Hardening OpenStack Identity Management with Keystone

LAB #3

Bryan Daniel, Justin, Cockrell, and Dylan Hagy

University of South Carolina Aiken

CSCI A591-PC

Dr. Ali AlSabeh

November 13, 2025

Scenario: Stonewall Consultants, Securing LumaTech's OpenStack Deployment

Prerequisites:

- Basic Linux command-line knowledge
- Understanding of virtualization concepts
- Access to a laptop that can run VMWare Workstation Pro
- Snapshot of Insecure VM

Learning Objectives:

Implement and test OpenStack identity and access security configurations to demonstrate how proper IAM, policy enforcement, and MFA integration reduce the risk of unauthorized or excessive access in a cloud environment. [4], [5], [6]

Background:

LumaTech, a fast-growing tech startup, recently deployed its own private cloud using OpenStack to host development and testing environments. However, their initial setup focused on functionality, not security. Concerned about potential insider misuse and unauthorized access, LumaTech hired your team, Stonewall Consultants, to establish a robust identity and access management framework that aligns with cybersecurity best practices.

Your Mission:

As Stonewall Consultants, your team is tasked with strengthening LumaTech's OpenStack security posture by configuring and demonstrating three key layers of protection:

1. **Structured identity management** – properly defining users, roles, and project access.
2. **Granular authorization policies** – enforcing least-privilege principles through service-level policy files.
3. **Strong authentication mechanisms** – enabling multi-factor authentication (MFA) for privileged users.

By the end of this engagement, LumaTech should have a clearly defined access hierarchy, policy-based restrictions for sensitive operations, and MFA protection for critical accounts.

Section 0: The Current state

Understanding the Company

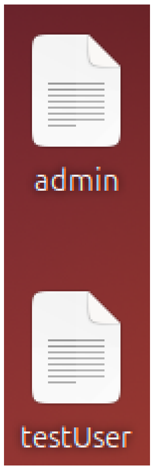
LumaTech had a need for their cloud hosting and as a fast start up, they started out by making an environment that only one user worked on. Over time, they added more people to the project but never revisited their security posture.

LumaTech's cloud server needs are simple. They require a test instance to develop and a dev instance to push code to once tested, but as is common in startups, security fell by the wayside. In the current setup, there are two environments that we will explore, luma-dev and luma-test. These environments help develop critical systems for the startup company.

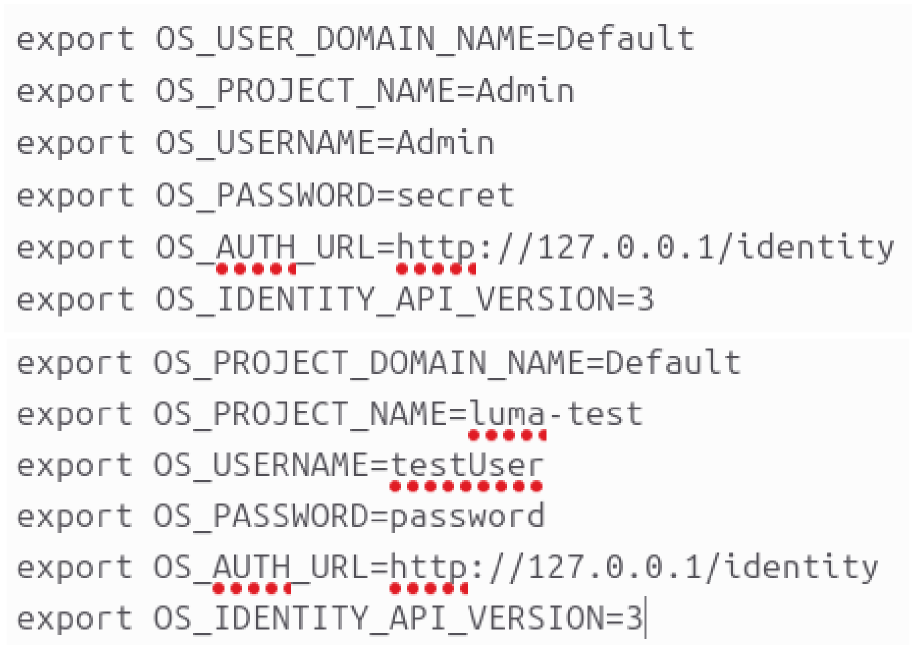
The current company structure involves Developers, QA, Network Admins and Security. In the current state, QA is given "TestUser" and the rest utilize the admin account. The DEV team will develop within both instances; QA only needs limited access to the test environment.

0.1 Investigate the RC Files on the Desktop

1. You're told that a user has been made for the QA team, which is a good start! Let's investigate the current setup by logging into the server!



2. Upon logging into the test environment, you will notice that there is an RC file for each of the users. Open each of these up by double clicking to reveal that the credentials are in plain text!



```
export OS_USER_DOMAIN_NAME=Default
export OS_PROJECT_NAME=Admin
export OS_USERNAME=Admin
export OS_PASSWORD=secret
export OS_AUTH_URL=http://127.0.0.1/identity
export OS_IDENTITY_API_VERSION=3
```

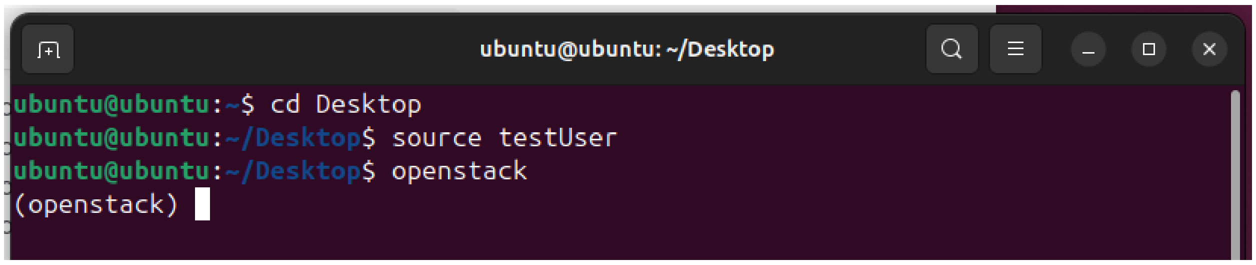
```
export OS_PROJECT_DOMAIN_NAME=Default
export OS_PROJECT_NAME=luma-test
export OS_USERNAME=testUser
export OS_PASSWORD=password
export OS_AUTH_URL=http://127.0.0.1/identity
export OS_IDENTITY_API_VERSION=3
```

Both login files are in plain text and right on the desktop

0.2 Investigate the Permissions of TestUser

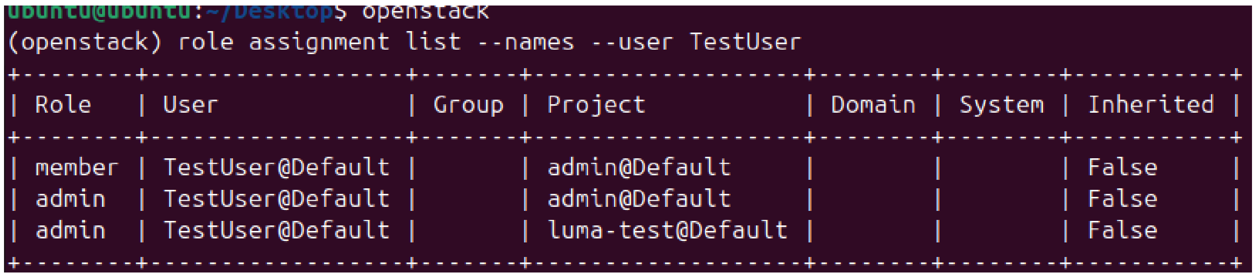
Next, we'll investigate the permissions of testUser through the terminal.

1. Launch your terminal and navigate to the desktop (cd Desktop)
2. Open the testUser file (source testUser)
3. Type openstack to open a session



```
ubuntu@ubuntu:~$ cd Desktop
ubuntu@ubuntu:~/Desktop$ source testUser
ubuntu@ubuntu:~/Desktop$ openstack
(openstack)
```

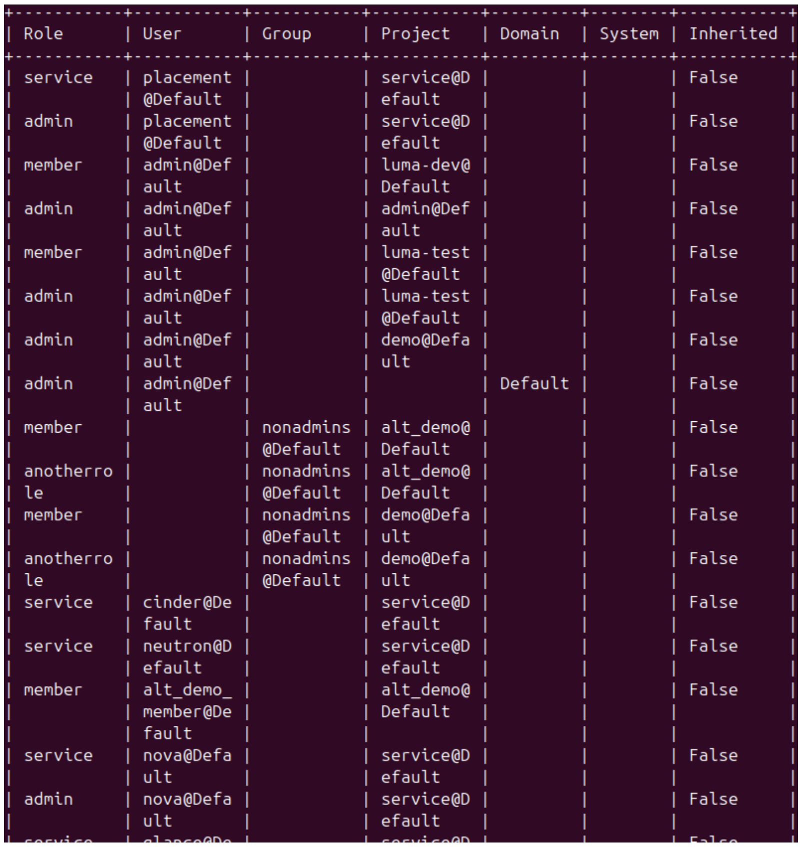
4. To investigate the roles that TestUser has, role assignment list --names --user TestUser



```
ubuntu@ubuntu:~/Desktop$ openstack
(openstack) role assignment list --names --user TestUser
+--------+-----------------+-------+------------------+--------+--------+-----------+
| Role   | User            | Group | Project          | Domain | System | Inherited |
+--------+-----------------+-------+------------------+--------+--------+-----------+
| member | TestUser@Default |      | admin@Default    |        |        | False     |
| admin  | TestUser@Default |      | admin@Default    |        |        | False     |
| admin  | TestUser@Default |      | luma-test@Default |       |        | False     |
+--------+-----------------+-------+------------------+--------+--------+-----------+
```

We discovered that though TestUser is not a member of Dev, it is an admin for the admin project

