Deploy OpenStack on Multi-Controller and Integration of Keystone with Centralized LDAP Server

25 June 2015

1705 South Main Street • Milpitas, CA 95035 • http://www.1-cloud.net

ATTENTION

The information contained in this guide is for training purpose only. This guide contains information and activities that, beneficial for purpose of training in close, non-production environment, can result in downtime or other severe consequences and therefore are not intended as a reference guide.

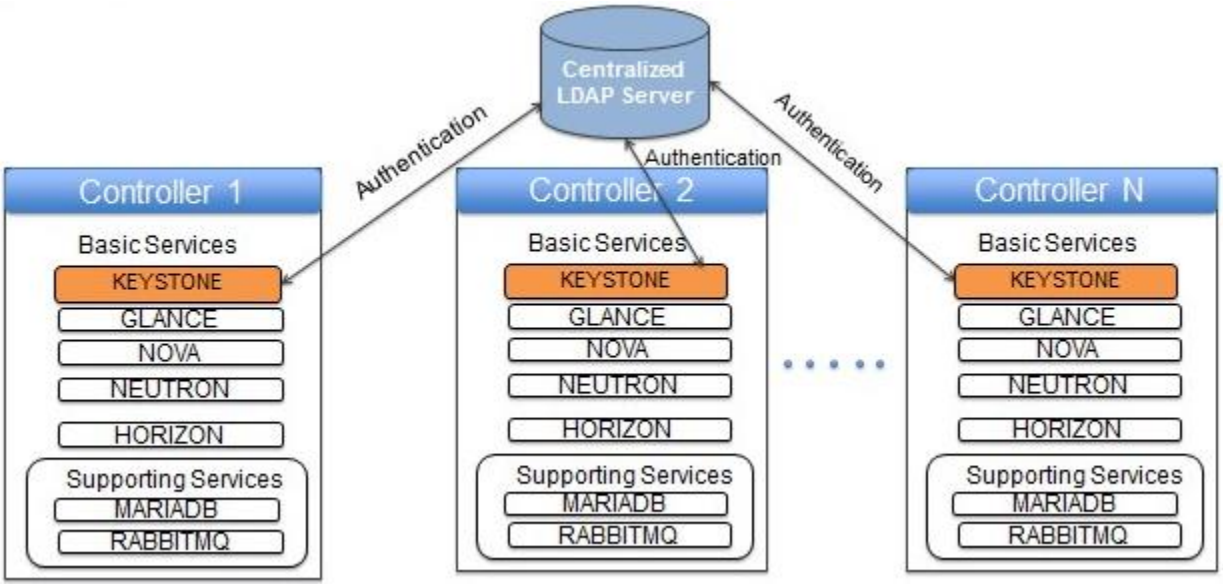
No part of this book may be reproduced in any form or by any means (graphic, electronic, mechanical including photocopying, recording or taping, or storage in an electronic retrieval system) without permission of OneCloud Consulting.

OneCloud Consulting reserves the right to change the contents herein without any prior notice.

CONTENTS

Configure OpenStack to use LDAP for identity management

Lab Topology



Objective:

This Lab exercise gives information about OpenStack deployment on multi-controllers using DevStack and understanding of OpenStack Identity service(Keystone) core concepts, including users, roles, tenants, and tokens, and working knowledge of keystone integration with Centralized LDAP and Active Directory.

Prerequisites:

- Centos 7 Operating system
- 1 or more physical machines with internet connectivity (higher the performance, better)
- In Virtualbox networking option, use 1 NAT interface and 1 host-only interface.

Introduction:

The open source OpenStack project provides an Infrastructure as a Service (IaaS) layer for building public and private clouds. Corporations, service providers, value-added

resellers, small and mid-sized businesses, researchers, and global data centers all use OpenStack to deploy large-scale private or public clouds.

Lightweight Directory Access Protocol (LDAP) is a client/server protocol for accessing and managing directory information. OpenLDAP Software is a free, open source implementation of the Lightweight Directory Access Protocol (LDAP) developed by the OpenLDAP Project.

Lightweight Directory Access Protocol (LDAP) is a solution to access centrally stored information over network. This centrally stored information is organized in a directory that follows X.500 standard. The advantage of this approach is that the information can be grouped into containers and clients can access these containers whenever needed.

This Lab describes how to configure Keystone to use a Centralized Lightweight Directory Access Protocol (LDAP) server as its back end for identity services, instead of the default SQL back end. Learn how to:

- Install an LDAP server by using DevStack, a tool for building OpenStack development environments.
- Configure Keystone to use the installed LDAP server through Keystone's LDAP identity driver.
- Validate keystone against LDAP/AD through CLI and Dashboard.

LDAP Config Files:

- config.ldif – The LDAP default configuration is stored under a file in /etc/openldap/slapd.d/cn=config.ldif that is created in the LDIF format. This is the LDAP Input Format (LDIF), a specific format that allows you to enter information in to the LDAP directory.

- olcDatabase{2}bdb.ldif – You can also modify the settings like number of connections the server can support, timeouts and other database settings under the file /etc/openldap/slapd.d/cn=config/olcDatabase{2}bdb.ldif. This is the file that also contains the parameters like LDAP root user and the base DN.

- The **slaptest -u** command to Verify the Configuration Files

Minimal Install:

You need to have a system with a fresh install of Centos 7/RedHat. You can download the Minimal CD for Centos releases since DevStack will download & install all the additional dependencies.

Basic Network Configuration:

Configure each node with a static IP. For Centos edit /etc/sysconfig/network-scripts/ifcfg-enp0s3

$ vi /etc/sysconfig/network-scripts/ifcfg-enp0s3

```
HWADDR=08:00:27:6C:FF:91
TYPE=Ethernet
BOOTPROTO=static
DEFROUTE=yes
PEERDNS=yes
PEERROUTES=yes
IPV4_FAILURE_FATAL=no
IPADDR=192.168.122.196
IPV6INIT=yes
IPV6_AUTOCONF=yes
IPV6_DEFROUTE=yes
IPV6_PEERDNS=yes
IPV6_PEERROUTES=yes
IPV6_FAILURE_FATAL=no
NAME=enp0s3
UUID=ea68db6e-461e-427d-b9a8-bfcf6e1a4fc6
ONBOOT=yes
```

Save and exit.

