BEST PRACTICES FOR DEPLOYING RED HAT OPENSTACK PLATFORM with Open vSwitch and Red Hat Ceph Storage

Jacob Liberman
Field Product Manager -- OpenStack
6/30/2016

INTRODUCTION

Best practices for deploying and scaling Red Hat OpenStack Platform with Open vSwitch and Red Hat Ceph storage

Abstract:

In this deep dive implementation session, you'll learn how to successfully deploy and scale Red Hat OpenStack Platform with Red Hat's best practices for integration of Open vSwitch and Red Hat Ceph Storage, taking into consideration high availability, IPv6 networking, and the deployment and usage of Director for massive scalability. Learn the tips and tricks, while avoiding typical pitfalls to ensure you're successful.

Introduction

Current role:

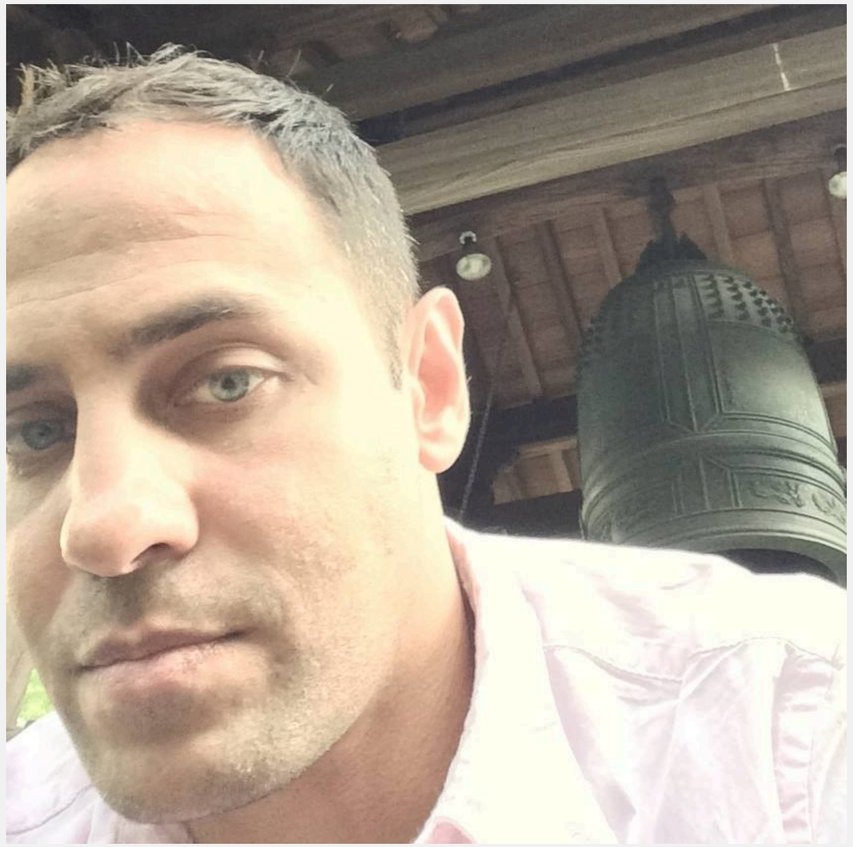
-- Field product manager for OpenStack

-- Drive customer feedback into product

Prior role:

-- Systems Engineer

-- Wrote OpenStack reference architectures



AGENDA

Best practices for deploying and scaling Red Hat OpenStack Platform with Open vSwitch and Red Hat Ceph storage

Setting Context
Introduction and agenda

OpenStack in a Minute or So...
OpenStack review

TripleO Concept
Understanding TripleO

How It Works
How it works, example architecture, roles

Developer mindset
Key director features, current and roadmap

Deployment best practices
Undercloud, Overcloud, Storage, networking

Scaling best practices
Compute, controller, Ceph storage

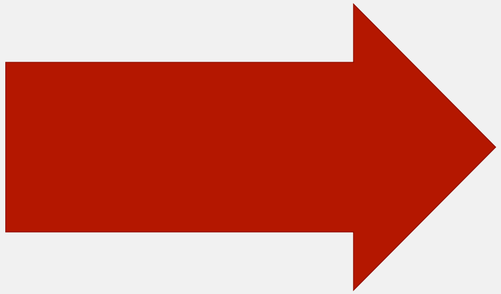
Future directions
Ansible, containers, etc

OPENSTACK…
…in a minute or so

OpenStack - a quick review...

OpenStack components* manage compute, network and storage resources

- Nova
- Neutron
- Glance
- Swift
- Heat



Virtual Machine Deployment

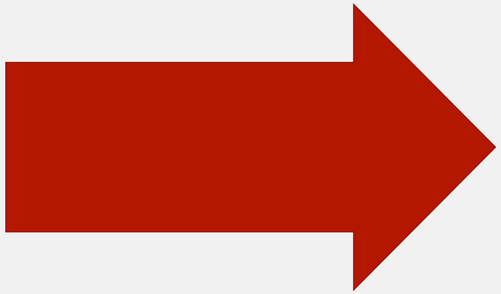
*Only a subset of OpenStack components are shown here.

OpenStack - Nova, Glance & Heat

OpenStack components* manage compute, network and storage resources

- Nova
- Neutron
- Glance
- Swift
- Heat

Nova provides command and control services for compute nodes.



It orchestrates rendering of virtual machines.

Virtual Machine Deployment

*Only a subset of OpenStack components are shown here.

OpenStack - Nova, Glance & Heat

OpenStack components* manage compute, network and storage resources

- Nova
- Neutron
- Glance
- Swift
- Heat

Glance is an image service, storing images and metadata mainly used for building virtual machines.

