Deploying OpenStack at Scale

...with TripleO, Ansible and Containers

OpenStack Summit, Sydney

6th November 2017

Steve Hardy (shardy@redhat.com)

Who am I

- Software Engineer at Red Hat
- Working on OpenStack since 2012
- Heat & TripleO Core reviewer
- Former Heat & TripleO PTL

HEAT
an OpenStack Community Project

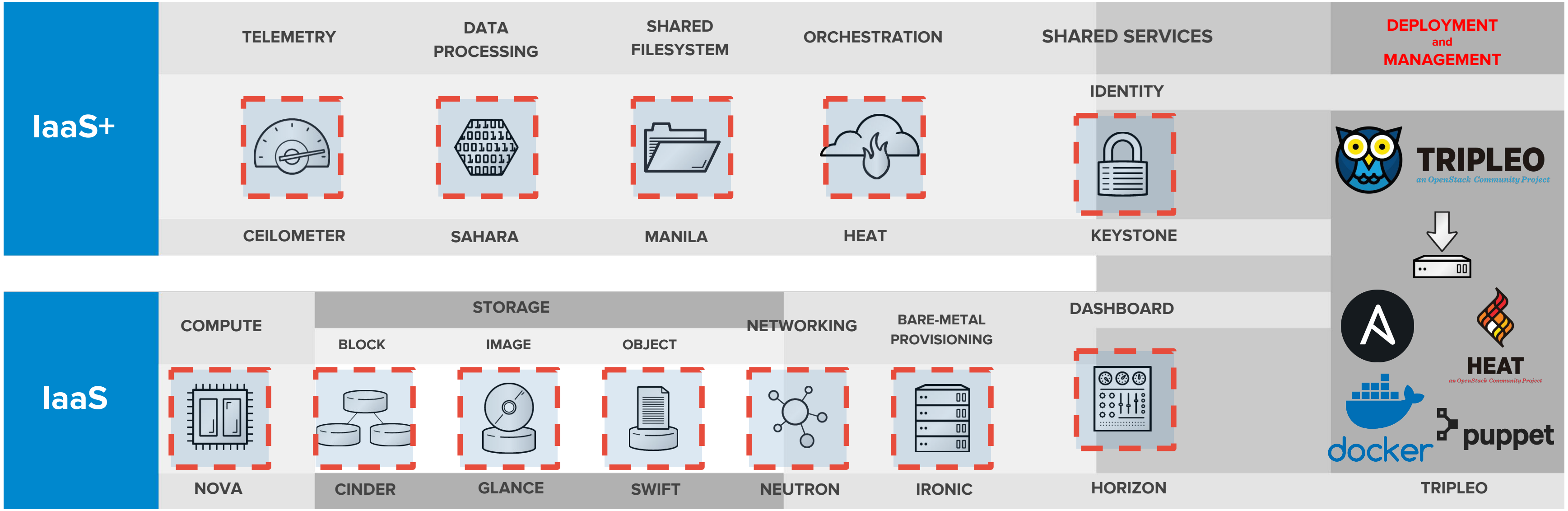
TRIPLEO
an OpenStack Community Project

openstack®

redhat.

What is this talk about?

What this talk IS about.
Deploying/managing the OpenStack services



IaaS+

TELEMETRY — CEILOMETER

DATA PROCESSING — SAHARA

SHARED FILESYSTEM — MANILA

ORCHESTRATION — HEAT

SHARED SERVICES

IDENTITY — KEYSTONE

DASHBOARD — HORIZON

DEPLOYMENT and MANAGEMENT

TRIPLEO an OpenStack Community Project

HEAT an OpenStack Community Project

docker

puppet

TRIPLEO

IaaS

COMPUTE — NOVA

STORAGE

BLOCK — CINDER

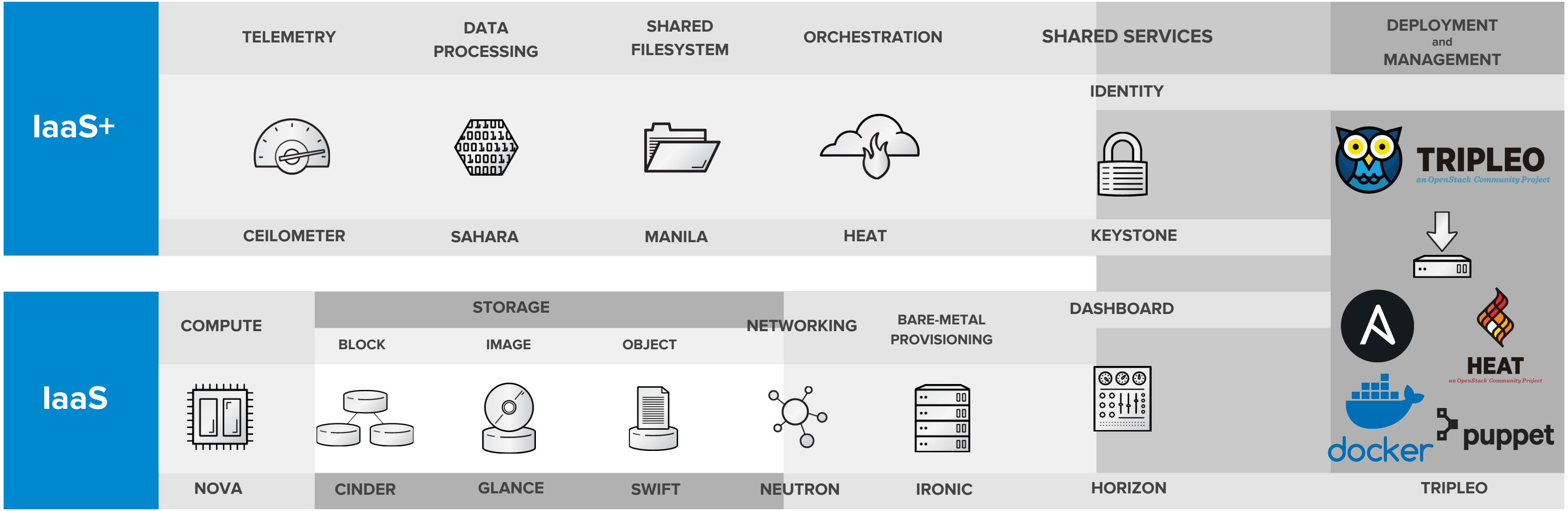
IMAGE — GLANCE

OBJECT — SWIFT

NETWORKING — NEUTRON

BARE-METAL PROVISIONING — IRONIC

What this talk IS NOT about.
Containerizing workloads **RUNNING ON** OpenStack.



