March 23, 2017

OpenStack Superuser (http://superuser.openstack.org)

Moving OpenStack beyond borders (and possibly to Mars) (http://superuser.openstack.org/articles/moving-openstack-beyond-borders-possibly-mars/)

The inaugural OpenStack Days Poland (http://www.openstackday.pl/) event drew more than 300 users, upstream developers, operators and vendors to the Copernicus Science Center (http://www.kopernik.org.pl/en/) in the heart of Warsaw, Poland on March 22.

Although Warsaw (https://en.wikipedia.org/wiki/Warsaw) is Poland's capital city—and according to "Forbes (https://www.forbes.com/sites/alisoncoleman/2016/05/20/poland-on-track-to-becoming-a-major-european-tech-startup-hub/#b43027f2085c)," a hotbed of startups (http://startuphub.pl/warsaw/) and multinational tech companies' European branches—this meetup traces its roots west to Wroclaw (https://www.forbes.com/sites/alisoncoleman/2016/05/20/poland-on-track-to-becoming-a-major-european-tech-startup-hub/#b43027f2085c), the Silicon Valley of Poland, according to some event speakers.

Wroclaw's large technology community formed an OpenStack group and held five local meetups before planning this Warsaw-based event designed to attract the attention of both users and developers from throughout Poland. The event is also considered the first meetup for the just-forming Warsaw- and Krakow (https://en.wikipedia.org/wiki/Krak%C3%B3w)-based Openstack user groups (https://groups.openstack.org/).

<figure class="wp-caption aligncenter" id="attachment_5897" style="width: 605px">

(http://superuser.openstack.org/wp-content/uploads/2017/03/room.jpeg)
<figcaption class="wp-caption-text"> *The audience came from throughout Poland and beyond, with several dozen users and some upstream developers, including Dragonflow's PTL Omer Anson. Photo: Heidi Joy Tretheway. //* </figcaption> </figure>
Indeed, there was good-natured arguing (over beers) about which city was best among those hailing from Krakow in the south, Gdansk in the north, Warsaw in the center and Wroclaw (say it VROT-swahv) in the west.

Sponsors and speakers hailed from all of these locations—and from both local companies and multinational vendors that are familiar names in OpenStack. A handful of attendees came from Ukraine and other neighboring countries, and a few presentations were in English, but most in Polish.

<figure class="wp-caption aligncenter" id="attachment_5898" style="width: 605px">

(http://superuser.openstack.org/wp-content/uploads/2017/03/pierogi.jpeg)
<figcaption class="wp-caption-text"> *If you can't learn Polish, at least learn Polish cooking. Here, both potato-onion and strawberry pierogis (https://en.wikipedia.org/wiki/Pierogi) are under construction at Polish Your Cooking (https://www.polishyourcooking.com/?lang=pl#top). Photo: Heidi Joy Tretheway. //* </figcaption> </figure>
"I think many companies in Warsaw are adopting OpenStack rapidly now because there is a strong ecosystem of companies that can make it happen with a lesser learning curve," a local representative from one OpenStack sponsor company said. "This year, we're seeing much more rapid adoption and companies moving to large-scale deployments."

Some of the organizations mentioned as running OpenStack are the browser Opera (http://www.opera.com/pl?gclid=CIrD1ISY6tICFYQ4Gwod6YEOmg), the Central and Eastern European online auction site Allegro (https://allegro.pl/), and one of the largest Polish hosting providers Nazwa (https://www.nazwa.pl/kampania,hosting,9711.html?gclid=CLGa1ayY6tICFQMM0wodsMIAbQ).

Cloud services provider OVH (https://www.ovh.pl/discover/poland.xml) noted that it hosts more than 80 petabytes (https://en.wikipedia.org/wiki/Petabyte) of data on Swift through its Dropbox-like infrastructure offering, and it will soon launch a new public cloud from a data center near Warsaw. OVH customers include Mailjet (https://eu.mailjet.com/email-platform?gclid=CJffna2Z6tICFcUp0wodx1QFMg), Villeroy & Boch (https://www.villeroy-boch.com), and PrestaShop (https://www.prestashop.com/en/translations).

<figure class="wp-caption aligncenter" id="attachment_5901" style="width: 605px">

(http://superuser.openstack.org/wp-content/uploads/2017/03/hall.jpeg)
<figcaption class="wp-caption-text"> *The market hall of vendors and sponsors (http://www.openstackday.pl/partnerzy/) was busy nonstop, with the "hallway track" very popular throughout the event. Photo: Heidi Joy Tretheway. //* </figcaption> </figure>
Rob McMahon (https://twitter.com/RobMcMahon5), director of cloud for Red Hat's EMEA region, told the audience that they are processing exabytes (https://en.wikipedia.org/wiki/Exabyte) (1 EB = 1,000 PB) of data on behalf of NASA's Jet Propulsion Laboratory. "I'd like to think that Martian life can be found with an OpenStack platform," he said.

This is Heidi Joy Tretheway's first visit to Warsaw and her fourth cycle leading OpenStack's User Survey (https://www.openstack.org/user-survey/survey-2017/landing?BackURL=/user-survey/survey-2017/). She was thrilled to see many other presenters quoting aspects of the User Survey and shared a sneak peek at the April 2017 statistics (https://www.dropbox.com/s/t9bs81cd9svimfe/Warsaw%20speech%20printed.pdf?dl=0) with attendees.

The post Moving OpenStack beyond borders (and possibly to Mars) (http://superuser.openstack.org/articles/moving-openstack-beyond-borders-possibly-mars/) appeared first on OpenStack Superuser (http://superuser.openstack.org).

by Heidi Joy Tretheway at March 23, 2017 09:00 AM (http://superuser.openstack.org/articles/moving-openstack-beyond-borders-possibly-mars/)

March 22, 2017

OpenStack Superuser (http://superuser.openstack.org)

Open source management as a marathon, not a sprint (http://superuser.openstack.org/articles/open-source-management/)

John Dickinson has been a project team lead (PTL) for Swift, OpenStack's object storage service, pretty much since it took off in 2011. At the time he was working at Rackspace, since 2012 he's been director of technology at San Francisco-based startup SwiftStack.

A frequent speaker (https://www.openstack.org/community/members/profile/101/john-dickinson) at OpenStack Summits and meetups, you can also find him at the upcoming Boston Summit giving an update on Swift. (https://www.openstack.org/summit/boston-2017/summit-schedule/speakers/922)

Dickinson (https://twitter.com/notmyname) offers Superuser advice for approaching PTLs and talks about how he's lasted this long without getting winded. Spoiler alert: modesty appears to be the secret to his longevity.

You're an example of a marathoner PTL – how do you do it?

The Swift (https://governance.openstack.org/tc/reference/projects/swift.html) community is great. I've never worked with such a talented and dedicated group of people and their support and passion is what keeps me going.

What are some of the biggest changes/advancements you've seen with Swift since you started?

Swift has grown from being a storage engine custom-built for a public cloud service provider into a more complete storage system that can be used for public cloud and private cloud at all scales. Swift is the best open-source object storage system available. I'm tremendously proud of what the community has produced over the last seven years.

My vision for Swift has always been for it to be used by everyone, every day. As more companies use Swift for more things, we get closer to that goal.

Significant features that we've written include global cluster support, storage policies, erasure coding and at-rest encryption. These are user-driven features not developed in a vacuum, but with actual use cases attached to them. That's how we succeed in the vision. We prioritize work that users are asking for, making changes based on data, without compromising on the stability and reliability Swift is known for.

> Swift keeps a lot of contributors in the room even it's the last day of PTG! Awesome! pic.twitter.com/IFTOueWbnp (https://t.co/IFTOueWbnp)
>
> — bloodeagle40234 (@bloodeagle40234) February 24, 2017 (https://twitter.com/bloodeagle40234/status/835134256745693184)

<script async="async" charset="utf-8" src="http://platform.twitter.com/widgets.js"></script>

What advice do you have for new contributors approaching a PTL or project?

Being a PTL requires a different skill set than being a developer contributor. You won't be great without practice, and you need great people around you to help out. So have a clear vision of what you want to see happen and surround yourself with good people.

How does the PTG affect your work with Swift?

Now that the PTG is over, I can reflect on the good things that happened. The best part of the PTG was spending time with my fellow Swift contributors. Not only did we get to spend time together discussing code changes, in-person gatherings are a great time to spend together as friends.

> Fried chicken dinner with #openstack (https://twitter.com/hashtag/openstack?src=hash) #swift (https://twitter.com/hashtag/swift?src=hash) #PTG (https://twitter.com/hashtag/PTG?src=hash) pic.twitter.com/rWb1Hbgp6H (https://t.co/rWb1Hbgp6H)
>
> — Janie Richling (@jrichli2) February 21, 2017 (https://twitter.com/jrichli2/status/833836616573710336)

<script async="async" charset="utf-8" src="http://platform.twitter.com/widgets.js"></script>

From a feature/design perspective, we made great progress at the PTG discussing some exciting features and optimizations that have been in-progress for quite some time. Some of the biggest upcoming changes in Swift are focused on optimization for larger scale deployments and features for data migration.

The PTG (http://superuser.openstack.org/articles/reflections-on-the-first-openstack-project-teams-gathering_amrith/)/Forum (http://superuser.openstack.org/articles/openstack-forum/)split is a big change to how things have been organized in the past. The PTG was successful, from my perspective and I hope the upcoming Summit in Boston (https://www.openstack.org/summit/austin-2016/summit-schedule/global-search?t=swift) will similarly be productive. I'm looking forward from hearing from the ops teams that run Swift clusters and learning from them.

Get involved!
Use Ask OpenStack (https://ask.openstack.org/) for general questions about Swift or OpenStack
For Swift road map or development issues, subscribe to the OpenStack development mailing list (http://lists.openstack.org/cgi-bin/mailman/listinfo/openstack-dev) and use the tag [swift]
Participate in the weekly Swift meetings: (https://wiki.openstack.org/wiki/Meetings/Swift) Wednesdays at 21:00UTC in #openstack-meeting on freenode IRC

Get involved!
Use Ask OpenStack (https://ask.openstack.org/) for general questions
For roadmap or development issues, subscribe to the OpenStack development mailing list, (http://lists.openstack.org/cgi-bin/mailman/listinfo/openstack-dev) and use the tag [kuryr]

Participate in the meetings: (http://eavesdrop.openstack.org/#Kuryr_Project_Meeting)

- Every two weeks (on odd weeks) on Tuesday at 0300 UTC in #openstack-meeting-4 (IRC webclient)
- Every two weeks (on even weeks) on Monday at 1500 UTC in #openstack-meeting-4 (IRC webclient)

The post Open source management as a marathon, not a sprint (http://superuser.openstack.org/articles/open-source-management/) appeared first on OpenStack Superuser (http://superuser.openstack.org).

by Nicole Martinelli at March 22, 2017 11:46 AM (http://superuser.openstack.org/articles/open-source-management/)

Aptira (https://aptira.com)

Sydney OpenStack Summit – The countdown is on! (https://aptira.com/sydney-openstack-summit/)

1
(https://feeds.feedburner.com/Aptira?format=xml)
1
(https://feeds.feedburner.com/Aptira?format=xml)

The Sydney OpenStack Summit (https://www.openstack.org/summit/sydney-2017/) is drawing closer, and we're excited to check out the new International Convention Center (http://www.iccsydney.com.au/).

This must-attend Open Source event will feature a mix of open technologies building the modern infrastructure stack, including OpenStack, Kubernetes, Docker, Ansible, Ceph, OVS, OpenContrail, OPNFV and more. You'll hear from some of the top Australian influencers, as well as industry leaders from around the globe.

If you're planning to attend, make sure you visit our stand. Not only will you get to meet the Aptira team, but there be prizes, giveaways, and we'll be showcasing our new Managed Cloud (https://aptira.com/managed-cloud/) offering.

There may even be a sneaky after party or two. Stay tuned for details!

The post Sydney OpenStack Summit – The countdown is on! (https://aptira.com/sydney-openstack-summit/) appeared first on Aptira Cloud Solutions (https://aptira.com).

by Tristan at March 22, 2017 03:50 AM (https://aptira.com/sydney-openstack-summit/)

March 21, 2017

OpenStack Superuser (http://superuser.openstack.org)

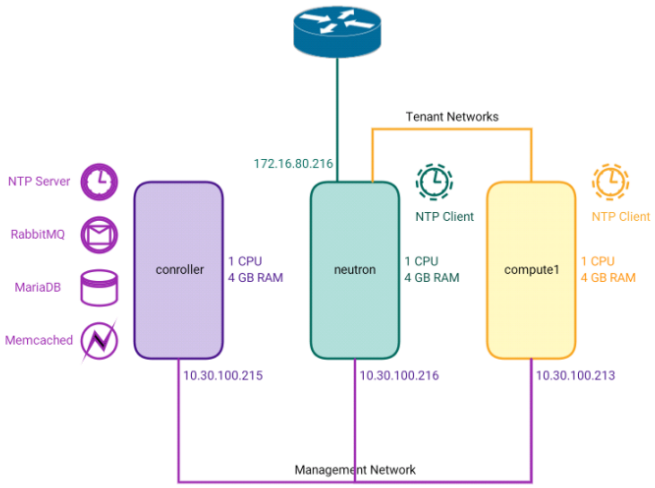
Getting to know the essential OpenStack components better (http://superuser.openstack.org/articles/openstack-essentials-tutorial/)

If you skipped the first part of this series, I am not responsible for the technical confusion and emotional stress created while you read this one. I highly recommend reading part one (http://superuser.openstack.org/articles/openstack-install-guide-beginner/) before continuing. However, if you like to live dangerously, then please carry on.

Part 1 recap:

- You were introduced to the main character, OpenStack.
- You learned its whereabouts and became acquainted with the "neighborhood."
- You learned what it needs to thrive and how to set it up for yourself.

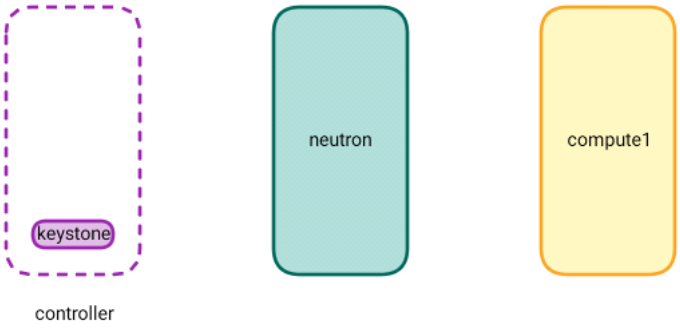
Here's an overview of what we learned in the last post:



Basic setup for deploying OpenStack

The first step to implementing OpenStack is to set up the identity service, **Keystone**.

A. Keystone



Keystone

Simply put, Keystone (https://docs.openstack.org/developer/keystone/) is a service that manages all identities. These identities can belong to your customers who you have offered services to, and also to all the micro-services that make up OpenStack. These identities have usernames and passwords associated with them in addition to the information on who is allowed to do what. There is much more to it but we will leave some detail out for now. If you work in tech, you're already familiar with the concept of identity/access cards. These cards not only identify who you are but also control which doors you can and cannot open on company premises. So in order to set up this modern day OpenStack watchman, perform the following steps:

@controller
Log in to MariaDB:

```
sudo mysql –u root –p
```

Create a database for Keystone and give the user full Keystone privileges to the newly created database:

```
CREATE DATABASE keystone;
GRANT ALL PRIVILEGES ON keystone.* TO 'keystone'@'localhost' \
  IDENTIFIED BY 'MINE_PASS';
GRANT ALL PRIVILEGES ON keystone.* TO 'keystone'@'%' \
  IDENTIFIED BY 'MINE_PASS';
exit
```

Install the Keystone software:

```
sudo apt install keystone
```

Edit the Keystone configuration file:

```
sudo vi /etc/keystone/keystone.conf
   #Tell keystone how to access the DB
   [database]
   connection = mysql+pymysql://keystone:MINE_PASS@controller/keystone (Comment out the exiting connection entry)

   #Some token management I don't fully understand. But put it in, its important)
   [token]
   provider = fernet
```

This command will initialize your Keystone database using the configuration that you just did above:

```
sudo su –s /bin/sh –c "keystone–manage db_sync" keystone
```

Since we have no identity management (because you are setting it up right now!), we need to bootstrap the identity management service to create an admin user for Keystone

```
sudo keystone–manage fernet_setup ––keystone–user keystone ––keystone–group keystone
sudo keystone–manage credential_setup ––keystone–user keystone ––keystone–group keystone
```

Since OpenStack is composed of a lot of micro-services, each service that we define will need to have an endpoint URL(s). This is how other services will access this service (notice, in the sample below, that there are three URLs). Run the following:

```
sudo keystone–manage bootstrap ––bootstrap–password MINE_PASS \
  ––bootstrap–admin–url http://controller:35357/v3/ \
  ––bootstrap–internal–url http://controller:35357/v3/ \
  ––bootstrap–public–url http://controller:5000/v3/ \
  ––bootstrap–region–id RegionOne
```

You need to configure Apache for Keystone. Keystone uses Apache to entertaining requests it receives from its other buddy services in OS. Lets just say that Apache is like a good secretary that is better at handling and managing requests than if Keystone tried to do it independently.

```
sudo vi /etc/apache2/apache2.conf
    ServerName controller
sudo service apache2 restart
sudo rm –f /var/lib/keystone/keystone.db
```

One of the most useful ways to interact with OS is via the command line, since, if you want to interact with OS, you need to be authenticated and authorized. An easy way to do this is to create the following file and then source it in your command line.

```
sudo vi ~/keystone_admin
    export OS_USERNAME=admin
    export OS_PASSWORD=MINE_PASS
    export OS_PROJECT_NAME=admin
    export OS_USER_DOMAIN_NAME=default
    export OS_PROJECT_DOMAIN_NAME=default
    export OS_AUTH_URL=http://controller:35357/v3
    export OS_IDENTITY_API_VERSION=3
    export PS1='[\u@\h \W(keystone_admin)]$ '
```

If you want to source just use the following command on the controller:

```
source ~/keystone_admin
```

Before we proceed, we need to talk about a few additional terms. OpenStack utilizes the concepts of domains, projects and users.

- **Users** are, well, just users of OpenStack.
- **Projects** are similar to customers in an OpenStack environment. So, if I am using my OpenStack environment to host VMS for Customer ABC and Customer XYZ, then ABC and XYZ could be two projects.
- **Domains** are a recent addition (as if things weren't already complex enough) that allows you further granularity. If you wanted to have administrative divisions within OpenStack so each division could manage their own environments, you would use domains. So you could put ABC and XYZ in different domains and have separate administration for both, or you could put them in the same domain and manage them with a single administration. Its just an added level of granularity. And you thought your relationships were complex!

Create a special project to hold all the internal users (most micro-services in OpenStack will have their own service users and they are associated to this special project.)

```
openstack project create --domain default \
  --description "Service Project" service
```

Verify Operations

Run the following command to request an authentication token using the admin user:

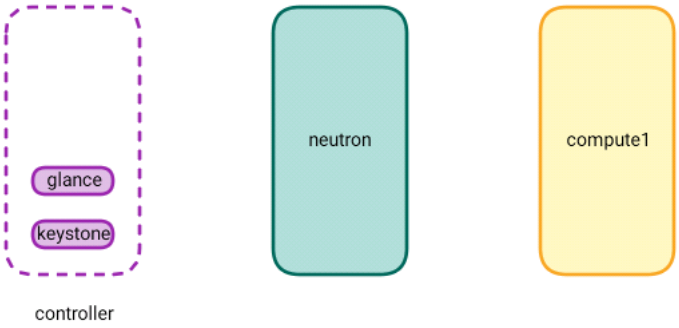
```
openstack --os-auth-url http://controller:35357/v3 \
  --os-project-domain-name default --os-user-domain-name default \
  --os-project-name admin --os-username admin token issue

Password:
+------------+-----------------------------------------------------------------+
| Field      | Value                                                           |
+------------+-----------------------------------------------------------------+
| expires    | 2016-11-30 13:05:15+00:00                                       |
| id         | gAAAAABYPsB7yua2kfIZhoDlm20y1i5IAHfXxIcqiKzhM9ac_MV4PU5OPiYf_   |
|            | m1SsUPOMSs4Bnf5A4o8i9B36c-gpxaUhtmzWx8WUVLpAtQDBgZ607ReW7cEYJGy |
|            | yTp54dskNkMji-uofna35ytrd2_VLIdMWk7Y1532HErA7phiq7hwKTKex-Y     |
| project_id | b1146434829a4b359528e1ddada519c0                                |
| user_id    | 97b1b7d8cb0d473c83094c795282b5cb                                |
+------------+-----------------------------------------------------------------+
```

Congratulations, you have the keys!

Now we meet, Glance (https://docs.openstack.org/developer/glance/) the image service. Glance is essentially a store for all the different flavors (images) of virtual machines that you may want to offer to your customers. When a customer requests a particular type of virtual machine, Glance finds the correct image in its repository and hands it over to another service (which we will discuss later).

B. Glance

<figure class="wp-caption alignnone" id="attachment_349">



<figcaption>Glance</figcaption> </figure>
To configure OpenStack's precious image store, perform the following steps on the **@controller**:

Log in to the DB:

```
sudo mysql –u root –p
```

Create the database and give full privileges to the Glance user:

```
CREATE DATABASE glance;
GRANT ALL PRIVILEGES ON glance.* TO 'glance'@'localhost' \
  IDENTIFIED BY 'MINE_PASS';
GRANT ALL PRIVILEGES ON glance.* TO 'glance'@'%' \
  IDENTIFIED BY 'MINE_PASS';
exit
```

Source the keystone_admin file to get command line access:

```
source ~/keystonerc_admin
```

Create the Glance user:

```
openstack user create ––domain default ––password–prompt glance
```

Give the user rights:

```
openstack role add ––project service ––user glance admin
```

Create the Glance service:

```
openstack service create ––name glance \
  ––description "OpenStack Image" image
```

Create the Glance endpoints:

```
openstack endpoint create ––region RegionOne \
  image public http://controller:9292
openstack endpoint create ––region RegionOne \
  image internal http://controller:9292
openstack endpoint create ––region RegionOne \
  image admin http://controller:9292
```

Install the Glance software:

```
sudo apt install glance
```

Configure the configuration file for Glance:

```
sudo vi /etc/glance/glance-api.conf
  #Configure the DB connection
  [database]
  connection = mysql+pymysql://glance:MINE_PASS@controller/glance

  #Tell glance how to get authenticated via keystone. Every time a service needs to do something it needs to be authenticated via keyst
one.
  [keystone_authtoken]
  auth_uri = http://controller:5000
  auth_url = http://controller:35357
  memcached_servers = controller:11211
  auth_type = password
  project_domain_name = default
  user_domain_name = default
  project_name = service
  username = glance
  password = MINE_PASS
  #(Comment out or remove any other options in the [keystone_authtoken] section.)

  [paste_deploy]
  flavor = keystone

  #Glance can store images in different locations. We are using file for now
  [glance_store]
  stores = file,http
  default_store /[= file
  filesystem_store_datadir = /var/lib/glance/images/
```

Edit another configuration file:

```
sudo vi /etc/glance/glance-registry.conf
  #Configure the DB connection
  [database]
  connection = mysql+pymysql://glance:MINE_PASS@controller/glance
  #Tell glance-registry how to get authenticated via keystone.
  [keystone_authtoken]
  auth_uri = http://controller:5000
  auth_url = http://controller:35357
  memcached_servers = controller:11211
  auth_type = password
  project_domain_name = default
  user_domain_name = default
  project_name = service
  username = glance
  password = MINE_PASS
  #(Comment out or remove any other options in the [keystone_authtoken] section.)

  #No Idea just use it.
  [paste_deploy]
  flavor = keystone
```

This command will initialize the Glance database using the configuration files above:

```
sudo su -s /bin/sh -c "glance-manage db_sync" glance
```

Start the Glance services:

```
sudo service glance-registry restart
sudo service glance-api restart
```

Verify Operation

Download a cirros cloud image:

```
wget http://download.cirros-cloud.net/0.3.4/cirros-0.3.4-x86_64-disk.img
```

Log in to the command line:

```
source ~/keystonerc_admin
```

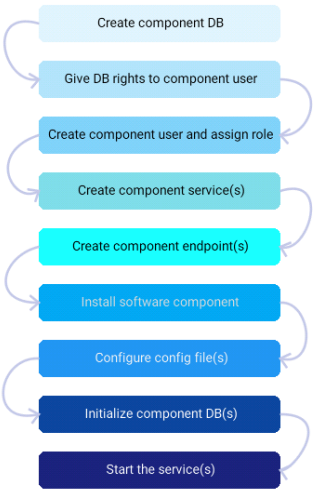
Create an OpenStack image using the command below:

```
openstack image create "cirros" \
  --file cirros-0.3.4-x86_64-disk.img \
  --disk-format qcow2 --container-format bare \
  --public
```

List the images and ensure that your image was created successfully:

```
openstack image list
+--------------------------------------+--------+--------+
| ID                                   | Name   | Status |
+--------------------------------------+--------+--------+
| d5edb2b0-ad3c-453a-b66d-5bf292dc2ee8 | cirros | active |
+--------------------------------------+--------+--------+
```

Are you noticing a pattern here? In part on (http://superuser.openstack.org/articles/openstack-install-guide-beginner/)e, I mentioned that the OpenStack components are similar but subtly different. As you will continue to see, most OpenStack services follow a standard pattern in configuration. The pattern is as follows:



General set of steps for configuring OpenStack components

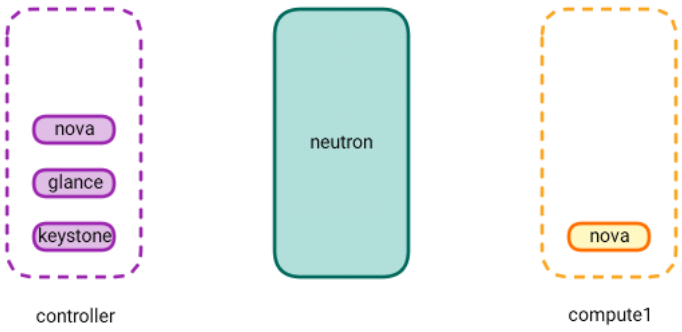
Most OpenStack components will follow the sequence above with minor deviations. So, if you're having trouble configuring a component, it's a good idea to refer to this list and see what steps you're missing.

What follows is probably one of the most important parts of OpenStack. It's called Nova (https://docs.openstack.org/developer/nova/) and, although it has nothing to do with stars and galaxies, it is still quite spectacular. At the end of this last section you got operating system image (from here on out, references to "image" mean "Glance image").

Now I'll introduce another concept, called an **instance**. An instance is what is created out of an image. This is the virtual machine that you use to provide services to your customers. In simpler terms, let's imagine you had a Windows CD and you use this CD to install Windows on a laptop. Then you use the same CD to install another Windows on another laptop. You input different license keys for both, create different users for each and they are then two individual and independent laptops running Windows from the same CD.

Using this analogy, an image is equivalent to the Windows CD with the Windows running on each of the laptops acting as different instances. Nova does the same thing by taking the CD from Glance and creating, configuring and managing instances in the cloud which are then handed to customers.

C. Nova



Nova

Nova is one of the more complex components that resides on more than one machine and does different things on each. A component of Nova sits on the controller and is responsible for overall management and communication with other OpenStack services and external world services. The second component sits on each compute (yes, you can have multiple computes, but we will look at those later). This service is primarily responsible for talking to the virtual machine monitors and making them launch and manage instances. Perform the following configuration to configure the Nova component:

@controller

Create the database(s) and grant relevant privileges:

```
sudo mysql -u root -p

CREATE DATABASE nova_api;
CREATE DATABASE nova;

GRANT ALL PRIVILEGES ON nova_api.* TO 'nova'@'localhost' \
  IDENTIFIED BY 'MINE_PASS';
GRANT ALL PRIVILEGES ON nova_api.* TO 'nova'@'%' \
  IDENTIFIED BY 'MINE_PASS';
GRANT ALL PRIVILEGES ON nova.* TO 'nova'@'localhost' \
  IDENTIFIED BY 'MINE_PASS';
GRANT ALL PRIVILEGES ON nova.* TO 'nova'@'%' \
  IDENTIFIED BY 'MINE_PASS';

exit
```

Log in to the command line:

```
source ~/keystonerc_admin
```

Create the user and assign the roles:

```
openstack user create --domain default \
  --password-prompt nova
openstack role add --project service --user nova admin
```

Create the service and the corresponding endpoints:

```
openstack service create --name nova \
  --description "OpenStack Compute" compute
openstack endpoint create --region RegionOne \
  compute public http://controller:8774/v2.1/%\(tenant_id\)s
openstack endpoint create --region RegionOne \
  compute internal http://controller:8774/v2.1/%\(tenant_id\)s
openstack endpoint create --region RegionOne \
  compute admin http://controller:8774/v2.1/%\(tenant_id\)s
```

Install the software:

```
sudo apt install nova-api nova-conductor nova-consoleauth \
  nova-novncproxy nova-scheduler
```

Configure the configuration file:

```
sudo vi /etc/nova/nova.conf
  #Configure the DB-1 access
  [api_database]
  connection = mysql+pymysql://nova:MINE_PASS@controller/nova_api

  #Configure the DB-2 access (nova has 2 DBs)
  [database]
  connection = mysql+pymysql://nova:MINE_PASS@controller/nova

  [DEFAULT]
  #Configure how to access RabbitMQ
  transport_url = rabbit://openstack:MINE_PASS@controller
  #Use the below. Some details will follow later
  auth_strategy = keystone
  my_ip = 10.30.100.215
  use_neutron = True
  firewall_driver = nova.virt.firewall.NoopFirewallDriver

  #Tell Nova how to access keystone
  [keystone_authtoken]
  auth_uri = http://controller:5000
  auth_url = http://controller:35357
  memcached_servers = controller:11211
  auth_type = password
  project_domain_name = default
  user_domain_name = default
  project_name = service
  username = nova
  password = MINE_PASS

  #This is a remote access to instance consoles (French ? Just take it on faith. We will explore this in a much later episode)
  [vnc]
  vncserver_listen = $my_ip
  vncserver_proxyclient_address = $my_ip

  #Nova needs to talk to glance to get the images
  [glance]
  api_servers = http://controller:9292

  #Some locking mechanism for message queing (Just use it.)
  [oslo_concurrency]
  lock_path = /var/lib/nova/tmp
```

Initialize both of the databases using the configuration done above:

```
sudo su -s /bin/sh -c "nova-manage api_db sync" nova
sudo su -s /bin/sh -c "nova-manage db sync" nova
```

Start all the Nova services:

```
sudo service nova-api restart
sudo service nova-consoleauth restart
sudo service nova-scheduler restart
sudo service nova-conductor restart
sudo service nova-novncproxy restart
```

@compute1
Install the software:

```
sudo apt install nova-compute
```

Configure the configuration file:

```
sudo vi /etc/nova/nova.conf
  [DEFAULT]
  #Define DB access
  transport_url = rabbit://openstack:MINE_PASS@controller
  #Take it on faith for now
  auth_strategy = keystone
  my_ip = 10.30.100.213
  use_neutron = True
  firewall_driver = nova.virt.firewall.NoopFirewallDriver

  #Tell the nova-compute how to access keystone
  [keystone_authtoken]
  auth_uri = http://controller:5000
  auth_url = http://controller:35357
  memcached_servers = controller:11211
  auth_type = password
  project_domain_name = default
  user_domain_name = default
  project_name = service
  username = nova
  password = MINE_PASS

  #This is a remote access to instance consoles (French ? Just take it on faith. We will explore this in a much later episode)
  [vnc]
  enabled = True
  vncserver_listen = 0.0.0.0
  vncserver_proxyclient_address = $my_ip
  novncproxy_base_url = http://controller:6080/vnc_auto.html

  #Nova needs to talk to glance to get the images
  [glance]
  api_servers = http://controller:9292

  #Some locking mechanism for message queuing (Just use it.)
  [oslo_concurrency]
  lock_path = /var/lib/nova/tmp
```

The following requires some explanation. In a production environment, your compute will be a physical machine, so the below steps will **NOT** be required. But since this is a lab, we need to set the virtualization type for KVM hypervisor to **qemu** (as opposed to KVM). This setting runs the hypervisor *without* looking for the **hardware acceleration** that is provided by KVM on a physical machine. So, you are going to run virtual machines inside a virtual machine in the lab and, trust me, it works.😉

For a virtual compute:

```
sudo vi /etc/nova/nova-compute.conf
  [libvirt]
  virt_type = qemu
```

Start the Nova service:

```
sudo service nova-compute restart
```

Verify Operation

@controller

Log in to the command line:

```
source ~/keystonerc_admin
```

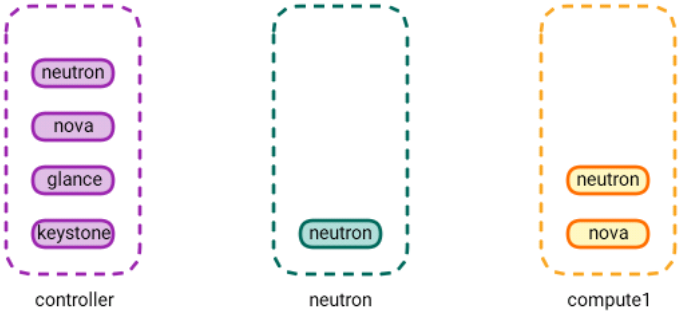
Run the following command to list the Nova services. Ensure the state is up as show below:

```
openstack compute service list
+----+------------------+-----------------------+----------+---------+-------+----------------------------+
| ID | Binary           | Host                  | Zone     | Status  | State | Updated At                 |
+----+------------------+-----------------------+----------+---------+-------+----------------------------+
|  3 | nova-consoleauth | controller            | internal | enabled | up    | 2016-11-30T12:54:39.000000 |
|  4 | nova-scheduler   | controller            | internal | enabled | up    | 2016-11-30T12:54:36.000000 |
|  5 | nova-conductor   | controller            | internal | enabled | up    | 2016-11-30T12:54:34.000000 |
|  6 | nova-compute     | compute1              | nova     | enabled | up    | 2016-11-30T12:54:33.000000 |
+----+------------------+-----------------------+----------+---------+-------+----------------------------+
```

Notice the similarity in the sequence of steps followed to configure Nova?

Time for some physics! Remember that our goal is to provide services to our customers. These services are in the form of virtual machines, or services running over these virtual machines. If we want to cater to many customers, then each of them will have their own set of services they are consuming. These services, like any other infrastructure, will require a network. You may need things like routers, firewalls, load balancers, VPN and so on. Now imagine setting these up manually for each customer. Yeah, that's not happening. This is exactly what **Neutron** does for you.

D. Neutron



Neutron

Of course, it's never going to be easy. In my personal opinion, Neutron (https://wiki.openstack.org/wiki/Neutron)is of OpenStack's personality that has the worst temper and suffers from very frequent mood swings. In simpler terms, it's complex. In our setup, major Neutron services will be residing on two servers, namely the controller and the Neutron. You could put everything on one machine and it would work, however splitting is suggested by the official documentation. (My hunch is that it has something to do with easier scaling.) There will also be a Neutron component on the compute node which I will explain later.

However, before we get into detailing the configuration, I need to explain a few minor terms:

- When we talk about networks under neutron, we will come across mainly two types of networks. One is called an **external network**. An external network is usually configured once and represents the network used by OS to access the external world. The second is **tenant networks**, and these are the networks assigned to customers.
- An OpenStack environment also requires a virtual switching (bridging) component in order to manage virtualized networking across neutron and compute nodes. The two components mostly used are Linux Bridge and Open vSwitch. (If you want to understand a bit more about Open vSwitch, you can refer to one of my other entries Understanding Open vSwitch (http://superuser.openstack.org/articles/openvswitch-openstack-sdn/)). We will be using Open vSwitch for our environment. Please note that if you intend to use Linux Bridge, the configuration will be different.

To deploy Neutron, perform the following configuration:

@controller
Create the database and assign full rights to the Neutron user (yawn!):

```
sudo mysql –u root –p
CREATE DATABASE neutron;
GRANT ALL PRIVILEGES ON neutron.* TO 'neutron'@'localhost' \
  IDENTIFIED BY 'MINE_PASS';
GRANT ALL PRIVILEGES ON neutron.* TO 'neutron'@'%' \
  IDENTIFIED BY 'MINE_PASS';
exit
```

Log in to command line:

```
source ~/keystonerc_admin
```

Create the Neutron user and add the role:

```
openstack user create --domain default --password-prompt neutron
openstack role add --project service --user neutron admin
```

Create the Neutron service and the respective endpoints:

```
openstack service create --name neutron \
  --description "OpenStack Networking" network
openstack endpoint create --region RegionOne \
  network public http://controller:9696
openstack endpoint create --region RegionOne \
  network internal http://controller:9696
openstack endpoint create --region RegionOne \
  network admin http://controller:9696
```

Install the software components:

```
sudo apt install neutron-server neutron-plugin-ml2
```

Configure the Neutron config file:

```
sudo vi /etc/neutron/neutron.conf
  [DEFAULT]
  #This is a multi-layered plugin
  core_plugin = ml2
  service_plugins = router
  allow_overlapping_ips = True
  notify_nova_on_port_status_changes = True
  notify_nova_on_port_data_changes = True

  [DATABASE]
  #Configure the DB connection
  connection = mysql+pymysql://neutron:MINE_PASS@controller/neutron

  [keystone_authtoken]
  #Tell neutron how to talk to keystone
  auth_uri = http://controller:5000
  auth_url = http://controller:35357
  memcached_servers = controller:11211
  auth_type = password
  project_domain_name = default
  user_domain_name = default
  project_name = service
  username = neutron
  password = MINE_PASS

  [nova]
  #Tell neutron how to talk to nova to inform nova about changes in the network
  auth_url = http://controller:35357
  auth_type = password
  project_domain_name = default
  user_domain_name = default
  region_name = RegionOne
  project_name = service
  username = nova
  password = MINE_PASS
```

Configure the plugin file:

```
sudo vi /etc/neutron/plugins/ml2/ml2_conf.ini
  [ml2]
  #In our environment we will use vlan networks so the below setting is sufficient. You could also use vxlan and gre, but that is for a
 later episode
  type_drivers = flat,vlan
  #Here we are telling neutron that all our customer networks will be based on vlans
  tenant_network_types = vlan
  #Our SDN type is openVSwitch
  mechanism_drivers = openvswitch,l2population
  extension_drivers = port_security

  [ml2_type_flat]
  #External network is a flat network
  flat_networks = external

  [ml2_type_vlan]
  #This is the range we want to use for vlans assigned to customer networks.
  network_vlan_ranges = external,vlan:1381:1399

  [securitygroup]
  #Use Ip tables based firewall
  firewall_driver = iptables_hybrid
```

Note that I tried to run the su command directly using sudo and, for some reason, it fails for me. An alternative is to use “sudo su” (get root access) and then run the database using the config files above. Run the following sequence to instantiate the database.

```
sudo su -
su -s /bin/sh -c "neutron-db-manage --config-file /etc/neutron/neutron.conf \
  --config-file /etc/neutron/plugins/ml2/ml2_conf.ini upgrade head" neutron
```

Start all the Neutron services:

```
sudo servive neutron-* restart
```

Edit the Nova configuration file:

```
sudo vi /etc/nova/nova.conf
  #Tell nova how to get in touch with neutron, to get network updates
  [neutron]
  url = http://controller:9696
  auth_url = http://controller:35357
  auth_type = password
  project_domain_name = default
  user_domain_name = default
  region_name = RegionOne
  project_name = service
  username = neutron
  password = MINE_PASS
  service_metadata_proxy = True
  metadata_proxy_shared_secret = MINE_PASS
```

Restart Nova services:

```
sudo service nova-* restart
```

@neutron
Install the required services:

```
sudo apt install neutron-plugin-ml2 \
  neutron-l3-agent neutron-dhcp-agent \
  neutron-metadata-agent neutron-openvswitch-agent
```

Run the following Open vSwitch commands to create the following bridges:

Create a bridge named"**br-ex**;" this will connect OS to the external network:

```
sudo ovs-vsctl add-br br-ex
```

Add a port from the br-ex bridge to the **ens10** interface. In my environment, **ens10** is the interface connected on the External Network. You should change it as per your environment.

```
sudo ovs-vsctl add-port br-ex ens10
```

The following bridge that we will add is used by the VLAN networks for the customer networks in OS. Run the following command to create the bridge.

```
sudo ovs-vsctl add-br br-vlan
```

Add a port fromt the br-vlan bridge to the **ens9** interface. In my environment, ens9 is the interface connected on the tunnel network. You should change it as per your environment.

```
sudo ovs-vsctl add-port br-vlan ens9
```

For our Open vSwitch configuration to persist beyond server reboots, we need to configure the interface file accordingly.

```
sudo vi /etc/network/interfaces
  # This file describes the network interfaces available on your system
  # and how to activate them. For more information, see interfaces(5).

  source /etc/network/interfaces.d/*

  # The loopback network interface
  # No Change
  auto lo
  iface lo inet loopback

  #No Change on management network
  auto ens3
  iface ens3 inet static
  address 10.30.100.216
  netmask 255.255.255.0

  # Add the br-vlan bridge
  auto br-vlan
  iface br-vlan inet manual
  up ifconfig br-vlan up

  # Configure ens9 to work with OVS
  auto ens9
  iface ens9 inet manual
  up ip link set dev $IFACE up
  down ip link set dev $IFACE down

  # Add the br-ex bridge and move the IP for the external network to the bridge
  auto br-ex
  iface br-ex inet static
  address 172.16.8.216
  netmask 255.255.255.0
  gateway 172.16.8.254
  dns-nameservers 8.8.8.8

  # Configure ens10 to work with OVS. Remove the IP from this interface
  auto ens10
  iface ens10 inet manual
  up ip link set dev $IFACE up
  down ip link set dev $IFACE down
```

Reboot to ensure the new configuration is fully applied

```
sudo reboot
```

Configure the neutron config file

```
sudo vi /etc/neutron/neutron.conf
  [DEFAULT]
  auth_strategy = keystone
  #Tell neutron how to access RabbitMQ
  transport_url = rabbit://openstack:MINE_PASS@controller

  #Tell neutron how to access keystone
  [keystone_authtoken]
  auth_uri = http://controller:5000
  auth_url = http://controller:35357
  memcached_servers = controller:11211
  auth_type = password
  project_domain_name = default
  user_domain_name = default
  project_name = service
  username = neutron
  password = MINE_PASS
```

Configure the Open vSwitch agent config file

```
sudo vi /etc/neutron/plugins/ml2/openvswitch_agent.ini
  #Configure the section for OpenVSwitch
  [ovs]
  #Note that we are mapping alias(es) to the bridges. Later we will use these aliases (vlan,external) to define networks inside OS.
  bridge_mappings = vlan:br-vlan,external:br-ex

  [agent]
  l2_population = True

  [securitygroup]
  #Ip table based firewall
  firewall_driver = iptables_hybrid
```

Configure the Layer 3 Agent configuration file:

```
sudo vi /etc/neutron/l3_agent.ini
  [DEFAULT]
  #Tell the agent to use the OVS driver
  interface_driver = neutron.agent.linux.interface.OVSInterfaceDriver
  #This is required to be set like this by the official documentation (If you don't set it to empty as show below, sometimes your route
r ports in OS will not become Active)
  external_network_bridge =
```

Configure the DHCP Agent config file:

```
sudo vi /etc/neutron/dhcp_agent.ini
  [DEFAULT]
  #Tell the agent to use the OVS driver
  interface_driver = neutron.agent.linux.interface.OVSInterfaceDriver
  enable_isolated_metadata = True
```

Configure the Metadata Agent config file:

```
sudo vi /etc/neutron/metadata_agent.ini

  [DEFAULT]
  nova_metadata_ip = controller
  metadata_proxy_shared_secret = MINE_PASS
```

Start all Neutron services:

```
sudo service neutron-* restart
```

@compute1

Install the ml2 plugin and the openvswitch agent:

```
sudo apt install neutron-plugin-ml2 \
 neutron-openvswitch-agent
```

Create the Open vSwitch bridges for tenant VLANs:

```
sudo ovs-vsctl add-br br-vlan
```

Add a port on the br-vlan bridge to the ens9 interface. In my environment ens9 is the interface connected on the tunnel network. You should change it as per your environment.

```
sudo ovs-vsctl add-port br-vlan ens9
```

In order for our Open vSwitch configuration to persist beyond reboots we need to configure the interface file accordingly.

```
sudo vi /etc/network/interfaces

  # This file describes the network interfaces available on your system
  # and how to activate them. For more information, see interfaces(5).

  source /etc/network/interfaces.d/*

  # The loopback network interface
  #No Change
  auto lo
  iface lo inet loopback

  #No Change to management network
  auto ens3
  iface ens3 inet static
  address 10.30.100.213
  netmask 255.255.255.0

  # Add the br-vlan bridge interface
  auto br-vlan
  iface br-vlan inet manual
  up ifconfig br-vlan up

  #Configure ens9 to work with OVS
  auto ens9
  iface ens9 inet manual
  up ip link set dev $IFACE up
  down ip link set dev $IFACE down
```

Reboot to ensure the new network configuration is applied successfully:

```
sudo reboot
```

Configure the Neutron config file:

```
sudo vi /etc/neutron/neutron.conf
  [DEFAULT]
  auth_strategy = keystone
  #Tell neutron component how to access RabbitMQ
  transport_url = rabbit://openstack:MINE_PASS@controller

  #Configure access to keystone
  [keystone_authtoken]
  auth_uri = http://controller:5000
  auth_url = http://controller:35357
  memcached_servers = controller:11211
  auth_type = password
  project_domain_name = default
  user_domain_name = default
  project_name = service
  username = neutron
  password = MINE_PASS
```

Configure the Nova config file:

```
sudo vi /etc/nova/nova.conf
  #Tell nova how to access neutron for network topology updates
  [neutron]
  url = http://controller:9696
  auth_url = http://controller:35357
  auth_type = password
  project_domain_name = default
  user_domain_name = default
  region_name = RegionOne
  project_name = service
  username = neutron
  password = MINE_PASS
```

Configure the Open vSwitch agent configuration:

```
sudo vi /etc/neutron/plugins/ml2/openvswitch_agent.ini

#Here we are mapping the alias vlan to the bridge br-vlan
 [ovs]
 bridge_mappings = vlan:br-vlan

[agent]
 l2_population = True

[securitygroup]
 firewall_driver = iptables_hybrid
```

It's a good idea to reboot the compute at this point. (I was getting connectivity issues without rebooting. Let me know how it goes for you.)

```
sudo reboot
```

Start all Neutron services:

```
sudo service neutron-* restart
```

Verify Operation

@controller

Log in to the command line:

```
source ~/keystonerc_admin
```

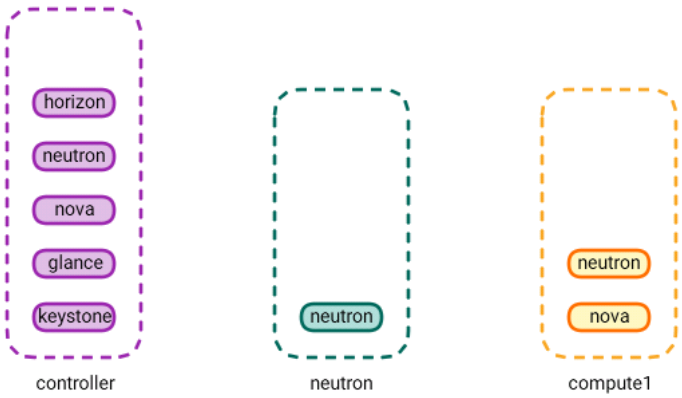
Run the following command to list the neutron agents. Ensure the "Alive" status is "True" and "State" is "Up" as show below:

```
openstack network agent list
+--------------------------------------+--------------------+----------------------+-------------------+-------+-------+--------------
-------------+
| ID                                   | Agent Type         | Host                 | Availability Zone | Alive | State | Binar
y                    |
+--------------------------------------+--------------------+----------------------+-------------------+-------+-------+--------------
-------------+
| 84d81304-1922-47ef-8b8e-c49f83cff911 | Metadata agent     | neutron              | None              | True  | UP    | neutron-metad
ata-agent    |
| 93741a55-54af-457e-b182-92e15d77b7ae | L3 agent           | neutron              | None              | True  | UP    | neutron-l3-ag
ent          |
| a3c9c1e5-46c3-4649-81c6-dc4bb9f35158 | Open vSwitch agent | neutron              | None              | True  | UP    | neutron-openv
switch-agent |
| ba9ce5bb-6141-4fcc-9379-c9c20173c382 | DHCP agent         | neutron              | nova              | True  | UP    | neutron-dhcp-
agent        |
| e458ba8a-8272-43bb-bb83-ca0aae48c22a | Open vSwitch agent | compute1             | None              | True  | UP    | neutron-openv
switch-agent |
+--------------------------------------+--------------------+----------------------+-------------------+-------+-------+--------------
-------------+
```

It's becoming a bit of a drag, I know. But hang on: we're almost there.

Up to this point, our interactions with OpenStack have taken place in a mostly black-and-white environment. It's time to add some color to this relationship and see what we can achieve by pressing the right buttons.

E. Horizon



Horizon

Horizon (https://docs.openstack.org/developer/horizon/) is the component that handles the graphical user interface for OpenStack. It's simple and sweet — and so is configuring it.

Perform the following configuration to deploy Horizon:

@controller
Install the software:

```
sudo apt install openstack-dashboard
```

Configuration file updates. Please search for these entries in the file and then replace them to avoid any duplicates:

```
sudo vi /etc/openstack-dashboard/local_settings.py
  OPENSTACK_HOST = "controller"
  ALLOWED_HOSTS = ['*', ]

  SESSION_ENGINE = 'django.contrib.sessions.backends.cache'

  CACHES = {
   'default': {
   'BACKEND': 'django.core.cache.backends.memcached.MemcachedCache',
   'LOCATION': 'controller:11211',
   }
  }

  OPENSTACK_KEYSTONE_URL = "http://%s:5000/v3" % OPENSTACK_HOST
  OPENSTACK_KEYSTONE_MULTIDOMAIN_SUPPORT = True
  OPENSTACK_API_VERSIONS = {
   "identity": 3,
   "image": 2,
   "volume": 2,
  }
  OPENSTACK_KEYSTONE_DEFAULT_DOMAIN = "default"
  OPENSTACK_KEYSTONE_DEFAULT_ROLE = "user"
  TIME_ZONE = "TIME_ZONE"
```

Replace `TIME_ZONE` with an appropriate time zone identifier. For more information, see the list of time zones (http://en.wikipedia.org/wiki/List_of_tz_database_time_zones).

Start the dashboard service:

```
sudo service apache2 reload
```

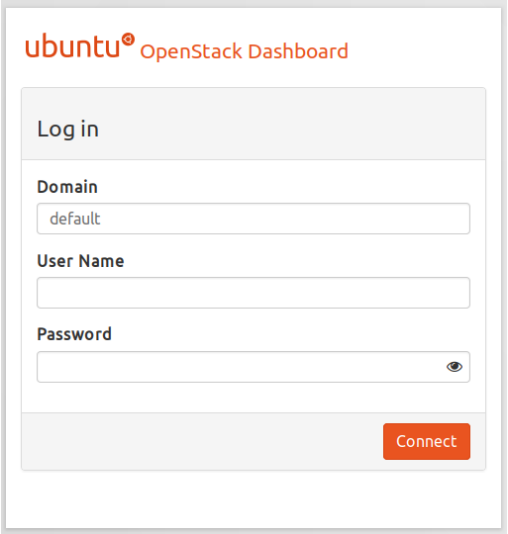
Verify Operation

Using an computer that has access to the controller open the following URL in the browser:

http://controller/horizon (http://controller/horizon)

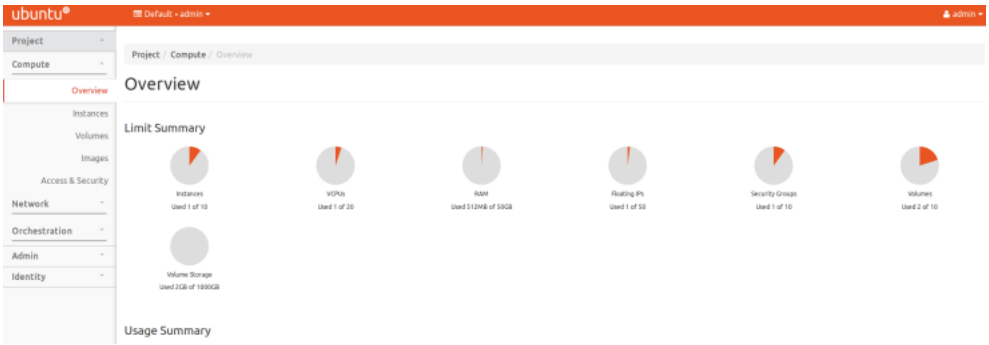
Please replace **controller** in the above URL with the controller IP if you cannot resolve the controller by its alias. So in my case the URL will become: **http://10.30.1.215/horizon (http://10.30.1.215/horizon)**

You should see a login screen similar to the one below



OpenStack Dashboard login screen

Enter **admin** as the username and the password you used to set up the user. In my case, it is **MINE_PASS.** If you are successfully logged in and see an interface similar (not necessarily the same) to the one below then your Horizon dashboard is working just fine.

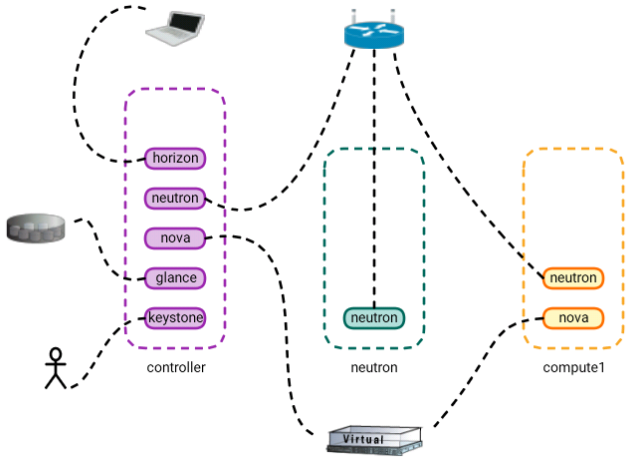


OpenStack Dashboard

Recap:

Look at the diagram below. Here's what we've achieved so far.