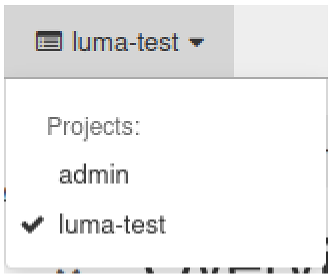
5. Run an overall query of the role assignments and take note (role assignment list –names)



```
+-----------+-----------+-----------+-----------+---------+--------+-----------+
| Role      | User      | Group     | Project   | Domain  | System | Inherited |
+-----------+-----------+-----------+-----------+---------+--------+-----------+
| service   | placement |           | service@D |         |        | False     |
|           | @Default  |           | efault    |         |        |           |
| admin     | placement |           | service@D |         |        | False     |
|           | @Default  |           | efault    |         |        |           |
| member    | admin@Def |           | luma-dev@ |         |        | False     |
|           | ault      |           | Default   |         |        |           |
| admin     | admin@Def |           | admin@Def |         |        | False     |
|           | ault      |           | ault      |         |        |           |
| member    | admin@Def |           | luma-test |         |        | False     |
|           | ault      |           | @Default  |         |        |           |
| admin     | admin@Def |           | luma-test |         |        | False     |
|           | ault      |           | @Default  |         |        |           |
| admin     | admin@Def |           | demo@Defa |         |        | False     |
|           | ault      |           | ult       |         |        |           |
| admin     | admin@Def |           |           | Default |        | False     |
|           | ault      |           |           |         |        |           |
| member    |           | nonadmins | alt_demo@ |         |        | False     |
|           |           | @Default  | Default   |         |        |           |
| anotherro |           | nonadmins | alt_demo@ |         |        | False     |
| le        |           | @Default  | Default   |         |        |           |
| member    |           | nonadmins | demo@Defa |         |        | False     |
|           |           | @Default  | ult       |         |        |           |
| anotherro |           | nonadmins | demo@Defa |         |        | False     |
| le        |           | @Default  | ult       |         |        |           |
| service   | cinder@De |           | service@D |         |        | False     |
|           | fault     |           | efault    |         |        |           |
| service   | neutron@D |           | service@D |         |        | False     |
|           | efault    |           | efault    |         |        |           |
| member    | alt_demo_ |           | alt_demo@ |         |        | False     |
|           | member@De |           | Default   |         |        |           |
|           | fault     |           |           |         |        |           |
| service   | nova@Defa |           | service@D |         |        | False     |
|           | ult       |           | efault    |         |        |           |
| admin     | nova@Defa |           | service@D |         |        | False     |
|           | ult       |           | efault    |         |        |           |
```

A preview of some of the members and access

6. Navigate in Firefox to http://127.0.0.1 and login with TestUser (Password: password)
    a. Notice that you are not prompted for MFA
    b. Your Default project is Luma-Test
    c. You can select the admin project



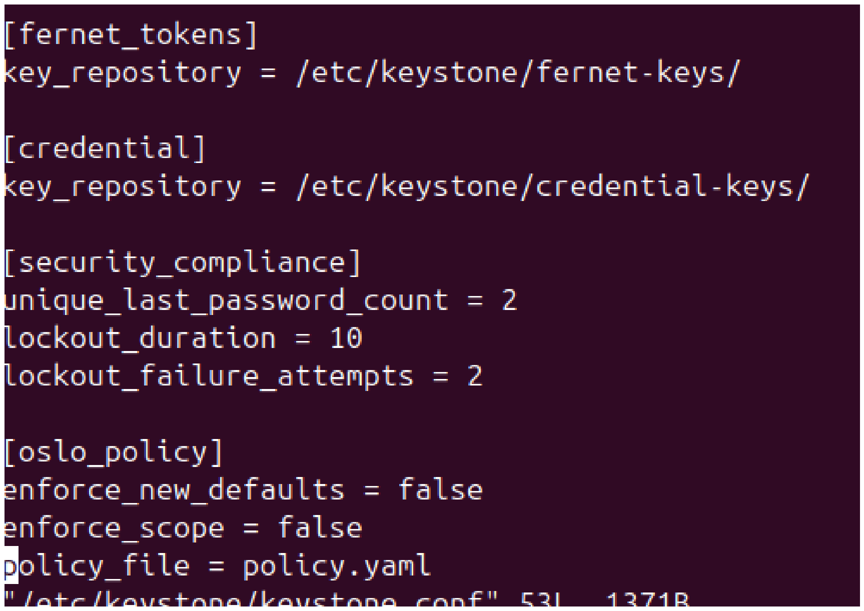
No MFA and access to the admin portal

After reviewing the member account creation, we have determined the following:

- All users use a shared account (Either admin or TestUser)
- TestUser has more access than intended
- The credentials of both accounts exist in plain text
- The password strength is very weak

0.3 Review Policy Files

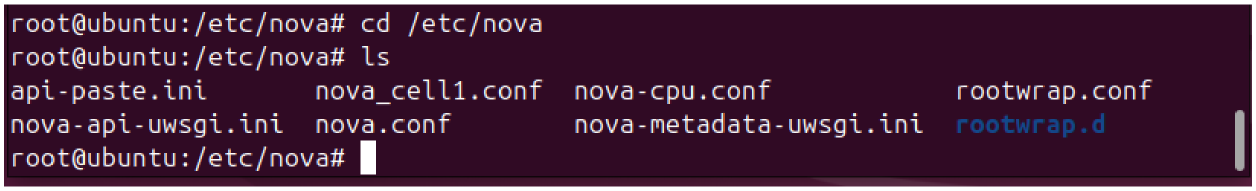
1. In a new terminal window, type (sudo vim /etc/keystone/keystone.conf)



```
[fernet_tokens]
key_repository = /etc/keystone/fernet-keys/

[credential]
key_repository = /etc/keystone/credential-keys/

[security_compliance]
unique_last_password_count = 2
lockout_duration = 10
lockout_failure_attempts = 2

[oslo_policy]
enforce_new_defaults = false
enforce_scope = false
policy_file = policy.yaml
"/etc/keystone/keystone.conf" 53L, 1371B
```

Default Keystone configuration with very little restrictions on logins [1]

2. Navigate to /etc/nova and investigate the contents of the directory



```
root@ubuntu:/etc/nova# cd /etc/nova
root@ubuntu:/etc/nova# ls
api-paste.ini       nova_cell1.conf  nova-cpu.conf            rootwrap.conf
nova-api-uwsgi.ini  nova.conf        nova-metadata-uwsgi.ini  rootwrap.d
root@ubuntu:/etc/nova# 
```

There is no Policy.yaml present meaning nothing designates access within NOVA. [3]

0.4 Assessment of the Current Environment

Upon review of the current cloud environment, we discovered the following:

1. Generic shared user accounts are currently in place with no password policies or multifactor authentication
2. The current shared TestUser account has more access than intended.
3. There is no user groups currently made, and all access is segmented between 2 static shared user accounts.
4. Developers are currently sharing an admin account with full permission to the cloud environment
5. No policies exist within the Nova Environment meaning all users have full access within nova to modify instances
6. No network segmentation exists. Neutron is not configured, and we operate with default IPs [1], [5]

Section 1: Identity and Access Segmentation

Your Assessment of the Cloud

After assessing LumaTech's initial environment, it became clear that the company's rapid growth outpaced its access control design. The environment relied on a few shared accounts that granted broad administrative permissions across all projects. This approach made it impossible to distinguish accountability or apply the principle of least privilege. [1] [5]

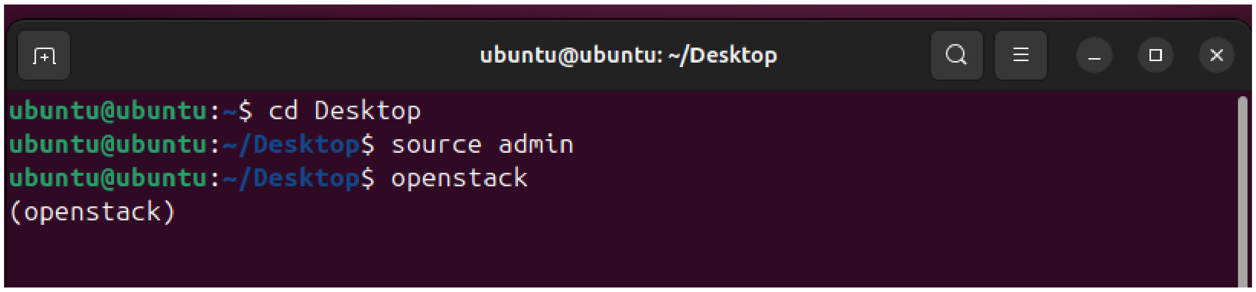
To begin remediating these issues, the first step focuses on redefining identity and access boundaries. Each user and team will receive individual accounts, assigned to clearly defined roles within their appropriate projects. Two main projects form the basis of this segmentation, Luma-Dev and Luma-Test. Developers will work solely within the Dev environment, QA testers will be restricted to the Test environment, and administrators will maintain oversight of both.

This phase establishes the foundation for all later security controls by moving from generic, shared credentials to a structured role-based model. Once implemented, LumaTech's cloud will have traceable user actions, compartmentalized project spaces, and a framework that enforces accountability and least-privilege access.

1.1 Create Dedicated Users and Roles

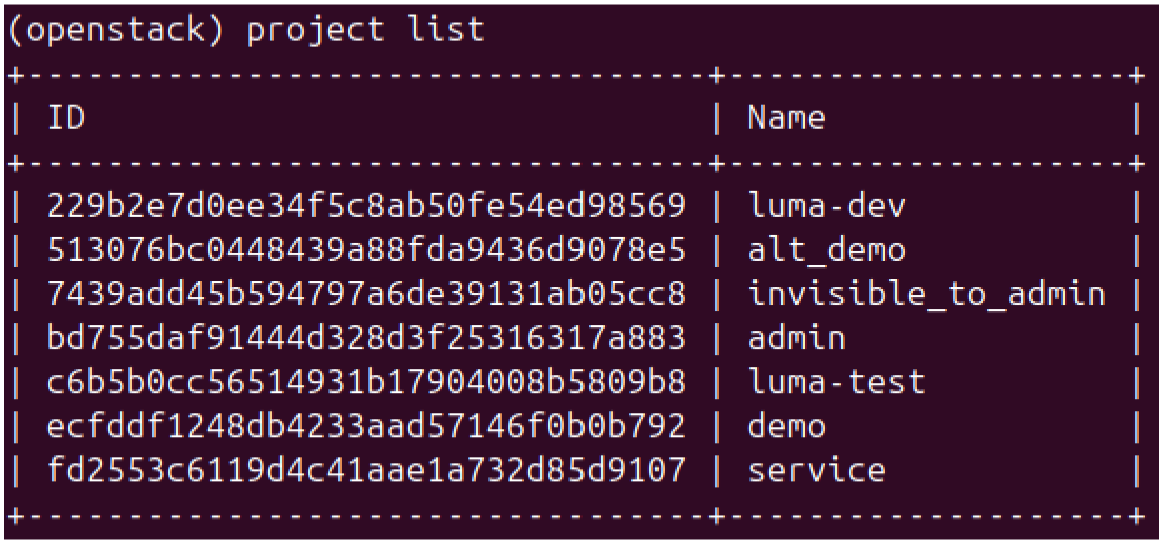
First, we will begin by creating dedicated users and roles using the terminal.