IaaS+

TELEMETRY — CEILOMETER

DATA PROCESSING — SAHARA

SHARED FILESYSTEM — MANILA

ORCHESTRATION — HEAT

SHARED SERVICES

IDENTITY — KEYSTONE

DASHBOARD — HORIZON

DEPLOYMENT and MANAGEMENT — TRIPLEO

IaaS

COMPUTE — NOVA

STORAGE: BLOCK — CINDER; IMAGE — GLANCE; OBJECT — SWIFT

NETWORKING — NEUTRON

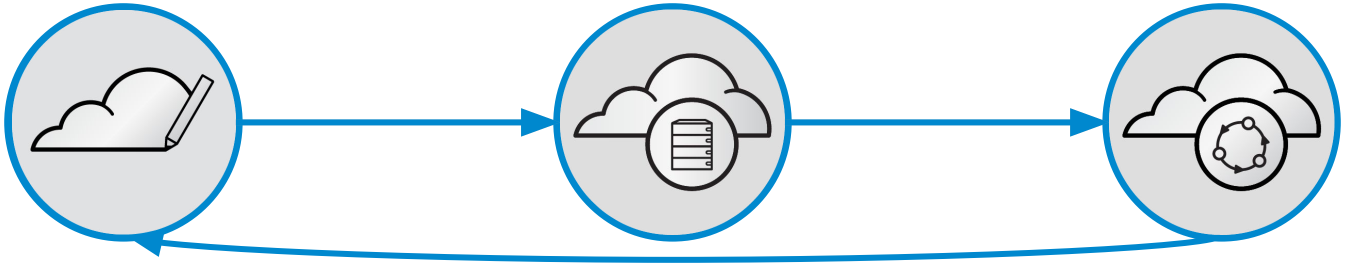
BARE-METAL PROVISIONING — IRONIC

TRIPLEO
an OpenStack Community Project
HEAT
an OpenStack Community Project
docker
puppet

Introduction to TripleO

TRIPLEO

an OpenStack Community Project



PLANNING

Network topology

Service parameters

Resource capacity

DEPLOYMENT

Pre-flight checks/validations

Deployment orchestration

Service configuration

OPERATIONS

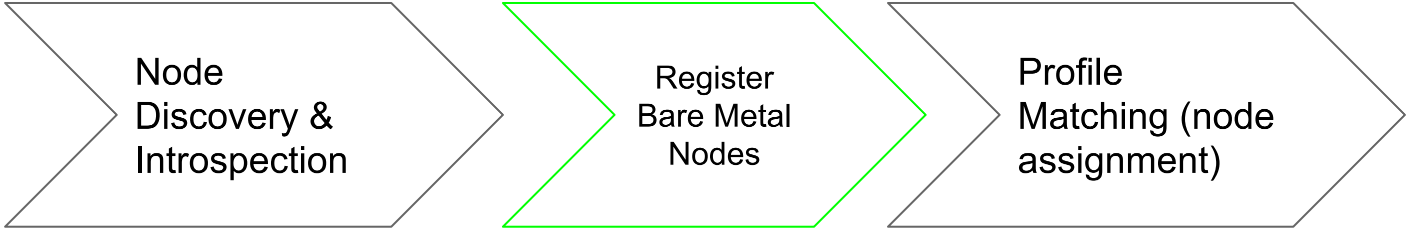
Major version upgrades

Applying minor updates

Scaling up and down

Planning

Manage hardware inventory, capacity planning



Node Discovery & Introspection

Register Bare Metal Nodes

Profile Matching (node assignment)

