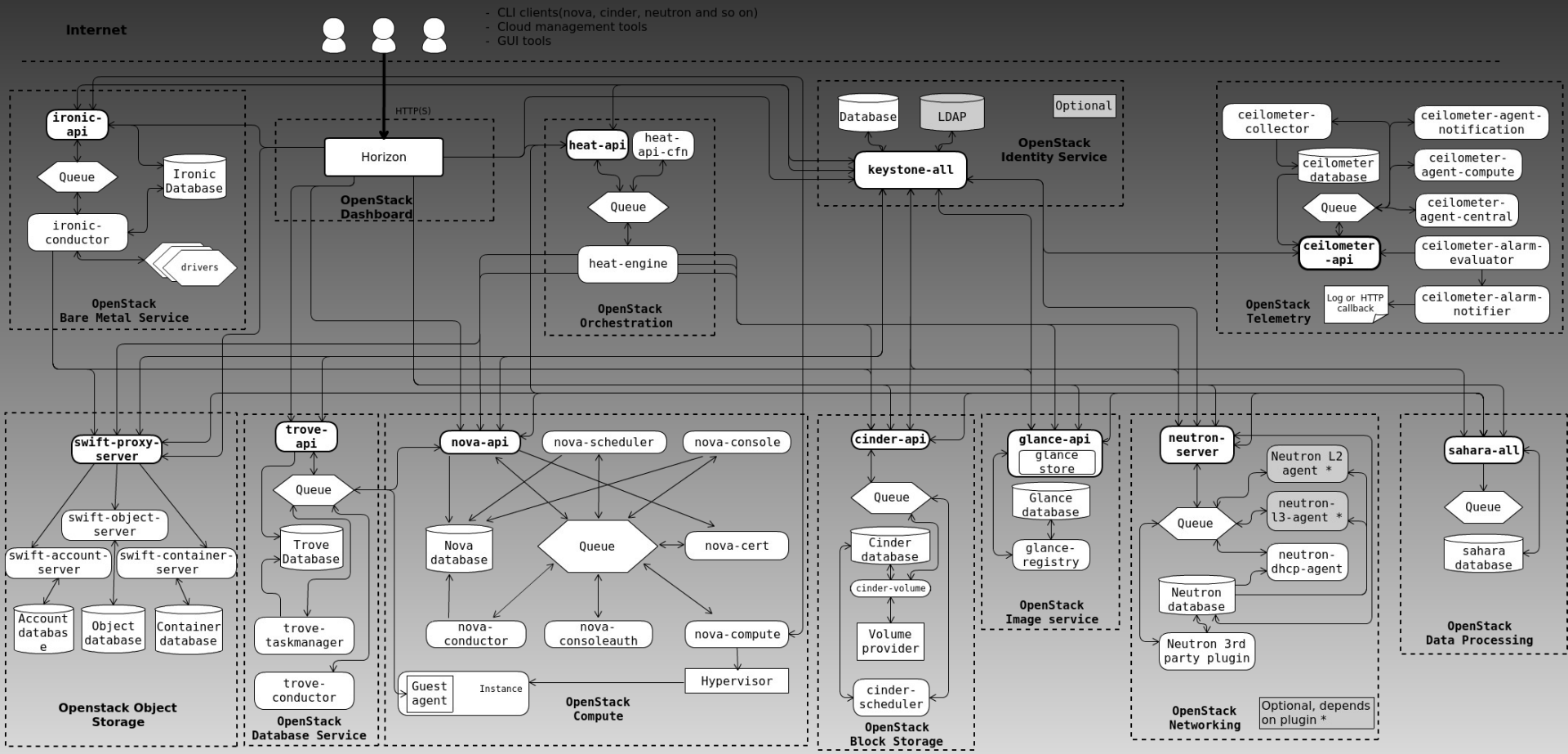
RHEL OSP Director

Jon Jozwiak
Sr. Cloud Solutions Architect
Red Hat, Inc.
November 2015

WHY ARE WE HERE?

- Learn about RHEL OSP Director
  - What is it? How it works?
  - Walk through the typical workflow of a deployment

OPENSTACK INSTALLERS – SIMPLIFY DEPLOYING THIS



redhat.

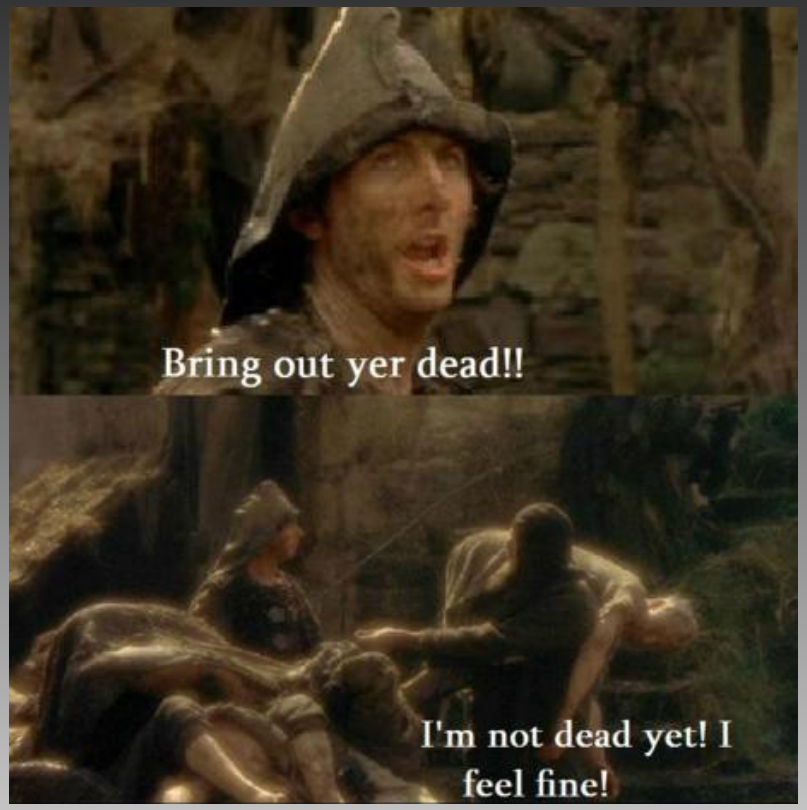
~~NECRONOMICON~~ OPENSTACK INSTALLATION MANUAL

