# MODULE 3

# Working with OpenStack

# Module 4b Objectives

By the end of this module, you should be able to explain:

- What you can use the TWC Jira Help Desk for
- How to deploy a VM using the UI (Horizon)
- How to deploy a VM using the OpenStack CLI tools
- Basic IaaS VM deployment operations
  - Adding a cloud-init initialization file for cloud images
  - Attaching to Network
  - Booting from a Volume
- Using the CLI to manage an application across multiple regions

OneCloud Consulting

# TWC OpenStack Portal Onboarding

OpenStack Cloud Service Desk provides a single point of entry for potential and existing customers to get access and help to the TWC OpenStack Cloud.

Getting access to the TWC OpenStack Cloud:

URL: https://cloud.twc.net/jira/servicedesk/customer/portal/25

OneCloud
Consulting

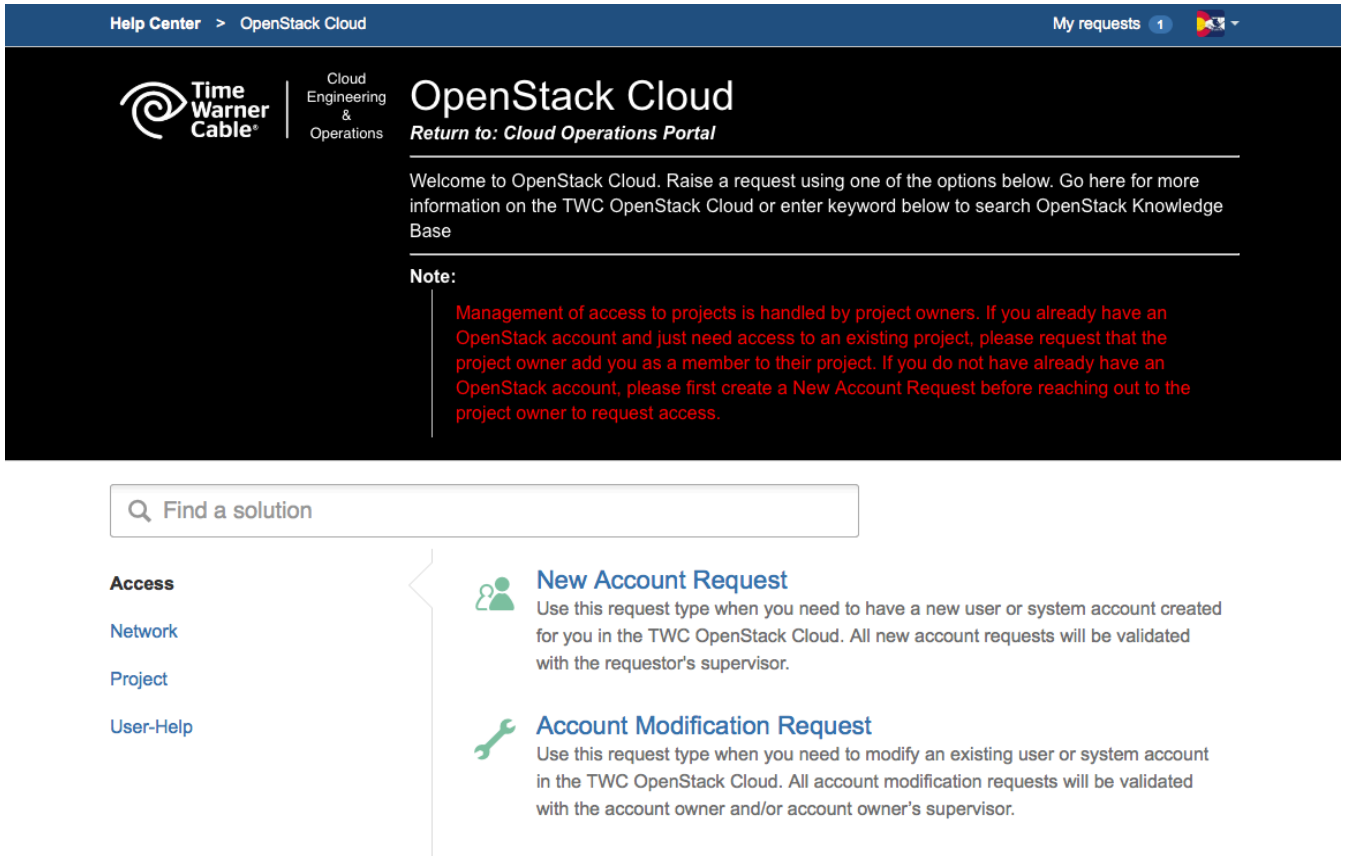
# Service Desk Capabilities and Services

Use the OpenStack Cloud Service Desk for:

- Access
  - New Account Request
  - Account Modification Request
- Network
  - DNS Requests
- Project
  - New project Request
  - Project Modification Request
- User-Help
  - OpenStack Help
  - Report a Bug
  - New Feature Request

