MIGRATION IN OPENSTACK LIBERTY

Dr. Shankariah[1], Kalma TMR[2]

Professor (EC Department) [1], PG Student (NIE) [2]
[1,2] S.J.C.E College of Engineering, Mysuru, India

Abstract: **A Cloud is a virtual asset convenient to install the applications, whereas Cloud Computing is a general term for anything that involves delivering hosted services over the Internet. Open Stack is free open source cloud computing software initially launched by Rackspace and NASA, which aims to close the gap in the lack of a comprehensive cloud platform with a quick development and innovation, and assisted by both an active community of people and large companies. Cloud migration involves migrating the services or applications from live system to the cloud and planning to set up a basic cloud migration for managing load on running Open Stack environment. This presents the methodology of setting up migration in OpenStack Liberty environment. It serves as a reference for migration looking at the possibilities of applying cloud solutions, in order to profit from getting the basics and a view on the different aspects of cloud migration concepts.**

Keywords: **Cloud migration, Load balancing, Live migration, Static/Offline migration and Liberty.**

I. INTRODUCTION

Cloud is a buzzword in today’s technology trend. Cloud computing is a growing idea all over the globe. It brings the consumer ingress to data, applications and storage that are not stored on their own computer. Cloud Computing allows, using computing resources without keeping them through a process of Cloud Service Ordering. Cloud Computing allows to run IT environment and software applications without hosting them at own premises but using a virtual machine with disk space, processing power and bandwidth configuration including network functionality like load balancing and firewall setting and much more. Virtualization can exist without cloud computing, but cloud computing needs virtualization for its existence.

OpenStack is an open cloud operating system used for building clouds. The primary deployment of OpenStack is as Infrastructure as a service (IaaS). OpenStack began as a joint project from Rackspace and NASA in July 2010 and is maintained by OpenStack Foundation. OpenStack offers scalability, flexibility and has highly engaged global circle consisting of innovators, developers, software and hardware vendors, service providers, and group of individuals, OpenStack enthusiasts and more than 430 companies including Rackspace, Dell, HP, IBM, Red Hat and etc.

Migrating compute instances is very use full. It allows an administrator to free up a compute node for maintenance or updates. It also permits an administrator to finer distribution of resources between compute nodes. OpenStack provides a few different ways to migrate virtual instances from one compute node to another. OpenStack cloud migration scenarios and use case scenarios help to recognize the real business situations and spotlight where cloud migration technology is a part of the OpenStack cloud solution. The framework will involve private, public and hybrid cloud models. Here we will describe OpenStack private cloud deployment and migration in OpenStack.

II. RELATED WORK

OpenStack is a global collaboration of developers and cloud computing technologists providing the open source cloud computing platform for public and private clouds. The technology consists of a number of different interrelated projects all delivering various components which allow creation of cloud infrastructure. Cloud basically refers to the internet oriented services primarily being compute storage and network services. OpenStack is an open source tool to build the cloud environment based on the requirements. Also the deployment of OpenStack is a tricky business tactic depending upon the perspectives from user, admin or a developer. The OpenStack deployment refers to the process of installing the OpenStack and Configuring the OpenStack to suite the requirements.

OpenStack lets cloud consumers to deploy virtual machines and other instances which handle different tasks for managing a cloud environment on the fly. As a cloud service provider dedicated to deploying a variety of cloud solutions built on OpenStack are dealing on basis with issues of high availability, disaster recovery and security. High Availability is surely one of the key point in building any cloud infrastructure. Migration is a hugely sought-after face by cloud administrators it's used essentially to achieve zero downtime during cloud maintenance.

Usually there are three significant ways to deploy an OpenStack cloud they are Manual deployment, DevStack and PackStack. DevStack has evolved to support a large number of configuration options and alternative platforms and support services. PackStack is a service deployed using Puppet modules, it deploys various parts of OpenStack on multiple pre-installed servers automatically over SSH. Currently only Fedora , Red Hat Enterprise Linux (RHEL) and compatible derivatives of both are supported. The Manual deployment involves installing each OpenStack service manually. The Multi-node installation setup runs different OpenStack services on different nodes. A basic installation requires the three nodes they are Controller-node, Network-node and Compute-node. Controller Node runs control services, such as message queue, database and API services for the Identity Service (Keystone). Network Node runs networking services and is responsible for virtual networking needed for people to create private or public networks, and uplink their virtual machines into external networks. Compute Node runs the virtual machine instances in OpenStack. The Fig. 1 depicts OpenStack deployment on bare metal with multi-controllers architecture.