1. Launch your terminal and navigate to the desktop (cd Desktop)
2. Open the admin file (source admin)
3. Type openstack to open a session



```
ubuntu@ubuntu:~$ cd Desktop
ubuntu@ubuntu:~/Desktop$ source admin
ubuntu@ubuntu:~/Desktop$ openstack
(openstack)
```

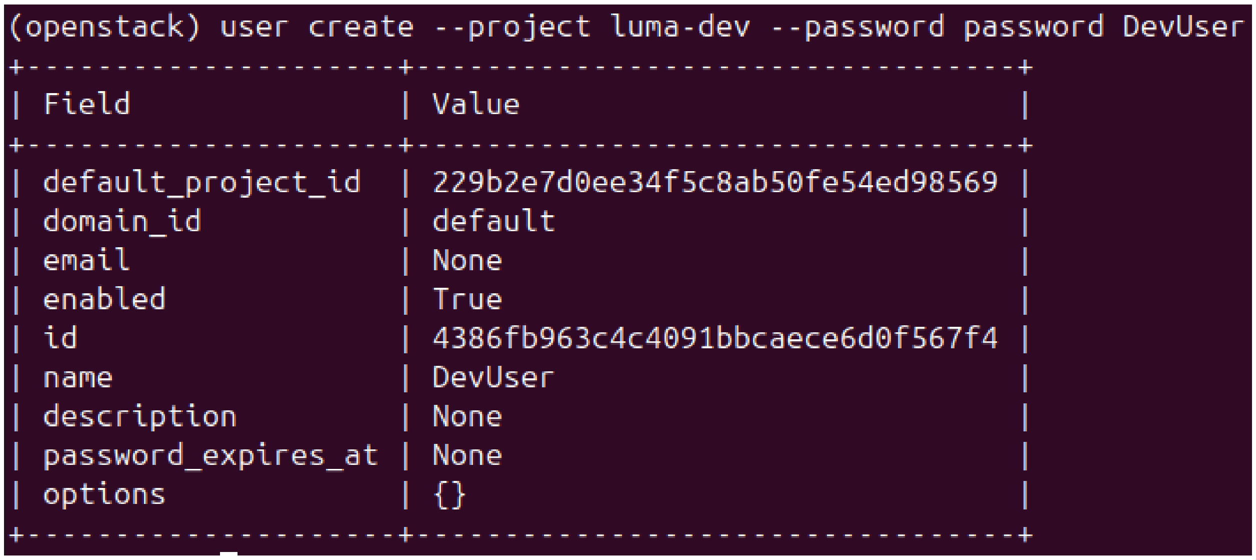
4. In the OpenStack CLI, check the list of projects (project list)



```
(openstack) project list
+----------------------------------+--------------------+
| ID                               | Name               |
+----------------------------------+--------------------+
| 229b2e7d0ee34f5c8ab50fe54ed98569 | luma-dev           |
| 513076bc0448439a88fda9436d9078e5 | alt_demo           |
| 7439add45b594797a6de39131ab05cc8 | invisible_to_admin |
| bd755daf91444d328d3f25316317a883 | admin              |
| c6b5b0cc56514931b17904008b5809b8 | luma-test          |
| ecfddf1248db4233aad57146f0b0b792 | demo               |
| fd2553c6119d4c41aae1a732d85d9107 | service            |
+----------------------------------+--------------------+
```

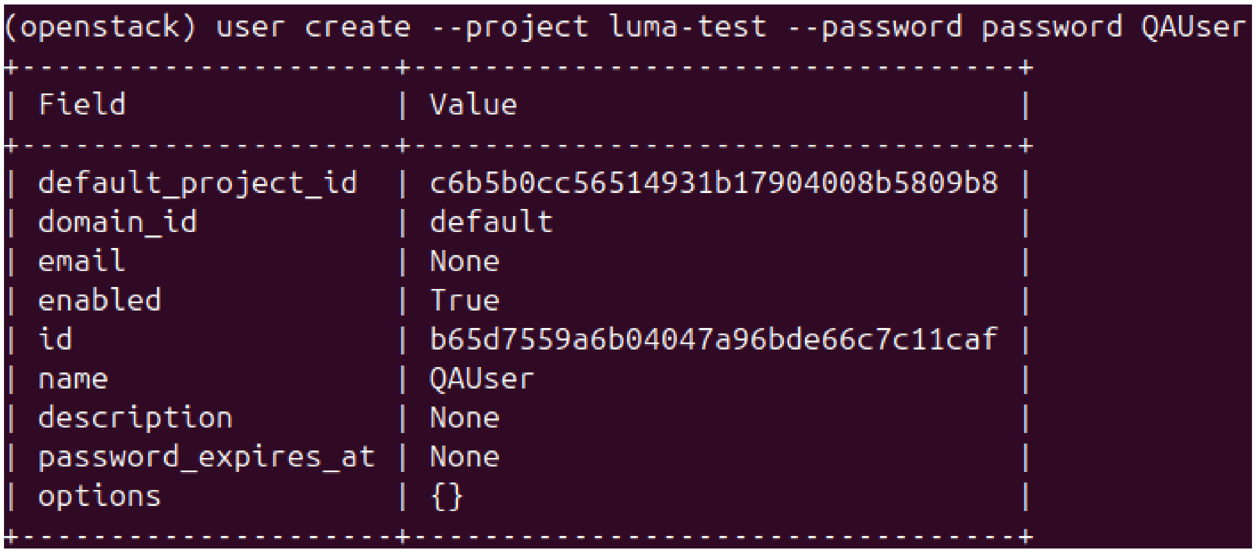
The admin, luma-dev, and luma-test projects will be the ones we will harden.

5. Next, we will create 3 new users with the following CLI commands:



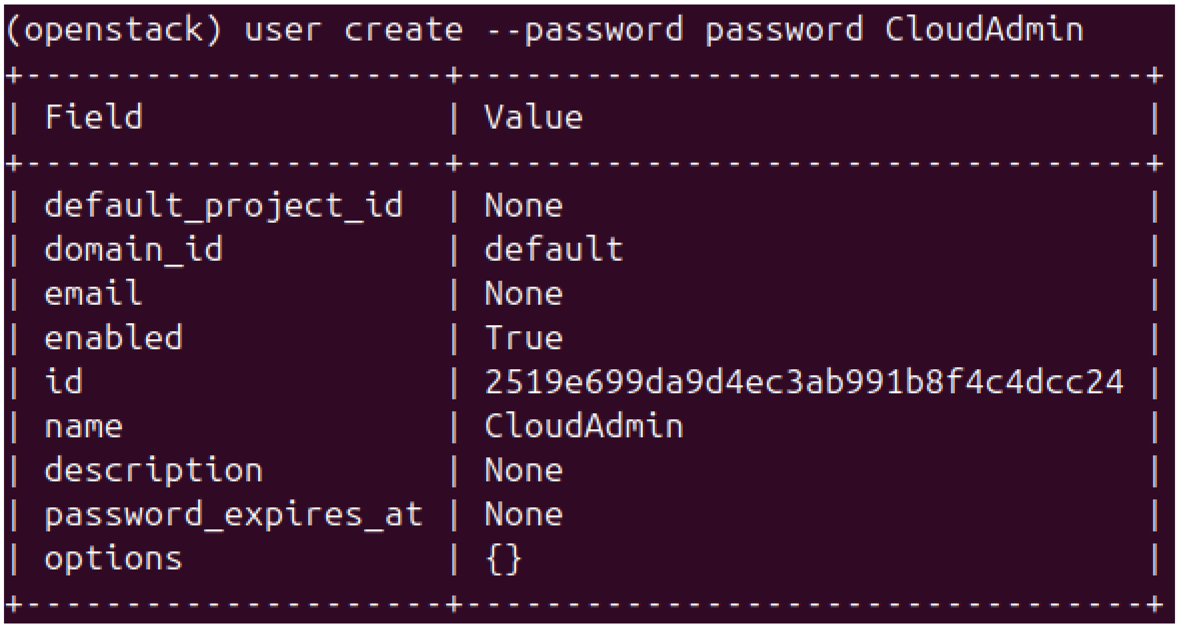
```
(openstack) user create --project luma-dev --password password DevUser
+---------------------+----------------------------------+
| Field               | Value                            |
+---------------------+----------------------------------+
| default_project_id  | 229b2e7d0ee34f5c8ab50fe54ed98569 |
| domain_id           | default                          |
| email               | None                             |
| enabled             | True                             |
| id                  | 4386fb963c4c4091bbcaece6d0f567f4 |
| name                | DevUser                          |
| description         | None                             |
| password_expires_at | None                             |
| options             | {}                               |
+---------------------+----------------------------------+
```

The DevUser is now created with their default project set to luma-dev



```
(openstack) user create --project luma-test --password password QAUser
+---------------------+----------------------------------+
| Field               | Value                            |
+---------------------+----------------------------------+
| default_project_id  | c6b5b0cc56514931b17904008b5809b8 |
| domain_id           | default                          |
| email               | None                             |
| enabled             | True                             |
| id                  | b65d7559a6b04047a96bde66c7c11caf |
| name                | QAUser                           |
| description         | None                             |
| password_expires_at | None                             |
| options             | {}                               |
+---------------------+----------------------------------+
```

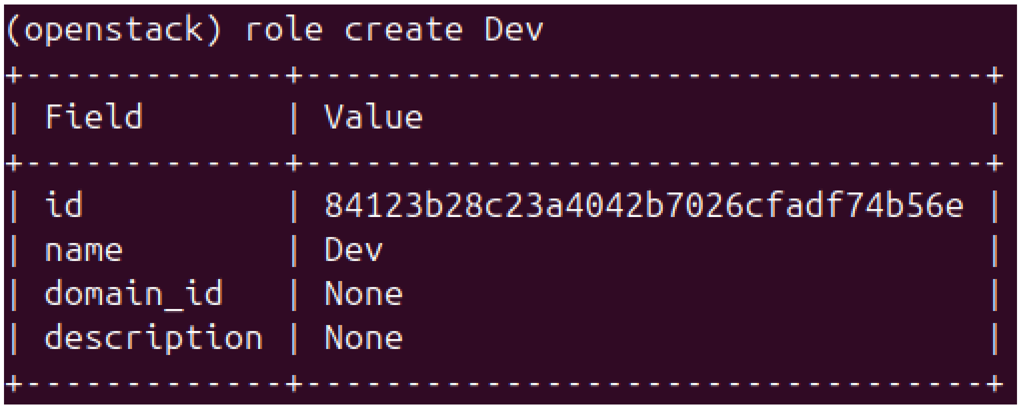
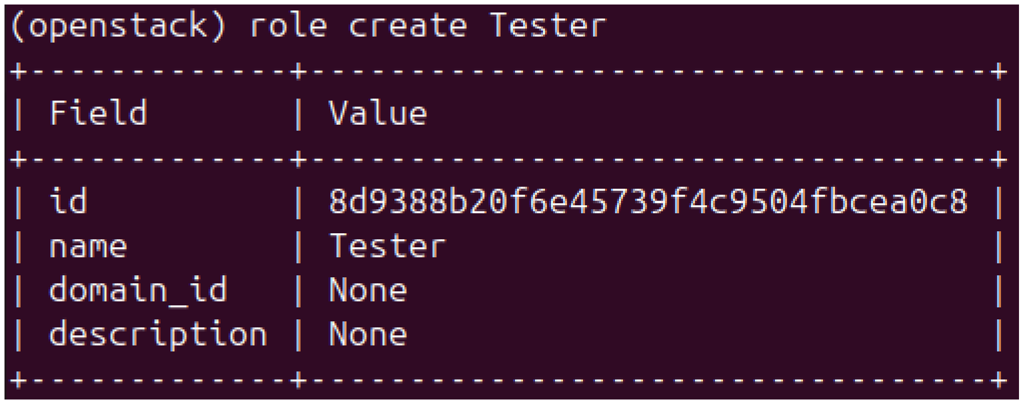
The QAUser is now created with their default project set to luma-test