Users can upload and discover data assets that are meant to be used with other services.

VM

OS image, network, disk

Virtual Machine Deployment

*Only a subset of OpenStack components are shown here.

OpenStack - Nova, Glance & Heat

OpenStack components* manage compute, network and storage resources

- Nova
- Neutron
- Glance
- Swift
- Heat

Heat provides an orchestration engine to launch multiple cloud applications based on templates.

Heat manages the whole lifecycle of the application by managing the underlying virtual infrastructure.

VM

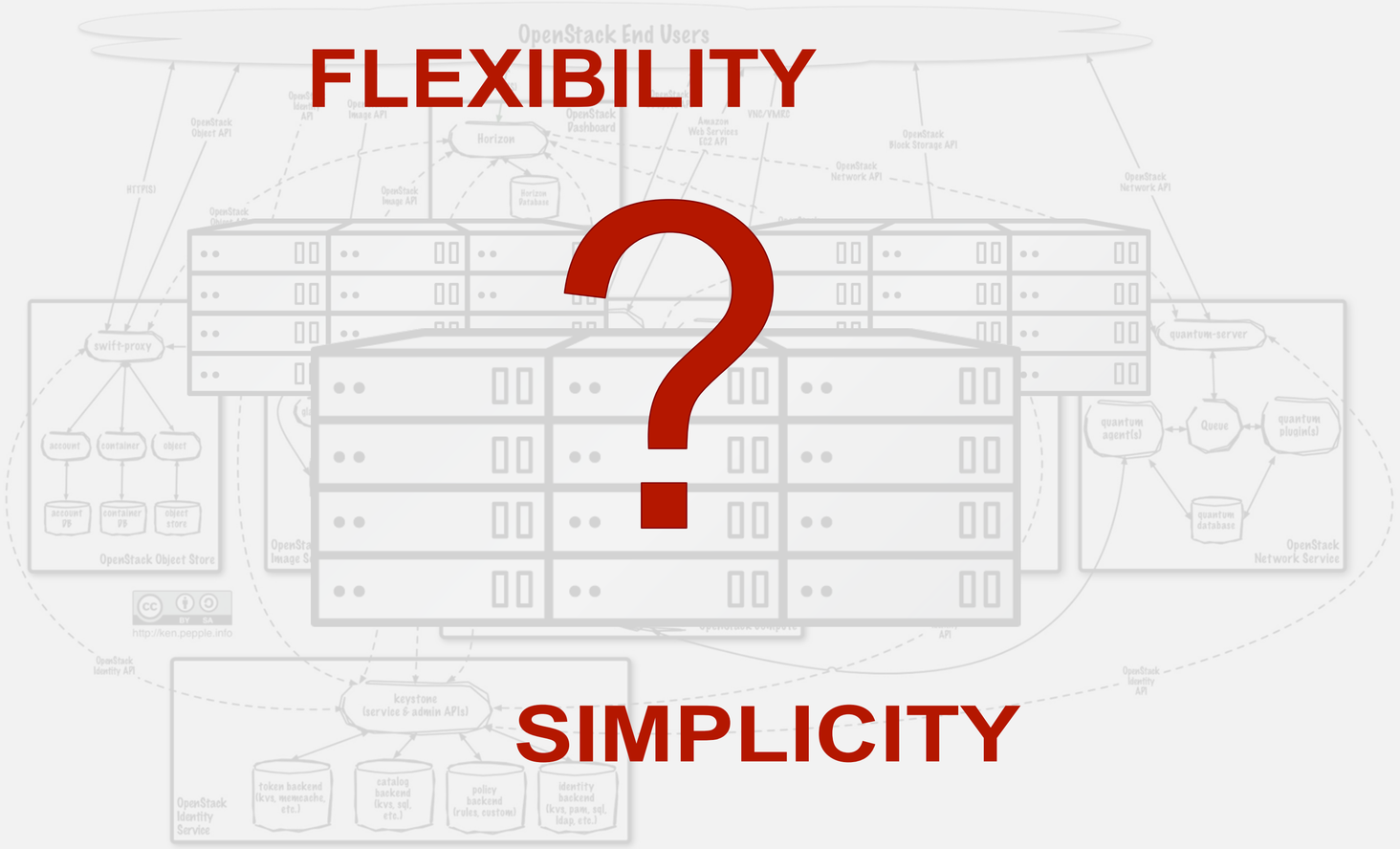
OS image, network, disk

Virtual Machine Deployment

*Only a subset of OpenStack components are shown here.

