**KEYSTONE (Identity Management)**

Lets take an example, a new employee named ‘Nilesh’ is selected in an IT Company named ‘Tecknix’. On his first day, before he enters into his cabin, he was advised to take the ‘electronic access card’ from the ‘access control team’ of the company.

The ‘access control team’ issues him an ‘electronic access card’. This access card will have a unique ID for each user. The card should be used as an identity (authentication) card and also has authorization information based on the role mapped for the user profile.

For example, Mr. Nilesh could access the server room or else his access to the server room will be denied. The ‘access control team’ will take care of mapping the selected role to each and every employee based on his designation.

The Keystone service does the same thing as the ‘access control team’. The KeyStone service will take care of providing the unique token ID for each user and also assign the selected roles (admin or member) to the user. The provided token ID will be used for authenticating, and the authorization of end users accessing the dashboard based on the role mapped.

Usually Keystone is the first component to be installed when starting an OpenStack cloud. It is responsible for the authentication and verification between all of the OpenStack cloud services. Keystone supports multiple forms of authentication, including login name and password, token-based credentials, and REST API log ins.

The OpenStack Identity service authenticates users and projects by sending a validated authorization token between all OpenStack services. This token is passed to the other services, such as Storage and Compute, to grant user access to specific functionalities. Therefore, configuration of the OpenStack Identity service must be completed first before using any of the other services. Setting up of the Identity service involves the creation of appropriate roles for users and services, projects, the user accounts, and the service API endpoints that make up our cloud infrastructure.

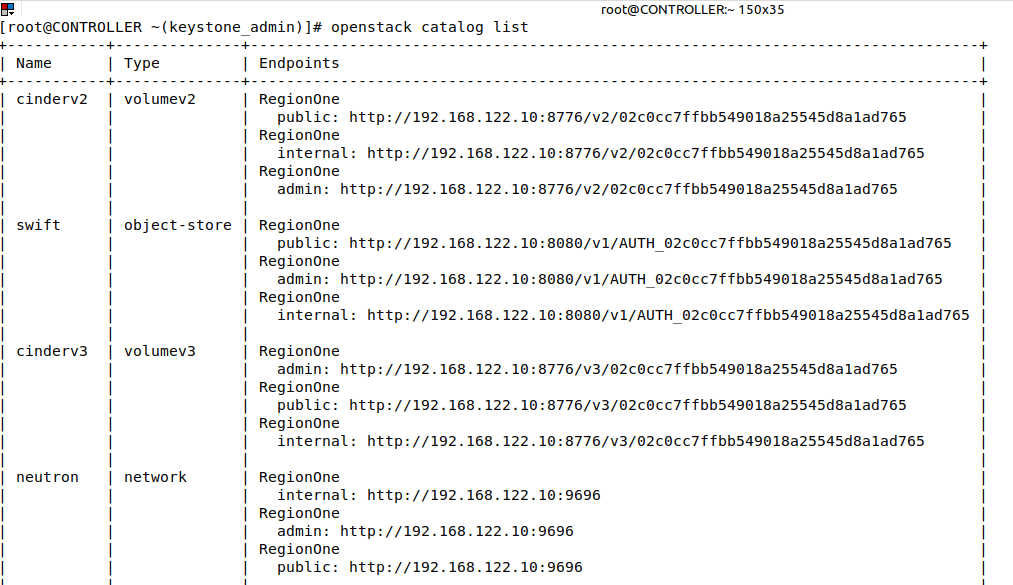
**NOTE:-**

**Authentication** is the act of confirming the identity of a specific user—in other words, proving that a user is whom she or he claims to be.

**Authorization** is the function of determining access rights for that specific user.

**Identity concepts in Keystone**

• **Service** : All the openstack services/components which uses the keystone service for their authentication and authorization. These OpenStack cloud services can be listed by using **openstack catalog list** command. Service provides one or more endpoints through which users can access service’s API.





• **Endpoint/Service Catalog** : The Service Catalog is a map stored in

Keystone that contains a list of services and endpoints. It provides a listing of

API endpoints for all services installed in the environment i.e. URL from which the service is available. There are three URLs provided for each service in the catalog: internal, public, and administration. They can have different subsets of API calls. Endpoint can look like https://controller.my-domain.com:35357/v2.0 . At this URL you would find

that the service is listening to incoming calls on port number 35357 and the API version is

2.0. Common port numbers for OpenStack services are shown:-

*Common Port Numbers for OpenStack Services*

**Network Port Number OpenStack Service**

5000 Public API endpoint port for Keystone

35357 Admin API endpoint port for Keystone

8776 Cinder Block Storage service

9292 Image service Glance

9191 Glance Registry

8774 Compute service Nova

8080 and 6001-6003 Object Storage services Swift

9696 Networking service Neutron

8777 Telemetry service Ceilometer

1. Orchestration service Heat

* **Domains:** A Keystone domain is a high level OpenStack Identity resource that contains projects, users, and groups.
* **Projects:** A project represents an area where resources get created. A project has resources such as users, images, and instances, as well as networks in it that can be restricted only to that particular project, unless explicitly shared with others. A user could certainly belong to one or more projects, but they can only be authenticated to one project at one time. The concept of authenticating oneself to a single project is called **scope**. Any resource an authenticated user creates is always owned by the project to which the user is currently scoped.