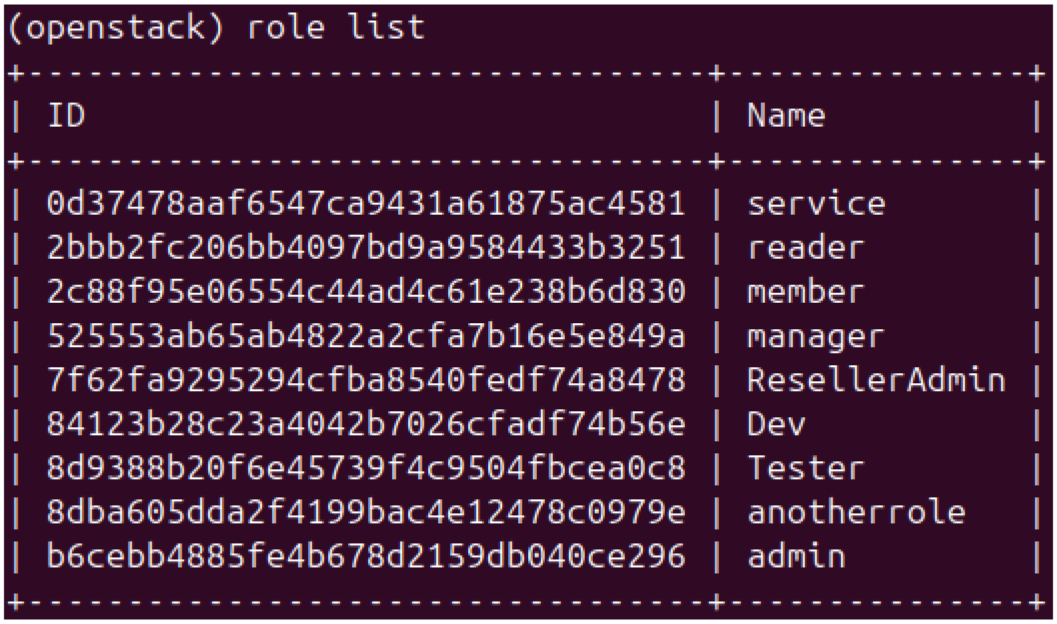


```
(openstack) user create --password password CloudAdmin
+---------------------+----------------------------------+
| Field               | Value                            |
+---------------------+----------------------------------+
| default_project_id  | None                             |
| domain_id           | default                          |
| email               | None                             |
| enabled             | True                             |
| id                  | 2519e699da9d4ec3ab991b8f4c4dcc24 |
| name                | CloudAdmin                       |
| description         | None                             |
| password_expires_at | None                             |
| options             | {}                               |
+---------------------+----------------------------------+
```

The CloudAdmin is now created with no default project set

6. Then we need to create two new roles for developers and testers and confirm they exist (role create & role list). The admin role has already been created.

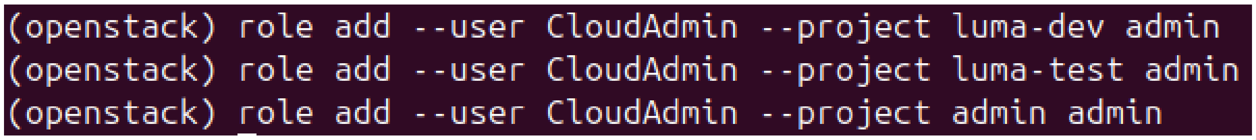

```
(openstack) role create Dev
+-------------+----------------------------------+
| Field       | Value                            |
+-------------+----------------------------------+
| id          | 84123b28c23a4042b7026cfadf74b56e |
| name        | Dev                              |
| domain_id   | None                             |
| description | None                             |
+-------------+----------------------------------+
```

```
(openstack) role create Tester
+-------------+----------------------------------+
| Field       | Value                            |
+-------------+----------------------------------+
| id          | 8d9388b20f6e45739f4c9504fbcea0c8 |
| name        | Tester                           |
| domain_id   | None                             |
| description | None                             |
+-------------+----------------------------------+
```
Both above screenshots show the two roles being created


```
(openstack) role list
+----------------------------------+---------------+
| ID                               | Name          |
+----------------------------------+---------------+
| 0d37478aaf6547ca9431a61875ac4581 | service       |
| 2bbb2fc206bb4097bd9a9584433b3251 | reader        |
| 2c88f95e06554c44ad4c61e238b6d830 | member        |
| 525553ab65ab4822a2cfa7b16e5e849a | manager       |
| 7f62fa9295294cfba8540fedf74a8478 | ResellerAdmin |
| 84123b28c23a4042b7026cfadf74b56e | Dev           |
| 8d9388b20f6e45739f4c9504fbcea0c8 | Tester        |
| 8dba605dda2f4199bac4e12478c0979e | anotherrole   |
| b6cebb4885fe4b678d2159db040ce296 | admin         |
+----------------------------------+---------------+
```
The Dev, Tester, and admin roles all exist in the role list.

1.2 Assign Roles to Users

We will now assign roles to the users to ensure each user only has access to the correct projects. [2]

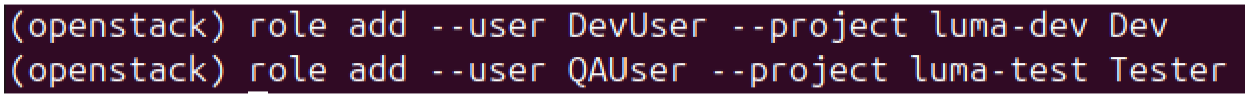
1. First, we ensure the new CloudAdmin is assigned as an admin in all three projects



```
(openstack) role add --user CloudAdmin --project luma-dev admin
(openstack) role add --user CloudAdmin --project luma-test admin
(openstack) role add --user CloudAdmin --project admin admin
```

CloudAdmin will get admin-level access to all three projects

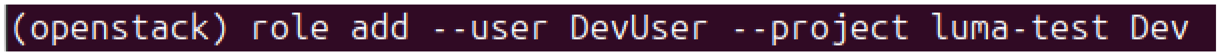
2. Then, we assign the proper roles to both DevUser and QAUser



```
(openstack) role add --user DevUser --project luma-dev Dev
(openstack) role add --user QAUser --project luma-test Tester
```

DevUser and QAUser will only have access to their respective projects

3. Next, we will go ahead and add our developer to the test group



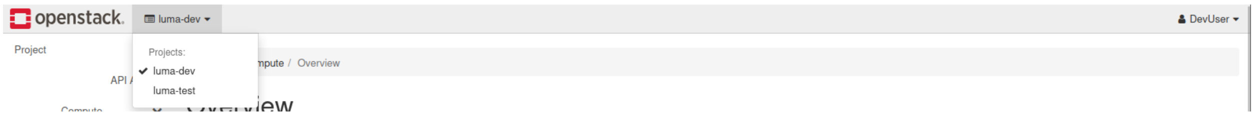
```
(openstack) role add --user DevUser --project luma-test Dev
```

Our developer will get privileged access to the luma-test environment

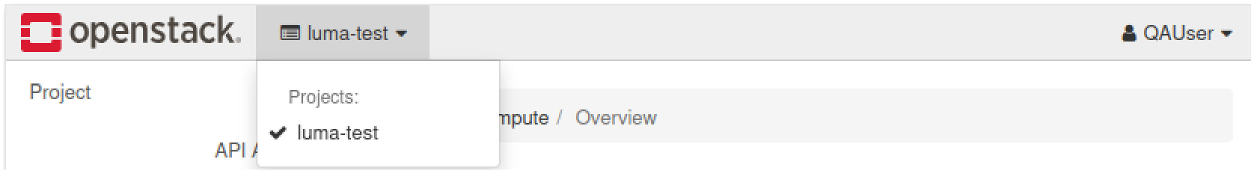
1.3 Verify Scoped Access and Remove Shared Accounts [1], [5]

At this point, we have set up the three new users (CloudAdmin, DevUser, and QAUser) that we plan to use for LumaTech. Next, we will verify that each user's access is correctly scoped to their project and retire from the old, shared accounts to enforce least-privilege practices.

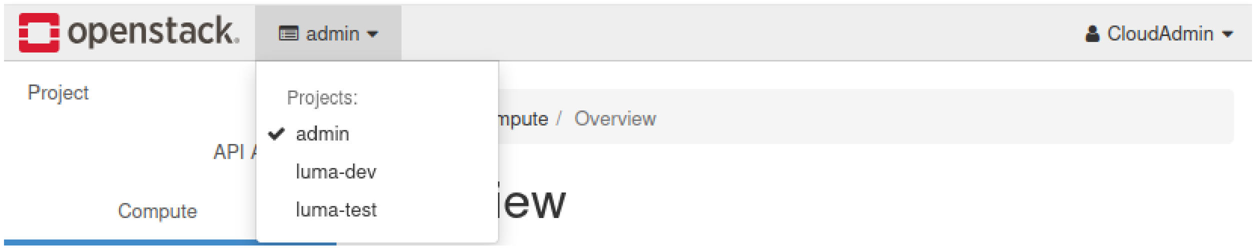
1. Login to the Horizon Dashboard with all three of the user accounts (username for each account and "password") and verify the correct projects are showing up for each user.



DevUser can view luma-dev and luma-test

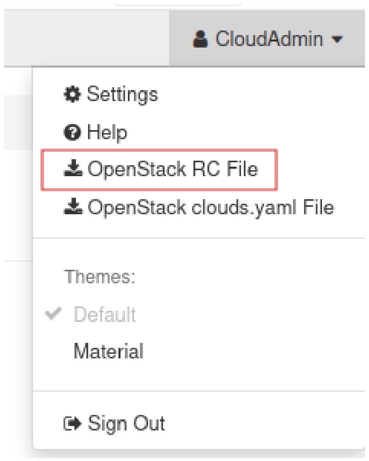


QAUser can only view luma-test project

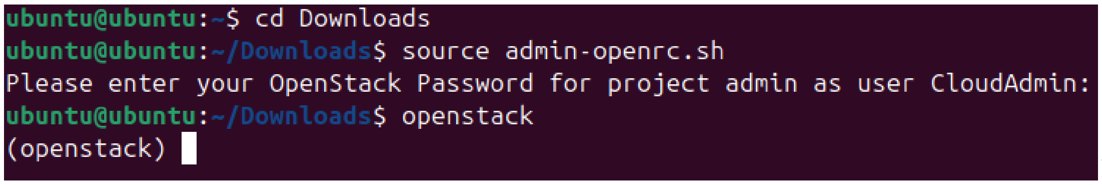


CloudAdmin can view all three projects

2. While logged into the CloudAdmin account, download the OpenStack RC File, then open the OpenStack CLI using that file in a new terminal using the correct password ("password").