Now, configure default getaway:

$ vi /etc/sysconfig/network

```
NETWORKING=yes
HOSTNAME=centos7
GATEWAY=192.168.122.1
```

Configure DNS Server:

$ vi /etc/resolv.conf

```
nameserver 8.8.8.8
nameserver 8.8.4.4
```

Then set SELinux to permissive:

```
$ vi /etc/selinux/config
```

Change SELINUX=enforcing to SELINUX=permissive

Now restart your network or rebooting system.

```
$ /etc/init.d/network restart
```

Setting up an LDAP Server with DevStack

Set LDAP as the Keystone back end through the standard OpenStack development environment installation tool, DevStack. DevStack is a well-maintained and documented shell script for building complete OpenStack development environments.
(If you have installed openssh server in the machine, you can simply ssh them and install DevStack remotely).

Setting up first Controller Node:

Step 1: ssh from your local machine to the remote instance

```
$ ssh onecloud@192.168.122.196
```

Note: Run DevStack as non root user. Add your username previliges in /etc/sudoers file.

Step 2: Get an update

```
$ sudo yum update -y
```

Step 3: Install git

```
$ sudo yum install git
```

Step 4: Clone DevStack (Juno Version) from git
```
$ git clone https://github.com/openstack-dev/devstack.git
```

Step 5: Login to devstack folder

```
$ cd devstack/
```

Step 6: Configure user customizations for OpenStack in localrc. Let's make the localrc file with the below mentioned settings

```
$ vi localrc

DEST=/opt/stack

#logging

LOGFILE=$DEST/logs/stack/stack.sh.log
VERBOSE=True
LOG_COLOR=False
SCREEN_LOGDIR=$DEST/logs/screen

#credentials
ADMIN_PASSWORD=onecloud
RABBIT_PASSWORD=onecloud
MYSQL_PASSWORD=onecloud
SERVICE_PASSWORD=onecloud
SERVICE_TOKEN=onecloud

HOST_IP=192.168.122.196

#services
disable_service n-net
enable_service q-svc
enable_service q-agt
enable_service q-l3
enable_service q-dhcp
enable_service q-meta
enable_service neutron
disable_service heat h-api h-api-cfn h-api-cw h-eng

#Enable DevStack to install an LDAP server
enable_service ldap

#Inform DevStack that you want Keystone to use its LDAP back-end identity driver
KEYSTONE_IDENTITY_BACKEND=ldap

#If you want DevStack to clear out an existing Keystone LDAP tree and start fresh
KEYSTONE_CLEAR_LDAP=yes
```

#Set LDAP Password
LDAP_PASSWORD=onecloud

#slappass command to create a root password you want to use
#SLAPPASS=onecloud

Save and close localrc.

Note: HOST_IP is the static ip of Controller node. Once it is done, you can stack the controller node.

Step 8: Now run the stack.sh script from the devstack root directory:

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

The output looks like:

```
Horizon is now available at http://192.168.122.196/
Keystone is serving at http://192.168.122.196:5000/v2.0/
Examples on using novaclient command line is in exercise.sh
The default users are: admin and demo
The password: onecloud
This is your host ip: 192.168.122.196
```

Open your browser to see Horizon by **<https://Host_IP>**

After the script finishes, you can see that:

- OpenLDAP was installed.
- Keystone was configured to use its LDAP back-end identity driver.
- An initial Keystone LDAP tree was created that uses the data in devstack\files\ldap\keystone.ldif.in

Contents of keystone.ldif.in

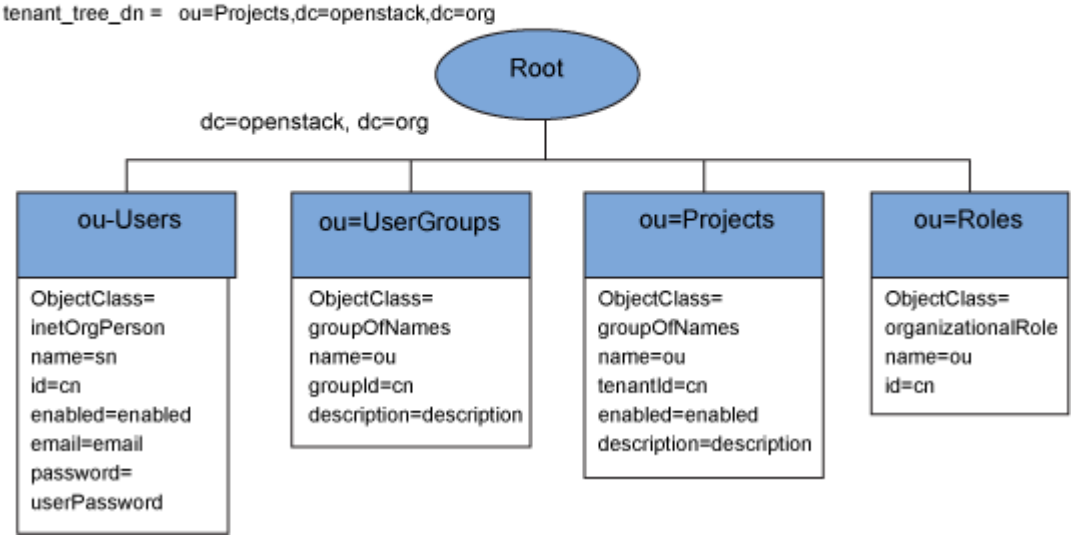
```
dn: ${BASE_DN}
objectClass: dcObject
objectClass: organizationalUnit
dc: ${BASE_DC}
ou: ${BASE_DC}

dn: ou=UserGroups,${BASE_DN}
objectClass: organizationalUnit
ou: UserGroups

dn: ou=Users,${BASE_DN}
objectClass: organizationalUnit
ou: Users

dn: ou=Roles,${BASE_DN}
objectClass: organizationalUnit
ou: Roles

dn: ou=Projects,${BASE_DN}
objectClass: organizationalUnit
ou: Projects

dn: cn=9fe2ff9ee4384b1894a90878d3e92bab,ou=Roles,${BASE_DN}
objectClass: organizationalRole
ou: _member_
cn: 9fe2ff9ee4384b1894a90878d3e92bab
```

Note: The OpenLDAP hierarchy is almost similar to the DNS hierarchy. The following are the two most commonly used objects in OpenLDAP:

- cn (common name) – This refers to the leaf entries, which are end objects (for example: users and groups)
- dc (domain component) – This refers to one of the container entries in the LDAP hierarchy. If in a setup the LDAP hierarchy is mapped to a DNS hierarchy, typically all DNS domains are referred to as DC objects.