TRIPLEO CONCEPT

OpenStack World Challenge



Past Approaches

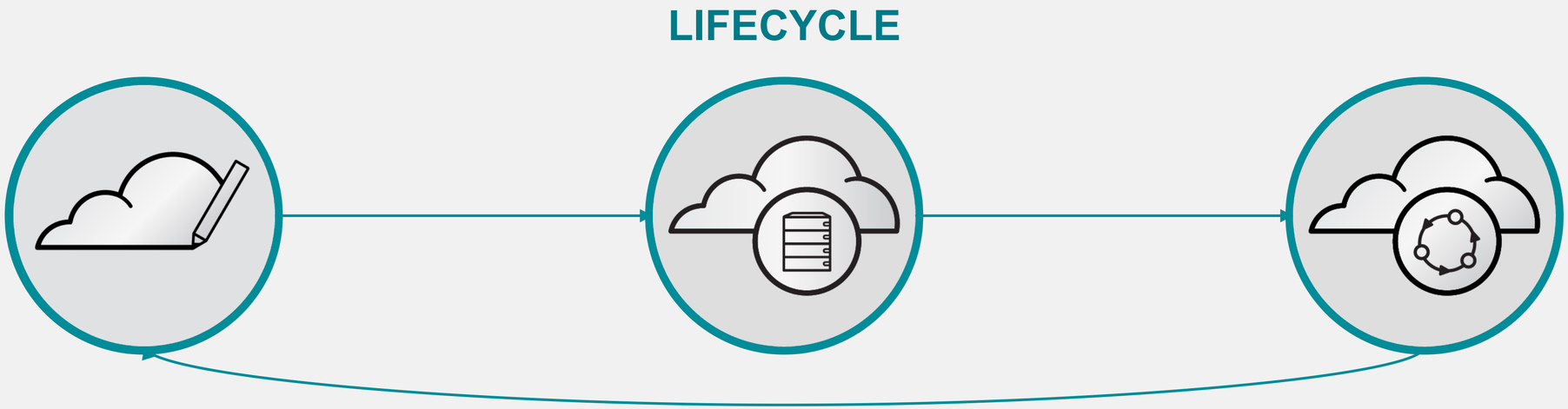
RED HAT OPENSTACK PLATFORM INSTALLER

RED HAT OPENSTACK PLATFORM DIRECTOR

SPINALSTACK

Red Hat OpenStack Platform director

Project Mission



PLANNING

Network topology
Service parameters
Resource capacity

DEPLOYMENT

Deployment orchestration
Service configuration
Sanity checks

OPERATIONS

Updates and upgrades
Scaling up and down
Change management

From Upstream to Product

UPSTREAM OPENSTACK

TripleO

MIDSTREAM COMMUNITY

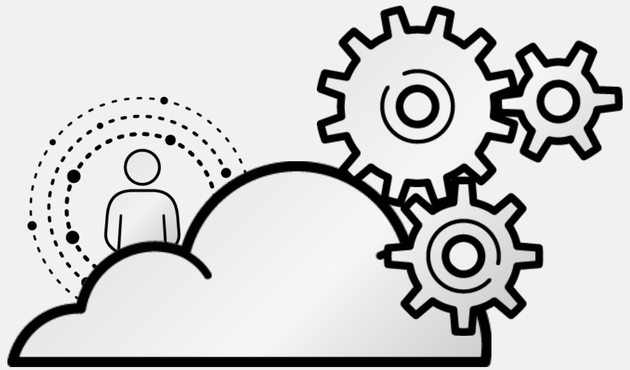
RDO

DOWNSTREAM PRODUCT

Red Hat OpenStack Platform director

RHEL OpenStack Platform director

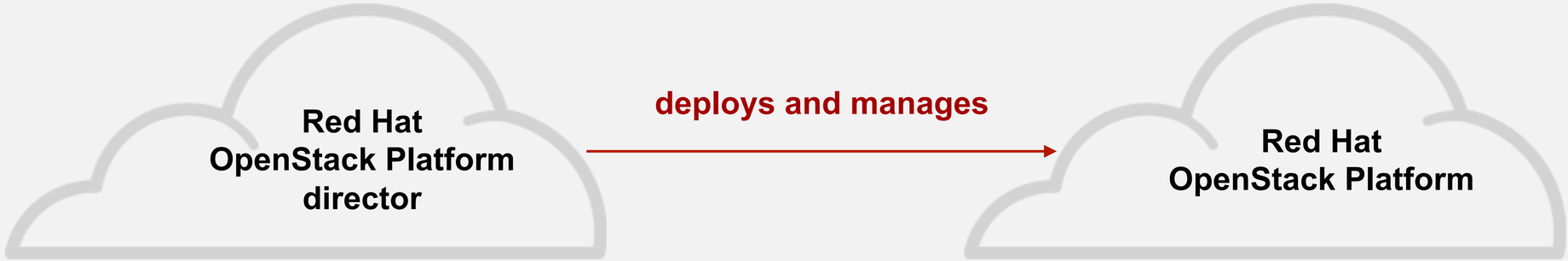
Key values



RHEL OpenStack Platform director:

-- complete OpenStack **lifecycle management**,
-- part of the upstream OpenStack **community**,
-- a rich **partner ecosystem**,
-- **scalable**, **API-based** architecture,
-- strong in community & product **support**,
-- co-engineered with the Red Hat product stack

Key Concept: We Have Two Clouds



Deployment and management cloud

- Infrastructure command and control
- Cloud operator visibility only
- Also known as the “Undercloud”

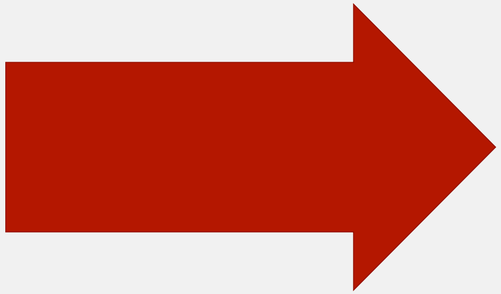
Production, tenant facing cloud

- The OpenStack you know and love
- The cloud that your tenants will use
- Also known as the “Overcloud”

The Concept of TripleO

Re-uses OpenStack components to deploy OpenStack on **hardware**

- Nova
- Neutron
- Glance
- Swift
- Heat



Bare Metal

OS image, network, disk

Hardware Node Deployment

*Only a select subset of OpenStack components are shown here.

OpenStack - Nova, Glance & Heat

TripleO uses Nova and Ironic to deploy to hardware

- Nova
- Neutron
- Glance
- Swift
- Heat

Nova provides control services for hardware node instances. Nova's scheduler matches bare metal flavors to hardware.

It orchestrates the rendering of bare metal by treating Ironic's API as a special bare metal hypervisor.

Bare Metal

OS image, network, disk

Hardware Node Deployment

*Only a select subset of OpenStack components are shown here.