PACKSTACK

- Answer-file based solution for deploying test/dev environments
  - Relies on puppet
- Does not handle bare metal provisioning. Bring your own OS(s)
- Does not provision HA

RHEL OSP INSTALLER

- Wizard based tool built on Foreman
  - Relies on Puppet for configuration
- Difficult to customize to meet needs
- Doesn't handle ongoing lifecycle management (fire and forget)



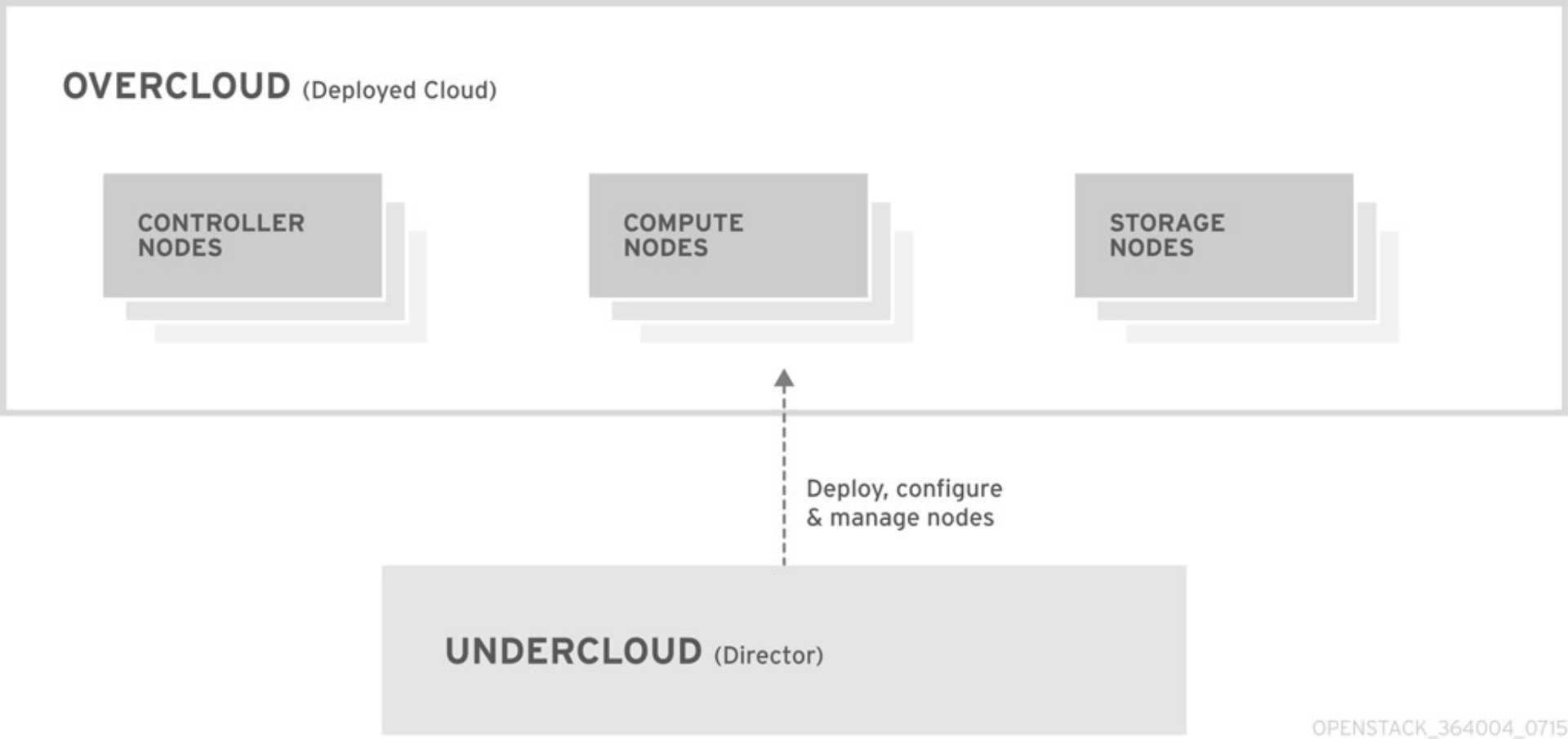
SPINALSTACK

- Jenkins based solution for installing OpenStack
  - Came from eNovance acquisition
  - Relies on Puppet for configuration
- Image based deployment
- Hardware Bench-marking
- Doesn't deploy Red Hat's HA architecture