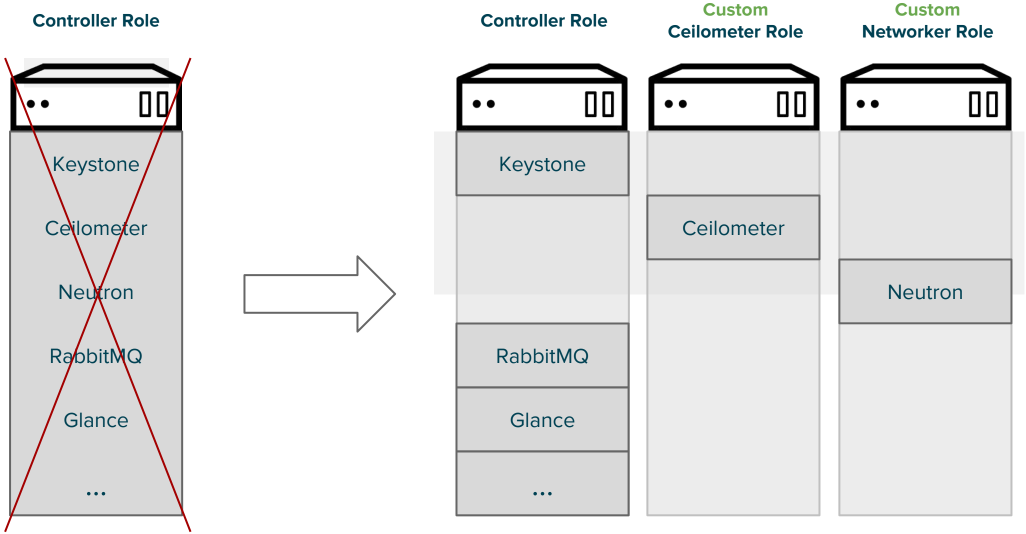
Controller Nodes

Compute Nodes

Networker Nodes

Planning

Composable roles, customize enabled services and placement

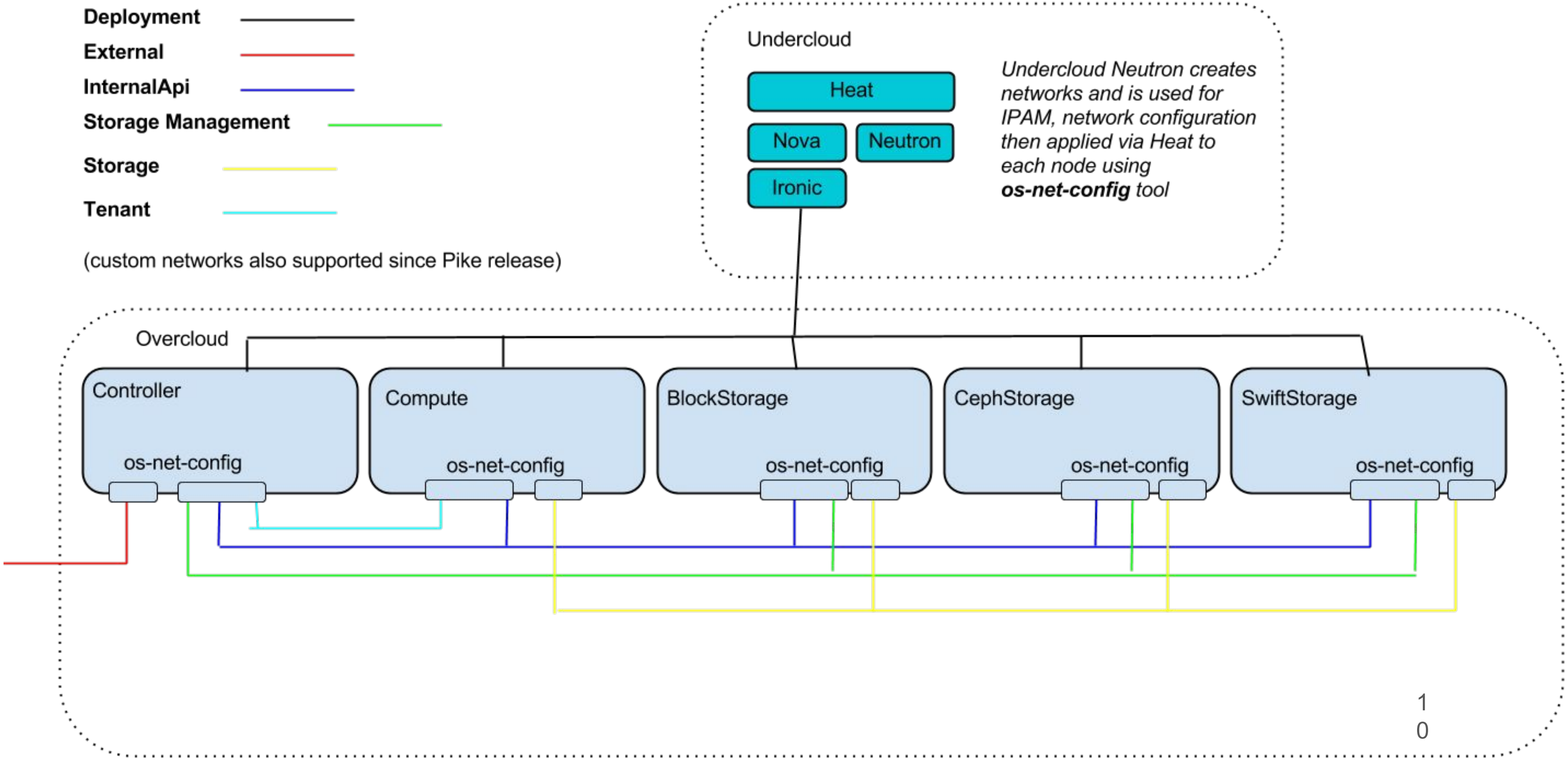


Note TripleO role != ansible role...

✓

Planning

Customizable network isolation, declarative nic configuration



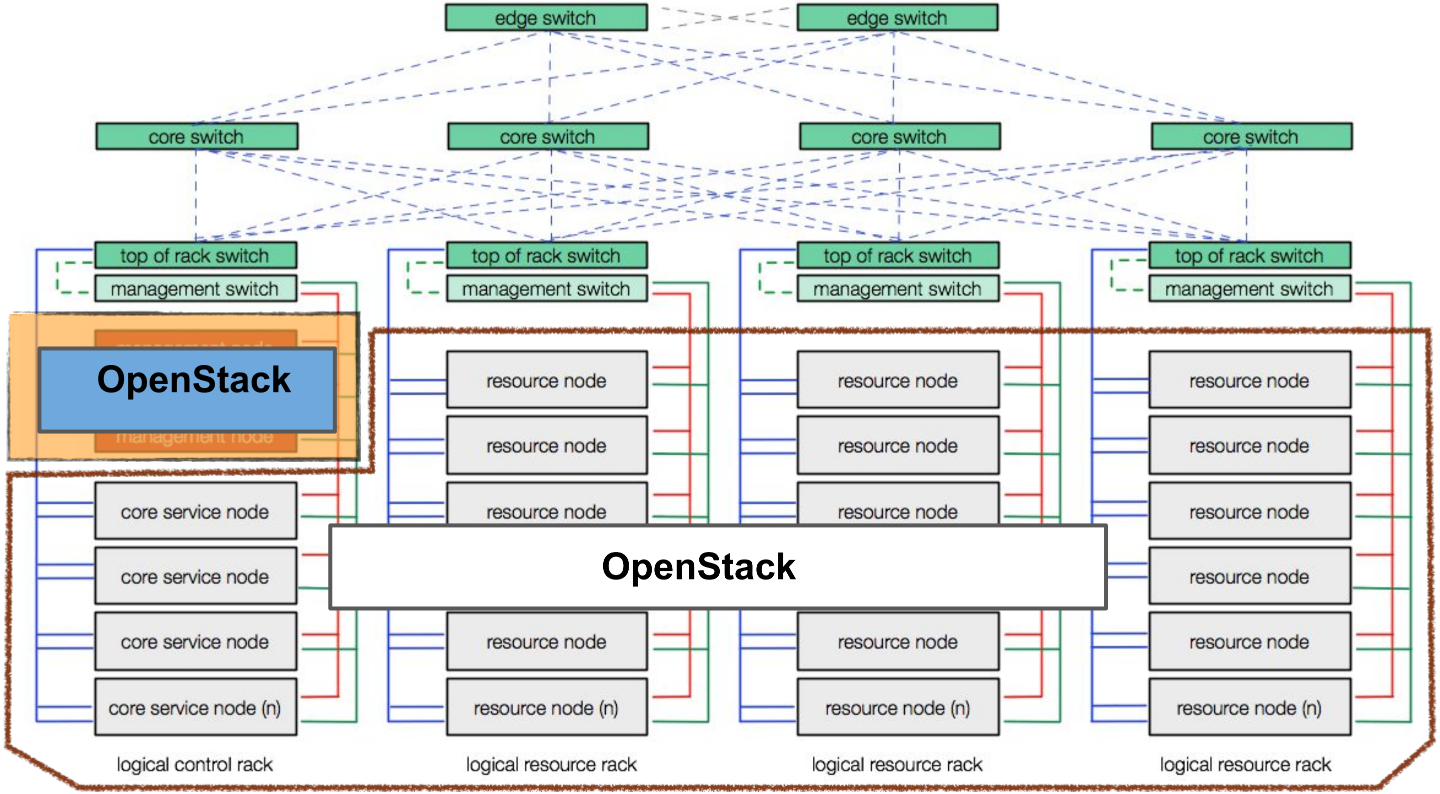
(custom networks also supported since Pike release)

Undercloud Neutron creates networks and is used for IPAM, network configuration then applied via Heat to each node using ***os-net-config*** *tool*