Example schema used by the Keystone LDAP back-end identity driver:

suffix =	dc=openstack, dc=org
user_tree_dn =	ou=Users,dc=openstack,dc=org
group_tree_dn =	ou=UserGroups,dc=openstack,dc=org
role_tree_dn =	ou=Roles,dc=openstack,dc=org
tenant_tree_dn =	ou=Projects,dc=openstack,dc=org



In the above example LDAP tree, Users, UserGroups, Projects, and Roles each is its own subtree that uses a standard LDAP ObjectClass. In the Users subtree, for example, ObjectClass=inetOrgPerson.

Step 9: Verify the LDAP Search

To verify the ldap server is configured successfully, you can use the below command and verify that the domain entry is present.

```
$ ldapsearch -x -b 'dc=openstack,dc=org' '(objectclass=*)'
```

```
# extended LDIF
#
# LDAPv3
# base <dc=openstack,dc=org> with scope subtree
# filter: (objectclass=*)
# requesting: ALL
#

# openstack.org
```

```
dn: dc=openstack,dc=org
objectClass: dcObject
objectClass: organizationalUnit
dc: openstack
ou: openstack

# UserGroups, openstack.org
dn: ou=UserGroups,dc=openstack,dc=org
objectClass: organizationalUnit
ou: UserGroups

# Users, openstack.org
dn: ou=Users,dc=openstack,dc=org
objectClass: organizationalUnit
ou: Users

# Roles, openstack.org
dn: ou=Roles,dc=openstack,dc=org
objectClass: organizationalUnit
ou: Roles

# Projects, openstack.org
dn: ou=Projects,dc=openstack,dc=org
objectClass: organizationalUnit
ou: Projects

# 9fe2ff9ee4384b1894a90878d3e92bab, Roles, openstack.org
dn: cn=9fe2ff9ee4384b1894a90878d3e92bab,ou=Roles,dc=openstack,dc=org
objectClass: organizationalRole
ou: _member_
cn: 9fe2ff9ee4384b1894a90878d3e92bab

# 2b38d01115004911b11c91c0a903ea3d, Users, openstack.org
dn: cn=2b38d01115004911b11c91c0a903ea3d,ou=Users,dc=openstack,dc=org
objectClass: person
objectClass: inetOrgPerson
userPassword:: b25lY2xvdWQ=
cn: 2b38d01115004911b11c91c0a903ea3d
sn: admin

# 739e142b0d0b4ac98cb870fe841448b5, Users, openstack.org
dn: cn=739e142b0d0b4ac98cb870fe841448b5,ou=Users,dc=openstack,dc=org
objectClass: person
```

```
objectClass: inetOrgPerson
userPassword:: b25lY2xvdWQ=
cn: 739e142b0d0b4ac98cb870fe841448b5
sn: demo

# 93e759b126af4a838046aa39138047a4, UserGroups, openstack.org
dn: cn=93e759b126af4a838046aa39138047a4,ou=UserGroups,dc=openstack,dc=org
objectClass: groupOfNames
description: openstack admin group
ou: admins
cn: 93e759b126af4a838046aa39138047a4
member: cn=dumb,dc=nonexistent

# 590138ec585a408baa67247e77e65938, UserGroups, openstack.org
dn: cn=590138ec585a408baa67247e77e65938,ou=UserGroups,dc=openstack,dc=org
objectClass: groupOfNames
description: non-admin group
ou: nonadmins
cn: 590138ec585a408baa67247e77e65938
member: cn=dumb,dc=nonexistent

# 20da445f6a174af9a23111b2df491c96, Users, openstack.org
dn: cn=20da445f6a174af9a23111b2df491c96,ou=Users,dc=openstack,dc=org
objectClass: person
objectClass: inetOrgPerson
userPassword:: b25lY2xvdWQ=
cn: 20da445f6a174af9a23111b2df491c96
sn: nova

# fe9725d070bc4fde9a2d52f15c00ee70, Users, openstack.org
dn: cn=fe9725d070bc4fde9a2d52f15c00ee70,ou=Users,dc=openstack,dc=org
objectClass: person
objectClass: inetOrgPerson
userPassword:: b25lY2xvdWQ=
cn: fe9725d070bc4fde9a2d52f15c00ee70
sn: glance

# 67d78c9ee5964463abb5edd818416ebf, Users, openstack.org
dn: cn=67d78c9ee5964463abb5edd818416ebf,ou=Users,dc=openstack,dc=org
objectClass: person
objectClass: inetOrgPerson
userPassword:: b25lY2xvdWQ=
cn: 67d78c9ee5964463abb5edd818416ebf
```

```
sn: cinder

# b2db38d689fa44aab4c4e171169248be, Users, openstack.org
dn: cn=b2db38d689fa44aab4c4e171169248be,ou=Users,dc=openstack,dc=org
objectClass: person
objectClass: inetOrgPerson
userPassword:: b25lY2xvdWQ=
cn: b2db38d689fa44aab4c4e171169248be
sn: neutron

# 5f8efc84be1f4d75975fc95e07bd1449, Users, openstack.org
dn: cn=5f8efc84be1f4d75975fc95e07bd1449,ou=Users,dc=openstack,dc=org
objectClass: person
objectClass: inetOrgPerson
userPassword:: b25lY2xvdWQ=
cn: 5f8efc84be1f4d75975fc95e07bd1449
sn: alt_demo

# search result
search: 2
result: 0 Success

# numResponses: 16
# numEntries: 15
```

Validating keystone credentials against LDAP/AD

Once we successfully run the ./stack.sh, Source your keystone credentials to LDAP.

```
$ source openrc admin admin
```

If the LDAP mappings are correct in keystone.conf, the **user-list** command should show the list of users in the LDAP database.