RHEL OSP DIRECTOR

- Use OpenStack to Deploy OpenStack
  - Heat for Orchestration
  - Puppet for configuration
  - Highly Customizable
- Image-Based Deployment
- Hardware Benchmarking
- Optionally deploys Ceph
- Facilitates future upgrades
- Tempest for validation

DIRECTOR ARCHITECTURE



DIRECTOR WORKFLOW

- Design Your Cloud
- Deploy Your Undercloud
- Prepare for Overcloud
- Discover Nodes (Introspection) and Match to Roles
- Make Necessary Customizations (Network, Storage)
- Deploy your Overcloud
- Validate your Overcloud

DESIGN YOUR CLOUD

- Hardware must have IPMI and minimum 2 NICs
  - Recommended minimum 10 nodes (1 Director, 3 Controller, 3 Compute, and 3 Ceph)
  - Absolute minimum of 3 nodes (1 Director, 1 Controller, 1 Compute)
- Block Storage
  - Decide which back-ends are required
  - Where is ephemeral storage hosted? Live Migration?
  - Ideally primary back-end aligns with Glance storage to enable copy-on-write cloning
- Is Object Storage Required?

ADVANCED NETWORKING

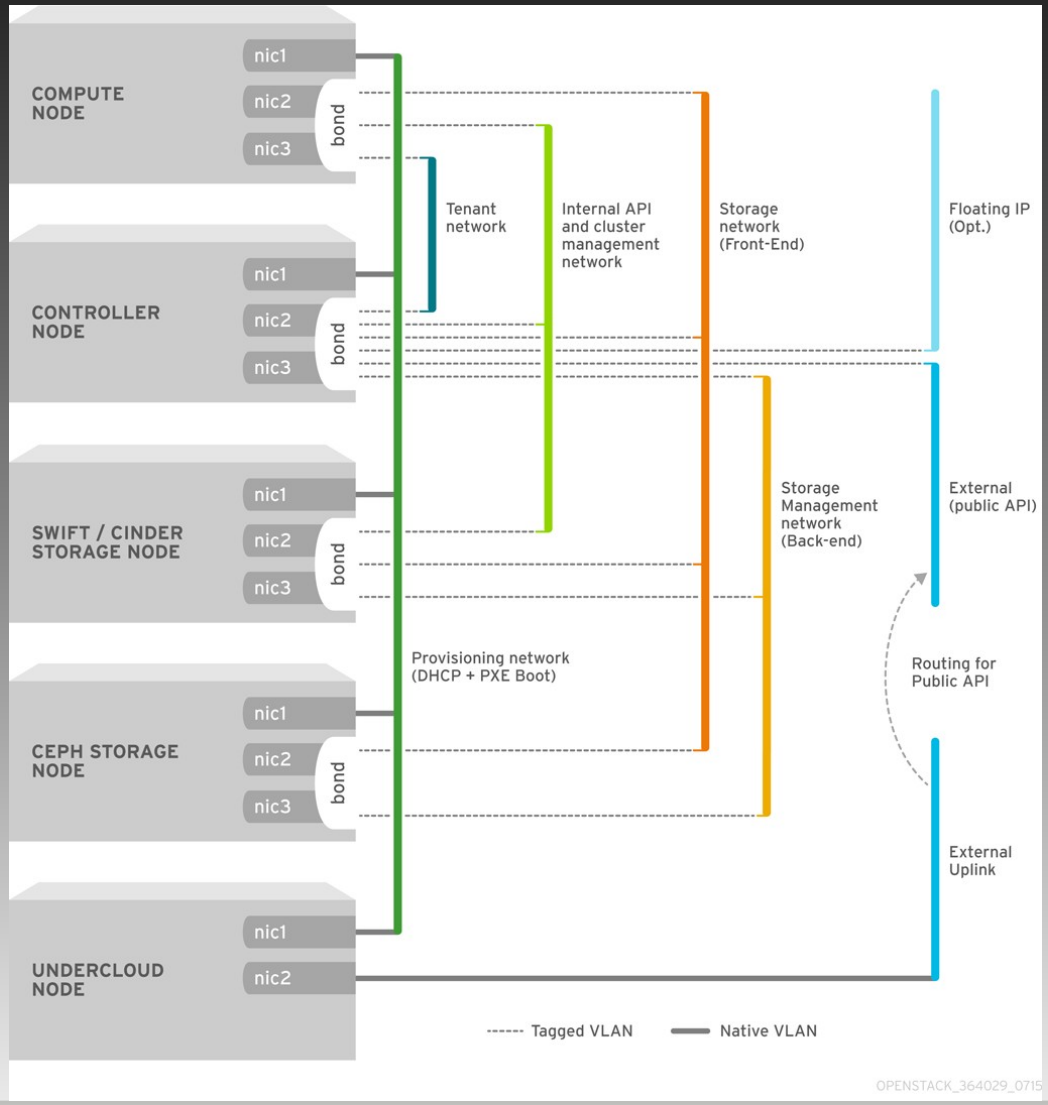
- Networks
  - Provisioning – Undercloud control plane for deployment and management – usually native VLAN on port
  - Internal API – OpenStack internal API, RPC, and DB
  - Tenant – Tenant private networks via VLAN trunk or VxLAN/GRE tunneling
  - Storage – Storage data path to nodes
  - Storage Management – Storage Replication, Ceph back-end services
  - External – Public OpenStack APIs, Horizon dashboard
  - Floating IP (Optional, can be combined with External)

ADVANCED NETWORKING

- Tenant Networking
  - VLAN-based – Pass a trunk of VLANs without assigning IP addresses. Tenant networks limited by number of VLANs available
  - VxLAN-based – Each host is a VxLAN endpoint and has an IP. Offers greater scalability

- Typical NIC Configurations
  - Rack-mount or blades
    - 1x1GB provisioning
    - 2x10GB bonded pair with VLANs for each network type
  - Cisco UCS
    - Bonded in server profile. Individual NICs for each network type

ADVANCED NETWORKING



DOCUMENT YOUR CONFIGURATION!