During cloud maintenance, this function is used to achieve zero downtime. A hugely sought-after feature by cloud administrators, it's used especially to achieve zero downtime in the time of cloud maintenance and can also be a useful feature to achieve performance as live instances can be transferred from a large loaded compute node to a less loaded compute node.

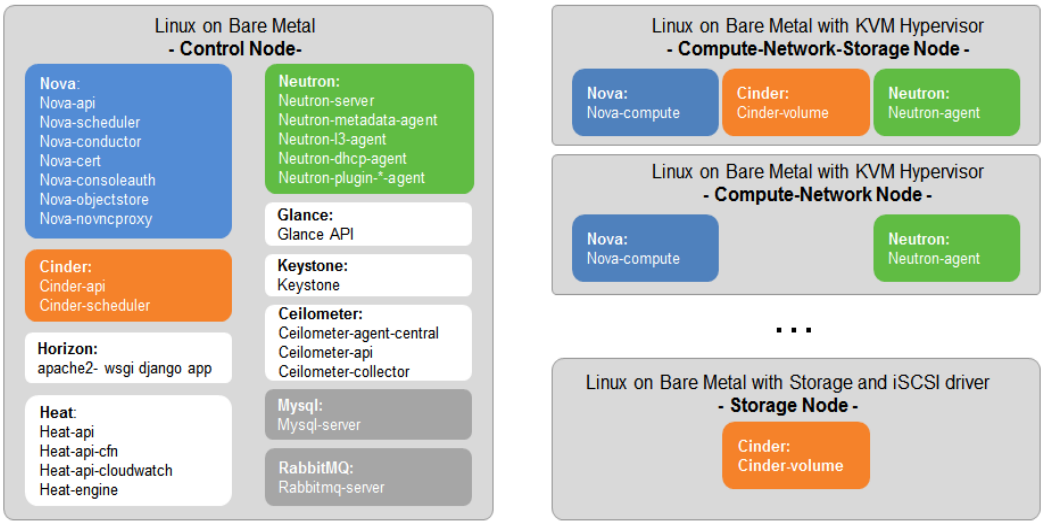


Fig.1: OpenStack deployed on Bare Metal with Multi-Controllers Topology

III. MIGRATE BETWEEN ENVIRONMENTS OR PROJECTS

We use existing migration feature in OpenStack and allow for verifiability of the feature. After describing the way to use migration feature, we propose a static and live based migration in Liberty. Migration is the procedure of moving data, applications or other business elements from an organization's location to the cloud, or moving them from one cloud environment to another. Cloud migration sometimes involves moving data or other organization business elements between cloud environments, which is known as cloud-to-cloud migration. This describes procedures for migrating a static instance between hypervisors in the same OpenStack cloud or between OpenStack environments (or projects). Here static refer to instance state, before migration we will shutoff instance and migrate. It is also known as offline migration.

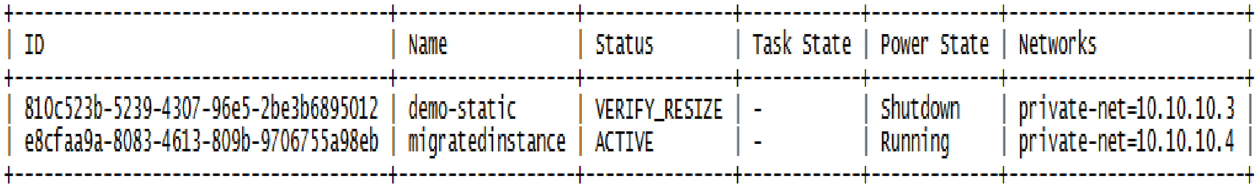
The instance will be shut down in order to be moved to another Compute Node, leading to instance downtime and all consequences of shutting down an instance. The destination of the instance cannot be defined, instead it is picked by the nova scheduler. Offline migrations can be invoked by the cloud user.

Launch instances on a private network using cirros image. Launch demo-static instance which will be used for static migration. The size of instance is 512 MB that is nothing but tiny flavor from nova flavor list. We are providing nova as availability zone. The name of instance is demo-static. SSH-tunneling steps are provided for user reference only. Only users with root access can set up SSH tunneling. To migrate instances between nodes using SSH tunneling or to resize instance in a distributed environment, each node must be set up with SSH key authentication so that the Compute service can use SSH to move disks to other nodes. For example, compute nodes could use the same SSH key to ensure communication. Then stop the running instance using “nova stop” command. Nova migrate command used for migrating instance.

nova migrate <instance name>:

The Compute service selects the target compute node based on service configuration (balancing weights and filters). Wait until the instance state has changed to VERIFY_RESIZE (can use nova list to see this), then confirm the migration with nova resize-confirm. We can check it with migration list. The Fig. 2 depicts status of migrating instance and Fig. 3 depicts list of migrated instance in OpenStack project.