Check Keystone user-list.

```
$ keystone user-list
```

```
+----------------------------------+---------+---------+-------+
|                id                |   name  | enabled | email |
+----------------------------------+---------+---------+-------+
| c7d3ad78f3f841e5be190b01ce12f5b2 |  admin  |         |       |
```

```
| 436ca551705f480ebf30422d71cb986d |  cinder |         |       |
| 427c6741e2194b2db79edbcee1dc4768 |  glance |         |       |
| d9d5595776fc4840b95264639906105d | neutron |         |       |
| 6ac2f7a16ff249c3bfa5b3d4b9b95037 |   nova  |         |       |
+----------------------------------+---------+---------+-------+
```

Check Keystone tenant-list.

```
$ keystone tenant-list
```

```
+----------------------------------+---------+---------+
|                id                |   name  | enabled |
+----------------------------------+---------+---------+
| 07bfaa36253644c9960bf6ab29d64e8a |  admin  |   True  |
| 7f2552bc1acb4a489ad44f70557b2786 | service |   True  |
+----------------------------------+---------+---------+
```

Check Keystone role-list.

```
$ keystone role-list
```

```
+----------------------------------+---------+
|                id                |   name  |
+----------------------------------+---------+
| f2d77a5a663041739d1a4e3bed876c2d |  admin  |
| acf7fdf7df4e4bdea50afde8505528de | service |
+----------------------------------+---------+
```

NOTE:

If you want to interact with the services, just remember that DevStack doesn't use any script init (upstart or service don't exist). It simply runs the services in a standalone mode (foreground running daemon). To bring up all the services DevStack uses a big parent screen where it encapsulates child screens.

This how to access them:

```
./rejoin-stack.sh or screen -r
```

To navigate to child screens use the command:
Ctrl + a + "

Select one and press enter to get into one child screen. If you want to change the behavior of a daemon, let's say **nova-api**, just modify your **nova.conf** then kill the process in the child **n-api** with **ctrl + c**, re-run it.

Finally detach the screen by using the command.

```
ctrl + a + d
```

Creating Users, Tenants, Roles on LDAP Server through Keystone

Step 1: Lets verify that the existing domain entry is present in LDAP Server using the below command.

```
$ ldapsearch -x -b 'dc=openstack,dc=org' '(objectclass=*)'
```

Note: This command will display all the records stored in LDAP/Active Directory currently.

Step 2: Creating Openstack User

```
$ keystone user-create --name onecloud --email onecloud@demo.com
```

```
+----------+----------------------------------+
| Property |              Value               |
+----------+----------------------------------+
|  email   |        onecloud@demo.com         |
| enabled  |               True               |
|    id    | fb0f19a89f1a4beab1b4246e4e3d7edd |
|   name   |             onecloud             |
| username |             onecloud             |
+----------+----------------------------------+
```

Step 3: Updating Password for the new user.

```
$ keystone user-password-update onecloud
```

```
New Password: *********
Repeat New Password: *********
```

Step 4: Check the newly created Openstack User.

```
$ keystone user-list
```

```
+----------------------------------+----------+---------+-------+
|                id                |   name   | enabled | email |
+----------------------------------+----------+---------+-------+
| c7d3ad78f3f841e5be190b01ce12f5b2 |  admin   |         |       |
```

```
| 436ca551705f480ebf30422d71cb986d |  cinder  |          |        |
| 427c6741e2194b2db79edbcee1dc4768 |  glance  |          |        |
| d9d5595776fc4840b95264639906105d | neutron  |          |        |
| 6ac2f7a16ff249c3bfa5b3d4b9b95037 |   nova   |          |        |
| fb0f19a89f1a4beab1b4246e4e3d7edd | onecloud |          |        |
+----------------------------------+----------+----------+-------+
```

Step 5: Creating New Openstack Tenant.

```
$ keystone tenant-create --name onecloud --description "OneCloud Demo Tenant"
```

```
+-------------+----------------------------------+
|   Property  |               Value              |
+-------------+----------------------------------+
| description |        OneCloud Demo Tenant      |
|   enabled   |               True               |
|      id     | f13c97bc5e3c4f01a8c4e17ae7203645 |
|     name    |             onecloud             |
|  parent_id  |                                  |
+-------------+----------------------------------+
```

Step 6: Check the newly created Openstack Tenant

```
$ keystone tenant-list
```

```
+----------------------------------+----------+---------+
|                id                |   name   | enabled |
+----------------------------------+----------+---------+
| 07bfaa36253644c9960bf6ab29d64e8a |  admin   |   True  |
| f13c97bc5e3c4f01a8c4e17ae7203645 | onecloud |   True  |
| 7f2552bc1acb4a489ad44f70557b2786 | service  |   True  |
+----------------------------------+----------+---------+
```

Step 7: Lastly, grant the admin role to the OpenStack Administrator account in the OneCloud Demo Tenant.

```
$ keystone user-role-add --user-id onecloud --tenant-id onecloud --role-id admin
```

Step 8: Now check the newly created Openstack User in LDAP Server.

```
$ ldapsearch -x -b 'dc=openstack,dc=org' '(sn=onecloud)'
```

```
# extended LDIF
```

```
#
# LDAPv3
# base <dc=openstack,dc=org> with scope subtree
# filter: (sn=onecloud)
# requesting: ALL
#

# fb0f19a89f1a4beab1b4246e4e3d7edd, Users, openstack.org
dn:
cn=fb0f19a89f1a4beab1b4246e4e3d7edd,ou=Users,dc=openstack,dc=org
objectClass: person
objectClass: inetOrgPerson
cn: fb0f19a89f1a4beab1b4246e4e3d7edd
sn: onecloud
userPassword:: b25lY2xvdWQxMjM=

# search result
search: 2
result: 0 Success

# numResponses: 2
# numEntries: 1
```

Step 9: Validating LDAP User by sourcing Keystone credentials.