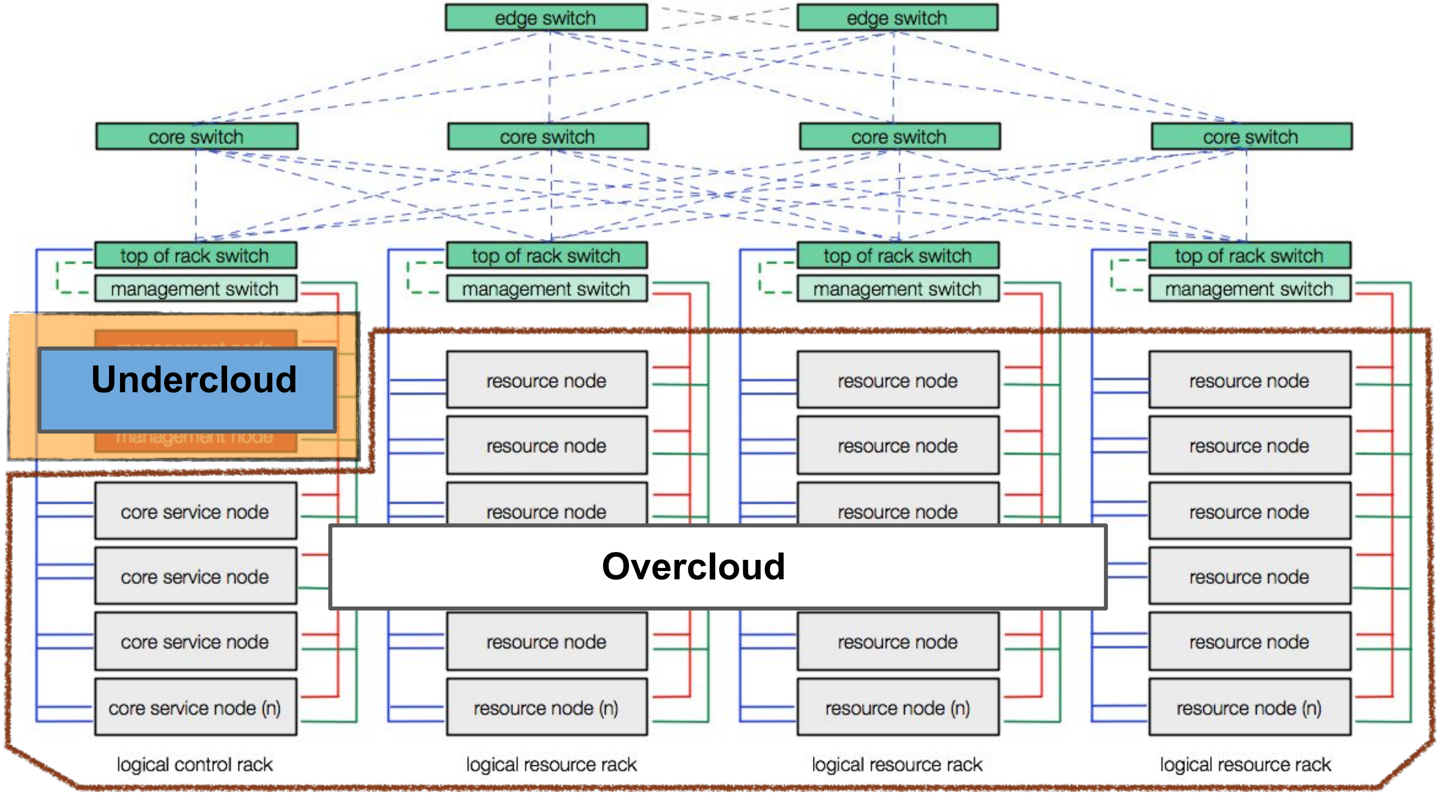
TripleO - Planning

- Node discovery, introspection & grouping (Ironic)
- Customizable “roles” (group of nodes)
- Pluggable “composable services” (Heat templates)
- Per-service network isolation (Neutron/os-net-config)
- HA support (Pacemaker)
- Predictable placement & preassigned IPs (Ironic/Nova)
- Flexible service configuration (Heat parameters)
- Version control your infrastructure “as code” (yaml)