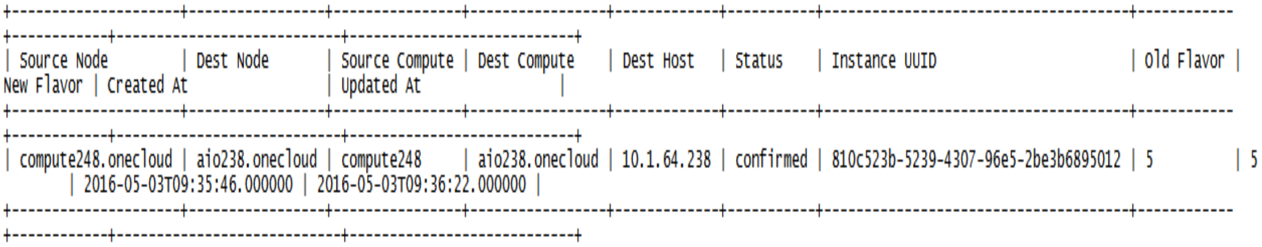
nova migrate <instance name>:



```
+--------------------------------------+-----------------+--------------+------------+-------------+-----------------------+
| ID                                   | Name            | Status       | Task State | Power State | Networks              |
+--------------------------------------+-----------------+--------------+------------+-------------+-----------------------+
| 810c523b-5239-4307-96e5-2be3b6895012 | demo-static     | VERIFY_RESIZE | -         | Shutdown    | private-net=10.10.10.3 |
| e8cfaa9a-8083-4613-809b-9706755a98eb | migratedinstance | ACTIVE      | -          | Running     | private-net=10.10.10.4 |
+--------------------------------------+-----------------+--------------+------------+-------------+-----------------------+
```

Fig.2: Migrating an Instance

nova migration-list:



```
+--------------------+------------------+---------------+-----------------+------------+----------+--------------------------------------+------------
+-----------+--------------------------+--------------------------+
| Source Node        | Dest Node        | Source Compute | Dest Compute   | Dest Host  | Status   | Instance UUID                        | Old Flavor |
New Flavor | Created At                | Updated At               |
+--------------------+------------------+---------------+-----------------+------------+----------+--------------------------------------+------------
+-----------+--------------------------+--------------------------+
| compute248.onecloud | aio238.onecloud | compute248    | aio238.onecloud | 10.1.64.238 | confirmed | 810c523b-5239-4307-96e5-2be3b6895012 | 5          | 5
          | 2016-05-03T09:35:46.000000 | 2016-05-03T09:36:22.000000 |
+--------------------+------------------+---------------+-----------------+------------+----------+--------------------------------------+------------
+-----------+--------------------------+--------------------------+
```

Fig.3: Listing Migrated Instances

Live migration is the motion of a live instance from one compute node to another. A hugely sought-after feature by cloud administrators, it’s used essentially to achieve zero downtime during cloud maintenance and can also be a helpful feature to achieve accomplishment as live instances can be moved from a heavily loaded compute node to a less loaded compute node. During the move to another Compute Node there is only very little downtime of the virtual machine instance. Live migrations can only be handled by cloud administrators. When executing a live migration, the cloud administrator can select to which Compute Node to move the instance.

The live-migration requires the similar CPU on both hyper visors. It is possible to set a generic CPU for the virtual machine's, or a general set of CPU features. This does not work on versions lower than Kilo due to a bug where Nova diffentiate the actual CPU instead of the virtual CPU. In my case, all the hypervisor machines are the same. This is fixed in Kilo release or later releases.

On versions older than Kilo, the Compute service does not use libvirt's live migration service by default, therefore guests are suspended before migration and might experience several minutes of downtime. This is due to a risk that the migration process will never end. This can happen if the guest OS uses blocks on the disk quiker than they can be migrated. To enable true live migration using libvirt's migrate functionality refer OpenStack documents.