OpenStack - Nova, Glance & Heat

TripleO uses Glance to uploading and accessing deployment images

- Nova
- Neutron
- Glance
- Swift
- Heat

Glance functions as an image catalog very the same way as in tenant facing OpenStack cloud.

It uploads and provides images, which are used for hardware node depoyment.

Bare Metal

OS image, network, disk

Hardware Node Deployment

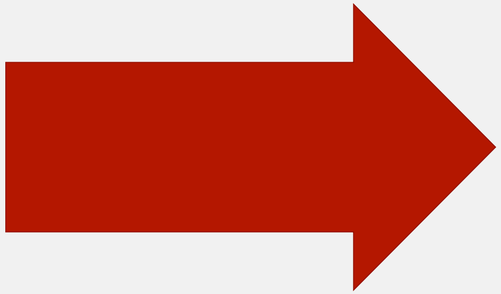
*Only a select subset of OpenStack components are shown here.

OpenStack - Nova, Glance & Heat

Heat templates encapsulate the equivalent of a cloud resource reference architecture

- Nova
- Neutron
- Glance
- Swift
- Heat

Here, Heat provides deployment orchestration for hardware nodes.



Heat templates capture hardware configuration for consistent and repeatable deployment scenarios.

Bare Metal
OS image, network, disk

Bare Metal
OS image, network, disk

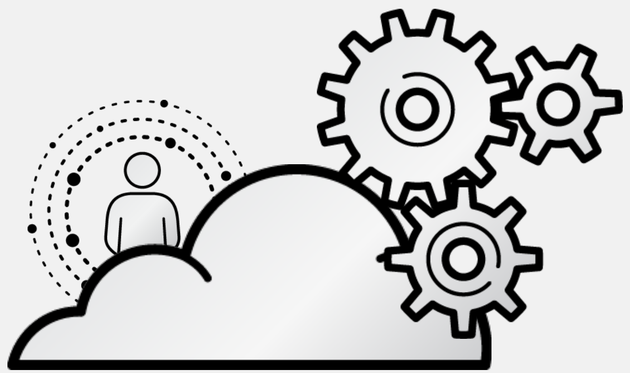
Bare Metal
OS image, network, disk

Hardware Node Deployment

*Only a select subset of OpenStack components are shown here.

Multipliers

Advantages of building OpenStack with OpenStack



Reinvestment in OpenStack
--improve TripleO, improve OpenStack

Flexibility
-- Rich, reusable descriptions

Automation and scalability
-- Scalable OpenStack APIs

Fidelity
-- Foundational components are OpenStack

HOW IT WORKS

Heat Template (1/2)

```
[stack@undercloud ~]$ cat templates/rhel-registration/rhel-registration.yaml
heat_template_version: 2014-10-16

description: >
  RHEL Registration and unregistration software deployments.

parameters:
  server:
    type: string
  # To be defined via a local or global environment in parameter_defaults
  rhel_reg_activation_key:
    type: string
…
```

Heat Template (2/2)

```
…
resources:

  RHELRegistration:
    type: OS::Heat::SoftwareConfig
    properties:
      group: script
      inputs:
        - name: REG_ACTIVATION_KEY
        - name: REG_AUTO_ATTACH
…

outputs:
  deploy_stdout:
    description: Deployment reference, used to trigger puppet apply on changes
    value: {get_attr: [RHELRegistrationDeployment, deploy_stdout]}
```

Heat Environment File and Resource Registery

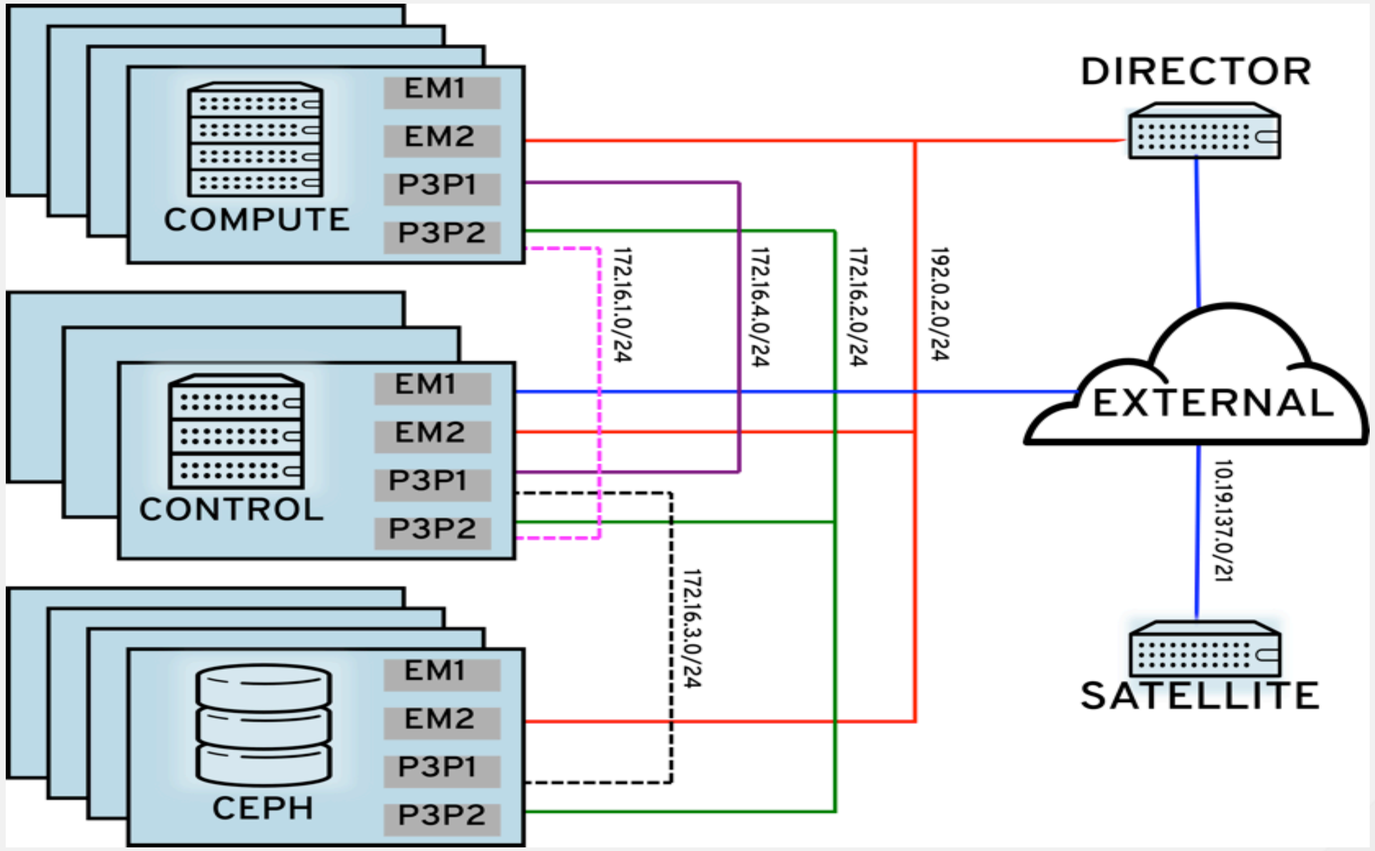
```
[stack@undercloud ~]$ cat templates/rhel-registration/environment-rhel-registration.yaml
parameter_defaults:
  rhel_reg_activation_key: OSP7
  rhel_reg_sat_url: http://sat6.e2e.bos.redhat.com
  rhel_reg_method: satellite

[stack@undercloud ~]$ cat templates/rhel-registration/rhel-registration-resource-registry.yaml
resource_registry:
  OS::TripleO::NodeExtraConfig: /home/stack/templates/rhel-registration/rhel-registration.yaml
```