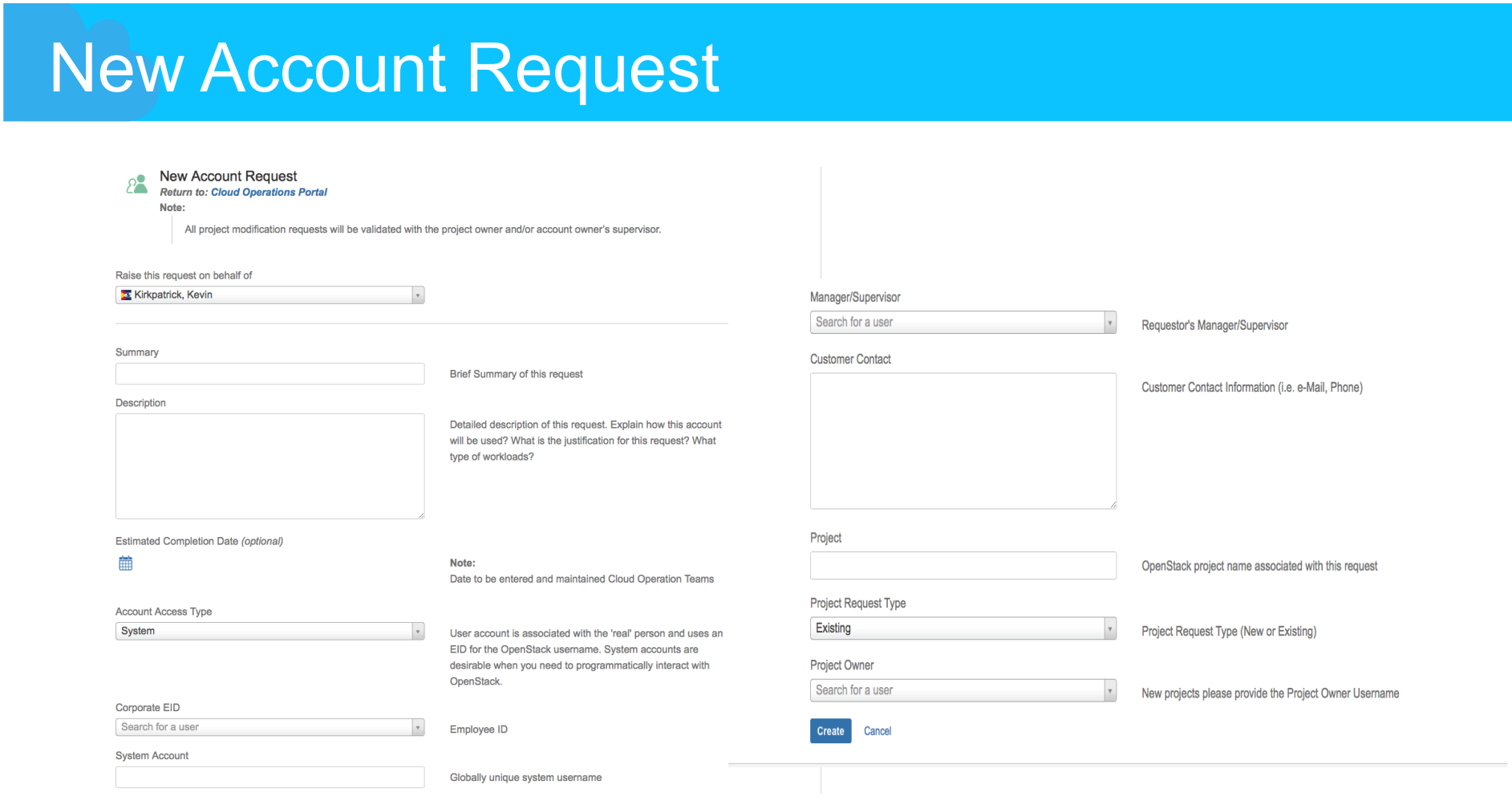
OneCloud
Consulting

# Access

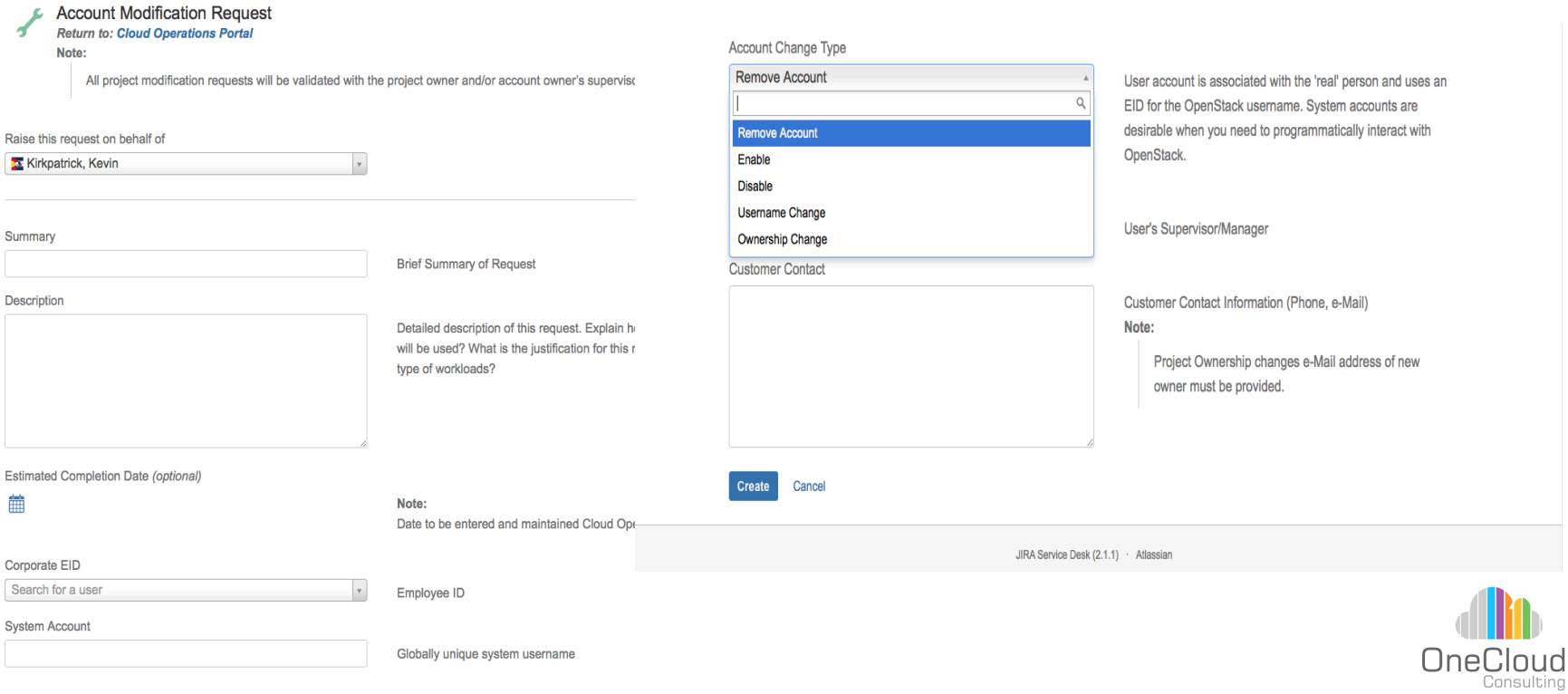



OneCloud Consulting

# New Account Request

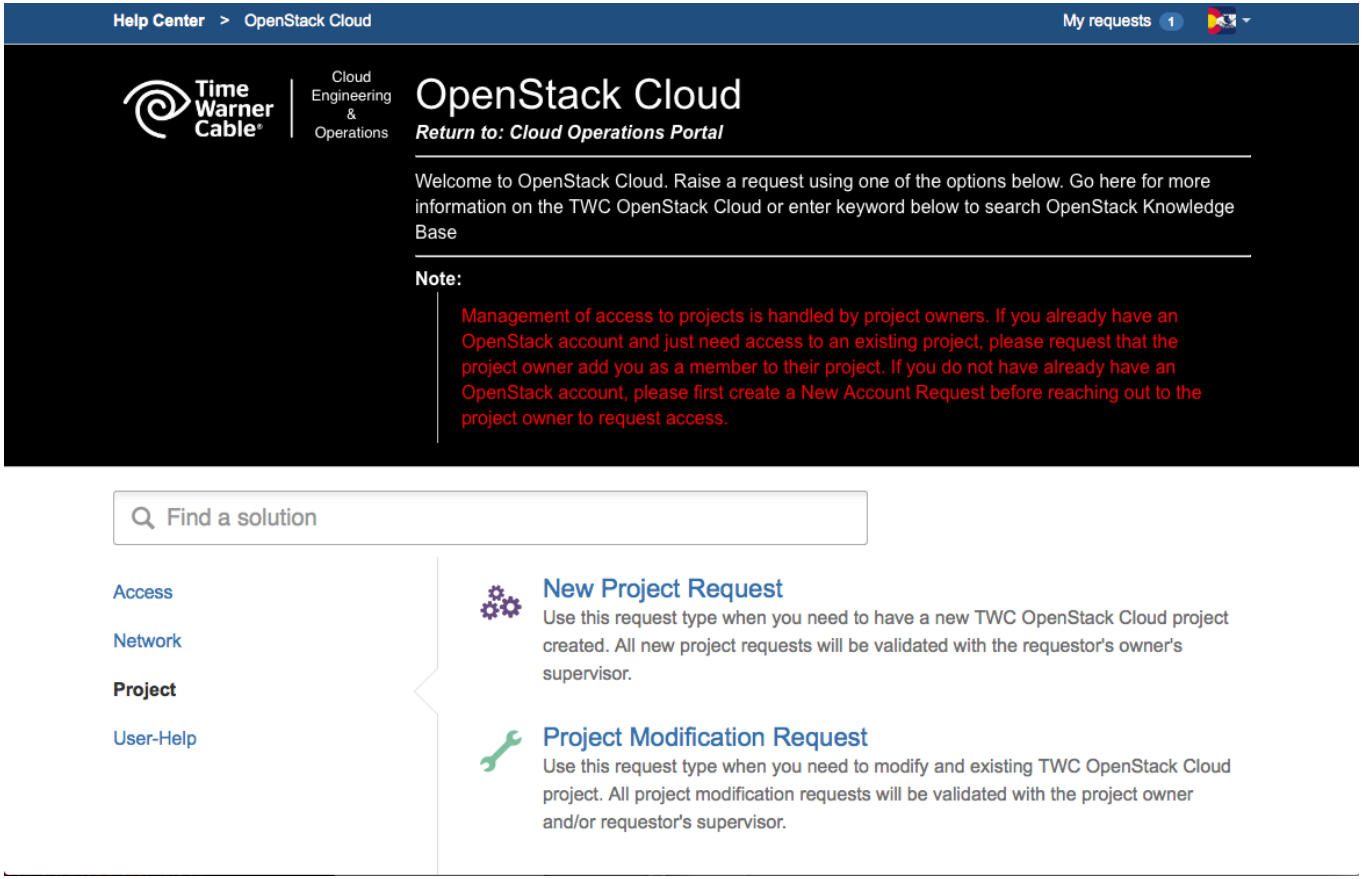


# Account Modification Request



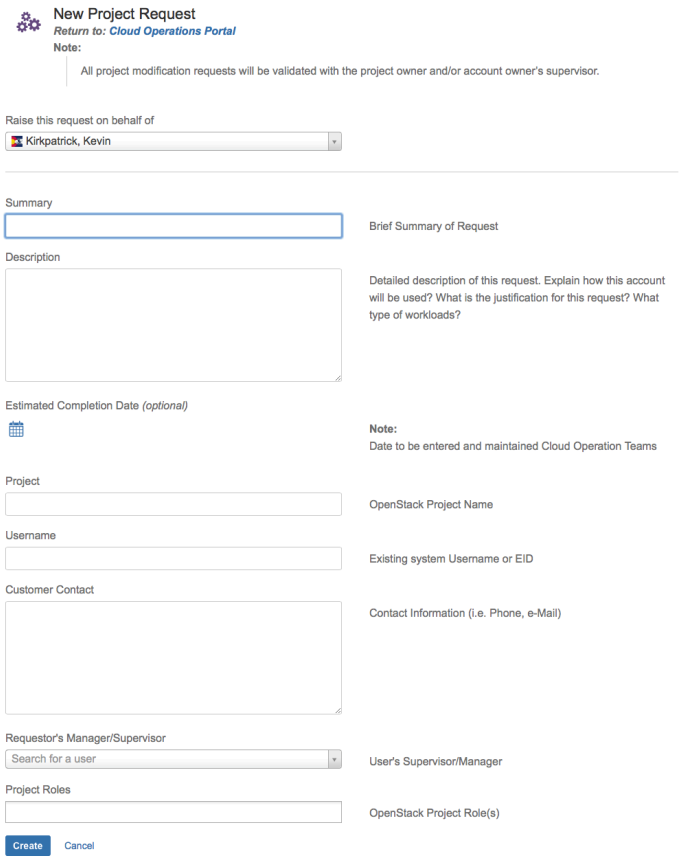

OneCloud Consulting

# Project



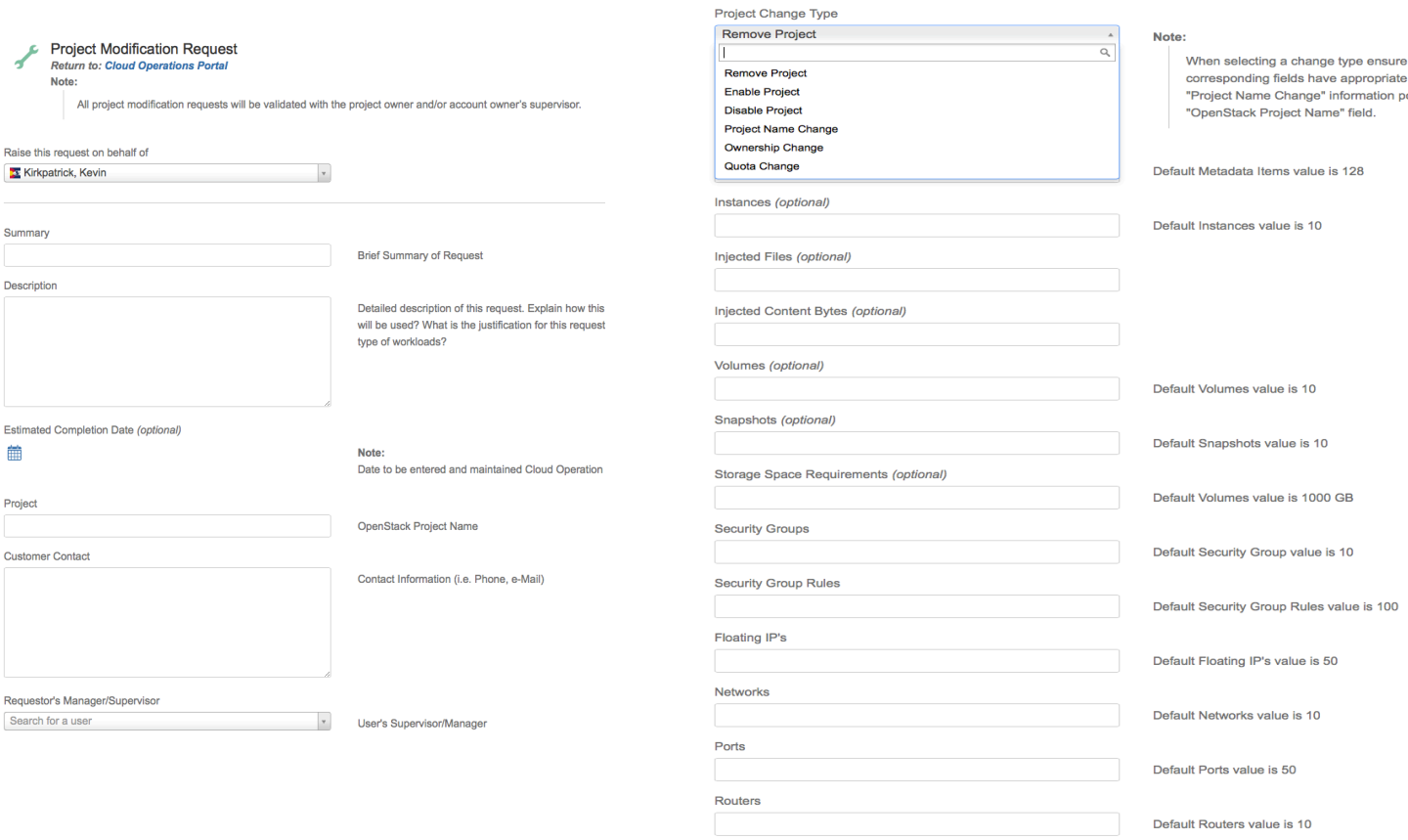

OneCloud Consulting

# New Project Request

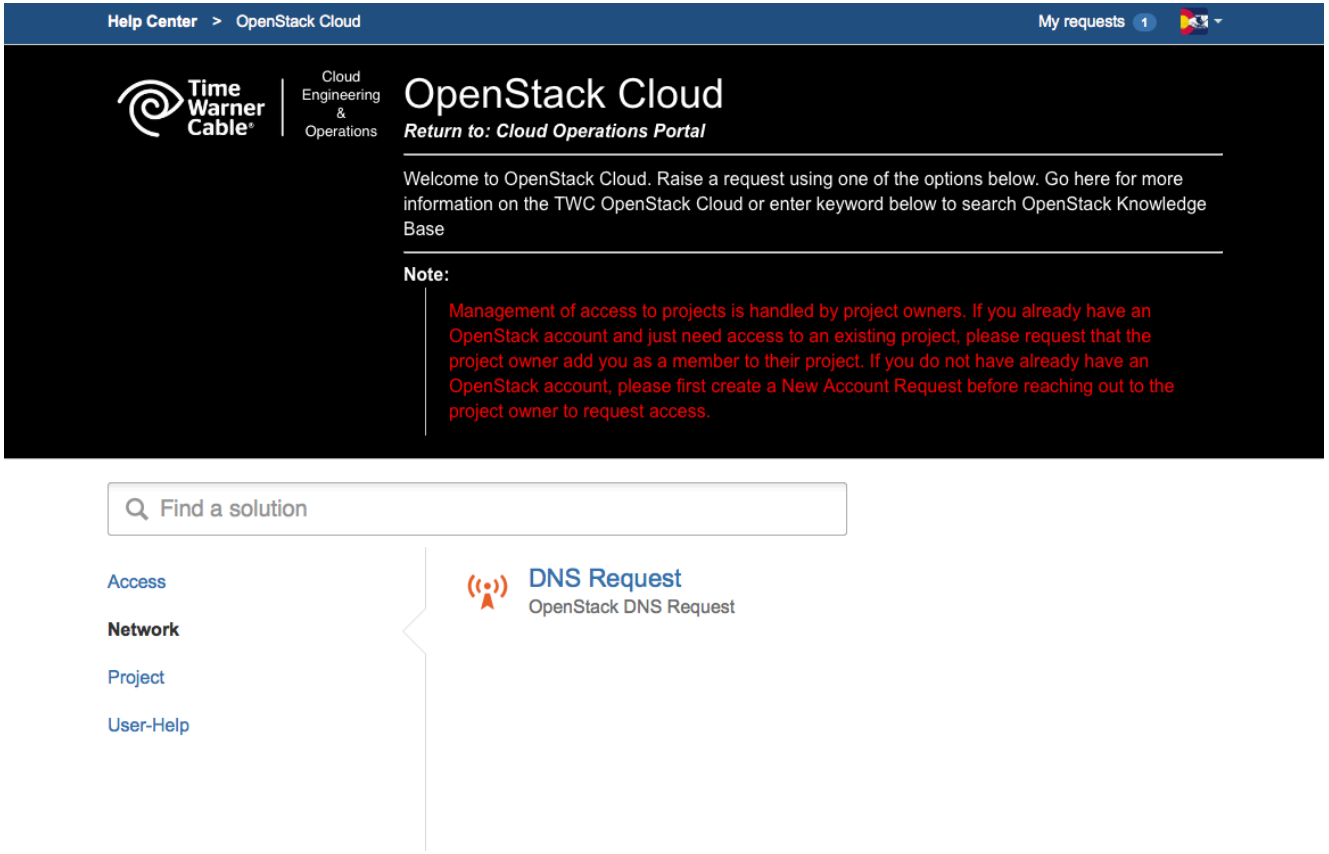



OneCloud Consulting

# Project Modification Request

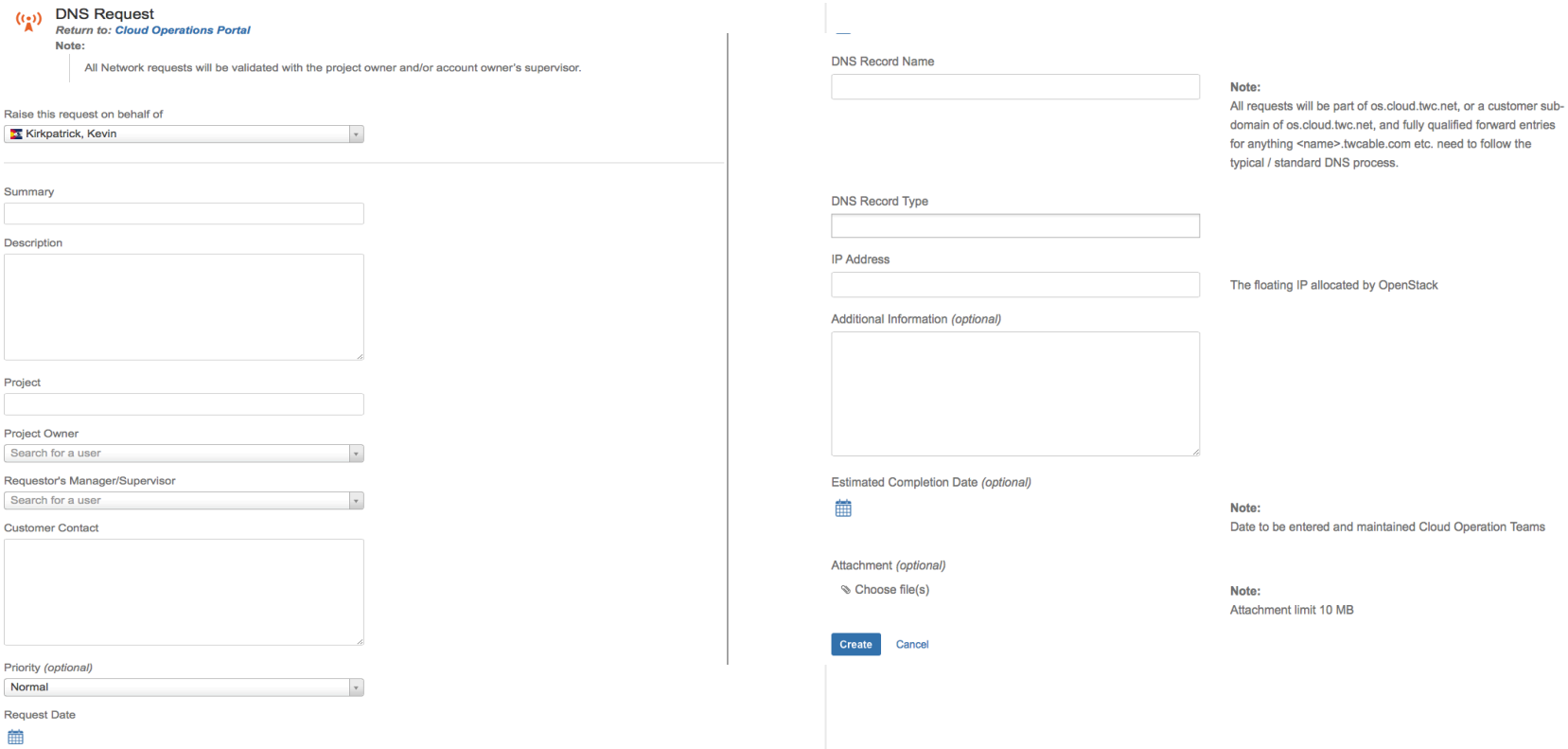


# Network



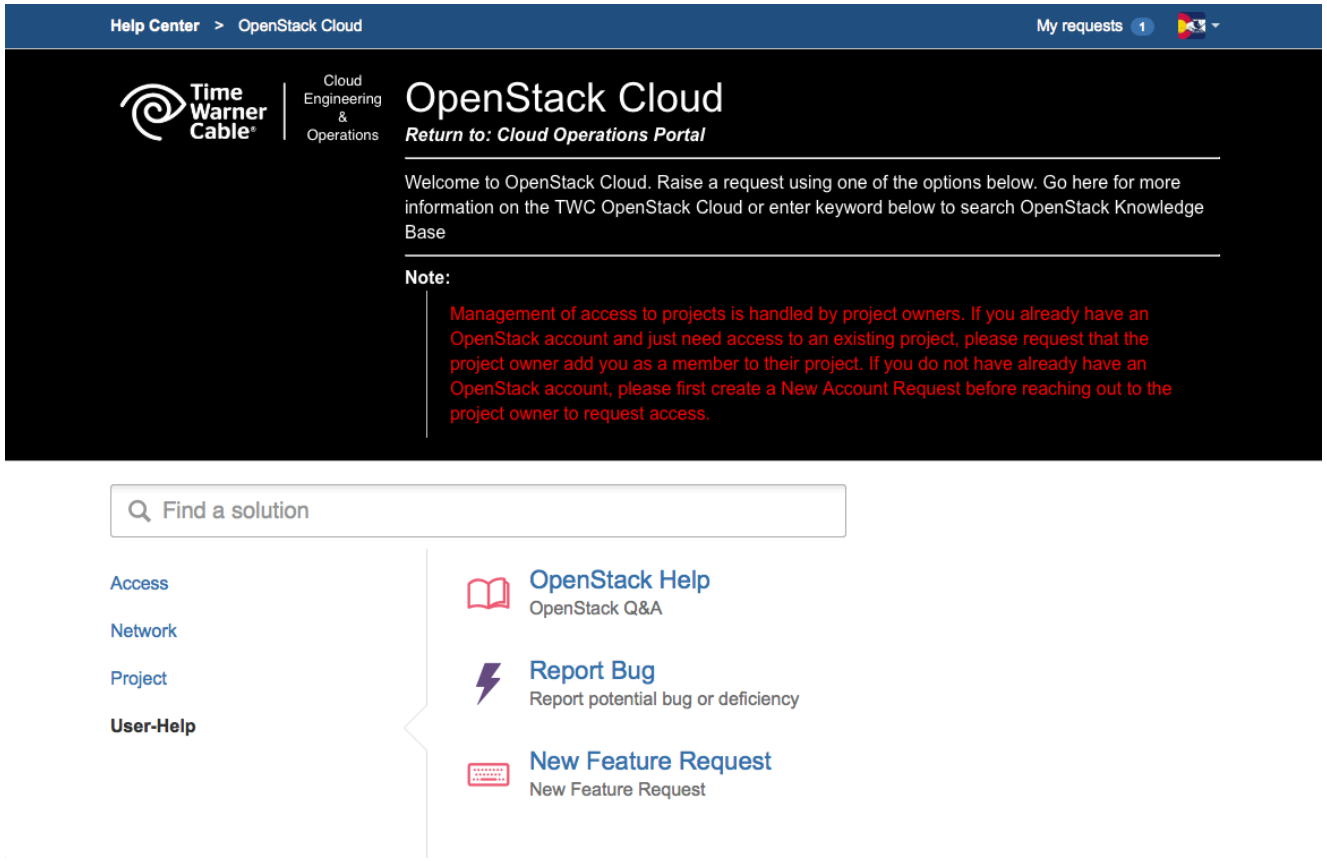

OneCloud Consulting

# DNS Request



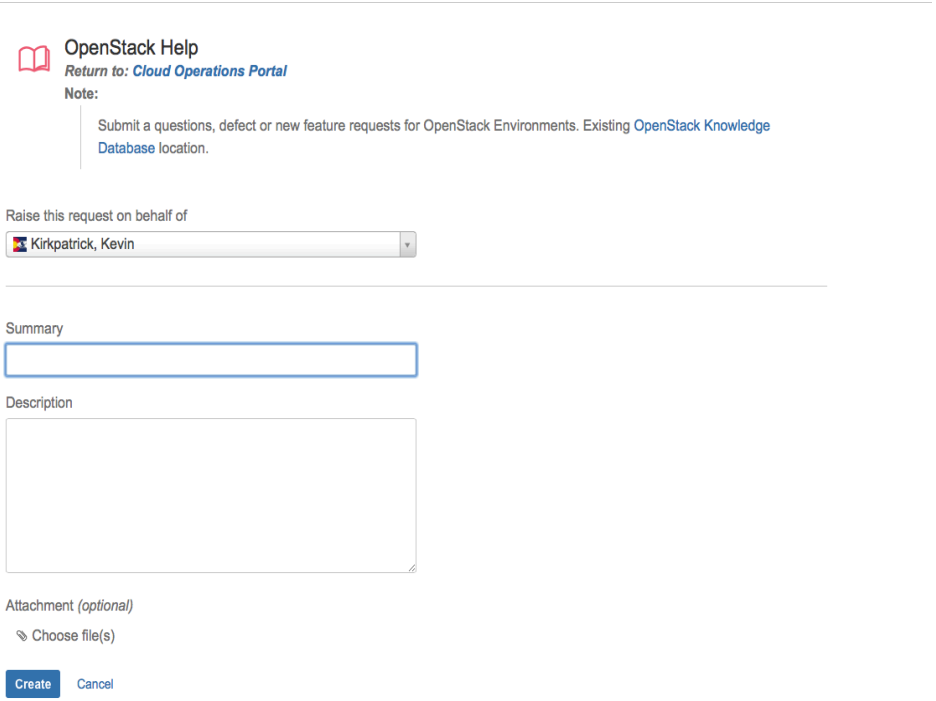

Will be replaced by a OpenStack DNSaaS function in 2015