<figure class="wp-caption alignnone" id="attachment_350">



</figure>

Next:

- We will learn about OpenStack's block storage component...
- ...and its orchestration service
- The fun continues!

Once again thanks for reading. If you have any questions/comments please comment below so everyone can benefit from the discussion.

This post first appeared on the WhatCloud (https://whatcloud.wordpress.com/2016/11/30/ose2gtnhb/) blog. Superuser is always interested in community content, email: editorATsuperuser.org (mailto:editor@superuser.org).

Cover Photo (https://flic.kr/p/b4GqqX) // CC BY NC (https://creativecommons.org/licenses/by-nc/2.0/)

The post Getting to know the essential OpenStack components better (http://superuser.openstack.org/articles/openstack-essentials-tutorial/) appeared first on OpenStack Superuser (http://superuser.openstack.org).

by Nooruddin Abbas at March 21, 2017 11:31 AM (http://superuser.openstack.org/articles/openstack-essentials-tutorial/)

March 20, 2017

Cloudwatt (https://dev.cloudwatt.com/en/blog/index.html)

5 Minutes Stacks, episode 55: Wordpress (https://dev.cloudwatt.com/en/blog/5-minutes-stacks-episode-fifty-five-wordpress.html)

Episode Two : Wordpress

In the CMS Open-Source galaxy, WordPress is the most used in term of community, available features and user adoption. The Automattic compagny, which develop and distribute Wordpress, provides a SaaS offer allowing a user to create its blog in few minutes. However, those who experiments know that limits can be easily found by the wordpress.com hosting capabilities.

Today, Cloudwatt provides the necessary toolset to start your Wordpress instance in a few minutes and to become its master.

The deployement base is an Ubuntu xenial instance. The Apache and MariaDB servers are deployed on a single instance.

Preparations

The versions

- Ubuntu 16.04
- Apache 2.4.18
- Wordpress 4.7.1
- MariaDB 10.0.28
- PHP 7.0.13

A one-click deployment sounds really nice...

... Good! Go to the Apps page (https://www.cloudwatt.com/en/apps/) on the Cloudwatt website, choose the apps, press **DEPLOY** and follow the simple steps... 2 minutes later, a green button appears... **ACCESS**: you have your Wordpress!.

All of this is fine, but you do not have a way to run the stack thru the console ?

Yes ! Using the console, you can deploy a Wordpress server:

1. Go the Cloudwatt Github in the applications/bundle-xenial-wordpress repository
2. Click on the file nammed bundle-xenial-wordpress.heat.yml
3. Click on RAW, a web page appear with the script details
4. Save as its content on your PC. You can use the default name proposed by your browser (just remove the .txt)
5. Go to the « Stacks (https://console.cloudwatt.com/project/stacks/) » section of the console
6. Click on « Launch stack », then click on « Template file » and select the file you've just saved on your PC, then click on « NEXT »
7. Named your stack in the « Stack name » field
8. Enter your keypair in the « keypair_name » field
9. Choose the instance size using the « flavor_name » popup menu and click on « LAUNCH »

The stack will be automatically created (you can see its progress by clicking on its name). When all its modules will become "green", the creation will be completed. Then you can go on the "Instances" menu to discover the flotting IP value that has been automatically generated. Now, just run this IP adress in your browser and enjoy !

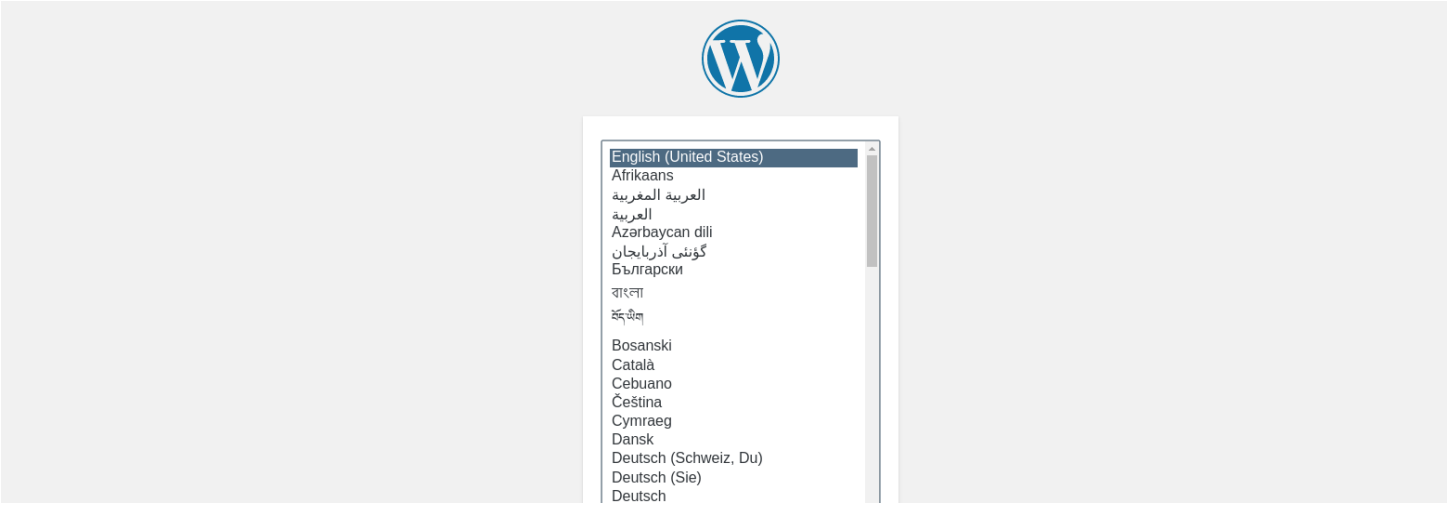
It is (already) FINISH !

Install cli

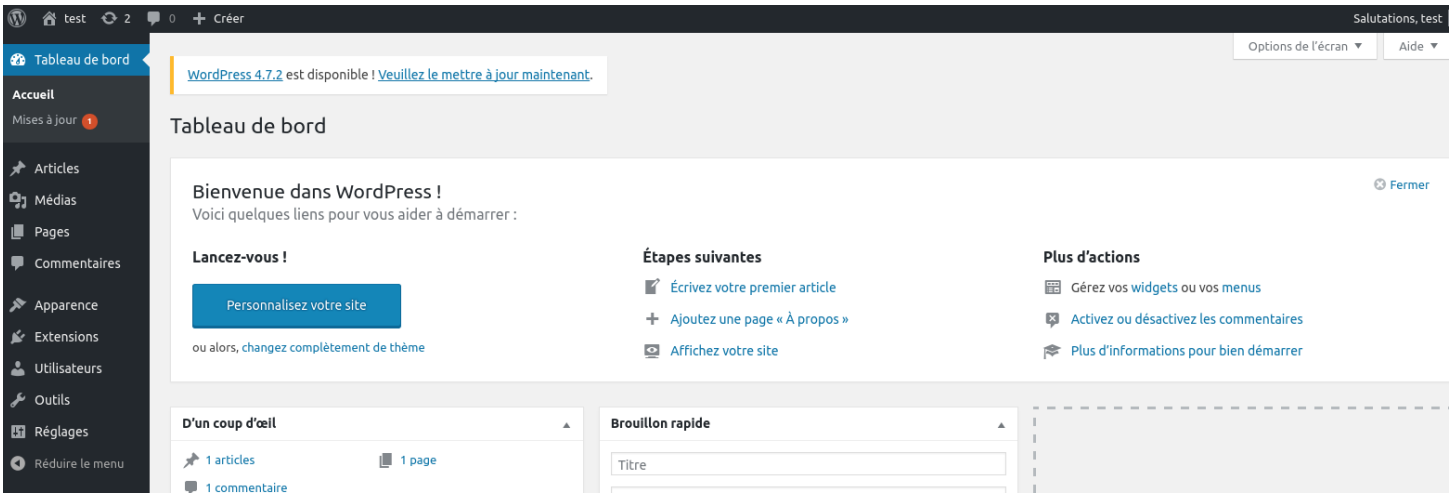
If you like only the command line, you can go directly to the "CLI launch" version by clicking this link (https://dev.cloudwatt.com/en/blog/rss.xml# cli)

For further

Install your Wordpress



Homepage + BackOffice



Configuration of the database

The configuration of the database is in `/data/wordpress/wp-config.php` file.

So watt ?

The goal of this tutorial is to accelarate your start. At this point you are the master of the stack. You have a SSH access point on your virtual machine thru the flotting IP and your private keypair (default user name `cloud`).

The interesting entry access points are:

- `/data/wordpress` : Wordpress installation repository.
- `/data/mysql` : Mariadb nodes datadir is a cinder volume.

Resources you could be interested in:

- wordpress (https://wordpress.com/)

Install cli

The prerequisites to deploy this stack

- an internet acces
- a Linux shell
- a Cloudwatt account (https://www.cloudwatt.com/cockpit/#/create-contact), with an existing keypair (https://console.cloudwatt.com/project/access_and_security/?tab=access_security_tabs__keypairs_tab)
- the tools OpenStack CLI (http://docs.openstack.org/cli-reference/content/install_clients.html)

- a local clone of the git repository Cloudwatt applications (https://github.com/cloudwatt/applications)

Size of the instance

Per default, the script is proposing a deployement on an instance type “Small” (s1.cw.small-1). Instances are charged by the minute and capped at their monthly price (you can find more details on the Tarifs page (https://www.cloudwatt.com/fr/produits/tarifs.html) on the Cloudwatt website). Obviously, you can adjust the stack parameters, particularly its defaut size.

By the way...

If you do not like command lines, you can go directly to the “run it thru the console” section by clicking here (https://dev.cloudwatt.com/en/blog/rss.xml#console)

What will you find in the repository

Once you have cloned the github, you will find in the `bundle-xenial-wordpress/` repository:

- `bundle-xenial-wordpress.heat.yml` : HEAT orchestration template. It will be use to deploy the necessary infrastructure.
- `stack-start.sh` : Stack launching script. This is a small script that will save you some copy-paste.
- `stack-get-url.sh` : Flotting IP recovery script.

Start-up

Initialize the environment

Have your Cloudwatt credentials in hand and click HERE (https://console.cloudwatt.com/project/access_and_security/api_access/openrc/). If you are not logged in yet, you will go thru the authentication screen then the script download will start. Thanks to it, you will be able to initiate the shell acccesses towards the Cloudwatt APIs.

Source the downloaded file in your shell. Your password will be requested.

```
$ source COMPUTE-[...]-openrc.sh
Please enter your OpenStack Password:
```

Once this done, the Openstack command line tools can interact with your Cloudwatt user account.

Adjust the parameters

With the `bundle-xenial-wordpress.heat.yml` file, you will find at the top a section named `parameters` . The sole mandatory parameter to adjust is the one called `keypair_name` . Its `default` value must contain a valid keypair with regards to your Cloudwatt user account. This is within this same file that you can adjust the instance size by playing with the `flavor` parameter.

```
heat_template_version: 2015-04-30


description: All-in-one Wordpress stack


parameters:
  keypair_name:
    default: amaury-ext-compute         <-- Indicate here your keypair
    description: Keypair to inject in instances
    type: string

  flavor_name:
      default: s1.cw.small-1              <-- Indicate here the flavor size
      description: Flavor to use for the deployed instance
      type: string
      constraints:
        - allowed_values:
            - s1.cw.small-1
            - n1.cw.standard-1
            - n1.cw.standard-2
            - n1.cw.standard-4
            - n1.cw.standard-8
            - n1.cw.standard-12
            - n1.cw.standard-16

[...]
```

Start up the stack

In a shell, run the script `stack-start.sh` with the name you want to give it as parameter:

```
$ ./stack-start.sh Wordpress
+--------------------------------------+------------+--------------------+----------------------+
| id                                   | stack_name | stack_status       | creation_time        |
+--------------------------------------+------------+--------------------+----------------------+
| ed4ac18a-4415-467e-928c-1bef193e4f38 | Wordpress  | CREATE_IN_PROGRESS | 2015-04-21T08:29:45Z |
+--------------------------------------+------------+--------------------+----------------------+
```

Last, wait 5 minutes until the deployement been completed.

At each new deployement of the stack, a password is generated directly in the `/data/wordpress/config-default.php` configuration file.

Enjoy

Once all of this done, you can run the `stack-get-url.sh` script.

```
./stack-get-url.sh Wordpress
Wordpress 82.40.34.249
```

It will gather the assigned flotting IP of your stack. You can then paste this IP in your favorite browser and start to configure your Wordpress instance.

In the background

The `start-stack.sh` script is taking care of running the API necessary requests to:

- start an Ubuntu Xenial based instance
- do an update of the system packages
- install Apache, PHP, MariaDB and Wordpress
- configure MariaDB with a wordpress dedicated user and database, with a generated password
- show a flotting IP on the internet

Have fun. Hack in peace.

by Mohamed-Ali at March 20, 2017 11:00 PM (https://dev.cloudwatt.com/en/blog/5-minutes-stacks-episode-fifty-five-wordpress.html)

Mirantis (https://www.mirantis.com)

Using Kubernetes Helm to install applications (https://www.mirantis.com/blog/install-kubernetes-apps-helm/)

The post Using Kubernetes Helm to install applications (https://www.mirantis.com/blog/install-kubernetes-apps-helm/) appeared first on Mirantis | Pure Play Open Cloud (https://www.mirantis.com).

Difficulty is a relative thing. Deploying an application using containers can be much easier than trying to manage deployments of a traditional application over different environments, but trying to manage and scale multiple containers manually is much more difficult than orchestrating them using Kubernetes. But even managing Kubernetes applications looks difficult compared to, say, “apt-get install mysql”. Fortunately, the container ecosystem has now evolved to that level of simplicity. Enter Helm.

Helm (https://github.com/kubernetes/helm) is a Kubernetes-based package installer. It manages Kubernetes “charts”, which are “preconfigured packages of Kubernetes resources.” Helm enables you to easily install packages, make revisions, and even roll back complex changes.

Next week, my colleague Maciej Kwiek will be giving a talk at Kubecon about Boosting Helm with AppController (https://cloudnativeeu2017.sched.com/event/9Tcb), so we thought this might be a good time to give you an introduction to what it is and how it works.

Let’s take a quick look at how to install, configure, and utilize Helm.

After reading this introduction to Kubernetes Helm, you will know how to:

- Install Helm
- Configure Helm
- Use Helm to determine available packages
- Use Helm to install a software package
- Retrieve a Kubernetes Secret
- Use Helm to delete an application
- Use Helm to roll back changes to an application

Install Helm

Installing Helm is actually pretty straightforward. Follow these steps:

1. Download the latest version of Helm from https://github.com/kubernetes/helm/releases (https://github.com/kubernetes/helm/releases). (Note that if you are using an older version of Kubernetes (1.4 or below) you might have to downgrade Helm due to breaking changes.)
2. Unpack the archive:
   $ **gunzip helm-v2.2.3-darwin-amd64.tar.gz**
   $ **tar -xvf helm-v2.2.3-darwin-amd64.tar**
   x darwin-amd64/
   x darwin-amd64/helm
   x darwin-amd64/LICENSE
   x darwin-amd64/README.md
3. Next move the helm executable to your path:
   ```
   $ mv dar*/helm /usr/local/bin/.
   ```
4. Finally, initialize helm to both set up the local environment and to install the server portion, Tiller, on your cluster. (Helm will use the default cluster for Kubernetes, unless you tell it otherwise.)

```
$ helm init
Creating /Users/nchase/.helm
Creating /Users/nchase/.helm/repository
Creating /Users/nchase/.helm/repository/cache
Creating /Users/nchase/.helm/repository/local
Creating /Users/nchase/.helm/plugins
Creating /Users/nchase/.helm/starters
Creating /Users/nchase/.helm/repository/repositories.yaml
Writing to /Users/nchase/.helm/repository/cache/stable-index.yaml
$HELM_HOME has been configured at /Users/nchase/.helm.

Tiller (the helm server side component) has been instilled into your Kubernetes Cluster.
Happy Helming!
```

Note that you can also upgrade the Tiller component using:

```
helm init --upgrade
```

That's all it takes to install Helm itself; now let's look at using it to install an application.

Install an application with Helm

One of the things that Helm does is enable authors to create and distribute their own applications using charts; to get a full list of the charts that are available, you can simply ask:

```
$ helm search
NAME                            VERSION DESCRIPTION
stable/aws-cluster-autoscaler 0.2.1   Scales worker nodes within autoscaling groups.
stable/chaoskube              0.5.0   Chaoskube periodically kills random pods in you...
stable/chronograf             0.1.2   Open-source web application written in Go and R...
...
```

In our case, we're going to install MySQL from the stable/mysql chart. Follow these steps:

1. First update the repo, just as you'd do with apt-get update:

   ```
   $ helm repo update
   Hang tight while we grab the latest from your chart repositories...
   ...Skip local chart repository
   Writing to /Users/nchase/.helm/repository/cache/stable-index.yaml
   ...Successfully got an update from the "stable" chart repository
   Update Complete. ⎈ Happy Helming!⎈
   ```

2. Next, we'll do the actual install:

   ```
   $ helm install stable/mysql
   ```

   This command produces a lot of output, so let's take it one step at a time. First, we get information about the release that's been deployed:

   ```
   NAME:   lucky-wildebeest
   LAST DEPLOYED: Thu Mar 16 16:13:50 2017
   NAMESPACE: default
   STATUS: DEPLOYED
   ```

   As you can see, it's called lucky-wildebeest, and it's been successfully DEPLOYED.

   Your release will, of course, have a different name. Next, we get the resources that were actually deployed by the stable/mysql chart:

   ```
   RESOURCES:
   ==> v1/Secret
   NAME                    TYPE    DATA  AGE
   lucky-wildebeest-mysql  Opaque  2     0s

   ==> v1/PersistentVolumeClaim
   NAME                    STATUS  VOLUME                                     CAPACITY  ACCESSMODES  AGE
   lucky-wildebeest-mysql  Bound   pvc-11ebe330-0a85-11e7-9bb2-5ec65a93c5f1  8Gi       RWO          0s

   ==> v1/Service
   NAME                    CLUSTER-IP  EXTERNAL-IP  PORT(S)   AGE
   lucky-wildebeest-mysql  10.0.0.13   <none>       3306/TCP  0s

   ==> extensions/v1beta1/Deployment
   NAME                    DESIRED  CURRENT  UP-TO-DATE  AVAILABLE  AGE
   lucky-wildebeest-mysql  1        1        1           0          0s
   ```

   This is a good example because we can see that this chart configures multiple types of resources: a Secret (for passwords), a persistent volume (to store the actual data), a Service (to serve requests) and a Deployment (to manage it all).

The chart also enables the developer to add notes:

```
NOTES:
MySQL can be accessed via port 3306 on the following DNS name from within your cluster:
lucky-wildebeest-mysql.default.svc.cluster.local

To get your root password run:
    kubectl get secret --namespace default lucky-wildebeest-mysql -o jsonpath="{.data.mysql-root-password}" | base64 --decode; echo

To connect to your database:
 Run an Ubuntu pod that you can use as a client:
    kubectl run -i --tty ubuntu --image=ubuntu:16.04 --restart=Never -- bash -il

 Install the mysql client:
    $ apt-get update && apt-get install mysql-client -y

 Connect using the mysql cli, then provide your password:
$ mysql -h lucky-wildebeest-mysql -p
```

These notes are the basic documentation a user needs to use the actual application. There let's see how we put it all to use.

Connect to mysql

The first lines of the notes make it seem deceptively simple to connect to MySql:

```
MySQL can be accessed via port 3306 on the following DNS name from within your cluster:
lucky-wildebeest-mysql.default.svc.cluster.local
```

Before you can do anything with that information, however, you need to do two things: get the root password for the database, and get a working client with network access to the pod hosting it.

Get the mysql password

Most of the time, you'll be able to get the root password by simply executing the code the developer has left you:

```
$ kubectl get secret --namespace default lucky-wildebeest-mysql -o jsonpath="{.data.mysql-root-password}" | base64 --decode; echo
DBTzmbAikO
```

Some systems — notably MacOS — will give you an error:

```
$ kubectl get secret --namespace default lucky-wildebeest-mysql -o jsonpath="{.data.mysql-root-password}" | base64 --decode; echo
Invalid character in input stream.
```

This is because of an error in base64 that adds an extraneous character. In this case, you will have to extract the password manually. Basically, we're going to execute the same steps as this line of code, but one at a time.

Start by looking at the Secrets that Kubernetes is managing:

```
$ kubectl get secrets
NAME                     TYPE                                  DATA      AGE
default-token-0q3gy      kubernetes.io/service-account-token   3         145d
lucky-wildebeest-mysql   Opaque                                2         20m
```

It's the second, lucky-wildebeest-mysql that we're interested in. Let's look at the information it contains:

```
$ kubectl get secret lucky-wildebeest-mysql -o yaml
apiVersion: v1
data:
  mysql-password: a1p1THdRcTVrNg==
  mysql-root-password: REJUem1iQWlrTw==
kind: Secret
metadata:
  creationTimestamp: 2017-03-16T20:13:50Z
  labels:
    app: lucky-wildebeest-mysql
    chart: mysql-0.2.5
    heritage: Tiller
    release: lucky-wildebeest
  name: lucky-wildebeest-mysql
  namespace: default
  resourceVersion: "43613"
  selfLink: /api/v1/namespaces/default/secrets/lucky-wildebeest-mysql
  uid: 11eb29ed-0a85-11e7-9bb2-5ec65a93c5f1
type: Opaque
```

You probably already figured out where to look, but the developer's instructions told us the raw password data was here:

```
jsonpath="{.data.mysql-root-password}"
```

So we're looking for this:

```
apiVersion: v1
data:
  mysql-password: a1p1THdRcTVrNg==
  mysql-root-password: REJUem1iQWlrTw==
kind: Secret
metadata:
...
```

Now we just have to go ahead and decode it:

```
$ echo "REJUem1iQWlrTw==" | base64 --decode
DBTzmbAikO
```

Finally! So let's go ahead and connect to the database.

Create the mysql client

Now we have the password, but if we try to just connect iwt the mysql client on any old machine, we'll find that there's no connectivity outside of the cluster. For example, if I try to connect with my local mysql client, I get an error:

```
$ ./mysql -h lucky-wildebeest-mysql.default.svc.cluster.local -p
Enter password:
ERROR 2005 (HY000): Unknown MySQL server host 'lucky-wildebeest-mysql.default.svc.cluster.local' (0)
```

So what we need to do is create a pod on which we can run the client. Start by creating a new pod using the ubuntu:16.04 image:

```
$ kubectl run -i --tty ubuntu --image=ubuntu:16.04 --restart=Never

$ kubectl get pods
NAME                                  READY     STATUS             RESTARTS   AGE
hello-minikube-3015430129-43g6t       1/1       Running            0          1h
lucky-wildebeest-mysql-3326348642-b8kfc   1/1       Running            0          31m
ubuntu                                1/1       Running            0          25s
```

When it's running, go ahead and attach to it:

```
$ kubectl attach ubuntu -i -t

Hit enter for command prompt
```

Next install the mysql client:

```
root@ubuntu2:/# apt-get update && apt-get install mysql-client -y
Get:1 http://archive.ubuntu.com/ubuntu xenial InRelease [247 kB]
Get:2 http://archive.ubuntu.com/ubuntu xenial-updates InRelease [102 kB]
...
Setting up mysql-client-5.7 (5.7.17-0ubuntu0.16.04.1) ...
Setting up mysql-client (5.7.17-0ubuntu0.16.04.1) ...
Processing triggers for libc-bin (2.23-0ubuntu5) ...
```

Now we should be ready to actually connect. Remember to use the password we extracted in the previous step.

```
root@ubuntu2:/# mysql -h lucky-wildebeest-mysql -p
Enter password:

Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 410
Server version: 5.7.14 MySQL Community Server (GPL)

Copyright (c) 2000, 2016, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
```

Of course you can do what you want here, but for now we'll go ahead and exit both the database and the container:

```
mysql> exit
Bye
root@ubuntu2:/# exit
logout
```

So we've successfully installed an application — in this case, MySql, using Helm. But what else can Helm do?

Working with revisions

So now that you've seen Helm in action, let's take a quick look at what you can actually do with it. Helm is designed to let you install, upgrade, delete, and roll back revisions. We'll get into more details about upgrades in a later article on creating charts, but let's quickly look at deleting and rolling back revisions:

First off, each time you make a change with Helm, you're creating a Revision. By deploying MySql, we created a Revision, which we can see in this list:

```
NAME              REVISION UPDATED                  STATUS CHART         NAMESPACE
lucky-wildebeest     1        Sun Mar 19 22:07:56 2017 DEPLOYEmysql-0.2.5   default
operatic-starfish    2        Thu Mar 16 17:10:23 2017 DEPLOYEredmine-0.4.0 default
```

As you can see, we created a revision called lucky-wildebeest, based on the mysql-0.2.5 chart, and its status is DEPLOYED.

We could also get back the information we got when it was first deployed by getting the status of the revision:

```
$ helm status intended-mule
LAST DEPLOYED: Sun Mar 19 22:07:56 2017
NAMESPACE: default
STATUS: DEPLOYED

RESOURCES:
==> v1/Secret
NAME                 TYPE    DATA  AGE
intended-mule-mysql  Opaque  2     43m

==> v1/PersistentVolumeClaim
NAME                 STATUS  VOLUME                                     CAPACITY  ACCESSMODES  AGE
intended-mule-mysql  Bound   pvc-08e0027a-0d12-11e7-833b-5ec65a93c5f1  8Gi       RWO          43m
...
```

Now, if we wanted to, we could go ahead and delete the revision:

```
$ helm delete lucky-wildebeest
```

Now if you list all of the active revisions, it'll be gone.

```
$ helm ls
```

However, even though the revision s gone, you can still see the status:

```
$ helm status lucky-wildebeest
LAST DEPLOYED: Sun Mar 19 22:07:56 2017
NAMESPACE: default
STATUS: DELETED

NOTES:
MySQL can be accessed via port 3306 on the following DNS name from within your cluster:
lucky-wildebeest-mysql.default.svc.cluster.local

To get your root password run:

    kubectl get secret --namespace default lucky-wildebeest-mysql -o jsonpath="{.data.mysql-root-password}" | base64 --decode; echo

To connect to your database:

 Run an Ubuntu pod that you can use as a client:

    kubectl run -i --tty ubuntu --image=ubuntu:16.04 --restart=Never -- bash -il

 Install the mysql client:

    $ apt-get update && apt-get install mysql-client -y

 Connect using the mysql cli, then provide your password:

    $ mysql -h lucky-wildebeest-mysql -p
```

OK, so what if we decide that we've changed our mind, and we want to roll back that deletion? Fortunately, Helm is designed for that. We can specify that we want to rollback our application to a specific revision (in this case, 1).

```
$ helm rollback lucky-wildebeest 1
Rollback was a success! Happy Helming!
```

We can see that the application is back, and the revision has been incremented:

```
NAME              REVISION UPDATED                  STATUS CHART         NAMESPACE
lucky-wildebeest     2        Sun Mar 19 23:46:52 2017 DEPLOYEmysql-0.2.5   default
```

We can also check the status:

```
$ helm status intended-mule
LAST DEPLOYED: Sun Mar 19 23:46:52 2017
NAMESPACE: default
STATUS: DEPLOYED

RESOURCES:
==> v1/Secret
NAME                 TYPE    DATA  AGE
intended-mule-mysql  Opaque  2     21m

==> v1/PersistentVolumeClaim
NAME                 STATUS  VOLUME                                     CAPACITY  ACCESSMODES  AGE
intended-mule-mysql  Bound   pvc-dad1b896-0d1f-11e7-833b-5ec65a93c5f1  8Gi       RWO          21m
...
```

Next time, we'll talk about how to create charts for Helm. Meanwhile, if you're going to be at Kubecon, don't forget Maciej Kwiek's talk on Boosting Helm with AppController (https://cloudnativeeu2017.sched.com/event/9Tcb).

The post Using Kubernetes Helm to install applications (https://www.mirantis.com/blog/install-kubernetes-apps-helm/) appeared first on Mirantis | Pure Play Open Cloud (https://www.mirantis.com).

by Nick Chase at March 20, 2017 09:20 PM (https://www.mirantis.com/blog/install-kubernetes-apps-helm/)

Arie Bregman (http://abregman.com)

InfraRed: Deploying and Testing Openstack just made easier! (http://abregman.com/2017/03/20/infrared-deploying-and-testing-openstack-just-made-easier/)

Deploying and testing OpenStack is very easy If you read the headline and your eyebrows raised, you are at the right place. I believe that most of us, who experienced at least one deployment of OpenStack, will agree that deploying OpenStack can be a quite frustrating experience. It doesn't matter if you are using it for [...]

by bregman at March 20, 2017 08:05 PM (http://abregman.com/2017/03/20/infrared-deploying-and-testing-openstack-just-made-easier/)

Alessandro Pilotti (https://cloudbase.it)

Setting the Windows admin password in OpenStack (https://cloudbase.it/openstack-windows-admin-password/)

We're getting quite a few questions about how to set the admin password in OpenStack Windows instances, so let's clarify the available options.

nova get-password

The secure and proper way to set passwords in OpenStack Windows instances is by letting Cloudbase-Init (https://cloudbase.it/cloudbase-init) generate a random password and post it encrypted on the Nova metadata service. The password can then be retrieved with:

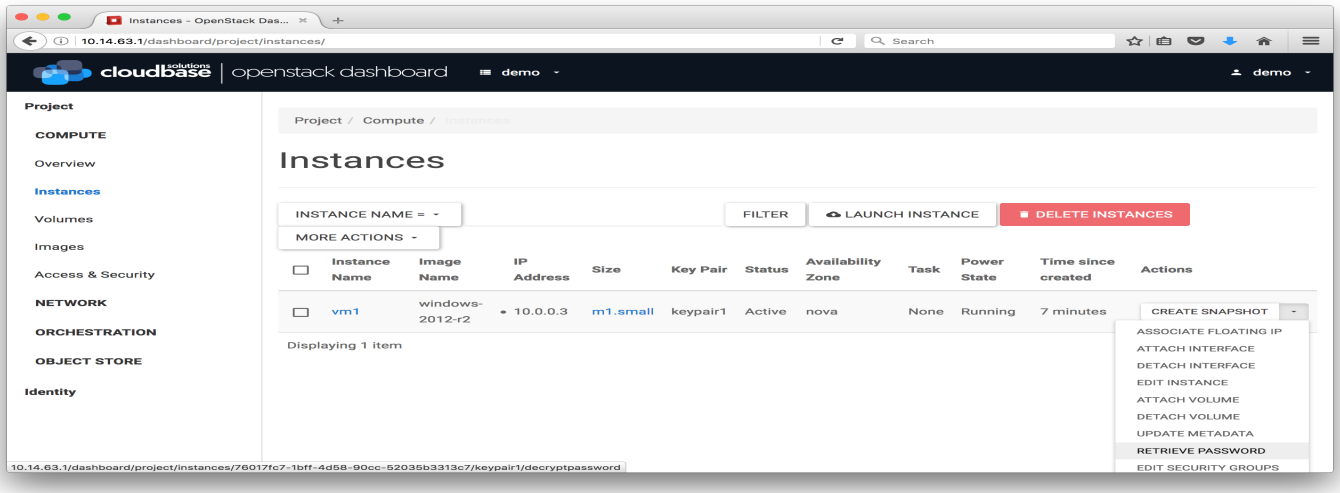
```
nova get-password <instance> [<ssh_private_key_path>]
```

You need to boot your instance with a SSH keypair (exactly like you would do on Linux for SSH public key authentication). In this case the public key is used to encrypt the password before posting it to the Nova HTTP metadata service. This way nobody will be able to decrypt it without having the keypair's private key.

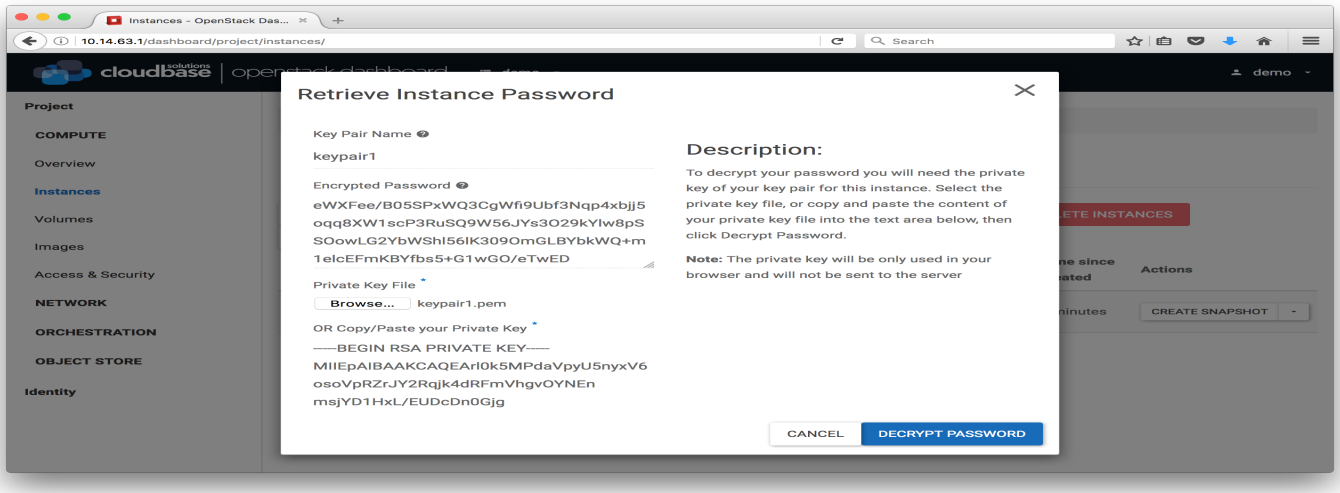
This option is also well supported in Horizon, but not enabled by default. To enable it, just edit *openstack_dashboard/local/local_settings.py* and add:

```
OPENSTACK_ENABLE_PASSWORD_RETRIEVE = True
```

To retrieve the password in Horizon, select "RETRIEVE PASSWORD" from the instance dropdown menu:

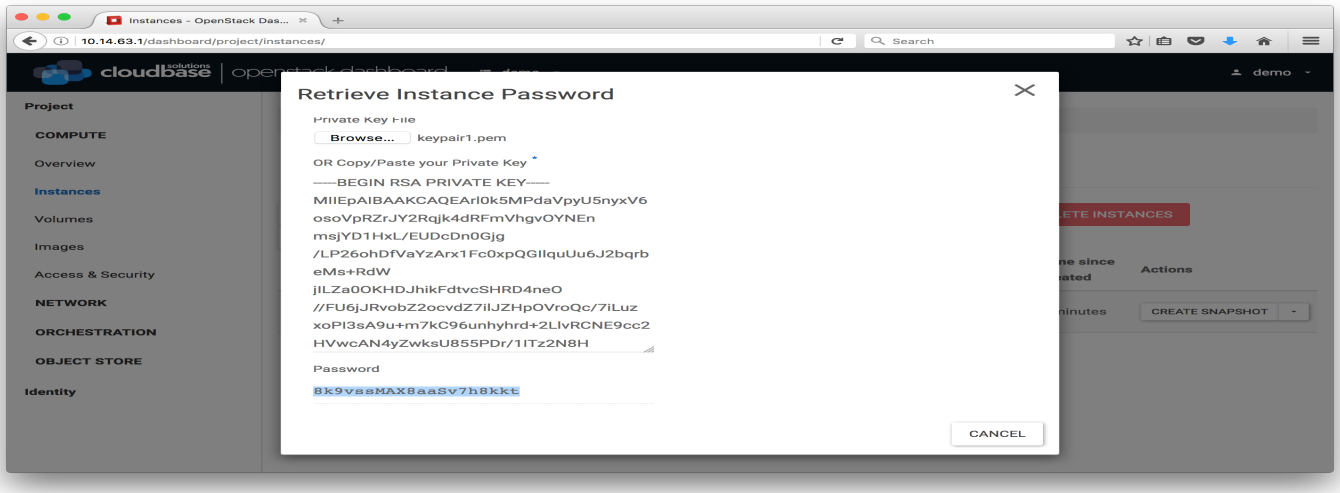


(https://cloudbase.it/openstack-windows-admin-password/horizon_retrieve_password_1/)
(https://cloudbase.it/openstack-windows-admin-password/horizon_retrieve_password_2/)Browse for your private key:



(https://cloudbase.it/openstack-windows-admin-password/horizon_retrieve_password_3/)

Click “DECRYPT PASSWORD” (de decryption will occur in the browser, no data will be sent to the server) and retrieve your password:



(https://cloudbase.it/openstack-windows-admin-password/horizon_retrieve_password_4/)

nova boot –meta admin_pass

In case a password automatically generated is not suitable, there's an option to provide a password via command line. This is **NOT RECOMMENDED** due to the security implications of sharing clear text passwords in the metadata content.

In this case the password is provided to the Nova instance via metadata service and assigned by Cloudbase-Init to the admin user:

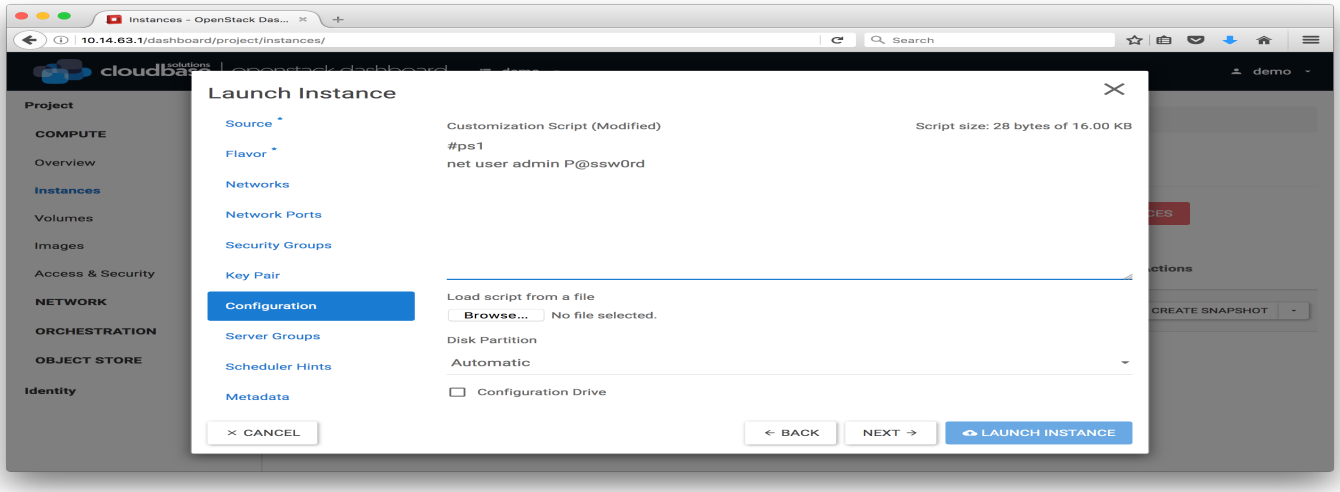
```
nova boot --meta admin_pass="<password>" ...
```

Given the previously mentioned security concerns this feature is disabled by default in Cloudbase-Init. In order to enable it *inject_user_password* must be set to *true* in the *cloudbase-init.conf* and *cloudbase-init-unattend.conf* config files*:*

```
inject_user_password = true
```

Password change in userdata script

The userdata can contain any PowerShell content (note the starting **#ps1** line to identify it as such), including commands for creating users or setting passwords, providing a much higher degree of flexibility. The same security concerns for clear text content apply as above.
The main limitation is that it does not work with Heat or other solutions that already employ the userdata content for other means.



(https://cloudbase.it/openstack-windows-admin-password/horizon_userdata_password-2/)

Passwordless authentication

Nova allows X509 keypairs to support passwordless authentication for Windows (http://www.cloudbase.it/windows-without-passwords-in-openstack). This is highly recommended as it does not require any password, similarly to SSH public key authentication on Linux. The limitations of this option is that it works only for remote PowerShell and WinRM and not for RDP.

The post Setting the Windows admin password in OpenStack (https://cloudbase.it/openstack-windows-admin-password/) appeared first on Cloudbase Solutions (https://cloudbase.it).

by Alessandro Pilotti at March 20, 2017 04:02 PM (https://cloudbase.it/openstack-windows-admin-password/)

RDO (http://rdoproject.org/blog/)

Blog posts, week of March 20 (http://rdoproject.org/blog/2017/03/blog-posts-week-of-march-20/)

Here's what the RDO community has been blogging about in the last week.

Joe Talerico and OpenStack Performance at the OpenStack PTG in Atlanta by Rich Bowen

> Last month at the OpenStack PTG in Atlanta, Joe Talerico spoke about his work on OpenStack Performance in the Ocata cycle.

Read more at http://rdoproject.org/blog/2017/03/joe-talerico-and-openstack-performance-at-the-openstack-ptg-in-atlanta/ (http://rdoproject.org/blog/2017/03/joe-talerico-and-openstack-performance-at-the-openstack-ptg-in-atlanta/)

RDO CI promotion pipelines in a nutshell by amoralej

> One of the key goals in RDO is to provide a set of well tested and up-to-date repositories that can be smoothly used by our users:

Read more at http://rdoproject.org/blog/2017/03/rdo-ci-in-a-nutshell/ (http://rdoproject.org/blog/2017/03/rdo-ci-in-a-nutshell/)

A tale of Tempest rpm with Installers by chandankumar

> Tempest is a set of integration tests to run against OpenStack Cloud. Delivering robust and working OpenStack cloud is always challenging. To make sure what we deliver in RDO is rock-solid, we use Tempest to perform a set of API and scenario tests against a running cloud using different installers like puppet-openstack-integration, packstack, and tripleo-quickstart. And, it is the story of how we integrated RDO Tempest RPM package with installers so it can be consumed by various CI rather than using raw upstream sources.

Read more at http://rdoproject.org/blog/2017/03/a-tale-of-tempest-rpm-with-installers/ (http://rdoproject.org/blog/2017/03/a-tale-of-tempest-rpm-with-installers/)

An even better Ansible reporting interface with ARA 0.12 by dmsimard

> Not even a month ago, I announced the release of ARA 0.11 with a bunch of new features and improvements.

Read more at https://dmsimard.com/2017/03/12/an-even-better-ansible-reporting-interface-with-ara-0-12/ (https://dmsimard.com/2017/03/12/an-even-better-ansible-reporting-interface-with-ara-0-12/)

Let rdopkg manage your RPM package by

> rdopkg is a RPM packaging automation tool which was written to efortlessly keep packages in sync with (fast moving) upstream.

Read more at http://rdoproject.org/blog/2017/03/let-rdopkg-manage-your-RPM-package/ (http://rdoproject.org/blog/2017/03/let-rdopkg-manage-your-RPM-package/)

Using Software Factory to manage Red Hat OpenStack Platform lifecycle by Maria Bracho, Senior Product Manager OpenStack

> by Nicolas Hicher, Senior Software Engineer – Continuous Integration and Delivery Software-Factory Software-Factory is a collection of services that provides a powerful platform to build software. It enables the same workflow used to develop OpenStack: using Gerrit for code reviews, Zuul/Nodepool/Jenkins as a CI system, and Storyboard for stories and issues tracker. Also, it ensures a reproducible test environment with ephemeral Jenkins slaves.

Read more at http://redhatstackblog.redhat.com/2017/03/08/using-software-factory-to-manage-red-hat-openstack-platform-lifecycle/ (http://redhatstackblog.redhat.com/2017/03/08/using-software-factory-to-manage-red-hat-openstack-platform-lifecycle/)

by Rich Bowen at March 20, 2017 01:53 PM (http://rdoproject.org/blog/2017/03/blog-posts-week-of-march-20/)

OpenStack Superuser (http://superuser.openstack.org)

Pay it forward: Sign up for Speed Mentoring at the OpenStack Summit Boston (http://superuser.openstack.org/articles/speed-mentoring-openstack/)

After a successful launch at the Austin Summit, Speed Mentoring is back in action in Boston.

Organized by the Women of OpenStack (https://wiki.openstack.org/wiki/Women_of_OpenStack), it's designed to be a lightweight mentoring initiative to provide technical or career guidance to beginners in the community. Mentees should already be part of the community; they should have gone through, or be familiar with Upstream Training (http://superuser.openstack.org/articles/openstack-upstream-training-revamp/).

Intel is sponsoring the Boston edition, which is now seeking 10-20 mentors (https://openstackfoundation.formstack.com/forms/boston_woo_speed_mentoring_baseballcard_info) interested in offering either technical or career advice.

Who should apply?

"We're happy for any mentor of any gender from a technical or non-technical background who has worked in OpenStack long enough to know their way around — at least one year of experience," Emily Hugenbruch (https://twitter.com/EKHugen), software engineer at IBM and driving force behind the project, tells Superuser. "They should be excited about OpenStack and interested in sharing their advice and expertise with others."

In terms of what mentees are seeking, she says they're looking to network, for advice on how to break into the community and how to advance their careers. "It's up to mentors if they want to allow mentees to contact them after the Speed Mentoring event, although we encourage it."

How does it work?

Mentors sign up here (https://openstackfoundation.formstack.com/forms/boston_woo_speed_mentoring_baseballcard_info), filling out a survey about their areas of interest and expertise. That info is turned into special baseball cards, which are given out at the event by way of introduction. Mentors meet with small groups of mentees in 15-minute intervals and answer their questions about how to grow in the community.

It's a fast-paced event and a great way to meet new people, introduce them to the Summit and welcome them into the OpenStack community, Hugenbruch says. Mentors are provided with mentee questions in advance and should plan to arrive at 7:15 a.m. (Breakfast and caffeine are provided!)

Mentors will be contacted ahead of time to go over logistics before the Summit. Hugenbruch (https://twitter.com/EKHugen) says a call is planned on April 17 (it'll be recorded for those who can't make it) so that serves as the deadline for mentor applications.

In the Austin edition (http://superuser.openstack.org/articles/women-of-openstack-mentoring-program-launches-at-austin-summit/), 150 people attended and 66 matches were made.

The post Pay it forward: Sign up for Speed Mentoring at the OpenStack Summit Boston (http://superuser.openstack.org/articles/speed-mentoring-openstack/) appeared first on OpenStack Superuser (http://superuser.openstack.org).

by Superuser at March 20, 2017 12:23 PM (http://superuser.openstack.org/articles/speed-mentoring-openstack/)

Cisco Cloud Blog (http://blogs.cisco.com)

Building A Cloud Community (http://blogs.cisco.com/cloud/building-a-cloud-community)

Six months ago, I inherited a stagnant OpenStack San Diego user group and its dozen orphaned members. I had discovered the benefits of working with OpenStack the previous year when a client asked me to develop a cyber security solution for its OpenStack powered cloud. OpenStack was a breath of fresh air after my experience with closed, proprietary public cloud environments. I was motivated to ensure other people in the industry know its benefits. To really get people excited about OpenStack, I needed to include hands-on experience; give people some “stick time” using OpenStack. This user group needed an OpenStack cloud for its (no longer) orphaned members.

by John Studarus at March 20, 2017 11:00 AM (http://blogs.cisco.com/cloud/building-a-cloud-community)

Cloudbau Blog (https://cloudbau.github.io)

IPVS direct routing on top of OpenStack (https://cloudbau.github.io/openstack/loadbalancing/networking/ipvs/2017/03/20/ipvs-direct-routing-on-top-of-openstack.html)

Motivation

March 20, 2017 10:00 AM (https://cloudbau.github.io/openstack/loadbalancing/networking/ipvs/2017/03/20/ipvs-direct-routing-on-top-of-openstack.html)

Opensource.com (https://opensource.com/taxonomy/term/5126/feed/feed)

Community leadership planning, new board members, and more OpenStack news (https://opensource.com/article/17/3/openstack-news-march-20)

Explore what's happening this week in OpenStack, the open source cloud computing project.

by Jason Baker at March 20, 2017 05:00 AM (https://opensource.com/article/17/3/openstack-news-march-20)

March 19, 2017

OpenStack Blog (https://www.openstack.org/blog)

OpenStack Developer Mailing List Digest March 11-17 (https://www.openstack.org/blog/2017/03/openstack-developer-mailing-list-digest-20170317/)

SuccessBot Says

- Dims [1] (http://eavesdrop.openstack.org/irclogs/#openstack-python3/#openstack-python3.2017-01-23.log.html): Nova now has a python35 based CI job in check queue running Tempest tests (everything running on py35)
- jaypipes [2] (http://eavesdrop.openstack.org/irclogs/#openstack-nova/#openstack-nova.2017-01-31.log.html): Finally got a good functional test created that stresses the Ironic and Nova integration and migration from Newton to Ocata.
- Lbragstad [3] (http://eavesdrop.openstack.org/irclogs/#openstack-keystone/#openstack-keystone.2017-02-09.log.html): the OpenStack-Ansible project has a test environment that automates rolling upgrade performance testing
- annegentle [4] (http://eavesdrop.openstack.org/irclogs/#openstack-dev/#openstack-dev.2017-02-16.log.html): Craig Sterrett and the App Dev Enablement WG: New links to more content for the appdev docs [5] (https://developer.openstack.org/)
- jlvillal [6] (http://eavesdrop.openstack.org/irclogs/#openstack-ironic/#openstack-ironic.2017-03-16.log.html): Ironic team completed the multi-node grenade CI job
- Tell us yours via OpenStack IRC channels with message “#success <message>”
- All: [7 (https://wiki.openstack.org/wiki/Successes)]

Pike Release Management Communication

- The release liaison is responsible for:
  - Coordinating with the release management team.
  - Validating your team release team requests.
  - Ensure release cycle deadlines are met.
  - It’s encouraged to nominate a release liaison. Otherwise this tasks falls back to the PTL.
- Ensure the releaase liaison has time and ability to handle the communication necessary.

    - Failing to follow through on a needed process step may block you from meeting deadlines or releasing as our milestones are date-based, not feature-based.
- Three primary communication tools:
    - Email for announcements and asynchronous communication
        - "[release]" topic tag on the openstack-dev mailing list.
        - This includes the weekly release countdown emails with details on focus, tasks, and upcoming dates.
    - IRC for time sensitive interactions
        - With more than 50 teams, the release team relies on your presence in the freenode #openstack-release channel.
    - Written documentation for relatively stable information
        - The release team has published the schedule for the Pike cycle [8] (http://releases.openstack.org/pike/schedule.html)
        - You can add the schedule to your own calendar [9] (https://releases.openstack.org/schedule.ics)
- Things to do right now:
    - Update your release liaisons [10] (https://wiki.openstack.org/wiki/CrossProjectLiaisons#Release_management).
    - Make sure your IRC and email address listed in projects.yaml [11] (http://git.openstack.org/cgit/openstack/governance/tree/reference/projects.yaml).
- Update your mail filters to look for "[release]" in the subject line.
- Full thread [12] (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113303.html)

OpenStack Summit Boston Schedule Now Live!