Deploy Command

```
[stack@undercloud ~]$ openstack overcloud deploy --templates \
--ntp-server 10.5.26.10 \
--control-scale 3 --compute-scale 4 --ceph-storage-scale=4 \
-e ~/templates/rhel-registration/environment-rhel-registration.yaml \
-e ~/templates/rhel-registration/rhel-registration-resource-registry.yaml
```

Reference Architecture

- Server roles
- Isolated networks
- Shared Storage
- Satellite registration

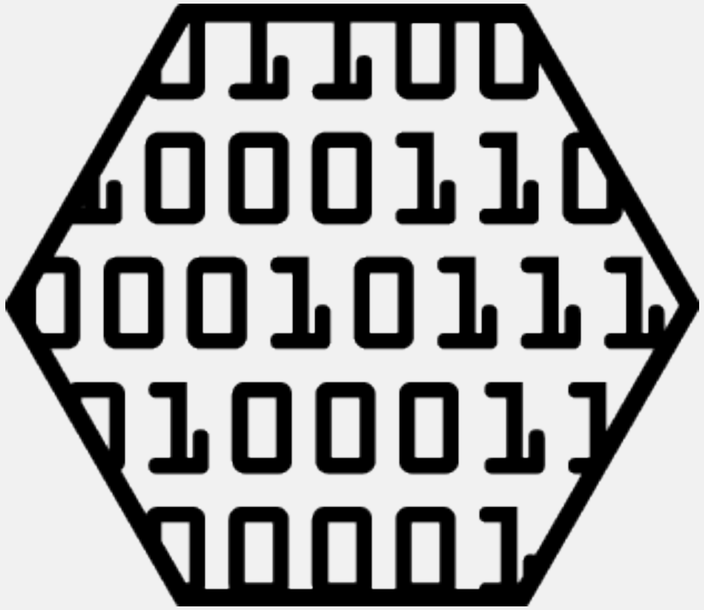


DEPLOYING

DEVELOPER MINDSET

Vision of a Software Defined Datacenter

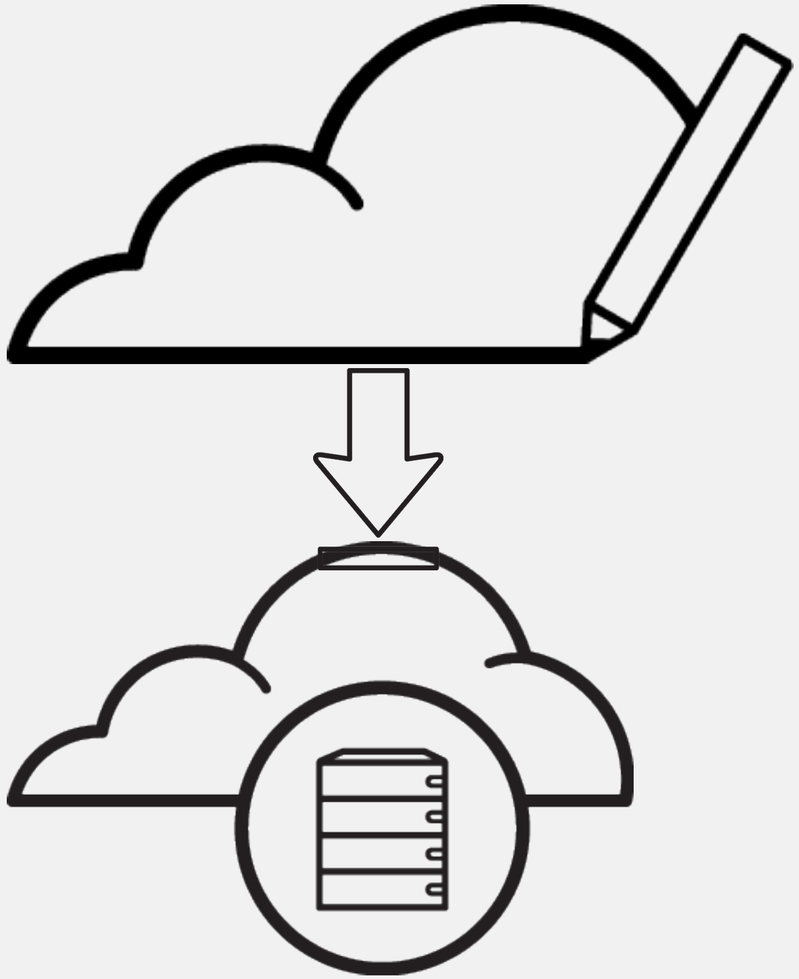
API driven
Programmatic
Resources
    … defined in software
    … deployed to commodity hardware
Scalable
Flexible