TripleO - Deployment

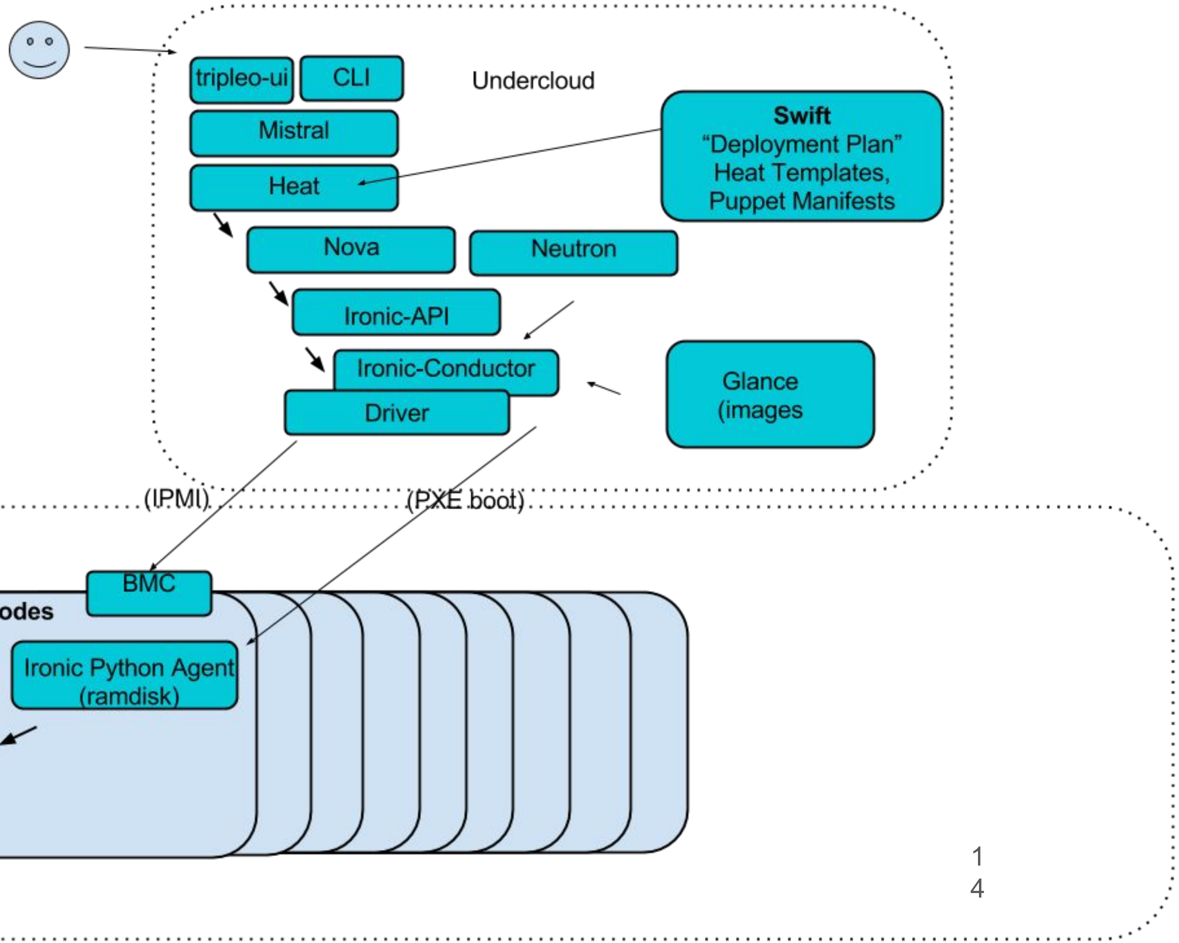


TripleO - Deployment



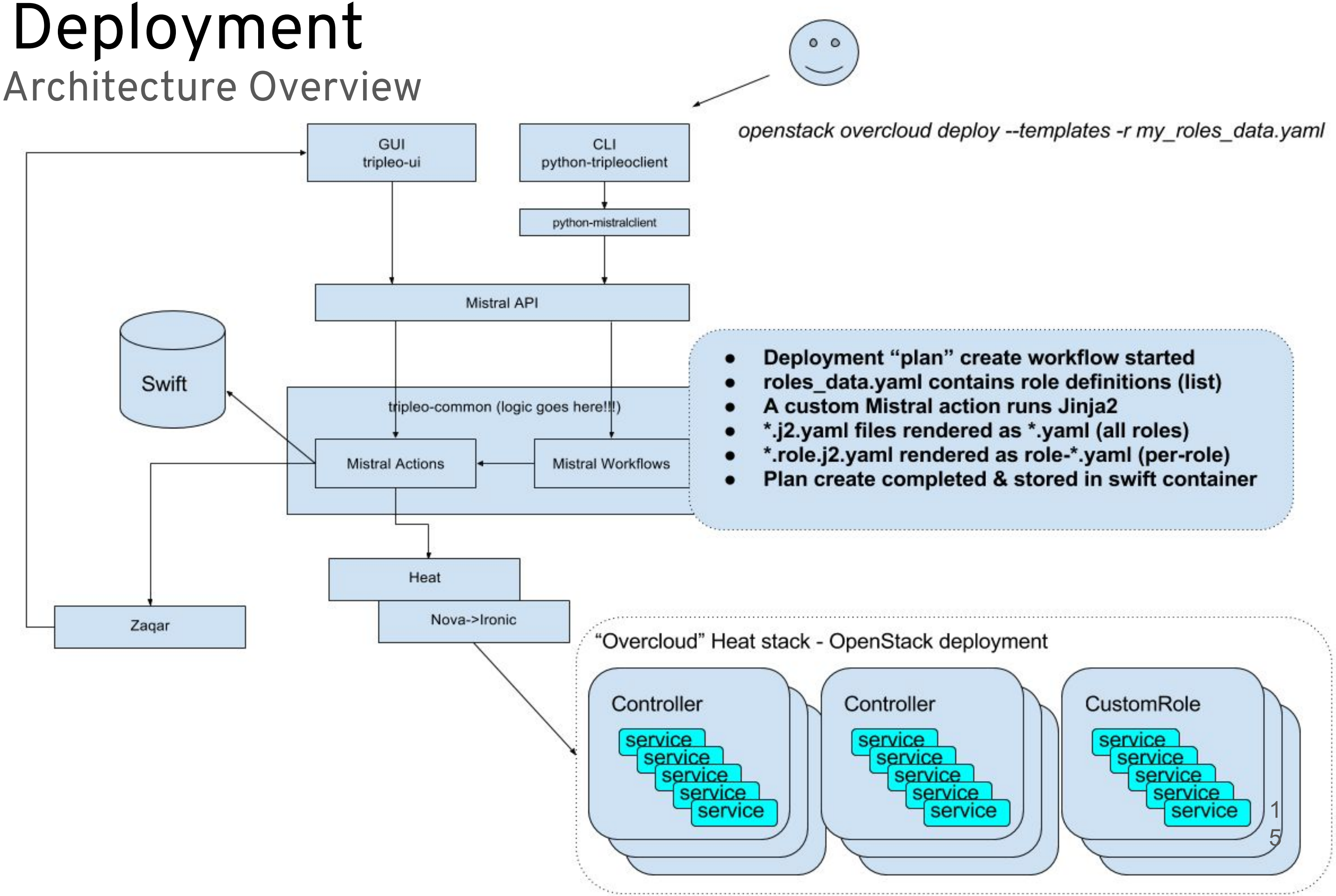
Deployment

Architecture Overview



Deployment

Architecture Overview



```
openstack overcloud deploy --templates -r my_roles_data.yaml
```

- Deployment “plan” create workflow started
- roles_data.yaml contains role definitions (list)
- A custom Mistral action runs Jinja2
- *.j2.yaml files rendered as *.yaml (all roles)
- *.role.j2.yaml rendered as role-*.yaml (per-role)
- Plan create completed & stored in swift container

TripleO - Deployment

- Pre-flight validations (Ansible based)
- Disk image based deployment (Glance)
- Baremetal provisioning (Ironic)
- Baremetal service configuration (Puppet)
- Heat templates define each service & configuration
- Mistral workflow API provides pluggable interface for UX
- Increasingly using Ansible “under the hood”

TripleO - Operations

- Automated configuration of monitoring clients (Sensu)
- Support for centralized logging
- Major version upgrades, orchestrated by Heat+Ansible
- Minor version updates - rolling no downtime
- Automated scale up to add capacity
- Orchestrated scale down and node removal