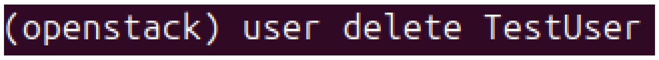
Download the RC File from this location



```
ubuntu@ubuntu:~$ cd Downloads
ubuntu@ubuntu:~/Downloads$ source admin-openrc.sh
Please enter your OpenStack Password for project admin as user CloudAdmin:
ubuntu@ubuntu:~/Downloads$ openstack
(openstack) 
```

You are now logged in as the CloudAdmin user

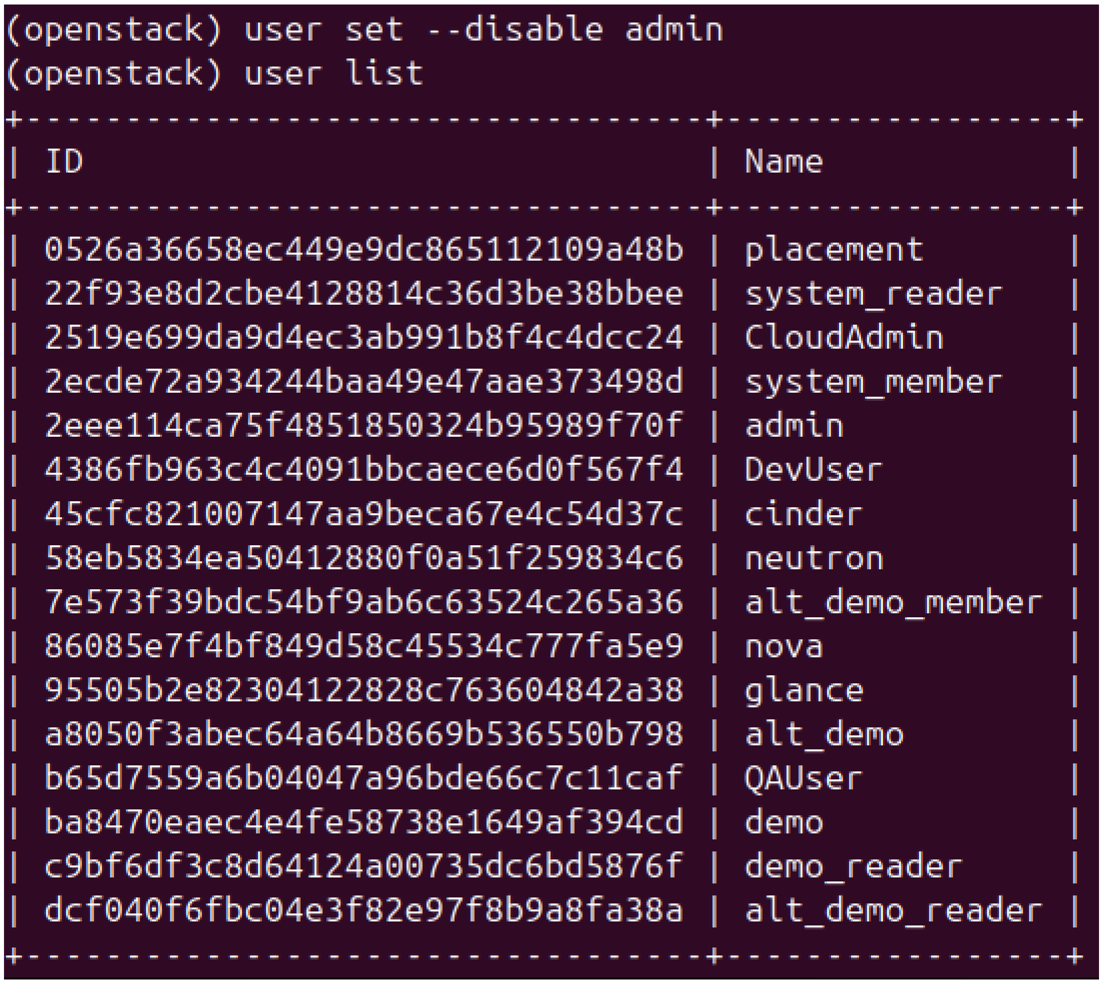
3. Delete the old TestUser account (user delete TestUser)



```
(openstack) user delete TestUser
```

This removes the old TestUser account since it is being replaced

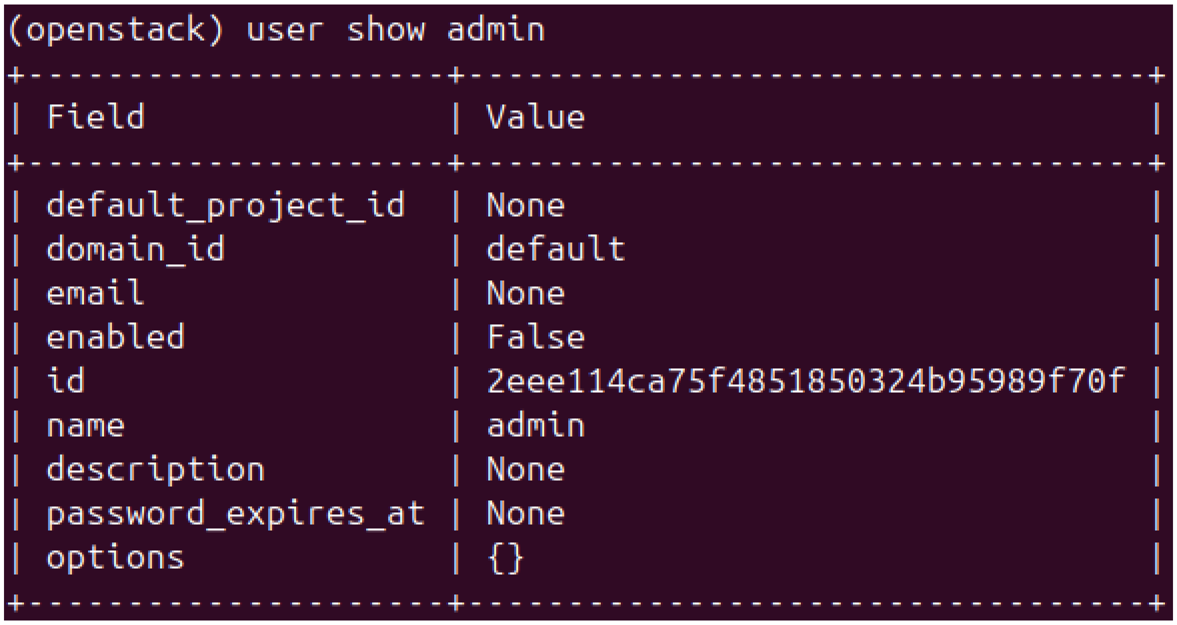
4. We will disable the old admin account, since a new one has been set up (CloudAdmin). Then, we will view the user list to see what has changed



```
(openstack) user set --disable admin
(openstack) user list
+----------------------------------+-----------------+
| ID                               | Name            |
+----------------------------------+-----------------+
| 0526a36658ec449e9dc865112109a48b | placement       |
| 22f93e8d2cbe4128814c36d3be38bbee | system_reader   |
| 2519e699da9d4ec3ab991b8f4c4dcc24 | CloudAdmin      |
| 2ecde72a934244baa49e47aae373498d | system_member   |
| 2eee114ca75f4851850324b95989f70f | admin           |
| 4386fb963c4c4091bbcaece6d0f567f4 | DevUser         |
| 45cfc821007147aa9beca67e4c54d37c | cinder          |
| 58eb5834ea50412880f0a51f259834c6 | neutron         |
| 7e573f39bdc54bf9ab6c63524c265a36 | alt_demo_member |
| 86085e7f4bf849d58c45534c777fa5e9 | nova            |
| 95505b2e82304122828c763604842a38 | glance          |
| a8050f3abec64a64b8669b536550b798 | alt_demo        |
| b65d7559a6b04047a96bde66c7c11caf | QAUser          |
| ba8470eaec4e4fe58738e1649af394cd | demo            |
| c9bf6df3c8d64124a00735dc6bd5876f | demo_reader     |
| dcf040f6fbc04e3f82e97f8b9a8fa38a | alt_demo_reader |
+----------------------------------+-----------------+
```

The TestUser account was deleted, but the original admin account remains because it was only disabled

5. To verify that the original admin account was disabled, we view the admin user to ensure that the enabled field is set to false



```
(openstack) user show admin
+---------------------+----------------------------------+
| Field               | Value                            |
+---------------------+----------------------------------+
| default_project_id  | None                             |
| domain_id           | default                          |
| email               | None                             |
| enabled             | False                            |
| id                  | 2eee114ca75f4851850324b95989f70f |
| name                | admin                            |
| description         | None                             |
| password_expires_at | None                             |
| options             | {}                               |
+---------------------+----------------------------------+
```

The original admin user is disabled correctly

1.4 Overview

So far, we set up and confirmed the following:

1. Three new users (CloudAdmin, DevUser, and QAUser) have been created with defined roles and project scopes.
2. The old, shared accounts have been deleted/disabled to prevent unauthorized or untracked access.
3. Scoped access has been verified, ensuring that DevUser and QAUser can only see and act within their respective projects, while CloudAdmin retains full administrative privileges.
4. The environment now enforces least-privilege principles, laying the foundation for policy enforcement and authentication hardening in the following sections. [3], [4]

Section 2: Policy Configuration (Fine Grained Access Control)

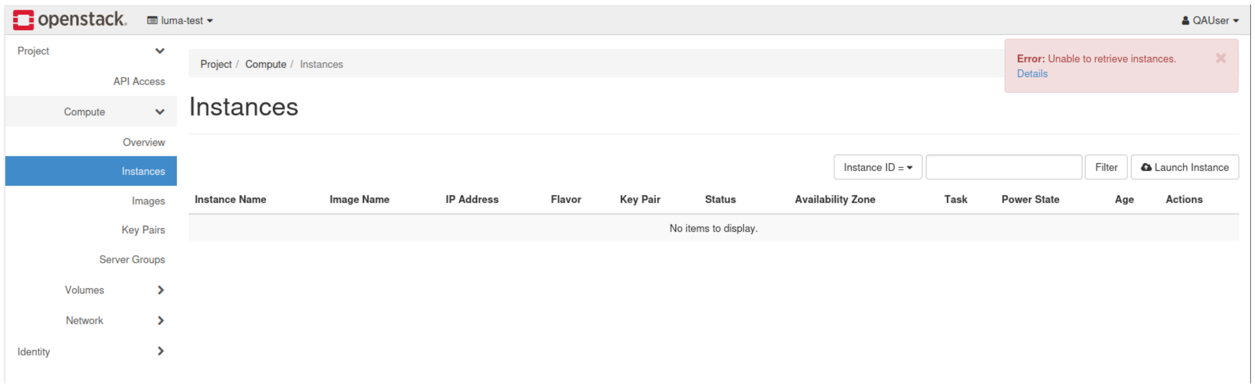
Now that we've created appropriate user accounts, users are successfully isolated within their intended projects. As it stands now, they have full control within their given projects which is not intended. In this section, we will be further limiting the use of various features within the project to ensure we are granting least Privileged Access. [1], [5]

2.1 Investigate Roles

Login to the web console as QAUser and investigate Instances.

Username: QAUser

Password: password



openstack. luma-test QAUser

Project / Compute / Instances

Error: Unable to retrieve instances.
Details

Project
API Access
Compute
Overview
Instances
Images
Key Pairs
Server Groups
Volumes
Network
Identity

Instances

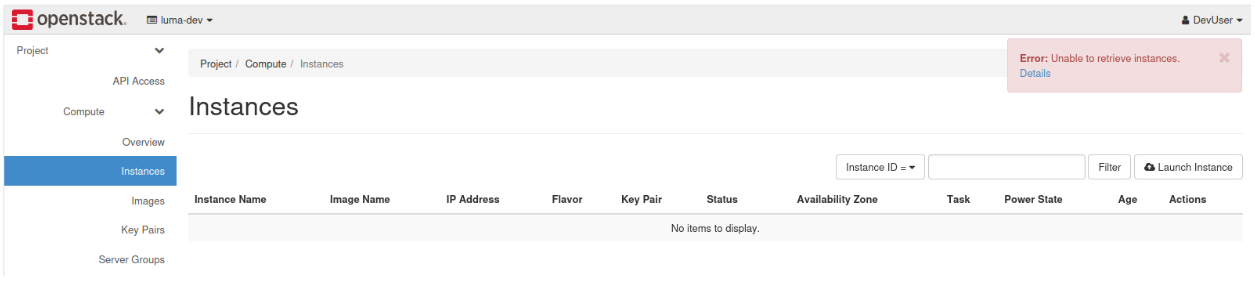
Instance ID = Filter Launch Instance

Instance Name	Image Name	IP Address	Flavor	Key Pair	Status	Availability Zone	Task	Power State	Age	Actions
No items to display.										