# User-help



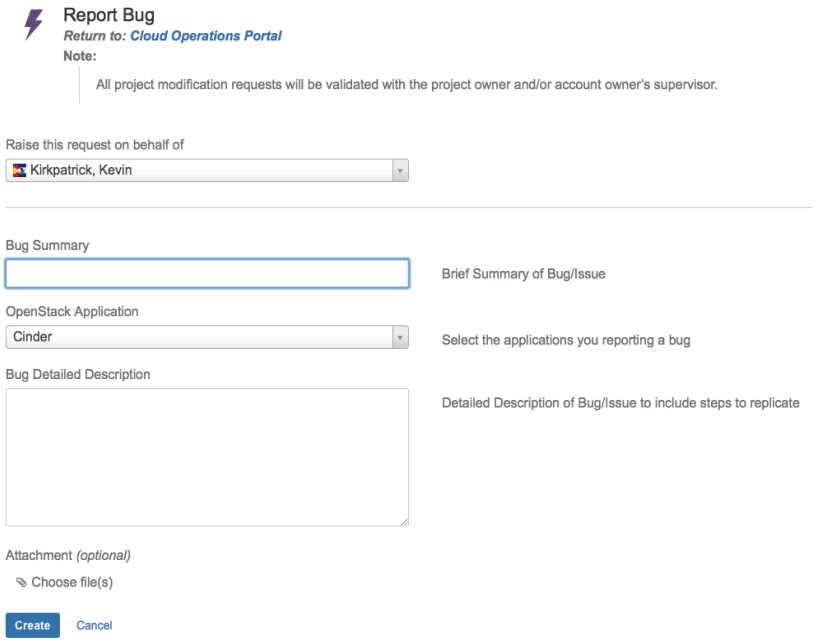

OneCloud Consulting

# OpenStack Help



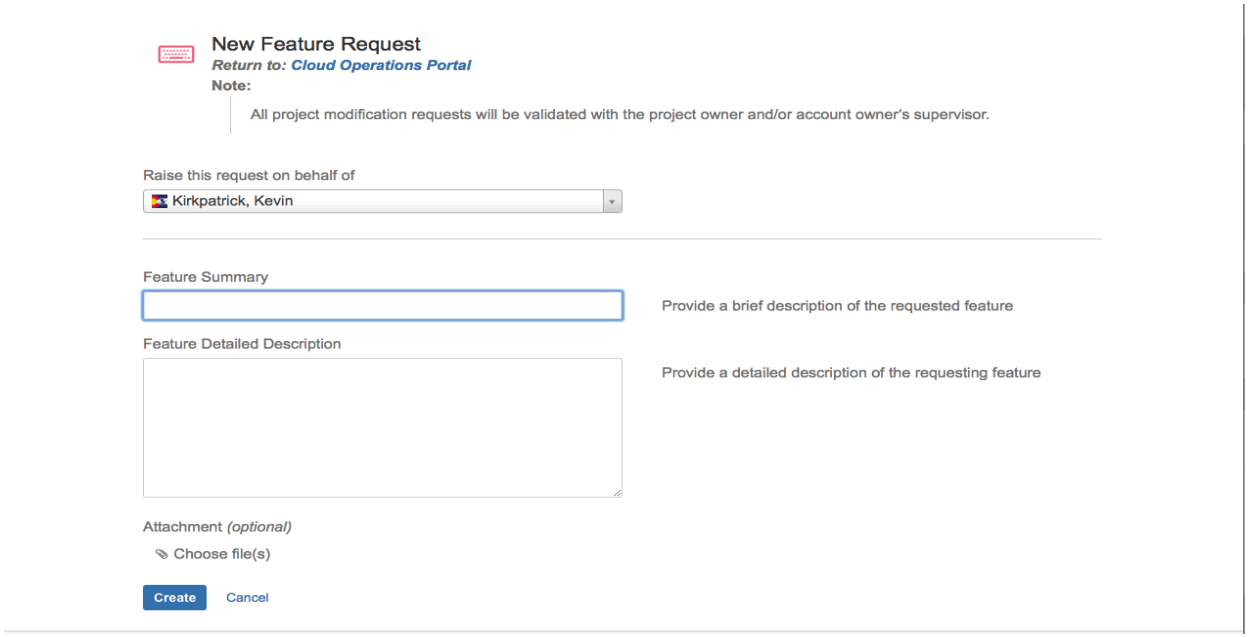

OneCloud Consulting

# Report a Bug




OneCloud Consulting

# New Feature Request




OneCloud Consulting

# MODULE 4

## Deploying Virtual Machines in OpenStack

# Setting up CLI Tools

- There are multiple ways of interacting with OpenStack, but one of the most straight forward is to use the CLI tools.
  - Supported on any platform that supports Python 2.7 (no 3.0 support yet):
    - Linux
    - Windows
    - OS-X
  - Installation is simplified with “pip” but easy_install can also be used.