Multipath Requirements are when migrating an instance with multipathing configured, need to ensure consistent multipath device naming across the source and destination nodes. The migration will fail if the instance cannot solve multipath device names in the destination node. Ensure consistent multipath device naming by forcing both source node and destination node to use device WWIDs. Install the multipath packages on both aio and compute nodes and restart the multipath service using below commands

yum install device-mapper-multipath –y:

Disable user-friendly names and restart these commands on both source and destination nodes

mpathconf --enable --user_friendly_names n:

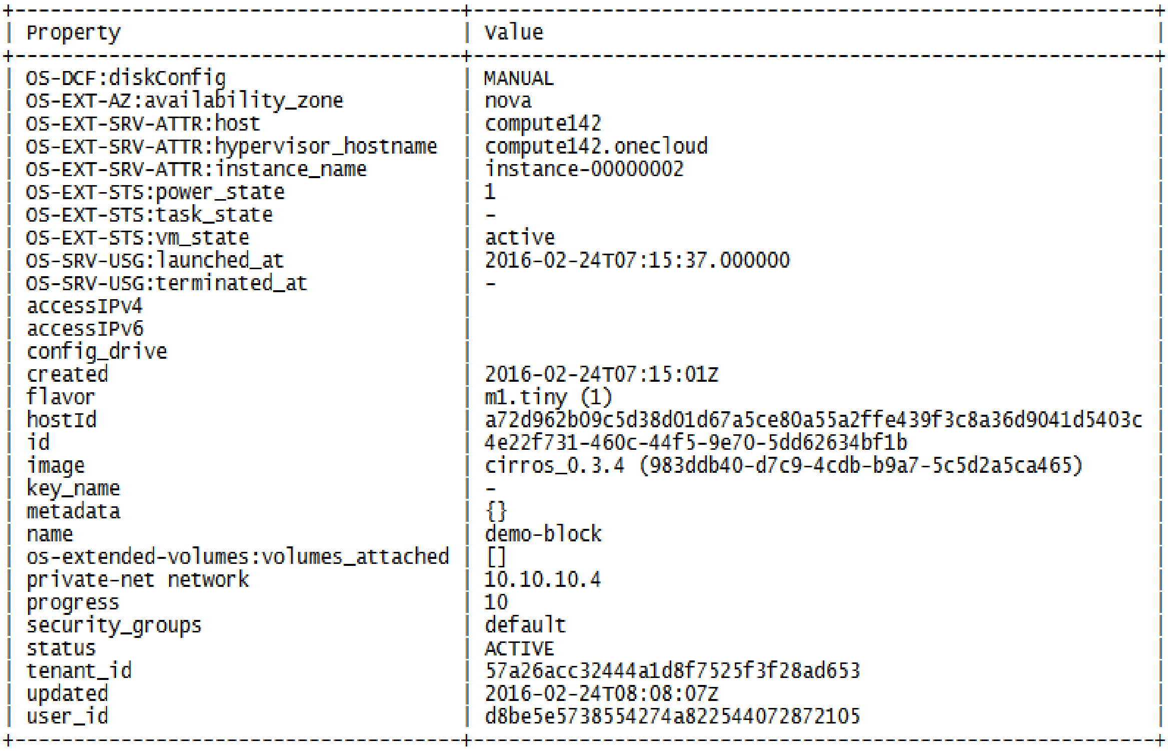
Restart the multipath service using the below command

systemctl restart multipathd:

If don't have shared storage and the VM is not backed by a volume as root disk (image based VM's) a live-migration requires an extra parameter:

nova live-migration –block-migrate $UUID $COMPUTE-HOST:

Block-based migration takes a bit longer because the entire virtual machine image is moved over from the source host to the target. However, this type of migration does not require the back-end distributed file system. Each compute node must be set up with enough disks to store the instances it hosts. Fig. 4 depicts live migrated instance details.



```
+--------------------------------------+------------------------------------------------------------------+
| Property                             | Value                                                            |
+--------------------------------------+------------------------------------------------------------------+
| OS-DCF:diskConfig                    | MANUAL                                                           |
| OS-EXT-AZ:availability_zone          | nova                                                             |
| OS-EXT-SRV-ATTR:host                 | compute142                                                       |
| OS-EXT-SRV-ATTR:hypervisor_hostname  | compute142.onecloud                                              |
| OS-EXT-SRV-ATTR:instance_name        | instance-00000002                                                |
| OS-EXT-STS:power_state               | 1                                                                |
| OS-EXT-STS:task_state                | -                                                                |
| OS-EXT-STS:vm_state                  | active                                                           |
| OS-SRV-USG:launched_at               | 2016-02-24T07:15:37.000000                                       |
| OS-SRV-USG:terminated_at             | -                                                                |
| accessIPv4                           |                                                                  |
| accessIPv6                           |                                                                  |
| config_drive                         |                                                                  |
| created                              | 2016-02-24T07:15:01Z                                             |
| flavor                               | m1.tiny (1)                                                      |
| hostId                               | a72d962b09c5d38d01d67a5ce80a55a2ffe439f3c8a36d9041d5403c         |
| id                                   | 4e22f731-460c-44f5-9e70-5dd62634bf1b                             |
| image                                | cirros_0.3.4 (983ddb40-d7c9-4cdb-b9a7-5c5d2a5ca465)              |
| key_name                             | -                                                                |
| metadata                             | {}                                                               |
| name                                 | demo-block                                                       |
| os-extended-volumes:volumes_attached | []                                                               |
| private-net network                  | 10.10.10.4                                                       |
| progress                             | 10                                                               |
| security_groups                      | default                                                          |
| status                               | ACTIVE                                                           |
| tenant_id                            | 57a26acc32444a1d8f7525f3f28ad653                                 |
| updated                              | 2016-02-24T08:08:07Z                                             |
| user_id                              | d8be5e5738554274a822544072872105                                 |
+--------------------------------------+------------------------------------------------------------------+
```