Create keystonerc_onecloud file.

```
$ vim keystonerc_onecloud
```

```
export OS_USERNAME=onecloud
export OS_TENANT_NAME=onecloud
export OS_PASSWORD=onecloud
export OS_AUTH_URL=http://192.168.122.196:5000/v2.0/
export PS1='[\u@\h \W(keystonerc_onecloud)]\$ '
```

Source your keystone credentials to LDAP.

```
$ source keystonerc_onecloud
```

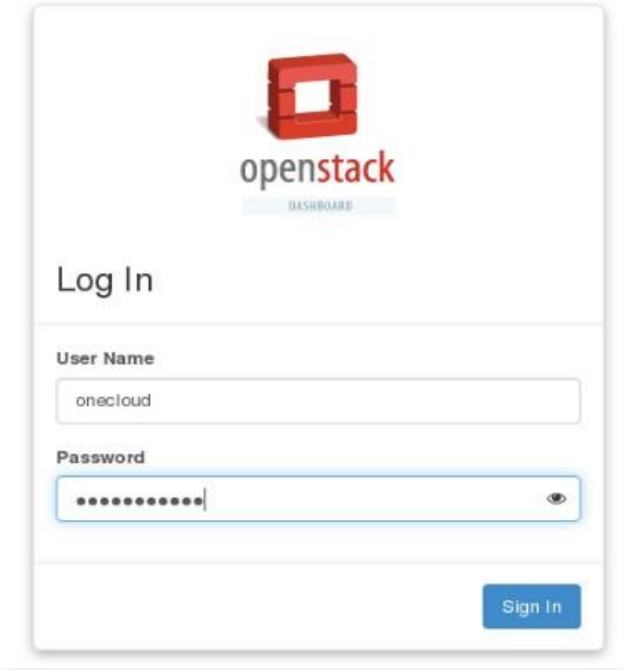
```
$ keystone token-get
```

```
+-----------+----------------------------------+
|  Property |               Value              |
+-----------+----------------------------------+
```

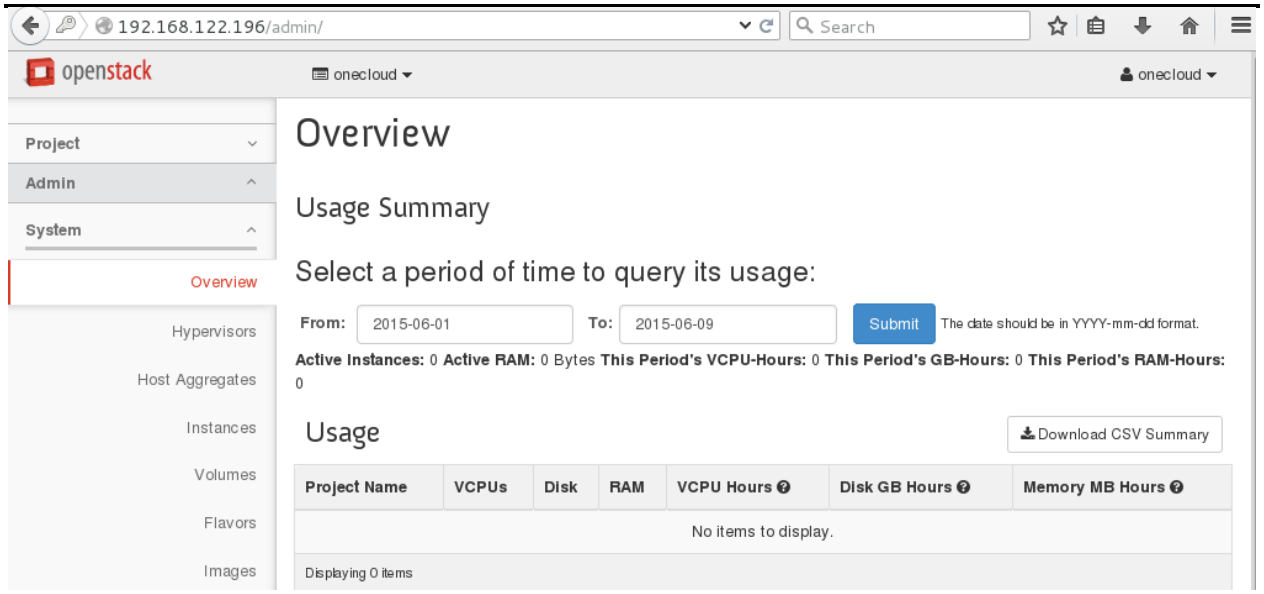
```
|  expires   |        2015-02-26T12:39:59Z        |
|     id     | a1b07e196eb543f19d4b20835a391fd5 |
| tenant_id | f13c97bc5e3c4f01a8c4e17ae7203645 |
|  user_id   | fb0f19a89f1a4beab1b4246e4e3d7edd |
+-----------+----------------------------------+
```

Step 10: Validating LDAP User through Horizon.

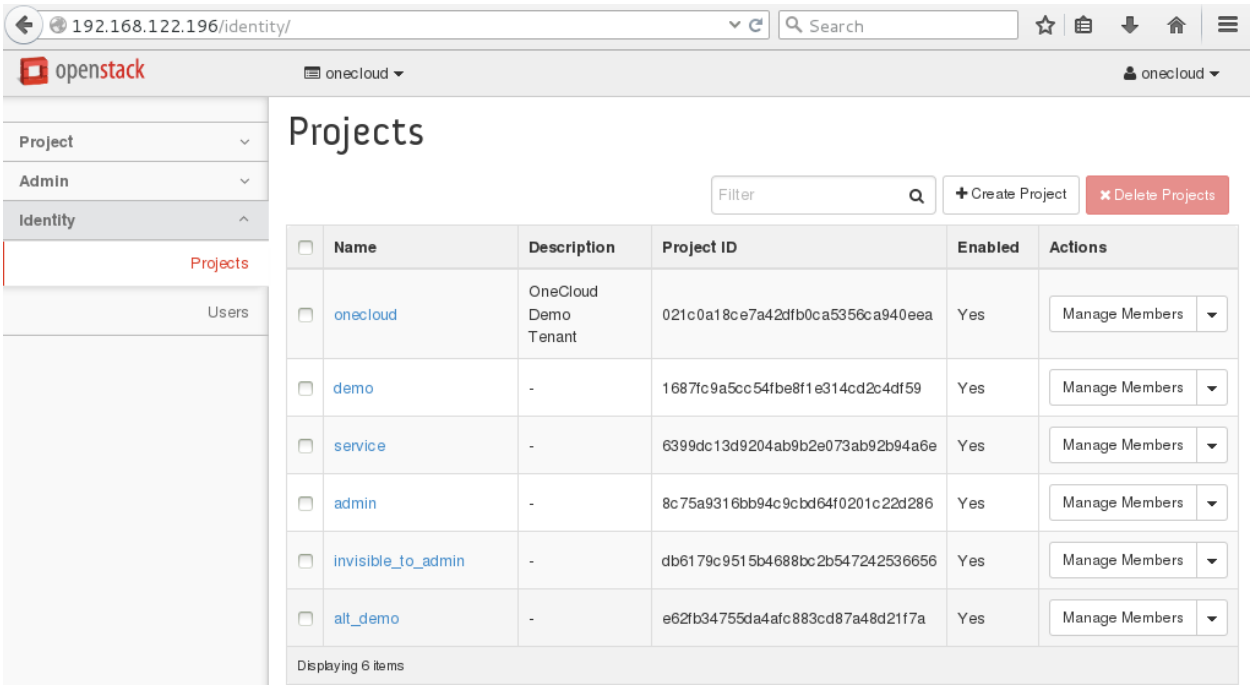
Open your browser to see Horizon by https://Host_IP and Login through your newly created Username and Password.