- Microsoft use case is slightly more difficult because the environment is set via system level resources, but otherwise works fine
- CLI tools can be used either directly or in a script, which can dramatically simplify re-use, especially for parametric scripts.
- http://docs.openstack.org/user-guide/content/install_clients.html

OneCloud
Consulting

# BASIC VM DEPLOYMENT

## Understanding Nova

# Nova - OpenStack Compute

- OpenStack Compute (Nova) is a  cloud computing fabric controller (orchestrator)
- Provides a highly scalable management framework for virtual machines
- The original OpenStack orchestrator of orchestrators: Interacts with Neutron, Cinder, Glance to create and provision a VM
- Called by other services to implement compute aspects (Heat, Trove, Sahara)

# Key Features

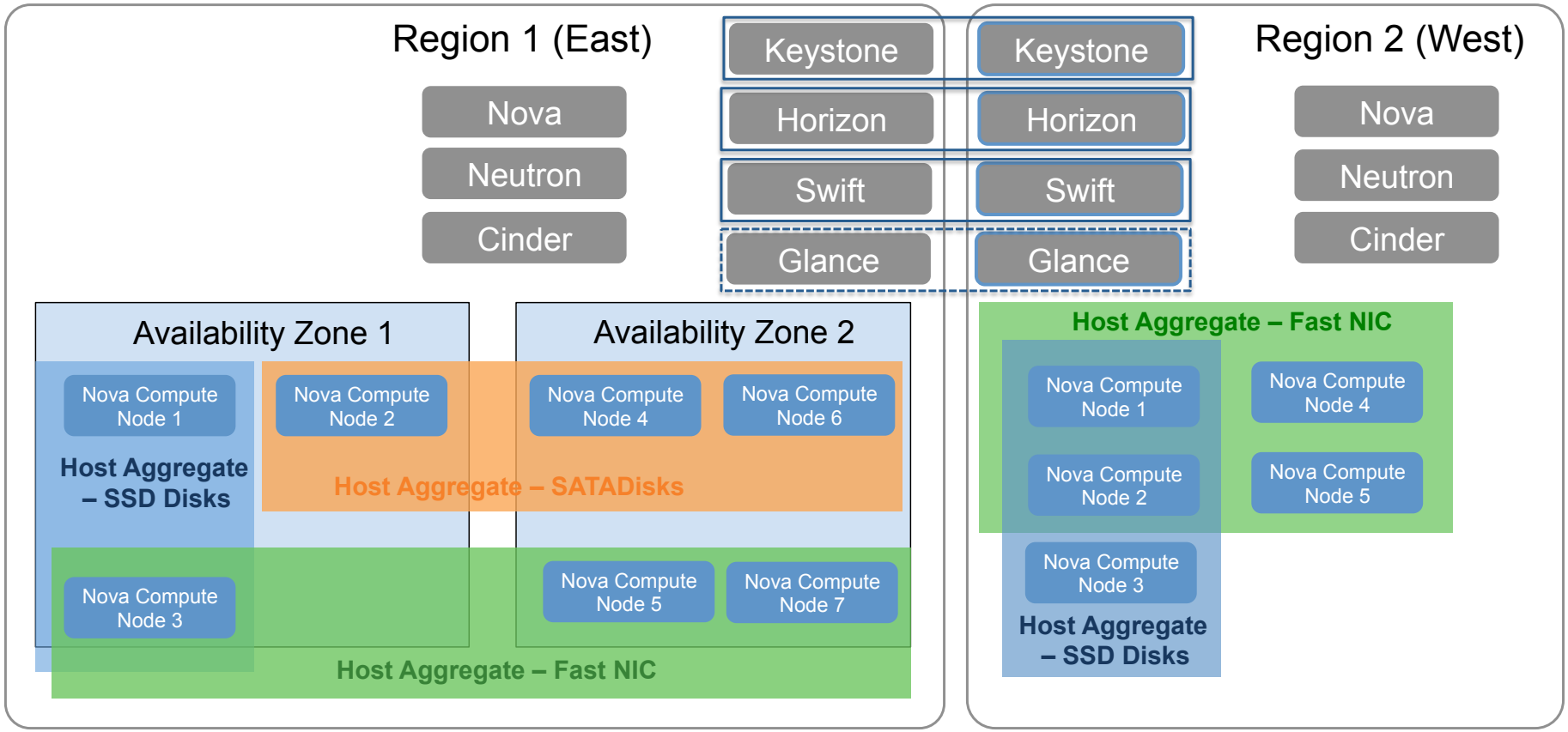
- Compute Entity management
  - Create/Delete operations
  - Power management (shutdown, power on, sleep)
  - Root Disk management (snapshots with Glance)
  - Migration (note: not available on all deployments)
  - Network attachment
  - Network edge security via ACL
  - Storage attachment (root or additional volume data)
- Scheduling Capabilities
  - Scheduling as in resource targeting
  - Flavor/”type” of system
  - Tag mechanism for affinity/anti-affinity
- VM access
  - Log access
  - “KVM” keyboard/video/mouse console access
  - Serial interface access (not available on all systems/target compute)