We can see that our new role does not have access to the instances by default

Username: DevUser

Password: password



openstack. luma-dev DevUser

Project / Compute / Instances

Error: Unable to retrieve instances.
Details

Project
API Access
Compute
Overview
Instances
Images
Key Pairs
Server Groups

Instances

Instance ID = Filter Launch Instance

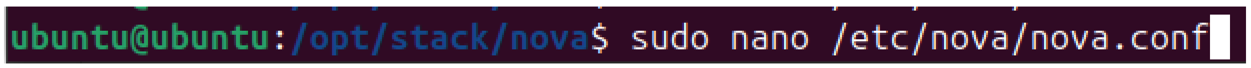
Instance Name	Image Name	IP Address	Flavor	Key Pair	Status	Availability Zone	Task	Power State	Age	Actions
No items to display.										

Our dev user has the same problem within his environment.

2.2 Defining permissions

In order to allow our users access to our instances, they need customized permissions. The sky is the limit with custom permissions, and we will get a great start in this module. In this module we will be utilizing syntax defined by the official OpenStack documentation [3]

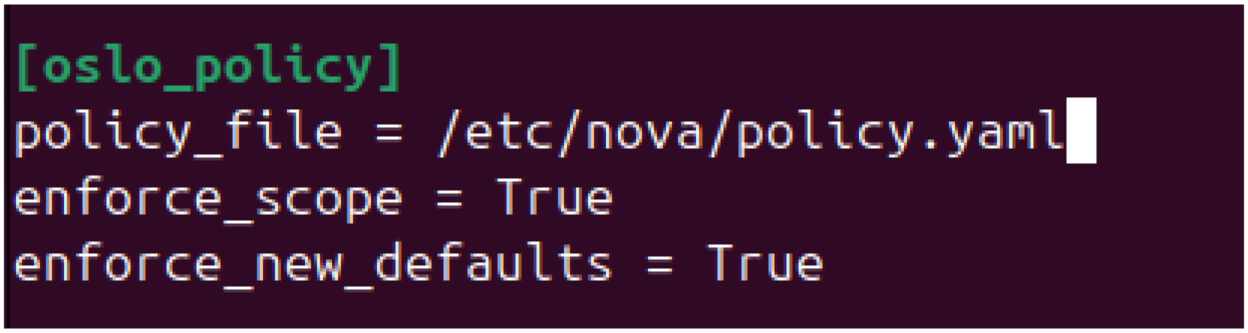
1. Open up a terminal and use nano to edit the nova.conf



```
ubuntu@ubuntu:/opt/stack/nova$ sudo nano /etc/nova/nova.conf
```

Opens the existing nova.conf in a nano editor

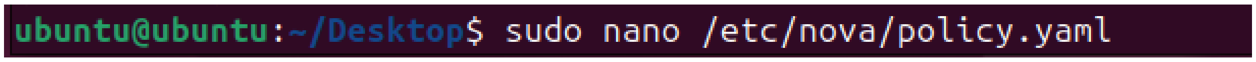
2. Under [oslo_policy] add a line to specify a custom policy file as seen below



```
[oslo_policy]
policy_file = /etc/nova/policy.yaml
enforce_scope = True
enforce_new_defaults = True
```

Policy.yaml does not yet exist but this will tell nova to look for it

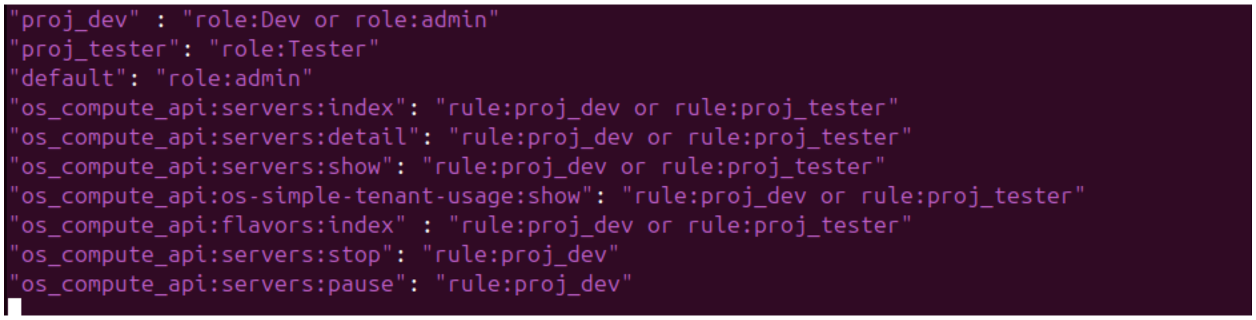
3. Open up the policy.yaml in nano as specified before.



```
ubuntu@ubuntu:~/Desktop$ sudo nano /etc/nova/policy.yaml
```

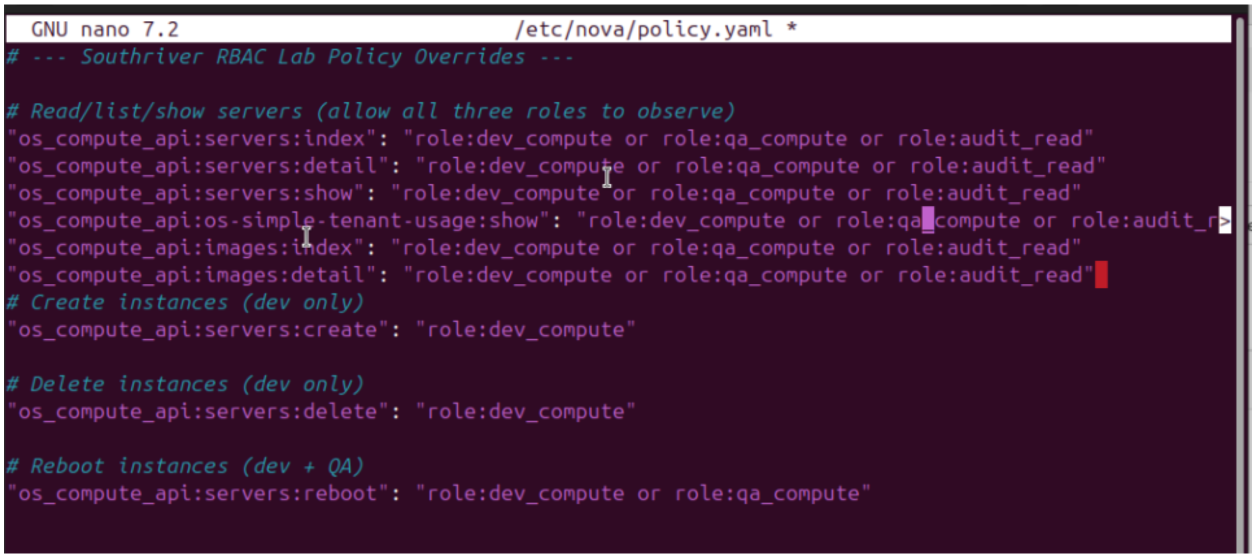
This file does not yet exist, so we need to make it!

4. In the newly opened blank file, copy the code as shown below and save the file



```
"proj_dev" : "role:Dev or role:admin"
"proj_tester": "role:Tester"
"default": "role:admin"
"os_compute_api:servers:index": "rule:proj_dev or rule:proj_tester"
"os_compute_api:servers:detail": "rule:proj_dev or rule:proj_tester"
"os_compute_api:servers:show": "rule:proj_dev or rule:proj_tester"
"os_compute_api:os-simple-tenant-usage:show": "rule:proj_dev or rule:proj_tester"
"os_compute_api:flavors:index" : "rule:proj_dev or rule:proj_tester"
"os_compute_api:servers:stop": "rule:proj_dev"
"os_compute_api:servers:pause": "rule:proj_dev"
```

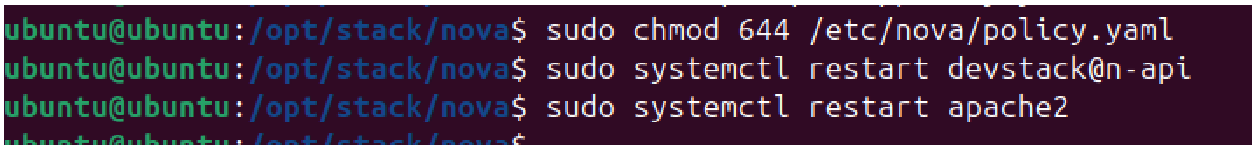
Justin edit below temporarily.



```
  GNU nano 7.2                          /etc/nova/policy.yaml *
# --- Southriver RBAC Lab Policy Overrides ---

# Read/list/show servers (allow all three roles to observe)
"os_compute_api:servers:index": "role:dev_compute or role:qa_compute or role:audit_read"
"os_compute_api:servers:detail": "role:dev_compute or role:qa_compute or role:audit_read"
"os_compute_api:servers:show": "role:dev_compute or role:qa_compute or role:audit_read"
"os_compute_api:os-simple-tenant-usage:show": "role:dev_compute or role:qa_compute or role:audit_r>
"os_compute_api:images:index": "role:dev_compute or role:qa_compute or role:audit_read"
"os_compute_api:images:detail": "role:dev_compute or role:qa_compute or role:audit_read"
# Create instances (dev only)
"os_compute_api:servers:create": "role:dev_compute"

# Delete instances (dev only)
"os_compute_api:servers:delete": "role:dev_compute"

# Reboot instances (dev + QA)
"os_compute_api:servers:reboot": "role:dev_compute or role:qa_compute"
```

The admin gets all permissions that the dev gets and then reverts to default

5. This config allows the dev and test groups to view instances, but only the dev can interact with them on a limited basis.
6. To see the changes live we will grant permissions to the new file and restart nova.



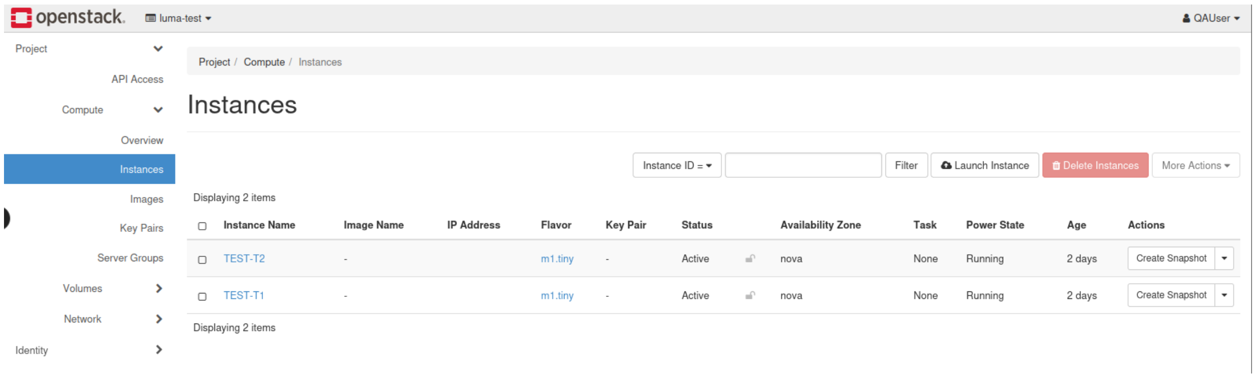
```
ubuntu@ubuntu:/opt/stack/nova$ sudo chmod 644 /etc/nova/policy.yaml
ubuntu@ubuntu:/opt/stack/nova$ sudo systemctl restart devstack@n-api
ubuntu@ubuntu:/opt/stack/nova$ sudo systemctl restart apache2
ubuntu@ubuntu:/opt/stack/nova$
```

Grants permission to the file and restarts both nova and apache

2.3 Analyzing our new permission set

Now that we have applied a custom policy, let's analyze the results!