- Main conference schedule [13] (https://www.openstack.org/summit/boston-2017/summit-schedule/)
- Register now [14] (https://openstacksummit2017boston.eventbrite.com/)
- Hotel discount rates for attendees [15] (https://www.openstack.org/summit/boston-2017/travel/#hotels)
- Stackcity party [16] (https://www.openstack.org/summit/boston-2017/summit-schedule/events/18621/save-the-date-stackcity-boston)
- Take the certified OpenStack Administrator exam [17] (https://www.openstack.org/summit/boston-2017/#openstackcoa)
- City guide of restaurants and must see sites [18] (https://www.openstack.org/summit/boston-2017/boston-city-guide/)
- Full thread [19] (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113669.html)

Some Information About the Forum at the Summit in Boston

- "Forum" proper
    - 3 medium sized fishbowl rooms for cross-community discussions.
    - Selected and scheduled by a committee formed of TC and UC members, facilitated by the Foundation staff members.
    - Brainstorming for topics [20] (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113115.html)
- "On-boarding" rooms
    - Two rooms setup classroom style for projects teams and workgroups who want to on-board new team members.
    - Examples include providing introduction to your codebase for prospective new contributors.
    - These should not be tradiitonal "project intro" talks.
- Free hacking/meetup spaces
    - Four to five rooms populated with roundtables for ad-hoc discussions and hacking.
- Full thread [21] (http://lists.openstack.org/pipermail/openstack-dev/2017-March/thread.html#113459)

The Future of the App Catalog

- Created early 2015 as a market place of pre-packaged applications [22] (http://apps.openstack.org/) that you can deploy using Murano.
- This has grown to 45 Glance images, 13 Heat templates and 6 Tosca templates. Otherwise did not pick up a lot of steam.
- ~30% are just thin wrappers around Docker containers.
- Traffic stats show 100 visits per week, 75% of which only read the index page.
- In parallel, Docker developed a pretty successful containerized application marketplace (Docker Hub) with hundreds or thousands regularly updated apps.
    - Keeping the catalog around makes us look like we are unsuccessfully trying to compete with that ecosystem, while OpenStack is in fact complimentary.
- In the past, we have retired projects that were dead upstream.
    - The app catalog is however has an active maintenance team.
    - If we retire the app catalog, it would not be a reflection on that team performance, but that the beta was arguably not successful in build an active market place and a great fit from a strategy perspective.
- Two approaches for users today to deploy docker apps in OpenStack:
    - Container-native approach using "docker run" after using Nova or K8s cluster using Magnum.
    - OpenStack Native approach "zun create nginx".
- Full thread [23] (http://lists.openstack.org/pipermail/openstack-dev/2017-March/thread.html#113362)[24] (http://lists.openstack.org/pipermail/openstack-dev/2017-March/thread.html#113508)

ZooKeeper vs etcd for Tooz/DLM

- Devstack defaults to ZooKeeper and is opinionated about it.
- Lots of container related projects are using etcd [25] (http://codesearch.openstack.org/?q=etcd&i=nope&files=devstack/.*&repos=), so do we need to avoid both ZooKeeper and etcd?
- For things like databases and message queues, it's more than time for us to contract on one solution.
  - For DLMs ZooKeepers gives us mature/ featureful angle. Etcd covers the Kubernetes cooperation / non-java angle.
- OpenStack interacts with DLM's via the library Tooz. Tooz today only supports etcd v2, but v3 is planned which would support GRPC.
- The OpenStack gate will begin to default to etcd with Tooz.
- Full thread [26] (http://lists.openstack.org/pipermail/openstack-dev/2017-March/thread.html#113885)

Small Steps for Go

- An etherpad [27] (https://etherpad.openstack.org/p/go-and-containers) has been started to begin tackling the new language requirements [28] (https://governance.openstack.org/tc/reference/new-language-requirements.html) for Go.
- An golang-commons repository exists [29] (http://git.openstack.org/cgit/openstack/golang-commons/)
- Gopher cloud versus having a golang-client project is being discussed in the etherpad. Regardless we need support for os-client-config.
- Full thread [30] (http://lists.openstack.org/pipermail/openstack-dev/2017-March/thread.html#113463)

POST /api-wg/news

- Guidelines under review:
  - Add API capabilities discovery guideline [31] (https://review.openstack.org/#/c/386555/)
  - Refactor and re-validate API change guidelines [32] (https://review.openstack.org/#/c/421846/)
  - Microversions: add next_min_version field in version body [33] (https://review.openstack.org/#/c/446138/)
  - WIP: microversion architecture archival doc [34] (https://review.openstack.org/444892)
- Full thread [35] (http://lists.openstack.org/pipermail/openstack-dev/2017-March/114167.html)

Proposal to Rename Castellan to oslo.keymanager

- Castellan is a python abstraction to different keymanager solutions such as Barbican. Implementations like Vault could be supported, but currently is not.
- The rename would emphasize the Castellan is an abstraction layer.
  - Similar to oslo.db supporting MySQL and PostgreSQL.
- Instead of oslo.keymanager, it can be rolled into the oslo umbrella without a rename. Tooz sets the precedent of this.
- Full thread [36] (http://lists.openstack.org/pipermail/openstack-dev/2017-March/thread.html#113863)

Release Countdown for week R-23 and R-22

- Focus:
  - Specification approval and implementation for priority features for this cycle.
- Actions:
  - Teams should research how they can meet the Pike release goals [37] (https://governance.openstack.org/tc/goals/pike/index.html)[38] (https://governance.openstack.org/tc/goals/).
  - Teams that want to change their release model should do so before end of Pike-1 [39] (http://git.openstack.org/cgit/openstack/releases/tree/deliverables/pike).
- Upcoming Deadlines and Dates
  - Boston Forum topic formal submission period: March 20 – April 2
  - Pike-1 milestone: April 13 (R-20 week)
  - Forum at OpenStack Summit in Boston: May 8-11
- Full thread [40] (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113691.html)

Deployment Working Group

- Mission: To collaborate on best practices for deploying and configuring OpenStack in production environments.
- Examples:
  - OpenStack Ansible and Puppet OpenStack have been collaborating on Continuous Integration scenarios but also on Nova upgrades orchestration
  - TripleO and Kolla share the same tool for container builds.
  - TripleO and Fuel share the same Puppet OpenStack modules.
  - OpenStack and Kubernetes are interested in collaborating on configuration management.
  - Most of tools want to collect OpenStack parameters for configuration management in a common fashion.
- Wiki [41] (https://wiki.openstack.org/wiki/Deployment) has been started to document how the group will work together. Also an etherpad [42] (https://etherpad.openstack.org/p/deployment-pike) for brainstorming.

by Mike Perez at March 19, 2017 12:54 AM (https://www.openstack.org/blog/2017/03/openstack-developer-mailing-list-digest-20170317/)

March 17, 2017

RDO (http://rdoproject.org/blog/)

Joe Talerico and OpenStack Performance at the OpenStack PTG in Atlanta (http://rdoproject.org/blog/2017/03/joe-talerico-and-openstack-performance-at-the-openstack-ptg-in-atlanta/)

Last month at the OpenStack PTG (http://openstack.org/ptg) in Atlanta, Joe Talerico spoke about his work on OpenStack Performance in the Ocata cycle.

Subscribe to our YouTube channel (https://www.youtube.com/channel/UCWYIPZ4lm4P3_pzZ9Hx9awg) for more videos like this.

<iframe allowfullscreen="" frameborder="0" height="315" src="https://www.youtube.com/embed/8xIsPqAKeHs?list=PLOuHvpVx7kYksG0NFaCaQsSkrUlj3Oq4S" width="560"></iframe>
Joe: Hi, I'm Joe Talerico. I work on OpenStack at Red Hat, doing OpenStack performance. In Ocata, we're going to be looking at doing API and dataplane performance and performance CI. In Pike we're looking at doing mix/match workloads of Rally (https://wiki.openstack.org/wiki/Rally), Shaker (https://github.com/openstack/shaker), and perfkit benchmarker (http://googlecloudplatform.github.io/PerfKitBenchmarker/), and different styles, different workloads running concurrently. That's what we're looking forward to in Pike.

Rich: How long have you been working on this stuff?

Joe: OpenStack performance, probably right around four years now. I started with doing Spec Cloud development, and Spec Cloud development turned into doing performance work at Red Hat for OpenStack ... actually, it was Spec Virt, then Spec Cloud, then performance at OpenStack.

Rich: What kind of things were in Ocata that you find interesting?

Joe: In Ocata ... for us ... well, in Newton, composable roles, but building upon that, in TripleO, being able to do ... breaking out the control plane even further, being able to scale out our deployments to much larger clouds. In Ocata, we're looking to work with CNCF, and do a 500 node deployment, and then put OpenShift on top of that, and find some more potential performance issues, or performance gains, going from Newton to Ocata. We've done this previously with Newton, we're going to redo it with Ocata.

by Rich Bowen at March 17, 2017 06:59 PM (http://rdoproject.org/blog/2017/03/joe-talerico-and-openstack-performance-at-the-openstack-ptg-in-atlanta/)

OpenStack Superuser (http://superuser.openstack.org)

How a small team keeps Twitter's Fail Whale at bay (http://superuser.openstack.org/articles/twitter-fail-whale-containers-microservices/)

Following the strong wake created by the Fail Whale, Twitter created a life raft in the form of stateless containerized micro services. In just a few years, they've scaled to hundreds of teams running thousands of services on tens of thousands of hosts and in hundreds of thousands of containers.

Ian Downes (https://twitter.com/ndwns) is engineering manager for the compute platform team at Twitter. His team of about 10 engineers and a few other staffers buoys a platform providing container infrastructure to much of the stateless services powering twitter.com and its advertising business. Downes spoke recently at Container World (https://tmt.knect365.com/container-world/) on "Twitter's Micro Services Architecture: Operational & Technical Challenges."

When people talk about containerization, he says, it's often about how it can enable scale and disruption, but that doesn't interest Downes much.

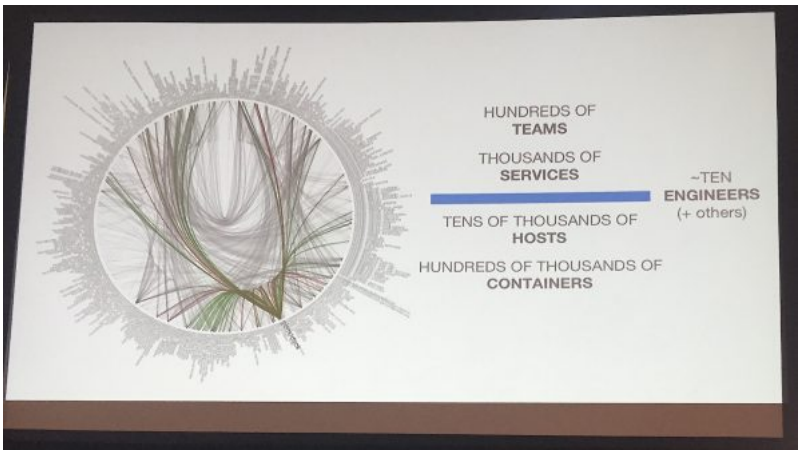
"What I'm more interested in are scaleable operations — independent of what scale you're at," he says. "What happens when you increase in size, number of machines, number of VMs, number of customers, the number of services you're running? If you double that, or quadruple, does your operational workload double or quadruple along with it, or does it basically stay the same?"

Thanks to concerted efforts in the last few years, Twitter has infrastructure that can scale very suddenly -doubling, if necessary – and, Downes says, has blown through several orders of magnitudes of growth without a corresponding operations burden.

> You might wonder where did the Fail Whale go? Truth to be told: It is failed to fail — now becomes a WIN Whale! pic.twitter.com/aWIG9HLN2y (http://t.co/aWIG9HLN2y)
>
> — Fail Whale (@failwhale) May 29, 2014 (https://twitter.com/failwhale/status/472003290180231168)

<script async="async" charset="utf-8" src="http://platform.twitter.com/widgets.js"></script>

Early on, Twitter was a monolithic application, made infamous by Fail Whale outages in 2010-2012. The social media company was towed under by events including a cascading bug, the Summer Olympics and the FIFA World Cup. They now run thousands of services for hundreds of teams inside the company, all done on their own infrastructure, their own machines, including tens of thousands of hosts and hundreds of thousands of containers.



(http://superuser.openstack.org/wp-content/uploads/2017/03/numbers.jpg)

"What's interesting, though, is that the platform that my team manages that infrastructure and provides that platform for those users with a team of about 10 engineers. That's a pretty amazing number."

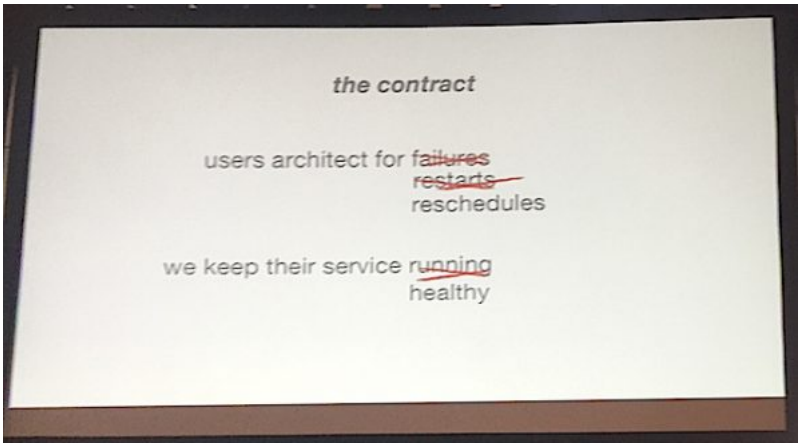
The journey from a monolithic application to micro services wasn't a simple one. Before coming to the common platform, each one of those micro services ran their own structure and each ops team had their own way of doing things. Downes likens it to cat herding — customers have habits, are opinionated and want to do things in a particular way. That meant keeping a ratio of 10:1 – 10 customers or 10 machines for every engineer.

"For a platform to be successful at the scale we intended, we had to view customers more like sheep," he says. The soft-spoken Downes underlines that the analogy isn't intended in a disparaging way, but simply that, like in his native New Zealand, a whole herd of sheep can be managed with two or three skilled sheepdogs. "It's all nice and smooth and very low fuss."

How they did it

It sounds like a tall order, but Downes says the solution is simple: a contract with customers that decouples availability and applications from operations. This contract asks users to architect for individual instances of their application being rescheduled and, in turn, his team promises to keep those services healthy. (A service may have 100 or 1,000 instances that are identical copies of the service, he adds.)

"Obviously, failures still happen," he says. "It doesn't matter whether you're running on virtualized infrastructure or the cloud. You have to architect for it, but that's not sufficient because it doesn't give any leeway for operations and we don't want them to architect and solve their problems in a different way."

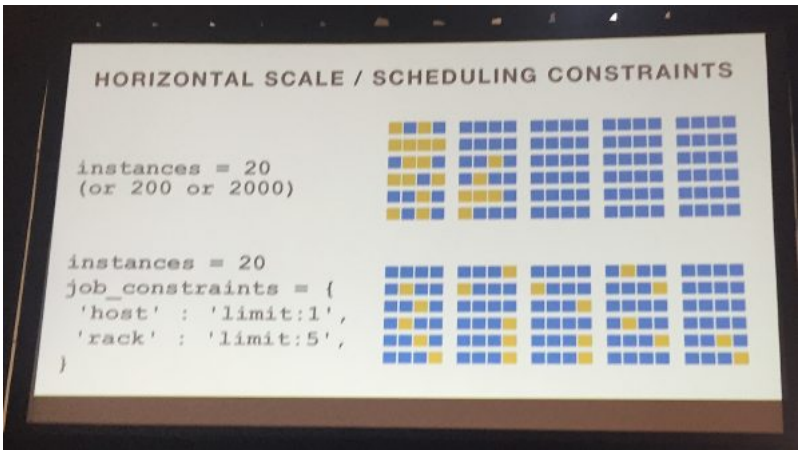


(http://superuser.openstack.org/wp-content/uploads/2017/03/contract.jpg)

At any point in time, the compute team can schedule instances of applications and move them around inside the cluster. Doing that will keep the service healthy – not running. "Running is not sufficient," Downes emphasizes. However, his team doesn't guarantee how many instances will run for each application. If they request to run 100 instances, they may be running fewer (due to failure or rescheduling) but the engineering team also doesn't guarantee that it won't exceed the number of instances, either.

"This seems a little strange, but it can happen with partitioning," he says. For example, when the agent installed on the host running the instance loses connectivity. Because the compute team doesn't know the state of those instances, they'll spin up additional ones somewhere else in the cluster, so an application could exceed the number of instances. In general, the number of instances running at any given time doesn't matter.

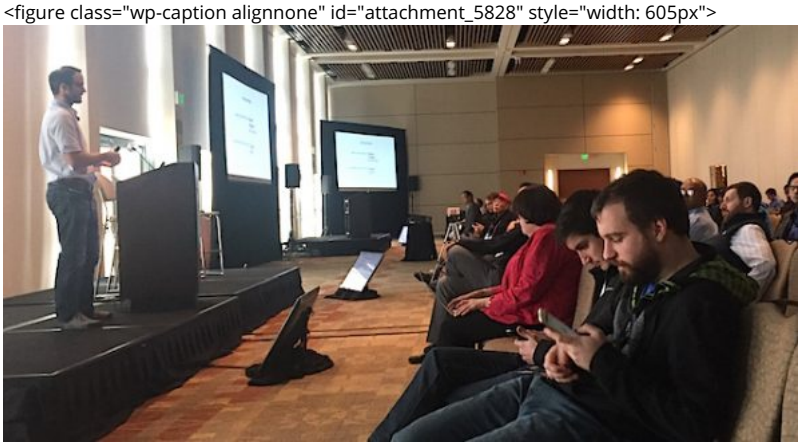
The compute team set a target of keeping 95 percent of instances healthy for at least 5 minutes as part of the contract. "We say 'we won't take out too much of the service at any given time and we won't do it too quickly.'



(http://superuser.openstack.org/wp-content/uploads/2017/03/IMG_6320-1.jpg)

To make it work for customers, there are a few caveats in place. They make it easy for customers to scale without fretting over instances — the more the better. But customers are also required to express how sensitive those instances are to different failure rates.

Downes says it's a way to chop up instances into groups that can be operated on. He offered an example of 20 distributed across five different racks. If the host goes down, they can lose one out of 20 (only five percent) if the rack goes down, they lose 25 percent of capacity (obviously a large fraction) so they're encouraged to scale way beyond 20..."It's good for them, it's good for failures and it's good for us." (In response to a follow-up question, Downes says Twitter uses Apache Aurora (https://aurora.apache.org/) Mesos (https://blog.twitter.com/2013/mesos-graduates-from-apache-incubation) but details were beyond the scope of his 20-minute talk.)

<figure class="wp-caption alignnone" id="attachment_5828" style="width: 605px">

(http://superuser.openstack.org/wp-content/uploads/2017/03/nsmail-2.jpg)<figcaption class="wp-caption-text">*Ian Downes of Twitter at Container World. Photo: Nicole Martinelli, OpenStack Foundation.*
</figcaption> </figure>
Every operation in the cluster is aware of the contract for each job. "So if we want to migrate something – we actually kill and then restart – we'll give it five minutes once it becomes healthy before we act on anything that might impact that job again." One host running 10 different containers, 10 different user loads have to keep track of across all the machines. "This means we can do actions on the cluster without impacting our customers."

The next question is whether it's sufficient to enable the compute team to manage the operational work load. The answer: Absolutely. Downes says they can roll an entire 30,000-node cluster running a full production workload — meaning they can reboot those nodes — in 24 hours with zero customer impact. While Downes calls that "aggressive" and not something they regularly do, they have done it for a kernel update and successfully, too. "There was no panic, no fires to put out, no alerts were triggered — we didn't get a single customer request."

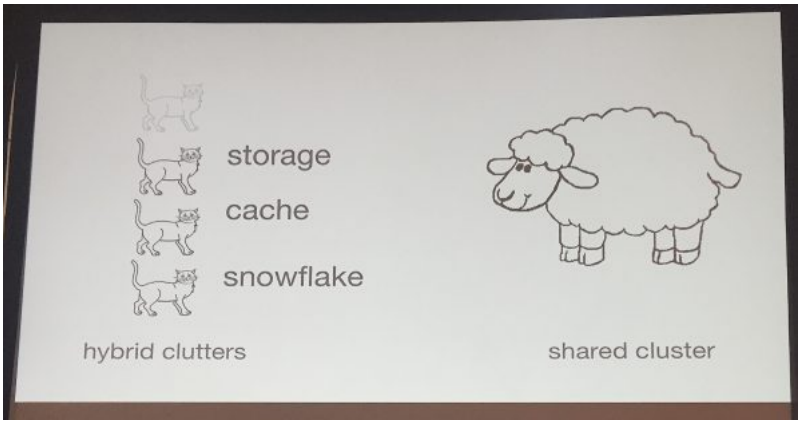

(http://superuser.openstack.org/wp-content/uploads/2017/03/8126547765_0d421a5658_z.jpg)

What's next: Extreme cat makeover

Downes says the team is exploring whether the contract concept can be taken further. "We see ourselves almost as a public cloud. Other teams inside the company provide machines to us. If those machines fail, we go over the wall and say, 'Can you give us a machine that's healthy?' It's very much like running in the public cloud in that way."

He admits that the timing is different — if a host fails and it's taken it offline they may not get a replacement for a day or two. "We are, in effect, customers. We tell those providers, 'We don't care if you need to act on our machines, if it's at the rack level, just go in and do it.'" For example, the operations team may want to take the top rack switch offline and do some maintenance on it — they're welcome to without any interference or notice from the compute team.

This means his team has decoupled itself from the infrastructure and decoupled users from the infrastructure. "It's incredibly powerful, we have a resilient platform that we can poke and prod and perform operations on and it doesn't affect our users."



(http://superuser.openstack.org/wp-content/uploads/2017/03/nsmail3.jpg)

The picture that he painted of a flock of sheep expertly herded by a few dogs isn't the whole picture, he admits. "The reality is a little different. We have a lot of sheep and a few cats in the mix as well." The bulk of Twitter's shared cluster is where the sheep run – the customers who accept that contract run their services there. On the side, running through the same scheduler etc., are the cats. These customers have special requirements — persistent storage services, special hardware requirements, churning, etc. — that don't fit into the contract but want to take advantage of the orchestration, containerization and tooling.

"The unfortunate thing is that they dominate our operation workload even though they're only about 15 percent of our cluster capacity," he says. "In theory, we had a hybrid contract, the 'cats' could bring machines into the hybrid cluster and we'll run them through our infrastructure but you need to maintain the host, update them, etc."

Those good intentions often go astray, however. And, despite the agreement, Downes and team end up maintaining them. That's where the yowling begins: "it's very painful to manage these machines," Downes says. "Our operations burden is dominated by these special cases."

Converting those cats into sheep is what they're working on now. It may entail extending that contract, loosening it — they'll only restart or reschedule some applications over a few days rather than five minutes. Another solution may be that stateful services have a "best effort restart" on the same host (so customers can reattach to the same storage) or delegating more work to internal teams.

"These are all questions that we're trying to answer," Downes (https://www.linkedin.com/in/ndwns) says. "We've been incredibly successful in scaling up the stateless infrastructure, the next question is whether can we take those existing customers already in the cluster that are causing us this burden and take on new customers while maintaining operational scalability."

Stay tuned.

Cover Photo (http://superuser.openstack.org/feed/%26quot%3Bhttps%3A//www.flickr.com/photos/tveskov/3387394098/in/photostream/%E2%80%9C) // CC BY NC (https://creativecommons.org/licenses/by-nc/2.0/)

Photo:
Photo (http://www.petful.com/) // CC BY NC (https://creativecommons.org/licenses/by-nc/2.0/)

The post How a small team keeps Twitter's Fail Whale at bay (http://superuser.openstack.org/articles/twitter-fail-whale-containers-microservices/) appeared first on OpenStack Superuser (http://superuser.openstack.org).

by Nicole Martinelli at March 17, 2017 12:04 PM (http://superuser.openstack.org/articles/twitter-fail-whale-containers-microservices/)

RDO (http://rdoproject.org/blog/)

An OpenStack Distribution

RDO CI promotion pipelines in a nutshell (http://rdoproject.org/blog/2017/03/rdo-ci-in-a-nutshell/)

One of the key goals in RDO is to provide a set of **well tested and up-to-date** repositories that can be smoothly used by our users:

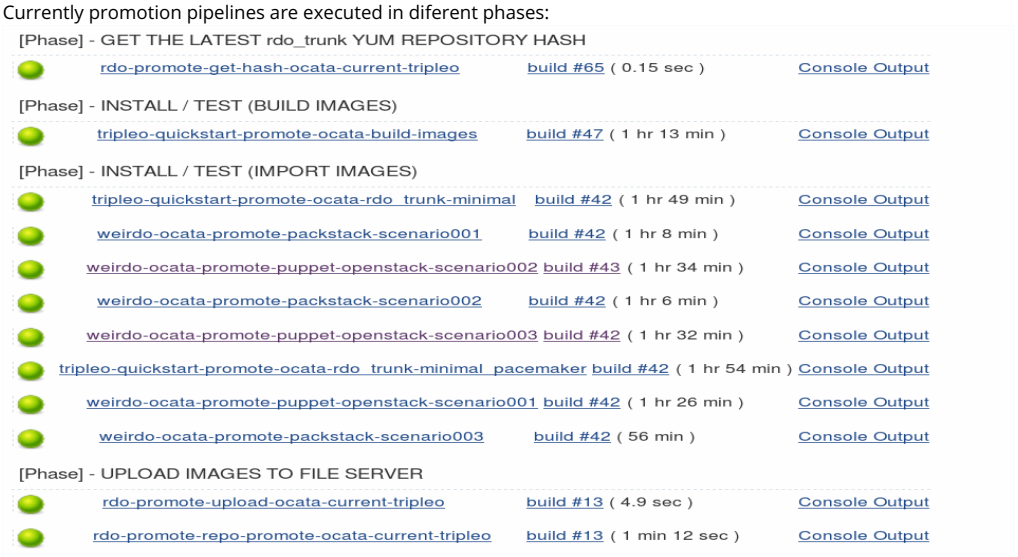
- Operators deploying OpenStack with any of the available tools.
- Upstream projects using RDO repos to develop and test their patches, as OpenStack puppet modules, TripleO or kolla.

To include new patches in RDO packages as soon as possible, in RDO Trunk repos (https://www.rdoproject.org/blog/2016/05/newbie-in-rdo-2-rdo-trunk-from-a-bird-s-eye-view/) we build and publish new packages when commits are merged in upstream repositories. To ensure the content of theses packages is trustworthy, we run different tests which helps us to identify any problem introduced by the changes committed.

This post provides an overview of how we test RDO repositories. If you are interested in collaborate with us in running an improving it, feel free to let us know in #rdo channel in freenode or rdo-list mailing list.

Promotion Pipelines

Promotion pipelines are composed by a set of related CI jobs that are executed for each supported OpenStack release to test the content of a specific RDO repository. Currently promotion pipelines are executed in diferent phases:



1. **Define the repository to be tested.** RDO Trunk repositories are identified by a hash based on the upstream commit of the last built package. The content of these repos doesn't change over time. When a promotion pipeline is launched, it grabs the latest **consistent** hash repo and sets it to be tested in following phases.
2. **Build TripleO images.** TripleO (https://www.rdoproject.org/tripleo/) is the recommended deployment tool for production usage in RDO and as such, is tested in RDO CI jobs. Before actually deploying OpenStack using TripleO the required images are built.
3. **Deploy and test RDO.** We run a set of jobs which deploy and test OpenStack using different installers and scenarios to ensure they behave as expected. Currently, following deployment tools and configurations are tested:
   - *TripleO deployments*. Using tripleo-quickstart (https://github.com/openstack/tripleo-quickstart) we deploy two different configurations, minimal (https://github.com/openstack/tripleo-quickstart/blob/master/config/general_config/minimal.yml) and minimal_pacemaker (https://github.com/openstack/tripleo-quickstart/blob/master/config/general_config/minimal_pacemaker.yml) which apply different settings that cover most common options.
   - *OpenStack Puppet scenarios*. Project puppet-openstack-integration (https://github.com/openstack/puppet-openstack-integration/) (a.k.a. p-o-i) maintains a set of puppet manifest to deploy different OpenStack services combinations and configurations (scenarios) in a single server using OpenStack puppet modules and run tempest smoke tests for the deployed services. The tested services on each scenario can be found in the README (https://github.com/openstack/puppet-openstack-integration/blob/master/README.md#description) for p-o-i. Scenarios 1, 2 and 3 are currently tested in RDO CI as
   - *Packstack deployment.* As part of the upstream testing, packstack defines three deployment scenarios (https://github.com/openstack/packstack/blob/master/README.md#packstack-integration-tests) to verify the correct behavior of the existing options. Note that tempest smoke tests are executed also in these jobs. In RDO-CI we leverage those scenarios to test new packages built in RDO repos.
4. **Repository and images promotion.** When all jobs in the previous phase succeed, the tested repository is considered good and it is promoted so that users can use these packages:
   - The repo is published using CentOS CDN in https://buildlogs.centos.org/centos/7/cloud/x86_64/rdo-trunk-<release>-tested/ (https://buildlogs.centos.org/centos/7/cloud/x86_64/rdo-trunk-ocata-tested/))
   - The images are copied to https://buildlogs.centos.org/centos/7/cloud/x86_64/tripleo_images/<release>/delorean/ (https://buildlogs.centos.org/centos/7/cloud/x86_64/tripleo_images/master/delorean/) to be used by TripleO users.

Tools used in RDO CI

- Jobs definitions are managed using *Jenkings Job Builder, JJB* (https://docs.openstack.org/infra/jenkins-job-builder/), via gerrit review workflow in review.rdoproject.org (https://review.rdoproject.org/r/#/q/project:rdo-infra/ci-config)
- *weirdo* (http://weirdo.readthedocs.io/en/latest/how.html) is the tool we use to run p-o-i and Packstack testing scenarios defined upstream inside RDO CI. It's composed of a set of ansible roles and playbooks that prepares the environment and deploy and test the installers using the testing scripts provided by the projects.
- *TripleO Quickstart* (https://docs.openstack.org/developer/tripleo-quickstart/) provides a set of scripts, ansible roles and pre-defined configurations to deploy an OpenStack cloud using TripleO (https://docs.openstack.org/developer/tripleo-docs/) in a simple and fully automated way.
- *ARA* (https://github.com/openstack/ara) is used to store and visualize the results of ansible playbook runs, making easier to analize and troubleshoot them.

Infrastructure

RDO is part of the CentOS Cloud Special Interest Group so we run promotion pipelines in CentOS CI (https://wiki.centos.org/QaWiki/CI) infrastructure where Jenkins is used as continuous integration server..

Handling issues in RDO CI

An important aspect of running RDO CI is managing properly the errors found in the jobs included in the promotion pipelines. The root cause of these issues sometimes is in the OpenStack upstream projects:

- Some problems are not catched in devstack-based jobs running in upstream gates.
- In some cases, new versions of OpenStack services require changes in the deployment tools (puppet modules, TripleO, etc...).

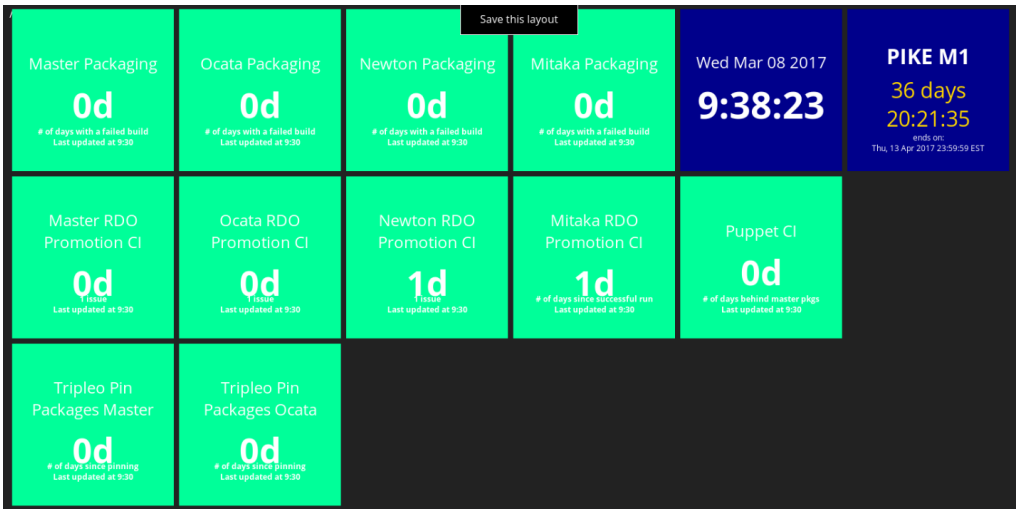
One of the contributions of RDO to upstream projects is to increase test coverage of the projects and help to identify the problems as soon as possible. When we find them we report it upstream as Launchpad bugs and propose fixes when possible.

Every time we find an issue, a new card is added to the TripleO and RDO CI Status Trello board (https://trello.com/b/WXJTwsuU/tripleo-and-rdo-ci-status) where we track the status and activities carried out to get it fixed.

Status of promotion pipelines

If you are interested in the status of the promotion pipelines in RDO CI you can check:

- *CentOS CI RDO view* (https://ci.centos.org/view/rdo/view/promotion-pipeline) can be used to see the result and status of the jobs for each OpenStack release.
- *RDO Dashboard* (https://dashboards.rdoproject.org/rdo-dev) shows the overal status of RDO packaging and CI.



More info

- TripleO quickstart demostration (https://www.youtube.com/watch?v=4O8KvC66eeU) by trown
- Weirdo: A talk about CI in OpenStack and RDO (https://dmsimard.com/2016/03/02/openstack-montreal-a-talk-about-ci-in-openstack-and-rdo/) by dmsimard.
- ARA blog posts (https://dmsimard.com/tag/ara.html) - from dmsimard blog
- Ci in RDO: What do we test? (https://amoralej.fedorapeople.org/slides/RDO-CI-summit-bcn-final.pdf) - presentation in RDO and Ceph Meetup BCN.

by amoralej at March 17, 2017 09:00 AM (http://rdoproject.org/blog/2017/03/rdo-ci-in-a-nutshell/)

March 16, 2017

NFVPE @ Red Hat (https://blog.nfvpe.site)

A (happy happy joy joy) ansible-container hello world! (http://dougbtv.com//nfvpe/2017/03/16/ansible-container/)

Today we're going to explore ansible-container (https://github.com/ansible/ansible-container), a project that gives you Ansible workflow for Docker. It provides a method of managing container images using ansible commands (so you can avoid a bunch of dirty bash-y Dockerfiles), and then provides a specification of "services" which is eerily similar (on purpose) to docker-compose. It also has paths forward for managing the instances of these containers on Kubernetes & OpenShift – that's pretty tight. We'll build two images "ren" and "stimpy", which contain nginx and output some Ren & Stimpy quotes so we can get a grip on how it's all put together. It's better than bad – it's good (https://www.youtube.com/watch?v=-fQGPZTECYs)!

by Doug Smith at March 16, 2017 01:50 PM (http://dougbtv.com//nfvpe/2017/03/16/ansible-container/)

OpenStack Superuser (http://superuser.openstack.org)

Exploring the OpenStack neighborhood: An offbeat install guide (http://superuser.openstack.org/articles/openstack-install-guide-beginner/)

We've all seen the plethora of installation, configuration, best practices and not-so-best practices books, guides, articles and blogs on OpenStack. This series will not be one of them.

I don't want to produce yet another installation guide. Rather, I want to share my experiences learning and understanding the OpenStack platform. (My fear is that, by the time I am done with these articles, they will sound like yet another guide, but I will try to make this series a bit more entertaining by adding my two cents where applicable.)

I believe that, sometimes, OpenStack is misunderstood. It's not complicated; however, like all magnanimous things, it is complex. So what do you do when you encounter something complex? You take a 180-degree turn and run like your life depended on it! Just kidding.

However, if you're like me and adore this nerdy misery in your life, you'll try to make some sense out of the complexity. You will never understand it in its entirety and attempting to do so is futile. Rather, you need to ask yourself what you want it, then learn more about those aspects accordingly.

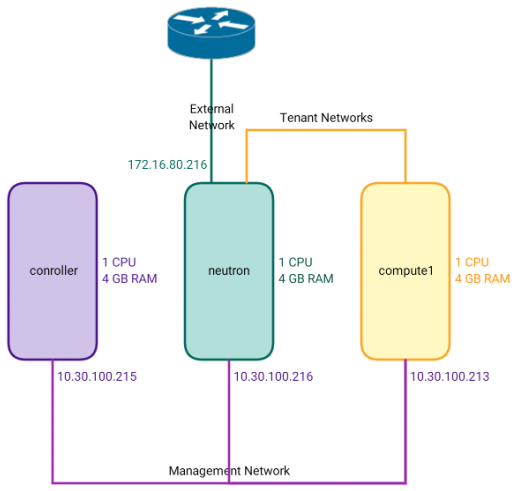
However, before we jump in, let me introduce the main character of this series, OpenStack. What is it? Simply put, it's a software platform that will allow you to embrace the goodies of the cloud revolution. In simpler terms, if you want to rent servers, VMs, containers, development platforms and middleware to customers while charging them for your services, OpenStack will allow you to do this. There are other, more complicated questions you may have, such as: Who are your customers? How are they paying? What are they getting? All of these will be answered in due time.

OpenStack has a great community that meets twice per year (http://openstack.org/summit) in wonderful locations across the globe, bringing the best brains of the business together to talk about technology and how it's changing the world.

Like anything important in life, if you want something and you're serious about it, you have got to define your goals and do the ground work. This platform is no exception. Right now, our goal is to perform a manual installation of OpenStack.

For the record: I think that all vendors that offer OpenStack are doing a wonderful job with their respective installers and, in a production environment, you may want to use these installers. Besides being automated and simple to setup, they also take care of high availability and upgrades among other things. But, if one day you have to troubleshoot an OpenStack environment (and believe me, you will), you will be thankful that you did the manual install in your lab to understand how things are configured. That's precisely our goal here.

So let's explore the neighborhood. Go through the diagram below:



OpenStack Base Environment

The above setup describes where OpenStack will reside in our setup. Below are the details:

Server 1	**controller**
Comment	*Here is where it keeps all the important stuff*
Server OS	Ubuntu 16.04
Resources	1 CPU, 4GB RAM
IP Addresses	10.30.100.215
Technical Brief	This is the main hub that hosts most of the fundamental services that form OpenStack
———————-	———————————————-
Server 2	**neutron**

Comment	*This is how it interacts with the outside world*
Server OS	Ubuntu 16.04
Resources	1 CPU, 4GB RAM
IP Addresses	10.30.100.216, 172.16.80.216
Technical Brief	This is the networking component that handles all IP based communications for components and hosted guests

Server 3	**compute1**
Comment	*This is where it entertains all its guests*
Server OS	Ubuntu 16.04
Resources	1 CPU, 4GB RAM
IP Addresses	10.30.100.213
Technical Brief	This is the server where you host the virtual machines that are rented to your customers.

Management Network (inner voice)	Used for all internal communication within OpenStack. When one OS component wants to talk to the other.
Tenant Networks (chatty guests)	Network used by customer virtual machines. When customers are rented virtual machines, they can also be given certain networks, in case they want to setup more than one machine and make them talk to each other. This network is used by such customer networks.
External Network (Lets call overseas)	Used by OS to talk to the outside world. Lets say your customer machine needs internet, then this is the network used by the respective virtual machines to access the internet.

On each of my three servers, the **/etc/hosts** file looks like this:

A. Host file configuration: /etc/hosts

```
# controller
10.30.100.215 controller

# compute1
10.30.100.213 compute1

# neutron
10.30.100.216 neutron
```

Note that I am assigning “controller”, “compute1” and “neutron” as aliases. If you change these, make sure to use the changed references when you use them in the configuration files later.

Also, networking seems to have given me some issues in the past so below is a sample **/etc/network/interfaces** file for each of the servers:

B. Interface file configuration: /etc/network/interfaces

From this point on, whenever you see “@SERVERNAME,” it simply implies that the configuration that I am talking about needs to be done on that server. So “@controller” means, “please perform these configurations/installations on the controller server.”

@controller

```
# This file describes the network interfaces available on your system
```

```
# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

# The management1 interface
auto ens3
iface ens3 inet static
address 10.30.100.215
netmask 255.255.255.0
```

@neutron

```
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

# The management1 interface
auto ens3
iface ens3 inet static
address 10.30.100.216
netmask 255.255.255.0

# The Tunnel interface
auto ens9
iface ens9 inet manual
up ifconfig ens9 up

# The external interface
auto ens10
iface ens10 inet static
address 172.16.80.216
netmask 255.255.255.0
gateway 172.16.80.254
dns-nameservers 8.8.8.8
```

@compute1

```
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

# The management1 interface
auto ens3
iface ens3 inet static
address 10.30.100.213
netmask 255.255.255.0

# The tunnel interface
auto ens9
iface ens9 inet manual
up ifconfig ens9 up
```

I'm not going into details for each interface, but there is one thing to keep in mind: the **tunnel interface** in both the Neutron and compute nodes will *not* have any IP since it is used as a trunk for multiple tenant (or customer) networks in OpenStack. Also if you are using VLAN tagging, make sure that this particular interface is untagged.

@controller

C. NTP configuration:

NTP is network time protocol. Since our protagonist (OpenStack) resides across three houses (servers), we need to make sure that the clocks in all these houses show the same time. What happens if they don't? Ever get off a 15-hour flight? You are jet lagged, completely out of sync with the local time, rendering you useless until you have rested and the body has readjusted. Something similar happens to distributed software working across systems with out of sync clocks. The symptoms are sometimes quite erratic and it is not always straight forward to pin point the problem. So setup NTP as follows:

Configure @controller as the master clock:

- Install the service:

```
sudo apt install chrony
```

*NOTE: using **sudo vi** means that I want you to edit the file. The indented lines that follow are what need to be edited in the file*

- Configure a global NTP server in the configuration file:

```
sudo vi /etc/chrony/chrony.conf
 server 0.asia.pool.ntp.org iburst
 allow 10.30.100.0/24
```

- Start the NTP service:

```
sudo service chrony restart
```

Configure @neutron and @compute1 to sync time with the controller:

Install the service:

```
sudo apt install chrony
```

Configure controller as the master NTP server in the configuration file:

```
sudo vi /etc/chrony/chrony.conf
  server controller iburst  #(Comment all other pool or server entries)
```

Start the NTP service:

```
sudo service chrony restart
```

D. OpenStack packages

Since we are performing the install on an Ubuntu Linux system, we need to add the corresponding repositories to get the OpenStack Software. For the record, I'm working with the Newton release of OpenStack.

On @controller, @neutron and @compute1 perform the following configuration:

```
sudo apt install software-properties-common
sudo add-apt-repository cloud-archive:newton

sudo apt update && sudo apt dist-upgrade
sudo reboot #(This seems to be a good idea to avoid surprises)
sudo apt install python-openstackclient #(Just the OS client tools)
```

E. Maria DB

Like most distributed systems, OS requires a database to store configuration and data. In our case, it resides on the controller and almost all components will access it. Do note that the base configuration of most components reside in configuration files, however what you create inside OS is saved in the database. Consider this the intellectual capital of the OS environment. Perform the following steps on the **@controller** node:

Install the software:

```
sudo apt install mariadb-server python-pymysql
```

Configure the database:

```
sudo vi /etc/mysql/mariadb.conf.d/99-openstack.cnf
  [mysqld]
  bind-address = 10.30.100.215
  #The bind-address needs to be set to the Management IP of the controller node.
  default-storage-engine = innodb
  innodb_file_per_table
  max_connections = 4096
  collation-server = utf8_general_ci
  character-set-server = utf8
```

Restart the service:

```
sudo service mysql restart
```

Secure the installation:

```
sudo mysql_secure_installation
```

(Answer the questions that follow with information relevant to your environment.)

F. RabbitMQ

By now, you must have realized that OpenStack, our main character, is unusual. All of its parts are not tightly knit together. Rather, it's composed of a number of autonomous/semi-autonomous modules that perform their respective functions to form the greater whole. However, in order to do their respective parts, they need to communicate effectively and efficiently. In certain cases, a centralized message queuing system, RabbitMQ (http://www.rabbitmq.com/), is used to pass messages between components within OpenStack. Please do note that certain components sit on more than one system and perform different functions in different situations; we will leave that for a later episode. For now, perform the following configuration on the **@controller** node:

Install the software:

```
sudo apt install rabbitmq-server
```

Create a master user. Replace “MINE_PASS” with your own password:

```
sudo rabbitmqctl add_user openstack MINE_PASS
```

Allow full permissions for the user created above:

```
sudo rabbitmqctl set_permissions openstack ".*" ".*" ".*"
```

G. Memcached:

Honestly, I’m not an expert at this one, but I do know that it makes loading the web pages for the application a bit faster (one way of giving OpenStack a better short-term memory, I suppose).

Install the software on the **@controller** node:

```
sudo apt install memcached python-memcache
```

Configure the IP for the controller management interface in the “conf” file:

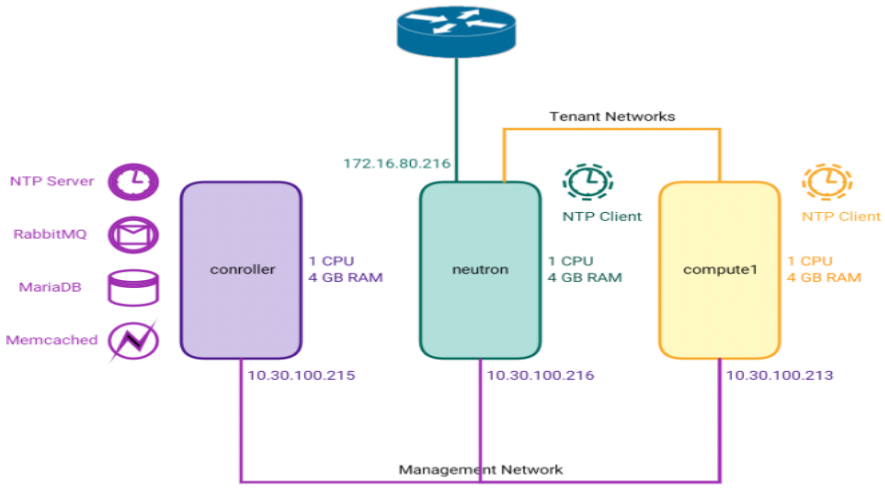
```
sudo vi /etc/memcached.conf
   -l 10.30.100.215
```

Start the service:

```
sudo service memcached restart
```

RECAP:

- You now know the central character’s name.
- You now know its whereabouts.
- You now know what things it needs to survive and how to set them up, if you wanted one for yourself.
- You are now EXCITED for the next episode of these tutorials! (hopefully.)



What we’ve achieved so far

If you’ve gotten this far, thanks for your patience. If you have any questions or comments, please comment below section so that everyone can benefit from the discussion.

This post first appeared on the WhatCloud (https://whatcloud.wordpress.com/2016/11/30/ose1tm/) blog. Superuser is always interested in community content, email: editor@superuser.org (mailto:editor@superuser.org).

Cover Photo (https://flic.kr/p/b4Gr4Z) // CC BY NC (https://creativecommons.org/licenses/by-nc/2.0/)

The post Exploring the OpenStack neighborhood: An offbeat install guide (http://superuser.openstack.org/articles/openstack-install-guide-beginner/) appeared first on OpenStack Superuser (http://superuser.openstack.org).

by Nooruddin Abbas at March 16, 2017 12:46 PM (http://superuser.openstack.org/articles/openstack-install-guide-beginner/)

March 15, 2017

OpenStack Superuser (http://superuser.openstack.org)

Community leadership charts course for OpenStack (http://superuser.openstack.org/articles/community-leadership-charts-course-openstack/)

Last week, about 40 people from the OpenStack Technical Committee, User Committee, Board of Directors and Foundation Staff convened in Boston to talk about the future of OpenStack. We candidly discussed the challenges we face as a community, but also why our mission to deliver open infrastructure is more important than ever.

To kick things off, Mark Collier (http://twitter.com/sparkycollier) opened with a state of the union address, talking about the strength of our community, the number of users running OpenStack at scale across various industries and the progress we've made working across adjacent open source projects. OpenStack is one of the largest, global open source communities. In 2016 alone, we had 3,479 unique developers from dozens of countries and hundreds of organizations contribute to OpenStack, and the number of merged changes increased 26 percent year-over-year. The size and diversity of the OpenStack community is a huge strength, but like any large organization, scale presents its own set of challenges.

Allison Randal (https://twitter.com/allisonrandal), who had done a lot of work beforehand to organize the strategy session, then laid out topical categories which present a unique set of challenges and opportunities for OpenStack. Each workshop participant was then challenged to define the first, most important action we should take to make progress in each category.

The five topical categories were: 1) how we communicate about 'What is OpenStack,' 2) unanswered requirements in OpenStack, 3) interacting with and supporting adjacent technologies, 4) changes to the technology and 5) community health. Throughout the day, we developed a plan of action for each category that will help focus community efforts and allow us to make significant progress over the next six months.

Six-month Community Roadmap TL;DR Version

Over the next six months, we'll be focusing community efforts on:

1. Better communicating and categorizing projects within the OpenStack landscape currently known as "the big tent" to help users understand what is OpenStack and the state of different projects
2. Bringing together developers/users/product teams at the Forum in Boston to improve our process for turning requirements into code
3. Making individual OpenStack projects like Cinder block storage or Keystone identity service easier to consume by adjacent technology communities like Kubernetes (breaking the perception that you must use all of OpenStack or nothing)
4. Simplifying the existing projects by reducing the number of supported configurations and options
5. Growing the next generation of community leaders and helping them rise up

Each action also has a specific owner, and will be fleshed out over the next few weeks so we can talk about progress in the April 11th board meeting. For now, I'll dive into more detail around each category, including the context, conversations in the room and different ideas for those who want to dig in.

How we communicate about 'What is OpenStack'

Today, any related open source project can add themselves to the OpenStack git, communication tools and infrastructure for testing. The general community has witnessed this through an explosion of new programs and innovative ideas. While such growth and interest is an immensely positive result, retaining a process to define trademark use, official projects, core capabilities and code requirements has challenged the basis of what is OpenStack

Over the last two years, there have been a series of changes in how we communicate about the projects that make up OpenStack. Previously, new projects would often start in Stackforge (https://docs.openstack.org/infra/system-config/stackforge.html) and once they wanted to become an official part of OpenStack, they would apply to the Technical Committee (https://www.openstack.org/foundation/tech-committee/) (TC) to become incubated until they met criteria to become part of the integrated release.

To solve growing pains, about two years ago the TC implemented two different policies: 1) adopted a new framework commonly referred to as "the big tent" (now a somewhat controversial name) and 2) stopped using the Stackforge branding. The combination of these changes essentially laid the groundwork for a two-tier model (official/unofficial projects) rather than three-tier model (Stackforge/incubated/integrated). The Interop Working Group (https://wiki.openstack.org/wiki/Governance/InteropWG) defines "core" as capabilities and code requirements with testing to validate commercial products, and the concept of the "integrated release" no longer exists. To provide more visibility into the state of official projects, the TC and User Committee (https://www.openstack.org/foundation/user-committee/) also define "tags" expressing varying states of maturity, development processes, etc.

Proposals to improve the current state included better communicating the value and position of different projects, defining “constellations” or deployment patterns consisting of groups of projects for different use cases (e.g. OpenStack for NFV), and better categorizing the existing OpenStack official projects. There was a lot of discussion about subjective versus objective judgments in how to achieve this goal. Ultimately, it was decided that better mapping projects within OpenStack is the first, most important step. Thierry Carrez (https://www.openstack.org/community/members/profile/154/thierry-carrez), chairman of the TC, will be spearheading these important efforts with a cross-community team of volunteers.

Adjacent Technologies

We’ve been talking as a community about building the LAMP stack of the cloud (http://superuser.openstack.org/articles/innovate-collaborate-replicate-openstack/), thinking of OpenStack as programmable infrastructure and recognizing the important technologies above, around and below it that people are combining for different use cases. How we better integrate and collaborate with these different technology communities was a key topic of conversation in Boston.

Proposals ranged from more focus on cross-community engagement, including upstream work and technical collaboration in adjacent communities to making sure we avoid “not-invented-here” syndrome (http://superuser.openstack.org/articles/eliminating-not-invented-here-syndrome/)and consume technologies outside of OpenStack. Ultimately, the group decided that cross-community work was critical and efforts were underway, but one of the first, most important things we need to do is make individual OpenStack services like Cinder block storage (http://superuser.openstack.org/articles/what-you-need-to-know-about-openstack-cinder/) and Keystone identity service (http://superuser.openstack.org/articles/common_openstack_deployments_book/) easily consumable on their own, alongside these other technologies. We need to change the mindset that you have to consume all of the common OpenStack services and demonstrate that each project is valuable on its own and will be combined with different technologies in unique and valuable ways. Chris Price (https://www.openstack.org/zh_CN/community/members/profile/18901/christopher-price), recently elected to the Board by the Gold Members as part of Ericsson and who also participates in OPNFV, will be coordinating these efforts. It was also one of the more popular teams for volunteers.

Unanswered Requirements

While we’ve done a great job building a community of users and operators who participate and contribute directly in OpenStack, optimizing the feedback loop for a project of this scale has been an ongoing challenge. We now have an elected User Committee that oversees 11 working groups, including a Product Working Group that helps create user stories and communicate the road map for key projects. However, the challenge discussed in the strategy workshop was bridging the user stories created by the Product Working Group to real blueprints (with applied resources) for the technical contributors, which require more in-depth gap analysis and community buy in.

Discussions ranged from how we prioritize requirements to how we reduce the number of new requirements and focus on refactoring / embracing adjacent technologies to focusing on scale, but the group ultimately decided to bring the primary stakeholders (User Committee/TC/Product WG) together at the Boston Summit Forum to collaborate/communicate around user stories, gap analysis, what fits in the current state of tech, prioritize what would have the greatest impact in reducing pain for users. Melvin Hillsman (https://www.openstack.org/community/members/profile/10659/melvin-hillsman), a newly elected member of the User Committee, will be wrangling this effort.

Changes to the Technology

Changes to the technology was added as the fifth category after we realized some of the proposals for change didn’t quite fit into ‘Communicating about OpenStack’ or ‘Unanswered Requirements.’ In order to address user feedback around complexity, proposed ideas in this category included culling official projects that may not be strategic or meet our quality standards, welcoming competing implementations within the OpenStack umbrella to enable greater change and innovation, converging the number of deployment tools (especially container-based deployment tools) and recording tribal knowledge. Ultimately, the group decided the first, most important action we need to take is simplifying existing OpenStack projects, including reducing the number of configuration options. Mike Perez (https://www.openstack.org/community/members/profile/4840/mike-perez), who works as a cross-project development coordinator at the OpenStack Foundation and is also an elected member of the TC, will be taking the lead on this effort working closely with the TC.

(http://superuser.openstack.org/wp-content/uploads/2017/03/whiteboard2.jpeg)

Cultivating Community Health

Our goal is to create a sustainable and productive community where diversity is valued and leadership opportunities are successful. There were several different proposals around improving community health, including on-boarding efforts, improvements to processes and tools and recognizing relevant contributions to adjacent communities and growing leaders in the community. The vote was very close between on-boarding and growing leadership, but it was generally recognized there are a number of on-boarding efforts like Upstream University already in place, while growing leadership was a new important focal point. Steven Dake (https://www.openstack.org/community/members/profile/360/steven-dake), who works at Cisco and is an individually elected member of the Board of Directors, volunteered to lead the group defining the next steps toward that goal.

Get Involved!

Throughout the day, there was lively participation and discussion about all of the topics. A broad consensus emerged that we're entering an exciting and important phase for OpenStack that presents new challenges but also huge opportunities. If you want to help drive the future of OpenStack in any of these areas, please email the Foundation mailing list (http://lists.openstack.org/cgi-bin/mailman/listinfo/foundation), contact one of the team leaders directly (their names are linked to their unique OpenStack profile in each section) or join the next open board meeting (https://wiki.openstack.org/wiki/Governance/Foundation#OpenStack_Board_of_Director_Meetings) on April 11.

The post Community leadership charts course for OpenStack (http://superuser.openstack.org/articles/community-leadership-charts-course-openstack/) appeared first on OpenStack Superuser (http://superuser.openstack.org).

by Lauren Sell at March 15, 2017 04:34 PM (http://superuser.openstack.org/articles/community-leadership-charts-course-openstack/)

RDO (http://rdoproject.org/blog/)

A tale of Tempest rpm with Installers (http://rdoproject.org/blog/2017/03/a-tale-of-tempest-rpm-with-installers/)

Tempest (https://docs.openstack.org/developer/tempest/overview.html) is a set of integration tests to run against OpenStack Cloud. Delivering robust and working OpenStack cloud is always challenging. To make sure what we deliver in RDO is rock-solid, we use Tempest to perform a set of API and scenario tests against a running cloud using different installers like puppet-openstack-integration (http://git.openstack.org/cgit/openstack/puppet-openstack-integration/), packstack (http://git.openstack.org/cgit/openstack/packstack), and tripleo-quickstart (http://git.openstack.org/cgit/openstack/tripleo-quickstart). And, it is the story of how we integrated RDO Tempest RPM package with installers so it can be consumed by various CI rather than using raw upstream sources.