OneCloud Consulting

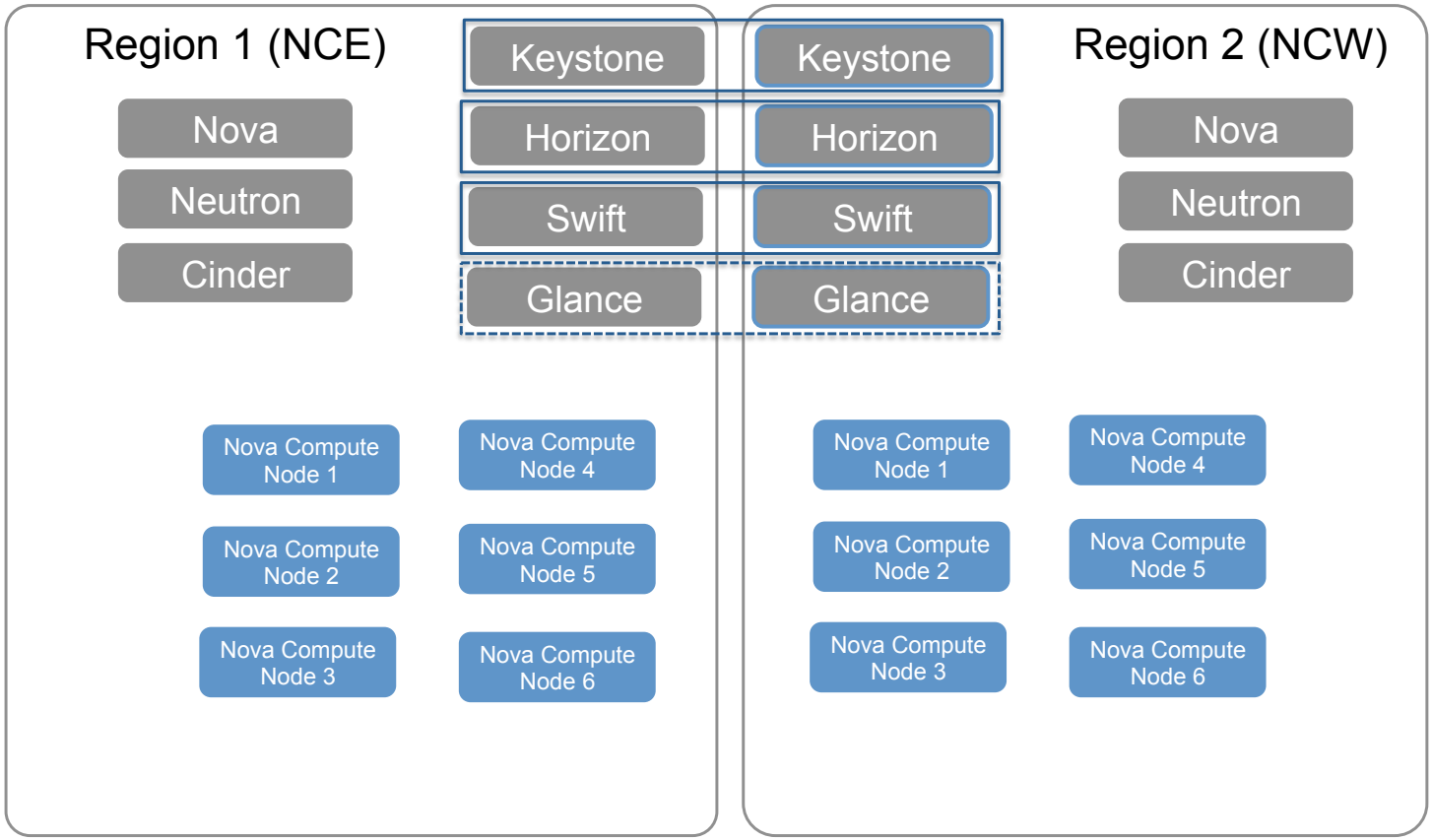
# Nova Concept - Flavors

- A flavor is an available hardware configuration for a server.
- Flavors define the configuration of the instance, including:
  - Number of Virtual CPUs
  - Amount of RAM
  - Size of ephemeral disks
- OpenStack provides number of predefined flavors which cloud administrators may edit or add to
- Users **must** pass a flavor from the set of available flavors defined
- Flavors are how a host aggregate are presented to a user

# OpenStack Segregation Methods (Generic)



# OpenStack Segregation Methods (TWC)



# Nova-Scheduler

- The Filter Scheduler supports filtering and weighting to make informed decisions on where a new instance should be created base on infrastructure status at time request
- All compute hosts periodically publish their status, available resources and hardware capabilities to nova-scheduler through the queue
- Physical placement of VM in the environment influenced  or dictated based on required flavor, segregation methods, host capabilities and capacity, and other environmental filters.
- E.g. affinity or anti-affinity can assure close physical placement of VMs (reduce communication's latency)  in a datacenter, or anti-affinity filters can disperse VMs (e.g. provide backup functionality)

OneCloud
Consulting

# Nova Affinity/Anti-affinity

- Create a group per “domain” e.g. per servers to be deployed that need either affinity or anti-affinity:
  - nova server-group-create anti-group anti-affinity{/affinity}
- To use:
  - nova boot --flavor standard.small --image Ubuntu-Server-14.04 --hint group=anti-group --key-name default node
- Affinity is only available via CLI/API calls (e.g. HEAT)

OneCloud
Consulting

# Nova-Compute - Storage Management

| Ephemeral Storage | Persistent Volume Storage |
| --- | --- |
| • Exists only for the life of an instance.<br>• Persistent across reboots of the guest Operating System.<br>• Will be deleted when instance is deleted. | • Volumes are persistent virtualized block.<br>• Independent from instance.<br>• Volume can be attached to a single instance at a time.<br>• May be detached or reattached to a different instance while retaining all data, much like an USB drive. |

# Provisioning Instances - CLI Ephemeral Disk

**Command to boot your instance from CLI**

```
nova boot --image imageid --flavor flavorid --key-name name --
security-groups name --nic net-id=networkid instancename
```

- **Image id:** list of images available in the image repository (nova image-list or glance image-list)
- **Flavor id:** list of flavors available (nova flavor-list)
- **Key-name:** key pair name from the list of key pairs (nova key pair-list)
- **Security-groups:** security group name with appropriate security group rules assigned. (Nova secgroup-list.)
- **Net-id:** network id from the list of networks. (Neutron net-list)

# Nova Boot From Volume

- Why?
  - Persistent storage (delete the VM, keep the Volume)
  - Performance?
  - IT managed (e.g. storage and storage replication)

- Data needed:
  - Image source type and id (or name)
  - Destination type (volume or local disk)
  - Size (disk in GB)
  - Shutdown goal (preserve/delete)

```
nova boot --flavor standard.small --block-device source=image,id=Ubuntu-
Server-14.04,dest=volume,size=10,shutdown=preserve,bootindex=0 \
    myInstanceFromVolume
```

OneCloud Consulting

# Nova - Meta-data / Cloud-Init

- How do you configure a virtual machine in a cloud after it has been booted from a ‘generic’ image?
- Web service, managed by Nova with nova-network, managed by Neutron with Neutron based network service
- Can pass: user scripts, ssh keys, configuration data (YAML formatted data)
- Requires that ‘generic’ images include the cloud-init package
- Alternate model exists as well: “config-drive” (Cloud-Init)
- Same capability for configuration, and same OS package, but provides an auto-generated ISO that is attached to the instance at boot time

OneCloud Consulting

# Provisioning Instances - Continued

Inject the cloud-init scripts to the instance

- Userdata flag is used to customize the instance provisioning via cloud-init package. Provide the location of the script to –user-data while provisioning
- `nova boot --user-data ./user-data.sh --image imageid --flavor flavorid --key-name name --security-groups name --nic net-id=networkid instancename`
- user-data.sh contains a script, e.g. to install puppet, git and python-pip

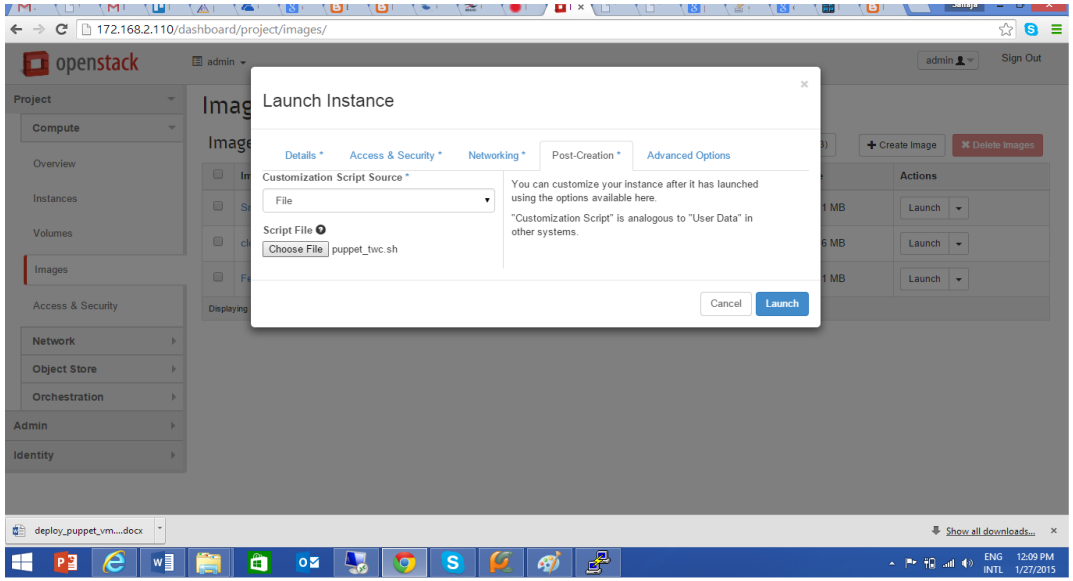
An example user-data.sh script might be:

```
#!/bin/bash
apt-get update -y
apt-get install git
git clone https://github.com/onecloud/todo-app
```

OneCloud Consulting

# App Deployment via Dashboard

We can use Post creation tab and add the user-data as a shell script attachment or copy paste the code to the text area available and launch the instance.