- Much easier to troubleshoot problems later!
  - Network Port to Server Port
  - MAC Addresses
  - VLAN(s)
  - Network Type
  - IP Range per network
  - Gateways
  - IPMI Cards, Credentials
  - Storage Connectivity

DEPLOY YOUR UNDERCLOUD

- Undercloud Minimum Requirements
  - Virtual or Physical RHEL 7.1
  - 6 GB RAM
  - 40 GB Disk Space
  - 2 x 1GB Network Interfaces
  - Access to IPMI Interfaces

DEPLOY YOUR UNDERCLOUD

- Subscribe host, setup NTP, yum update
- Create stack user & install Director

```
useradd stack

echo "NotMyPassword" | passwd stack --stdin

echo "stack ALL=(root) NOPASSWD:ALL" | tee -a /etc/sudoers.d/stack

chmod 0440 /etc/sudoers.d/stack

su - stack

sudo yum -y install python-rdomanager-oscplugin openstack-utils
```

DEPLOY YOUR UNDERCLOUD

```
cp /usr/share/instack-undercloud/undercloud.conf.sample ~/undercloud.conf
#### Provisioning Interface Configuration
openstack-config --set undercloud.conf DEFAULT local_ip 192.0.55.1/24
openstack-config --set undercloud.conf DEFAULT undercloud_public_vip  192.0.55.10
openstack-config --set undercloud.conf DEFAULT undercloud_admin_vip 192.0.55.11
openstack-config --set undercloud.conf DEFAULT local_interface eth0
openstack-config --set undercloud.conf DEFAULT masquerade_network 192.0.55.0/24
openstack-config --set undercloud.conf DEFAULT dhcp_start 192.0.55.20
openstack-config --set undercloud.conf DEFAULT dhcp_end 192.0.55.120
openstack-config --set undercloud.conf DEFAULT network_cidr 192.0.55.0/24
openstack-config --set undercloud.conf DEFAULT network_gateway 192.0.55.1
openstack-config --set undercloud.conf DEFAULT discovery_iprange
192.0.55.150,192.0.55.180
openstack-config --set ~/undercloud.conf DEFAULT discovery_runbench true

openstack undercloud install
```

PREPARE FOR OVERCLOUD

- Download RHEL OSP 7 Images
  - https://access.redhat.com/downloads/content/191/ver=7.0/rhel---7/7.0/x86_64/product-downloads

```
cd /home/stack/images

for tarfile in *.tar; do tar -xf $tarfile; done

source ~/stackrc

openstack overcloud image upload --image-path /home/stack/images

openstack image list
```

PREPARE FOR OVERCLOUD

- Point to a DNS server for your subnet

```
PROVSUBNET=$(neutron subnet-list | grep 192.0.55 | awk '{print $2}')

neutron subnet-update $PROVSUBNET --dns-nameserver 10.19.143.247
```

DISCOVER NODES / MATCH ROLES

- Create instackenv.json

```
{
 "nodes": [
  {
   "pm_addr": "10.19.143.61",
   "mac": [
    “c8:1f:66:65:33:44"
   ],
   "cpu": "4",
   "memory": "8192",
   "disk": "40",
   "arch": "x86_64",
   "pm_type": "pxe_ipmitool",
   "pm_user": "root",
   "pm_password": "NotMyPassword"
  }
 ]
}
```

DISCOVER NODES / MATCH ROLES

- Easier instackenv.json

```
cd /home/stack

git clone https://github.com/jtaleric/csv-to-instack.git

cat << EOF >> labservers.csv

c8:1f:66:65:33:44,ra-m1000e-02-drac1.cloud.lab.eng.bos.redhat.com,root,NotMyPassword,pxe_ipmitool

EOF

cd csv-to-instack

python csv-to-instack.py --csv=/home/stack/labservers.csv > /home/stack/instackenv.json
```

DISCOVER NODES / MATCH ROLES

- Validate instackenv.json

```
git clone https://github.com/rthallisey/clapper.git

clapper/instackenv-validator.py -f
/home/stack/instackenv.json
```

- Import / Register Nodes

```
openstack baremetal import --json
~/instackenv.json
```

DISCOVER NODES / MATCH ROLES

- Assign Kernel / Ramdisk to nodes

```
openstack baremetal configure boot
```

- Import the hardware

```
openstack baremetal introspection bulk start
openstack baremetal introspection bulk status
```

AUTOMATED HEALTH CHECK (AHC)

- Benchmarking run during introspection if 'discovery_runbench = true' in undercloud.conf

```
ahc-report --full
```

- Show ways to match

```
ahc-report --categories

ahc-report --categories | grep -A3 "3 identical systems"
```