TripleO as an Integrated Development Stack

Developer Mindset

Software development lifecycle
… Plan
… Iterate
… Test
… Optimize



WORKING WITH DIRECTOR

Version Control (1/3)

```
[stack@undercloud ~]$ cd templates

[stack@undercloud templates]$ ls
deploy_command   inject-trust-anchor.yaml  overcloud-privkey.pem
storage-environment.yaml enable-tls.yaml  limits.yaml
post-deploy.yaml test firstboot-config.yaml ...

[stack@undercloud templates]$ git remote -v
origin   git@github.com:jliberma/tripleo_hardening_templates.git (fetch)
origin   git@github.com:jliberma/tripleo_hardening_templates.git (push)
```

Version Control (2/3)

```
[stack@undercloud templates]$ git status -s
 M deploy_command
 M nova_host.sh
 M scale_command
 M storage-environment.yaml

[stack@undercloud templates]$ git add -A

[stack@undercloud templates]$ git commit -a -m "update deploy to include
ceph"
[master eb9b225] update deploy to include ceph
 4 files changed, 32 insertions(+), 22 deletions(-)
 rewrite scale_command (97%)
```

Version Control (3/3)

```
[stack@undercloud templates]$ git push origin master -v
Pushing to git@github.com:jliberma/tripleo_hardening_templates.git
To git@github.com:jliberma/tripleo_hardening_templates.git
'refs/remotes/origin/master'

[stack@undercloud templates]$ git status
# On branch master
nothing to commit, working directory clean

[stack@undercloud templates]$ git log -n 1
commit eb9b2257d093b070262a9e04eca4a267f1051128
Author: Jacob Liberman <jacobliberman@gmail.com>
Date:   Tue Jun 21 17:55:40 2016 -0400

    update deploy to include ceph
```

Sandbox Environment in Virt (1/5) – Configure Host

1. Enable nested KVM

```
# cat /etc/modprobe.d/kvm_intel.conf
options kvm-intel nested=1
options kvm-intel enable_shadow_vmcs=1
options kvm-intel enable_apicv=1
options kvm-intel ept=1
```

2. Enable communication between the host and guests

```
# cat /etc/sysctl.d/98-rp-filter.conf
net.ipv4.conf.default.rp_filter = 0
net.ipv4.conf.all.rp_filter = 0
```

Sandbox Environment in Virt (2/5) Configure Hypervisor

1. Verify system supports nested virtual machines.

```
# egrep -c '(vmx|svm)' /proc/cpuinfo
16
```

2. Install and enable libvirt + kvm

```
# modprobe kvm && modprobe kvm_intel
# yum install libvirt qemu-kvm virt-install libguestfs-tools -y
# systemctl enable libvirtd && systemctl start libvirtd
```

3. Configure a provisioning network.

```
# virsh net-list
Name                 State      Autostart     Persistent
----------------------------------------------------------
default              active     yes           yes
provisioning         active     yes           yes
```

Sandbox Environment in Virt (3/5) Remote Access

1. Allow password-less remote access to libvirt via SSH to simulate IPMI

```
# cat /etc/polkit-1/localauthority/50-local.d/50-libvirt-user-stack.pkla
[libvirt Management Access]
Identity=unix-user:stack
Action=org.libvirt.unix.manage
ResultAny=yes
ResultInactive=yes
ResultActive=yes

undercloud$ ssh-copy-id -i ~/.ssh/id_rsa.pub stack@192.168.122.1

undercloud$ virsh --connect qemu+ssh://stack@192.168.122.1/system list
Id    Name                State
----------------------------------------------------
1  undercloud          running
```

Sandbox Environment in Virt (4/5) Deploy Cloud Systems

1. Create virtual machine for overcloud nodes

```
# virt-install --ram 8192 --vcpus 4 --os-variant rhel7 --disk \
path=/var/lib/libvirt/images/overcloud-node
$i.qcow2,device=disk,bus=virtio,format=qcow2 \
--noautoconsole --vnc --network network:provisioning \
--network network:default --name overcloud-node \
--cpu SandyBridge,+vmx
```

Sandbox Environment in Virt (5/5) Deploy Cloud Systems

```
# undercloud$ cat ~/instackenv.json
{ "ssh-user": "stack",
"ssh-key": "$(cat ~/.ssh/id_rsa)",
"power_manager”:
"nova.virt.baremetal.virtual_power_driver.VirtualPowerManager",
"host-ip": "192.168.122.1",
"nodes":
 [ { "name": "overcloud-node1",
"pm_addr": "192.168.122.1",
"pm_password": "$(cat ~/.ssh/id_rsa)",
"pm_type": "pxe_ssh",
"mac": [ "52:54:00:6d:66:26” ],
      "cpu": "4",
      "memory": "8192",
      "disk": "60” ...
```