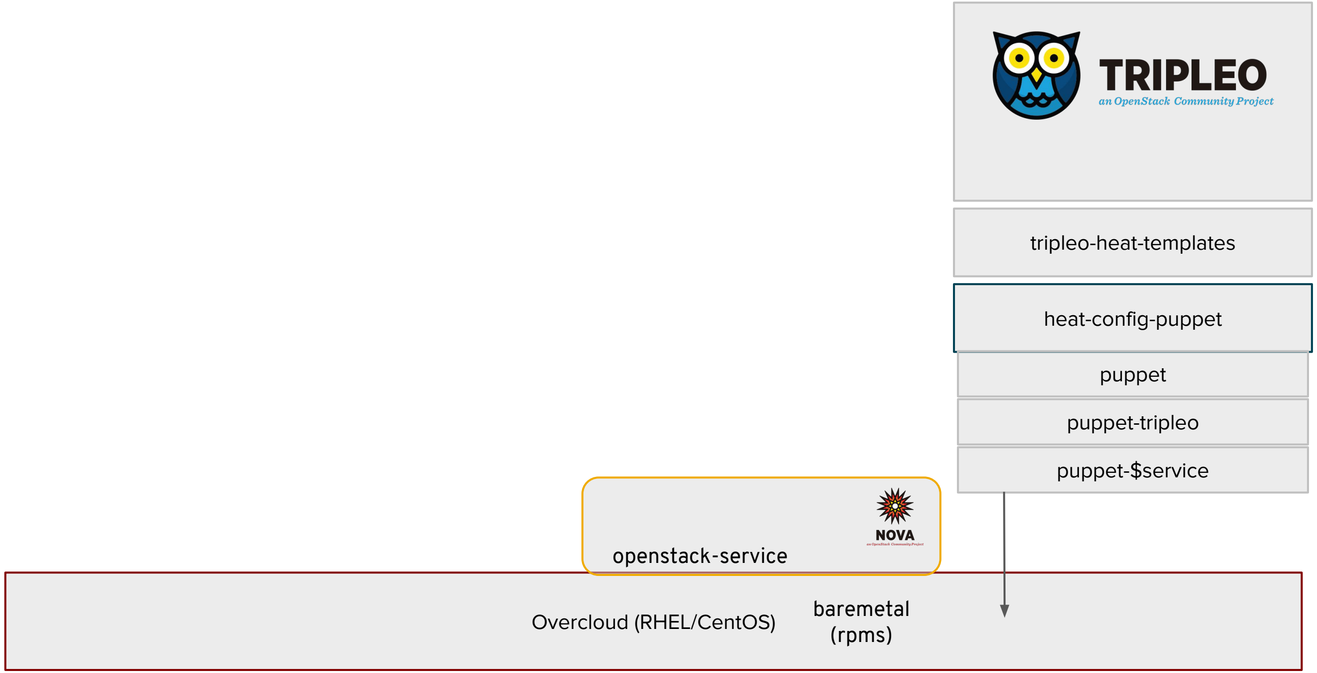
TripleO + Ansible

TRIPLEO

an OpenStack Community Project

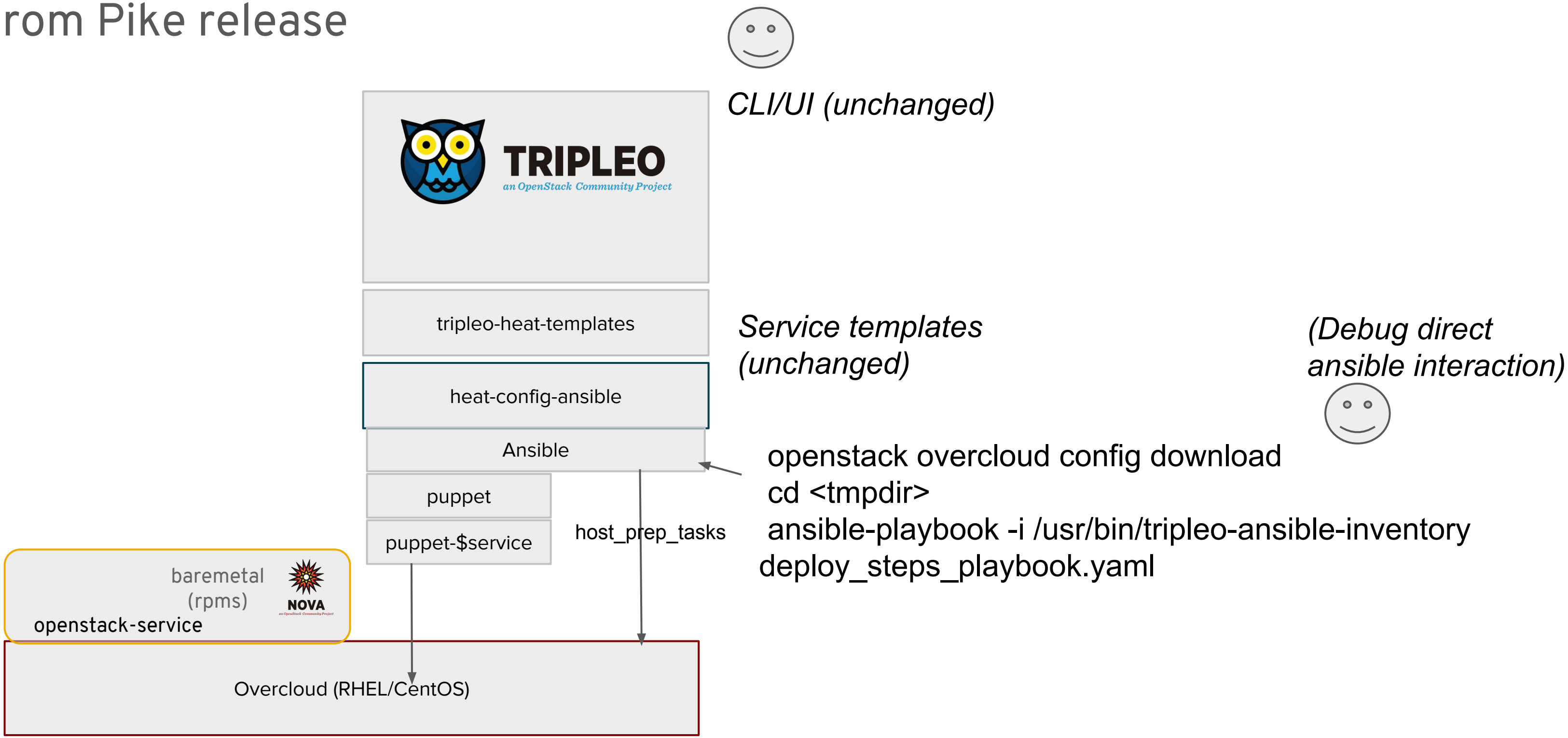
Heat+Puppet Architecture

How we used to do it...



Heat+Ansible Architecture

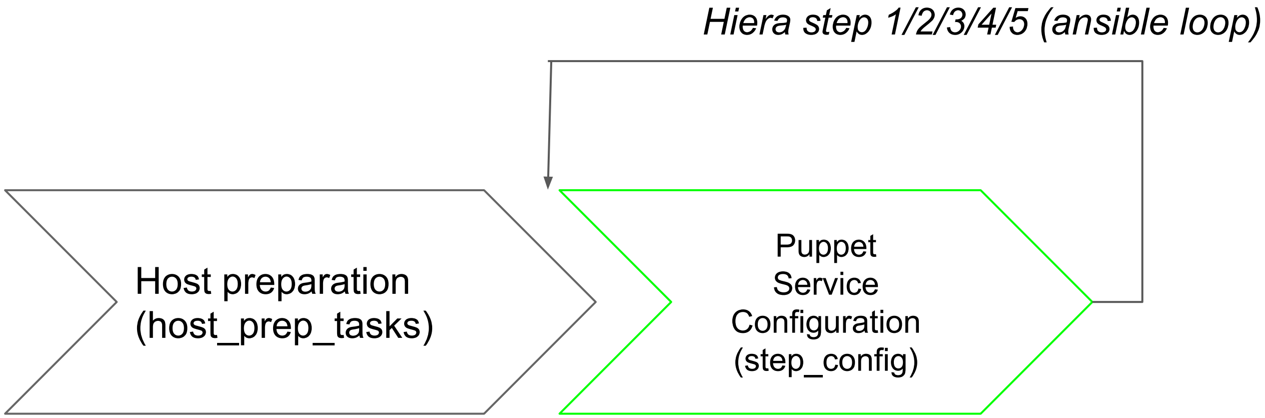
From Pike release



```
openstack overcloud config download
cd <tmpdir>
ansible-playbook -i /usr/bin/tripleo-ansible-inventory
deploy_steps_playbook.yaml
```

Deployment

Ansible + puppet deploy workflow



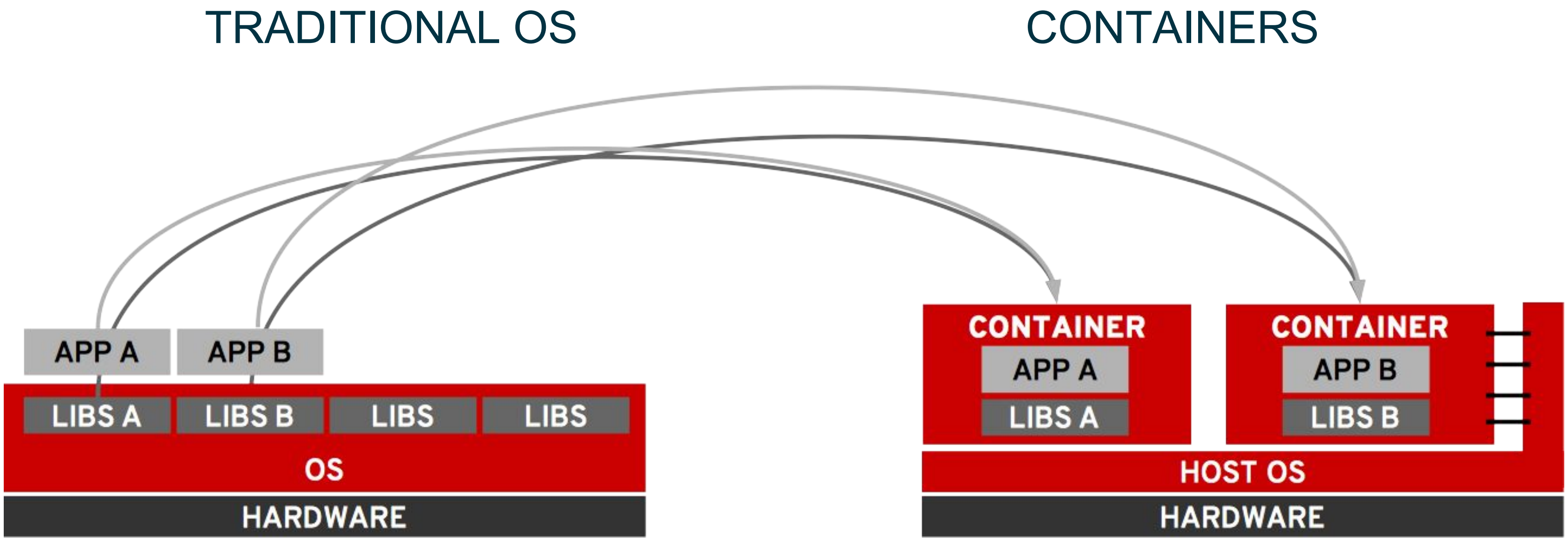
Hiera step 1/2/3/4/5 (ansible loop)

