
```

```yaml
clusters:
  green-cluster:
    dns:
      nameservers:
        - 10.11.11.1
      data_domain: nodes.zbi.cba
    etcd:
      token: green-cluster
    marathon:
      username: marathon
      password: ZjX4F9nslBeoo7pbDkto
    masters:
      able:
        provision_id: 1
        lan:
          -
            mac: 0c:c4:7a:c1:2e:92
            ip:  10.11.11.151/24
            vlan: 11
            gateway: 10.11.11.1
        ironic_id: a7af76ad-6583-4209-ba5f-cf1477b6405e
        flavor: ramish-baremetal-flavor2
        image: *mesos-master-green
      theta:
        provision_id: 2
        lan:
          -
            mac: 0c:c4:7a:a9:04:0c
            ip:  10.11.11.53/24
            vlan: 11
            gateway: 10.11.11.1
        ironic_id: 8ff1fd1c-4893-11e6-a447-2f366077ca0e
        flavor: ramish-baremetal-flavor2
```


drone.br.zbi.cba/ZBI/deploy.cluster-sop/645/1
Search...
ZBI / deploy.cluster-sop › 645
Merge pull request #236 from ZBI/deploy-demo
Provision demo cluster.
stephanh authored 31 minutes ago to release
RUNNING
started a few seconds ago
CANCEL
[info] Pulling image plugins/drone-git:latest

WHAT WORKED WELL

- PR update with deployment plan
- Cluster level abstraction
- Using one tool to manage all our changes
- Functional programming – Interpreter pattern

PAIN POINTS

- Lots of tooling to build
- Deployment cycles

FUTURE WORK

- Mesos/Marathon integration
- Zookeeper/Etcd orchestration
- Workload integration
- Read only OS filesystem

QUESTIONS?