Validation Tools (1/2) Standalone

```
[stack@undercloud ~]$ openstack baremetal instackenv validate
SUCCESS: found 0 errors

[stack@undercloud ~]$ openstack overcloud netenv validate -f ~/
templates/network-environment.yaml
SUCCESSFUL Validation with 0 error(s)
```

Validation Tools (2/2) Clapper

https://github.com/rthallisey/clapper/blob/master/README.md

```
[stack@undercloud ~]$ sudo yum install -y ansible

[stack@undercloud ~]$ git clone https://github.com/rthallisey/
clapper.git

[stack@undercloud ~]$ cd clapper

[stack@undercloud clapper]$ ssh heat-admin@172.16.0.88 'python' <
check_overcloud_controller_settings.py
Found potential issues in /etc/my.cnf.d/galera.cnf:
   * mysqld max_connections is unset, recommend at least 4096
Found potential issues in /etc/haproxy/haproxy.cfg:
   * global maxconn is unset, recommend at least 20480
```

Post-Deployment Testing

Test deployed Overcloud performance and scaling using common benchmark workloads

1. Baseline performance and scaling using Rally
https://wiki.openstack.org/wiki/Rally

2. Use Browbeat to identify and correct potential performance issues
https://browbeatproject.org/

3. Re-configure and re-test

Reference: *Guidelines and Considerations for Performance and Scaling OSP 7*
https://access.redhat.com/articles/2165131

DEPLOYMENT BEST PRACTICES

Undercloud (1/2)

1. Run as a virtual machine
   -- Snapshot: roll back or forward
   -- Protect with RHEV mechanisms
2. Static IP address assignment from Director node
3. External default gateway

```
[root@redhat ~]# virsh list
 Id    Name                           State
----------------------------------------------------
 9     undercloud                     running

[root@redhat ~]# virsh snapshot-list undercloud
 Name                 Creation Time             State
------------------------------------------------------------
 undercloud-snap-1    2016-06-16 14:03:36 +0000 running
 undercloud-snap-4    2016-06-16 19:59:04 +0000 running
```

Undercloud (2/2)

1. Optimize Nova settings for working at scale
2. Optimize Ironic settings for working at scale

```
[root@undercloud ~]# openstack-config --get /etc/nova/nova.conf DEFAULT
rpc_response_timeout
600
[root@undercloud ~]# openstack-config --get /etc/nova/nova.conf DEFAULT
max_concurrent_builds
4
[root@undercloud ~]# openstack-config --get /etc/ironic/ironic.conf DEFAULT
rpc_response_timeout
600
[root@undercloud ~]# openstack-config --get /etc/ironic/ironic.conf DEFAULT
rpc_thread_pool_size
8
```

Overcloud (1/6) Controllers

1. Out of box settings appropriate for most use cases
   -- 3 controllers, 32 logical CPUs per controller

2. Increase controller connection settings when needed:

```
$ cat ~/templates/limits.yaml
parameters:
  MysqlMaxConnections: 8192
  RabbitFDLimit: 65436
  controllerExtraConfig:
    tripleo::loadbalancer::haproxy_default_maxconn: 8192
```

3. Deploy:

```
$ openstack overcloud deploy –templates --ntp-server 10.5.26.10 --control-scale
3 --compute-scale 1 –e ~/templates/limits.yaml
```

Overcloud (2/6) Controllers

4. Test new connection limits:

```
$ sudo grep max_connections /etc/my.cnf.d/galera.cnf
max_connections = 8192

$ sudo grep maxconn /etc/haproxy/haproxy.cfg
  maxconn  20480
  maxconn  8192

$ sudo cat /etc/security/limits.d/rabbitmq-server.conf
rabbitmq soft nofile 65436
rabbitmq hard nofile 65436

$ sudo rabbitmqctl report | grep file_descriptor
 {file_descriptors,[{total_limit,65336},
 {file_descriptors,[{total_limit,65336},
 {file_descriptors,[{total_limit,65336},
```

Overcloud (3/6) Ceph storage – via director

Via director:

-- Use `-e ~/templates/environments/storage-environment.yaml`

Considerations:

-- Director co-locates controllers and monitors – spec as monitors

-- Tune parameters per Ceph best practices:

-- OSD journal location, size, disk type (defaults set for PoC)

-- pg_num, pgp_num, etc

Calculating placement groups:
https://access.redhat.com/documentation/en/red-hat-ceph-storage/1.3/storage-strategies/chapter-14-pg-count

-- # OSDs * 100 / # pool replicas -> round result UP to nearest power of 2

Overcloud (4/6) Ceph storage – standalone

Standalone:

- Use `-e ~/templates/puppet/extraconfig/ceph/puppet-ceph-external.yaml`
- Set:
  - `ceph_storage_count`
  - `ceph_external_mon_ips`
  - `ceph_mon_ips`

Considerations:

- Marrying cloud and storage lifecycle
- Shared storage with other datacenter resources
- Re-attach to a different OSP cluster if needed

Overcloud (5/6) Interface bonding

1. LACP bonding to combine provisioning interface with bond and redundant switches

```
$ openstack overcloud deploy --templates ~/templates -e ~/templates/network-
environment.yaml -e ~/templates/environments/net-bond-with-vlans.yaml
```

~/templates/network-enviroments.yaml

```
parameter_defaults:
  BondInterfaceOvsOptions: "mode=802.3ad"
```

~/templates/environments/net-bond-with-vlans.yaml

```
resource_registry:
  OS::TripleO::Compute::Net::SoftwareConfig: ../network/config/bond-with-vlans/
compute.yaml
```