- Report on outliers

```
ahc-report --outliers | grep -i underperformance | head -n5

ahc-report --outliers | grep -i inconsistent | head -n5
```

DISCOVER NODES / MATCH ROLES

- Define matching rules for all node types
  - CPU >= 32, < 40, Memory <= 110 GB, root disk > 25GB

```
cat << EOF > /etc/ahc-tools/edeploy/compute.specs
[
 ('cpu', 'logical', 'number', 'and(ge(32), lt(40))'),
 ('memory', 'total', 'size', 'le(110000000000)'),
 ('disk', 'sda', 'size', 'gt(25)'),
]
EOF
```

DISCOVER NODES / MATCH ROLES

- Set number of nodes for each role

```
cat << EOF > /etc/ahc-tools/edeploy/state
[('control', '3'), ('ceph', '3'), ('compute', '*')]
EOF
```

- Assign Ironic nodes to profiles

```
ahc-match
```

- List Matches

```
for i in $(ironic node-list | awk ' /available/ { print $2 } ');
do ironic node-show $i | grep capabilities; done
```

DISCOVER NODES / MATCH ROLES

- Create Nova flavors for each node type – specs must be lower than actuals

```
openstack flavor create --id auto --ram 32768 --disk 25 --vcpus 32 control
openstack flavor create --id auto --ram 61440 --disk 25 --vcpus 32 compute
openstack flavor create --id auto --ram 98304 --disk 25 --vcpus 40 ceph
```

- baremetal is not used, but a default is needed

```
openstack flavor create --id auto --ram 8192 --disk 40 --vcpus 4 baremetal
```

DISCOVER NODES / MATCH ROLES

- Map Ironic profiles to flavors

```
openstack flavor set --property "cpu_arch"="x86_64"
--property "capabilities:boot_option"="local" --property
"capabilities:profile"="control" control

openstack flavor set --property "cpu_arch"="x86_64"
--property "capabilities:boot_option"="local" --property
"capabilities:profile"="compute" compute

openstack flavor set --property "cpu_arch"="x86_64"
--property "capabilities:boot_option"="local" --property
"capabilities:profile"="ceph" ceph

openstack flavor set --property "cpu_arch"="x86_64"
--property "capabilities:boot_option"="local" baremetal
```

CUSTOMIZE DEPLOYMENT

- Primary needs are to align the deployment with your specific network and storage needs
- Create your own copy of the templates

```
cp -rf /usr/share/openstack-tripleo-heat-templates/* ~/templates/
```

- Network Customization
  - OSP director defaults to using PXE/Management for ALL traffic
  - Copy the network isolation to customize it

```
cp ~/templates/environments/network-isolation.yaml ~/templates/environments/custom-network-isolation.yaml
```

NETWORK CUSTOMIZATION

- Network Customization
  - Modify to allow per-role NIC configurations

```
cat << EOF >> ~/templates/environments/custom-network-isolation.yaml
  # NIC Configs for our roles
  OS::TripleO::Compute::Net::SoftwareConfig:
/home/stack/templates/nic-configs/compute.yaml
  OS::TripleO::Controller::Net::SoftwareConfig:
/home/stack/templates/nic-configs/controller.yaml
  OS::TripleO::CephStorage::Net::SoftwareConfig:
/home/stack/templates/nic-configs/ceph-storage.yaml
EOF
```

NETWORK CUSTOMIZATION

- Create the NIC config files for each role.
  - Examples exist in templates/network/config or upstream at https://github.com/openstack/tripleo-heat-templates/tree/master/network/config
  - Examples cover typical NIC configurations (bond with VLANS, or multiple NICs with one per network type)

```
-
        type: interface
        name: nic2
        use_dhcp: false
        addresses:
          -
            ip_netmask: {get_param: StorageIpSubnet}
```

NETWORK CUSTOMIZATION

- Add network address specifics for each network

```
cat << EOF >> ~/templates/advanced-networking.yaml

parameter_defaults:

# Internal API used for private OpenStack Traffic
InternalApiNetCidr: 172.16.1.0/24

InternalApiAllocationPools: [{'start': '172.16.1.10', 'end': '172.16.1.200'}]

InternalApiNetworkVlanID: 1600

...
```

STORAGE CUSTOMIZATION

- Verify and Update Storage Back-end Configuration

vi ~/templates/environments/storage-environment.yaml

```
CinderEnableRbdBackend: true

GlanceBackend: rbd
```

STORAGE CUSTOMIZATION

- If Ceph, update your disk layout:

