

# Deploying OpenStack with TripleO

OpenInfra Days Nordics

October 2<sup>nd</sup> 2019

# Hej!

- Name: Gauvain Pocentek
- Job: Infrastructure engineer @ Kindred
- Been deploying, running, and using OpenStack since 2013
- Started playing with TripleO 2 years ago

# TripleO overview

# TripleO - Overview

- Official deployment tool for OpenStack
- Base for RedHat OpenStack Director
- Originally called "OpenStack On OpenStack"
  - Uses a single-host OpenStack deployment (**undercloud**) to deploy a production OpenStack (**overcloud**)
- Deploys everything:
  - Baremetal nodes
  - OpenStack
  - Ceph (optional)
  - High availability features
- Also manages the life cycle: minor and major upgrades, scaling



# TripleO – deployment steps

- Plan – some settings are very hard to change after the initial deployment
- Install the `tripleo` client on CentOS/RedHat
- Configure and deploy the undercloud
- Register the physical nodes for the overcloud
- Configure the overcloud
- Run the overcloud deployment



# TripleO – planning

- Decide on features to enable:
  - TLS? Yes! So you need DNS and certificates
  - Instance HA?
  - Telemetry?
- Decide on OpenStack components to deploy
- Architecture:
  - How to split services on hosts (performance, availability)
  - Storage has a strong impact on availability of VMs (pets vs cattle)
  - Classic setup:
    - 3 controllers
    - Optionally network nodes
    - Compute farm, possibly with different hardware types
- Plan the network setup carefully, it is very hard to change things once you are in production



# OpenStack for the undercloud- the tools

- Neutron as IPAM
- Nova/Ironic to deploy the baremetal machines
  - iPXE network
  - IPMI to manage power state
- TripleO client, which relies on
  - Mistral for managing workflows
  - Heat as entry point to deploy everything else

# Overcloud deployment – the steps

- The tripleo CLI generates heat templates
- Heat creates a stack from these templates:
  - Initiates the baremetal deployment through nova/ironic
  - Generates puppet configuration and ansible playbooks for application deployment
- The tripleo CLI downloads the playbooks from heat
- Then it runs the playbooks



Workshop environment

# Workshop environment - Overview

- Fully virtualized: OpenStack on OpenStack... on OpenStack
- 3 nodes: 1 undercloud, 1 controller, 1 compute node
- Overcloud networks:
  - Deployment network (eth1 on the undercloud, eth0 on the overcloud)
  - Application networks using a single interface and VLANs (eth2 on the undercloud, eth1 on the overcloud)
- Admin network for SSH and IPMI on the undercloud (eth0)
- VirtualBMC on the admin network to provide IPMI for VMs

# Workshop environment – Time and resource constraints

- Undercloud installation is very slow in our setup (1h30)
  - Already configured and installed
  - We will have a look at the configuration file
- Only 2 nodes for Overcloud:
  - No HA setup
  - Limited set of components enabled

# Let's get started!

- Open your web browser and connect to the Etherpad
  - <https://etherpad.openstack.org/p/oidn2019-tripleo>
- Pick a VM, put your (nick)name next to the IP
- To access the VM:
  - `ssh centos@IP`
  - Password: TripleOpenInfra
- Get the workshop resources:
  - `git clone https://github.com/gpocentek/openinfra-days-nordics-2019.git`
- `docs/start.md`

Undercloud deployment

# Undercloud requirements

- Minimal installation of CentOS 7 or RHEL 7
- 2 networks:
  - Admin network (SSH access)
  - Deployment network for PXE boot and access to resources from the overcloud nodes
- A bit of preparation:
  - Creation of a stack user
  - Configuration of OpenStack and TripleO repositories
  - Installation of the tripleo client (python-tripleoclient rpm)
- Unless you mirror everything Internet access is required
- [https://docs.openstack.org/project-deploy-guide/tripleo-docs/latest/deployment/install\\_undercloud.html](https://docs.openstack.org/project-deploy-guide/tripleo-docs/latest/deployment/install_undercloud.html)

# Configuration file and installation

- Everything must be done as the stack user
- A sample configuration file is available in `/usr/share/python-tripleoclient/undercloud.conf.sample`
- Let's go through the important settings:
  - `docs/uc-settings.md`
- To deploy the undercloud environment: `openstack undercloud install`
- After deployment you can start exploring your undercloud:
  - `docs/uc-explore.md`

# Post-installation steps

- Resources must be created in the UC to prepare the OC deployment:
  - Glance images:
    - To introspect the machines
    - To deploy an operating system
  - Flavors to match physical hosts to OpenSack roles
  - Inventory of physical machines
- Once these resources have been defined nodes can be introspected
- Let's do it:
  - docs/uc-post-install.md



Overcloud configuration

# Overview

- The OC configuration is:
  - The definition of server roles: what should be installed and where
  - The network information: available networks, and how they relate to hosts and services
  - A set of options for the Heat templates: how services should be configured
- Get ready:
  - Documentation: <https://docs.openstack.org/project-deploy-guide/tripleo-docs/latest/features/index.html>
  - Docs might not enough, but going through the heat templates could help you: <https://github.com/openstack/tripleo-heat-templates>
  - Still missing something? Puppet to the rescue: <https://github.com/openstack?q=puppet>

# About Heat templates

- Heat templates describe how resources should be created in OpenStack
- Written in yaml:
  - Files must have a **.yaml extension**, not ~~.yml~~
- They provide:
  - A list of **resources**
  - **Parameters**
  - Outputs
- Heat has a set of supported resources, but you can create new resources from templates; these new resources can be named in a **resource registry**.
- What about an example?
  - heat-example/\*

# Custom roles

- Roles let you define the list of services to install on each type of machines
- The default roles provided by TripleO cover multiple use cases
- But you can create your own roles
- For this workshop
  - We will use the existing Controller and Compute roles
  - But we will make them lighter
- Roles also define how the network should be setup on the associated hosts
- Let's create our `roles_data.yaml` configuration file:
  - docs/oc-roles.md

# Network data

- Network configuration is easy if you use the default TripleO setup:
  - But you need to configure the physical network according to what TripleO expects
  - You still need to define VLAN IDs and CIDRs
- You can also adapt the network deployment in TripleO to what you need/what
- In that case you need to define:
  - The global network parameters (CIDR, VLANs, routes, gateways, MTUs, and so on)
  - The network configuration for each type of host
  - How services will use the networks
- Let's define our networks:
  - docs/oc-network.md

# Overcloud configuration

- A set of parameters must be defined in a Heat-compatible yaml file
- There are dozens of parameters
  - Can be a challenge to find what you need
  - Some parameters cannot be defined as Heat parameters
    - They need to be defined as hieradata (puppet configuration)
- Once you are ready, you can start the deployment:
  - `openstack overcloud deploy [a_bunch_of_additional_parameters_here]`
  - Everytime you make a change you need to run the command with the same arguments
  - It's a good idea to create a wrapper script to avoid mistakes
- Let's configure our overcloud:
  - `docs/oc-config.md`

# Overcloud deployment

# Overcloud deployment

- We can deploy everything at once, but we can also split the deployment:
  - Deploy the baremetal machines
  - Generate the ansible playbook to deploy OpenStack
  - Run the playbook step by step
- docs/oc-deploy.md



Questions?