OneCloud Consulting

# Security Groups

- What is it?
  - Mini ACL based firewall mapped to the VM interface port.
  - Implemented as IPTables on a per-VM linuxbridge bridge.
- Can define Source, Dest, type (ICMP, TCP, UDP)
  - Source (for ingress rules) or Dest (for egress rules) can be either CIDR addr or a security group (even the same one)
  - Can define ports or port ranges for each ACL
- Can define and apply security groups per VM, either at time of creation or after VM boot
- Default rule:
  - Ingress: ICMP (ping), ssh (22), RDP (3389) {note: 22 blocked from greater internet}
  - Egress: Open

OneCloud Consulting

# Create Security Groups

- Create a security group called “redis-db”, and allow 6379 TCP

```
nova secgroup-create redis-db "allow ingress Redis"
nova secgroup-add-rule redis-db tcp 6379 6379 0.0.0.0/0
```

- Now apply the “redis-db” rule to your vm(s)

```
nova add-secgroup web-101 redis-db
```

OneCloud Consulting

# Network Service Interaction

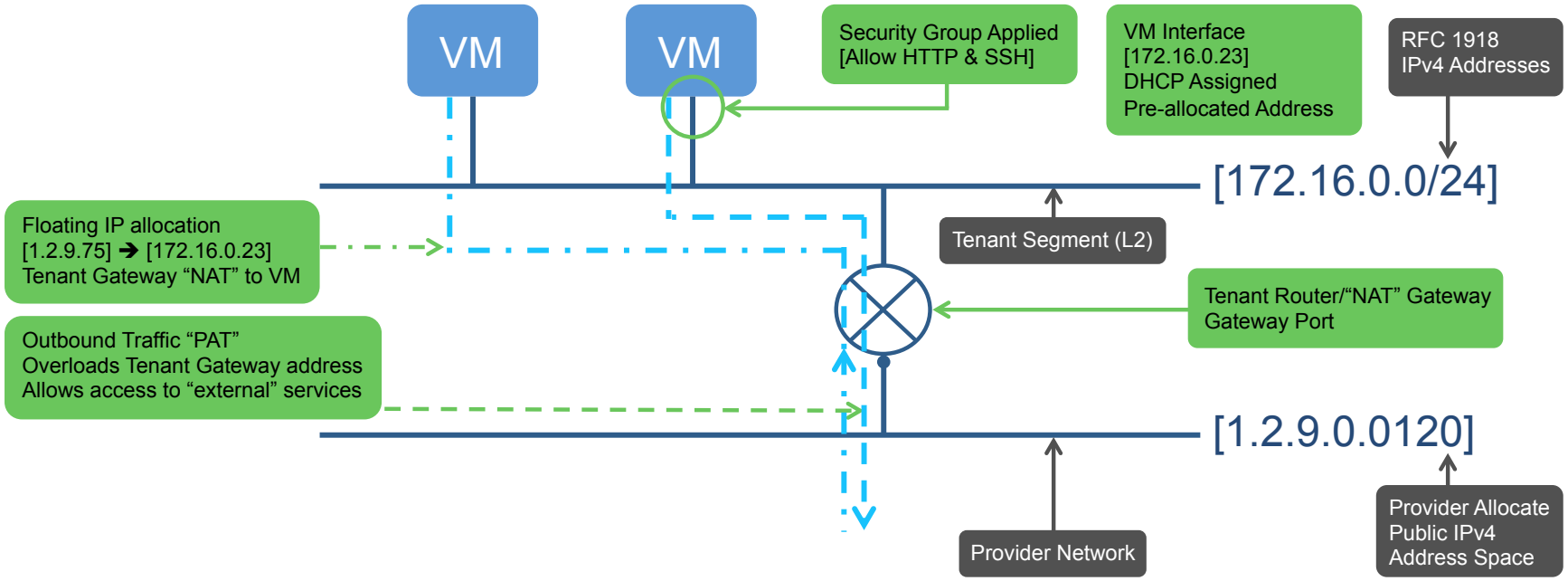
- Neutron network model
- Network as deployed at TWC
- Attaching a port to a VM

- Floating IPs

- Load Balancing - How to interact (external capability via TWC portal)

# Cloud Network Model

## OpenStack Generic Network



# Dealing with multiple possible Networks

- If a tenant has only a single network, it is not necessary to define it to the nova boot command.

- If however, there are more than one networks _visible_ to the tenant (perhaps two private networks), then it is necessary to define the network in the boot command
- This is true in Horizon or the GUI as well

OneCloud
Consulting

# Find a network and associate to a VM

- Find a network:

```
# neutron net-list
+--------------------------------------+----------+-------------------------------------------------------+
| id                                   | name     | subnets                                               |
+--------------------------------------+----------+-------------------------------------------------------+
| e9bee325-a6f4-49b6-a59d-d26709cfc56f | flat     | 45b4192c-6e33-4278-8b2b-c387f5ac4fc1 10.1.10.0/24 |
| 4aa9927b-4a77-43f0-88de-dcb0f1c8d9a9 | private  | 656f5c91-fe81-440c-abcc-2c31b2a78023 10.0.10.0/24 |
+--------------------------------------+----------+-------------------------------------------------------+
```

- Select the network_id from the network of choice and apply to a nic:

```
nova boot --flavor 3 --image trusty --nic net-id=4aa9927b-4a77-43f0-88de-dcb0f1c8d9a9 net_vm
```

OneCloud Consulting

# Find and associate a Floating-IP

- Floating IPs are addresses that are a part of an “external” network that can be associated with a private (tenant-network attached) virtual machine.
- The network function that enables this is call NAT for Network Address Translation
- First find the external network:

```
neutron net-list
```

- If the network isn’t obvious (name includes something like Ext or Public) then you can sort through the list looking for a network with “router:external” set to True:

```
neutron net-show Ext-Net | grep router:external | grep True
```

OneCloud Consulting

# Create and associate the floating IP

- Create a Floating IP

neutron floatingip-create Ext-Net

- Determine the port you want to associate with the floating IP

neutron port-list | grep IP.OF.TENANT.VM

- Then associate them.

neutron floatingip-associate ID_FROM_NEW_FLOATING_IP ID_FROM_PORT

OneCloud
Consulting

# BASIC VM DEPLOYMENT

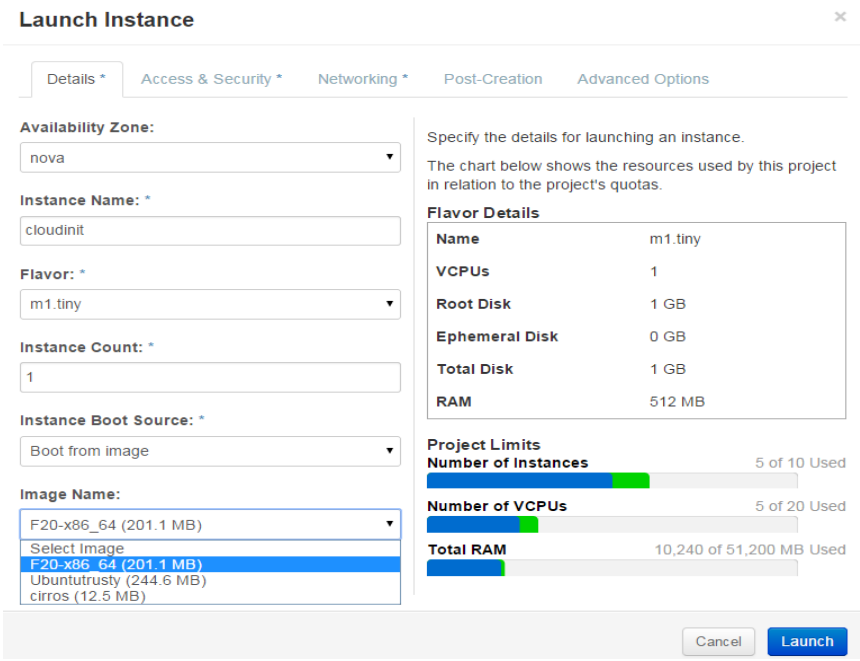
## Create a VM, deploy your app

OneCloud
Consulting

# Compute Deployment - “OpenStack” Methods

- 2 Basic Methods
  - CLI
    - nova boot …
  - Horizon UI

- More integrated methods
  - SDK Based (python leveraging the same capabilities as the CLI tools
    - Fog.io - Ruby
    - Jclouds - Java
  - Environment based
    - Vagrant - Ruby based