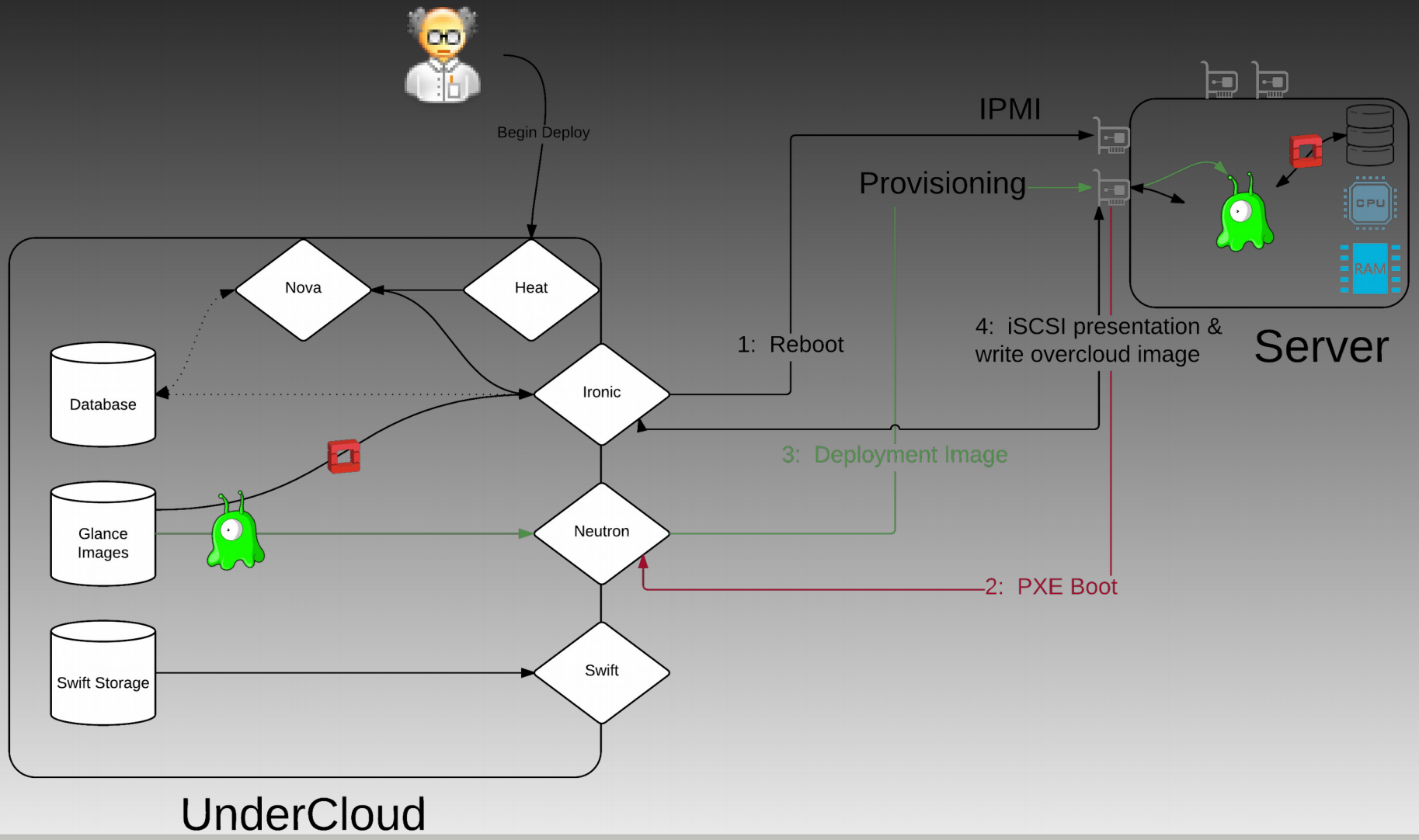
```
vi ~/templates/puppet/hieradata/ceph.yaml
---
ceph::profile::params::osds:
  '/dev/sdb':
    journal: '/dev/sdn'
  '/dev/sdc':
    journal: '/dev/sdn'
  '/dev/sdd':
    journal: '/dev/sdn'
  '/dev/sde':
    journal: '/dev/sdn'
...
```