Fig.4: Live migrating an instance

IV. ADVANTAGES OF MIGRATION IN OPENSTACK

1. Migrating compute instances is very use full. It allows an administrator to free up a compute node for maintenance/updates.
2. It also allows an administrator to better distribute resources between compute nodes.
3. Helpful for load balancing and disaster recovery
4. Easy to deploy ready environments or services with few steps

V. LIMITATION OF MIGRATION IN OPENSTACK

It's essential to consider all possibilities -- both good and bad. Cloud migration issues, such as unexpected costs, interoperability, security gaps and unanticipated application rework, can create significant obstacles. To help smooth a frequently bumpy path, need to craft a well-thought-out migration strategy. Cloud migration is generally processed via WAN. It means sensitive company's data is moving through internet and it can be hacked by malicious behaviour. Customer data should be replicated to cloud with encrypted status. Sensitive company's migrating data must not be delivered to cloud via any internet storage or repository. Migrating data should be replicated directly to cloud instance because it may be hacked or lost at the Internet Storage or Repository during migration.

VI. SOLUTION TO OVERCOME OF LIMITATIONS

Encryption is staple solution in virtually every data moving through internet. There are many proposed cryptic technology for encrypting and decrypting data over internet. It aims to secure data from hacking, eavesdropping and masquerading kind of attacks on internet. This ensures confidentiality and authentication of user's sensitive data while migration.

VII. CONCLUSION

OpenStack provides many ways to migrate machines from one compute node to another. Each way is applicable in certain scenario's, and if all else fails we can manually migrate machines using the underlying linux tools. Live migration is an essential feature to achieve zero downtime during OpenStack cloud maintenance where some compute nodes need to be shut down. As described the implementing shared storage and migrating a live instance-were followed to get a working live migration on an OpenStack cloud running on Ubuntu 12.04 or CentOS 7, using NFS shared storage. Compute nodes are the workhorse of our cloud and the place where our users' applications will run. They are likely to be affected by our decisions on what to deploy and how we deploy it. Their requirements should be reflected in the choices we make.

Here are some scenarios when to use which option:

- If it is important to choose the compute host or to have very little downtime we need to use the nova live-migration command.
- If we don't want to choose the compute host, or have a config-drive enabled, we need to use the nova migrate command.
- If we need to specify the compute host and we have a config-drive enabled, we need to manually migrate the machine, or use a dirty trick to fool nova migrate.

While not a production deployment, hopefully this will help people using OpenStack for the first time, particularly with Pack Stack, get more confident with configuring networks as enough background to get started on creating something more like a real production deployment.

REFERENCES

[1] "OpenStack Docs: Liberty". releases. openstack.org. Retrieved 2016-02-20.

[2] Cloud computing service composition: A systematic literature review Amin Jula, Elankovan Sundararajan, Zalinda Othman Expert Systems with Applications 41 (2014) 3809–3824

[3] Engineering Enterprise Applications To Ensure The Highest Level Of Availability And Fault Tolerance In The Cloud", by brian jimerson, published june 2012.

[4] Stratus Technologies, "Server Virtualization and Cloud Computing: Four Hidden Impacts on Uptime and Availability," A White Paper by Stratus Technologies, June 2013.

[5] Oracle, "Architectural Strategies for Cloud Computing," An Oracle White Paper in Enterprise Architecture, August 2009.

[6] Cloud Standards Customer Council, "Security for Cloud Computing 10 Steps to Ensure Success", June 2012.

[7] https://wiki.openstack.org/wiki/Documentation/Migrate