OneCloud Consulting

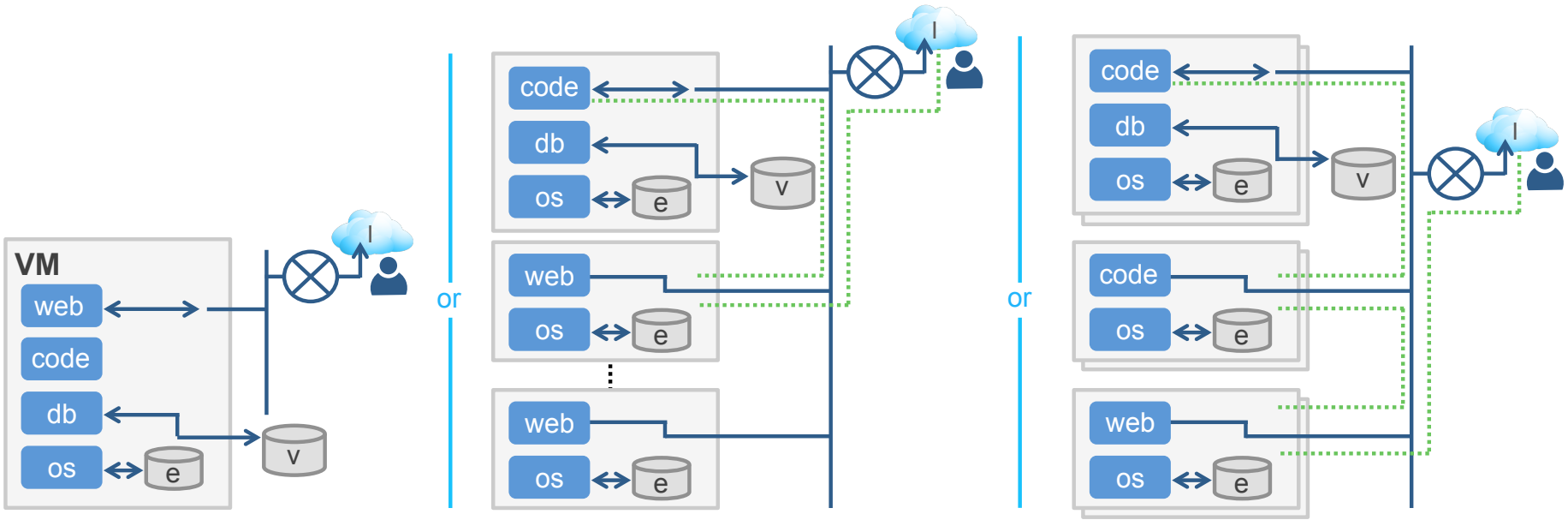
# Model for Application to be deployed

- Datastore on one node (how to configure, cloud-init?)
- App/interface on another node (configure, etc.)
- Network  model
- Security model

Model ➔ need data store =

View ➔ provides interface = port

Controller ➔ transforms = code



# What would you do with Physical Infra?

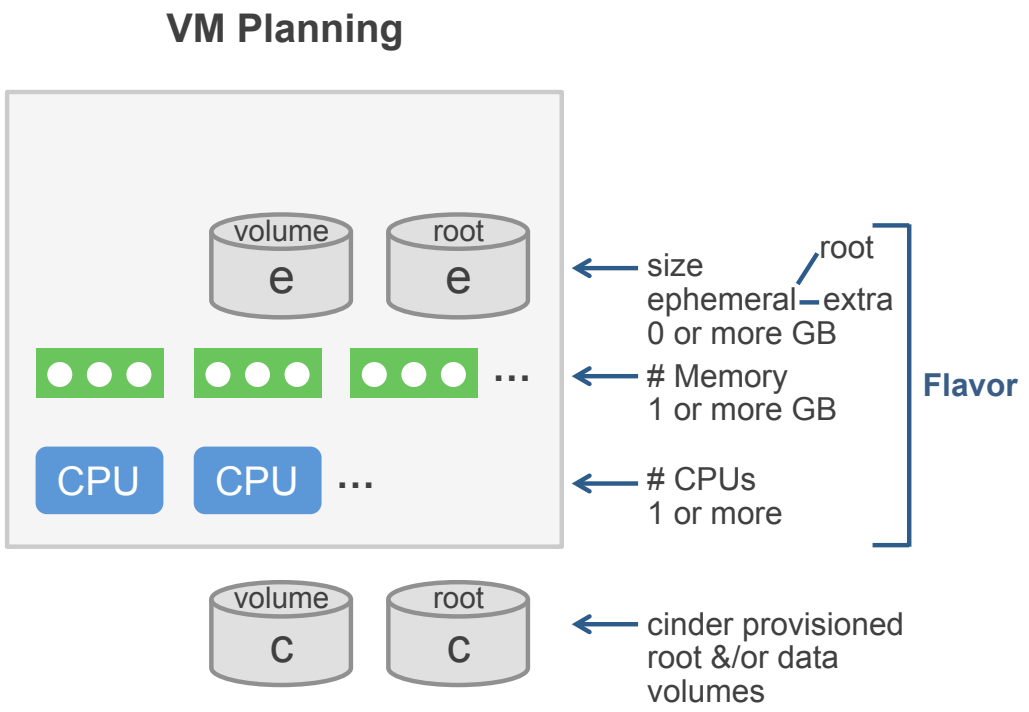
- Plan your application based on deployment constraints
  - Number of servers available
  - Type and quantity of storage available (perhaps also performance)
  - Network services needed (security, load balancing, DNS)
- Procure… wait… rack/stack/base-os… deploy!

- How is this different in OpenStack with Nova?

OneCloud
Consulting

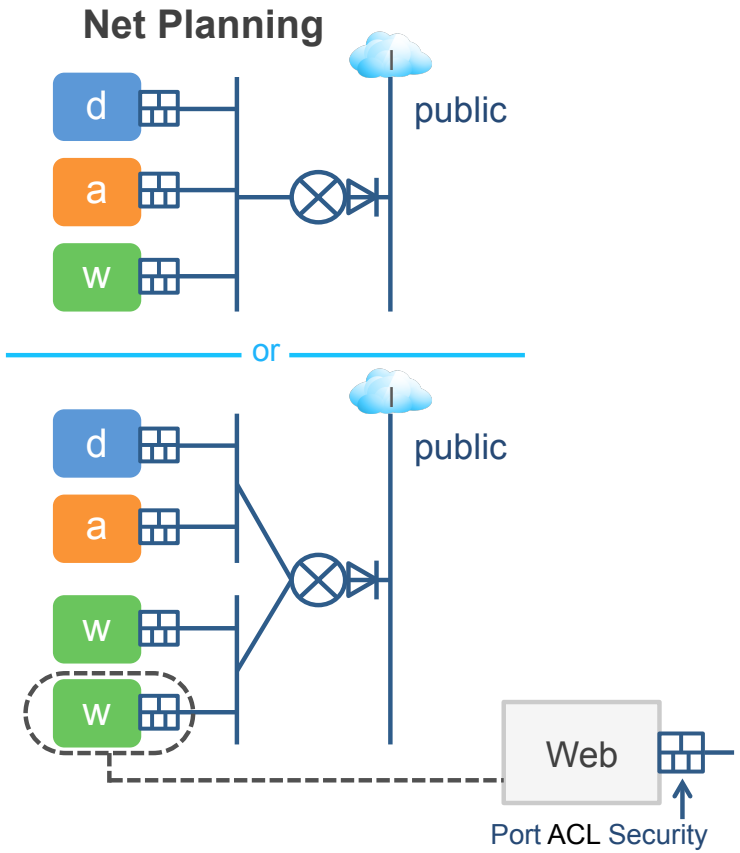
# Planning a VM deployment

- How big a VM do I need for each different service
  - Drives a decision on initial Flavor
    - changing can be done but is not automatic (today)
    - One big VM, many smaller instances
    - Small instance, and add as load increases
  - Affinity/anti-affinity
  - Do I need persistent storage
    - can I separate OS and Data
    - What is my recovery model if the VM dies



# Planning your VM network

- What Network segregation models should I use
  - Do I need to segregate services onto different networks
  - Is FloatingIP for public services enough segregation
- Do I need server load balancing
- What Security groups might have
  - Will I have more than one service on the tenant network
  - Are there groups of devices that should be allowed to talk to each other
  - Will I have separate backend networks that share a router



# How to deploy?

- Horizon UI
  - Graphical interface
  - Wizard to capture critical elements (name, image, keypair, network, etc.)
  - Can automatically create N identical instances
  - No way to re-do a deployment other than re-entering all the data
- CLI
  - Python based command utilities
  - Need to know parameter names (there is online help in the cli tools)
  - Can do an auto-repeat, and can poll for completion
  - Can be scripted to allow for repeating the same operation multiple times
- Heat
  - We will cover in detail later

# Multi-Site Deployments

- Same considerations for basic platform (flavor, disk, etc.)
- Database replication? Old models still apply, or, leverage new database model:
  - Cassandra, provides multi-site auto-replication
  - CouchDB, can be configured in a replicated model
- Don’t move golden images if possible (use generic pre-vetted base images and build out)
- Leverage Object Storage, both for application content, and possibly even for application code (auto-replicated)

# Multi-Site – Continued

- Application sync/replication, if possible over the public network, potentially provides the ability to scale out of a private environment to a public, or vice versa
- Consistent naming of objects (tenant networks, VM images, flavors, affinity groups) will simplify scripting multi-site deployments
- For Multi-site deployments, automation via CLI scripts (or other methods to talk to OpenStack APIs), is critical

OneCloud
Consulting