Intermission

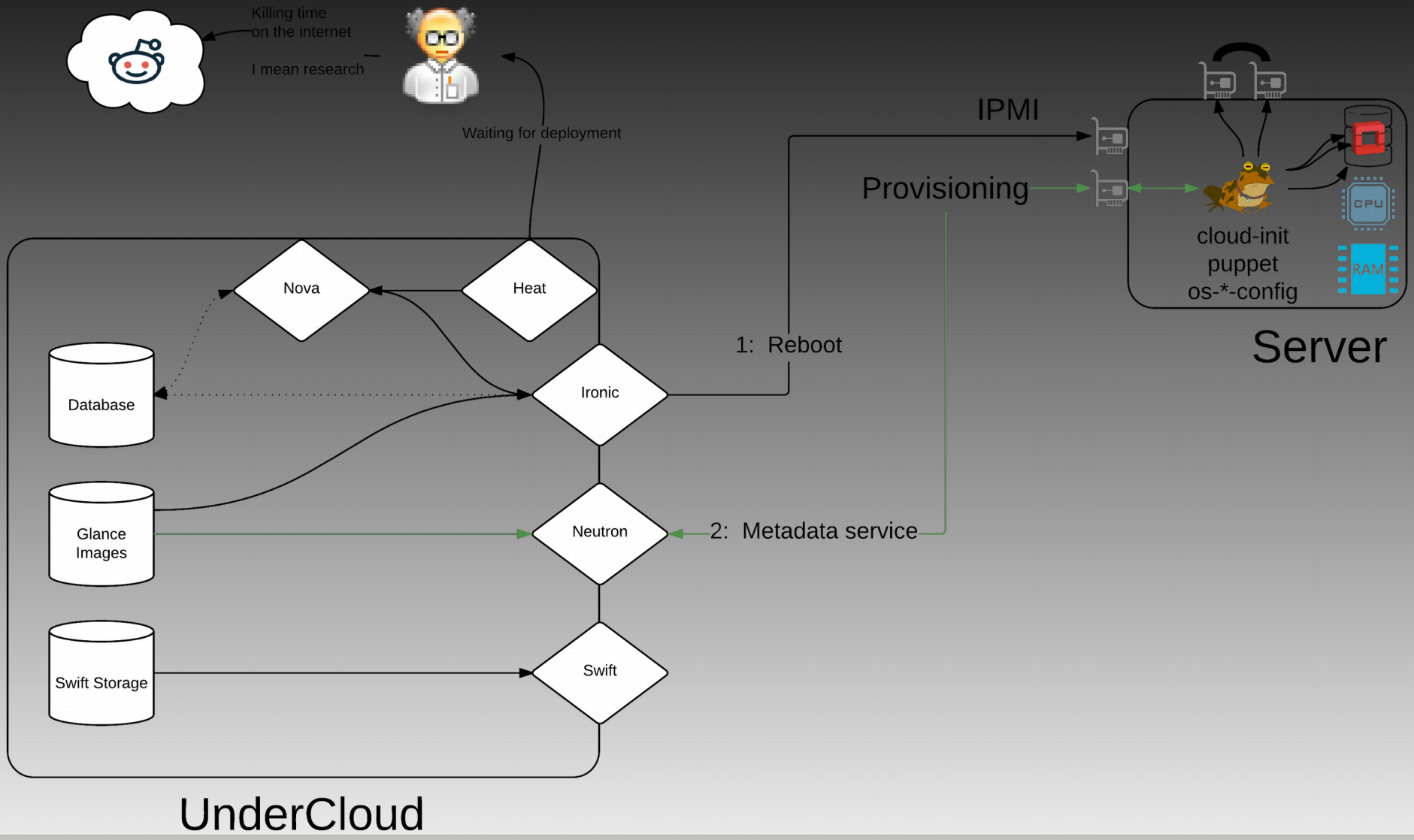
DEPLOY YOUR OVERCLOUD

```
openstack overcloud deploy \
--templates ~/templates/ --ntp-server 10.5.26.10 \
--control-flavor control --compute-flavor compute
--ceph-storage-flavor ceph \
--control-scale 3 --compute-scale 2 -ceph-storage-scale 3 \
--neutron-tunnel-types vxlan --neutron-network-type vxlan \
-e ~/templates/environments/storage-environment.yaml \
-e ~/templates/environments/custom-network-isolation.yaml
```