1. Login to openstack on the web using our QAUser account
   Username: QAUser
   Password: password



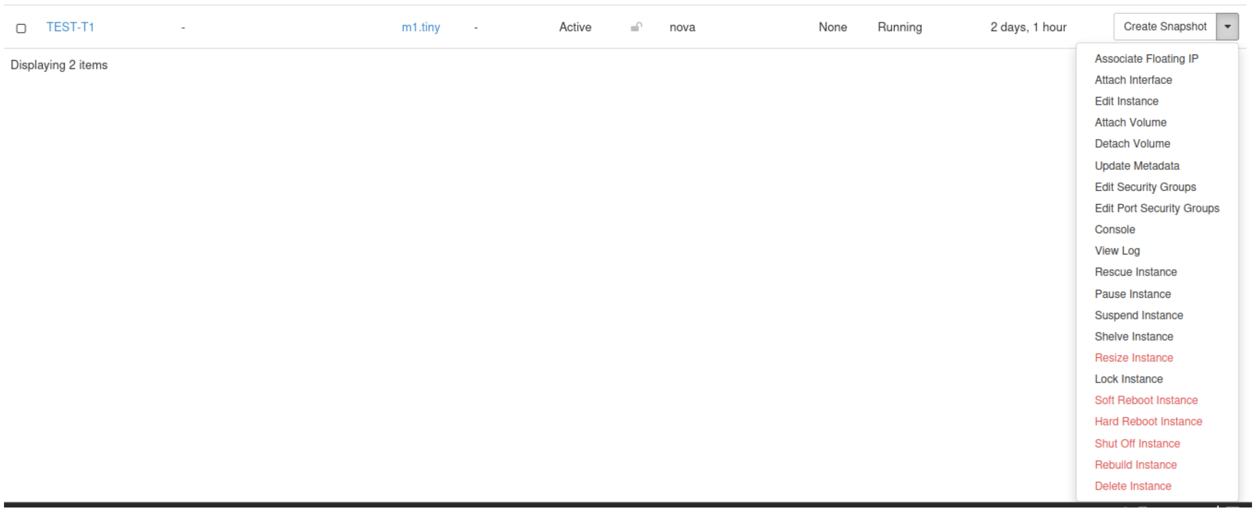
Displaying 2 items

Instance Name	Image Name	IP Address	Flavor	Key Pair	Status	Availability Zone	Task	Power State	Age	Actions
TEST-T2	-		m1.tiny	-	Active	nova	None	Running	2 days	Create Snapshot
TEST-T1	-		m1.tiny	-	Active	nova	None	Running	2 days	Create Snapshot

Displaying 2 items

With custom permissions, we can see our instances!

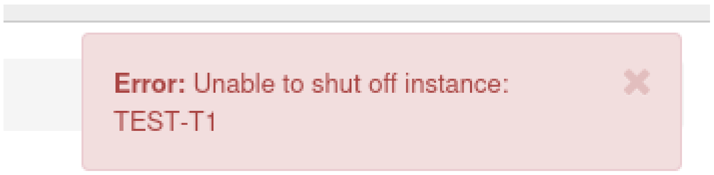
2. Utilize Horizon to attempt to shut down one of our instances



TEST-T1	-	m1.tiny	-	Active	nova	None	Running	2 days, 1 hour	Create Snapshot

Displaying 2 items

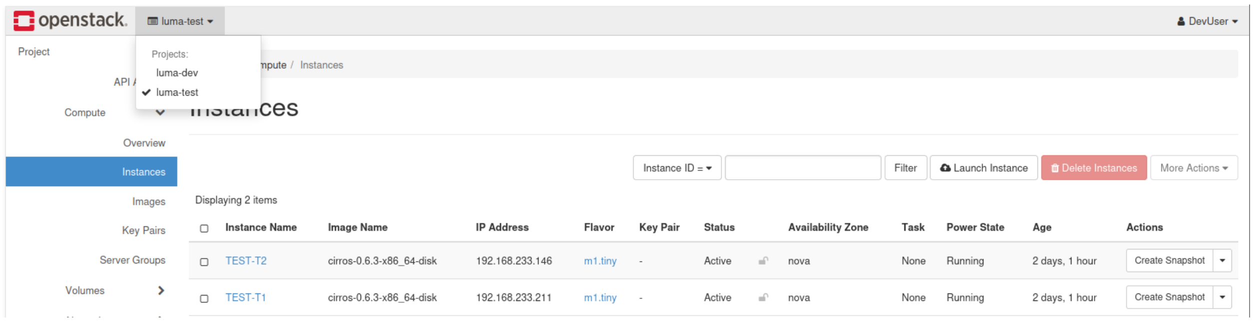
Attempt to "Shut Off Instance" and observe the behavior



Error: Unable to shut off instance: TEST-T1

Our test User is unable to shut down the instance

3. Log in to the DevUser account and attempt switch to the instances under luma-test
   Username: DevUser
   Password: password



Compute / Instances

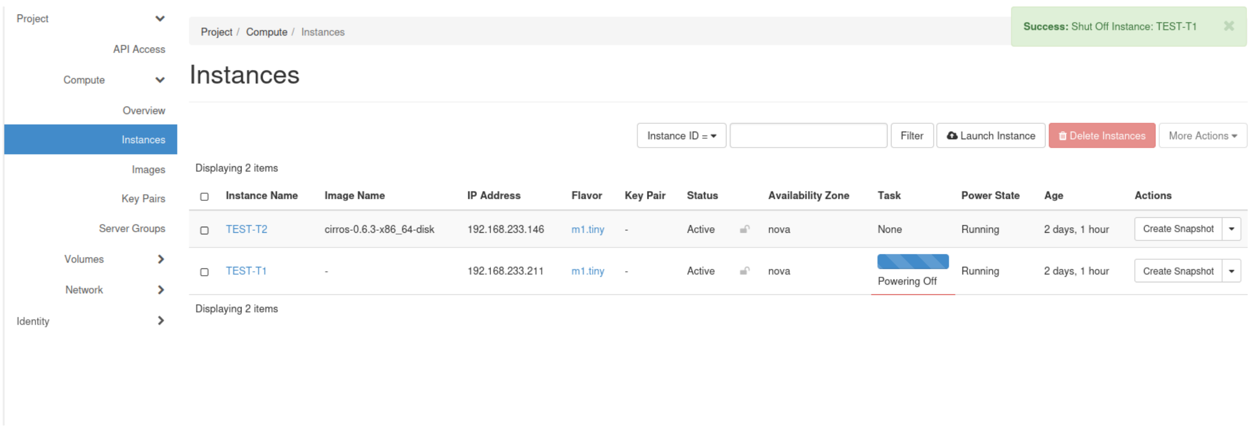
Instances

Displaying 2 items

Instance Name	Image Name	IP Address	Flavor	Key Pair	Status	Availability Zone	Task	Power State	Age	Actions
TEST-T2	cirros-0.6.3-x86_64-disk	192.168.233.146	m1.tiny	-	Active	nova	None	Running	2 days, 1 hour	Create Snapshot
TEST-T1	cirros-0.6.3-x86_64-disk	192.168.233.211	m1.tiny	-	Active	nova	None	Running	2 days, 1 hour	Create Snapshot

The dashboard under DevUser looks very much the same. Are there differences?

4. Attempt to shut down the same machine to determine if you run into the same error



Project / Compute / Instances

Success: Shut Off Instance: TEST-T1

Instances

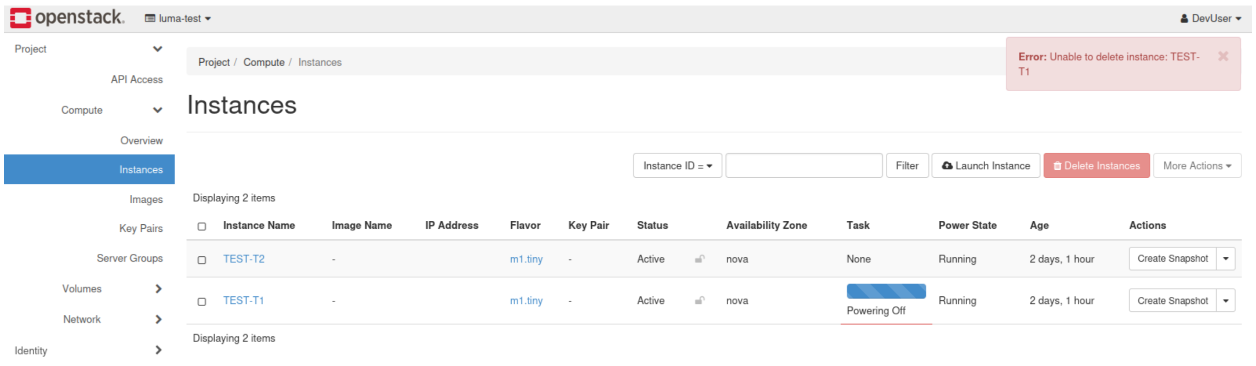
Displaying 2 items

Instance Name	Image Name	IP Address	Flavor	Key Pair	Status	Availability Zone	Task	Power State	Age	Actions
TEST-T2	cirros-0.6.3-x86_64-disk	192.168.233.146	m1.tiny	-	Active	nova	None	Running	2 days, 1 hour	Create Snapshot
TEST-T1	-	192.168.233.211	m1.tiny	-	Active	nova	Powering Off	Running	2 days, 1 hour	Create Snapshot

Displaying 2 items

The Dev user has no issue shutting down the instance.

5. While under the Dev user, attempt to delete an instance



Project / Compute / Instances

Error: Unable to delete instance: TEST-T1

Instances

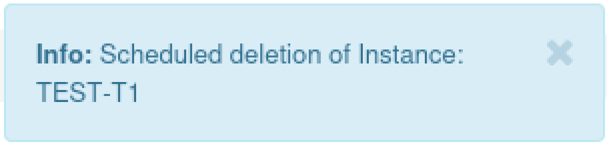
Displaying 2 items

Instance Name	Image Name	IP Address	Flavor	Key Pair	Status	Availability Zone	Task	Power State	Age	Actions
TEST-T2	-		m1.tiny	-	Active	nova	None	Running	2 days, 1 hour	Create Snapshot
TEST-T1	-		m1.tiny	-	Active	nova	Powering Off	Running	2 days, 1 hour	Create Snapshot

Displaying 2 items

Our dev account is unable to delete instances which saves us from data loss

6. Log into the Horizon portal as cloud admin and attempt to delete TEST-T1
   Username: CloudAdmin
   Password: Password



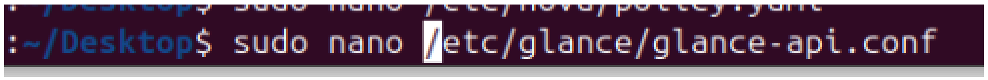
Info: Scheduled deletion of Instance: TEST-T1

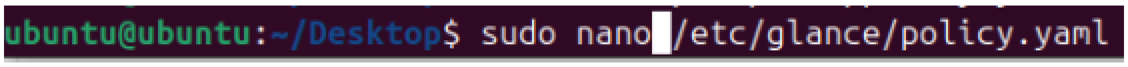
Following the same steps, TEST-T1 is successfully deleted

2.4 Overview

In this module our startup now has a base model of granular access control designed for how they interact with Virtual Machines. Lumatech can safely grant access to multiple test users and rest assured that they cannot cause any harm to the environment. The user accounts will maintain a very limited set of permissions and allow them to login to the instances but not disrupt operations.