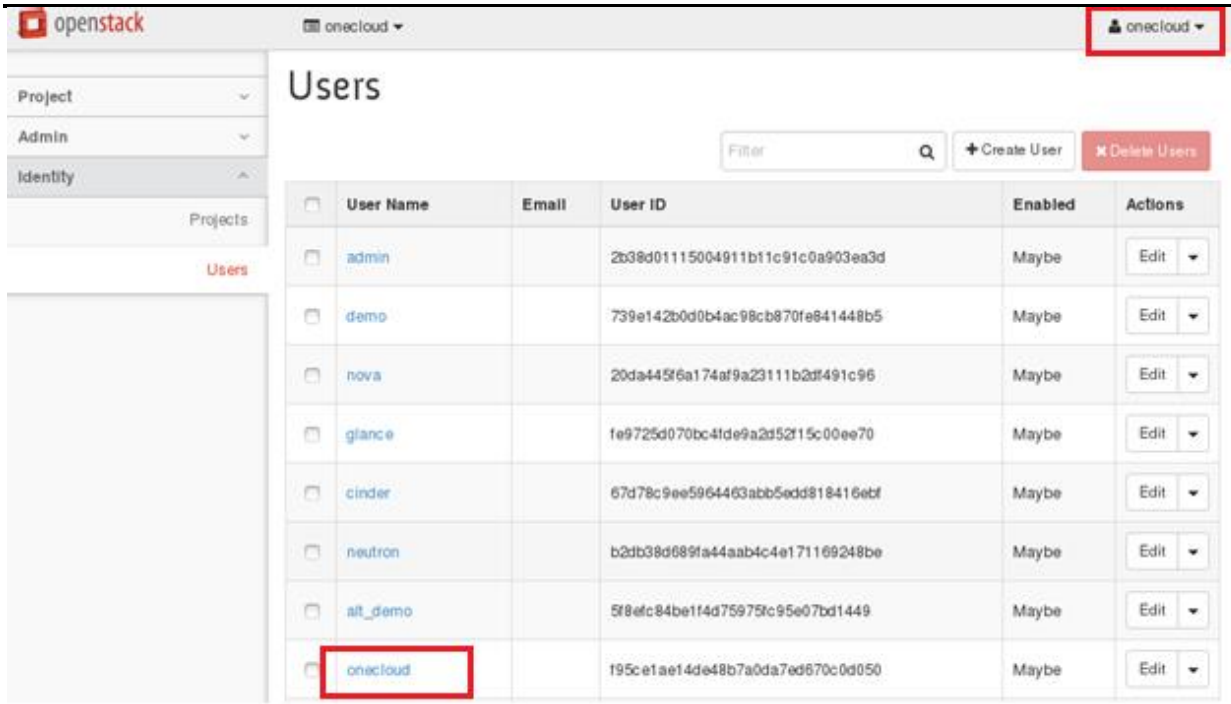


After Login, You can see the separate Dashboard for newly created Tenant/Project with the respected User Role.



In Identity section, check projects and users tab to see the newly added project and user in the database.





Setting up second Controller Node:

Follow the same steps in setting up Controller node with proper static IP mentioned above in the document.

Step 1: Let's make the local.conf file with the below mentioned settings on second Controller Node.

```
$ vi local.conf
```

```
DEST=/opt/stack

#logging

LOGFILE=$DEST/logs/stack/stack.sh.log
VERBOSE=True
LOG_COLOR=False
SCREEN_LOGDIR=$DEST/logs/screen

#credentials
```

```
ADMIN_PASSWORD=onecloud
RABBIT_PASSWORD=onecloud
MYSQL_PASSWORD=onecloud
SERVICE_PASSWORD=onecloud
SERVICE_TOKEN=onecloud

HOST_IP=192.168.122.148

#services
disable_service n-net
enable_service q-svc
enable_service q-agt
enable_service q-l3
enable_service q-dhcp
enable_service q-meta
enable_service neutron
disable_service heat h-api h-api-cfn h-api-cw h-eng
```

Step 2: Start DevStack

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

The output looks like:

```
Horizon is now available at http://192.168.122.148/
Keystone is serving at http://192.168.122.148:5000/v2.0/
Examples on using novaclient command line is in exercise.sh
The default users are: admin and demo
The password: onecloud
This is your host ip: 192.168.122.148
```

Open your browser to see Horizon by **<https://Host_IP>**

Step 3: Identity Service supports integration with an Centralized LDAP directory for authentication and authorization services. And to check how the Keystone is validating the entries stored in LDAP/AD.