And the story begins from here:

In RDO, we deliver Tempest as an rpm to be consumed by anyone to test their cloud. Till Newton release, we maintained a fork of Tempest (https://github.com/redhat-openstack/tempest) which contains the config_tempest.py script to auto generate tempest.conf for your cloud and a set of helper scripts to run Tempest tests as well as with some backports for each release. From Ocata, we have changed the source of Tempest rpm from forked Tempest to upstream Tempest by keeping the old source till Newton in RDO through rdoinfo (https://review.rdoproject.org/r/#/q/topic:switch-upstream-tempest). We are using rdo-patches branch to maintain patches backports starting from Ocata release.

With this change, we have moved the config_tempest.py script from the forked Tempest repository to a separate project python-tempestconf (https://github.com/redhat-openstack/python-tempestconf) so that it can be used with vanilla Tempest to generate Tempest config automatically.

What have we done to make a happy integration between Tempest rpm and the installers?

Currently, puppet-openstack-integration, packstack, and tripleo-quickstart heavily use RDO packages. So using Tempest rpm with these installers will be the best match. Before starting the integration, we need to make the initial ground ready. Till Newton release, all these installers are using Tempest from source in their respective CI. We have started the match making of Tempest rpm with installers. puppet-openstack-integration and packstack consume puppet-modules. So in order to consume Tempest rpm, first I need to fix the puppet-tempest (http://git.openstack.org/cgit/openstack/puppet-tempest).

puppet-tempest (http://git.openstack.org/cgit/openstack/puppet-tempest)

It is a puppet-module to install and configure Tempest and openstack-services Tempest plugins based on the services available from source as well as packages. So we have fixed puppet-tempest to install Tempest rpm from the package and created a Tempest workspace. In order to use that feature through puppet-tempest module [https://review.openstack.org/#/c/425085/]. you need to add *install_from_source => 'false'* and *tempest_workspace => 'path to tempest workspace'* to tempest.pp and it will do the job for you. Now we are using the same feature in puppet-openstack-integration and packstack.

puppet-openstack-integration (http://git.openstack.org/cgit/openstack/puppet-openstack-integration/)

It is a collection of scripts and manifests for puppet module testing (which powers the openstack-puppet CI). From Ocata release, we have added a flag TEMPEST_FROM_SOURCE flag in run_tests.sh script (http://git.openstack.org/cgit/openstack/puppet-openstack-integration/tree/run_tests.sh). Just change TEMPEST_FROM_SOURCE to false in the run_test.sh (https://review.openstack.org/#/c/427578/), Tempest is then installed and configured from packages using puppet-tempest.

packstack (http://git.openstack.org/cgit/openstack/packstack)

It is a utility to install OpenStack on CentOS, Red Hat Enterprise Linux or other derivatives in proof of concept (PoC) environments. Till Newton, Tempest is installed and ran by packstack from the upstream source and behind the scenes, puppet-tempest does the job for us. From Ocata, we have replaced this feature by using Tempest RDO package (https://review.openstack.org/#/c/428102/). You can use this feature by running the following command:

```
$ sudo packstack --allinone --config-provision-tempest=y --run-tempest=y
```

It will perform packstack all in one installation and after that, it will install and configure Tempest and run smoke tests on deployed cloud. We are using the same in RDO CI.

tripleo-quickstart (https://docs.openstack.org/developer/tripleo-quickstart/)

It is an ansible based project for setting up TripleO virtual environments. It uses triple-quickstart-extras (http://git.openstack.org/cgit/openstack/tripleo-quickstart-extras/tree/roles/validate-tempest) where validate-tempest roles exist which is used to install, configure and run Tempest on a tripleo deployment after installation. We have improved the validate-tempest role to use Tempest rpm package for all releases (supported by OpenStack upstream) by keeping the old workflow and as well as using Ocata Tempest rpm and using ostestr for running Tempest tests for all releases and using python-tempestconf to generate tempest.conf through this patch (https://review.openstack.org/#/c/431916/).

To see in action, Run the following command:

```
$ wget https://raw.githubusercontent.com/openstack/tripleo-quickstart/master/quickstart.sh
$ bash quickstart.sh --install-deps
$ bash quickstart.sh -R master --tags all $VIRTHOST
```

So finally the integration of Tempest rpm with installers is finally done and they are happily consumed in different CI and this will help to test and produce more robust OpenStack cloud in RDO as well as catch issues of Tempest with Tempest plugins early.

Thanks to apevec, jpena, amoralej, Haikel, dmsimard, dmellado, tosky, mkopec, arxcruz, sshnaidm, mwhahaha, EmilienM and many more on #rdo channel for getting this work done in last 2 and half months. It was a great learning experience.

by chandankumar at March 15, 2017 11:41 AM (http://rdoproject.org/blog/2017/03/a-tale-of-tempest-rpm-with-installers/)

Cisco Cloud Blog (http://blogs.cisco.com)

Why Millennials Don't Think In Boxes (http://blogs.cisco.com/cloud/why-millennials-dont-think-in-boxes)

I was born in the mid-80s. I started breaking down computers quite early and I had PSTN internet by the time I was 16 (having frequent arguments with my parents when they regularly disconnected me in order to hold meaningless important conversations with other members of our extended Greek family).

by Kostas Roungeris at March 15, 2017 11:00 AM (http://blogs.cisco.com/cloud/why-millennials-dont-think-in-boxes)

SUSE Conversations (https://www.suse.com/communities/blog)

OpenStack Private Cloud is Doing Just Fine (https://www.suse.com/communities/blog/openstack-private-cloud-just-fine/)

Sometimes you have to dig beneath the surface of headlines to understand what's really going on. I guess that's one thing most of us have learned in recent months. Sometimes you need some careful analysis to get to the real story. It seems like that's as true in the IT world as it is with tabloid ...

+read more (https://www.suse.com/communities/blog/openstack-private-cloud-just-fine/)

The post OpenStack Private Cloud is Doing Just Fine (https://www.suse.com/communities/blog/openstack-private-cloud-just-fine/) appeared first on SUSE Blog (https://www.suse.com/communities/blog). Mark_Smith (https://www.suse.com/communities/blog/author/mark_smith/)

by Mark_Smith at March 15, 2017 09:50 AM (https://www.suse.com/communities/blog/openstack-private-cloud-just-fine/)

March 14, 2017

StackHPC Team Blog (https://www.stackhpc.com/)

Logging Services for Guest Workloads: A Step Closer (https://www.stackhpc.com/monasca-log-api.html)

How can we make a workload easier on cloud? In a previous article we presented the lay of the land (https://www.stackhpc.com/openstack-and-hpc-workloads.html) for HPC workload management in an OpenStack environment. A substantial part of the work done to date focuses on automating the creation of a software-defined workload management environment - *SLURM-as-a-Service*. The projects that look at enriching the environment available to workload management services once they are up and running in the cloud appear to be less common.

One example that came along last week was the merge upstream of a new spec for multi-tenant log retrieval in Monasca (https://review.openstack.org/#/c/433016/). This proposal was made and seen through by StackHPC's Steve Simpson.

Monasca and Multi-Tenant Monitoring

Monasca monitors OpenStack, but it goes further than that.

From its inception, Monasca has been designed with the distinction of supporting multi-tenant telemetry. Any tenant host, service or workload can submit telemetry data to a Monasca API endpoint, and have it collected and salted away. Later, the user can log in to a dashboard (Grafana in many cases), and interactively explore the telemetry data that they collected about the operation of their instances.

Can your tenants do that?

The intention is that complex services like telemetry and monitoring are provided as a service, without requiring the users to create and deploy their own.

Adding Logging to the Mix

Time-series telemetry is certainly useful, but is only one part of a comprehensive solution. We also want to gather data on events that occur, and logs of activity from the services and operating systems that underpin our research computing platforms.

The Monasca project (led by the team from Fujitsu) have been working on logging support for a little while. They first presented their work at the Tokyo summit:

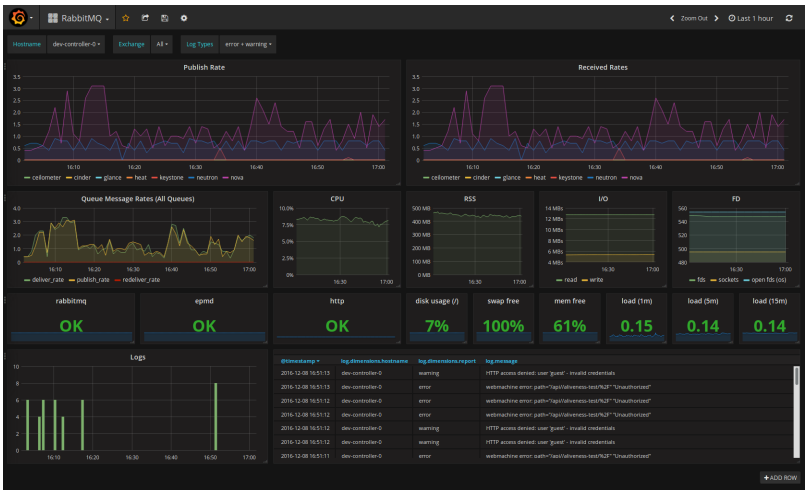
<iframe allowfullscreen="allowfullscreen" frameborder="0" height="500" seamless="seamless" src="https://www.youtube.com/embed/ghZ5gnySlWo" width="750"> </iframe>
Logging for system and OpenStack services has been up and running in Monasca for a few releases.

What has been missing (until now) has been a way of providing multi-tenant access to log retrieval.

Reducing the Time to Science

It's clear our users have work to do, and our OpenStack projects exist to support that.

Using Monasca, we can already present log data inline with telemetry data for system administration use cases. For example, here's log and telemetry data collected from monitoring RabbitMQ services, drawn from Monasca and presented together on a Grafana dashboard:



Once the new multi-tenant logging API is implemented, we'll be providing our users with the same services for telemetry and logging of their own infrastructure, platforms and workloads.

Further Reading

- Monasca Logging Wiki (https://wiki.openstack.org/wiki/Monasca/Logging)
- OpenStack and HPC Workload Management, StackHPC blog (http://www.stackhpc.com/openstack-and-hpc-workloads.html)
- OpenStack for Scientific Research (https://www.openstack.org/science/)

by Steve Simpson at March 14, 2017 12:29 PM (https://www.stackhpc.com/monasca-log-api.html)

OpenStack Superuser (http://superuser.openstack.org)

Carrots and sticks in an open source community (http://superuser.openstack.org/articles/leadership-open-source-community/)

Leading a community group, it turns out, is completely different than being a manager. I learned this the hard way; I made mistakes, started (and continued) arguments, and sometimes, when faced with a large hole, simply continued to dig.

In late 2014 I had been working on OpenStack documentation (https://docs.openstack.org/) for about a year. We were preparing the Kilo release (http://superuser.openstack.org/articles/navigating-openstack-community-release-cycles-events/), when the current project team lead (PTL) asked if I would be willing to put my name forward as a candidate for Liberty PTL. In early 2015, I was elected unopposed to lead the documentation team for the Liberty release and all of a sudden I realized: I had no idea how to run a community group.

At this point, I had managed docs teams of various sizes, across many time zones, for five years. I had a business management degree and an MBA to my name, had run my own business, seen a tech startup fail and a new corporate docs team flourish. I felt as though I understood what being a manager was all about. And I guess I did. But I didn't know what being a PTL was all about. All of a sudden, I had a team where I couldn't name each individual, couldn't rely on any one person to come to work on any given day, couldn't delegate tasks with any authority and couldn't compensate team members for good work. The only tool I had in my arsenal to get work done was my own ability to convince people that they should.

If you've spent time as a manager in a corporate environment, you'll be used to keeping secrets, actively managing your employees' careers and pretending you know the answers to things (especially if you don't). This is entirely the wrong way to go about managing a community and if your team decides you're treating them like that, they will probably only give you a couple of chances before they eat you alive. If you get it right (even if it's on the second go-round), the opportunities for growth and satisfaction are unbeatable. Not to mention the sense of achievement!

My first release as a PTL was basically me stumbling around in the dark and poking at the things I encountered. I relied heavily on the expertise of the existing members of the group to work out what needed to be done and I gradually started to learn that the key to getting things done was not just to talk and delegate, but to listen and collaborate. I had to not only tell people what to do, but also convince them that it was a good idea, help them to see it through, and pick up the pieces if they didn't. You will end up stumbling around in front of your team a lot in open source. Here's how to make it work for you:


(http://superuser.openstack.org/wp-content/uploads/2017/03/405782341_226672923b_z.jpg)

Rewarding the good, punishing the bad

The phrase 'carrots and sticks' is often used to describe the process of rewarding good behavior (carrots) while punishing bad behavior (sticks). Rewards are everywhere in today's corporations. From monetary incentives (bonuses, share schemes, gift cards and shopping vouchers), to various team activities (lunches, dinners, paintball, barista courses, and all manner of competitive activities), these awards are designed to make all your colleagues envy and despise you. Not to mention all those little branded tchotchkes (a stress ball in every imaginable shape!) we all seem to accumulate. There are so many carrots given out, that not getting any can often be a form of stick.

Of course, this is the first and most obvious thing you will notice missing when you start managing a community rather than a team in an organization. You don't get to pay them. You don't get to give bonuses. And the tchotchkes are harder to find and more jealously guarded.

The first thing you have to do is work out what you **have** got. Got access to stickers, t-shirts, or other branded merchandise? Give them out! What about privileges (and no, giving someone more responsibility is not a privilege, so don't count things like granting core access)? Things like discounted tickets or travel to conferences, access to company-sponsored events like code sprints or meetups, or things like the ability to vote on leadership positions can all be considered perks of being part of a community.

> Most of the people in the your community already have a job and a career path and a manager who (hopefully) cares about those things. And that person isn't you.

There are also other, less obvious, things that you can do to motivate people: always be willing to call out great work (or even mediocre work, especially if it came from someone surprising) in a public way, but keep any criticism private. Thank people, a lot, for everything. Make sure when people email you asking questions you add other people into the conversation, with a note like "this person is an expert on this topic, and I'd love to hear their opinion." Flattery and thankfulness are some of the best tools you have to motivate people on your team, and they're entirely free. Just try and keep your ego in check and let other people take credit wherever possible.

(http://superuser.openstack.org/wp-content/uploads/2017/03/7482639284_acae94d487_b.jpg)

Performance measurement as a behavior management tool

One of the more popular performance measurement methods is referred to as a nine box, or a talent matrix (http://www.employee-performance.com/pdf/Build-your-own-nine-box.pdf). The idea of ranking employees can be quite distasteful, so it's important to remember that it's not about ranking staff **against** each other, but against themselves and their own past performance. You should be able to see each employee move from the lower left to the top right of the matrix as they improve

in their role. Once they hit the top right box, you must be ready to promote them, at which point they drop to the center of the matrix, and start the journey up and to the right again. Of course, the opposite is also true: if an employee starts to track down and to the left, you need to be having some serious discussions about the role that the person is in, whether or not they're having personal issues, or whether they're the right fit for your team, or your company. The point is, there's something of a science to this when you're in a corporate environment.

That science becomes much more of a dark art when you're leading a community. For starters, though, you need to remember that it's not really your responsibility. Most of the people in the your community already have a job, and a career path, and a manager who (hopefully) cares about those things. And that person isn't you. Also, those things are (and should be) fairly opaque to you, it's really none of your business.

But that doesn't mean that you shouldn't care about your team's performance within the constraints of your group. Most communities have levels of responsibility within them. From leading sub-teams, to being involved in testing, sitting on advisory boards, or becoming core contributors with greater levels of trust and expectation, right up to leading the group itself, you need to be aware of the aspirations of your community members, and ensure you're letting people know the options available to them should they wish to progress. An extra added complication is where these two things intersect. You never know if a team member is getting pressure from their corporate manager to achieve a certain role within your community, or what value (if any) companies might place on metrics, roles, or positions of trust within your community. You can't always rely on team members to tell you what they're trying to achieve, either, sometimes they make you guess.

The best way to ensure you're helping individuals succeed in their performance goals is to make sure you understand what those goals are. This isn't always easy and you can't necessarily assume that all team members **want** to progress, either. This is even more true in communities than in companies, since a promotion often means more responsibility without any increase in benefits (you're not paying them, after all). The best way to do this is to ask people, and the best way to get a good answer is to do in private. Don't be afraid, as a community leader, to reach out to people directly, saying something along the lines of 'hey, I noticed you're doing great work, and wanted to have a chat to you about the kinds of things you're interested in working on, and how I can help you achieve your goals within our community.' You won't always get all the answers you might want, in the detail you might need to really help them, but at least you'll have a better idea than just assuming everyone wants to become a team leader some day.

The performance art of trust

Perhaps more important than rewarding staff, or accurately recording their performance improvements, is proving that you trust them. A little trust goes a long way and can positively impact loyalty, morale and retention. As a manager, there are many occasions where you are privy to information you can't share with staff: financial or company performance information, restructures, or even layoffs. The thing is, your team members are probably pretty smart and they probably know that this is a thing. You need to reassure them that as soon as you can tell them, you will. But there's more to it than that; there's something I like to call the performance art of trust, which is more about telling them when you **don't** know something than when you do. If someone asks you a question, and you don't know the answer, don't make things up! Just come right out and say it: "I don't know." You might find it helpful to practice, because it's not easy to do when you're trying to be all manager-y and stuff. In fact, most of your training to become a manager has probably been about pretending you know the answers when you don't, which is exactly the wrong way to go about it.

Of course, as a community leader, you almost never have access to information before anyone else, so you may be wondering why this is relevant. It's because the principles of honesty and trust are just as important, if not more so, in community situations than they are in a workplace. Community members will work with you if they **want** to work with you, and they won't work with you if you're a twit. The best way to look like you're someone worth working with is if you appear open, honest, communicative, trustworthy, and reasonable.

One of the hallmarks of many open-source communities is the somewhat impassioned email communications that occur. Flame wars on mailing lists are a feature of open source communities and they happen out in public. Treat them as performance art, where your audience is focused on one thing: is this person someone I trust to lead this group?

It's easy to come out of a flame war looking like a buffoon, so here's a couple of tips:

- Always read the email you're replying to thoroughly, several times, and try to understand the sender's point of view. Work out what the question is (if there isn't a question, then don't send a reply).
- Start your reply by thanking the sender: for bringing up a point you hadn't considered, for asking questions, or even just for taking the time to put their thoughts into words. This forces you to assume good intent.
- Answer the question, AND ONLY THE QUESTION. Give reasons for your answer, using dot points if necessary, but don't bring other issues into it, and certainly don't launch into personal attacks. Be prepared to question your own beliefs about things (you don't have to change your mind, but you need to be open to other perspectives).
- Ask a question of your own: Do they think this is reasonable? Can they think of something you might have missed? Do they have further comments?
- Don't hit send yet. Leave it as long as possible; at least a couple of hours, preferably overnight. This not only gives you time to calm down a little, but it also slows down the exchange on the mailing list, which will hopefully remove some heat.

- Before you hit send, proofread, and take out all the emotional language. Work out if you can consolidate some points, reorganise the content to be clearer, or take out irrelevant information. Be as concise and to-the-point as possible.
- If the thread has been dragging on and there's no progress, take it offline. Admit that perhaps you don't fully understand their viewpoint and offer a video or phone call, even if it's an early morning or late night for you. Be the bigger person, take the hit. People are always much nicer on the phone, and if nothing else, it stops the flamewar.

To conclude, communities are definitely more forgiving than corporations, so you'll probably get away with stumbling around a little bit before you find your feet. However, they're also run almost entirely on trust and goodwill and you can erode that really quickly and sometimes without noticing. Be honest when you don't know something, own up to your mistakes, never shift blame (even if you really didn't do it), and always lift your team members up. If you can nail those things, then you'll find a way through, and I'm willing to bet that you'll become a better manager in the process as well.

Superuser is always interested in community content, get in touch at editorATsuperuser.org

Cover Photo (https://www.flickr.com/photos/clement127/1) // CC BY NC (https://creativecommons.org/licenses/by-nc/2.0/)

Photo (https://www.flickr.com/photos/71007218@N04/22434721265/in/photolist-kVSHr-chpcGu-pLCWqj-cyXgpq-qNkcms-qnpBEq-hwCZ8b-bVhZd6-cGa4hu-dbpuH2-bPnFD4-c6FLcA-AbtP24-m67QAi-m67Z3i-qYGtML-m67fJT-Eq5eGq-oHRgs3-m67NAr-9u3mVn-rLanDS-8zf5w7-Axb3fr-6gK8Qq-doz9rV-nfDVu8-9PMAay-QZj1ji-991KPd-4QtLKB-4Qtykr-AxbcAk-B3setQ-4JhH1D-GUMNWA-H5bAoy-BuAdJM-w1r3Ex-GGW9Wj-kskWGJ) // CC BY NC (https://creativecommons.org/licenses/by-nc/2.0/)

Photo (https://www.flickr.com/photos/audreyjm529/405782341/in/photolist-ncEj45-ncEoyA-BRJXg-7nv5EH-cn5hwo-h9w4RM-4jKWps-esWgF-6i3AMf-tCRXi-35rfXk-4m8vfD-6mSsAF-6mWwzb-9xC2fJ-7fP3iu-6mSrkB-8QR6EE-6mWtJq-6mSqUD-7fNWNC-65L1bv-fefVox-7fJZU4-vmSZQ-pnW3L1-o3jPDp-4Dp7Ke-4fybMe-6cDUA2-6mWAaQ-7fJZFi-7fNWeQ-8SqHRi-qsPh26-kv1J9-7fJZsk-6cJ4oq-2C2c3z-7fNVm3-7fNWub-6cDU5n-6DXQkD-k8zD2-kGfbq-Nry7G-4LhQ4d-7jRQKX-4Z1ucX-4UumNS) // CC BY NC (https://creativecommons.org/licenses/by-nc/2.0/)

Photo (https://www.flickr.com/photos/legojunkie/7482639284/) // CC BY NC (https://creativecommons.org/licenses/by-nc/2.0/)

The post Carrots and sticks in an open source community (http://superuser.openstack.org/articles/leadership-open-source-community/) appeared first on OpenStack Superuser (http://superuser.openstack.org).

by Lana Brindley at March 14, 2017 12:20 PM (http://superuser.openstack.org/articles/leadership-open-source-community/)

Mark McLoughlin (https://crustyblaa.com/)

March 9, 2017 OpenStack Foundation Board Meeting (https://crustyblaa.com/march-9-2017-openstack-foundation-board-meeting.html)

The OpenStack Foundation Board of Directors met in-person for two days (https://wiki.openstack.org/wiki/Governance/Foundation/8Mar2017BoardMeeting) in Boston last week.

The first day was a strategic planning workshop (https://crustyblaa.com/march-8-2017-openstack-foundation-strategic-planning-workshop.html) and the second day was a regular board meeting.

Executive Director Update

After a roll call and approving the minutes of the previous meeting, Jonathan gave a presentation outlining his perspective on where OpenStack is at (http://lists.openstack.org/pipermail/foundation/attachments/20170313/73838d4b/attachment-0002.pdf).

He talked about perception challenges that the Foundation is working to address, emphasizing the messages of "costs less, does more" and "all apps need Open Infrastructure".

He described how the Foundation has been iterating on a "pitch deck" and had completed twenty or so interviews with journalists where they presented this deck in a concerted effort to "change the narrative". This begins by talking about some recent industry trends, including a slowdown in the rate of public cloud growth and the coverage of Snap's significant spending on public cloud services compared to their revenues.

It goes on to compare "Gen 1" and "Gen 2" private clouds, with the challenges moving from technology and people to culture and processes. The advantage of a multi-cloud strategy, with sophisticated workload placement was discussed as well as some detailed information backing up the claim that companies are achieving cost savings with OpenStack.

Jonathan described how this presentation had been very well received, and had resulted in some very positive coverage (https://techcrunch.com/2017/02/22/openstacks-latest-release-focuses-on-containers/).

Jonathan briefly touched on key improvements in Ocata and information about the profile of our contributors. This resulted in some debate about the value of "vanity metrics" and there was general agreement that while there continues to be demand for this information, we we should all be cautious in over-emphasizing it.

Finally, Jonathan reflected on feedback from the Project Team Gathering in Atlanta. Attendance was strong and attendees reported feeling less distractions and a greater ability to have important discussions because of the smaller venue and attendee count. While it was mostly seen as a great success, there are areas for improvement including the approach to choosing hotels and committing to hotel room bookings, how scheduling is organized, the size of some rooms, etc. Jonathan mentioned the success of the Forum and the OpenStack Summit in Boston will be key.

User Committee and Compensation Committee

The board spent a fairly short time discussing two matters from its committees.

Firstly, a proposal from the User Committee Product Working Group for facilitating the organization of the schedule for The Forum in Boston was well received, but since the board encouraged all interested parties to work with the Foundation staff who are coordinating the planning, particularly Tom Fifield and Thierry Carrez.

Secondly, the board approved Jonathan's goals for 2017, which had been prepared by the Compensation Committee and sent to board members earlier for review. The board only sets Jonathan's goals, and empower him to set the goals for the rest of the staff. Related to Jonathan's goals, the board also had some discussion later around documenting the goals of the board itself.

Membership Applications

Significant time during the day was given over to considering some corporate membership applications.

With HPE resigning its Platinum membership, we had invited applications to take over its slot and received applications from Ericsson and Huawei. Both companies had previously applied in November, 2014 at the OpenStack Summit in Paris to replace Nebula as Platinum member, but Intel had been successful that time.

Anni Lai presented for Huawei (http://www.huawei.com/en/) and Chris Price presented for Ericsson (https://www.ericsson.com/). Both described their employer's vision for OpenStack, and their wide range of contributions to date. Both also talked about their position in the market, their key customers, and how they are growing the OpenStack ecosystem. The presentations were well received and, after an executive session, the board voted to approve Huawei as a Platinum member. The board expressed their gratitude for Ericsson's interest and preparation, observing that having multiple companies interested in Platinum membership opportunities is a sign of the strength of our community.

Representatives from H3C (http://www.h3c.com/en/) also presented their application for Gold membership. H3C is a prominent IT vendor in China's enterprise IT market, distributes an OpenStack based cloud product, has several very large reference customers, and is establishing itself as a technical contributor to the project. After executive session, the board voted to approve H3C as a Gold member.

Wrapping Up

One piece of good news wasn't public during the meeting, but has since been announced by Jonathan (http://lists.openstack.org/pipermail/foundation/2017-March/002477.html):

> The Board also approved the promotion of Thierry Carrez to VP of Engineering for the OpenStack Foundation. Thierry has been a leader in the technical community since the beginning of OpenStack and has also built a team within the Foundation focused on upstream collaboration.

I think it's safe to say the board were warmly supportive of this change, wished Thierry every success in this new role, and looked forward to working more closely with Thierry than ever before.

And, with that, the board dispersed feeling pretty fried after an intense couple of days of discussions!

by markmc at March 14, 2017 11:00 AM (https://crustyblaa.com/march-9-2017-openstack-foundation-board-meeting.html)

Hugh Blemings (http://hugh.blemings.id.au)

Lwood-20170312 (http://hugh.blemings.id.au/2017/03/14/lwood-20170312/)

Introduction

Welcome to Last week on OpenStack Dev (“Lwood”) for the week just past. For more background on Lwood, please refer here (http://hugh.blemings.id.au/openstack/lwood/).

Basic Stats for the week 6 to 12 March for openstack-dev:

- ~447 Messages (down nearly 22% relative to the long term average)
- ~135 Unique threads (down just shy of 25% relative to the long term average)

Traffic about the same as last week, if anything up slightly. A busy few days conspired against me so Lwood is a bit short and a bit late this week, apologies to those who I know set their clocks by it’s arrival... ;)

Notable Discussions – openstack-dev

OpenStack Summit Boston Schedule Available

Erin Disney writes (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113669.html) that the schedule is now up for the Boston Summit later this year.

Call for Mentors at upcoming Summit

Emily Hugenbruch notes (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113580.html) that the upcoming Boston summit will again provide an opportunity for Mentors to assist newcomers to OpenStack in getting up to speed. If you’re interested, please follow the info in Emily’s email and sign up.

OpenStack PTG Atlanta summary of summaries

As mentioned in the previous couple of Lwoods, with the Atlanta event concluded summaries of the event, mostly from a projects standpoint are rolling in. There were a few more this week past, listed below, and the original blog post (http://hugh.blemings.id.au/2017/03/07/openstack-ptg-atlanta-2017-summary-of-summaries/) has been updated too.

- [Acceleration] Atlanta PTG Summary (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113800.html) – Zhipeng Huang

- [Heat] Heat PTG recap (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113436.html) – Rico Lin
- [Ironic] Pike PTG report (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113546.html) – Dimitry Tantsur
- [Mistral][PTG] Following up on Pike PTG sessions (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113680.html) – Renat Akhmerov
- [Neutron] Upgrades PTG recap (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113371.html) – Ihar Hrachyshka
- [Octavia] Octavia PTG summary – Cross-project teams Pt1 (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113430.html) & Pt2 (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113432.html) – Michael Johnson

End of Week Wrap-ups, Summaries and Updates

Two this week; Ironic (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113411.html) (Ruby Loo) and Nova (Balazs Gibizer (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113358.html))

People and Projects

Core nominations & changes

- [Kolla] Proposing duonghq (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113513.html) for core – Michał Jastrzębski
- [Neutron] Stepping down from Neutron roles – Nate Johnston (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113416.html)
- [Nova] Core team changes – removing Daniel Berrange (danpb) (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113672.html) and Michael Still (mikal) (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113672.html) from nova-core, adding Jay Pipes (jaypipes) (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113672.html) and Sylvain Bauza (bauzas) (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113672.html) to nova-specs-core – Matt Riedemann

Miscellanea

Further reading

Don't forget these excellent sources of OpenStack news – most recent ones linked in each case

- What's Up, Doc? (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113704.html) by Alexandra Settle
- API Working Group newsletter (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113645.html) – Chris Dent and the API WG
- OpenStack Developer Mailing List Digest (http://lists.openstack.org/pipermail/openstack-dev/2017-January/111139.html) by Mike Perez & Kendall Nelson
- OpenStack news over on opensource.com (https://opensource.com/article/17/3/openstack-news-march-13) by Jason Baker
- OpenStack Foundation Events Page (https://www.openstack.org/community/events/) for a frequently updated list of events

Credits

No tunes this week, was again working remotely and needed all the concentration I could muster despite the relative simplicity of the task at hand! :)

by hugh at March 14, 2017 09:37 AM (http://hugh.blemings.id.au/2017/03/14/lwood-20170312/)

Mark McLoughlin (https://crustyblaa.com/)

March 8, 2017 OpenStack Foundation Strategic Planning Workshop (https://crustyblaa.com/march-8-2017-openstack-foundation-strategic-planning-workshop.html)

The OpenStack Foundation Board of Directors met in-person for two days (https://wiki.openstack.org/wiki/Governance/Foundation/8Mar2017BoardMeeting) in Boston last week.

The first day was a joint workshop with the Technical Committee, User Committee, and Foundation staff. The workshop was planned in response to the "OpenStack Futures" discussion at our three previous board conference calls in November (https://crustyblaa.com/november-17-openstack-foundation-board-meeting.html), December (https://crustyblaa.com/december-6-openstack-foundation-board-meeting.html), and January.

Introductions

We began the workshop with very brief personal introductions, followed by Alan, Thierry, and Edgar giving an overview of the roles and responsibilities of the Board of Directors, the Technical Committee (https://docs.google.com/presentation/d/17U3Lf83nKrhfqT0HLr3gq5xqQmwUFcW28OcfUrS1n3E), and the User Committee (https://docs.google.com/presentation/d/1dsRa35TaqeiSrB-n76pj2jONBnWW_-L1UryEbT1vqNE).

State of the Union

Next, Mark Collier presented his view of the state of OpenSack (http://lists.openstack.org/pipermail/foundation/attachments/20170313/73838d4b/attachment-0003.pdf), with a particular emphasis on the four areas planned for discussion during the day (https://etherpad.openstack.org/p/ocata-strategic-review-board). Mark began by talking about the exciting opportunity we had by having such breadth and depth of expertise in the room, and appealed to everyone to put aside their particular roles and work together as a single leadership team to talk about the work.

Evolving The Architecture

Mark spent some time trying to demystify the Big Tent change, describing how the previous Stackforge/Incubation/Integrated stages have been replaced by almost any project being welcome to use OpenStack infrastructure with the TC responsible for reviewing applications to join the set of official OpenStack projects known as "The Big Tent".

Mark then described one of the key changes happening in OpenStack right now as the containerization of the control plane, with projects like Kolla, openstack-helm, and TripleO all tackling this area. He also talked about the work happening around running containers on OpenStack itself with projects like Kuryr, Fuxi, Magnum, and Zun, but he also wondered aloud whether we're addressing all the right integration points. He also described some of the ongoing debates about the scope of OpenStack and our technology choices, with topics like the use of golang, the Gluon project, whether we welcome competition within the Big Tent, and community-wide goals.

Finally, Mark gave us a preview of the work happening around version 2 of the OpenStack Project Navigator (https://www.openstack.org/software/project-navigator/) and talked about how this will play a key role in helping people understand what OpenStack provides and how it can be used.

Unanswered Requirements

Mark talked briefly about the working groups under the User Committee (https://wiki.openstack.org/wiki/Governance/Foundation/UserCommittee) and the transition from Design Summit to Project Team Gathering (https://www.openstack.org/ptg/) and Forum (https://wiki.openstack.org/wiki/Forum) formats. These concepts are all important in understanding how OpenStack thinks about our requirements gathering, strategic long-term planning, and implementation planning.

Mark also gave a preview of some detractor quotes from our user survey, and emphasized a common theme - the perceived and actual complexity of OpenStack, both in terms of understanding and operating the software.

Adjacent Communities

Mark classified the various sets of adjacent communities that we are particularly interested in developing strong relationships with. Container technologies like Kubernetes, Docker, and Mesos. PaaS technologies like CloudFoundry and OpenShift. NFV projects like OPNFV and Cloudify. Provisioning technologies like Terraform, Puppet, and Saltstack. And specific ecosystem relationships, with companies like CoreOS.

Mark described the change in the Foundation's event strategy, targeting events like KubeCon, DockerCon, CoreOS Fest, etc. as key events where we should be positioning the OpenStack brand and developing relationships.

He also described particular focus areas of individual staff members which are relevant to the topic - Chris Hoge working with upstream Kubernetes and running OpenStack SIG meetings, David Flanders working on a report around the gaps when running platforms on OpenStack (like Cloud Foundry, OpenShift, Kubernetes, and Terraform), and how Ildiko Vancsa and Kathy Cacciatore are both working closely with OPNFV.

Finally, Mark talked about the Open Source Days event at the OpenStack Summit in Boston, as well as some very early stage discussions for an OpenDev event which would be a small, focused event around improving the integration between applications frameworks and open infrastructure.

Community Health

The final area of discussion was the subject of community health, and Mark first put out some statistics that he felt painted a very reassuring picture of the community's health. In 2016, we had 3,500 unique contributors, 1,850 of which were retained from 2015. In Ocata, we had fewer developers than Newton, most likely because it was a shorter cycle.

Mark contrasted challenges with projects like Trove and Designate losing contributors, while projects like Kuryr, Kolla, and Zun seeing the greatest number of new contributors.

Similarly, Mark talked about HPE laying off upstream developers, Cisco killing off intercloud, a small slowdown in Summit sponsorships, while we have also added 7 more Gold members, and many first-time corportate members and Summit sponsors.

Strategic Planning Exercise

The rest of the day was given over to a multi-stage strategic planning exercise prepared by Allison and Alan. The idea was to discuss these focus areas, gather everyone's ideas for improvement (https://photos.google.com/share/AF1QipM10xRXuSCufsV8g8O60nnWgAW0n3yDteHqM1BPSgUfxNNZ6fZulsJ8BnYtcY5pNA?key=UEk5SXlaYlRSdllyYXZDVTlvN2EwR1NidjhjUGI3), summarize and categorize these ideas, vote on ideas in each focus area, and finally agree on how to proceed with concrete goals for the next 6-9 months.

Discussion

The initial discussion covered a lot of ground. Allison introduced each focus area by describing the input we gathered via the etherpads and input she gathered through 1:1 interviews with a variety of people.

One topic of discussion related to how OpenStack can simplify how we describe OpenStack, particularly to reduce confusion introduced with the Big Tent change. Various ideas around categorization, tagging, vertical definitions, a concept of constellations, maturity ratings, and much more, were discussed.

We talked about the promise for the future that OpenStack provides. That there will be evolution over time, that we deliver the cloud solutions of today and will deliver the solutions of tomorrow. That the challenge of smooth upgrades is part of our challenge in delivering "future proof infrastructure".

We talked about the challenges of scalability, manageability, and complexity. The theme of containerized deployments, the need for vertically focused views of OpenStack, for example for Telco users. We discussed the need for OpenStack to be able to evolve over time, with refactoring or rewriting components being only one of the possible approaches we may see over time.

We talked at great length about how OpenStack could work more closely with adjacent communities. How the relationship with these communities should bring value to both communities. We particularly emphasized the need for a closer relationship with the CNCF and the Kubernetes community.

Gathering Ideas

Over lunch, everyone wrote their concrete, actionable ideas for improvement on sticky notes and put them on flipcharts for each of the areas of discussion (https://photos.google.com/share/AF1QipM10xRXuSCufsV8g8O60nnWgAW0n3yDteHqM1BPSgUfxNNZ6fZulsJ8BnYtcY5pNA?key=UEk5SXlaYlRSdllyYXZDVTlvN2EwR1NidjhjUGI3). Later, Jonathan volunteered to group the ideas into themes, and summarized these themes for the group, facilitating further discussion before voting on which theme in each area we should particularly focus on.

On the subject of communicating about "what is OpenStack", the main themese were marketing activities, various categorization ideas, and idea Allison talk about earlier referred to as "constellations". We later voted to focus on the categorization area and formed a group of interested parties:

> Communicate about OpenStack: Categorize (objective data) and map (subjective approach) OpenStack projects as base versus optional (within a specific use case), integrated versus independent release, emerging versus mature, stability, adoption metrics, what works together, services versus consumption (operational tools/client libraries), and other criteria
>
> Names: Thierry Carrez [lead], Alan Clark, Allison Randal, Jon Proulx, Melvin Hillsman, Lauren Sell, Tim Bell, Mark Baker, Kenji Kaneshige

For unanswered requirements, we discussed how to prioritize, ideas around a solution focus, scalability challenges, and a list of specific features that people felt were important. A counter-point was made that rather than focusing on any of these ideas, perhaps the focus should be on working with adjacent communities. Later, we discussed the need to grow the connection between the Product Working Group, the TC, and individual projects. The outcome and group for this was:

> Requirements: Bring different groups (UC/technical/etc) together at Forum to collaborate/communicate aroud user stories, gap analysis, what fits in the current state of tech, prioritize what would have the greatest impact in reducing pain for users.
>
> Names: Melvin Hillsman [lead], Yih Leong Sun, Jon Proulx, Rob Esker, Emilien Macchi, Doug Hellmann, Tim Bell, Shamail Tahir

On the topic of adjacent communities, we observed that by far the most dominant area of discussion was the need to create better connection with the Kubernetes community. The themes were community engagement, technical engagement, OpenStack consuming technology from the Kubernetes and containers world, and making OpenStack technology more consumable by Kubernetes. In the end, there was strong consensus to focus on the consumability of OpenStack technologies:

> Adjacent Technologies: Make our technology more consumable (independently) by other communities/projects.
>
> Names: Chris Price [lead], Alan Clark, Dims, Rob Esker, Mark Collier, Steven Dake, Mark McLoughlin, Shamail Tahir

For changes to the technology, we discussed simplifications, making containers first class citizens, recording tribal knowlede, culling failed efforts, converging deployment tools, and welcoming emerging or competing projects. The theme we voted to focus on was:

> Changes to the Technology: Workstream to simplify existing projects, reduce dependency options, reduce config options.
>
> Names: Mike Perez [lead], --> TC project

Finally, on the subject of community health, we talked about onboarding contributors, reworking our processes, community tools, growing leaders, corporate involvement in the project, and recognizing work with adjacent communities. We voted to focus on the leadership theme:

> Community Health: Grow next generation of leadership/experts/cross-project devs within the community
>
> Names: Steven Dake [lead], Chris Price, Jeremy Stanley, Dims, AlanClark, Joseph Wang

Next Steps

For each of these focus areas, the lead person in the group committed to organizing a kick-off meeting by March 22nd. The real work will begin there!

by markmc at March 14, 2017 07:00 AM (https://crustyblaa.com/march-8-2017-openstack-foundation-strategic-planning-workshop.html)

March 13, 2017

Opensource.com (https://opensource.com/taxonomy/term/5126/feed/feed)

Erasing complexity, submitting a summit talk, and more OpenStack news (https://opensource.com/article/17/3/openstack-news-march-13)

Are you interested in keeping track of what is happening in the open source cloud? Opensource.com is your source for news in OpenStack (https://opensource.com/resources/what-is-openstack), the open source cloud infrastructure project.

OpenStack around the web

From news sites to developer blogs, there's a lot being written about OpenStack every week. Here are a few highlights.

by Jason Baker at March 13, 2017 04:45 PM (https://opensource.com/article/17/3/openstack-news-march-13)

OpenStack Superuser (http://superuser.openstack.org)

How to set up your work environment to become an OpenStack developer (http://superuser.openstack.org/articles/openstack-developer-beginner-setup/)

Although there are developer (http://docs.openstack.org/infra/manual/developers.html) and wiki (https://wiki.openstack.org/wiki/How_To_Contribute) guides on how to get started with OpenStack, I have found them bit overwhelming as a beginner. After reading various docs and asking for help from my mentor, who is a core contributor in OpenStack, I came up with the following easy-to-follow guide. If you still face any problems while setting up your environment, feel free to reach out by commenting below.

1. Install Ubuntu in an Oracle Virtual Box

A virtual machine (VM) is recommended because usually we don't want a ton of dependencies installed on our everyday environment. Also, if at any point we mess up things, it's easier to start over from scratch using a VM.

1. Download (https://www.virtualbox.org/wiki/Downloads) a suitable Oracle Virtual Box for your operating system.
2. Download (http://www.ubuntu.com/download/desktop) the desired Ubuntu iso file.
3. Install and start Oracle Virtual Box.
4. Click on the "New" button in the wizard, give the new virtual machine a name, check "Linux" in the "Type" area and check "Ubuntu" in the "Version" area (32 or 64-bit, depending on downloaded iso file).
5. Set the amount of RAM (ideally not more than 50% of your total RAM). Something to keep in mind: DevStack will perform best with 4GB or more of RAM.
6. Select "Create a Virtual Hard Disk Now," check "VDI (VirtualBox Disk Image)," then select "Dynamically Allocated" and finally set the hard disk size (60- 100 GB ideally)
7. Double-click your new machine in the left menu and select the downloaded iso file.
8. Next, click "Install Ubuntu". Click "Continue" and then select "Erase Disk and Install Ubuntu". (Note: this will not erase files on your local machine). Complete the rest of the wizard and, finally, you'll have Ubuntu installed inside your VM.
9. Before restarting the VM:
    - Select your machine and click on "Settings".
    - Under the "Storage" tab, check if the installation iso file is still present; if it is, select and remove it.
10. To work in full screen, install guest additions. To do so, restart your VM, click on "Devices" from menu, select "Insert Guest Additions CD image" and press "Run". After completion, press "Enter" and restart your VM.

NOTE: Press Ctrl+Alt+t to open the Terminal application. To distinguish commands from normal sentences, $ sign has been added at their beginning. So, you have to write the words after '$' on the terminal and then press "enter" to run the command. To use copy/paste option for commands, you should open this web page inside any web browser of your created VM. After selecting the command text press Ctrl+C to copy and then inside the terminal press Ctrl+Shft+V to paste. Also, the words which have to be replaced by specific information pertaining to you have been indicated by capitals words, like YOUR_FIRST_NAME.

2. Set up a Stack user with superuser permissions

Devstack should be run as a non-root user with sudo enabled (standard logins to cloud images such as "ubuntu" or "cloud-user" are usually fine). Since this user will be making many changes to your system, it will need to have sudo privileges.

a) Create the group *stack* and add the user *stack* in it:

```
$ sudo groupadd stack
```

```
$ sudo useradd –g stack –s /bin/bash –d /opt/stack –m stack
```

b) Grant superuser permissions to the *stack* user:

```
$ sudo su
```

```
$ echo "stack ALL=(ALL) NOPASSWD: ALL" >> /etc/sudoers
```

c) Logout as your default user:

```
Ctrl+d
```

d) Create a password for the *stack* user:

```
$ sudo passwd stack
```

e) Now to login as the *stack* user:

```
$ su stack
```

f) Go to the home directory of the *stack* user:

```
$ cd ~
```

3. Set up SSH Keys : Source (https://help.github.com/enterprise/11.10.340/user/articles/generating-ssh-keys/#platform-linux)

a) Check the present working directory; it should output "/opt/stack":

```
$ pwd
```

If it doesn't output that, redo steps 2.e) and 2.f) before moving on.

b) Create a new SSH key, using your provided email as a label:

```
$ ssh–keygen –t rsa –C "YOUR_EMAIL@EMAIL_ADDRESS.com"
```

c) Press the "Enter" key to accept the default file location in which to save the key.

d) At the prompt, type a secure passphrase. You may keep it empty by directly pressing the "Enter" key for no passphrase.

e) Start the *ssh-agent* in the background

```
$ eval "$(ssh–agent –s)"
```

f) Add your SSH key to *ssh-agent*.

```
$ ssh–add ~/.ssh/id_rsa
```

g) Download and install *xclip*.

```
$ sudo apt–get install xclip
```

h) Copy the SSH key (i.e. contents of the id_rsa.pub file) to your clipboard.

```
$ sudo xclip -sel clip < ~/.ssh/id_rsa.pub
```

i) If you don't have a github account, first create one (https://github.com/join). Then, login into your github account, go to "Settings", click "SSH and GPG keys" then select "new SSH key". Write a description in "Title" and paste your key into the "Key" field (for pasting, press Ctrl+V). Finally press "Add SSH Key".

j) Test your connection using *ssh*:

```
$ ssh -T git@github.com
```

Now if the fingerprint matches, type "yes." If you now see you username in the message, you have successfully set up your SSH key!

4. Set up Git : Source (https://help.github.com/articles/set-up-git/#platform-linux)

a) Check the present working directory, it should output "/opt/stack":

```
$ pwd
```

If it doesn't output that, redo steps 2.e) and 2.f) before moving on.

b) Install *git* from terminal:

```
$ sudo apt-get install git
```

c) Use *config* and write in your full name:

```
$ git config --global user.name "YOUR_FIRSTNAME YOUR_LASTNAME"
```

d) Use *config* and write in your email address:

```
$ git config --global user.email YOUR_EMAIL@EMAIL_ADDRESS.com
```

e) Check your *git* configuration

```
$ git config --list
```

5. Set up DevStack

a) Check the present working directory, it should output "/opt/stack":

```
$ pwd
```

If it doesn't output that, redo steps 2.e) and 2.f) before moving on.

b) Download DevStack

```
$ git clone https://git.openstack.org/openstack-dev/devstack (https://git.openstack.org/openstack-dev/devstack)
```

c) Go to the *devstack* directory

```
$ cd devstack
```

d) Copy the sample configurations into this current directory

```
$ cp samples/local.conf .
```

e) Notice that the main projects (Keystone, Nova, etc.) are already downloaded, but the clients (like *python-keystoneclient* or *python-novaclient*) services are not.

The following commands will download the source code from *python-keystoneclient* and will let you modify it for your development. Then, you can test it against the DevStack's OpenStack cloud:

```
$ sudo apt-get install vim
$ vim local.conf
```
To get into INSERT MODE `Press i`
At the bottom, add `LIBS_FROM_GIT=python-keystoneclient`
To get back into COMMAND MODE `Press Esc`
To save and quit, write `:wq`

f) Run the *stack* script:

```
$ ./stack.sh
```

You will get the following output after successful completion of the command.

```
The default users are: admin and demo
The password: nomoresecret
```

g) In any web browser open:

```
http://localhost:5000/ (http://localhost:5000/)
```

If you are able to establish a connection, then you have successfully setup your DevStack!

6. Set up Gerrit : Source (http://docs.openstack.org/infra/manual/developers.html)

Gerrit is the code review system used in OpenStack development. Git-review tool is a git subcommand that handles all the details of working with Gerrit.

a) Join the OpenStack Foundation (https://www.openstack.org/join/)

b) Create a LaunchPad account (https://login.launchpad.net/)

c) Open a terminal window and check the present working directory, it should output "/opt/stack".

```
$ pwd
```

If it doesn't output that, redo steps 2.e) and 2.f) before moving on.

d) Copy your SSH key:

```
$ sudo xclip -sel clip < ~/.ssh/id_rsa.pub
```

e) Open https://review.openstack.org/ (https://review.openstack.org/), click "Sign In" (top-right corner) and log in with your Launchpad ID.

f) Now click on "Settings," then select "SSH Public Keys" and press "Add Key". Press Ctrl+V to paste the key and then click "Add."

g) Install git review

```
$ sudo apt-get install git-review
```

h) Check if git review works inside the Keystone (or any other project for that matter) directory of OpenStack:

```
$ cd keystone
```

```
$ git review -s
```

7. Install dependencies : Source (http://docs.openstack.org/developer/keystone/setup.html)

a) Check the present working directory, it should output "/opt/stack":

```
$ pwd
```

If it doesn't output that, redo steps 2.e) and 2.f) before moving on.

b) Install the dependencies:

```
$ sudo apt-get install python-dev python3-dev libxml2-dev libsqlite3-dev libssl-dev libldap2-dev libffi-dev
```

8. Run the tox command

Each project like Keystone, Nova, Cinder etc. has a tox.ini file defined in it. It defines the tox environment and the commands to run for each environment. Please note that the subsequent runs of tox will be faster because everything fetched will be in .tox already.

a) Check the present working directory, it should output "/opt/stack":

```
$ pwd
```

If it doesn't output that, redo steps 2.e) and 2.f) before moving on.

b) Install tox and pbr:

```
$ sudo apt-get install python-tox
```

```
$ sudo pip install pbr
```

c) Update and upgrade:

```
$ sudo apt-get update
```

```
$ sudo apt-get upgrade
```

d) Go inside any project directory like Keystone, Cinder or Nova:

```
$ cd keystone
```

e) Run tox

```
$ tox
```

f) If you get the following output on the terminal after entering the command, the tox has installed successfully.

```
_____________________________summary _____________________________
py27: commands succeeded
pep8: commands succeeded
api-ref: commands succeeded
docs: commands succeeded
genconfig: commands succeeded
releasenotes: commands succeeded
```

Now that you have set up your work environment, you can start contributing as a developer. For more tips on that, stay tuned! 😉

Nisha Yadav (https://www.linkedin.com/in/nishaprofile/) is a former OpenStack Outreachy intern and current OpenStack contributor. This post first appeared on her blog (https://nishayadavsite.wordpress.com/2016/06/29/how-to-set-up-work-environment-to-become-an-openstack-developer/). Superuser is always interested in community content, email: editor@superuser.org (mailto:editor@superuser.org).

Cover Photo (https://www.pexels.com/photo/lego-walpaper-computer-59628/) // CC BY NC (https://creativecommons.org/licenses/by-nc/2.0/)
The post How to set up your work environment to become an OpenStack developer (http://superuser.openstack.org/articles/openstack-developer-beginner-setup/) appeared first on OpenStack Superuser (http://superuser.openstack.org).

by Nisha Yadav at March 13, 2017 11:48 AM (http://superuser.openstack.org/articles/openstack-developer-beginner-setup/)

Cloudify Engineering ()

Cloudify's TOSCA Journey - The Convergence of ARIA and TOSCA (An Infographic) (http://getcloudify.org/2017/03/13/cloudify-TOSCA-ARIA-convergence-infographic.html)

Click the above image to zoom. Cloudify have been early adopters and implementers of TOSCA, betting on TOSCA as early...

March 13, 2017 12:00 AM (http://getcloudify.org/2017/03/13/cloudify-TOSCA-ARIA-convergence-infographic.html)

March 12, 2017

David Moreau Simard (https://dmsimard.com/)

An even better Ansible reporting interface with ARA 0.12 (https://dmsimard.com/2017/03/12/an-even-better-ansible-reporting-interface-with-ara-0-12/)

Not even a month ago, I announced the release of ARA 0.11 (https://dmsimard.com/2017/02/13/announcing-the-ara-0.11-release/) with a bunch of new features and improvements.

Today, I'm back with some more great news and an awesome new release, ARA 0.12(.3) !

That's right, 0.12.3!

Due to the nature of this new release, I wanted to be sure to get feedback from the users before getting the word out.

We got a lot of great input! This allowed us to fix some bugs and significantly improve the performance.

0.12 features a completely re-written and re-designed web application user interface. Let's look at some of the highlights !

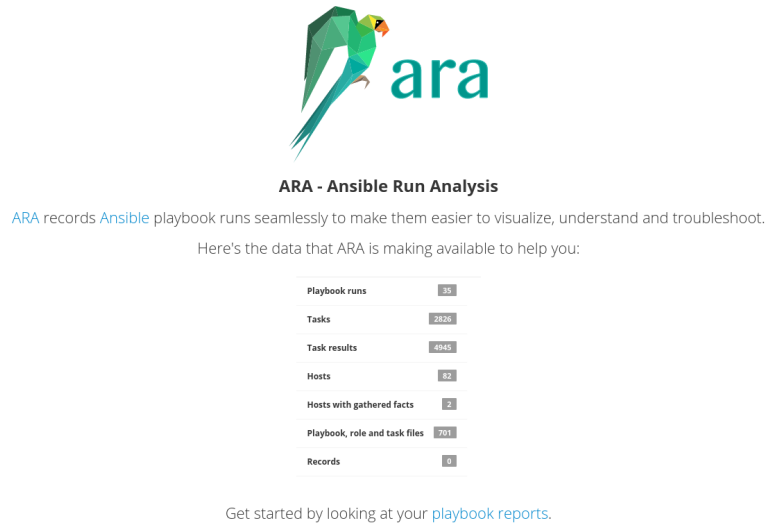
A new web application interface

I know what you're most interested in is... WHAT DOES IT LOOK LIKE !?