Host preparation (host_prep_tasks)

Puppet Service Configuration (step_config)

Deployment with Containers

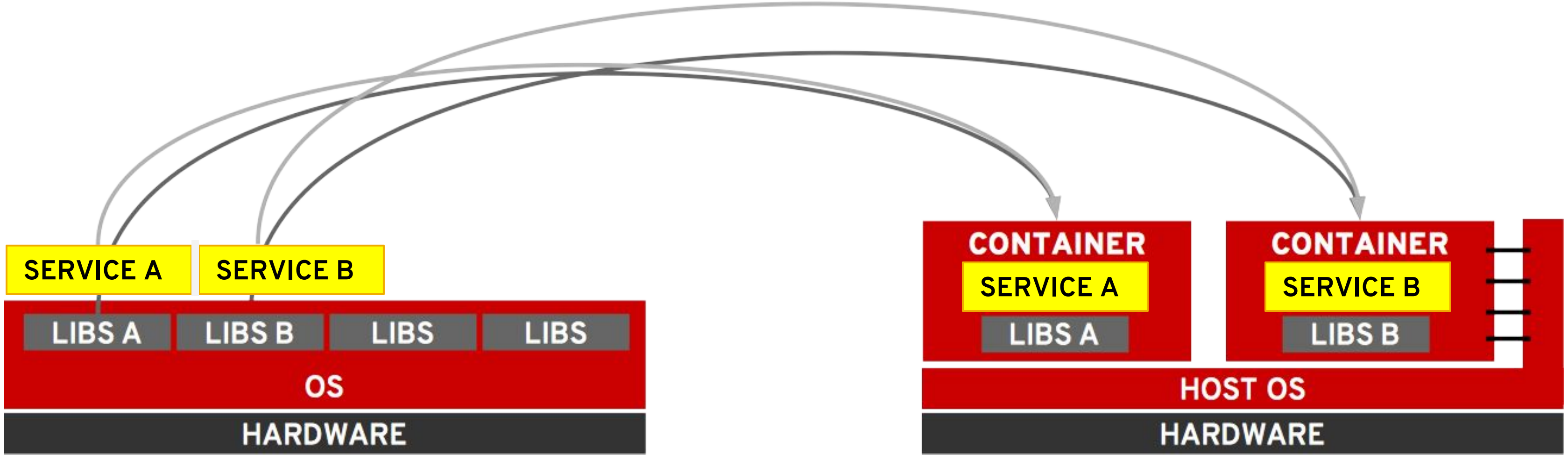
Containers for apps …



Containers for OpenStack services

TRADITIONAL OPENSTACK

CONTAINERIZED OPENSTACK

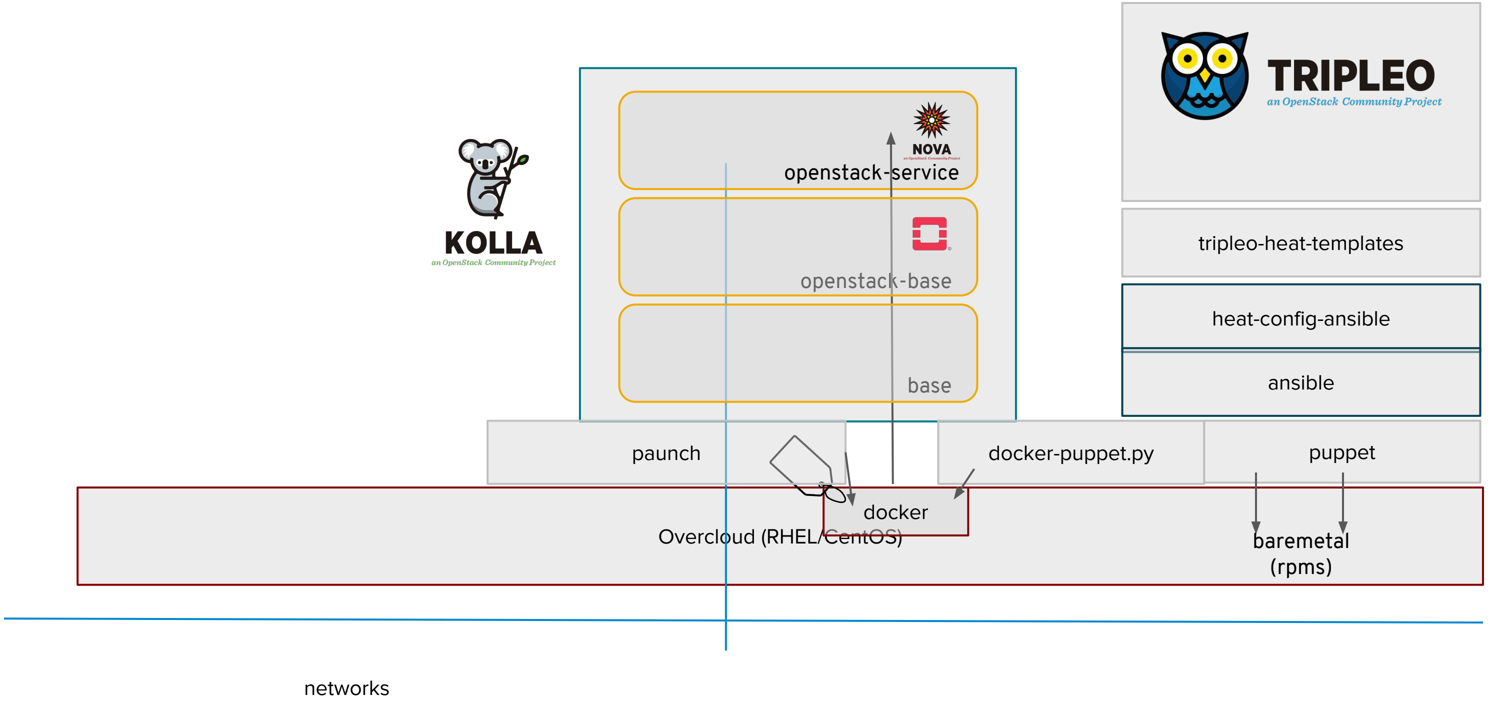


TripleO - Containers

- Pike release supports most services in containers
- A hybrid baremetal/container model is possible
  - Assists with migration of vendor plugins e.g Neutron
- Includes upgrade support (from baremetal)
  - Most state e.g DB is left on the host filesystem
- Reuses the same “composable services” architecture
  - Some new interfaces specific to containers added
- HA support (Pacemaker managed container bundles)
- Host networking used to retain network isolation
- Backwards compatibility (Heat parameters & puppet)