To ensure that, check in the keystone.conf file in the [identity] section,

replaced with driver = ldap
instead of driver = sql

Then search for [ldap] section and add the below configuration as per your domain.

```
url = ldap://ldap_server_IP
user = cn=Manager,dc=openstack,dc=org
password = onecloud
suffix = cn=openstack,cn=org
use_dumb_member = True
tree_dn = dc=openstack,dc=org

user_tree_dn = ou=Users,dc=openstack,dc=org
user_objectclass = inetOrgPerson
user_id_attribute = cn
user_name_attribute = sn
user_pass_attribute = userPassword
```

```
user_allow_create = True
user_allow_update = True
user_enabled_attribute = enabled
user_enabled_default = True
user_domain_id_attribute = None

#tenant_tree_dn = ou=Tenants,dc=example,dc=com
tenant_tree_dn = ou=Projects,dc=openstack,dc=org
#tenant_tree_dn = ou=Tenants,dc=openstack,dc=org
tenant_objectclass = groupOfNames
tenant_id_attribute = cn
tenant_member_attribute = member
tenant_name_attribute = ou
tenant_domain_id_attribute = None
tenant_allow_create = True
tenant_allow_update = True


role_tree_dn = ou=Roles,dc=openstack,dc=org
role_objectclass = groupOfNames
role_member_attribute = member
role_id_attribute = cn
role_name_attribute = ou
role_allow_create = True
role_allow_update = True
```

Step 4: Restart keystone service

./rejoin-stack.sh or **screen -r**

To navigate to child screens use the command:

Ctrl + a + ”

Select key option and press enter to get into one child screen kill the process with ctrl + c, re-run it.
Now the keystone is successfully integrated with centralized LDAP server.

Validating keystone credentials against LDAP/AD

Source your keystone credentials to LDAP.

```
$ source openrc admin admin
```

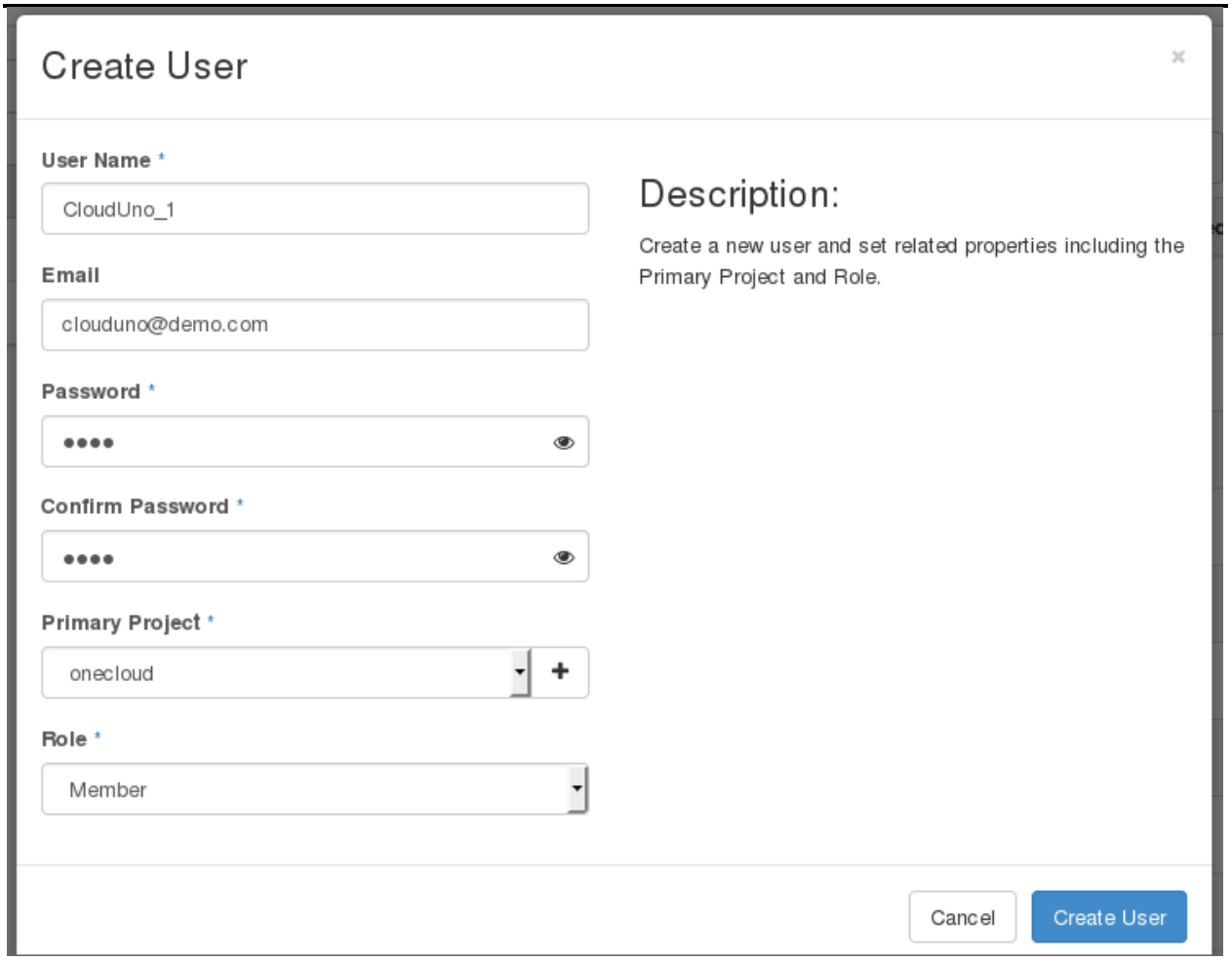
Check Keystone user-list.

```
$ keystone user-list
```