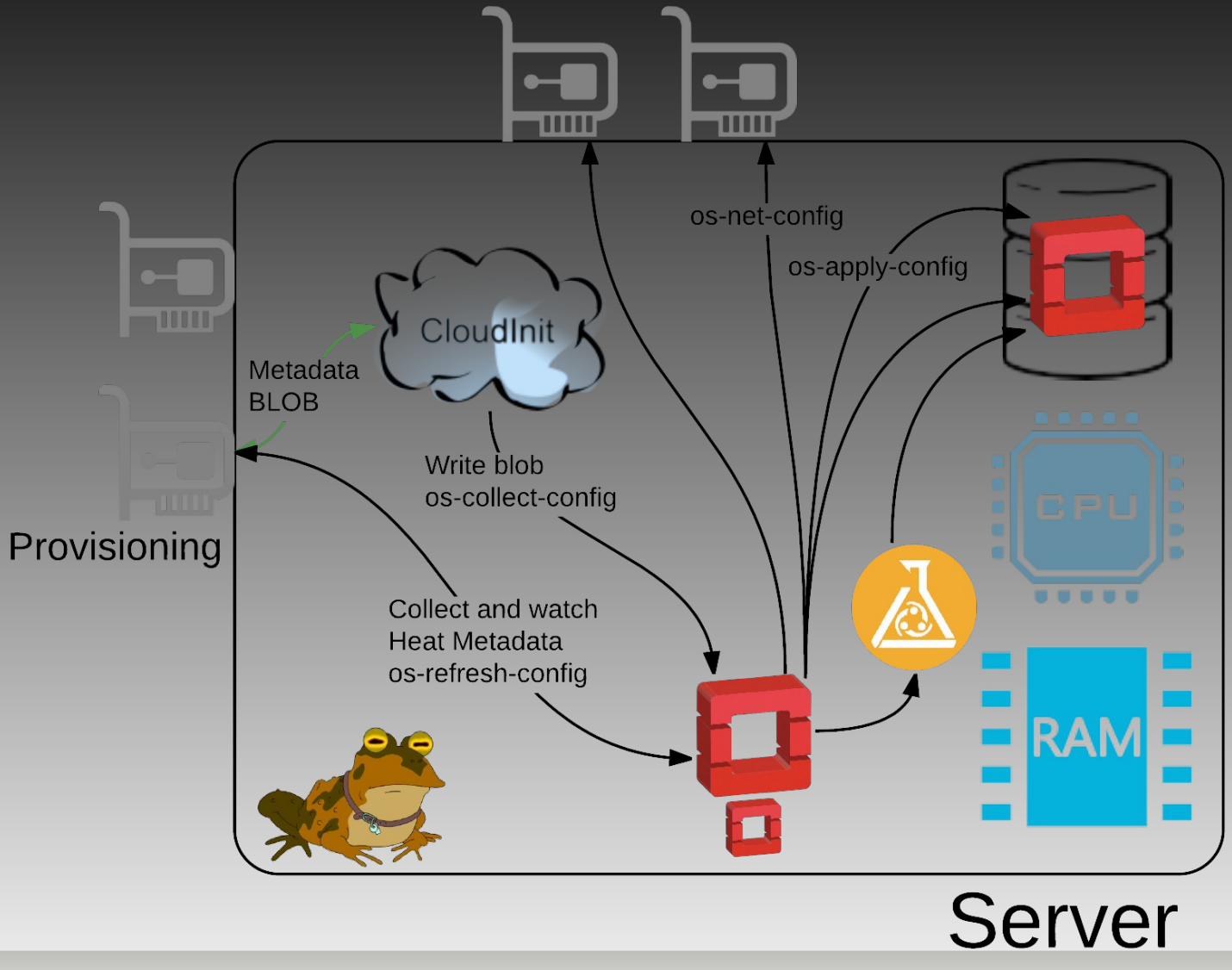
INSTALLING OVERCLOUD



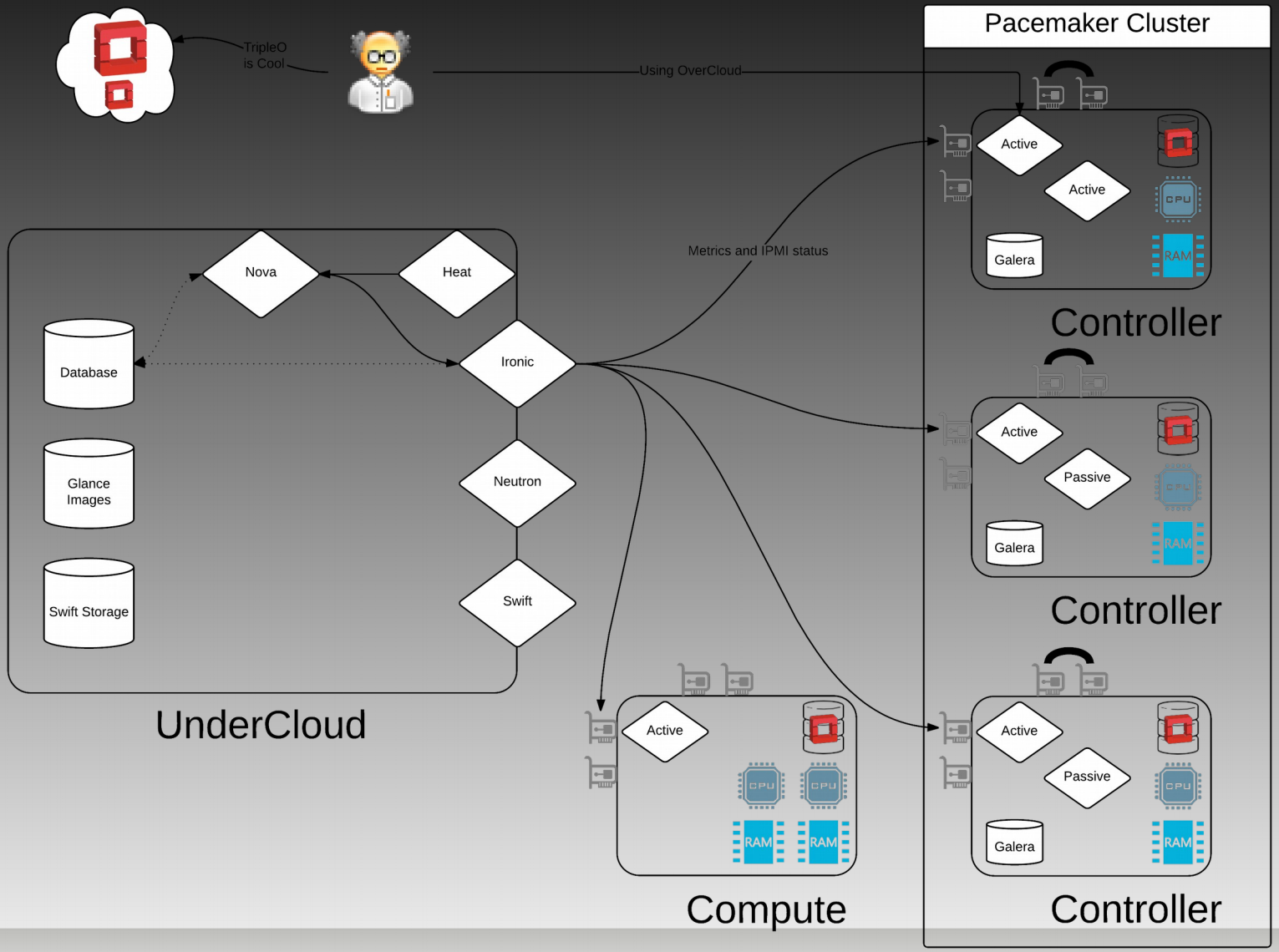
INSTALLING OVERCLOUD



INSTALLING OVERCLOUD



INSTALLING OVERCLOUD



redhat.

VALIDATE YOUR DEPLOYMENT

- Deployment will take about 45 minutes – depending on hardware
- HA fencing must be setup manually afterwards
- Tempest allows a full test or basic smoke test
  - Full test can take a number of hours
  - Smoke test gives general understanding on whether environment is running

```
source ~/overcloudrc
openstack overcloud validate \
    --overcloud-auth-url $OS_AUTH_URL \
    --overcloud-admin-password $OS_PASSWORD \
    --tempest-args '.*smoke'
```

INSTALL COMPLETE

Thank You

Any questions?