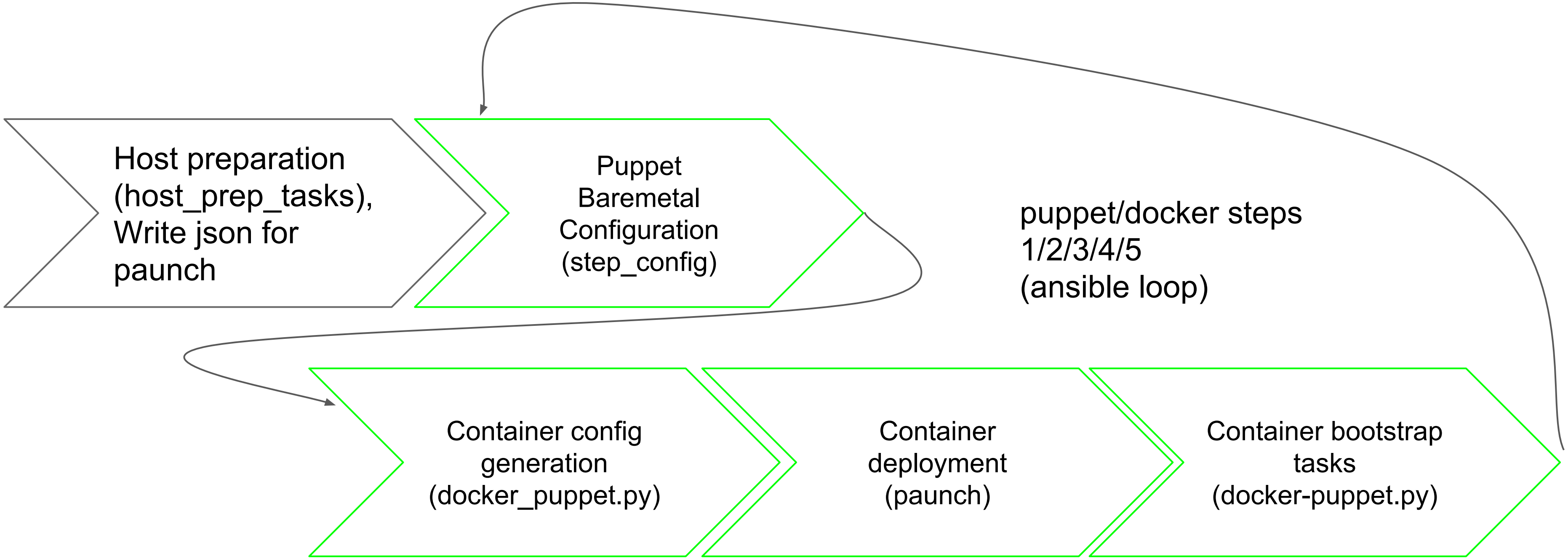
How does it all come together?

Sort of like this ...



Deployment

Ansible + puppet + containers deploy workflow



Host preparation (host_prep_tasks), Write json for paunch

Puppet Baremetal Configuration (step_config)

puppet/docker steps 1/2/3/4/5 (ansible loop)

Container config generation (docker_puppet.py)

Container deployment (paunch)

Container bootstrap tasks (docker-puppet.py)

How does it all come together?

Multi step deployments

- Baremetal configuration (still via puppet)
- Tool to generate config via puppet (docker-puppet.py).
- Ansible playbook runs paunch via command module
- Paunch takes that config and starts up the containers
- Docker-puppet.py runs again to perform bootstrapping via puppet
- ansible deploy tasks applied with an incrementing step

```
 - name: Start containers for step {{step}}
   command: >-
    paunch --debug apply
    --file
/var/lib/tripleo-config/hashed-docker-container-startup-config-step_{{step}}.json
    --config-id tripleo_step{{step}} --managed-by tripleo-{{role_name}}
```

Paunch 101

Container orchestration

Paunch is a ...

Utility to launch and manage containers using the configuration data found in the TripleO service template. (It’s an orchestrator of sorts)

It’s a **wrapper** for the docker-cli.

Supports standalone use outside of heat/TripleO (derived from the heat docker-cmd hook initially).

Paunch 101

http://git.openstack.org/cgit/openstack/paunch/

Single host only, operations are performed via the docker client on the currently configured docker service.

Zero external state, only labels on running containers are used when determining which containers an operation will perform on.

Single threaded and blocking, containers which are not configured to detach will halt further configuration until they exit (config generation & bootstrap containers)

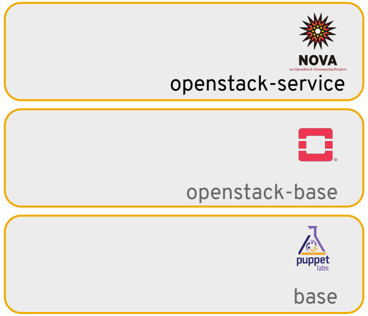
Co-exists with other container configuration tools - only containers tagged as being managed by paunch are modified

Idempotent - leave containers running when their config has not changed, but replace containers which have modified config (minimise minor update downtime)

How does it all come together?

Container Images

KOLLA
an OpenStack Community Project



openstack-service

openstack-base

base

TripleO uses Kolla containers recipes to build OCI-compliant container images - some additional packages are added to the default kolla build to support TripleO.

How does it all come together?

Container image management using openstack CLI plugins

Create a yaml file to set the image, location, namespace, etc:

```
$ openstack overcloud container image prepare
```

Consume the image file generated by prepare:

```
$ openstack overcloud container image upload
```

These steps generate an environment file that is then consumed by heat on deployment/update.

```
$ openstack overcloud deploy --templates $templates -e $templates/environments/docker.yaml
```

This command is used to deploy a (minimal) container enabled overcloud - only one additional environment file is needed to enable containerized deployment.

msc
MEDU 157745 3
22G1
msc
GLDU 504796 9
22G1
DONG FANG
DFSU 249654 7
22G1
XINES
XINU 128736 5
22G1
TCKU 216760 0
22G1
TAL
TCLU 298387 4
22G1
msc
MEDU 370814
22G1
CXDU 197143 9
22G1
GOLD
GLDU 568655 3
22G1
TAL
TCLU 303050 0
22G1
msc
MEDU 274487 4
22G1
msc
MEDU 140986 6
22G1
BEACON
BEAU 278029 0
22G1
msc
MEDU 675813
22G1
tex
TEMU 005573 4
22G1
DFSU 275472 3
22G1
XINES
XINU 109827 4
22G1
www.xines.com
Blue Sky
BSIU 274313 1
22G1
CRONOS
CRXU 339403 5
22G1
FLORENS
FSCU 738971 6
22G1
msc
MSCU 314442
22G1
msc
MEDU 673987 6
22G1
msc
MEDU 229987 1
22G1
msc
MEDU 146848 9
22G1
DONG FANG
DFSU 210225 8
22G1
msc
MSCU 625899 8
22G1
msc
MEDU 120927 7
22G1
msc
MEDU 21782
22G1
Matson
LEASING COMPANY
MLCU 249978 3
2210
IPXU 214850 6
22G1
msc
MEDU 163664 3
22G1
FLORENS
FCIU 604407 4
22G1
www.florens.com
GOLD
GLDU 526440 8
22G1
msc
MEDU 292017 1
22G1
msc
GLDU 394

Other talks this week...

- TripleO Project Update - Monday 3.10-3.50pm C3.2
- TripleO Project onboarding - Monday 4.20 - 5.00pm C4.7
- Putting OpenStack on Kubernetes - Monday 4.20 - 5.00pm Pyrmont Theatre
- Automated NFV deployment with TripleO - Tuesday 09.00 - 09.40am C3.3
- OpenStack in containers (lightning talk) - Wednesday 2.40-2.50pm - Parkside 2 foyer

Links & where to find out more

- IRC (Freenode) #tripleo
- openstack-dev mailing list [tripleo]
- https://docs.openstack.org/developer/tripleo-docs/
- https://docs.openstack.org/tripleo-quickstart/
- http://tripleo.org/planet.html (has links to blogs)
- https://hardysteven.blogspot.co.uk/
- https://etherpad.openstack.org/p/tripleo-deep-dive-topics
    - Has links to previous talks, we can do more if you request specific topics!