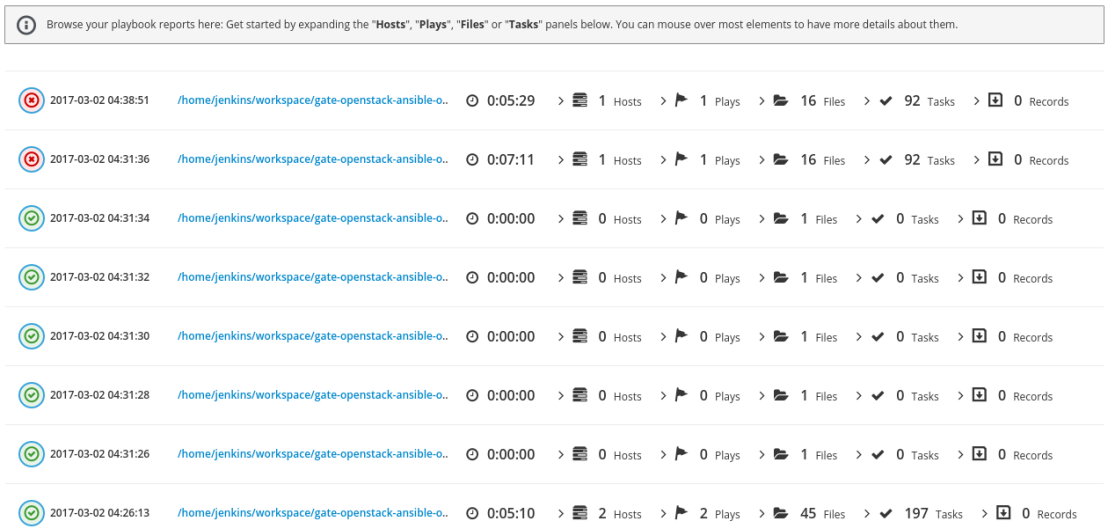
What it looks like

Here's some highlights of the new user interface — it doesn't end here so please read on !

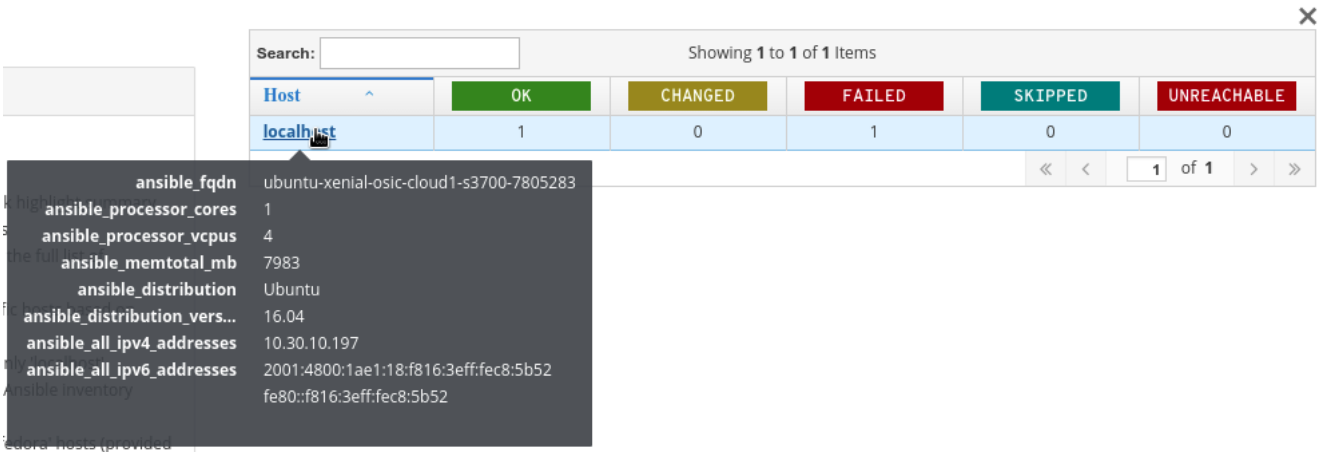
The home page now features the data recorded by ARA:



The core of the user interface now revolves around one and single page where you'll be able to find all the information about your playbooks:



Quickly have a glance at your playbook host summary:



Or dig into the host details to look at all the facts Ansible gathered for you:

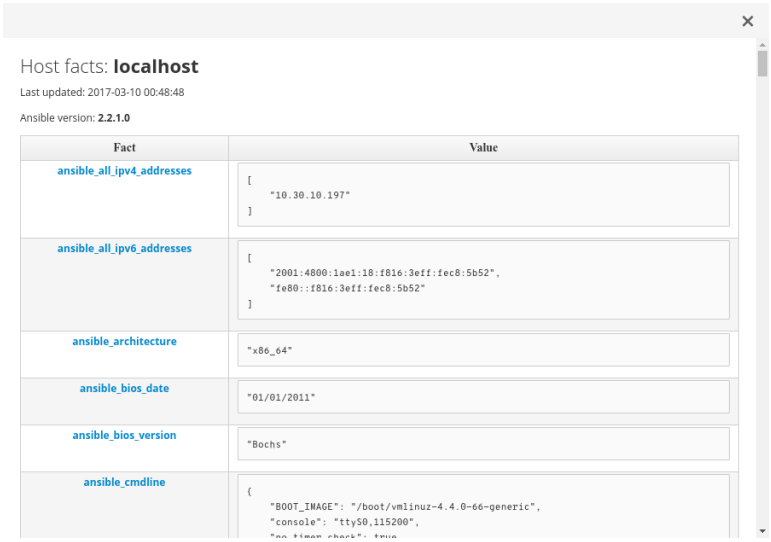
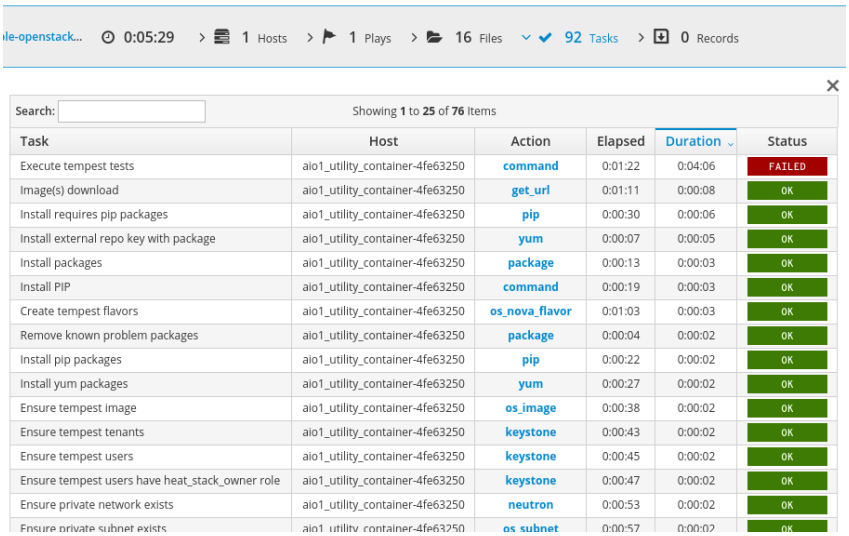
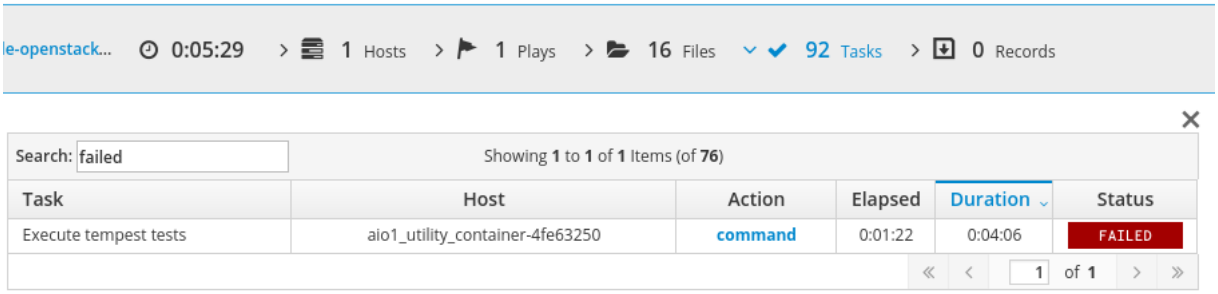


Figure out which tasks took the longest just by sorting the table accordingly:



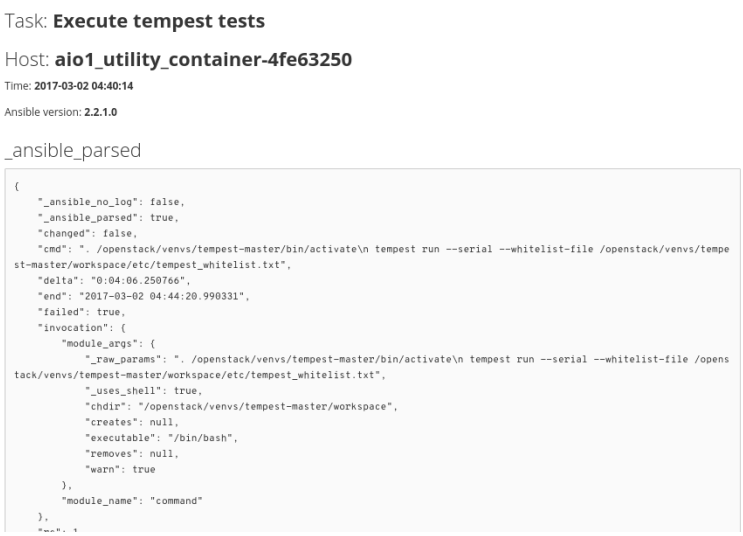
Or search to figure out which tasks failed:



Click on the action to get context on where this task ran:

```
13  # See the License for the specific language governing permissions and
14  # limitations under the License.
15
16  - name: Execute tempest tests
17    shell: |
18      . {{ tempest_venv_bin }}/activate
19      tempest run --serial --whitelist-file {{ tempest_test_whitelist_file_path }}
20    args:
21      chdir: "{{ tempest_venv_bin | dirname }}/workspace"
22      executable: /bin/bash
23    changed_when: false
24    tags:
25      # don't trigger ANSIBLE0013
26      - skip_ansible_lint
27
```

Or click on the status to take a look at all the details Ansible has to offer:



Task: **Execute tempest tests**

Host: **aio1_utility_container-4fe63250**

Time: **2017-03-02 04:40:14**

Ansible version: **2.2.1.0**

_ansible_parsed


```
{
    "_ansible_no_log": false,
    "_ansible_parsed": true,
    "changed": false,
    "cmd": ". /openstack/venvs/tempest-master/bin/activate\n tempest run --serial --whitelist-file /openstack/venvs/tempe
st-master/workspace/etc/tempest_whitelist.txt",
    "delta": "0:04:06.250766",
    "end": "2017-03-02 04:44:20.990331",
    "failed": true,
    "invocation": {
        "module_args": {
            "_raw_params": ". /openstack/venvs/tempest-master/bin/activate\n tempest run --serial --whitelist-file /opens
tack/venvs/tempest-master/workspace/etc/tempest_whitelist.txt",
            "_uses_shell": true,
            "chdir": "/openstack/venvs/tempest-master/workspace",
            "creates": null,
            "executable": "/bin/bash",
            "removes": null,
            "warn": true
        },
        "module_name": "command"
    },
```


The logic behind the UI changes

There were three main objectives with this refactor of the web interface.

Improve UX

A lot of effort was spent on the user experience. You need to be able to find what you want: intuitively, quickly and easily.

Data and result tables are now sortable and searchable by default and browsing tips were added to the interface to help you make the most of what it has to offer.

Scalability and performance

The interface must be fast, responsive, clutter-free, make sense and behave consistently across your use case scenarios, whether you are looking at reports which contains five tasks or ten thousand.

Pagination settings have been introduced in order to customize your browsing experience according to your needs.

Static report generation time and weight

Another objective of this user interface work was to optimize the static report generation performance and weight.

Static generation is one of the great features of ARA which is very heavily used in the context of continuous integration where the report is generated and attached to the artifacts of the job.

Here's a real-life example of the same database being generated on ARA 0.11 and ARA 0.12:

ARA integration tests (5 playbooks, 59 tasks, 69 results):

- Before: 5.4 seconds, 1.6MB (gzipped), 217 files
- After: 2 seconds, 1.2MB (gzipped), 119 files

OpenStack-Ansible (1 playbook, 1547 tasks, 1667 results):

- Before: 6m21 seconds, 31MB (gzipped), 3710 files
- After: 20 seconds, 8.9MB (gzipped), 1916 files

For larger scale playbooks, we're looking at a generation performance that is over 19 times faster. I'm really happy about the results.

But wait, there's more

If you thought the UI work was enough to warrant it's own release, you're right !

Some other changes also sneaked their way into this release as well.

First party WSGI support

A lot of ARA users were interested in scaling their centralized deployment. This meant helping users deploy the ARA web interface through WSGI with a web server.

To help people get going, we now ship a WSGI script bundled with ARA and documented how you can set it up with Apache and mod_wsgi. The documentation is available here (http://ara.readthedocs.io/en/latest/webserver.html).

Other things

- Fixed syntax highlighting when viewing files
- Preparations for supporting the upcoming Ansible 2.3 release
- Started working on full python 3 support
- Various performance improvements

Well, that's it for now

That was certainly a lot of stuff in one release !

I hope you're enjoying ARA - if you're not using it yet, it's easy !

Have a look at the documentation to learn how to install ARA (http://ara.readthedocs.io/en/latest/installation.html) and how to configure Ansible (http://ara.readthedocs.io/en/latest/configuration.html) to use it.

If you have any questions, feel free to drop by on IRC in #ara on the freenode server or hit me up on twitter: @dmsimard (https://twitter.com/dmsimard).

by dmsimard at March 12, 2017 03:00 PM (https://dmsimard.com/2017/03/12/an-even-better-ansible-reporting-interface-with-ara-0-12/)

March 11, 2017

Sean Roberts (https://sarob.com)

10 Steps for the Boss to Understand Your Upstream Project (https://sarob.com/2017/03/project-product-steps/)

Previous articles on Open Source First have been more strategy than recipe. You need a clear, easy to understand plan for making the case for an upstream project to your manager. To help you with your boss, I have rewritten the How to use Public Projects to Build Products article into a list of ten steps. These ...

Continue reading "10 Steps for the Boss to Understand Your Upstream Project" (https://sarob.com/2017/03/project-product-steps/)

The post 10 Steps for the Boss to Understand Your Upstream Project (https://sarob.com/2017/03/project-product-steps/) appeared first on sarob (https://sarob.com).

by sarob at March 11, 2017 09:12 PM (https://sarob.com/2017/03/project-product-steps/)

March 10, 2017

OpenStack Superuser (http://superuser.openstack.org)

Containers are here to stay – until they're not (http://superuser.openstack.org/articles/containers-kuberbetes-google/)

Containers are the price of admission to the modern platform, says Google's Kelsey Hightower, but they're just the start.

At the recent Container World conference (https://tmt.knect365.com/container-world/), Hightower, formerly of CoreOS and now a developer advocate at the company that created Kubernetes (http://superuser.openstack.org/articles/mars-open-source-kubernetes/), talked about the huge gap between deploying applications and managing them in production and how to move beyond the single-machine programming model and start adopting API-driven distributed systems to unlock the true value of containers.

For starters, Hightower (https://twitter.com/kelseyhightower) doesn't believe the hype about them. "Containers are the thing we talk about today, that we're excited about. Five years from now we'll hate them all. Guarantee it, we'll hate them all."

He prefers to talk about what can be done with this new set of ideas and concepts — "moving up the stack of thinking" — considering the container as just a box that allows user to embrace the concept of platform-as-a-system.

"Containers are not platforms, when people say 'We're going to adopt containers' to my mind it's like saying, 'We're going to adopt .zip files.' What they mean, he says, is they are planning to adopt Mesos, Swarm, Kubernetes, etc. — they are considering adopting a platform for the first time. "The container thing isn't that exciting anymore," he says, soliciting applause from the audience for Docker "for making containers real."

A lot of platforms look like Kubernetes now, he adds, and the interesting part about this platform is the API server.

The current platforms are like Linux distros, it's a platform but the APIs are all over the place — bash shell, cron jobs, package managers, some shell scripts, etc. Instead if you look at Kubernetes, you see an API server and if you give the user a container, all the components around it will do something fantastic with it, Hightower says.

"First we'll place it onto a machine — step one, that's the obvious thing, rolling updates — then once the app is deployed what happens next?" He then launched into a demo of iconic 80s video game Tetris, where the goal is to place the blocks in the right place, based on what shows up on the screen now.

"Users hit one button and blocks fall down stacking up randomly: it's fully automated and hands-off completely. But you can see to the left and to the right, however, that CPU and memory are totally being lost. You're just losing on this one, it's totally automated but you have no resource awareness. Your scripts aren't designed to handle any conditions that are real time."

<iframe allowfullscreen="allowfullscreen" frameborder="0" height="360" src="https://www.youtube.com/embed/a1VDS-8TBX0" width="640"></iframe>

In the Kubernetes world, each workload is examined and instead of assigning it statically to a machine or relying on scripts, each workload gets evaluated and a decision is made. Bin packing is used, so as the workload "falls," the user decides where it should land, moving it to the right place.

All of that looks great with new development but there's a reason to curb container enthusiasm if you're in a legacy environment. "[For] enterprise, the scenario is not so pretty- you start out in a different world, it's not greenfield (https://en.wikipedia.org/wiki/Greenfield_project), you already have some things," Hightower says. You can still use resource managers and get some benefit from them, he says. "One of those benefits is to fill in the blanks — you install these things right underneath your set-up — whether you're running OpenStack or VMware you can carve out resources within that cluster. Over time, as you do more and more workloads you start to defrag your cluster...You get benefits even in the brownfield world."

But If you're in the real enterprise, you work with databases, then what can you do? "Absolutely nothing. You're screwed.[laughs] I'll be honest: this stuff doesn't work in all environments," he says. "That's the thing most people don't talk about. You just can't put every workload in every operation...There are things to think about when you adopt one of these platforms."

Since the best things happen with modern architectures, Hightower then ran through a demo deploying an app and presenting it with an SSL certificate to process it at run time. In this example with micro services, some details change (the certificate, IIRC) but the core APIs remain the same 'automagically.' "If I scale to 1,000 instances they'd all get the certs and the background would refresh the certificate without notifying the developers or changing any of the Kubernetes API

(https://kubernetes.io/docs/api/). This is what makes these kinds of platforms really powerful. It's not just about deploying applications, it's about being able to build the tools like this and use all of the abstractions just to make it work."

This is where the road splits from the platforms that try do everything for the user. To answer the question of why a user would do what today is fairly tedious work — create all these YAML files, work with this whole system from the outside and have to learn how to deploy and configure every single app — Hightower says it's important to look past containers.

The most important thing is the HDI server and without that HDI server we have all these Kubernetes deployments – the agents, the nodes, the scheduler and the proxy etc. — that can tear user run time.

"Most people stop talking about these – they're getting to the point where they're not excited anymore — and this is a good thing," he says. "Because we don't want to be talking about container packaging forever. But these run times do provide value in communicating to the kernel so we don't have to do this in every platform. Then they fade away."

Cover Photo (https://www.flickr.com/photos/davidstrom/16373384801/) // CC BY NC (https://creativecommons.org/licenses/by-nc/2.0/)

The post Containers are here to stay – until they're not (http://superuser.openstack.org/articles/containers-kuberbetes-google/) appeared first on OpenStack Superuser (http://superuser.openstack.org).

by Nicole Martinelli at March 10, 2017 02:02 PM (http://superuser.openstack.org/articles/containers-kuberbetes-google/)

Mirantis (https://www.mirantis.com)

A Hard Road from Dev to Ops (https://www.mirantis.com/blog/a-hard-road-from-dev-to-ops/)

The post A Hard Road from Dev to Ops (https://www.mirantis.com/blog/a-hard-road-from-dev-to-ops/) appeared first on Mirantis | Pure Play Open Cloud (https://www.mirantis.com).

Last June, in his provocative blog, Mirantis' co-founder Boris Renski proclaimed to the world that infrastructure software was dead (https://www.mirantis.com/blog/infrastructure-software-is-dead/). That blog was a battle cry for us as a company, and the beginning of an organizational evolution away from our exclusive focus on delivering software and towards providing customers with a turnkey infrastructure experience.

It was clear that the future consumption model for infrastructure is defined by public clouds where everything is self-service, API-driven, fully managed and continuously delivered. It was also clear that most vendors, Mirantis included, had misinterpreted where the core of cloud disruption was, overemphasizing disruption in software capabilities around "self-service and API-driven," while largely ignoring the disruption in delivery approach codified as "fully managed and continuously delivered." Private cloud had become a label for the new type of software, whereas public cloud was a label for a combination of software and, most importantly, a new delivery model. Private cloud had failed and we needed to change.

As we started piercing the market with our new Build-Operate-Transfer delivery model for open cloud last year, we pulled the trigger on changing the company internally. Mirantis had to reinvent itself, re-examine every part of the company and ask if it was built correctly and/or was needed in order to deliver an awesome customer operations experience. Organically and through acquisition (http://www.businessinsider.com/mirantis-buys-tcp-cloud-to-take-on-red-hat-vmware-2016-9), we added new engineering and operations folks who brought with them the relentless focus on keeping things simple, and emphasized continuously integrating and managing change. We went away from using advanced computer science as the only means to avoid failures in favor of selecting simple configurations that are less likely to fail and investing heavily in the monitoring and intelligence that predicts failure before it occurs and proactively alerts the operator to avoid failures all together.

In the meantime, and despite the challenges, things were picking up in the field. We weren't alone in realizing that cloud operations are hard, so many OpenStack DIYers that had failed at operations got intrigued by our model. We started winning big managed cloud deals, and made meaningful strides in transitioning our existing marquee accounts like AT&T and VW toward managed open cloud. Most importantly, we weren't just winning new deals; we were expanding existing ones – a much more important sign of delivering customer value. Today, some of the world's most iconic companies are running their customer-facing businesses on our managed clouds without needing to pay much attention to how the cloud is run. They simply expect that it works.

Now we are staring at an explosion of new clouds in our sales pipeline. In order to scale and provide an awesome user experience, this week we've announced the final set of organizational changes that will complete our transformation, putting our 12 months of difficult transition behind:

- We are simplifying the services we offer in our portfolio, focusing less on one-off cloud infrastructure integration and more on strategy, site readiness and cloud tenant on-boarding and care.
- We are combining our 24×7 software support team and our managed operations team into a single-focused customer success team.
- Since many of our customers don't accept managed services from Russia and Ukraine locations (due to regulatory, compliance and corporate security policies), we are shifting roughly 70 jobs from those locations to the U.S., Poland and Czech Republic.

As founders, we felt it was important to share this update publicly, not just because we want the world to know that Mirantis is changing, but also because this transformation is personal to us. We founded Mirantis back in 2000 – originally a small IT services firm, and following this change, some of our best friends and colleagues who have travelled with us for well over a decade will no longer be with the company. We want those who are leaving to know that we are humbled by your brilliance and eternally grateful to have worked alongside such committed and true friends.

As we look at the last twelve months, we're proud of the change we persevered through as a company. Evolving a company is never easy – for management, employees, partners or customers. Many in our space will need to go through a similar evolution to stay relevant in the public cloud world, and not everybody will make it through. We are fully determined that Mirantis to be part of the pack that does.

Onwards and upwards!

The post A Hard Road from Dev to Ops (https://www.mirantis.com/blog/a-hard-road-from-dev-to-ops/) appeared first on Mirantis | Pure Play Open Cloud (https://www.mirantis.com).

by Alex Freedland at March 10, 2017 02:00 PM (https://www.mirantis.com/blog/a-hard-road-from-dev-to-ops/)

March 09, 2017

RDO (http://rdoproject.org/blog/)

Let rdopkg manage your RPM package (http://rdoproject.org/blog/2017/03/let-rdopkg-manage-your-RPM-package/)

rdopkg (https://github.com/openstack-packages/rdopkg) is a RPM packaging automation tool which was written to efortlessly keep packages in sync with (fast moving) upstream.

`rdopkg` is a little opinionated, but when you setup your environment right, most packaging tasks are **reduced to a single rdopkg command**:

- Introduce/remove patches: `rdopkg patch`
- Rebase patches on a new upstream version: `rdopkg new-version`

`rdopkg` builds upon the concept distgit (https://www.rdoproject.org/documentation/rdo-packaging/#distgit---where-the-spec-file-lives) which simply refers to maintaining RPM package source files in a `git` repository. For example, all Fedora and CentOS packages are maintained in `distgit`.

Using Version Control System for packaging is great, so `rdopkg` extends this by requiring patches to be also maintained using `git` as opposed to storing them as simple `.patch` files in distgit.

For this purpose, `rdopkg` introduces concept of patches branch (https://www.rdoproject.org/documentation/rdo-packaging/#patches-branch) which is simply a `git` branch containing... yeah, patches. Specifically, patches branch contains upstream `git` tree with optional downstream patches on top.

In other words, patches are maintained as `git` commits. The same way they are managed upstream. To introduce new patch to a package, just `git cherry-pick` it to patches branch and let `rdopkg patch` do the rest. Patch files are generated from `git`, `.spec` file is changed automatically.

When new version is released upstream, `rdopkg` can *rebase patches branch* on new version and update distgit automatically. Instead of hoping some `.patch` files apply on ever changing tarball, `git` can be used to rebase the patches which brings many advantages like automatically dropping patches already included in new release and more.

Requirements

upstream repo requirements

You project needs to be maintained in a `git` repository and use Semantic Versioning (http://semver.org/) tags for its releases, such as `1.2.3` or `v1.2.3`.

distgit

Fedora packages already live in `distgit` repos which packagers can get by

```
fedpkg clone package
```

If your package doesn't have a `distgit` yet, simply create a `git` repository and put all the files from `.src.rpm SOURCES` in there.

`el7` distgit branch is used in following example.

patches branch

Finally, you need a repository to hold your patches branches. This can be the same repo as `distgit` or a different one. You can use various processes to manage your patches branches, simplest one being packager maintaining them manually like he would with `.patch` files.

`el7-patches` patches branch is used in following example.

install rdopkg

rdopkg (https://github.com/openstack-packages/rdopkg) page contains installation instructions. Most likely, this will do:

```
dnf copr enable jruzicka/rdopkg
dnf install rdopkg
```

Initial setup

Start with cloning distgit:

```
git clone $DISTGIT
cd $PACKAGE
```

Add `patches` remote which contains/is going to contain patches branches (unless it's the same as `origin`):

```
git remote add -f patches $PATCHES_BRANCH_GIT
```

While optional, it's strongly recommended to also add `upstream` remote with project upstream to allow easy initial patches branch setup, cherry-picking and some extra `rdopkg` automagic detection:

```
git remote add -f upstream $UPSTREAM_GIT
```

Clean .spec

In this example we'll assume we'll building a package for EL 7 distribution and will use `el7` branch for our distgit:

```
git checkout el7
```

Clean the `.spec` file. Replace hardcoded version strings (especially in `URL`) with macros so that `.spec` is current when `Version` changes. Check `rdopkg pkgenv` to see what `rdopkg` thinks about your package:

```
editor foo.spec
rdopkg pkgenv
git commit -a
```

Prepare patches branch

By convention, rdopkg expects `$BRANCH` distgit branch to have appropriate `$BRANCH-patches` patches branch.

Thus, for our `el7` distgit, we need to create `el7-patches` branch.

First, see current `Version:`:

```
rdopkg pkgenv | grep Version
```

Assume our package is at `Version: 1.2.3`.

`upstream` remote should contain associated `1.2.3` version tag which should correspond to `1.2.3` release tarball so let's use that as a base for our new patches branch:

```
git checkout -b el7-patches 1.2.3
```

Finally, if you have some `.patch` files in your `el7` distgit branch, you need to apply them on top `el7-patches` now.

Some patches might be present in `upstream` remote (like backports) so you can `git cherry-pick` them.

Once happy with your patches on top of `1.2.3`, push your patches branch into the `patches` remote:

```
git push patches el7-patches
```

Update distgit

With `el7-patches` patches branch in order, try updating your `distgit`:

```
git checkout el7
rdopkg patch
```

If this fails, you can try lower-level `rdopkg update-patches` which skips certain magics but isn't reccommended normal usage.

Once this succeeds, inspect newly created commit that updated the `.spec` file and `.patch` files from `el7-patches` patches branch.

Ready to rdopkg

After this, you should be able to manage your package using `rdopkg`.

Please note that both `rdopkg patch` and `rdopkg new-version` will **reset local** `el7-patches` to remote `patches/el7-patches` unless you supply `-l` / `--local-patches` option.

To **introduce/remove patches**, simply modify remote `el7-patches` patches branch and let `rdopkg patch` do the rest:

```
rdopkg patch
```

To **update your package to new upstream version** including patches rebase:

```
git fetch --all
rdopkg new-version
```

Finally, if you just want to **fix your .spec** file without touching patches:

```
rdopkg fix
# edit .spec
rdopkg -c
```

More information

List all `rdopkg` actions with:

```
rdopkg -h
```

Most `rdopkg` actions have some handy options, see them with

```
rdopkg $ACTION -h
```

Read the friendly manual (https://github.com/openstack-packages/rdopkg/blob/master/doc/rdopkg.1.adoc):

```
man rdopkg
```

You can also read RDO packaging guide (https://www.rdoproject.org/documentation/rdo-packaging/#rdo-cloudsig-packaging-guide) which contains some examples of `rdopkg` usage in RDO.

Happy packaging!

March 09, 2017 04:58 PM (http://rdoproject.org/blog/2017/03/let-rdopkg-manage-your-RPM-package/)

Mirantis (https://www.mirantis.com)

Kubernetes Replication Controller, Replica Set and Deployments: Understanding replication options (https://www.mirantis.com/blog/kubernetes-replication-controller-replica-set-and-deployments-understanding-replication-options/)

The post Kubernetes Replication Controller, Replica Set and Deployments: Understanding replication options (https://www.mirantis.com/blog/kubernetes-replication-controller-replica-set-and-deployments-understanding-replication-options/) appeared first on Mirantis | Pure Play Open Cloud (https://www.mirantis.com).

As a container management tool, Kubernetes (https://www.mirantis.com/software/kubernetes/) was designed to orchestrate multiple containers and replication, and in fact there are currently several ways to do it. In this article, we'll look at three options: Replication Controllers, Replica Sets, and Deployments.

What is Kubernetes replication for?

Before we go into how you would do replication, let's talk about why. Typically you would want to replicate your containers (and thereby your applications) for several reasons, including:

- **Reliability:** By having multiple versions of an application, you prevent problems if one or more fails. This is particularly true if the system replaces any containers that fail.
- **Load balancing:** Having multiple versions of a container enables you to easily send traffic to different instances to prevent overloading of a single instance or node. This is something that Kubernetes does out of the box, making it extremely convenient.
- **Scaling:** When load does become too much for the number of existing instances, Kubernetes enables you to easily scale up your application, adding additional instances as needed.

Replication is appropriate for numerous use cases, including:

- **Microservices-based applications:** In these cases, multiple small applications provide very specific functionality.
- **Cloud native applications:** Because cloud-native applications are based on the theory that any component can fail at any time, replication is a perfect environment for implementing them, as multiple instances are baked into the architecture.
- **Mobile applications:** Mobile applications can often be architected so that the mobile client interacts with an isolated version of the server application.

Kubernetes has multiple ways in which you can implement replication.

Types of Kubernetes replication

In this article, we'll discuss three different forms of replication: the Replication Controller, Replica Sets, and Deployments.

Replication Controller

The Replication Controller is the original form of replication in Kubernetes. It's being replaced by Replica Sets, but it's still in wide use, so it's worth understanding what it is and how it works. A Replication Controller is a structure that enables you to easily create multiple pods, then make sure that that number of pods always exists. If a pod does crash, the Replication Controller replaces it. Replication Controllers also provide other benefits, such as the ability to scale the number of pods, and to update or delete multiple pods with a single command. You can create a Replication Controller with an imperative command, or declaratively, from a file. For example, create a new file called `rc.yaml` and add the following text:

```
apiVersion: v1
kind: ReplicationController
metadata:
  name: soaktestrc
spec:
  replicas: 3
  selector:
    app: soaktestrc
  template:
    metadata:
      name: soaktestrc
      labels:
        app: soaktestrc
    spec:
      containers:
      - name: soaktestrc
        image: nickchase/soaktest
        ports:
        - containerPort: 80
```

Most of this structure should look familiar from our discussion of Deployments (https://www.mirantis.com/blog/introduction-to-yaml-creating-a-kubernetes-deployment/); we've got the name of the actual Replication Controller (`soaktestrc`) and we're designating that we should have 3 replicas, each of which are defined by the template. The selector defines how we know which pods belong to this Replication Controller. Now tell Kubernetes to create the Replication Controller based on that file:

```
# kubectl create -f rc.yaml
replicationcontroller "soaktestrc" created
```

Let's take a look at what we have using the describe command:

```
# kubectl describe rc soaktestrc
Name:           soaktestrc
Namespace:      default
Image(s):       nickchase/soaktest
Selector:       app=soaktestrc
Labels:         app=soaktestrc
Replicas:       3 current / 3 desired
Pods Status:    3 Running / 0 Waiting / 0 Succeeded / 0 Failed
No volumes.
Events:
  FirstSeen     LastSeen        Count   From                            SubobjectPath   Type    Reason                  Message
  ---------     --------        -----   ----                            -------------   ----    ------                  -------
  1m            1m              1       {replication-controller }                       Normal  SuccessfulCreate Created pod: soaktestrc-g
  1m            1m              1       {replication-controller }                       Normal  SuccessfulCreate Created pod: soaktestrc-c
  1m            1m              1       {replication-controller }                       Normal  SuccessfulCreate Created pod: soaktestrc-r
```

As you can see, we've got the Replication Controller, and there are 3 replicas, of the 3 that we wanted. All 3 of them are currently running. You can also see the individual pods listed underneath, along with their names. If you ask Kubernetes to show you the pods, you can see those same names show up:

```
# kubectl get pods
NAME               READY     STATUS    RESTARTS   AGE
soaktestrc-cws05   1/1       Running   0          3m
soaktestrc-g5snq   1/1       Running   0          3m
soaktestrc-ro2bl   1/1       Running   0          3m
```

Next we'll look at Replica Sets, but first let's clean up:

```
# kubectl delete rc soaktestrc
replicationcontroller "soaktestrc" deleted

# kubectl get pods
```

As you can see, when you delete the Replication Controller, you also delete all of the pods that it created.

Replica Sets

Replica Sets are a sort of hybrid, in that they are in some ways more powerful than Replication Controllers, and in others they are less powerful. Replica Sets are declared in essentially the same way as Replication Controllers, except that they have more options for the selector. For example, we could create a Replica Set like this:

```
apiVersion: extensions/v1beta1
 kind: ReplicaSet
 metadata:
   name: soaktestrs
 spec:
   replicas: 3
   selector:
     matchLabels:
       app: soaktestrs
   template:
     metadata:
       labels:
         app: soaktestrs
         environment: dev
     spec:
       containers:
       - name: soaktestrs
         image: nickchase/soaktest
         ports:
         - containerPort: 80
```

In this case, it's more or less the same as when we were creating the Replication Controller, except we're using `matchLabels` instead of `label` . But we could just as easily have said:

```
...
spec:
   replicas: 3
   selector:
     matchExpressions:
      - {key: app, operator: In, values: [soaktestrs, soaktestrs, soaktest]}
      - {key: teir, operator: NotIn, values: [production]}
  template:
     metadata:
...
```

In this case, we're looking at two different conditions:

1. The app label must be soaktestrc, soaktestrs, or soaktest
2. The tier label (if it exists) must not be production

Let's go ahead and create the Replica Set and get a look at it:

```
# kubectl create -f replicaset.yaml
replicaset "soaktestrs" created

# kubectl describe rs soaktestrs
Name:           soaktestrs
Namespace:      default
Image(s):       nickchase/soaktest
Selector:       app in (soaktest,soaktestrs),teir notin (production)
Labels:         app=soaktestrs
Replicas:       3 current / 3 desired
Pods Status:    3 Running / 0 Waiting / 0 Succeeded / 0 Failed
No volumes.
Events:
  FirstSeen     LastSeen        Count   From                            SubobjectPath   Type    Reason                  Message
  ---------     --------        -----   ----                            -------------   ----    ------                  -------
  1m            1m              1       {replicaset-controller }                        Normal  SuccessfulCreate Created pod: soaktestrs-
  1m            1m              1       {replicaset-controller }                        Normal  SuccessfulCreate Created pod: soaktestrs-k
  1m            1m              1       {replicaset-controller }                        Normal  SuccessfulCreate Created pod: soaktestrs-

# kubectl get pods
NAME               READY     STATUS    RESTARTS   AGE
soaktestrs-8i4ra   1/1       Running   0          1m
soaktestrs-it2hf   1/1       Running   0          1m
soaktestrs-kimmm   1/1       Running   0          1m
```

As you can see, the output is pretty much the same as for a Replication Controller (except for the selector), and for most intents and purposes, they are similar. The major difference is that the `rolling-update` command works with Replication Controllers, but won't work with a Replica Set. This is because Replica Sets are meant to be used as the backend for Deployments. Let's clean up before we move on.

```
# kubectl delete rs soaktestrs
replicaset "soaktestrs" deleted

# kubectl get pods
```

Again, the pods that were created are deleted when we delete the Replica Set.

Deployments

Deployments are intended to replace Replication Controllers. They provide the same replication functions (through Replica Sets) and also the ability to rollout changes and roll them back if necessary. Let's create a simple Deployment using the same image we've been using. First create a new file, `deployment.yaml`, and add the following:

```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: soaktest
spec:
  replicas: 5
  template:
    metadata:
      labels:
        app: soaktest
    spec:
      containers:
      - name: soaktest
        image: nickchase/soaktest
        ports:
        - containerPort: 80
```

Now go ahead and create the Deployment:

```
# kubectl create -f deployment.yaml
deployment "soaktest" created
```

Now let's go ahead and describe the Deployment:

```
# kubectl describe deployment soaktest
Name:                   soaktest
Namespace:              default
CreationTimestamp:      Sun, 05 Mar 2017 16:21:19 +0000
Labels:                 app=soaktest
Selector:               app=soaktest
Replicas:               5 updated | 5 total | 5 available | 0 unavailable
StrategyType:           RollingUpdate
MinReadySeconds:        0
RollingUpdateStrategy:  1 max unavailable, 1 max surge
OldReplicaSets:         <none>
NewReplicaSet:          soaktest-3914185155 (5/5 replicas created)
Events:
  FirstSeen     LastSeen        Count   From                            SubobjectPath   Type    Reason                  Message
  ---------     --------        -----   ----                            -------------   ------------------              -------
  38s           38s             1       {deployment-controller }                        Normal  ScalingReplicaSet       Scaled up replic
  36s           36s             1       {deployment-controller }                        Normal  ScalingReplicaSet       Scaled up replic
```

As you can see, rather than listing the individual pods, Kubernetes shows us the Replica Set. Notice that the name of the Replica Set is the Deployment name and a hash value. A complete discussion of updates is out of scope for this article — we'll cover it in the future — but couple of interesting things here:

- The StrategyType is RollingUpdate. This value can also be set to Recreate.
- By default we have a `minReadySeconds` value of `0`; we can change that value if we want pods to be up and running for a certain amount of time — say, to load resources — before they're truly considered "ready".
- The `RollingUpdateStrategy` shows that we have a limit of 1 `maxUnavailable` — meaning that when we're updating the Deployment, we can have up to 1 missing pod before it's replaced, and 1 `maxSurge`, meaning we can have one extra pod as we scale the new pods back up.

As you can see, the Deployment is backed, in this case, by Replica Set `soaktest-3914185155`. If we go ahead and look at the list of actual pods...

```
# kubectl get pods
NAME                         READY     STATUS    RESTARTS   AGE
soaktest-3914185155-7gyja    1/1       Running   0          2m
soaktest-3914185155-lrm20    1/1       Running   0          2m
soaktest-3914185155-o28px    1/1       Running   0          2m
soaktest-3914185155-ojzn8    1/1       Running   0          2m
soaktest-3914185155-r2pt7    1/1       Running   0          2m
```

... you can see that their names consist of the Replica Set name and an additional identifier.

Passing environment information: identifying a specific pod

Before we look at the different ways that we can affect replicas, let's set up our deployment so that we can see what pod we're actually hitting with a particular request. To do that, the image we've been using displays the pod name when it outputs:

```
<?php
$limit = $_GET['limit'];
if (!isset($limit)) $limit = 250;
for ($i; $i < $limit; $i++){
     $d = tan(atan(tan(atan(tan(atan(tan(atan(tan(atan(123456789.123456789)))))))))));
}
echo "Pod ".$_SERVER['POD_NAME']." has finished!\n";
?>
```

As you can see, we're displaying an environment variable, `POD_NAME`. Since each container is essentially it's own server, this will display the name of the pod when we execute the PHP. Now we just have to pass that information to the pod. We do that through the use of the Kubernetes Downward API, which lets us pass environment variables into the containers:

```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: soaktest
spec:
  replicas: 3
  template:
    metadata:
      labels:
        app: soaktest
    spec:
      containers:
      - name: soaktest
        image: nickchase/soaktest
        ports:
        - containerPort: 80
        env:
        - name: POD_NAME
          valueFrom:
            fieldRef:
              fieldPath: metadata.name
```

As you can see, we're passing an environment variable and assigning it a value from the Deployment's metadata. (You can find more information on metadata here (https://kubernetes.io/docs/tasks/configure-pod-container/downward-api-volume-expose-pod-information/).) So let's go ahead and clean up the Deployment we created earlier...

```
# kubectl delete deployment soaktest
deployment "soaktest" deleted

# kubectl get pods
```

... and recreate it with the new definition:

```
# kubectl create -f deployment.yaml
deployment "soaktest" created
```

Next let's go ahead and expose the pods to outside network requests (https://www.mirantis.com/blog/introduction-to-yaml-creating-a-kubernetes-deployment/) so we can call the nginx server that is inside the containers:

```
# kubectl expose deployment soaktest --port=80 --target-port=80 --type=NodePort
service "soaktest" exposed
```

Now let's describe the services we just created so we can find out what port the Deployment is listening on:

```
# kubectl describe services soaktest
Name:                   soaktest
Namespace:              default
Labels:                 app=soaktest
Selector:               app=soaktest
Type:                   NodePort
IP:                     11.1.32.105
Port:                   <unset> 80/TCP
NodePort:               <unset> 30800/TCP
Endpoints:              10.200.18.2:80,10.200.18.3:80,10.200.18.4:80 + 2 more...
Session Affinity:       None
No events.
```

As you can see, the `NodePort` is `30800` in this case; in your case it will be different, so make sure to check. That means that each of the servers involved is listening on port `30800`, and requests are being forwarded to port `80` of the containers. That means we can call the PHP script with:

```
http://[HOST_NAME OR HOST_IP]:[PROVIDED PORT]
```

In my case, I've set the IP for my Kubernetes hosts to hostnames to make my life easier, and the PHP file is the default for nginx, so I can simply call:

```
# curl http://kube-2:30800
Pod soaktest-3869910569-xnfme has finished!
```

So as you can see, this time the request was served by pod `soaktest-3869910569-xnfme`.

Recovering from crashes: Creating a fixed number of replicas

Now that we know everything is running, let's take a look at some replication use cases. The first thing we think of when it comes to replication is recovering from crashes. If there are 5 (or 50, or 500) copies of an application running, and one or more crashes, it's not a catastrophe. Kubernetes improves the situation further by ensuring that if a pod goes down, it's replaced. Let's see this in action. Start by refreshing our memory about the pods we've got running:

```
# kubectl get pods
NAME                        READY     STATUS    RESTARTS   AGE
soaktest-3869910569-qqwqc   1/1       Running   0          11m
soaktest-3869910569-qu8k7   1/1       Running   0          11m
soaktest-3869910569-uzjxu   1/1       Running   0          11m
soaktest-3869910569-x6vmp   1/1       Running   0          11m
soaktest-3869910569-xnfme   1/1       Running   0          11m
```

If we repeatedly call the Deployment, we can see that we get different pods on a random basis:

```
# curl http://kube-2:30800
Pod soaktest-3869910569-xnfme has finished!
# curl http://kube-2:30800
Pod soaktest-3869910569-x6vmp has finished!
# curl http://kube-2:30800
Pod soaktest-3869910569-uzjxu has finished!
# curl http://kube-2:30800
Pod soaktest-3869910569-x6vmp has finished!
# curl http://kube-2:30800
Pod soaktest-3869910569-uzjxu has finished!
# curl http://kube-2:30800
Pod soaktest-3869910569-qu8k7 has finished!
```

To simulate a pod crashing, let's go ahead and delete one:

```
# kubectl delete pod soaktest-3869910569-x6vmp
pod "soaktest-3869910569-x6vmp" deleted

# kubectl get pods
NAME                        READY     STATUS    RESTARTS   AGE
soaktest-3869910569-516kx   1/1       Running   0          18s
soaktest-3869910569-qqwqc   1/1       Running   0          27m
soaktest-3869910569-qu8k7   1/1       Running   0          27m
soaktest-3869910569-uzjxu   1/1       Running   0          27m
soaktest-3869910569-xnfme   1/1       Running   0          27m
```

As you can see, pod `*x6vmp` is gone, and it's been replaced by `*516kx`. (You can easily find the new pod by looking at the AGE column.) If we once again call the Deployment, we can (eventually) see the new pod:

```
# curl http://kube-2:30800
Pod soaktest-3869910569-516kx has finished!
```

Now let's look at changing the number of pods.

Scaling up or down: Manually changing the number of replicas

One common task is to scale up a Deployment in response to additional load. Kubernetes has autoscaling, but we'll talk about that in another article. For now, let's look at how to do this task manually. The most straightforward way is to simply use the scale command:

```
# kubectl scale --replicas=7 deployment/soaktest
deployment "soaktest" scaled

# kubectl get pods
NAME                        READY     STATUS    RESTARTS   AGE
soaktest-3869910569-2w8i6   1/1       Running   0          6s
soaktest-3869910569-516kx   1/1       Running   0          11m
soaktest-3869910569-qqwqc   1/1       Running   0          39m
soaktest-3869910569-qu8k7   1/1       Running   0          39m
soaktest-3869910569-uzjxu   1/1       Running   0          39m
soaktest-3869910569-xnfme   1/1       Running   0          39m
soaktest-3869910569-z4rx9   1/1       Running   0          6s
```

In this case, we specify a new number of replicas, and Kubernetes adds enough to bring it to the desired level, as you can see. One thing to keep in mind is that Kubernetes isn't going to scale the Deployment down to be below the level at which you first started it up. For example, if we try to scale back down to 4...

```
# kubectl scale --replicas=4 -f deployment.yaml
deployment "soaktest" scaled

# kubectl get pods
NAME                        READY     STATUS    RESTARTS   AGE
soaktest-3869910569-l5wx8   1/1       Running   0          11s
soaktest-3869910569-qqwqc   1/1       Running   0          40m
soaktest-3869910569-qu8k7   1/1       Running   0          40m
soaktest-3869910569-uzjxu   1/1       Running   0          40m
soaktest-3869910569-xnfme   1/1       Running   0          40m
```

... Kubernetes only brings us back down to 5, because that's what was specified by the original deployment.

Deploying a new version: Replacing replicas by changing their label

Another way you can use deployments is to make use of the selector. In other words, if a Deployment controls all the pods with a `tier` value of `dev`, changing a pod's `teir` label to `prod` will remove it from the Deployment's sphere of influence. This mechanism enables you to selectively replace individual pods. For example, you might move pods from a dev environment to a production environment, or you might do a manual rolling update, updating the image, then removing some fraction of pods from the Deployment; when they're replaced, it will be with the new image. If you're happy with the changes, you can then replace the rest of the pods. Let's see this in action. As you recall, this is our Deployment:

```
# kubectl describe deployment soaktest
Name:                   soaktest
Namespace:              default
CreationTimestamp:      Sun, 05 Mar 2017 19:31:04 +0000
Labels:                 app=soaktest
Selector:               app=soaktest
Replicas:               3 updated | 3 total | 3 available | 0 unavailable
StrategyType:           RollingUpdate
MinReadySeconds:        0
RollingUpdateStrategy:  1 max unavailable, 1 max surge
OldReplicaSets:         <none>
NewReplicaSet:          soaktest-3869910569 (3/3 replicas created)
Events:
  FirstSeen     LastSeen        Count   From                            SubobjectPath   Type            Reason                  Message
  ---------     --------        -----   ----                            -------------   --------  ------                  -------
  50s           50s             1       {deployment-controller }                        Normal          ScalingReplicaSet       Scaled
```

And these are our pods:

```
# kubectl describe replicaset soaktest-3869910569
Name:           soaktest-3869910569
Namespace:      default
Image(s):       nickchase/soaktest
Selector:       app=soaktest,pod-template-hash=3869910569
Labels:         app=soaktest
                pod-template-hash=3869910569
Replicas:       5 current / 5 desired
Pods Status:    5 Running / 0 Waiting / 0 Succeeded / 0 Failed
No volumes.
Events:
  FirstSeen     LastSeen        Count   From                            SubobjectPath   Type            Reason                  Message
  ---------     --------        -----   ----                            -------------   --------  ------                  -------
  2m            2m              1       {replicaset-controller }                        Normal          SuccessfulCreate        Created
  2m            2m              1       {replicaset-controller }                        Normal          SuccessfulCreate        Created
  2m            2m              1       {replicaset-controller }                        Normal          SuccessfulCreate        Created
  1m            1m              1       {replicaset-controller }                        Normal          SuccessfulCreate        Created
  1m            1m              1       {replicaset-controller }                        Normal          SuccessfulCreate        Created
```

We can also get a list of pods by label:

```
# kubectl get pods -l app=soaktest
NAME                        READY     STATUS    RESTARTS   AGE
soaktest-3869910569-0577c   1/1       Running   0          7m
soaktest-3869910569-8cbo2   1/1       Running   0          6m
soaktest-3869910569-pwlm4   1/1       Running   0          6m
soaktest-3869910569-wje85   1/1       Running   0          7m
soaktest-3869910569-xuhwl   1/1       Running   0          7m
```

So those are our original soaktest pods; what if we wanted to add a new label? We can do that on the command line:

```
# kubectl label pods soaktest-3869910569-xuhwl experimental=true
pod "soaktest-3869910569-xuhwl" labeled

# kubectl get pods -l experimental=true
NAME                        READY     STATUS    RESTARTS   AGE
soaktest-3869910569-xuhwl   1/1       Running   0          14m
```

So now we have one experimental pod. But since the `experimental` label has nothing to do with the selector for the Deployment, it doesn't affect anything. So what if we change the value of the `app` label, which the Deployment **is** looking at?

```
# kubectl label pods soaktest-3869910569-wje85 app=notsoaktest --overwrite
pod "soaktest-3869910569-wje85" labeled
```

In this case, we need to use the overwrite flag because the app label already exists. Now let's look at the existing pods.

```
# kubectl get pods
NAME                        READY     STATUS    RESTARTS   AGE
soaktest-3869910569-0577c   1/1       Running   0          17m
soaktest-3869910569-4cedq   1/1       Running   0          4s
soaktest-3869910569-8cbo2   1/1       Running   0          16m
soaktest-3869910569-pwlm4   1/1       Running   0          16m
soaktest-3869910569-wje85   1/1       Running   0          17m
soaktest-3869910569-xuhwl   1/1       Running   0          17m
```

As you can see, we now have six pods instead of five, with a new pod having been created to replace `*wje85` , which was removed from the deployment. We can see the changes by requesting pods by label:

```
# kubectl get pods -l app=soaktest
NAME                        READY     STATUS    RESTARTS   AGE
soaktest-3869910569-0577c   1/1       Running   0          17m
soaktest-3869910569-4cedq   1/1       Running   0          20s
soaktest-3869910569-8cbo2   1/1       Running   0          16m
soaktest-3869910569-pwlm4   1/1       Running   0          16m
soaktest-3869910569-xuhwl   1/1       Running   0          17m
```

Now, there is one wrinkle that you have to take into account; because we've removed this pod from the Deployment, the Deployment no longer manages it. So if we were to delete the Deployment...

```
# kubectl delete deployment soaktest
deployment "soaktest" deleted
```

The pod remains:

```
# kubectl get pods
NAME                        READY     STATUS    RESTARTS   AGE
soaktest-3869910569-wje85   1/1       Running   0          19m
```

You can also easily replace all of the pods in a Deployment using the –all flag, as in:

```
# kubectl label pods --all app=notsoaktesteither --overwrite
```

But remember that you'll have to delete them all manually!

Conclusion

Replication is a large part of Kubernetes' purpose in life, so it's no surprise that we've just scratched the surface of what it can do, and how to use it. It is useful for reliability purposes, for scalability, and even as a basis for your architecture. What do you anticipate using replication for, and what would you like to know more about? Let us know in the comments!
The post Kubernetes Replication Controller, Replica Set and Deployments: Understanding replication options (https://www.mirantis.com/blog/kubernetes-replication-controller-replica-set-and-deployments-understanding-replication-options/) appeared first on Mirantis | Pure Play Open Cloud (https://www.mirantis.com).

by Nick Chase at March 09, 2017 01:40 PM (https://www.mirantis.com/blog/kubernetes-replication-controller-replica-set-and-deployments-understanding-replication-options/)

OpenStack Superuser (http://superuser.openstack.org)

How a research group avoids Franken-infrastructure with OpenStack (http://superuser.openstack.org/articles/hybrid-infrastructure-hpc-openstack/)

The Van Andel Institute (VAI) is an independent biomedical research and science education organization based in Grand Rapids, Michigan. VAI hosts thirty individual research groups who use genomic sequencing analysis, molecular dynamics simulation and modeling to investigate epigenetics, cancer and neurodegenerative diseases.