Overcloud (6/6) Interface bonding

~/templates/network/config/bond-with-vlans/compute.yaml:

```
type: ovs_bridge
        members:
                -
                  type: linux_bond
                  name: bond0
                  bonding_options: {get_param: BondInterfaceOvsOptions}
                  members:
                    -
                      type: interface
                      name: enp7s0
                      primary: true
                    -
                      type: interface
                      name: enp8s0
                -
                  type: vlan
                  device: bond0
```

SCALING

Scale Overview

- Compute nodes can be scaled up or down
- Ceph storage nodes can be scaled up

Red Hat, Dell, Cumulus Networks:
-- scaled to 301 physical compute nodes
https://www.youtube.com/watch?v=VXdYB0Xm1Ak

Scale/Update/Upgrade Checklist (1/2)

1.Check Overcloud Heat stack status on the director node:

```
$ heat stack-list
```

-- Should show status CREATE_COMPLETE or UPDATE_COMPLETE

2. Check Pacemaker service status on a controller:

```
$ sudo pcs status
```

-- All services should be running on all nodes

3. Check Galera sync status and replication size on all controller nodes:

```
$ sudo mysql --exec=\"SHOW STATUS LIKE 'wsrep_local_state_comment'\”
$ sudo mysql --exec=\"SHOW STATUS LIKE 'wsrep_cluster_size'\”
```

-- The state should be ‘synced’

-- The cluster size should be ‘3’

Scale/Update/Upgrade Checklist (2/2)

4. Check RabbitMQ status on a controller node:

```
$ sudo rabbitmqctl cluster_status
```

-- Should list all controller nodes on the running_nodes key

5. Check Nova status on the Overcloud node:

```
$ sudo systemctl status openstack-nova-compute
$ nova hypervisor-list
```

-- All compute nodes should show as status ‘Up’ and ‘Enabled’

6. Check Ironic for available systems:

```
$ ironic node-list
```

-- Available systems should be available, maintenance = false, and tagged with correct flavor

7. Make sure Undercloud OpenStack services are running:

```
$ sudo systemctl –t service | grep –e ‘openstack|neutron’
```

Scale Command

1. Deploy command:

```
[stack@undercloud ~]$ openstack overcloud deploy --templates \
--ntp-server 10.5.26.10 \
--control-scale 3 --compute-scale 4 --ceph-storage-scale=4 \
-e ~/templates/rhel-registration/environment-rhel-registration.yaml \
-e ~/templates/rhel-registration/rhel-registration-resource-registry.yaml
```

2. Scale command:

```
[stack@undercloud ~]$ openstack overcloud deploy --templates \
--ntp-server 10.5.26.10 \
--control-scale 3 --compute-scale 8 --ceph-storage-scale=4 \
-e ~/templates/rhel-registration/environment-rhel-registration.yaml \
-e ~/templates/rhel-registration/rhel-registration-resource-registry.yaml
```

Managing at Scale (1/2)

1.Parallel distributed shell: https://code.google.com/archive/p/pdsh/

2.Install:

```
$ cd pdsh-2.29
$ ./configure --with-ssh --without-rsh --with-machines=/etc/pdsh/machines
$ make && make install
$ cat /etc/pdsh/machines
overcloud-controller-0
overcloud-controller-1
overcloud-controller-2
overcloud-compute-0
overcloud-compute-1
```

Managing at Scale (2/2)

```
$ pdsh -a -l heat-admin uname -r
overcloud-controller-0: 3.10.0-327.10.1.el7.x86_64
overcloud-controller-1: 3.10.0-327.10.1.el7.x86_64
overcloud-compute-0: 3.10.0-327.10.1.el7.x86_64
overcloud-controller-2: 3.10.0-327.10.1.el7.x86_64
overcloud-compute-1: 3.10.0-327.10.1.el7.x86_64

$ pdsh -l heat-admin -w overcloud-compute-[0-2] rpm -q kernel | dshbak -c
----------------
overcloud-compute-[0-1]
----------------
kernel-3.10.0-327.10.1.el7.x86_64
kernel-3.10.0-327.18.2.el7.x86_64
----------------
overcloud-compute-2
----------------
kernel-3.10.0-327.10.1.el7.x86_64
```

FUTURE DIRECTIONS

Composable Services Within Roles

“Split up our monolithic template architecture so we can cleanly encapsulate each service deployment in a separate template, and more easily select which services are deployed/enabled on a particular role.”

https://blueprints.launchpad.net/tripleo/+spec/composable-services-within-roles

Split stack: TripleO with Already Deployed Servers

- Using TripleO with already deployed and provisioned servers
- Nova and Ironic would not be used to do the initial install
- Split single Overcloud stack into two or more stacks with different primary responsibilities:
- Infrastructure provisioning, OpenStack configuration
- Architecture change

http://blog-slagle.rhcloud.com/?p=299

TripleO + Containers

- Two models:
  - -- Containers in OpenStack
  - -- OpenStack in Containers

- Compute role in Docker container (OSP 8)
  - -- Heat + Puppet orchestration
  - -- Evaluating approach for other services

Work Based on the upstream Kolla project:
https://github.com/openstack/kolla

TripleO + Ansible

- Ansible: Beloved framework for simple IT automation
  -- Acquired by Red Hat in OCT 2015
  -- Ansible 2.1 released MAY 2015

- Integration with OpenStack:
  -- Ceph installer
  -- Clapper validation tools
  -- TripleO split stack?

- Red Hat's track record with other acquisitions:
  -- ManageIQ (CloudForms)
  -- Inktank (Ceph provider)
  -- eNovance (OpenStack)