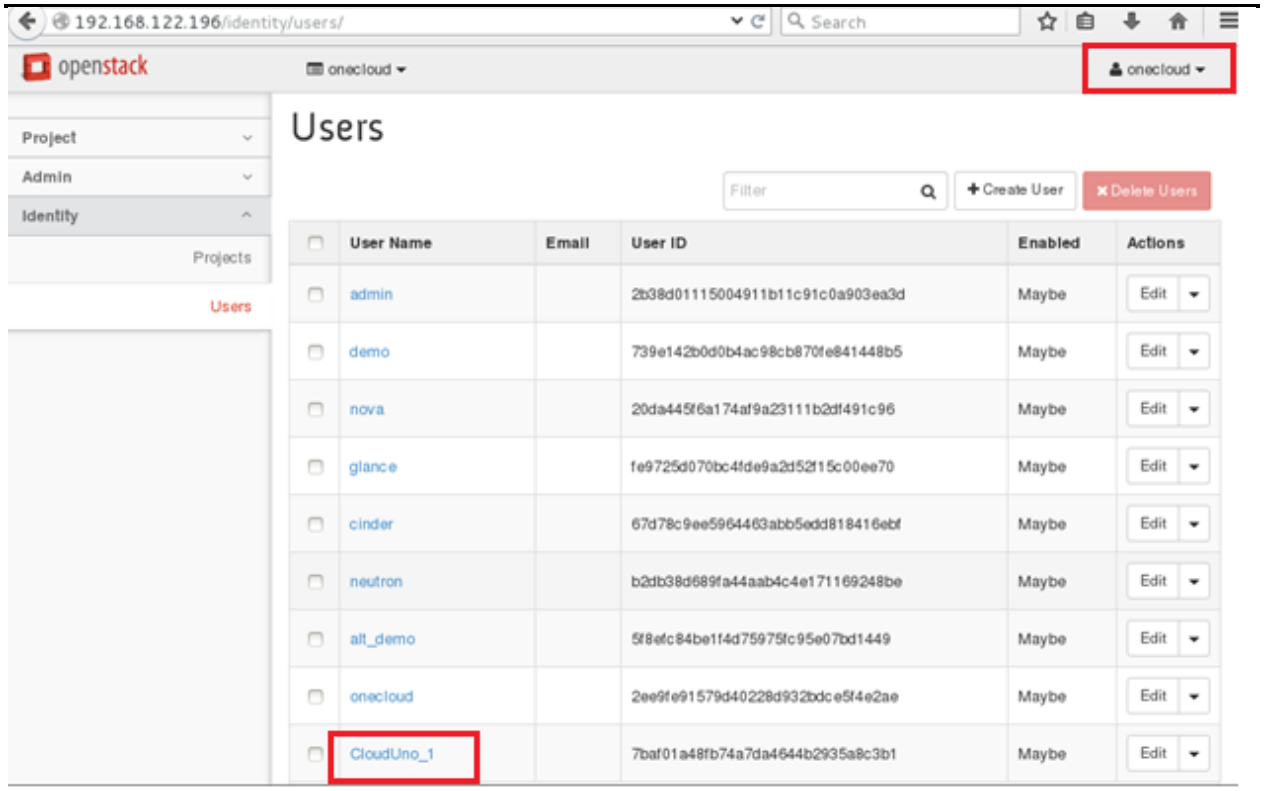
```
+----------------------------------+----------+---------+-------+
|                id                |   name   | enabled | email |
+----------------------------------+----------+---------+-------+
| 9e1651c5c3bd474d86a77a8c7a78cf7c |  admin   |         |       |
| aae8e5b5ffc844ff814bde4d6c31feea |  alt_demo|         |       |
| a2c9b7ca72b841eea43139084a9b2e07 | cinder   |         |       |
| 896758a5ce9044579ba17d04418c7efd |   demo   |         |       |
| a4a596b7d62d499fb1894c9f34084a34 |  glance  |         |       |
| 0e87c782eea54ae0893d5ae03b84c054 | neutron  |         |       |
| 8165ca189d1a47ba83d400a7831c7dd5 |   nova   |         |       |
| fb0f19a89f1a4beab1b4246e4e3d7edd | onecloud |         |       |
+----------------------------------+----------+---------+-------+
```

And login to the dashboard on second controller with the Openstack user credentials (onecloud) which exists in first controller.

Once login successfully it ensure that, keystone authenticated a user and can access the data from LDAP Server, creating another Openstack User to the existing onecloud tenant with member role through Dashboard.



In Identity section, check users tab to see the newly added OpenStack user in the database.



Now check the newly created Openstack User in LDAP Server.

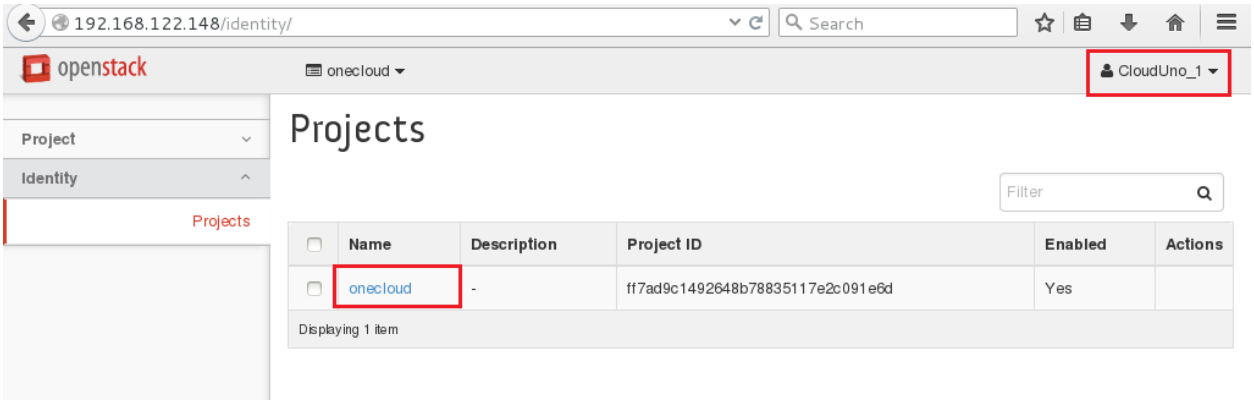
```
$ ldapsearch -x -b 'dc=openstack,dc=org' '(sn=CloudUno_1)'
```

```
# extended LDIF
#
# LDAPv3
# base <dc=openstack,dc=org> with scope subtree
# filter: (sn= CloudUno_1)
# requesting: ALL
#

# f996f045765b487593cab00d6de5fe8c, Users, openstack.org
dn:
cn=f996f045765b487593cab00d6de5fe8c,ou=Users,dc=openstack,dc=org
objectClass: person
objectClass: inetOrgPerson
sn: CloudUno_1
userPassword:: cGFzcw==
cn: f996f045765b487593cab00d6de5fe8c
```

```
# search result
search: 2
result: 0 Success

# numResponses: 2
# numEntries: 1
```

Validating LDAP User through Horizon.

Open your browser to see Horizon by https://Host_IP and Login through your newly created Username and Password.



That's it☺