Recently, VAI (https://www.vai.org/) deployed a unique new hybrid infrastructure featuring Bright Computing (http://www.brightcomputing.com/) OpenStack systems management software. The Bright OpenStack deployment wrapper (https://www.openstack.org/marketplace/distros/distribution/bright-computing/bright-openstack) lets VAI manage both high-performance compute (HPC) grid and cluster computing and cloud computing within the same infrastructure, greatly reducing the labor and effort needed for management and change control. Perhaps more importantly, it also helps VAI respond dynamically to the accelerating trend toward cloud computing they see coming down the highway. "Grid and cluster computing has been the standard for years, but we know that cloud computing is the wave of the future," said Zack Ramjan, research computing architect at VAI. "The hybrid approach we are getting with Bright is providing a path that helps us transition from one to the other."

The Challenge

Ramjan considers his main challenge finding a way to meld hardware, software and storage into a system that can handle the massive amount of data generated by VAI's varied research groups (https://www.eurekalert.org/pub_releases/2017-01/vari-nsr011217.php). To solve that challenge, he sought an environment that could handle the massively parallel processing (MPP) and analysis produced. "That's where Bright comes in," says Ramjan. "We have the cluster and grid and will eventually have the cloud, so we can use a "divide and conquer" approach, efficiently assigning tasks among 50 computers and the cloud resources."

> ORC as Loader of the Rings: Study details ringed structure of ORC in DNA replication – https://t.co/2jJCXiVZB3 (https://t.co/2jJCXiVZB3)
>
> — Van Andel Institute (@VAInstitute) February 21, 2017 (https://twitter.com/VAInstitute/status/834068684301799425)

<script async="async" charset="utf-8" src="http://platform.twitter.com/widgets.js"></script>

Another challenge he faces is that the 30 different research groups (https://vari.vai.org/vari-research/) are working on highly varied projects, each with unique requirements. Some of the researchers are doing genomics and others are doing simulation – and they needed a solution that would work for all of them. Ramjan explains that access to a cloud approach puts the user in the driver seat, so he would not have to develop a single solution that makes everyone happy. Each user can have their own solution, carving off their own virtual piece of the pie. “Bright is helping us manage that variety of approaches. With both the legacy cluster mode and the cloud mode, we are creating an environment in which it is easy for users to come on board and do their specific work efficiently.”

> “We know that cloud computing is the wave of the future. The hybrid approach we are getting with Bright is providing a path that helps us transition. ” — Zack Ramjan research computing architect at VAI

The Solution

The total solution includes three key components: Bright OpenStack software; 43 compute nodes, representing 1,100 CPU cores, provided by Silicon Mechanics; along with storage and data hosting supplied by Data Direct Network (http://www.ddn.com/). Bright OpenStack is the brains of the system and integrates the hardware, managing the CPU units and the storage devices. The user interacts with the Bright dashboard, and Bright OpenStack interacts with the physical elements of the system.

> We helped Van Andel Institute save 2 years of dev time with our OpenStack HPC system. Read the case study: https://t.co/K8lBm8Y8ia (https://t.co/K8lBm8Y8ia)
>
> — Silicon Mechanics (@ExpertIncluded) January 17, 2017 (https://twitter.com/ExpertIncluded/status/821435422580637696)

<script async="async" charset="utf-8" src="http://platform.twitter.com/widgets.js"></script>

An expert with many years of deploying HPC systems by hand, Ramjan decided to opt for the Bright solution because he didn’t have the time or the manpower in house to quickly set up a hybrid HPC-cloud system. He reasoned that engineering its solution would take years. “We are actually kind of lucky that we started from nothing because we had no legacy baggage, so we could design the solution as we saw fit from the beginning, and we did that with the help of Bright and others.” Bright partner and system Integrator for the project Silicon Mechanics (http://www.siliconmechanics.com/) was confident the Bright tool set would be a successful way to tackle the challenge. “As we worked with VAI to define their system architecture, we looked to the Bright OpenStack system management software, knowing its strength in managing complex, private cloud computing environments,” said Daniel Chow (https://www.linkedin.com/in/daniel-chow-3a1b701), COO/CTO at Silicon Mechanics. “We are excited to see the joint solution empower VAI researchers to accelerate their scientific findings.”

A key feature of the Bright OpenStack software is the ease of management and change control. Ramjan notes that it is very difficult and time consuming to scale up by managing many machines one by one. “With Bright you can see the entire resource, or see your hardware through a single pane of glass. Right now our HPC to cloud ratio may be 90-10, but tomorrow we know our end users are going to be more cloud-centric. With this solution, I can dynamically pull resources into the cloud portion, but if the next day it turns out there’s less cloud demand, we can pull it back. We can dynamically shift that ratio as we see fit without any down time.” According to Ramjan, several other institutions he has spoken to also have their eyes on transitioning to an OpenStack cloud. However, those larger institutions with significant investments in “old school” HPC infrastructure say they are probably three to five years away from doing so. “Because they have such a complex environment, everything has been custom designed in-house and changes must be done with their own labor, which makes them less flexible. With Bright, we bought the solution and everything came with the package. It definitely put us way ahead of the game.”

The Result

The HPC cluster went online in September 2015 and is already highly utilized, which Ramjan considers a good sign. Bright makes it easy for other team members to jump in to provide assistance to users without needing his intervention. He says that Linux can be quirky technically, so without Bright OpenStack, even basic tasks would have fallen solely in his lap, which would have slowed down deployment considerably.

“We know that scientific workloads do not get smaller every year, but are constantly expanding. From our experience, the size of data continues to grow exponentially. We have more than 40 compute nodes representing 1,100 CPU cores today – but what about next year when we get to 2,000 or 3,000 cores? We wanted an expandable and scalable solution – this is a core capability of Bright. The management tool makes it easy to buy new equipment, take it out of the box, put it on the shelf, and plug it in. Bright can pull it right into the existing environment.”

> The search is on for ways to mend hearts before they break: https://t.co/3ZdEllyjZm (https://t.co/3ZdEllyjZm) #congenitalheartdefect (https://twitter.com/hashtag/congenitalheartdefect?src=hash) #CHDAware (https://twitter.com/hashtag/CHDAware?src=hash)
>
> — Van Andel Institute (@VAInstitute) February 14, 2017 (https://twitter.com/VAInstitute/status/831586060429905920)

<script async="async" charset="utf-8" src="http://platform.twitter.com/widgets.js"></script>

The cloud portion has recently gone live and is available to VAI's users. Ramjan says that VAI's early adopters, typically power-users who are quite savvy, are starting to appreciate its value. He expects use of the cloud to grow in popularity as others get more familiar with the resource. Although VAI's existing workloads are 90 percent grid and cluster, they expect to move toward the cloud in the future. Bright is giving them an expandable and scalable turnkey solution that lets them combine HPC workloads with big data analytics workloads in the same infrastructure, and to have the choice of working in either a bare metal or virtualized infrastructure. It's also providing a path that helps them transition from one to the other.

This case study first appeared on www.brightcomputing.com. Superuser is always interested in community content, get in touch at editor@superuser.org

[Cover Photo](https://www.flickr.com/photos/le0nard0h0/14534898165/) // CC [BY NC](https://creativecommons.org/licenses/by-nc/2.0/)

The post How a research group avoids Franken-infrastructure with OpenStack (http://superuser.openstack.org/articles/hybrid-infrastructure-hpc-openstack/) appeared first on OpenStack Superuser (http://superuser.openstack.org).

by Superuser at March 09, 2017 12:58 PM (http://superuser.openstack.org/articles/hybrid-infrastructure-hpc-openstack/)

Red Hat Stack (http://redhatstackblog.redhat.com)

Using Software Factory to manage Red Hat OpenStack Platform lifecycle (http://redhatstackblog.redhat.com/2017/03/08/using-software-factory-to-manage-red-hat-openstack-platform-lifecycle/)

by Nicolas Hicher, Senior Software Engineer – Continuous Integration and Delivery

Software-Factory

Software-Factory is a collection of services that provides a powerful platform to build software. It enables the same workflow used to develop OpenStack: using Gerrit for code reviews, Zuul (https://docs.openstack.org/infra/zuul/)/Nodepool/Jenkins as a CI system, and Storyboard for stories and issues tracker. Also, it ensures a reproducible test environment with ephemeral Jenkins slaves.

In this video, Nicolas Hicher will demonstrate how to use Software-Factory (https://softwarefactory-project.io) to manage a Red Hat OpenStack Platform 9 lifecycle. We will do a deployment and an update on a virtual environment (within an OpenStack tenant).

<iframe allowfullscreen="true" class="youtube-player" height="390" src="http://www.youtube.com/embed/jtuZiYHd2Aw?version=3&rel=1&fs=1&autohide=2&showsearch=0&showinfo=1&iv_load_policy=1&wmode=transparent" style="border:0;" type="text/html" width="640"></iframe>

Python-tripleo-helper

For this demo, we will do a deployment within an OpenStack tenant. Using a tool, developed by the engineering team that builds DCI (http://redhatstackblog.redhat.com/2016/06/30/who-is-testing-your-cloud/), called python-tripleo-helper (https://github.com/redhat-openstack/python-tripleo-helper). With this tool, we can do a deployment within an OpenStack tenant using the same steps of a full deployment (boot server via IPMI, discover nodes, introspection and deployment). We also patched python-tripleo-helper to add an update command to update the OpenStack (changing parameters, not doing a major upgrade).

Workflow

The workflow is simple and robust:

- Submit a review with the templates, the installation script and the tests scripts. A CI job validates the templates.
- When the review is approved, the gate jobs are executed (installation or update).
- After the deployment/update is completed, the review is merged.

Deployment

For this demo, we will do a simple deployment (1 controller and 1 compute nodes) with Red Hat OpenStack 9.0

Limitations

Since we do the deployment in a virtual environment, we can't test some advanced features, especially for networking and storage. But other features of the deployed cloud can be validated using the appropriate environments.

Improvements

We plan to continue to improve this workflow to be able to:

- Do a major upgrade from Red Hat OpenStack Platform (X to X+1).
- Manage a bare metal deployment.
- Improve the Ceph deployment to be able to use more than one object storage device (OSD).
- Use smoke jobs like tempest to validate the deployment before merging the review.

Also, it should be possible to manage pre-production and production environments within a single git repository, the check job will do the tasks on pre production and after receiving a peer's validation, the same actions will be applied on production.

by Maria Bracho, Senior Product Manager OpenStack at March 09, 2017 01:59 AM (http://redhatstackblog.redhat.com/2017/03/08/using-software-factory-to-manage-red-hat-openstack-platform-lifecycle/)

March 08, 2017

Mirantis (https://www.mirantis.com)

What is the best NFV Orchestration platform? A review of OSM, Open-O, CORD, and Cloudify (https://www.mirantis.com/blog/which-nfv-orchestration-platform-best-review-osm-open-o-cord-cloudify/)

The post What is the best NFV Orchestration platform? A review of OSM, Open-O, CORD, and Cloudify (https://www.mirantis.com/blog/which-nfv-orchestration-platform-best-review-osm-open-o-cord-cloudify/) appeared first on Mirantis | Pure Play Open Cloud (https://www.mirantis.com).

As Network Functions Virtualization (NFV) technology matures, multiple NFV orchestration solutions have emerged and 2016 was a busy year. While some commercial products were already available on the market, multiple open source initiatives were also launched, with most delivering initial code releases, and others planning to roll-out software artifacts later this year. With so much going on, we thought we'd provide you with a technical overview of some of the various NFV orchestration options, so you can get a feel for what's right for you. In particular, we'll cover:

- Open Source MANO (OSM) (https://osm.etsi.org/)
- OPEN-O (https://www.open-o.org/)
- CORD (http://opencord.org/)
- Gigaspaces Cloudify (http://docs.getcloudify.org/)

In addition, multiple NFV projects have been funded under European Union R&D programs. Projects such as OpenBaton, T-NOVA/TeNor and SONATA have their codebases available in public repos, but industry support, involvement of external contributors and further sustainability might be a challenging for these projects, so for now we'll consider them out of scope for this post, where we'll review and compare orchestration projects across the following areas:

- General overview and current project state
- Compliance with NFV MANO reference architecture
- Software architecture
- NSD definition approach
- VIM and VNFM support
- Capabilities to provision End to End service
- Interaction with relevant standardization bodies and communities

General overview and current project state

We'll start with a general overview of each project, along with, its ambitions, development approach, the involved community, and related information.

OSM

The OpenSource MANO project was officially launched at the World Mobile Congress (WMC) in 2016. Starting with several founding members, including Mirantis, Telefónica, BT, Canonical, Intel, RIFT.io, Telekom Austria Group and Telenor, the OSM community now includes 55 different organisations. The OSM project is hosted at ETSI facilities and targets delivering an open source management and orchestration (MANO) stack closely aligned with the ETSI NFV reference architecture. OSM issued two releases,Rel 0 and Rel 1, during 2016. The most recent at the time of this writing, OSM Rel. 1, has been publicly available since October, 2016, and can be downloaded from the official website (https://osm.etsi.org). Project governance is managed via several groups, including the Technical Steering group responsible for OSM's technical aspects, the Leadership group, and the End User Advisory group. You can find more details about OSM project may be found at the official Wiki (https://osm.etsi.org/wikipub/index.php/Main_Page).

OPEN-O

The OPEN-O project (https://www.open-o.org/) is hosted by the Linux foundation and was also formally announced at 2016 MWC. Initial project advocates were mostly from Asian companies, such as Huawei, ZTE and China Mobile. Eventually, the project project got further support from Brocade, Ericsson, GigaSpaces, Intel and others. The main project objective is to enable end-to-end service agility across multiple domains using unified platform for NFV and SDN orchestration. The OPEN-O project delivered its first release in November, 2016 plans to roll-out future releases in a 6 month cycle. Overall project governance is managed by the project Board, with technology-specific issues managed by the Technical Steering Committee. You can find more general details about the OPEN-O project may be found at the project web-site (https://www.open-o.org/).

CORD/XOS

Originally CORD (Central Office Re-architected as a Datacenter) was introduced as one of the use cases for the ONOS SDN Controller, but it grew-up into a separate project under ON.Lab governance. (ON.Lab recently merged with the Open Networking Foundation.) The ultimate project goal is to combine NFV, SDN and the elasticity of commodity clouds to bring datacenter economics and cloud agility to the Telco Central Office (http://opencord.org/). The reference implementation of CORD combines commodity servers, white-box switches, and disaggregated access technologies with open source software to provide an extensible service delivery platform. CORD Rel.1 and Rel.2 integrate a number of open source projects, such as ONOS to manage SDN infrastructure, OpenStack to deploy NFV workloads, and XOS as a service orchestrator. To reflect use cases' uniqueness, CORD introduces a number of service profiles, such as Mobile (M-CORD), Residential (R-CORD), and Enterprise (E-CORD). You can find more details about CORD project can be found at the official project web site (http://opencord.org/).

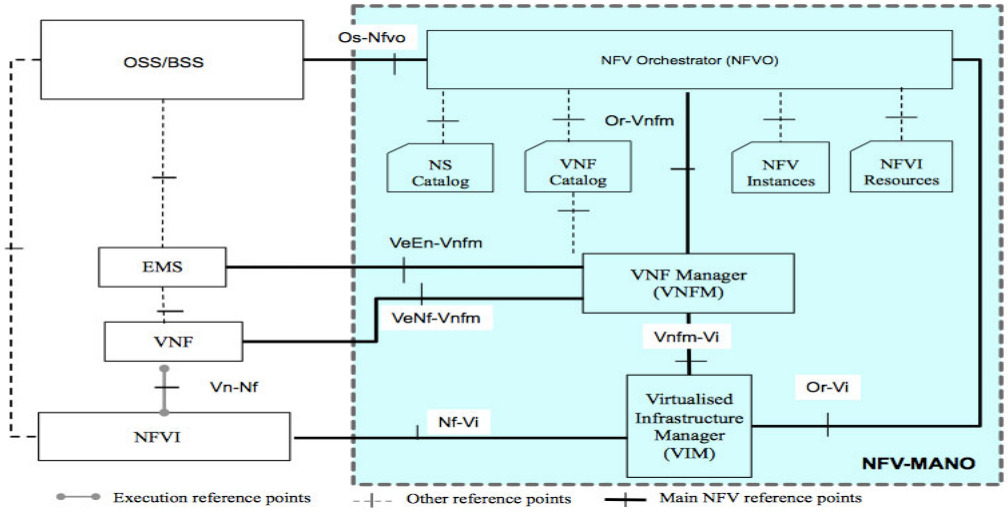
Gigaspaces Cloudify

Gigaspaces' Cloudify (http://docs.getcloudify.org/) is the open source TOSCA-based cloud orchestration software platform. Originally introduced as a pure cloud orchestration solution (similar to OpenStack HEAT), the platform was further expanded to include NFV-related use cases, and the Cloudify Telecom Edition emerged. Considering its original platform purpose, Cloudify has an extensible architecture and can interact with multiple IaaS/PaaS providers such as AWS, OpenStack, Microsoft Azure and so on. Overall, Cloudify software is open source under the Apache 2 license and the source code is hosted in a public repository. While the Cloudify platform is open source and welcomes community contributions, the overall project roadmap is defined by Gigaspaces. You can find more details about the Cloudify platform at the official web site (http://docs.getcloudify.org/).

Compliance with ETSI NFV MANO reference architecture

At the time of this writing, a number of alternatives and specific approaches, such as Lifecycle Service Orchestration (LSO) from Metro Ethernet Forum, have emerged, huge industry support and involvement has helped to promote ETSI NFV Management and Orchestration (MANO) (http://www.etsi.org/deliver/etsi_gs/NFV-

MAN/001_099/001/01.01.01_60/gs_NFV-MAN001v010101p.pdf) as the de-facto reference NFV architecture. From this standpoint, NFV MANO provides comprehensive guidance for entities, reference points and workflows to be implemented by appropriate NFV platforms (fig. 1):



(https://cdn.mirantis.com/wp-content/uploads/2017/03/ETSI-NFV-MANO-reference-architecture.png) Figure 1 – ETSI NFV MANO reference architecture

OSM

As this project is hosted by ETSI, the OSM community tries to be compliant with the ETSI NFV MANO reference architecture, respecting appropriate IFA working group specifications. Key reference points, such as Or-Vnfm and Or-Vi might be identified within OSM components. The VNF and Network Service (NS) catalog are explicitly present in an OSM service orchestrator (SO) component. Meanwhile, a lot of further development efforts are planned to reach feature parity with currently specified features and interfaces.

OPEN-O

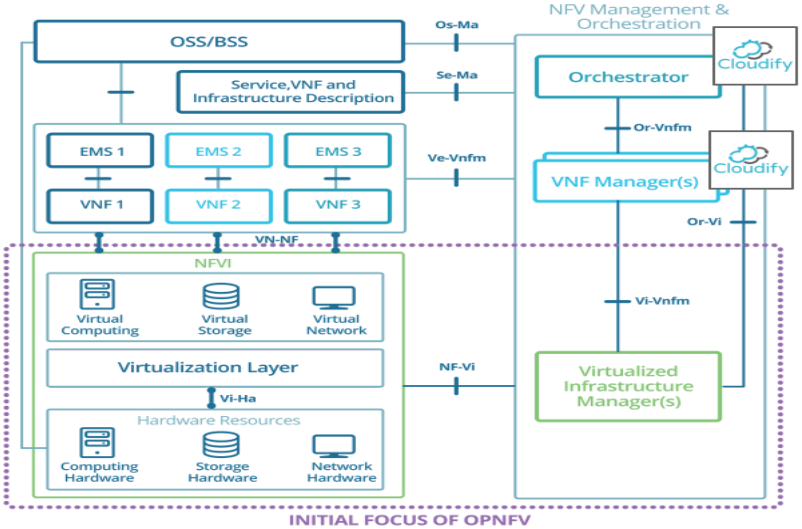
While the OPEN-O project in general has no objective to be compliant with NFV MANO, the NFVO component of OPEN-O is aligned with an ETSI reference model, and all key MANO elements, such as VNFM and VIM might be found in an NFVO architecture. Moreover, the scope of the OPEN-O project goes beyond just NFV orchestration, and as a result goes beyond the scope identified by the ETSI NFV MANO reference architecture. One important piece of this project relates to SDN-based networking services provisioning and orchestration, which might be further used either in conjunction with NFV services or as a standalone feature.

CORD

Since its invention, CORD has defined its own reference architecture and cross-component communication logic. The reference CORD implementation is very OpenFlow-centric around ONOS, the orchestration component (XOS), and whitebox hardware. Technically, most of the CORD building blocks might be mapped to MANO-defined NFVI, VIM and VNFM, but this is incidental; the overall architectural approach defined by ETSI MANO, as well as the appropriate reference points and interfaces were not considered in scope by the CORD community. Similar to OPEN-O, the scope of this project goes beyond just NFV services provisioning. Instead, NFV services provisioning is considered as one of the several possible use cases for the CORD platform.

Gigaspaces Cloudify

The original focus of the Cloudify platform was orchestration of application deployment in a cloud. Later, when the NFV use case emerged, the Telecom Edition of the Cloudify platform was delivered. This platform combines both NFVO and generic VNFM components of the MANO defined entities (fig. 2).



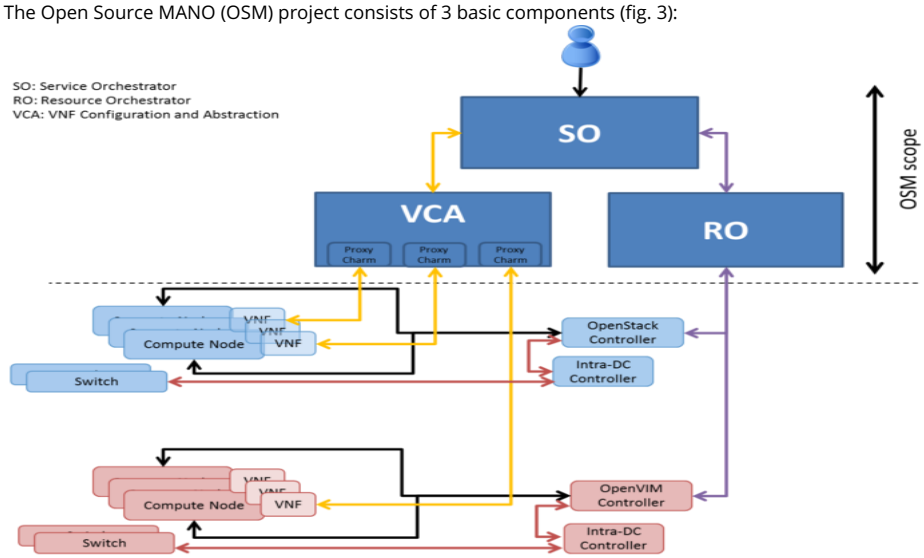
(https://cdn.mirantis.com/wp-content/uploads/2017/03/Cloudify-in-relation-to-a-NFV-MANO-reference-architecture.png) Figure 2 – Cloudify in relation to a NFV MANO reference architecture (http://getcloudify.org/network-function-virtualization-vnf-nfv-orchestration-sdn-platform.html) By its very nature, Cloudify Blueprints might be considered as the NS and VNF catalog entities defined by MANO. Meanwhile, some interfaces and actions specified by the NFV IFA subgroup are not present or considered as out of scope for the Cloudify platform. From this standpoint, you could say that Cloudify is aligned with the MANO reference architecture but not fully compliant.

Software architecture and components

As you might expect, all NFV Orchestration solutions are complex integrated software platforms combined from multiple components.

OSM

The Open Source MANO (OSM) project consists of 3 basic components (fig. 3):



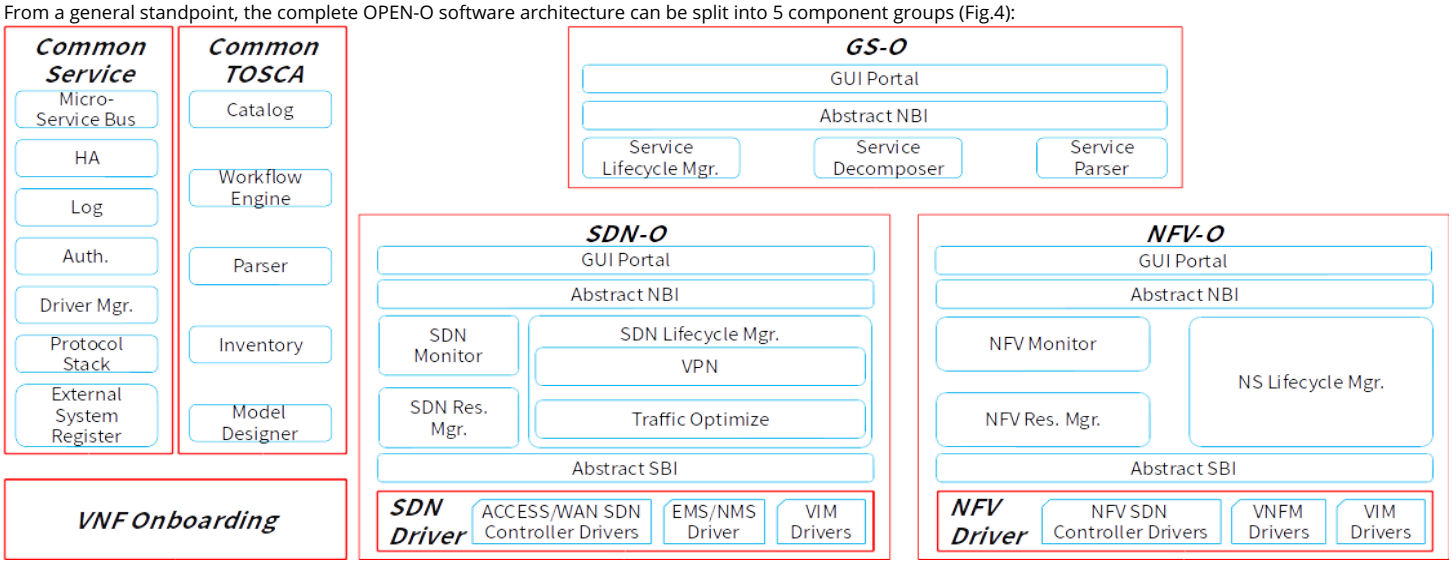
(https://cdn.mirantis.com/wp-content/uploads/2017/03/OSM-project-architecture.png) Figure 3 – OSM project architecture

- The **Service Orchestrator (SO)**, responsible for end-to-end service orchestration and provisioning. The SO stores the VNF definitions and NS catalogs, manages workflow of the service deployment and can query the status of already deployed services. OSM integrates the rift.io orchestration engine as an SO.
- The **Resource Orchestrator (RO)** is used to provision services over a particular IaaS provider in a given location. At the time if this writing, the RO component is capable of deploying networking services over OpenStack, VMware, and OpenVIM. The SO and RO components can be jointly mapped to the NFVO entity in the ETSI MANO architecture
- The **VNF Configuration and Abstraction (VCA)** module performs the initial VNF configuration using Juju Charms. Considering this purpose, the VCA module can be considered as a generic VNFM with a limited feature set.

Additionally, OSM hosts the OpenVIM project, which is a lightweight VIM layer implementation suitable for small NFV deployments as an alternative to heavyweight OpenStack or VMware VIMs. Most of the software components are developed in python, while SO, as a user facing entity, heavily relies on a JavaScript and NodeJS framework.

OPEN-O

From a general standpoint, the complete OPEN-O software architecture can be split into 5 component groups (Fig.4):



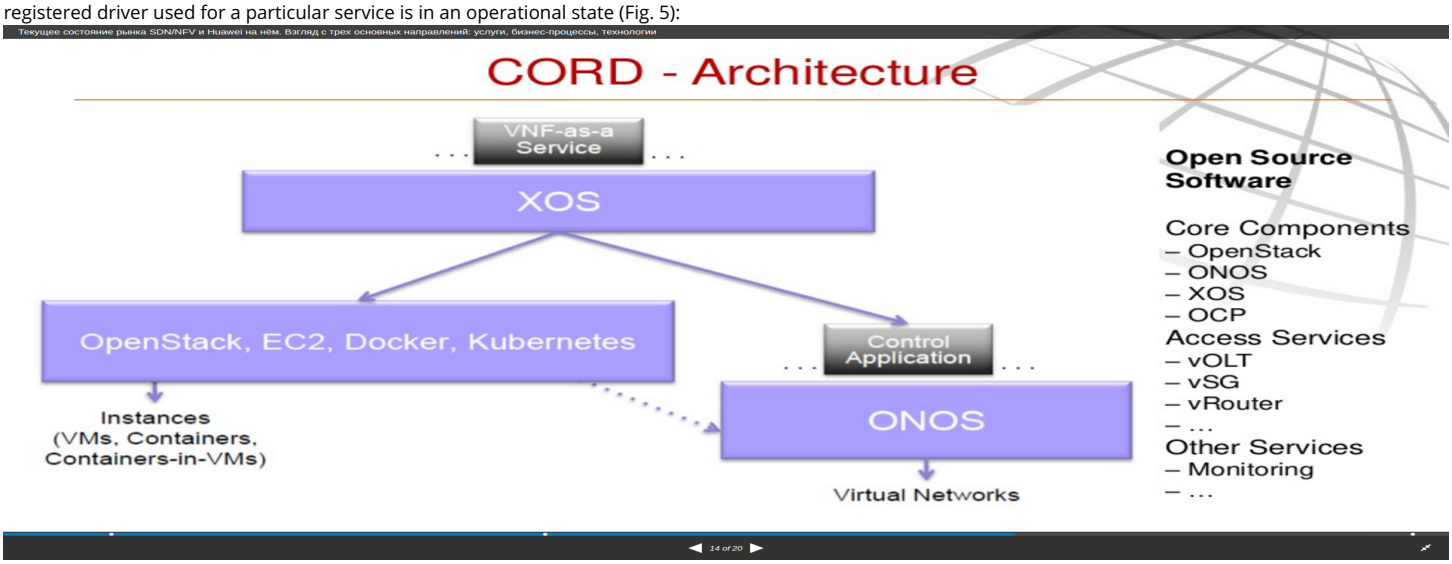
(https://cdn.mirantis.com/wp-content/uploads/2017/03/OPEN-O-project-software-architecture.png) Figure 4 – OPEN-O project software architecture

- **Common service:** Consists of shared services used by all other components.
- **Common TOSCA:** Provides TOSCA-related features such as NSD catalog management, NSD definition parsing, workflow execution, and so on; this component is based on the ARIA TOSCA (http://ariatosca.org/) project.
- **Global Service Orchestrator (GSO):** As the name suggests, this group provides overall lifecycle management of the end-to-end service.
- **SDN Orchestrator (SDN-O):** Provides abstraction and lifecycle management of SDN services; an essential piece of this block are the SDN drivers, which provide device-specific modules for communication with a particular device or SDN controller.
- **NFV Orchestrator (NFV-O):** This group provides NFV services instantiation and lifecycle management.

The OPEN-O project uses a microservices-based architecture, and consists of more than 20 microservices. The central platform element is the Microservice Bus, which is the core microservice of the Common Service components group. Each platform component should register with this bus. During registration, each microservice specifies exposed APIs and endpoint addresses. As a result, the overall software architecture is flexible and can be easily extended with additional modules. OPEN-O Rel. 1 consists of both Java and python-based microservices.

CORD/XOS

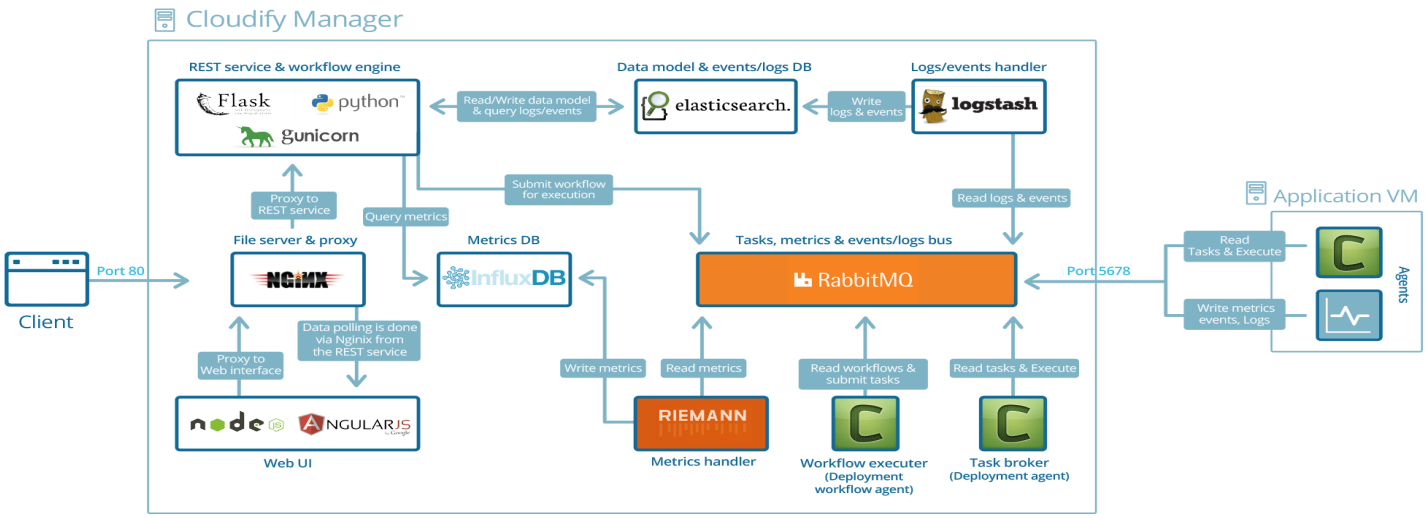
As mentioned above, CORD was introduced originally as an ONOS application, but grew into a standalone platform that covers both ONOS-managed SDN regions and service orchestration entities implemented by XOS. Both ONOS and XOS provide a service framework to enable the Everything-as-a-Service (XaaS) concept. Thus, the reference CORD implementation consists of both a hardware Pod (consisting of whitebox switches and servers) and a software platform (such as ONOS or XOS with appropriate applications). From the software standpoint, the CORD platform implements an agent or driver-based approach in which XOS ensures that each registered driver used for a particular service is in an operational state (Fig. 5):



(https://cdn.mirantis.com/wp-content/uploads/2017/03/CORD-platform-architecture.png) Figure 5 – CORD platform architecture The CORD reference implementation consists of Java (ONOS and its applications) and python (XOS) software stacks. Additionally, Ansible is heavily used by the CORD for automation and configuration management

Gigaspaces Cloudify

From the high-level perspective, platform consists of several different pieces, as you can see in figure 6:



(https://cdn.mirantis.com/wp-content/uploads/2017/03/Cloudify-platform-architecture.png) Figure 6 – Cloudify platform architecture

- **Cloudify Manager** is the orchestrator that performs deployment and lifecycle management of the applications or NSDs described in the templates, called blueprints.
- **The Cloudify Agents** are used to manage workflow execution via an appropriate plugin.

To provide overall lifecycle management, Cloudify integrates third-party components such as:

- **Elasticsearch,** used as a data store of the deployment state, including runtime data and logs data coming from various platform components.
- **Logstash,** used to process log information coming from platform components and agents.
- **Riemann,** used as a policy engine to process runtime decisions about availability, SLA and overall monitoring.
- **RabbitMQ,** used as an async transport for communication among all platform components, including remote agents.

The orchestration functionality itself is provided by the ARIA TOSCA project (http://ariatosca.org/), which defines the TOSCA-based blueprint format and deployment workflow engine. Cloudify “native” components and plugins are python applications.

Approach for NSD definition

The Network Service Descriptor (NSD) specifies components and the relations between them to be deployed on the top of the IaaS during the NFV service instantiation. Orchestration platforms typically use some templating language to define NSDs. While the industry in general considers TOSCA (http://docs.oasis-open.org/tosca/tosca-nfv/v1.0/tosca-nfv-v1.0.html) as a de-facto standard to define NSDs, alternative approaches are also available across various platforms.

OSM

OSM follows the official MANO specification (http://www.etsi.org/deliver/etsi_gs/NFV-MAN/001_099/001/01.01.01_60/gs_NFV-MAN001v010101p.pdf), which has definitions both for NSDs and VNF Descriptors (VNFD). To define NSD templates, YAML-based documents are used. NSD is processed by the OSM Service Orchestrator to instantiate a Network Service, which itself might include VNFs, Forwarding Graphs, and Links between them. A VNFD is a deployment template that specifies a VNF in terms of deployment and operational behaviour requirements. Additionally VNFD specifies connections between Virtual Deployment Units (VDUs) using the internal Virtual Links (VLs). Each VDU in an OSM presentation relates to a VM or a Container. OSM uses archived format both for NSD and VNFD. This archive consists of the service/VNF description, initial configuration scripts and other auxiliary details. You can find more information about OSM NSD/VNFD structure at the official website (https://osm.etsi.org/wikipub/index.php/Release_0_Data_Model_Details).

OPEN-O

In OPEN-O, the TOSCA-based templates is used to describe the NS/VNF Package. Both the TOSCA general service profile (http://docs.oasis-open.org/tosca/TOSCA-Simple-Profile-YAML/v1.0/csprd02/TOSCA-Simple-Profile-YAML-v1.0-csprd02.html) and the more recent NFV profile (http://docs.oasis-open.org/tosca/tosca-nfv/v1.0/tosca-nfv-v1.0.html) can be used for NSD/VNFD, which is further packaged according to the the Cloud Service Archive (CSAR) format. A CSAR is a zip archive that contains at least two directories: TOSCA-Metadata and Definitions. The TOSCA-Metadata directory contains information that describes the content of the CSAR and is referred to as the TOSCA metafile. The Definitions directory contains one or more TOSCA Definitions documents. These Definitions documents contain definitions of the cloud application to be deployed during CSAR processing. More details about OPEN-O NSD/VNFD definitions may be found at the official web site (https://www.open-o.org/).

CORD/XOS

To define a new CORD service, you need to define both TOSCA-based templates and Python-based software components. Particularly when adding a new service, depending on its nature, you might alter one of several platform elements:

- **TOSCA service definition files**, appropriate models, specified as YAML text files
- **REST APIs models**, specified in Python
- **XOS models**, implemented as a django application
- **Synchronizers,** used to ensure the Service instantiated correctly and transitioned to the required state.

The overall service definition format is based on the TOSCA Simple Profile language specification (http://docs.oasis-open.org/tosca/TOSCA-Simple-Profile-YAML/v1.0/csprd02/TOSCA-Simple-Profile-YAML-v1.0-csprd02.html) and presented in the YAML format.

Gigaspaces Cloudify

To instantiate a service or application, Cloudify uses templates called “Blueprints” which are effectively orchestration and deployment plans. Blueprints are specified in the form of TOSCA YAML files (http://docs.oasis-open.org/tosca/TOSCA-Simple-Profile-YAML/v1.0/csprd02/TOSCA-Simple-Profile-YAML-v1.0-csprd02.html) and describe the service topology as a set of nodes, relationships, dependencies, instantiation and configuration settings, monitoring, and maintenance. Other than the YAML itself, a Blueprint can include multiple external resources such as configuration and installation scripts (or Puppet Manifests, or Chef Recipes, and so on) and basically any other resource required to run the application. You can find more details about the structure of Blueprints here (http://docs.getcloudify.org/3.4.0/blueprints/overview/).

VNFM and VIM support

NFV service deployment is performed on the appropriate IaaS, which itself is a set of virtualized compute, network and storage resources. The ETSI MANO reference architecture identifies a component to manage these virtualized resources. This component is referred to as the Virtual Infrastructure Manager (VIM). Traditionally, the open source community treats OpenStack/KVM as a “de-facto” standard VIM. However, an NFV service might be span across various VIM types and various hypervisors. Thus multi-VIM support is a common requirement for an Orchestration engine. Additionally, a separate element in a NFV MANO architecture is the VNF Manager, which is responsible for lifecycle management of the particular VNF. The VNFM component might be either generic, treating the VNF as a black box and performing similar operations for various VNFs, or there might be a vendor-specific VNFM that has unique capabilities for management of a given VNF. Both VIM and VNFM communication are performed via appropriate reference points, as defined by the NFV MANO architecture.

OSM

The OSM project was initially considered a multi-VIM platform, and at the time of this writing, it supports OpenStack, Vmware and OpenVIM. OpenVIM is a lightweight VIM implementation that is effectively a python wrapper around libvirt and a basic host networking configuration. At the time of this writing, the OSM VCA has limited capabilities, but still can be considered a generic VNFM based on JuJu Charms. Further, it is possible to introduce support for vendor-specific VNFMs, but additional development and integration efforts might be required on the Service Orchestrator (Rift.io) side.

OPEN-O

Release 1 of the OPEN-O project supports only OpenStack as a VIM. This support is available as a Java-based driver for the NFVO component. For further releases, support for VMware as a VIM is planned. The Open-O Rel.1 platform has a generic VNFM that is based on JuJu Charms. Furthermore, the pluggable architecture of the OPEN-O platform can support any vendor-specific VNFM, but additional development and integration efforts will be required.

CORD/XOS

At the time of this writing the reference implementation of the CORD platform is architectured around OpenStack as a platform to spawn NFV workloads. While there is no direct relationship to the NFV MANO architecture, the XOS orchestrator is responsible for VNF lifecycle management, and thus might be thought of as the entity that provides VNFM-like functions.

Gigaspaces Cloudify

When Cloudify was adapted for the NFV use case, it inherited plugins for OpenStack, VMware, Azure and others that were already available for general-purpose cloud deployments. So we can say that Cloudify has MultiVIM support and any arbitrary VIM support may be added via the appropriate plugin. Following Gigaspaces’ reference model for NFV (http://getcloudify.org/network-function-virtualization-vnf-nfv-orchestration-sdn-platform.html), there is a generic VNFM that can be used with a Cloudify NFV orchestrator out of the box. Additional vendor-specific VNFM can be onboarded, but appropriate plugin development is required.

Capabilities to provision end-to-end service

NFV service provisioning consists of multiple steps, such as VNF instantiation, configuration, underlay network provisioning, and so on. Moreover, an NFV service might span multiple clouds and geographical locations. This kind of architecture requires complex workflow management by an NFV Orchestrator, and coordination

and synchronisation between infrastructure entities. This section provides an overview of the various orchestrators' abilities to provision end-to-end service.

OSM

The OSM orchestration platform supports NFV service deployment spanning multiple VIMs. In particular, the OSM RO component (openmano) stores information about all VIMs available for deployment, while the Service Orchestrator can use this information during the NSD instantiation process. Meanwhile, underlay networking between VIMs should be preconfigured. There are plans to enable End-to-End network provisioning in future, but OSM Rel. 1 has no such capability.

OPEN-O

By design, the OPEN-O platform considers both NFV and SDN infrastructure regions that might be used to provision end-to-end service. So technically, you can say that Multisite NFV service can be provisioned by OPEN-O platform. However, the OPEN-O Rel.1 platform implements just a couple of specific use cases, and at the time of this writing, you can't use it to provision an arbitrary Multisite NFV service.

CORD/XOS

The reference implementation of the CORD platform defines the provisioning of a service over a defined CORD Pod. To enable Multisite NFV Service instantiation, an additional orchestration level on the top of CORD/XOS is required. So from this perspective, at the time of this writing, CORD is not capable of instantiating a Multisite NFV service.

Gigaspaces Cloudify

As Cloudify originally supported application deployment over multiple IaaS providers, technically it is possible to create a blueprint to deploy an NFV service that spans across multiple VIMs. However underlay network provisioning might require specific plugin development.

Interaction with standardization bodies and relevant communities

Each of the reviewed projects has strong industry community support. Depending on the nature of each community and the priorities of the project, there is a different focus on collaboration with an industry, other open source projects and standardization bodies.

OSM

Being hosted by ETSI, the OSM project closely collaborates with the ETSI NFV working group and follows the appropriate specifications, reference points and interfaces. At the time of this writing there are no collaborations between OSM in the scope of the OPNFV project, but it is under consideration by the OSM community. The same relates to other relevant open source projects, such as OpenStack and OpenDaylight; these projects are used "AS-IS" by the OSM platform without cross collaboration.

OPEN-O

The OPEN-O project aims to integrate both SDN and NFV solutions to provide end-to-end service, so there is formal communication to the ETSI NFV group, while the project itself doesn't strictly follows interfaces defined by the ETSI NFV IFA working group. On other hand there is strong integration effort with the OPNFV community via initiation of the OPERA project (https://wiki.opnfv.org/display/PROJ/Opera+Project), which aims to integrate the OPEN-O platform as a MANO orchestrator for the OPNFV platform. Additionally there is strong interaction between OPEN-O and MEF as a part of the OpenLSO platform (https://www.mef.net/openlso-opencs/openlso-and-opencs-overview), and the ONOS project towards seamless integration and enabling end-to-end SDN Orchestration.

CORD/XOS

Having originated at the On.LAB (recently merged with ONF) this project follows the approach and technology stack defined by ONF. As of the time of this writing, the CORD project has no formal presence in OPNFV. Meanwhile, there is communication with MEF and ONF towards requirements gathering and use cases for the CORD project. In particular, MEF explicitly refers to E-CORD and its applicability for defining their OpenCS MEF project (https://wiki.mef.net/display/CESG/OpenCS).

Gigaspaces Cloudify

While the Cloudify platform is an open source product, it is mostly developed by a single company, thus the overall roadmap and community strategy is defined by Gigaspaces. This also relates to any collaboration with standardisation bodies: GigaSpaces participates in ETSI-approved NFV PoCs where Cloudify is used as a service orchestrator, and in an MEF-initiated LSO Proof of Concept, where Cloudify is used to provision E-Line EVPL service, and so on. Additionally, the Cloudify platform is used separately by the OPNFV community in the FuncTest project for vIMS test cases (https://www.mirantis.com/blog/opnfv-functional-testing-tosca-orchestration-and-vimsusecases/), but this mostly relates to Cloudify use cases, rather than vendor-initiated community collaboration.

Conclusions

Summarising the current state of the NFV orchestration platforms, we may conclude the following: The OSM platform is already suitable for evaluation purposes, and has relatively simple and straightforward architecture. Several sample NSDs and VNFDs are available for evaluation in the public gerrit repo. As a result, the platform can be easily installed and integrated with an appropriate VIM to evaluate basic NFV capabilities, trial use cases and PoCs. The project is relatively young, however, and a number of features still require development and will be available in upcoming releases. Furthermore, lack of support for end-to-end NFV service provisioning across multiple regions, including underlay network provisioning, should be considered in relation to your desired use case. Considering mature OSM community and close interaction with ETSI NFV group this project might emerge as a viable option for production-grade NFV Orchestration. At the time of this writing, the main visible benefit of the OPEN-O platform is the flexible and extendable microservices-based architecture. The OPEN-O approach considers End-to-End service provisioning spanning multiple SDN and NFV regions from the very beginning. Additionally, the OPEN-O project actively collaborates with the OPNFV community toward tight integration of the Orchestrator with OPNFV platform. Unfortunately, at the time of this writing, the OPEN-O platform requires further development to be capable of providing arbitrary NFV service provisioning. Additionally a lack of documentation makes it hard to understand the microservice logic and the interaction workflow. Meanwhile, the recent OPEN-O and ECOMP merge (https://www.open-o.org/news/news/2017/02/linux-foundation-announces-merger-open-source-ecomp-and-open-o-form-new-open) under the ONAP (https://www.onap.org/) project creates powerful open source community with strong industry support, which may reshape the overall NFV orchestration market. The CORD project is the right option when OpenFlow and whiteboxes are the primary option for computing and networking infrastructure. The platform considers multiple use cases, and a large community is involved in platform development. Meanwhile, at the time of this writing, the CORD platform is a relatively "niche" solution around OpenFlow and related technologies pushed to the market by ONF. Gigaspaces Cloudify is a platform that already has a relatively long history, and at the time of this writing emerges as the most mature orchestration solution among the reviewed platforms. While the NFV use case for a Cloudify platform wasn't originally considered, Cloudify's pluggable and extendable architecture and embedded workflow engine enables arbitrary NFV service provisioning. However, if you do consider Cloudify as an orchestration engine, be sure to consider the risk of having the decision-making process regarding the overall platform strategy controlled solely by Gigaspaces.

References

1. OSM official website (https://osm.etsi.org/)
2. OSM project wiki (https://osm.etsi.org/wikipub/index.php/Main_Page)
3. OPEN-O project official website (https://www.open-o.org/)
4. CORD project official website (http://opencord.org/)
5. Cloudify platform official website (http://docs.getcloudify.org/)
6. Network Functions Virtualisation (NFV); Management and Orchestration (http://www.etsi.org/deliver/etsi_gs/NFV-MAN/001_099/001/01.01.01_60/gs_NFV-MAN001v010101p.pdf)
7. Cloudify approach for NFV Management & Orchestration (http://getcloudify.org/network-function-virtualization-vnf-nfv-orchestration-sdn-platform.html)
8. ARIA TOSCA project (http://ariatosca.org/)
9. TOSCA Simple Profile Specification (http://docs.oasis-open.org/tosca/TOSCA-Simple-Profile-YAML/v1.0/csprd02/TOSCA-Simple-Profile-YAML-v1.0-csprd02.html)
10. TOSCA Simple Profile for Network Functions Virtualization (http://docs.oasis-open.org/tosca/tosca-nfv/v1.0/tosca-nfv-v1.0.html)
11. OPNF OPERA project (https://wiki.opnfv.org/display/PROJ/Opera+Project)
12. OpenCS project (https://wiki.mef.net/display/CESG/OpenCS)
13. MEF OpenLSO and OpenCS projects (https://www.mef.net/openlso-opencs/openlso-and-opencs-overview)
14. OPNFV vIMS functional testing (https://www.mirantis.com/blog/opnfv-functional-testing-tosca-orchestration-and-vimsusecases/)
15. OSM Data Models; NSD and VNFD format (https://osm.etsi.org/wikipub/index.php/Release_0_Data_Model_Details)
16. Cloudify Blueprint overview (http://docs.getcloudify.org/3.4.0/blueprints/overview/)

The post What is the best NFV Orchestration platform? A review of OSM, Open-O, CORD, and Cloudify (https://www.mirantis.com/blog/which-nfv-orchestration-platform-best-review-osm-open-o-cord-cloudify/) appeared first on Mirantis | Pure Play Open Cloud (https://www.mirantis.com).

by Guest Post at March 08, 2017 07:36 PM (https://www.mirantis.com/blog/which-nfv-orchestration-platform-best-review-osm-open-o-cord-cloudify/)

NFVPE @ Red Hat (https://blog.nfvpe.site)

Kuryr-Kubernetes will knock your socks off! (http://dougbtv.com//nfvpe/2017/03/08/kuryr-kubernetes/)

Seeing kuryr-kubernetes (https://github.com/openstack/kuryr-kubernetes) in action in my “Dr. Octagon NFV laboratory (https://github.com/dougbtv/droctagon-ansible)” has got me feeling that barefoot feeling (https://www.youtube.com/watch?v=kuprjBXuRls) – and henceforth has completely knocked my socks off. Kuryr-Kubernetes provides Kubernetes integration with OpenStack networking, and today we’ll walk through the steps so you can get your own instance up of it up and running so you can check it out for yourself. We’ll spin up kuryr-kubernetes with devstack, create some pods and a VM, inspect Neutron and verify the networking is working a charm.

by Doug Smith at March 08, 2017 05:01 PM (http://dougbtv.com//nfvpe/2017/03/08/kuryr-kubernetes/)

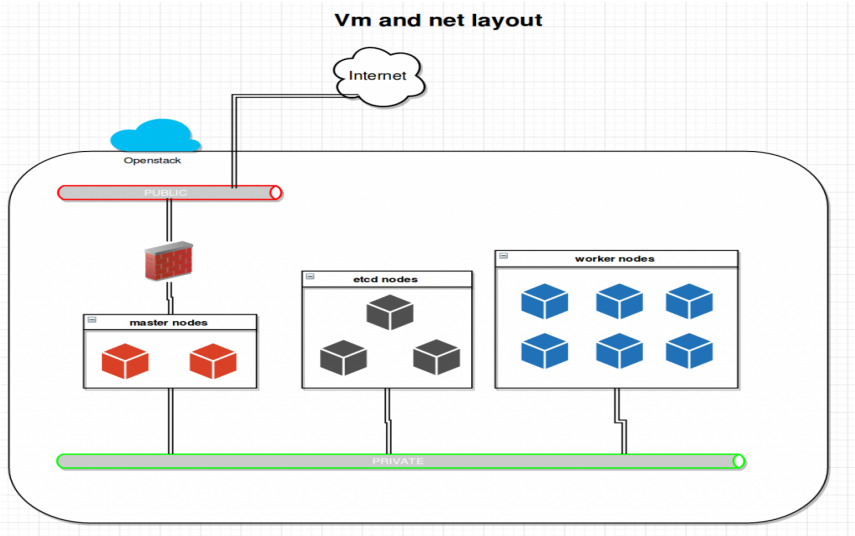
OpenStack Superuser (http://superuser.openstack.org)

How Kubernetes on OpenStack powers DreamHost’s new web builder (http://superuser.openstack.org/articles/kubernetes-openstack-dreamhost-remixer/)

When it comes to establishing a web presence, many small businesses really just need simple, great-looking web pages. They don’t have the time or the interest to learn how to use complicated site-building tools.

To address this need, DreamHost developed Remixer (https://www.dreamhost.com/website-builder/), an easy-to-use graphic interface to build websites quickly and easily. The application runs on a Kubernetes cluster deployed on top of DreamHost’s OpenStack-based cloud computing resource, DreamCompute (https://www.dreamhost.com/cloud/computing).

Thanks to OpenStack, the Kubernetes (https://kubernetes.io/) cluster is deployed with Terraform (https://www.terraform.io/), without needing extra plugins or weird extensions. Terraform uses OpenStack APIs to create the virtual machines and network topology necessary for Kubernetes, including all security rules, the persistent volumes for etcd (https://coreos.com/etcd/docs/latest/) and to store application’s logs. The whole cluster is created with a simple configuration script.