**Domain 1**

**Softnet Securities**

**Project 1: HR Department**

**Project 2 : Technical**

**Department**

**Project 3 : Account**

**Department**

**Project 4 : Sales**

**Department**

**Domain 3**

**OpenCloud**

**Domain 2**

**Teknix**

**Project 1: HR Department**

**Project 1: HR Department**

**Project 2 : Technical**

**Department**

**Project 2 : Technical**

**Department**

**Project 3 : Account**

**Department**

**Project 3 : Account**

**Department**

**Project 4 : Sales**

**Department**

**Project 4 : Sales**

**Department**

Domains are high-level logical containers used for users, groups, and projects. Projects live

inside domains and represent an area where virtual resources get created

**Users:** A user can belong to one or more projects and is able to switch between them to gain access to those resources. Users within a project can have various roles assigned. Users can be organized into user groups and the groups can have roles assigned to them.

**Domain 1**

**Softnet Securities**

**Project 1: HR Department**

**Project 2 : Technical**

**Department**

**User: Nilesh**

**Role: admin**

**Project 4 : Sales**

**Department**

**Domain 1**

**Softnet Securities**

**Project 1: HR Department**

**Project 2 : Technical**

**Department**

**Group: ADM Role: admin**

**Project 4 : Sales**

**Department**

**Group: TST**

**Role: \_member\_**

Groups are an optional container for users. They are an easy way to apply an existing role to

multiple users at one time

**Groups:** Groups are a newer Keystone concept that provide an easy way to apply an existing role to multiple users. They are optional and were created to eliminate the hassle of applying roles to one user at a time.

**Roles:** Roles are labels that are applied directly to a user or a group. A role is just a label associated with a user on a project. A file called policy.json handles authorization by determining what a user with a specific role can or cannot do. In the most basic scenario, a user can be assigned either the role of **admin** or just be a **\_member\_**. When a user has admin privileges within a project, the admin is able to utilize features that can affect the project (such as modifying external networks), whereas a normal user is assigned the member role. This member role is generally assigned to perform user-related roles, such as spinning up instances, creating volumes, and creating isolated, project-specific networks.

**Quotas**: Quotas determine the quantity or size of resources that can be created within a specific project. They act as guardrails by preventing users from exceeding capacity.

**Region** : Separates the OpenStack environment with dedicated API endpoints but with common Keystone service.

**Token** : Issued by Keystone service then passed to API requests and used by OpenStack to verify that the client is authorized to run the requested operation. The token is issued for a limited time and, if necessary, may be withdrawn prior to the expiration. In order to get the user token, the user must either provide a name and password, or the name and the key to access the API (API key). The token also contains a list of roles that defines the roles

available to the user.

The Keystone main configuration file /etc/keystone/keystone.conf :-

|  |  |
| --- | --- |
| **Example of Config Options** | **Description** |
| [DEFAULT]  admin\_token = ee224e8... | A “shared secret” that can be used to bootstrap and debug Keystone. This “token” does not represent a user. |
| [DEFAULT]  debug = True | Set logging level to DEBUG instead of default INFO level in journal. |
| [DEFAULT]  log\_dir = /var/log/keystone | The base directory used for log files. |
| [DEFAULT]  public\_port=5000  admin\_port=35357  admin\_bind\_host=0.0.0.0  public\_bind\_host=0.0.0.0 | The port numbers and IP address which the public and admin services listen on. In \*\_bind\_host options, four zeros mean all available ports on the server. |
| [database]  connection = mysql://keystone\_  admin:password@192.168.122.15/keystone | The SQLAlchemy connection string is used to connect to the database. |
| [oslo\_messaging\_rabbit]  rabbit\_host = localhost  rabbit\_port = 5672  rabbit\_userid = guest  rabbit\_password = guest | The RabbitMQ broker address, port, user name, and password. |
| [token]  expiration = 3600 | Token validity timeframe (in seconds). By default–1 hour. |

Before initiating something with OpenStack, you need to go through the authorization and authentication processes. You can use the CLI commands options, but it is better and easier to create a file with global variables for GNU/Linux environment and to process this file with the source command. You need to create in any text editor something like the following code:

unset OS\_SERVICE\_TOKEN

export OS\_AUTH\_URL=http://10.0.2.15:5000/v2.0

export OS\_TENANT\_NAME=admin

export OS\_REGION\_NAME=RegionOne

export OS\_USERNAME=admin

export OS\_PASSWORD=openstack

export OS\_IDENTITY\_API\_VERSION=3

You need to use your correct IP address and correct admin password for your environment. Now you can execute the script:

$ source keystonerc\_admin