(http://superuser.openstack.org/wp-content/uploads/2017/03/image00.png)

Kubernetes itself runs on custom Container Linux (previously known as CoreOS)-based images as a hypercube (https://github.com/kubernetes/kubernetes/tree/master/cluster/images/hyperkube) single-binary, the preferred deployment form of Kubernetes. A python script downloads the basic Container Linux image, configures it for DreamCompute, adds the single hypercube binary, bundles the image, and uploads it in Glance image service.

Remixer (https://techcrunch.com/2017/02/07/dreamhost-launches-its-remixer-website-builder-to-better-compete-with-squarespace-wix-and-others/) runs a custom version of Kubernetes to work around a known issue dealing with OpenStack volumes. DreamHost developers sent a pull request to Kubernetes upstream (https://github.com/kubernetes/kubernetes/pull/40423) to get that fixed, and are hopeful their contribution will improve things for all users in similar situations.

The Kubernetes cluster is made of three types of instances: master nodes, workers, and etcd instances. Many of these instances require persistent volumes from Cinder, and that's the reason for the custom patch.

> Without OpenStack and Kubernetes, Remixer would look very different today or would not exist at all. With the microservices distributed as Docker images, the team can focus on delivering features that add value to the customers instead of imaging bare-metal servers to keep them up with demand.

Master instances are used to provide the high-availability of the main Kubernetes components and they host the ingress controllers used to accept and route external traffic using declarative configuration. The resulting cluster can survive a master node loss. The ingress traffic is also managed by the master nodes, but with the use of an ingress controller which is based on Nginx. (https://www.nginx.com/) Modern browsers will request from the next IP address in the pool if one doesn't answer. As a result, the ingress management is also redundant if all master IPs are added as DNS records to the serviced domain.

Flannel (https://github.com/coreos/flannel) is used to provide network services for the pods across the node border. Every node registers itself in etcd with the help of register_node service and timer. Fleet is used to distribute services in the cluster along with services activated on a particular machine depending on its role.

The etcd cluster doesn't use the etcd discovery service, as the initial hosts for the cluster are generated via Terraform. Etcd instances use volumes to store the cluster data, so they won't go away if the instance is lost. To communicate with the etcd cluster every machine has a proxy etcd instance which is connected to the cluster. The etcd cluster is effectively the localhost for other services on the machine.

Remixer's Kubernetes cluster offers – right from the start – the dashboard, ingress support with automatic TLS certificates management via Let's Encrypt (https://letsencrypt.org/), log collection and reporting, cluster health/performance metrics in InfluxDB shipped from Heapster (https://github.com/kubernetes/heapster), and Grafana (http://grafana.org/) as a UI. The size of the cluster is elastic, with worker nodes added or removed with customer's demand.

<iframe allowfullscreen="allowfullscreen" frameborder="0" height="480" src="https://www.youtube.com/embed/hTcA3ThRqYY" width="854"></iframe>

On top of this infrastructure, everything runs Dockerized for the Remixer application itself. On a not-so-busy day there are over 25 pods in the cluster, each running a microservice with specific tasks. The Celery-based (http://www.celeryproject.org/) worker pods take care of more resource-intensive tasks, like publishing the website and building the page previews. Web pods take care of delivering the front-end application, based on Flux React (https://facebook.github.io/flux/), while the Nginx pods act as reverse proxies for the Python backend, to integrate with DreamHost APIs to create domains, manage billing, etc. Remixer also requires pods for cache (using Redis (https://redis.io/)) and a RabbitMQ (https://www.rabbitmq.com/) pod.

To complete the app, the DreamHost team designed from the beginning a state-of-the-art analytics service that monitors user experience. This analytics dashboard is designed to help product managers understand how the product is used, in order to minimize user interface pain points and gain visibility on front-end errors. Remixer users are at the center of the development cycle, which is based on hard data.

Without OpenStack and Kubernetes, Remixer would look very different today or would not exist at all. With the microservices distributed as Docker images, the team can focus on delivering features that add value to the customers instead of imaging bare-metal servers to keep up with demand.

Cover Photo (https://www.flickr.com/photos/sebilden/10308971474/) // CC BY NC (https://creativecommons.org/licenses/by-nc/2.0/)

The post How Kubernetes on OpenStack powers DreamHost's new web builder (http://superuser.openstack.org/articles/kubernetes-openstack-dreamhost-remixer/) appeared first on OpenStack Superuser (http://superuser.openstack.org).

by Stefano Maffulli at March 08, 2017 01:01 PM (http://superuser.openstack.org/articles/kubernetes-openstack-dreamhost-remixer/)

SUSE Conversations (https://www.suse.com/communities/blog)

SUSE Expert Days już 4 kwietnia w Warszawie (https://www.suse.com/communities/blog/suse-expert-days-juz-4-kwietnia-w-warszawie/)

Przygotowania do SUSE Expert Days idą pełną parą. Konferencja ta jest jedną z nielicznych w Polsce okazji do zdobycia wiedzy pomocnej w planowaniu przyszłości własnych centrów danych w oparciu o rozwiązania open source przygotowane na potrzeby biznesu. W tym roku można się będzie dowiedzieć na niej, jak dzięki otwartemu oprogramowaniu utrzymać przewagę na konkurencyjnym rynku w ...

+read more (https://www.suse.com/communities/blog/suse-expert-days-juz-4-kwietnia-w-warszawie/)

The post SUSE Expert Days już 4 kwietnia w Warszawie (https://www.suse.com/communities/blog/suse-expert-days-juz-4-kwietnia-w-warszawie/) appeared first on SUSE Blog (https://www.suse.com/communities/blog). Rafal Kruschewski (https://www.suse.com/communities/blog/author/rkruschewski/)

by Rafal Kruschewski at March 08, 2017 12:11 PM (https://www.suse.com/communities/blog/suse-expert-days-juz-4-kwietnia-w-warszawie/)

Opensource.com (https://opensource.com/taxonomy/term/5126/feed/feed)

Does your open source project need a president? (https://opensource.com/article/17/3/governance-needs-president)

Recently I was lucky enough to be invited to attend the Linux Foundation Open Source Leadership Summit (http://events.linuxfoundation.org/events/open-source-leadership-summit). The event was stacked with many of the people I consider mentors, friends, and definitely leaders in the various open source and free software communities that I participate in.

by SpamapS at March 08, 2017 08:00 AM (https://opensource.com/article/17/3/governance-needs-president)

March 07, 2017

OpenStack Blog (https://www.openstack.org/blog)

Helping PTG attendees and other developers get to the OpenStack Summit (https://www.openstack.org/blog/2017/03/helping-ptg-attendees-and-other-developers-get-to-the-openstack-summit/)

Although the OpenStack design events have changed, developers and operators still have a critical perspective to bring to the OpenStack Summits. At the PTG, a common whisper heard in the hallways was, “I really want to be at the Summit, but my [boss/HR/approver] doesn’t understand why I should be there.” To help you out, we took our original “Dear Boss” (https://www.openstack.org/blog/2016/12/dear-boss-i-want-to-attend-the-openstack-summit/) letter and made a few edits for the PTG crowd. If you’re a contributor or developer who wasn’t able to attend the PTG, with a few edits, this letter can also work for you. (Not great with words? Foundation wordsmith Anne can help you out–anne at openstack.org)

Dear **[Boss]**,

I would like to attend the OpenStack Summit in Boston, May 8-11, 2017. At the Pike Project Team Gathering in Atlanta (PTG), I was able to learn more about the new development event model for OpenStack. In the past I attended the Summit to participate in the Design Summit, which encapsulated the feedback and planning as well as design and development of creating OpenStack releases. One challenge was that the Design Summit did not leave enough time for “head down” work within upstream project teams (some teams ended up traveling to team-specific mid-cycle sprints to compensate for that). At the Pike PTG, we were able to kickstart the Pike cycle development, working heads down for a full week. We made great progress on both single project and OpenStack-wide goals, which will improve the software for all users, including our organization.

Originally, I––and many other devs––were under the impression that we no longer needed to attend the OpenStack Summit. However, after a week at the PTG, I see that I have a valuable role to play at the Summit’s “Forum” component. The Forum is where I can gather direct feedback and requirements from operators and users, and express my opinion and our organization’s about OpenStack’s future direction. The Forum will let me engage with other groups with similar challenges, project desires and solutions.

While our original intent may have been to send me only to the PTG, I would strongly like us to reconsider. The Summit is still an integral part of the OpenStack design process, and I think my attendance is beneficial to both my professional development and our organization. Because of my participation in the PTG, I received a free pass to the Summit, which I must redeem by March 14.

Thank you for considering my request.
[Your Name]

by Anne Bertucio at March 07, 2017 08:00 PM (https://www.openstack.org/blog/2017/03/helping-ptg-attendees-and-other-developers-get-to-the-openstack-summit/)

OpenStack Superuser (http://superuser.openstack.org)

Upcoming OpenStack Community Leadership Training (http://superuser.openstack.org/articles/openstack-community-leadership-training/)

Great community leaders are made, not born. With that idea in mind, OpenStack’s Stewardship Working Group (http://superuser.openstack.org/articles/openstack-stewardship-working-group/) is organizing a second edition of Community Leadership Training, April 11-13.

Who should attend?

“Anyone who wants to be involved in the OpenStack Community in any way should attend – that goes for users who want to be active, developers, or even members of any working group,” Colette Alexander (https://www.openstack.org/community/members/profile/13935), who is part of the SWG, tells Superuser.

These small, workshopped sessions are capped at 20 participants. The first edition was offered last year to technical committee members, Foundation staff and board members; sample sessions feature topics including “servant leadership” and “mindfulness in management” while time is also set aside for debriefing and reflection. There held at ZingTrain in Ann Arbor, Michigan. ZingTrain (https://www.zingtrain.com/) has a long track record in areas of leadership work that closely align with the values of democracy, openness and consensus building in the OpenStack community.

If you can confirm your ability to attend, the sign-up is here: https://etherpad.openstack.org/p/Leadershiptraining (https://etherpad.openstack.org/p/Leadershiptraining)

The Etherpad also has further details about timing, place, recommended locations to stay, etc. You can also scroll down to read a sample itinerary of subjects covered in the training. If you have further questions, Alexander suggests asking questions in this mailing list thread (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113267.html), or in #openstack-swg (http://eavesdrop.openstack.org/#OpenStack_Stewardship_Working_Group_Meeting) as many folks who frequent that channel have attended.

A few other things to note:

– This is the **exact** same training that was done last year
– There will be 1-2 attendees from the previous edition to give some context and continuity to the discussions.
– Training costs are fully funded by the Foundation. Attendees need to cover the cost of travel, lodging and some meals (breakfast and lunch during training is provided).
– Deadline for applications is March 24, so please start the work
of getting travel approvals, etc., now.

Any more questions? You can also ping Alexander on IRC at gothicmindfood.

Cover Photo (https://www.flickr.com/photos/kwl/4764667152/) // CC BY NC (https://creativecommons.org/licenses/by-nc/2.0/)

The post Upcoming OpenStack Community Leadership Training (http://superuser.openstack.org/articles/openstack-community-leadership-training/) appeared first on OpenStack Superuser (http://superuser.openstack.org).

by Nicole Martinelli at March 07, 2017 01:03 PM (http://superuser.openstack.org/articles/openstack-community-leadership-training/)

Aptira (https://aptira.com)

OpenStack Australia Day Melbourne – 3 months to go! (https://aptira.com/openstack-australia-day-melbourne-3-months-go/)

The countdown is officially on! Less than 3 months to go until OpenStack Australia Day comes to Melbourne.

This event will feature a range of sessions on the broader cloud and Software Defined Infrastructure ecosystem including OpenStack, containers, PaaS and automation with insights from some of the most talented members of the OpenStack community. Attendees will have the opportunity to hear real business cases, learn about new products, and participate in hands-on workshops, as well as a networking event for a less formal opportunity to engage with the community.

OpenStack Australia Day Melbourne is located at the Rydges Hotel within Melbourne's CBD. Situated in the heart of the city's vibrant theatre district with Chinatown, exclusive Collins Street boutiques and world-famous Bourke Street Mall only moments away. This venue will include free event wi-fi for all attendees, a restaurant featuring the best of Victorian and Australian produce, contemporary bars and on-site accommodation. Accommodation discount codes are available upon request and are subject to availability.

The sponsor and catering area is truly unique. Amazing decor, 3D theming of a bygone era, food carts and a donut wall (yes a DONUT WALL!). Delegates will have the opportunity to interact with the community in a setting unlike any other OpenStack event.

For more information regarding sponsorship, speaker submissions and ticket sales, please visit: http://australiaday.openstack.org.au (http://australiaday.openstack.org.au)

I hope to see you all there!

The post OpenStack Australia Day Melbourne – 3 months to go! (https://aptira.com/openstack-australia-day-melbourne-3-months-go/) appeared first on Aptira Cloud Solutions (https://aptira.com).

by Jessica Field at March 07, 2017 03:35 AM (https://aptira.com/openstack-australia-day-melbourne-3-months-go/)

Hugh Blemings (http://hugh.blemings.id.au)

OpenStack PTG Atlanta 2017 Summary of Summaries (http://hugh.blemings.id.au/2017/03/07/openstack-ptg-atlanta-2017-summary-of-summaries/)

Background

The last couple of editions of Lwood (http://hugh.blemings.id.au/openstack/lwood/) included a list of links to mailing list posts from the preceding week where the writers have provided a summary of particular PTG sessions and/or commentary about the overall event.

Like the previous ones for Austin (http://hugh.blemings.id.au/2016/05/17/openstack-summit-austin-2016-summary-of-summaries/) and Barcelona (http://hugh.blemings.id.au/2016/11/21/openstack-summit-barcelona-2016-summary-of-summaries/), the list below is aggregate of these weekly posts into one readily searchable list.

Summaries posted to the OpenStack-Dev mailing list

- [Acceleration] Atlanta PTG Summary (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113800.html) – Zhipeng Huang
- [All] [API] API-WG PTG recap (http://lists.openstack.org/pipermail/openstack-dev/2017-February/112995.html) – Chris Dent
- [Architecture] Pike PTG Architecture Working Group Recap (http://lists.openstack.org/pipermail/openstack-dev/2017-February/113016.html) – Clint Byrum
- [Cinder] PTG Recap (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113315.html) – Sean McGinnis
- [Docs] Docs PTG summary (http://lists.openstack.org/pipermail/openstack-dev/2017-February/112967.html) – Alexandra Settle
- [Glance] PTG Summary for Wednesday (http://lists.openstack.org/pipermail/openstack-dev/2017-February/112829.html) and Thursday (http://lists.openstack.org/pipermail/openstack-dev/2017-February/112876.html) – Brian Rosmaita
- [Heat] Heat PTG recap (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113436.html) – Rico Lin
- [Horizon] PTG Summary (http://lists.openstack.org/pipermail/openstack-dev/2017-February/112950.html) – Rob Cresswell
- [I8n] Pike PTG Summary (http://lists.openstack.org/pipermail/openstack-dev/2017-February/112889.html) – Ian Y Choi
- [Ironic] Pike PTG report (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113546.html) – Dimitry Tantsur
- [Keystone] Pike PTG Summary (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113068.html) – Lance Bragstad
- [Kuryr] 1st (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113093.html), 2nd (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113176.html) and Final day (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113290.html) recording for the Virtual Team Gathering – Antoni Segura Puimedon
- [Networking-sfc] PTG notes (http://lists.openstack.org/pipermail/openstack-dev/2017-February/113045.html) – Bernard Cafarelli
- [Neutron] PTG Summary (http://lists.openstack.org/pipermail/openstack-dev/2017-February/113032.html) – Kevin Benton
- [Nova][Keystone] Pike PTG recap – quotas (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113068.html) – Lance Bragstad
- [Nova][Keystone] Pike PTG recap – quotas (http://lists.openstack.org/pipermail/openstack-dev/2017-February/112991.html) – Matt Riedemann
- [Nova] Pike PTG recap – notifications (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113098.html) – Balazs Gibizer
- [Nova] Pike PTG recap – API (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113171.html) – Matt Riedemann
- [Nova] Pike PTG recap – Cells (http://lists.openstack.org/pipermail/openstack-dev/2017-February/112996.html) – Matt Riedemann
- [Nova][Cinder] Pike PTG recap – nova/cinder (http://lists.openstack.org/pipermail/openstack-dev/2017-February/113063.html)– Matt Riedemann
- [Manila] PTG summary (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113309.html) – Ben Swartzlander
- [Mistral][PTG] Following up on Pike PTG sessions (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113680.html) – Renat Akhmerov
- [Neutron] Upgrades PTG recap (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113371.html) – Ihar Hrachyshka
- [Octavia] Octavia PTG summary – Cross-project teams Pt1 (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113430.html) & Pt2 (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113432.html) – Michael Johnson
- [Oslo] PTG Summary (http://lists.openstack.org/pipermail/openstack-dev/2017-February/112435.html) – ChangBo Guo
- [Placement][Nova] PTG summary (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113306.html) – Jay Pipes
- [QA] Pike PTG Summary (http://lists.openstack.org/pipermail/openstack-dev/2017-February/112941.html) – Andrea Frittoli
- [Relmgt] Pike PTG recap (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113121.html) – Thierry Carrez
- [Sahara] Pike PTG summary (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113125.html) – Telles Nobrega
- [Storlets] PTG Summary (http://lists.openstack.org/pipermail/openstack-dev/2017-February/112919.html) – Eran Rom
- [Swift] Thoughts on the PTG (http://lists.openstack.org/pipermail/openstack-dev/2017-February/113062.html) – John Dickinson
- [TC][SWG] Summary – Atlanta PTG Stewardship Working Group Session (http://lists.openstack.org/pipermail/openstack-dev/2017-February/112959.html) – Colette Alexander
- [Tripleo] On the road to Pike (http://lists.openstack.org/pipermail/openstack-dev/2017-February/112995.html) – Emilien Macchi
- [Zuul] Zuul v3 – What's Coming: What to expect with the Zuul v3 Rollout (http://lists.openstack.org/pipermail/openstack-dev/2017-February/113065.html) – Monty Taylor

This list will be updated each week with any new summaries that are posted to the list. Additions/corrections welcome.

Updates: 20170314 – Added Acceleration, Heat, Ironic, Mistral, Neutron and Octavia links; 20170313 – Corrected TripleO link;

by hugh at March 07, 2017 03:33 AM (http://hugh.blemings.id.au/2017/03/07/openstack-ptg-atlanta-2017-summary-of-summaries/)

March 06, 2017

Gal Sagie (http://galsagie.github.io)

Submitting a Talk To OpenStack Summit (http://galsagie.github.io//2017/03/06/submitting-a-talk/)

I haven't written a post for some time now, been busy creating something very special which i hope to share about really soon. I usually write in this blog about technical things, and i will continue to do this after this post :) but i wanted to share some of the insights i gained both from being a returning speaker and track chair in the recent OpenStack summits.

I was fortunate to be a track chair in the last two OpenStack summits (And part of the team for OpenStack Israel 2016 and 2017), a task that i personally take very seriously and make sure to devote and allocate the needed time. To you who isn't familiar what a track chair is i suggest you read this link (https://www.openstack.org/summit/tokyo-2015/selection-process/) which describe how a talk is being selected for OpenStack summit

In this post i wanted to share some of the points that i personally think are important for the potential speaker to address and pay attention to, i believe at least some of this is shared by other track chairs and other committees for other conferences and hope that these tips will help the future track chairs make better decisions.

Your Bio

It is extremely important for me to understand who you are, some of the criteria for my selection is the speaker. No matter what title you hold or what your experience is, there is absolutely no reason that your bio will consist of only one sentence.

I want to hear what is your current role, but also what you work on and how it relates to OpenStack (That's already 2-3 sentences). If you are a contributor, or you write a blog, or you presented before, please share those details.

I tend to look up the speakers in order to find this information, but you can make this process much easier if you just supply it :)

The One Speaker Syndrome

OpenStack and i feel open source in general is a team work, it's a community effort and i personally like to see talks with more than one speaker, preferably from different companies.

Its of course not mandatory, and not the highest criteria in this list, but if you are really presenting a topic which is interesting and which the community is working on, i suggest you devote some time and try to find a co-speaker. From my personal experience, when you work in OpenStack it's not that hard to find this person.

Talks with diverse speakers help us track chairs identify that this is obviously a subject interesting for more than one company and let us feel safer that it's not going to be a marketing pitch (Of course, i have been disappointed before...) and make sure we will receive few points of view on the subject.

The Topic

This is a personal preference, but to me topics should be engaging, they should trigger your interest and make you read the abstract.

Give some thought to your topic, try to make it unique and capture the essence of what you are going to present and how it might be different than all the others. You will be surprised how many recurring topics, topics which try to be an abstract (too long) or ones that are not really clear i see...

Picking a good topic is probably an art, try to devote some thinking to it.

The Abstract

The abstract should give potential audience a pretty good and accurate picture of what is going to be presented. (in summary)

Keeping a "mysterious" abstract can be appealing at times, but it makes the audience and the track chair job difficult, how can we determine if this is interesting and should be included if we don't even understand what it is you plan to present.

If there is any information about what you are going to present, like project links, blog posts, code, previous presentations include this in the extra sections you have when submitting a talk.

Try to see if what you are presenting was presented before, it's hard but i think everyone prefer updated and new things.

Avoid marketing pitching, that's what the market place is for.

Fixing The World Yesterday

Abstracts that sounds and look unrealistic or are improving everything with nothing are less appealing to me. There are always exceptions, but i think it's important to show at least some potential feasibility (now or in the future) to whatever it is you are presenting.

Too Much Inner Topics

Your time is limited, your audience might be diverse from beginners to the creators of OpenStack, you need to focus! Don't try to cover too much, plan your time and presentation and do this before you list all these subjects you going to present in the abstract.

Filling the abstract with as many buzz words as you can is really not helping you getting selected and is really not helping us understand where you going to focus with the limited time you have.

Summary

I hope these tips will help you when you submit your next talk proposal. It's important to note that all of these thoughts are only my own and are in no way reflecting any official statement from OpenStack or anyone else, feel free to disagree and contradict them.

The selection process holds many other criterias that i didn't list here, in the end it's everyone's goal to have the most interesting and diverse agenda for the ENTIRE OpenStack community and the people who attend the summit. If you have any more suggestions or tips i would love to hear them.

Until next time...

March 06, 2017 11:25 PM (http://galsagie.github.io//2017/03/06/submitting-a-talk/)

RDO (http://rdoproject.org/blog/)

Blogs, week of March 6th (http://rdoproject.org/blog/2017/03/blogs-week-of-march-6th/)

There's lots of great blog posts this week from the RDO community.

RDO Ocata Release Behind The Scenes by Haïkel Guémar

> I have been involved in 6 GA releases of RDO (From Juno to Ocata), and I wanted to share a glimpse of the preparation work. Since Juno, our process has tremendously evolved: we refocused RDO on EL7, joined the CentOS Cloud SIG, moved to Software Factory.

Read more at http://tm3.org/ec (http://tm3.org/ec)

Developing Mistral workflows for TripleO by Steve Hardy

> During the newton/ocata development cycles, TripleO made changes to the architecture so we make use of Mistral (the OpenStack workflow API project) to drive workflows required to deploy your OpenStack cloud.

Read more at http://tm3.org/ed (http://tm3.org/ed)

Use a CI/CD workflow to manage TripleO life cycle by Nicolas Hicher

> In this post, I will present how to use a CI/CD workflow to manage TripleO deployment life cycle within an OpenStack tenant.

Read more at http://tm3.org/ee (http://tm3.org/ee)

Red Hat Knows OpenStack by Rich Bowen

> Clips of some of my interviews from the OpenStack PTG last week. Many more to come.

Read more at http://tm3.org/ef (http://tm3.org/ef)

OpenStack Pike PTG: TripleO, TripleO UI - Some highlights by jpichon

> For the second part of the PTG (vertical projects), I mainly stayed in the TripleO room, moving around a couple of times to attend cross-project sessions related to i18n.

Read more at http://tm3.org/eg (http://tm3.org/eg)

OpenStack PTG, trip report by rbowen

> last week, I attended the OpenStack PTG (Project Teams Gathering) in Atlanta.

Read more at http://tm3.org/eh (http://tm3.org/eh)

by Rich Bowen at March 06, 2017 09:51 PM (http://rdoproject.org/blog/2017/03/blogs-week-of-march-6th/)

Blogs, week of Feb 27th (http://rdoproject.org/blog/2017/02/blogs-week-of-feb-27th/)

Here's what RDO enthusiasts have been blogging about in the last couple of weeks. I encourage you to particularly read Julie' excellent writeup (http://tm3.org/eb) of the OpenStack Pike PTG last week in Atlanta. And have a look at my video series (https://www.youtube.com/playlist?list=PLOuHvpVx7kYksG0NFaCaQsSkrUlj3Oq4S) from the PTG for other engineers' perspectives.

OpenStack Pike PTG: OpenStack Client - Tips and background for interested contributors by jpichon

> Last week I went off to Atlanta for the first OpenStack Project Teams Gathering, for a productive week discussing all sort of issues and cross-projects concerns with fellow OpenStack contributors.

Read more at http://tm3.org/eb (http://tm3.org/eb)

SDN with Red Hat OpenStack Platform: OpenDaylight Integration by Nir Yechiel, Senior Technical Product Manager at Red Hat

> OpenDaylight is an open source project under the Linux Foundation with the goal of furthering the adoption and innovation of software-defined networking (SDN) through the creation of a common industry supported platform. Red Hat is a Platinum Founding member of OpenDaylight and part of the community alongside a list of participants that covers the gamut from individual contributors to large network companies, making it a powerful and innovative engine that can cover many use-cases.

Read more at http://tm3.org/e8 (http://tm3.org/e8)

Installing TripleO Quickstart by Carlos Camacho

> This is a brief recipe about how to manually install TripleO Quickstart in a remote 32GB RAM box and not dying trying it.

Read more at http://tm3.org/ea (http://tm3.org/ea)

RDO Ocata released by jpena

> The RDO community is pleased to announce the general availability of the RDO build for OpenStack Ocata for RPM-based distributions, CentOS Linux 7 and Red Hat Enterprise Linux. RDO is suitable for building private, public, and hybrid clouds. Ocata is the 15th release from the OpenStack project, which is the work of more than 2500 contributors from around the world (source).

Read more at http://tm3.org/e9 (http://tm3.org/e9)

OpenStack Project Team Gathering, Atlanta, 2017 by Rich Bowen

> Over the last several years, OpenStack has conducted OpenStack Summit twice a year. One of these occurs in North America, and the other one alternates between Europe and Asia/Pacific.

Read more at http://tm3.org/e0 (http://tm3.org/e0)

Setting up a nested KVM guest for developing & testing PCI device assignment with NUMA by Daniel Berrange

> Over the past few years OpenStack Nova project has gained support for managing VM usage of NUMA, huge pages and PCI device assignment. One of the more challenging aspects of this is availability of hardware to develop and test against. In the ideal world it would be possible to emulate everything we need using KVM, enabling developers / test infrastructure to exercise the code without needing access to bare metal hardware supporting these features.

Read more at http://tm3.org/e1 (http://tm3.org/e1)

ANNOUNCE: libosinfo 1.0.0 release by Daniel Berrange

> NB, this blog post was intended to be published back in November last year, but got forgotten in draft stage. Publishing now in case anyone missed the release...

Read more at http://tm3.org/e2 (http://tm3.org/e2)

Containerizing Databases with Kubernetes and Stateful Sets by Andrew Beekhof

> The canonical example for Stateful Sets with a replicated application in Kubernetes is a database.

Read more at http://tm3.org/e3 (http://tm3.org/e3)

Announcing the ARA 0.11 release by dmsimard

> We're on the road to version 1.0.0 and we're getting closer: introducing the release of version 0.11!

Read more at http://tm3.org/e4 (http://tm3.org/e4)

by Rich Bowen at March 06, 2017 09:51 PM (http://rdoproject.org/blog/2017/02/blogs-week-of-feb-27th/)

OpenStack Superuser (http://superuser.openstack.org)

Why open source is like a team sport (http://superuser.openstack.org/articles/open-source-team-sport-hockey/)

Heather Kirksey likes to call them as she sees them.

As director for Open Platform for NFV (OPNFV (https://www.opnfv.org/)) — a role she alternatively describes as coach, nerd matchmaker and diplomat — she oversees and provides guidance for all aspects of the project, from technology to community and marketing. At the recent Linux Foundation Open Source Leadership Summit (http://events.linuxfoundation.org/events/open-source-leadership-summit), she headed up a session titled "Open Source as a Team Sport" with and OPNFV's Chris Price (https://twitter.com/ChrisPriceAB) and OpenStack's Jonathan Bryce (https://twitter.com/jbryce).

Kirksey (https://twitter.com/HeatherRKirksey) put some thought into the proceedings — setting up a crackling fire video on a giant screen and producing a bottle of whiskey to facilitate a "fireside chat." She kicked off the session with clips from one of her favorite movies, "Miracle." (https://en.wikipedia.org/wiki/Miracle_(film)) Based on the true story of a player-turned-coach who brought the 1980 U.S. Olympic hockey team to victory over Russia, early scenes in the movie show how even successful team building can sometimes be a, well, contact sport.

<iframe allowfullscreen="allowfullscreen" frameborder="0" height="480" src="https://www.youtube.com/embed/TgjQn1z053k" width="854"></iframe>

Superuser sat down with Kirksey – in a busy hallway, minus the whiskey – to ask her more about the parallels between hockey and open source. She tells us why the brutality of hockey is a good metaphor for open source, about leveling the open source playing field for women and how you can get involved with OPNFV.

Of all the team sports, hockey is one of the most violent, right?

Why do you think I like hockey? I like my sports with a side of brutality...In most sports there are tensions that flare up and sometimes it can get raw and there are fisticuffs. At the end of the day, you need to come together because you're trying to accomplish a goal.

In the first clip I showed (https://youtu.be/TgjQn1z053k?t=88), the part where the coach starts talking about how to come together after the two guys started fighting. And he was like, 'hockey is about flow and creativity' that's really where you want to get to when you're creating something new and innovating, like we are in open-source communities. And remembering that aspect of it — the joy of making that and creating a space where you can find joy and have fun as a community is really important.

So there's still a Kumbaya at the end, but you may have to punch somebody in the face to get there.

Yes. It's like après-ski — after you've like fallen down a mountain for a couple days, that hot tub and whiskey feel awfully nice.


(http://superuser.openstack.org/wp-content/uploads/2017/03/IMG_6193.jpg)

Kirksey leads a "fireside chat" with Jonathan Bryce and Chris Price.

Do you see your role as coach, like the Kurt Russell character in the movie?

He [Herb Brooks (https://en.wikipedia.org/wiki/Herb_Brooks)] was one of the legendary coaches of hockey, so it'd be a little self-aggrandizing to make that comparison. But I do feel a large part of my role is to facilitate community, to remind people to have empathy...Sometimes it's reminding people that we are on the same team and we are trying to solve the same problems.

How does your approach change on a mailing list or IRC versus in-person?

The in-person interaction can be more focused on relationship building and helping people get to know each other. Sometimes if the conflict is really getting intense, there's no substitute for face-to-face or on-phone real-time interactions to talk through things.

Back when I chaired standards committee, we would have all these disagreements and contributions. And sometimes people would be arguing different aspects and talking past each other. So I developed a framework, for example, if someone was proposing a solution to a problem. First, do we agree that there is a problem? And do we agree that *this* is the problem? All right, now do we agree on how to solve the problem? Or, are we arguing details of the approach or are we arguing the entire approach itself? You learn to go through it in a structured way...

Earlier, you mentioned your other role as the "United Nations Secretary" between the OPNFV community and other communities. What's your diplomatic strategy?

You have to learn where the understanding point is for the other community, which involves getting to know them. And then do some education. A lot of people don't know how networks are deployed currently, nor what they're trying to transform to. So giving a little bit of 101 — you have a core and an access piece and the places where mobile and fixed line come together and we're trying to address these issues with those aspects. Or come up with analogies. I'm a big fan of analogies.

One of the things that I try to do a lot of at events, especially outside ones, is try to come up with activities at the event, or parties and things to facilitate people interacting. And trying to broker introductions: 'You're working on performance testing in this community. And you're working on performance testing in this community. Hey, why not talk — I'll buy you a beer.' I pick up a lot of beer tabs for developers. I'm kind of a professional beer-tab picker-upper.

Do you also see yourself as a matchmaker?

Yeah. A little bit sometimes, a matchmaker of nerds. But if you can find that common ground — say both people are doing the same thing but in different communities, then they immediately have a starting point. And generally people are excited to talk about what they're working on.

Congratulations on the nomination for the Women in Open Source Award (http://www.redhat.com/en/about/women-in-open-source). How long do you think it'll be before it's *not* about women in open source— just people who are outstanding, period?

Well, I have been in the tech industry for 19 years now and the percentage of women hasn't really gone up. Although I think we're having better conversations about it now. We're having more allies-oriented conversations, which I think are good.
I think a lot of folks — anywhere on the gender spectrum — are realizing this shit has got to change...For example, folks are now measuring diversity — you're not going to change what you don't measure. The fact that a lot of tech companies are now measuring their diversity and setting diversity goals matters. And the fact that the Linux Foundation worked with National Center for Women & Information Technology (NCWIT (https://www.ncwit.org/)) to create training (https://training.linuxfoundation.org/linux-courses/open-source-compliance-courses/inclusive-speaker-orientation?utm_source=pr&utm_campaign=training&utm_medium=press-release) for speakers and conferences matters. We're seeing more structure around it now, for sure.

What else might make a difference?

Root out the structural inequalities of our modern world? [laughs.] That's obviously is a bigger thing... The issues in tech reflect broader societal issues... Unconscious bias, for example, being one of the big things... Part of it is human, we get used to the way things are and change is difficult. We get used to people who are like us and share common background experiences or common cultural references, even. And so unconscious bias is a really hard thing to root out. Accepting that it exists and being aware of it and trying to catch yourself and being intentionally thoughtful and self-reflective in your hiring decisions, in your community processes and on your conference panels helps.

Get Involved with OPNFV
Whether you're an employee of a member company or just passionate about network transformation, you'll find the basics of how to get started (creating an account, mailing lists, Wiki, projects) here (https://www.opnfv.org/community/get-involved).

To meet OPNFVers in person (and maybe get some free beer?) you'll find them at the upcoming Open Networking Summit, (http://events.linuxfoundation.org/events/open-networking-summit)OpenStack Summit (https://www.openstack.org/summit/boston-2017/) and the OPNFV Summit (http://events.linuxfoundation.org/events/opnfv-summit).

For Open Stack individual contributors, what's a good starting point at OPNFV?

If you've got any CI/CD knowledge, that would be great. Also our testing projects are a good place to start, because really what we focus on as a community is NFV stat testing which spans a lot of different pieces from a lot of different communities. We never have enough tests or people writing test cases. That's a need and a lower barrier to entry, scripting a test case versus diving into changing a networking part of Neutron, for example. And then you get a lot more exposure to the software itself, how it's configured, how it's run, how it deploys, what it's supposed to do and what success is and what failure is.

You can also reach out to the individual Project Team Leads (PTLs)...They're great and they love to hear from people who want to contribute to the project.

Cover photo by: Martyn Hayes
(http://superuser.openstack.org/feed/"https://www.flickr.com/photos/photohayes/6151328560/)

The post Why open source is like a team sport (http://superuser.openstack.org/articles/open-source-team-sport-hockey/) appeared first on OpenStack Superuser (http://superuser.openstack.org).

by Nicole Martinelli at March 06, 2017 12:20 PM (http://superuser.openstack.org/articles/open-source-team-sport-hockey/)

Hugh Blemings (http://hugh.blemings.id.au)

Lwood-20170305 (http://hugh.blemings.id.au/2017/03/06/lwood-20170305/)

Introduction

Welcome to Last week on OpenStack Dev ("Lwood") for the week just past. For more background on Lwood, please refer here (http://hugh.blemings.id.au/openstack/lwood/).

Basic Stats for the week 27 February to 5 March for openstack-dev:

~424 Messages (down nearly 26% relative to the long term average)

~154 Unique threads (down a bit over 14% relative to the long term average)

Traffic picked up a bit relative to last week though once again a fairly brief Lwood – the main thing of note to many will I suspect be the summary of summaries from the PTG

Notable Discussions – openstack-dev

OpenStack Summit returns to Vancouver in 2018

Allison Price announced (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113141.html) that the Summit is returning to the fair city of Vancouver in May 2018.

OpenStack PTG Atlanta summary of summaries

With Atlanta PTG concluded the summaries are starting to come in – as I've done previously (http://hugh.blemings.id.au/2016/11/21/openstack-summit-barcelona-2016-summary-of-summaries/) I'll link them over the next few Lwoods then put together an aggregated list

- [All] [API] API-WG PTG recap (http://lists.openstack.org/pipermail/openstack-dev/2017-February/112995.html) – Chris Dent
- [Architecture] Pike PTG Architecture Working Group Recap (http://lists.openstack.org/pipermail/openstack-dev/2017-February/113016.html) – Clint Byrum
- [Cinder] PTG Recap (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113315.html) – Sean McGinnis
- [Docs] Docs PTG summary (http://lists.openstack.org/pipermail/openstack-dev/2017-February/112967.html) – Alexandra Settle
- [Horizon] PTG Summary (http://lists.openstack.org/pipermail/openstack-dev/2017-February/112950.html) – Rob Cresswell
- [Keystone] Pike PTG Summary (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113068.html) – Lance Bragstad
- [Kuryr] 1st (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113093.html), 2nd (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113176.html) and Final day (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113290.html) recording for the Virtual Team Gathering – Antoni Segura Puimedon
- [Networking-sfc] PTG notes (http://lists.openstack.org/pipermail/openstack-dev/2017-February/113045.html) – Bernard Cafarelli
- [Neutron] PTG Summary (http://lists.openstack.org/pipermail/openstack-dev/2017-February/113032.html) – Kevin Benton
- [Nova][Keystone] Pike PTG recap – quotas (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113068.html) – Lance Bragstad
- [Nova][Keystone] Pike PTG recap – quotas (http://lists.openstack.org/pipermail/openstack-dev/2017-February/112991.html) – Matt Riedemann
- [Nova] Pike PTG recap – notifications (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113098.html) – Balazs Gibizer
- [Nova] Pike PTG recap – API (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113171.html) – Matt Riedemann
- [Nova] Pike PTG recap – Cells (http://lists.openstack.org/pipermail/openstack-dev/2017-February/112996.html) – Matt Riedemann
- [Nova][Cinder] Pike PTG recap – nova/cinder (http://lists.openstack.org/pipermail/openstack-dev/2017-February/113063.html)– Matt Riedemann
- [Manila] PTG summary (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113309.html) – Ben Swartzlander
- [Placement][Nova] PTG summary (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113306.html) – Jay Pipes
- [QA] Pike PTG Summary (http://lists.openstack.org/pipermail/openstack-dev/2017-February/112941.html) – Andrea Frittoli
- [Relmgt] Pike PTG recap (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113121.html) – Thierry Carrez
- [Sahara] Pike PTG summary (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113125.html) – Telles Nobrega

- [Swift] Thoughts on the PTG (http://lists.openstack.org/pipermail/openstack-dev/2017-February/113062.html) – John Dickinson
- [TC][SWG] Summary – Atlanta PTG Stewardship Working Group Session (http://lists.openstack.org/pipermail/openstack-dev/2017-February/112959.html) – Colette Alexander
- [Tripleo] On the road to Pike (http://lists.openstack.org/pipermail/openstack-dev/2017-February/112995.html) – Emilien Macchi
- [Zuul] Zuul v3 – What's Coming: What to expect with the Zuul v3 Rollout (http://lists.openstack.org/pipermail/openstack-dev/2017-February/113065.html) – Monty Taylor

OpenStack Community Leadership Training open to all

The opportunity to take the well regarded leadership training program that had previously been made available to the TC, Board and Foundation staff is now being extended to all Community members writes (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113267.html) Colette Alexander.

End of Week Wrap-ups, Summaries and Updates

Three this week; Horizon (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113345.html) (Rob Cresswell), Ironic (http://lists.openstack.org/pipermail/openstack-dev/2017-February/112990.html) (Ruby Loo) and Zuul (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113148.html) (Robyn Bergernon)

People and Projects

Core nominations & changes

- [Mistral] Proposing Michal Gershenzon (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113134.html) to the core team – Renat Akhmerov
- [Neutron] Added Gary Kotton and Russell Boden (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113252.html) to release team – Kevin Benton
- [Tricircle] Nominating Victor Morales (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113072.html) as Tricircle Core – Joe Huang

Miscellanea

Further reading

Don't forget these excellent sources of OpenStack news – most recent ones linked in each case

- What's Up, Doc? (http://lists.openstack.org/pipermail/openstack-dev/2017-February/112885.html) by Alexandra Settle
- API Working Group newsletter (http://lists.openstack.org/pipermail/openstack-dev/2017-March/113248.html) – Chris Dent and the API WG
- OpenStack Developer Mailing List Digest (http://lists.openstack.org/pipermail/openstack-dev/2017-January/111139.html) by Mike Perez & Kendall Nelson
- OpenStack news over on opensource.com (https://opensource.com/article/17/3/openstack-news-march-6) by Jason Baker
- OpenStack Foundation Events Page (https://www.openstack.org/community/events/) for a frequently updated list of events

Credits

No tunes this week, was working remotely and wasn't an appropriate setting for tunes (aka I forgot headphones :)

by hugh at March 06, 2017 11:34 AM (http://hugh.blemings.id.au/2017/03/06/lwood-20170305/)

RDO (http://rdoproject.org/blog/)

RDO Ocata Release Behind The Scenes (http://rdoproject.org/blog/2017/02/rdo-ocata-release-behind-the-scenes/)

I have been involved in 6 GA releases of RDO (From Juno to Ocata), and I wanted to share a glimpse of the preparation work. Since Juno, our process has tremendously evolved: we refocused RDO on EL7, joined the CentOS Cloud SIG, moved to Software Factory (https://www.redhat.com/archives/rdo-list/2015-October/msg00123.html).

Our release process does not start when upstream announces GA or even a milestone, no it starts from the very beginning of upstream cycle.

Trunk chasing

We have been using DLRN (https://www.rdoproject.org/what/dlrn/) to track upstream changes and build continuously OpenStack as a RPM distribution. Then our CI (https://ci.centos.org/view/rdo/view/promotion-pipeline/) hosted on CentOS community CI (https://wiki.centos.org/QaWiki/CI) runs multiple jobs on DLRN snapshots. We use the WeIRDO (https://github.com/rdo-infra/weirdo) framework to run the same jobs as upstream CI on our packages. This allows us to detect early integration issues and get either our packaging or upstream projects fixed. This also includes installers such as OPM, TripleO or PackStack.

We also create Ocata tags in CentOS Community Build System (CBS) in order to build dependencies that are incompatible with currently supported releases.

Branching

We start branching RDO stable release around milestone 3, and have stable builds getting bootstrapped. This includes:

- registering packages in CBS, I scripted this part for Ocata using rdoinfo database.
- syncing requirements in packages.
- branching distgit repositories.
- building upstream releases in CBS, this part used to be semi-automated using rdopkg tool, Alfredo is consolidating that into a cron job creating reviews.

- tag builds in <release>-testing repositories, some automation is in preparation.</release>

Trunk chasing continues, but we pay attention in keeping promotions happening more frequently to avoid a gap between tested upstream commits and releases.

GA publication

Since OpenStack does releases slightly ahead of time, we have most of GA releases built in CBS, but some of them comes late. We also trim final GA repositories, use repoclosure utility to check if there's no missing dependencies. Before mass-tagging builds in <release>-release we launch stable promotion CI jobs and if they're green, we publish them.</release>

At this stage, CentOS Core team, creates final GA repositories and sign packages.

For Newton, it took 10 hours between upstream GA announcement and repositories publication, 4 hours up to stable tagging. As for Ocata, all stable builds + CI jobs were finished within 2 hours.

Fun fact, Alan and I were doing the last bits of the Ocata release in the Atlanta PTG hallway and even get to see Doug Hellmann to send the GA announcement live (which started the chronometer for us). So we sprinted to have RDO Ocata GA ready as soon as possible (CI included!). We still have room for improvement but we were the first binary OpenStack distro available!

Thoughts

As of Ocata, there are still areas of improvement:

- documenting releases process: many steps are still manual or require specific knowledge. During Newton/Ocata releases, we enabled Alfredo to do large chunks of the release preparation work. With post-mortems, this helped us clarifying the process, and prepare to allow more people helping in the release process.
- dependencies CI: dependencies are a critical factor to release RDO in time. We need to test dependencies against RDO releases, RDO against CentOS updates and ensure that nothing is broken. That's one of our goals for Pike.
- tag management: tags are used in CBS to determine where builds are to be published. Unlike Fedora, CBS has no automated pipeline to manage updates, so we have to manually tag builds. I'm currently working on having a gerrit-based process to manage tags. The tricky part is how to avoid inconsistencies in repositories (e.g avoid breaking dependencies, accidental untagging etc.)
- dependencies updates: we want dependencies to remain compatible with Fedora packages, as Fedora is the foundation of next RHEL/CentOS, some of them are maintained in Fedora, others in RDO common (https://github.com/rdo-common), some with basic patches to fix EL7 build issues (not acceptable in Fedora), the rest being forks that we effectively maintain (e.g MariaDB). As a first step, we want to have the last set of packages to be maintained in our gerrit instances to allow maintainers doing builds without any releng support.
- more contributions! Our effort into automating the release pipeline also serves the goal of empowering more contributors into the release work, so if you're interested, just come and tell us. ;-)

I hope this gave you an overview of how RDO is released and what are our next steps for Pike release.

by Haïkel Guémar at March 06, 2017 09:31 AM (http://rdoproject.org/blog/2017/02/rdo-ocata-release-behind-the-scenes/)

Sean Roberts (https://sarob.com)

Using Transparency for Building Strong Organizations (https://sarob.com/2017/03/transparency-building-strong-organization/)

I have broken down below a simple six part outline of what a technology platform organization is and how to run it. The platform organization is the bottom of the technology stack. The top of the stack will differ from the platform, in that, it will have external customers and more focus on product design and the user interface. However, ...

Continue reading "Using Transparency for Building Strong Organizations" (https://sarob.com/2017/03/transparency-building-strong-organization/)

The post Using Transparency for Building Strong Organizations (https://sarob.com/2017/03/transparency-building-strong-organization/) appeared first on sarob (https://sarob.com).

by sarob at March 06, 2017 06:37 AM (https://sarob.com/2017/03/transparency-building-strong-organization/)

Opensource.com (https://opensource.com/taxonomy/term/5126/feed/feed)

Improved container support, PTG recap, and more OpenStack news (https://opensource.com/article/17/3/openstack-news-march-6)

Explore what's happening this week in OpenStack, the open source cloud computing project.

by Jason Baker at March 06, 2017 06:00 AM (https://opensource.com/article/17/3/openstack-news-march-6)

March 04, 2017

Mirantis (https://www.mirantis.com)

What's new in OpenStack Ocata webinar — Q&A (https://www.mirantis.com/blog/whats-new-in-openstack-ocata-webinar-qa/)

The post What's new in OpenStack Ocata webinar — Q&A (https://www.mirantis.com/blog/whats-new-in-openstack-ocata-webinar-qa/) appeared first on Mirantis | Pure Play Open Cloud (https://www.mirantis.com).

On February 22, my colleagues Rajat Jain, Stacy Verroneau, and Michael Tillman and I held a webinar to discuss the new features in OpenStack's latest release, Ocata (https://content.mirantis.com/Webinar-Whats-New-in-Ocata_Landing-Page.html). Unfortunately, we ran out of time for questions and answers, so here they are. **Q: What are the benefits of using the cells capability? Rajat:** The cells concept was introduced in the Juno release, and as some of you may recall, it was to allow a large number of nova/compute instances to share openstack services. Therefore, Cells functionality enables you to scale an OpenStack Compute cloud in a more

distributed fashion without having to use complicated technologies like database and message queue clustering. It supports very large deployments. When this functionality is enabled, the hosts in an OpenStack Compute cloud are partitioned into groups called cells. Cells are configured as a tree. The top-level cell should have a host that runs a nova-api service, but no nova-compute services. Each child cell should run all of the typical nova-* services in a regular Compute cloud except for nova-api. You can think of cells as a normal Compute deployment in that each cell has its own database server and message queue broker. This was achieved by the nova cells and nova api services to provide the capabilities. One of the key changes in Ocata is the upgrade to cells v2, which now only relies on the nova api service for all the synchronization across the cells. **Q: What is the placement service and how can I leverage it? Rajat:** The placement service, which was introduced in the Newton release, is now a key part of OpenStack and also mandatory in determining the optimum placement of VMs. Basically, you set up pools of resources, provide an inventory of the compute nodes, and then set up allocations for resource providers. Then you can set up policies and models for optimum placements of VMs. **Q: What is the OS profiler, and why is it useful? Rajat:** OpenStack consists of multiple projects. Each project, in turn, is composed of multiple services. To process a request — for example, to boot a virtual machine — OpenStack uses multiple services from different projects. If something in this process runs slowly, it's extremely complicated to understand what exactly goes wrong and to locate the bottleneck. To resolve this issue, a tiny but powerful library, osprofiler, was introduced. The osprofiler library will be used by all OpenStack projects and their python clients. It provides functionality to be able to generate 1 trace per request, flowing through all involved services. This trace can then be extracted and used to build a tree of calls which can be quite handy for a variety of reasons (for example, in isolating cross-project performance issues). **Q: If I have keystone connected to a backend active directory, will i benefit from the auto-provisioning of the federated identity? Rajat:** Yes. The federated identity mapping engine now supports the ability to automatically provision projects for federated users. A role assignment will automatically be created for the user on the specified project. Prior to this, a federated user had to attempt to authenticate before an administrator could assign roles directly to their shadowed identity, resulting in a strange user experience. This is therefore a big usability enhancement for deployers leveraging the federated identity plugins. **Q: Is FWaaS really used out there? Stacy:** Yes it is, but its viability in production is debatable and going with a 3rd party with a Neutron plugin is still, IMHO, the way to go. **Q: When is Octavia GA planned to be released? Stacy:** Octavia is forecast to be GA in the Pike release. **Q: Are DragonFlow and Tricircle ready for Production? Stacy:** Those are young big tent projects but pretty sure we will see a big evolution for Pike. **Q: What's the codename for placement service please? Stacy:** It's just called the Placement API. There's no fancy name. **Q: Does Ocata continue support for Fernet tokens? Rajat:** Yes. **Q: With federated provider, can i integrate openstack env with my on-prem AD and allow domain users to use Openstack? Rajat:** This was always supported, and is not new to ocata. More details at https://docs.openstack.org/admin-guide/identity-integrate-with-ldap.html (https://docs.openstack.org/admin-guide/identity-integrate-with-ldap.html) What's new in this area is that the federated identity mapping engine now supports the ability to automatically provision projects for federated users. A role assignment will automatically be created for the user on the specified project. Prior to this, a federated user had to attempt to authenticate before an administrator could assign roles directly to their shadowed identity, resulting in a strange user experience. **Q: if i'm using my existing domain users from AD to openstack, how would i control their rights/role to perform specific tasks in the openstack project? Rajat:** You would first set up authentication via LDAP, then provide connection settings for AD and also set the identity driver to ldap in the keystone.conf. Next you will have to do an assignment of roles and projects to the AD users. Since Mitaka (http://lists.openstack.org/pipermail/openstack/2015-January/011337.html)), the only option that you can use is the SQL driver for the assignment in the keystone.conf, but you will have to do the mapping. Most users prefer this approach anyway, as they want to keep the AD as read only from the OpenStack connection. You can find more details on how to configure keystone with LDAP here (https://docs.openstack.org/admin-guide/identity-integrate-with-ldap.html). **Q: What, if anything, was pushed out of the "big tent" and/or did not get robustly worked? Nick:** You can get a complete view of work done on every project at Stackalytics (http://stackalytics.com/). **Q: So when is Tricircle being released for use in production? Stacy:** Not soon enough. Being a new Big Tent project, it needs some time to develop traction. **Q: Do we support creation of SRIOV ports from horizon during instance creation. If not, are there any plans there? Nick:** According to the Horizon team, you can pre-create the port and assign it to an instance. **Q: Way to go warp speed Michael! Good job Rajat and Stacy. Don't worry about getting behind, I blame Nick anyway. Then again I always I always blame Nick. Nick:** Thanks Ben, I appreciate you, too.
The post What's new in OpenStack Ocata webinar — Q&A (https://www.mirantis.com/blog/whats-new-in-openstack-ocata-webinar-qa/) appeared first on Mirantis | Pure Play Open Cloud (https://www.mirantis.com).

by Nick Chase at March 04, 2017 12:50 AM (https://www.mirantis.com/blog/whats-new-in-openstack-ocata-webinar-qa/)

March 03, 2017

Steve Hardy (http://hardysteven.blogspot.com/search/label/openstack)

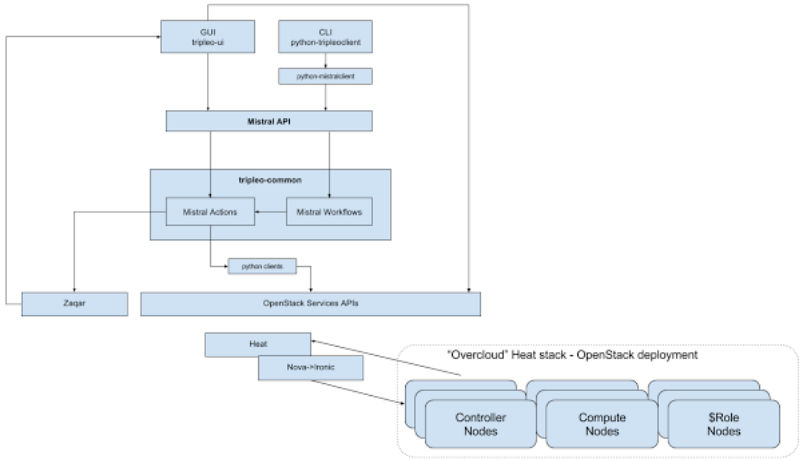
Developing Mistral workflows for TripleO (http://hardysteven.blogspot.com/2017/03/developing-mistral-workflows-for-tripleo.html)

During the newton/ocata development cycles, TripleO (https://docs.openstack.org/developer/tripleo-docs/) made changes to the architecture so we make use of Mistral (https://docs.openstack.org/developer/mistral/) (the OpenStack workflow API project) to drive workflows required to deploy your OpenStack cloud.

Prior to this change we had workflow defined inside python-tripleoclient (https://github.com/openstack/python-tripleoclient), and most API calls were made directly to Heat (https://docs.openstack.org/developer/heat/). This worked OK but there was too much "business logic" inside the client, which doesn't work well if non-python clients (such as tripleo-ui) want to interact with TripleO (https://docs.openstack.org/developer/tripleo-docs/).

To solve this problem, number of mistral workflows and custom actions have been implemented, which are available via the Mistral API on the undercloud. This can be considered the primary "TripleO API" for driving all deployment tasks now.

Here's a diagram showing how it fits together:



(https://1.bp.blogspot.com/-yJV_sCCMFe4/WLkzGkVPHgI/AAAAAAAAH1g/DWRdbXLd9yghwCF0ugq2k9KSNDkSqMNBACLcB/s1600/Mistral%2BAPI.png)
Overview of Mistral integration in TripleO

Mistral workflows and actions

There are two primary interfaces to mistral, ***workflows*** (https://docs.openstack.org/developer/mistral/dsl/dsl_v2.html#workflows) which are a yaml definition of a process or series of tasks, and ***actions*** (https://docs.openstack.org/developer/mistral/dsl/dsl_v2.html#actions) which are a concrete definition of how to do a specific task (such as call some OpenStack API).

Workflows and actions can defined directly via the mistral API (https://docs.openstack.org/developer/mistral/developer/webapi/index.html), or a wrapper called a ***workbook*** (https://docs.openstack.org/developer/mistral/dsl/dsl_v2.html#workbooks). Mistral actions are also defined via a python plugin interface (https://docs.openstack.org/developer/mistral/developer/creating_custom_action.html), which TripleO uses to run some tasks such as running jinja2 (http://jinja.pocoo.org/) on tripleo-heat-templates (https://github.com/openstack/tripleo-heat-templates) prior to calling Heat (https://docs.openstack.org/developer/heat/) to orchestrate the deployment.

Mistral workflows, in detail

Here I'm going to show how to view and interact with the mistral workflows used by TripleO directly, which is useful to understand what TripleO is doing "under the hood" during a deployment, and also for debugging/development.

First we view the mistral workbooks loaded into Mistral - these contain the TripleO specific workflows and are defined in tripleo-common (https://github.com/openstack/tripleo-common/tree/master/workbooks)

```
[stack@undercloud ~]$ . stackrc
[stack@undercloud ~]$ mistral workbook-list
+----------------------------+---------+---------------------+-------------+
| Name                       | Tags    | Created at          | Updated at  |
+----------------------------+---------+---------------------+-------------+
| tripleo.deployment.v1      | <none>  | 2017-02-27 17:59:04 | None        |
| tripleo.package_update.v1  | <none>  | 2017-02-27 17:59:06 | None        |
| tripleo.plan_management.v1 | <none>  | 2017-02-27 17:59:09 | None        |
| tripleo.scale.v1           | <none>  | 2017-02-27 17:59:11 | None        |
| tripleo.stack.v1           | <none>  | 2017-02-27 17:59:13 | None        |
| tripleo.validations.v1     | <none>  | 2017-02-27 17:59:15 | None        |
| tripleo.baremetal.v1       | <none>  | 2017-02-28 19:26:33 | None        |
+----------------------------+---------+---------------------+-------------+
```

The name of the workbook constitutes a namespace for the workflows it contains, so we can view the related workflows using grep (I also grep for tag_node to reduce the number of matches).

```
[stack@undercloud ~]$ mistral workflow-list | grep "tripleo.baremetal.v1" | grep tag_node
| 75d2566c-13d9-4aa3-b18d-8e8fc0dd2119 | tripleo.baremetal.v1.tag_nodes                          | 660c5ec71ce043c1a43d3529e7065a9d
| 7a4220cc-f323-44a4-bb0b-5824377af249 | tripleo.baremetal.v1.tag_node                           | 660c5ec71ce043c1a43d3529e7065a9d
```

When you know the name of a workflow, you can inspect the required inputs, and run it directly via a mistral execution, in this case we're running the tripleo.baremetal.v1.tag_node workflow, which modifies the profile assigned in the ironic node capabilities (see tripleo-docs for more information about manual tagging of nodes (https://docs.openstack.org/developer/tripleo-docs/advanced_deployment/profile_matching.html#manual-profile-tagging))

```
[stack@undercloud ~]$ mistral workflow-get tripleo.baremetal.v1.tag_node
+------------+----------------------------------------+
| Field      | Value                                  |
+------------+----------------------------------------+
| ID         | 7a4220cc-f323-44a4-bb0b-5824377af249   |
| Name       | tripleo.baremetal.v1.tag_node          |
| Project ID | 660c5ec71ce043c1a43d3529e7065a9d       |
| Tags       | <none>                                 |
| Input      | node_uuid, role=None, queue_name=tripleo |
| Created at | 2017-02-28 19:26:33                    |
| Updated at | None                                   |
+------------+----------------------------------------+
[stack@undercloud ~]$ ironic node-list
+--------------------------------------+-----------+---------------+-------------+--------------------+-------------+
| UUID                                 | Name      | Instance UUID | Power State | Provisioning State | Maintenance |
+--------------------------------------+-----------+---------------+-------------+--------------------+-------------+
| 30182cb9-eba9-4335-b6b4-d74fe2581102 | control-0 | None          | power off   | available          | False       |
| 19fd7ea7-b4a0-4ae9-a06a-2f3d44f739e9 | compute-0 | None          | power off   | available          | False       |
+--------------------------------------+-----------+---------------+-------------+--------------------+-------------+
[stack@undercloud ~]$ mistral execution-create tripleo.baremetal.v1.tag_node '{"node_uuid": "30182cb9-eba9-4335-b6b4-d74fe2581102", "r
+-------------------+--------------------------------------+
| Field             | Value                                |
+-------------------+--------------------------------------+
| ID                | 6a141065-ad6e-4477-b1a8-c178e6fcadcb |
| Workflow ID       | 7a4220cc-f323-44a4-bb0b-5824377af249 |
| Workflow name     | tripleo.baremetal.v1.tag_node        |
| Description       |                                      |
| Task Execution ID | <none>                               |
| State             | RUNNING                              |
| State info        | None                                 |
| Created at        | 2017-03-03 09:53:10                  |
| Updated at        | 2017-03-03 09:53:10                  |
+-------------------+--------------------------------------+
```

At this point the mistral workflow is running, and it'll either succeed or fail, and also create some output (which in the TripleO model is sometimes returned to the Ux via a Zaqar (https://docs.openstack.org/developer/zaqar/) queue). We can view the status, and the outputs (truncated for brevity):

```
[stack@undercloud ~]$ mistral execution-list | grep  6a141065-ad6e-4477-b1a8-c178e6fcadcb
| 6a141065-ad6e-4477-b1a8-c178e6fcadcb | 7a4220cc-f323-44a4-bb0b-5824377af249 | tripleo.baremetal.v1.tag_node
[stack@undercloud ~]$ mistral execution-get-output 6a141065-ad6e-4477-b1a8-c178e6fcadcb
{
    "status": "SUCCESS",
    "message": {
...
```

So that's it - we ran a mistral workflow, it suceeded and we looked at the output, now we can see the result looking at the node in Ironic, it worked! :)

```
[stack@undercloud ~]$ ironic node-show 30182cb9-eba9-4335-b6b4-d74fe2581102 | grep profile
|                       | u'cpus': u'2', u'capabilities': u'profile:test,cpu_hugepages:true,boot_o |
```

Mistral workflows, create your own!

Here I'll show how to develop your own custom workflows (which isn't something we expect operators to necessarily do, but is now part of many developers workflow during feature development for TripleO).

First, we create a simple yaml definition of the workflow, as defined in the v2 Mistral DSL (https://docs.openstack.org/developer/mistral/dsl/dsl_v2.html#workflows) - this example (https://gist.github.com/hardys/b1999c500185ccc832f0ac8731174b2a) lists all available ironic nodes, then finds those which match the "test" profile we assigned in the example above:

<script src="https://gist.github.com/hardys/b1999c500185ccc832f0ac8731174b2a.js"></script>
This example uses the mistral built-in "ironic" action, which is basically a pass-through action exposing the python-ironicclient interfaces. Similar actions exist for the majority of OpenStack python clients, so this is a pretty flexible interface.

Now we can now upload the workflow (not wrapped in a workbook this time, so we use workflow-create), run it via execution create, then look at the outputs - we can see that the *matching_nodes* output matches the ID of the node we tagged in the example above - success! :)

```
[stack@undercloud tripleo-common]$ mistral workflow-create shtest.yaml
+--------------------------------------+-------------------------+----------------------------------+--------+-------------+---------
| ID                                   | Name                    | Project ID                       | Tags   | Input       | Created
+--------------------------------------+-------------------------+----------------------------------+--------+-------------+---------
| 2b8f2bea-f3dd-42f0-ad16-79987c75df4d | test_nodes_with_profile | 660c5ec71ce043c1a43d3529e7065a9d | <none> | profile=test | 2017-03-
+--------------------------------------+-------------------------+----------------------------------+--------+-------------+---------
[stack@undercloud tripleo-common]$ mistral execution-create test_nodes_with_profile
+-------------------+--------------------------------------+
| Field             | Value                                |
+-------------------+--------------------------------------+
| ID                | 2392ed1c-96b4-4787-9d11-0f3069e9a7e5 |
| Workflow ID       | 2b8f2bea-f3dd-42f0-ad16-79987c75df4d |
| Workflow name     | test_nodes_with_profile              |
| Description       |                                      |
| Task Execution ID | <none>                               |
| State             | RUNNING                              |
| State info        | None                                 |
| Created at        | 2017-03-03 10:19:30                  |
| Updated at        | 2017-03-03 10:19:30                  |
+-------------------+--------------------------------------+
[stack@undercloud tripleo-common]$ mistral execution-list | grep  2392ed1c-96b4-4787-9d11-0f3069e9a7e5
| 2392ed1c-96b4-4787-9d11-0f3069e9a7e5 | 2b8f2bea-f3dd-42f0-ad16-79987c75df4d | test_nodes_with_profile
[stack@undercloud tripleo-common]$ mistral execution-get-output 2392ed1c-96b4-4787-9d11-0f3069e9a7e5
{
    "matching_nodes": [
        "30182cb9-eba9-4335-b6b4-d74fe2581102"
    ],
    "available_nodes": [
        "30182cb9-eba9-4335-b6b4-d74fe2581102",
        "19fd7ea7-b4a0-4ae9-a06a-2f3d44f739e9"
    ]
}
```

Using this basic example, you can see how to develop workflows which can then easily be copied into the tripleo-common workbooks, and integrated into the TripleO deployment workflow.

In a future post, I'll dig into the use of custom actions, and how to develop/debug those.

by Steve Hardy (noreply@blogger.com) at March 03, 2017 03:59 PM (http://hardysteven.blogspot.com/2017/03/developing-mistral-workflows-for-tripleo.html)

OpenStack Superuser (http://superuser.openstack.org)

Deploying OpenStack on AWS (http://superuser.openstack.org/articles/deploying-openstack-aws-tutorial/)

Cloud on a cloud!

Yes, you read that correctly! This post will show you how to install OpenStack on top of AWS EC2.

Installing OpenStack in a nested hypervisor environment is not a big deal when we use the QEMU emulator for launching virtual machines inside the virtual machine. However, unlike usual nested hypervisor setup, installing OpenStack on AWS EC2 instances, we will have to bypass a few restrictions for the OpenStack setup to work properly. This blog post outlines those limitations and their solutions to run OpenStack on top of AWS EC2 Virtual Machine.

Limitations

An AWS environment will allow the packets to flow in their network only when the MAC address is known/registered in the AWS network environment. Also, the MAC address and the IP address are tightly mapped, as the AWS environment will not allow the packet to flow if the MAC address registered for the given IP address is different.

You may be wondering if the above restrictions will impact the OpenStack setup on AWS EC2.
The answer is: yes, it will!

While configuring Neutron networking, we will be creating a virtual bridge (br-ex) for the Provider Network where all the VMs traffic will reach the Internet via external bridge followed by the actual physical NIC (eth1).

In this case, we will configure the external interface (NIC) with a special type of configuration.

The provider interface uses a special configuration without an IP address assigned to it. We will configure the second interface as the provider interface.

Replace INTERFACE_NAME with the actual interface name. For example, *eth1* or *ens224*.

- Edit the /etc/network/interfaces file to contain the following:

```
# The provider network interface
auto INTERFACE_NAME
iface INTERFACE_NAME inet manual
up ip link set dev $IFACE up
down ip link set dev $IFACE down
```

Ref: – http://docs.openstack.org/mitaka/install-guide-ubuntu/environment-networking-controller.html (http://docs.openstack.org/mitaka/install-guide-ubuntu/environment-networking-controller.html)

With this interface configuration, the restriction in AWS will hit the OpenStack networking. In mainstream OpenStack setup, the provider interface mentioned above can be configured with special NIC configuration that will have no IP for that interface.

Moreover, the VM packets reaching the Internet via this specially configured NIC will have the IP of OpenStack tenant router's gateway IP address as the source IP address in each packet.

Like I mentioned earlier, AWS will only allow the packet to flow when the MAC address is known/registered in their environment. Also, the IP address must match the MAC address.

In our case, the packet from the aforementioned OpenStack tenant router will have the IP address of router's Gateway in every single packet, but the packet source MAC address will be the same MAC address of router's interface.

Note: – You could see these details using the "ip netns show" followed by "ip netns exec qr-<router_ID> ifconfig" command in the OpenStack controller's terminal.

Since the MAC address in unknown/not registered in the AWS environment, the packets will be dropped when it reaches the AWS switch. To allow the VM packets to reach the Internet via AWS switch, we need to do some tricks on our OpenStack setup.

Making use of what we have

Possible hacks include:

1. Registering the router's MAC address and its IP address with the AWS environment. However, this hack is actually not feasible. AWS doesn't have the ability to register any random MAC address and IP address inside the VPC. Moreover, allowing this type of functionality will be a severe security threat to the environment. So, the trick we are going to use is:
2. Making use of what we have. Since we have used a special type of interface configuration for the provider NIC, you could note that the IP address assigned to the provider NIC (eth1) is left unused. We could use this available/unused IP address for the OpenStack router's gateway. The following command will do the trick:

```
neutron router-gateway-set router provider --fixed-ip ip_address=<Registered_IP_address*>
```

IP address and MAC address mismatch

After configuring the router gateway with the AWS registered IP address, each packet from router's gateway will have the AWS registered IP address as the source IP address but with an Open vSwitch-generated MAC address.

Like I mentioned in the AWS restriction session above, the IP address must match the MAC address registered or else all of the packets with mismatched MAC and IP address will be dropped by the AWS switch.

To make the registered MAC address match with the IP address, we need to change the MAC address of router's interface. The following steps will handle it for you:

1. Install the *macchanger* tool.
2. Note down the actual / Original MAC address of provider NIC (eth1)
3. Change the MAC address of Provider NIC (eth1)
4. Change the MAC address of Router's Gateway Interface to the original MAC address of eth1.
5. Now, try to ping 8.8.8.8 from router namespace

If you get a successful ping response, then we are done with Cloud on Cloud setup.

Key points to remember

- Change MAC address:

In my case, I had worked in an Ubuntu 14.04 LTS server which has no issue in changing the MAC address using the macchanger tool. However, when I tried it in an Ubuntu 16.04 LTS one, I got an error saying "No permission to modify the MAC address ." I suspect the error was due to the cloud-init tool not allowing the MAC address configuration.

So before installing OpenStack setup, try changing MAC address of the NIC.

Floating IP disabled

Associating a floating IP to any OpenStack's VM will send the packet through the router's gateway with the source IP address as floating IP's address. This will make the packets hit the AWS switch with non-registered IP and MAC addresses, resulting in the switch dropping the packets.

So, we could not use the floating IP functionality in this setup. However, we could still access the VM publicly by using the following NAT process.

NAT to access OpenStack VM

Like I mentioned above, we could access the OpenStack VM publicly using the registered IP address that we have assigned for router's gateway.

Use the below NAT command to access the Openstack VM using the AWS EC2 instance's elastic IP:

```
$ip netns exec qrouter-f85bxxxx-61b2-xxxx-xxxx-xxxxba0xxxx iptables -t nat -A PREROUTING -p tcp -d 172.16.20.101 --dport 522 -j DNAT --
to-destination 192.168.20.5:22
```

Using this command, all packets reaching 172.16.20.101 with port number 522 will be forwarded to 192.168.20.5:22.

Note: here, "172.16.20.101" is the registered IP address of the AWS EC2 instance and "192.168.20.5" is the local IP address of the OpenStack VM. Also worth noting is that "172.16.20.101" is already an NAT with AWS Elastic IP, which means all traffic that comes to the elastic IP (public IP) will be forwarded to this VPC local IP (172.16.20.101).

It means you could SSH the OpenStack VM globally by using the elastic IP address and the respective port number.

Elastic IP address

For this type of customized OpenStack installation, we required at least two NICs for the AWS EC2 instance:

1. One for accessing the VM terminal for the installation and for accessing the dashboard. In short, it acts as a Management network/ VM Tunnel network / API network.
2. Another for an external network with unique type interface configuration and mapped with provider network bridge (br-ex with eth1).

AWS will not allow any packets to travel out of the VPC unless the elastic IP is attached with that IP address.

To overcome this problem, we must attach the elastic IP to this NIC so that the OpenStack VM's packets will reach the OpenStack router's gateway. From the gateway, the packets will be embedded with the registered MAC address and matching IP address and will be sent to the AWS switch (VPC environment) via *br-ex* and *eth1* (special type interface configuration) on its way to AWS's actual VPC gateway. From there, the packets will reach the internet.

Other cloud platforms

In my analysis, most cloud providers (like DreamHost and Auro-cloud) have the same limitations for OpenStack networking. So we could use the tricks/hacks mentioned above in any of those cloud providers to run an OpenStack cloud on top of it.
Note: – Since we are using an QEMU emulator without KVM for the nested hypervisor environment, the VM performance will be slow.

If you want to try OpenStack on AWS, you can register for CloudEnabler's Cloud Lab-as-a-Service, (https://claas.corestack.io/clab#/clab/0) which provides a consistent and on-demand lab environment for OpenStack in AWS.

Selvaraj (http://www.hellovinoth.com/)is a cloud engineer at CloudEnablers.

The post Deploying OpenStack on AWS (http://superuser.openstack.org/articles/deploying-openstack-aws-tutorial/) appeared first on OpenStack Superuser (http://superuser.openstack.org).

by Vinoth Kumar Selvaraj at March 03, 2017 12:21 PM (http://superuser.openstack.org/articles/deploying-openstack-aws-tutorial/)

March 02, 2017

Cisco Metacloud (https://communities.cisco.com/community/technology/cloudsolutions/cisco-metacloud/blog)

Cisco Cloud at the upstream OpenStack Project Technical Gathering (PTG) (https://communities.cisco.com/community/technology/cloudsolutions/cisco-metacloud/blog/2017/03/02/cisco-cloud-at-the-upstream-openstack-project-technical-gathering-ptg)

Last week we had a great time at the upstream OpenStack Project Technical Gathering (PTG) in Atlanta, Georgia. I talked to a few of us at Cisco who had the opportunity to participate, asking similar questions of everyone:

What did you work on at the PTG?

What surprised you about the PTG?

What are you looking forward to at the next PTG?

Rob Cresswell, Project Technical Lead (PTL) for the OpenStack Dashboard (horizon), Cisco engineer

Mostly, I spoke with plugin authors about their concerns, and then did a ton of project management as a group. We worked on priorities, working out contributor availability, then a load of blueprint review, so we have a really nicely organised list of targets for Pike. I was surprised by how relaxed it was! I don't know if it was because there were 5000 less people, or the agendas were more open, but it felt much more like a bunch of programmers solving problems than some tense, crowded environment at the summit. I'm looking forward to improvements in the tooling around scheduling. It wasn't very transparent or organised. This seems to be a common concern though, so I think it will be addressed. I'd also really like it to be outside the US, but that seems unlikely. :)

Anne Gentle, Technical Product Manager for Cisco Metacloud

For my part, I listened and facilitated at the documentation sessions, met with the interoperability team to figure out why my refstack tests weren't working, got them working (score!), figured out a development environment to work on the Dashboard, started a patch for the Dashboard, and answered any questions I could about the OpenStack API documentation efforts. I I was surprised by the number of attendees who came from overseas, the global community was well-represented! I am looking forward to more relaxed collaboration as it was a distraction-free week.

Britt Houser, Cisco engineer

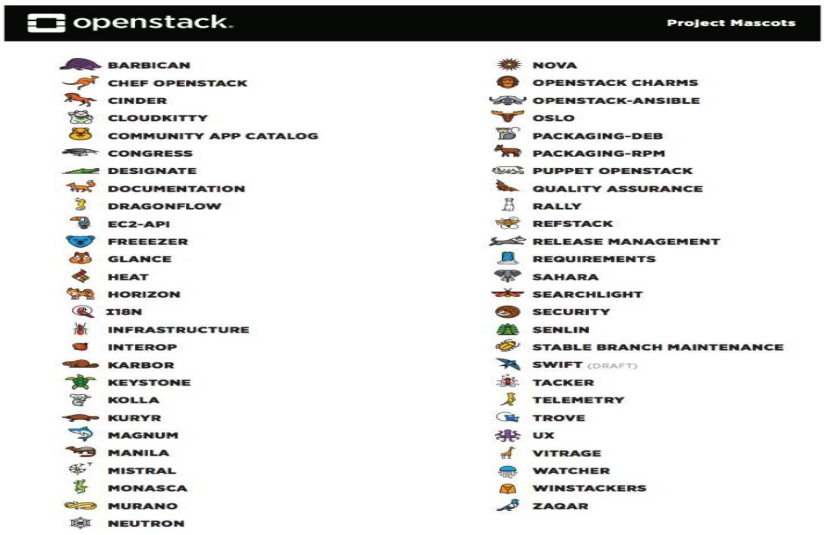
I split my time between Kolla and Ironic sessions. Between sessions I caught up with a lot of people (even from Cisco) that I don't get to see face-to-face except at OpenStack events. Making that personal connection is way more important that anything that goes on in the sessions.

I was surprised at how fast the mascot stickers disappeared! I'm looking forward to being able to put faces with IRC handles for people who haven't yet joined the community.

Nicolas Simonds, Cisco Metacloud engineer

Ostensibly I was there to work on Nova, but I did find a small bug with Glance and submitted a patch while I was there. So the answer is different based on whether you use "intention" versus "observed output." I was really surprised that the stated intention of "increasing/improving collaboration and communication between teams" was realized so effectively on the first go-round. It wasn't perfect, as everybody is still "finding their feet", but it was an auspicious start. To that end, I'm looking forward to seeing the refinements around cross-project collaboration, and the kinds of software that such things produces.

If you haven't gotten the cheat sheet for all the project mascots, here you go! These were used for signage and stickers and a lot of fun to see.



(https://communities.cisco.com/servlet/JiveServlet/showImage/38-8717-105061/openstack-mascots.jpg)

March 02, 2017 07:49 PM (https://communities.cisco.com/community/technology/cloudsolutions/cisco-metacloud/blog/2017/03/02/cisco-cloud-at-the-upstream-openstack-project-technical-gathering-ptg)

About

Planet OpenStack is a collection of thoughts from the developers and other key players of the OpenStack projects. If you are working on OpenStack technology you should add your OpenStack blog (http://wiki.openstack.org/AddingYourBlog).

Subscriptions

(http://blog.aaronorosen.com/category/openstack/feed/) Aaron Rosen (http://blog.aaronorosen.com)
(http://blog.adamspiers.org/tag/openstack/feed/atom/) Adam Spiers (http://blog.adamspiers.org)
(http://adam.younglogic.com/category/software/openstack/feed/) Adam Young (http://adam.younglogic.com)
(http://www.blogger.com/feeds/5556854748152045563/posts/default/-/OpenStack) Aditya Patawari (http://blog.adityapatawari.com/search/label/OpenStack)
(http://www.17od.com/tag/openstack/feed/) Adrian Smith (http://www.17od.com)
(https://cloudbase.it/category/openstack/feed/) Alessandro Pilotti (https://cloudbase.it)
(https://aababilov.wordpress.com/category/openstack-2/feed/) Alessio Ababilov (https://aababilov.wordpress.com)
(https://alvarolopez.github.io/feeds/openstack.atom.xml) Alvaro Lopez Garcia (http://alvarolopez.github.io/)
(http://www.blogger.com/feeds/1855272325744698559/posts/default) Amar Kapadia (http://www.buildcloudstorage.com/)
(https://hypecycles.com/category/OpenStack/feed/) Amrith Kumar (https://hypecycles.com)
(http://amalagon.github.io/blog/categories/openstack/atom.xml) Ana Malagon (http://amalagon.github.io/)
(http://www.blogger.com/feeds/8789296710216751709/posts/default/-/OpenStack) Andreas Jaeger (http://jaegerandi.blogspot.com/search/label/OpenStack)
(https://virtualandy.wordpress.com/tag/openstack/feed/atom/) Andy Hill (https://virtualandy.wordpress.com)
(https://ahsalkeld.wordpress.com/category/openstack/feed/atom/) Angus Salkeld (https://ahsalkeld.wordpress.com)
(http://anteaya.info/atom.xml) Anita Kuno (http://anteaya.info/)
(http://justwriteclick.com/tag/openstack/feed/) Anne Gentle (https://justwriteclick.com)
(http://ops.anthonygoddard.com/blog/categories/openstack/atom.xml) Anthony Goddard (http://ops.anthonygoddard.com/)
(https://www.reversengineered.com/tag/openstack/feed/) Antony Messerli (https://www.reversengineered.com)
(http://blog.appformix.com/rss.xml?tag=openstack) AppFormix (http://blog.appformix.com)
(https://feeds.feedburner.com/Aptira?format=xml) Aptira (https://aptira.com)
(http://abregman.com/category/openstack/feed/) Arie Bregman (http://abregman.com)
(http://www.berezins.com/category/openstack/feed/) Arthur Berezin (http://www.berezins.com)
(https://notartom.net/category/openstack/feed/) Artom Lifshitz (https://notartom.net)
(https://arunsag.wordpress.com/tag/openstack/feed/) Arun S A G (https://arunsag.wordpress.com)
(http://old.arxcruz.net/category/openstack/feed/) Arx Cruz (http://old.arxcruz.net)
(https://assafmuller.com/category/openstack/feed/) Assaf Muller (https://assafmuller.com)
(http://www.blogger.com/feeds/3693502231491518239/posts/default/-/openstack) Augustina Ragwitz (http://hackhackblog.blogspot.com/search/label/openstack)
(http://blog.nemebean.com/taxonomy/term/3/feed) Ben Nemec (http://blog.nemebean.com/tags/openstack)
(http://benjaminkerensa.com/category/openstack/feed) Benjamin Kerensa (http://benjaminkerensa.com)
(http://blog.cafarelli.fr/category/openstack/feed/) Bernard Cafarelli (http://blog.cafarelli.fr)
(https://blog.betacloud.io/tag/openstack-planet/feed/atom/) Betacloud (https://betacloud.io)
(http://www.logicative.com/category/openstack/feed/) Beth Elwell (http://www.logicative.com)
(http://www.blogger.com/feeds/8711250493465785199/posts/default/-/openstack) Boden Russell (http://bodenr.blogspot.com/search/label/openstack)
(http://www.bretpiatt.com/blog/category/openstack/feed/) Bret Piatt (http://www.bretpiatt.com/blog)
(http://blog.briancurtin.com/categories/openstack.xml) Brian Curtin (http://briancurtin.com/blog/)
(http://www.blogger.com/feeds/1977547450130135970/posts/default/-/OpenStack) Cameron Seader (http://blog.seader.us/search/label/OpenStack)
(http://captainkvm.com/tag/openstack/feed/) Captain KVM (http://captainkvm.com)
(http://blog.episodicgenius.com/tags/openstack/index.xml) Carl Baldwin (http://blog.episodicgenius.com/tags/openstack/index.xml)
(http://anstack.github.io/atom.xml) Carlos Camacho (http://anstack.github.io/)

(http://ceph.com/community/blog/tag/openstack/feed/) Ceph (http://ceph.com)
(http://chandankumar.dgplug.org/categories/openstack.xml) Chandan Kumar (http://chandankumar.dgplug.org/)
(https://blog.chmouel.com/category/openstack/feed/) Chmouel Boudjnah (https://blog.chmouel.com)
(https://anticdent.org/feeds/openstack.atom.xml) Chris Dent (https://anticdent.org/)
(http://chris.yeoh.info/category/openstack/feed/) Chris Yeoh (http://chris.yeoh.info)
(https://blog.christophersmart.com/category/foss/openstack/feed/) Christopher Smart (https://blog.christophersmart.com)
(http://cpallar.es/tagged/openstack/rss) Cindy Pallares (http://cpallar.es/)
(http://blogs.cisco.com/tag/openstack/feed) Cisco Cloud Blog (http://blogs.cisco.com)
(https://communities.cisco.com/community/technology/cloudsolutions/cisco-metacloud/blog/feeds/posts) Cisco Metacloud (https://communities.cisco.com/community/technology/cloudsolutions/cisco-metacloud/blog)
(http://citrix-openstack.siteleaf.net/feed.xml) Citrix OpenStack Blog (http://citrix-openstack.siteleaf.net/)
(http://fewbar.com/feed.xml) Clint Byrum (http://fewbar.com/)
(https://www.symantec.com/connect/item-feeds/all/all/feed/all/all/3249921) Cloud Platform @ Symantec (https://www.symantec.com/connect/item-feeds/all/all/feed/all/all/3249921)
(https://cloudbau.github.io/feed.openstack.xml) Cloudbau Blog (https://cloudbau.github.io)
(http://getcloudify.org/tags/OpenStack/atom.xml) Cloudify Engineering
(http://www.cloudscaling.com/blog/category/openstack/feed/) Cloudscaling Corporate Blog (http://www.cloudscaling.com)
(http://feeds.feedburner.com/SimplicityScales?format=xml) Cloudscaling Engineering (http://engineering.cloudscaling.com)
(https://dev.cloudwatt.com/en/blog/rss.xml) Cloudwatt (https://dev.cloudwatt.com/en/blog/index.html)
(http://openstack.prov12n.com/category/openstack/feed/) Cody Bunch (http://openstack.prov12n.com)
(https://mcwhirter.com.au/tags/OpenStack/index.atom) Craige McWhirter (http://mcwhirter.com.au//tags/OpenStack/)
(https://dronopenstack.wordpress.com/category/openstack/feed/) Dafna Ron (https://dronopenstack.wordpress.com)
(http://blog.thekingshotts.com/category/openstack/feed/) Dan Kingshott (http://blog.thekingshotts.com)
(https://dprince.github.io/feeds/tag_openstack.atom.xml) Dan Prince (https://dprince.github.io/)
(http://www.jaddog.org/category/openstack/feed/) Dan Radez (http://www.jaddog.org)
(http://www.danplanet.com/blog/category/openstack/feed/) Dan Smith (http://www.danplanet.com/blog)
(https://www.berrange.com/topics/openstack/feed/) Daniel P. Berrangé (https://www.berrange.com)
(https://www.stackevolution.com/taxonomy/term/2/feed) Darryl Weaver (https://www.stackevolution.com/openstack)
(http://www.blogger.com/feeds/9385141942695367 8/posts/default/-/openstack) David Medberry (http://dowdberry.blogspot.com/search/label/openstack)
(https://dmsimard.com/category/openstack/rss.xml) David Moreau Simard (https://dmsimard.com/)
(https://dstanek.com/tags/openstack/index.xml) David Stanek (https://dstanek.com/tags/openstack/index.xml)
(http://hackstack.org/x/blog/category/openstack/feed/) Dean Troyer (http://hackstack.org/x/blog)
(http://goodsquishy.com/tag/openstack/feed/atom/) Derek Higgins (http://goodsquishy.com)
(http://feeds.doughellmann.com/doughellmann/openstack) Doug Hellmann (https://doughellmann.com/blog)
(http://www.dougalmatthews.com/feeds/tag/openstack.atom.xml) Dougal Matthews (http://www.dougalmatthews.com/)
(http://www.blogger.com/feeds/1999608034451064872/posts/default/-/OpenStack) Dragonflow Team (http://www.dragonflow.net/search/label/OpenStack)
(https://www.dreamhost.com/blog/tag/openstack/feed/) DreamHost (https://www.dreamhost.com/blog)
(http://www.blogger.com/feeds/8825992/posts/default/-/openstack) Duncan McGreggor (http://oubiwann.blogspot.com/search/label/openstack)
(https://blog.leafe.com/category/openstack/feed/) Ed Leafe (https://blog.leafe.com)
(http://princessleia.com/journal/category/openstack/feed/) Elizabeth K. Joseph (http://princessleia.com/journal)
(http://my1.fr/blog/category/virtualization/openstack/feed/atom/) Emilien Macchi (http://my1.fr/blog)
(https://enriquetaso.com/category/OpenStack/feed/) Enriquez Laura Sofia (https://enriquetaso.com)
(http://itsonlyme.name/openstack/storlets/rss.xml) Eran Rom (http://itsonlyme.name/blog)
(http://blog.phymata.com/category/openstack.atom.xml) Everett Toews (http://blog.phymata.com/)
(http://blog.fsquat.net/?tag=openstackplanet&feed=rss2) Fabien Boucher (http://blog.fsquat.net)
(http://blog.flaper87.com/feeds/openstack.atom.xml) Flavio Percoco (https://blog.flaper87.com/)
(https://fleio.com/blog/tag/openstack/feed/) Fleio Blog (https://fleio.com/blog)
(http://www.florentflament.com/blog/feeds/openstack.atom.xml) Florent Flament (http://www.florentflament.com/blog/)
(http://galsagie.github.io/feed.xml) Gal Sagie (http://galsagie.github.io)
(http://galeracluster.com/category/planet-openstack/feed/) Galera Cluster by Codership (http://galeracluster.com/category/blog/)
() Geetika Batra
(http://giuliofidente.com/feeds/tag/openstack.atom.xml) Giulio Fidente (http://giuliofidente.com/)
(http://gorka.eguileor.com/feed/?cat=10&feed=rss2) Gorka Eguileor (https://gorka.eguileor.com)
(http://graham.hayes.ie/categories/cat_openstack.xml) Graham Hayes (http://graham.hayes.ie/)
(https://openstackgd.wordpress.com/feed/) Grid Dynamics OpenStack Team (https://openstackgd.wordpress.com)
(http://community.hpe.com/html/assets/hp-maint.png) HP Public Cloud (http://www.hpcloud.com/blog/categories/1246)
(http://ehaselwanter.com/en/openstack.xml) Haselwanter Edmund (http://ehaselwanter.com)
(http://blog.hendrikvolkmer.de/feed/OpenStack/atom.xml) Hendrik Volkmer (http://blog.hendrikvolkmer.de/)
(http://hugh.blemings.id.au/category/openstack/feed/) Hugh Blemings (http://hugh.blemings.id.au)
(https://developer.ibm.com/opentech/category/openstack/feed/) IBM OpenTech Team (https://developer.ibm.com/opentech)
(https://blog.zhaw.ch/icclab/category/articles/openstack-2/feed/) ICCLab (https://blog.zhaw.ch/icclab)
(http://drop.isi.edu/node/407/rss) ISI High Performance Cloud Computing Group (http://www.isi.edu/research_groups/HPCC/blog)
(http://blog.imaginea.com/tag/openstack/) Imaginea Technologies
(https://blog.imaginea.com/tag/openstack/feed/) Imaginea Technologies (https://blog.imaginea.com)
(http://www.infralovers.com/en/articles/categories/openstack/atom.xml) Infralovers (http://www.infralovers.com/)
(http://www.internap.com/feed/?cat=1001410) Internap (http://www.internap.com)
(http://jjasghar.github.io/blog/categories/openstack/atom.xml) JJ Asghar (http://jjasghar.github.io/)
(http://amo-probos.org/feed/rss2_0?tag=openstack) James E. Blair (http://amo-probos.org/)
(https://javacruft.wordpress.com/category/openstack/feed/) James Page (https://javacruft.wordpress.com)
(http://blog-slagle.rhcloud.com/?feed=rss2&tag=OpenStack) James Slagle (http://blog-slagle.rhcloud.com)
(http://www.jamielennox.net/blog/categories/openstack/atom.xml) Jamie Lennox (http://www.jamielennox.net/)
(http://www.jpena.net/?cat=7&feed=atom) Javier Peña (http://www.jpena.net)

(http://www.jasondotstar.com/feed.openstack.xml) Jay Clark (http://www.jasondotstar.com)
(http://www.joinfu.com/category/openstack/feed/) Jay Pipes (http://www.joinfu.com)
(http://odyssey4me.github.io/openstack.xml) Jesse Pretorius (http://odyssey4me.github.io/)
(https://software.intel.com/en-us/user/335367/feed) Jiangang Duan (Intel) (https://software.intel.com/en-us/recent/335367)
(https://jfehlig.wordpress.com/category/openstack/feed/) Jim Fehlig (https://jfehlig.wordpress.com)
(https://tropicaldevel.wordpress.com/category/openstack/feed/) John Bresnahan (https://tropicaldevel.wordpress.com)
(http://programmerthoughts.com/openstack/openstackposts.xml) John Dickinson (http://programmerthoughts.com/)
(http://john.eckersberg.com/feeds/tag/openstack.atom.xml) John Eckersberg (http://john.eckersberg.com/)
(https://griffithscorner.wordpress.com/category/openstack/feed/) John Griffith (https://griffithscorner.wordpress.com)
(http://blog.iwokeupcoveredinblood.com/blog/categories/openstack/atom.xml) Jon Proulx (http://blog.iwokeupcoveredinblood.com/)
(http://josh.people.rcbops.com/tag/openstack/feed/) Joshua Hesketh (http://josh.people.rcbops.com)
(http://www.jpichon.net/feeds/atom/tag/openstack/) Julie Pichon (http://www.jpichon.net/tag/openstack/)
(https://julien.danjou.info/blog/tags/OpenStack.xml) Julien Danjou (https://julien.danjou.info/blog/)
(http://www.juliosblog.com/tag/openstack/rss/) Julio Villarreal Pelegrino (http://www.juliosblog.com/)
(https://kashyapc.com/tag/openstack/feed/) Kashyap Chamarthy (https://kashyapc.com)
(https://cloud-guy.net/category/openstack/feed/atom/) Keith Tobin (https://cloud-guy.net)
(http://ken.pepple.info/openstack.atom) Ken Pepple (http://ken.pepple.info)
(https://cloudarchitectmusings.com/tag/openstack/feed/) Kenneth Hui (https://cloudarchitectmusings.com)
(https://kimizhang.wordpress.com/category/openstack/feed/) Kimi Zhang (https://kimizhang.wordpress.com)
(http://www.siliconloons.com/categories/openstack/index.xml) Kyle Mestery (https://www.siliconloons.com/categories/openstack/)
(http://lbragstad.com/category/openstack/feed/) Lance Bragstad (http://lbragstad.com)
(http://blog.oddbit.com/tag/openstack/rss/) Lars Kellogg-Stedman (http://blog.oddbit.com/)
(http://www.thequietstack.com/?tag=openstack&feed=rss) Laura Alves (http://www.thequietstack.com)
(http://uxd-stackabledesign.rhcloud.com/category/openstack/feed/) Liz Blanchard (http://uxd-stackabledesign.rhcloud.com)
(http://feeds.feedburner.com/logilaborg_openstack) Logilab (https://www.logilab.org/view?rql=Any%20X%20ORDERBY%20AA%20DESC%20LIMIT%2010%20WHERE%20E%20eid%20115182%2C%20E%20tags%20X%2C%20X%20creation_date%20AA%2C%20X%20i
(https://lorinhochstein.wordpress.com/category/openstack/feed/atom/) Lorin Hochstein (https://lorinhochstein.wordpress.com)
(http://dachary.org/?feed=rss2&cat=32) Loïc Dachary (http://dachary.org)
(http://www.blogger.com/feeds/5819640694385843490/posts/default/-/OpenStack) Maish Saidel-Keesing (http://technodrone.blogspot.com/search/label/OpenStack)
(https://major.io/tag/openstack/feed/) Major Hayden (https://major.io)
(http://www.blogger.com/feeds/7208049467061137140/posts/default/-/openstack) Manishanker Talusani (http://manishankert.blogspot.com/search/label/openstack)
(http://blog.mbonell.com/category/openstack/feed/) Marcela Bonell (http://blog.mbonell.com)
(https://crustyblaa.com/feeds/openstack.atom.xml) Mark McLoughlin (https://crustyblaa.com/)
(http://www.markshuttleworth.com/archives/tag/openstack/feed) Mark Shuttleworth (http://www.markshuttleworth.com)
(http://markvoelker.github.io/blog/index.xml) Mark T. Voelker (http://markvoelker.github.io/blog/)
(http://hybridcloudburst.com/category/openstack/feed/atom/) Marten Hauville (https://hybridcloudburst.wordpress.com)
(http://mkissam.tumblr.com/tagged/openstack/rss) Marton Kiss (http://mkissam.tumblr.com/)
(https://domsch.com/blog/?feed=atom&tag=openstack) Matt Domsch (https://domsch.com/blog)
(http://www.madorn.com/feeds/openstack.rss.xml) Matt Dorn (http://www.madorn.com/)
(https://engineeredweb.com/tag/OpenStack%20Planet/atom.xml) Matt Farina (https://engineeredweb.com/tag/OpenStack%20Planet/)
(http://www.mattfischer.com/blog/?feed=rss2&cat=50) Matt Fischer (http://www.mattfischer.com/blog)
(http://www.mattgriffin.com/tag/openstack/feed/) Matt Griffin (http://www.mattgriffin.com)
(https://leastresistance.wordpress.com/category/openstack/feed/) Matt Ray (https://leastresistance.wordpress.com)
(https://spinningmatt.wordpress.com/category/openstack/feed/atom/) Matthew Farrellee (https://spinningmatt.wordpress.com)
(http://blog.kortar.org/?cat=2&feed=rss2) Matthew Treinish (http://blog.kortar.org)
(https://www.matthias-runge.de/tags/openstack.atom.xml) Matthias Runge (http://www.matthias-runge.de/)
(https://blog.sileht.net/feeds/openstack.atom.xml) Mehdi Abaakouk (https://blog.sileht.net/)
(http://www.blogger.com/feeds/10113432198016145525/posts/default/-/openstack) Michael Davies (http://lifelog.michaeldavies.org/search/label/openstack)
(https://krotscheck.net/tag/openstack/feed) Michael Krotscheck (https://krotscheck.net)
(http://www.stillhq.com/openstack/index.rss) Michael Still (http://www.stillhq.com)
(http://acksyn.org/categories/openstack.xml) Michele Baldessari (http://acksyn.org/)
(https://blog.midonet.org/category/openstack/feed/) MidoNet (https://blog.midonet.org)
(http://blog.midokura.com/category/OpenStack/feed/) Midokura (http://blog.midokura.com)
(http://www.ajo.es/tagged/openstack/rss) Miguel Ángel Ajo (http://www.ajo.es/)
(http://www.blogger.com/feeds/4447965219658920183/posts/default/-/Openstack) Mika Ayenson (http://ayenson.blogspot.com/search/label/Openstack)
(http://www.dorm.org/blog/category/openstack/feed/) Mike Dorman (http://www.dorm.org/blog)
(https://mirandazhangq.wordpress.com/category/openstack/feed/atom/) Miranda ZHANG (https://mirandazhangq.wordpress.com)
(https://www.mirantis.com/blog/feed/) Mirantis (https://www.mirantis.com)
(http://inaugust.com/feed/rss2_0?tag=openstack) Monty Taylor (http://inaugust.com/)
(https://blog.nfvpe.site/feed/) NFVPE @ Red Hat (https://blog.nfvpe.site)
(https://blog-nkinder.rhcloud.com/?cat=2&feed=atom) Nathan Kinder (https://blog-nkinder.rhcloud.com)
(https://thenetworkway.wordpress.com/tag/openstack/feed/) Nir Yechiel (https://thenetworkway.wordpress.com)
(https://osic.org/blog/feed) OSIC - The OpenStack Innovation Center (https://osic.org/blog)
(https://www.olindata.com/taxonomy/term/309/feed) OlinData (https://www.olindata.com/tags/openstack)
(http://netapp.io/category/openstack/) OpenStack @ NetApp
(https://www.openstack.org/blog/feed/) OpenStack Blog (https://www.openstack.org/blog)
(https://openstackindia.wordpress.com/feed/) OpenStack India (https://openstackindia.wordpress.com)
(http://openStackindiaonline.github.io/feeds/all.atom.xml) OpenStack India Online (http://openStackindiaonline.github.io/)
(https://novarollup.wordpress.com/feed/) OpenStack Nova Developer Rollup (https://novarollup.wordpress.com)
(https://blog.chmouel.com/) OpenStack Reactions (https://blog.chmouel.com)

(http://secstack.org/category/openstack/feed/) OpenStack Security Blog (http://secstack.org)
(http://superuser.openstack.org/feed/) OpenStack Superuser (http://superuser.openstack.org)
(http://www.blogger.com/feeds/6032896665180559/posts/default) OpenStack in Production (http://openstack-in-production.blogspot.com/)
(https://opensource.com/taxonomy/term/5126/feed/feed) Opensource.com (https://opensource.com/taxonomy/term/5126/feed/feed)
(https://openstack-security.github.io/feed.xml) Openstack Security Project (https://openstack-security.github.io/)
(http://www.blogger.com/feeds/7675777825226307020/posts/default/-/openstack) Openstack-br (http://openstackbr.blogspot.com/search/label/openstack)
(http://mezzanine-lordkrandel.rhcloud.com/tag/openstack/feeds/rss/) Paolo Gatti (http://mezzanine-lordkrandel.rhcloud.com/)
(https://www.percona.com/blog/category/openstack/feed/) Percona (https://www.percona.com/blog)
(http://www.cisco.com/c/en/us/about/corporate-strategy-office/acquisitions/piston.html) Piston (http://pistoncloud.com)
(http://www.pixelbeat.org/feed/openstack.rss2.xml) Pádraig Brady (http://www.pixelbeat.org/)
(http://rdoproject.org/blog/feed.xml) RDO (http://rdoproject.org/blog/)
(https://blogs.rdoproject.org/category/openstackdev/feed) RDO Blogs (http://blogs.rdoproject.org)
(https://developer.rackspace.com/blog/categories/openstack/atom.xml) Rackspace Developer Blog (https://developer.rackspace.com)
(https://trickycloud.wordpress.com/category/openstack/feed/atom/) Ramon Acedo (https://trickycloud.wordpress.com)
(https://www.ravellosystems.com/blog/tag/openstack/feed) Ravello Systems (https://www.ravellosystems.com/blog)
(http://redhatstackblog.redhat.com/feed/) Red Hat Stack (http://redhatstackblog.redhat.com)
(http://drbacchus.com/tag/openstack/feed/) Rich Bowen (http://drbacchus.com)
(http://mechanicalcat.net/richard/log/OpenStack/rss) Richard Jones (http://mechanicalcat.net/richard/log/OpenStack)
(https://robhirschfeld.com/category/openstack/feed/) Rob Hirschfeld (https://robhirschfeld.com)
(https://rbtcollins.wordpress.com/tag/openstack/feed/atom/) Robert Collins (https://rbtcollins.wordpress.com)
(http://ronaldbradford.com/blog/category/cloud/openstack/feed/) Ronald Bradford (http://ronaldbradford.com/blog)
(http://roozbehshafiee.com/category/openstack/feed/) Roozbeh Shafiee (http://roozbehshafiee.com)
(https://rossella-sblendido.net/category/openstack/feed/) Rossella Sblendido (https://rossella-sblendido.net)
(https://blog.russellbryant.net/category/openstack/feed/) Russell Bryant (https://blog.russellbryant.net)
(http://feeds.feedburner.com/RyanLanesBlog_openstack) Ryan Lane (https://blog.ryandlane.com)
(https://www.suse.com/communities/blog/tag/openstack/feed/) SUSE Conversations (https://www.suse.com/communities/blog)
(https://cloudblog.switch.ch/category/openstack/feed/atom/) SWITCH Cloud Blog (https://cloudblog.switch.ch)
(http://blog.mathys.io/feeds/posts/default/-/OpenStack?alt=rss) Sandro Mathys (http://blog.mathys.io/search/label/OpenStack)
(https://saschpe.wordpress.com/category/openstack/feed/) Sascha Peilicke (https://saschpe.wordpress.com)
(https://dague.net/category/openstack/feed/) Sean Dague (https://dague.net)
(http://sarob.com/category/openstack/feed/) Sean Roberts (https://sarob.com)
(https://shaifaliagrawal.wordpress.com/category/openstack/feed/) Shaifali Agrawal (https://shaifaliagrawal.wordpress.com)
(http://www.debug-all.com/?feed=rss2&tag=openstack) Shannon McFarland (http://www.debug-all.com)
(http://solinea.com/tag/openstack/feed) Solinea (http://solinea.com)
(http://blog.warma.dk/tag/planetopenstack/feed/) Soren Hansen (http://blog.linux2go.dk)
(http://engineering.spilgames.com/category/planetopenstack/feed/) Spilgames Engineering (http://engineering.spilgames.com)
(https://www.stackhpc.com/feeds/stackhpc.atom.xml) StackHPC Team Blog (https://www.stackhpc.com/)
(http://www.stackmasters.eu/tag/openstack/feed/) Stackmasters team (http://www.stackmasters.eu)
(https://maffulli.net/category/openstack/feed/) Stefano Maffulli (https://maffulli.net)
(http://www.blogger.com/feeds/7938511026917047919/posts/default/-/openstack) Steve Baker (http://blog.stevebaker.org/search/label/openstack)
(https://sdake.io/tag/openstack/feed/) Steve Dake (https://sdake.io)
(http://www.blogger.com/feeds/7962934533489004582/posts/default/-/openstack) Steve Hardy (http://hardysteven.blogspot.com/search/label/openstack)
(http://blog.coolsvap.net/category/openstack/feed/) Swapnil Kulkarni (http://blog.coolsvap.net)
(http://swiftstack.com/blog/categories/planetopenstack/atom.xml) SwiftStack Team (http://swiftstack.com/)
(https://sbauza.wordpress.com/tag/openstack/feed/) Sylvain Bauza (https://sbauza.wordpress.com)
(http://www.sebastien-han.fr/openstack.xml) Sébastien Han (https://sebastien-han.fr/)
(http://telekomcloud.github.io/atom.xml) TelekomCloud DevOps team (http://telekomcloud.github.io)
(http://terriyu.info/blog/feeds/tag/openstack.atom.xml) Terri Yu (http://terriyu.info/blog/)
(http://blog.otherwiseguy.com/tag/openstack/rss/) Terry Wilson (http://blog.otherwiseguy.com/)
(http://www.stratoscale.com/products/stratoscale-acquires-tesora-database-as-a-service/) Tesora Corp (http://www.tesora.com)
(https://blog.rackspace.com/tag/openstack/feed) The Official Rackspace Blog (https://blog.rackspace.com)
(https://storyboard-blog.sotk.co.uk/) The StoryBoard Team
(https://ttx.re/feeds/openstack.atom.xml) Thierry Carrez (https://ttx.re/)
(https://toabctl.wordpress.com/tag/openstack/feed/) Thomas Bechtold (https://toabctl.wordpress.com)
(https://ubuntuserver.wordpress.com/tag/openstack/feed/) Ubuntu Server & Cloud blog (https://ubuntuserver.wordpress.com)
(http://vmartinezdelacruz.com/category/OpenStack/feed/) Victoria Martínez de la Cruz (http://vmartinezdelacruz.com)
(http://www.vuntz.net/journal/feed/tag/openstack/atom) Vincent Untz (http://www.vuntz.net/journal/)
(https://blog.xenproject.org/tag/openstack/feed/) Xen Project Blog (https://blog.xenproject.org)
(http://www.blogger.com/feeds/2528454834564901150/posts/default/-/openstack) Yun Mao (http://cloudystuffhappens.blogspot.com/search/label/openstack)
(http://www.zerobanana.com/feeds/tags/OpenStack) Zane Bitter (http://www.zerobanana.com/tags/OpenStack)
(http://www.zmanda.com/blogs/?feed=rss2&cat=22) Zmanda (http://www.zmanda.com/blogs)
(https://www.hastexo.com/feeds/openstack.atom.xml) hastexo (https://www.hastexo.com/)
(http://sites.rcbops.com/lca2014_openstack/feed/) linux.conf.au 2014 OpenStack miniconf (http://sites.rcbops.com/openstack_miniconf)
(https://lizards.opensuse.org/tag/openstack/feed/) openSUSE Lizards (https://lizards.opensuse.org)

Last updated:
March 23, 2017 03:24 PM
All times are UTC.

Powered by:
PLANET (http://www.planetplanet.org/)

OpenStack

About the Foundation (/foundation)
Projects (http://openstack.org/projects/)
OpenStack Security (http://openstack.org/projects/openstack-security/)
Common Questions (http://openstack.org/projects/openstack-faq/)
Blog (http://openstack.org/blog/)

Community

User Groups (http://openstack.org/community/)
Events (http://openstack.org/community/events/)
Jobs (http://openstack.org/community/jobs/)
Companies (http://openstack.org/foundation/companies/)
Contribute (https://wiki.openstack.org/wiki/How_To_Contribute)

Documentation

OpenStack Manuals (http://docs.openstack.org)
Getting Started (http://openstack.org/software/start/)
API Documentation (http://developer.openstack.org)
Wiki (https://wiki.openstack.org)

Branding & Legal

Logos & Guidelines (http://openstack.org/brand/)
Trademark Policy (http://openstack.org/brand/openstack-trademark-policy/)
Privacy Policy (http://openstack.org/privacy/)
OpenStack CLA (https://wiki.openstack.org/wiki/How_To_Contribute#Contributors_License_Agreement)

Stay In Touch

(https://twitter.com/OpenStackFoundation)