Our developer group now has access to stop servers as needed for troubleshooting but they cannot delete them. This will be reserved for our admin group which will house all future system administrators going forward. As the company expands, more roles will be created, and additional permissions will be added as the scope of work demands.


```
:~/Desktop$ sudo nano /etc/glance/glance-api.conf
```


```
ubuntu@ubuntu:~/Desktop$ sudo nano /etc/glance/policy.yaml
```

Section 3: Multi-Factor Authentication (MFA with Keystone)

Your Implementation of MFA

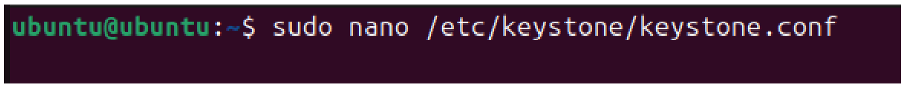
After evaluating LumaTech's existing authentication model, it was evident that password-only access posed a significant risk to the organization's cloud security posture. Despite the role-based access controls previously implemented, the lack of multi-factor authentication left user accounts vulnerable to credential theft, brute-force attacks, and unauthorized logins. This gap undermined the otherwise well-structured access framework and needed immediate remediation. [7] [8]

To strengthen authentication, this phase focuses on enabling Time-based One-Time Password (TOTP) multi-factor authentication within OpenStack Keystone. Users will register unique TOTP secrets and pair them with authenticator apps to generate time-sensitive codes. During login, both the standard password and a valid one-time code will be required, ensuring that a stolen password alone cannot grant system access. [9]

This enhancement establishes a critical second layer of defense that directly mitigates credential-based attacks. Once deployed, LumaTech's authentication process will provide strong verification of user identity, reinforce trust in administrative actions, and align the environment with modern cloud security standards.

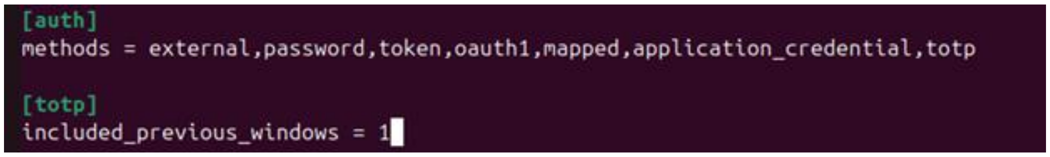
3.1 Configure Keystone for MFA Support [8]

Step 1: Edit Keystone configuration to enable MFA


```
ubuntu@ubuntu:~$ sudo nano /etc/keystone/keystone.conf
```

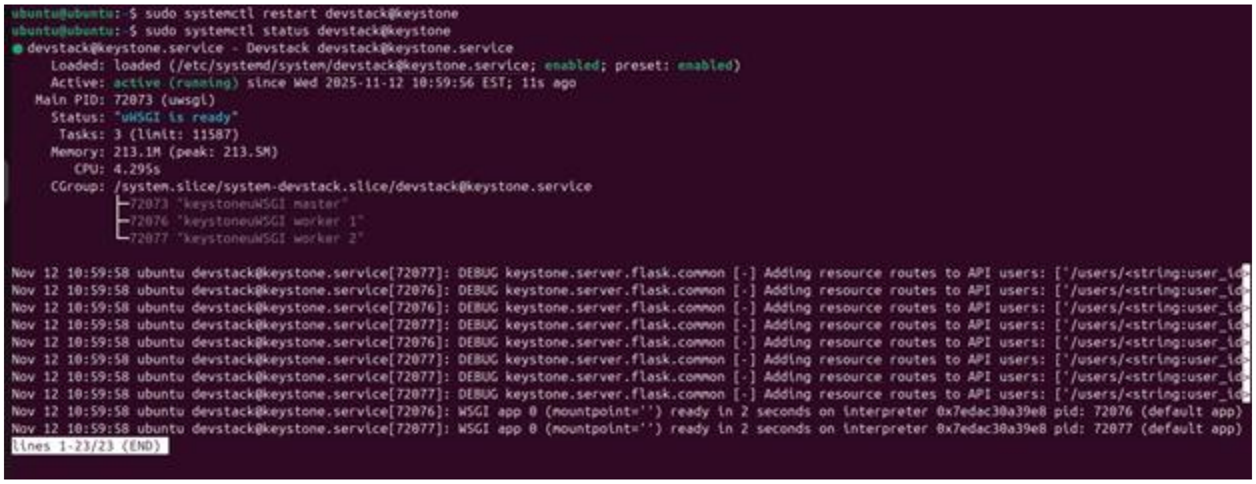
Step 2: Add MFA configuration sections

Add or modify the following sections in keystone.conf:



```
[auth]
methods = external,password,token,oauth1,mapped,application_credential,totp

[totp]
included_previous_windows = 1
```

Step 3: Restart Keystone service

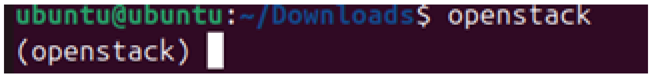


```
ubuntu@ubuntu:~$ sudo systemctl restart devstack@keystone
ubuntu@ubuntu:~$ sudo systemctl status devstack@keystone
● devstack@keystone.service - Devstack devstack@keystone.service
     Loaded: loaded (/etc/systemd/system/devstack@keystone.service; enabled; preset: enabled)
     Active: active (running) since Wed 2025-11-12 10:59:56 EST; 11s ago
   Main PID: 72073 (uwsgi)
     Status: "uWSGI is ready"
      Tasks: 3 (limit: 11587)
     Memory: 213.1M (peak: 213.5M)
        CPU: 4.295s
     CGroup: /system.slice/system-devstack.slice/devstack@keystone.service
             ├─72073 "keystoneuWSGI master"
             ├─72076 "keystoneuWSGI worker 1"
             └─72077 "keystoneuWSGI worker 2"

Nov 12 10:59:58 ubuntu devstack@keystone.service[72077]: DEBUG keystone.server.flask.common [-] Adding resource routes to API users: ['/users/<string:user_id>
Nov 12 10:59:58 ubuntu devstack@keystone.service[72076]: DEBUG keystone.server.flask.common [-] Adding resource routes to API users: ['/users/<string:user_id>
Nov 12 10:59:58 ubuntu devstack@keystone.service[72076]: DEBUG keystone.server.flask.common [-] Adding resource routes to API users: ['/users/<string:user_id>
Nov 12 10:59:58 ubuntu devstack@keystone.service[72077]: DEBUG keystone.server.flask.common [-] Adding resource routes to API users: ['/users/<string:user_id>
Nov 12 10:59:58 ubuntu devstack@keystone.service[72076]: DEBUG keystone.server.flask.common [-] Adding resource routes to API users: ['/users/<string:user_id>
Nov 12 10:59:58 ubuntu devstack@keystone.service[72077]: DEBUG keystone.server.flask.common [-] Adding resource routes to API users: ['/users/<string:user_id>
Nov 12 10:59:58 ubuntu devstack@keystone.service[72077]: DEBUG keystone.server.flask.common [-] Adding resource routes to API users: ['/users/<string:user_id>
Nov 12 10:59:58 ubuntu devstack@keystone.service[72077]: DEBUG keystone.server.flask.common [-] Adding resource routes to API users: ['/users/<string:user_id>
Nov 12 10:59:58 ubuntu devstack@keystone.service[72076]: WSGI app 0 (mountpoint='') ready in 2 seconds on interpreter 0x7edac30a39e8 pid: 72076 (default app)
Nov 12 10:59:58 ubuntu devstack@keystone.service[72077]: WSGI app 0 (mountpoint='') ready in 2 seconds on interpreter 0x7edac30a39e8 pid: 72077 (default app)
lines 1-23/23 (END)
```

Keystone configuration updated to support TOTP-based multi-factor authentication. The auth methods now include 'totp' and TOTP-specific settings are configured.

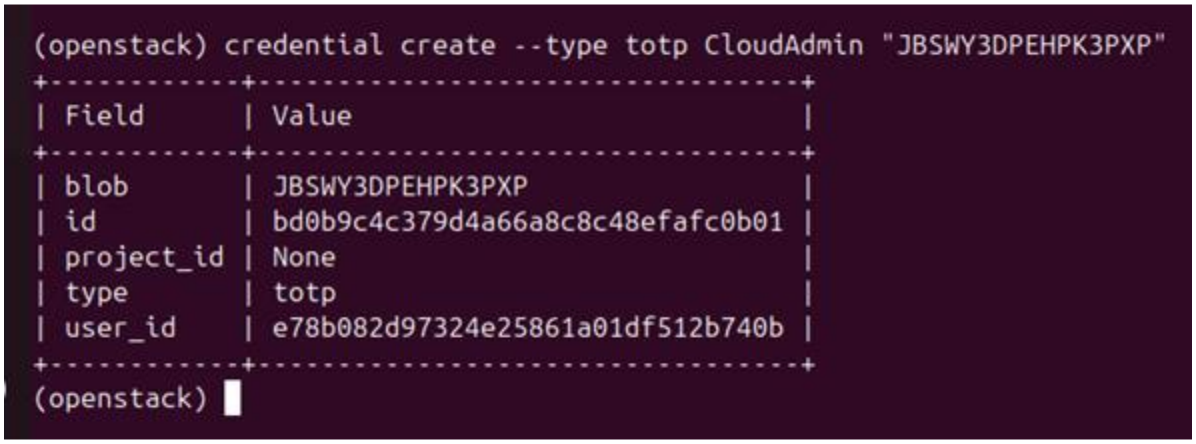
3.2 Register TOTP Secrets for Users [9]

Step 1: Source admin credentials and create TOTP credentials



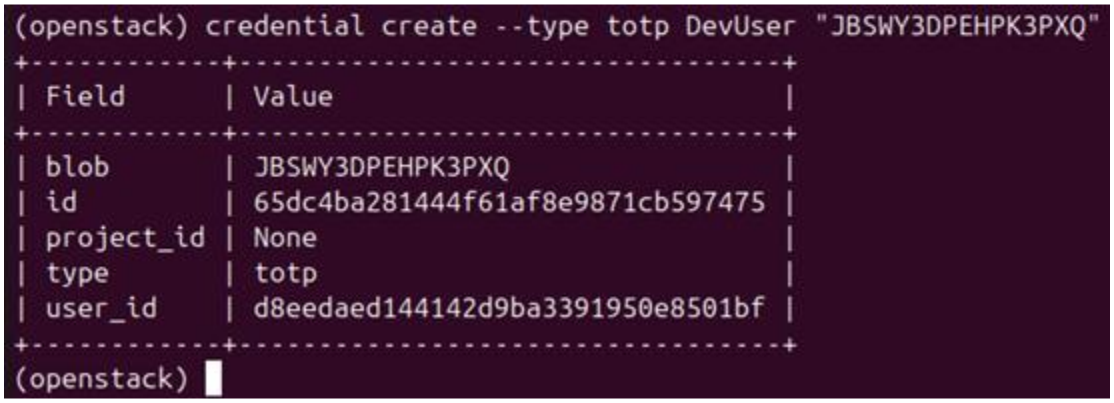
```
ubuntu@ubuntu:~/Downloads$ openstack
(openstack) 
```

Step 2: Create TOTP credentials for CloudAdmin



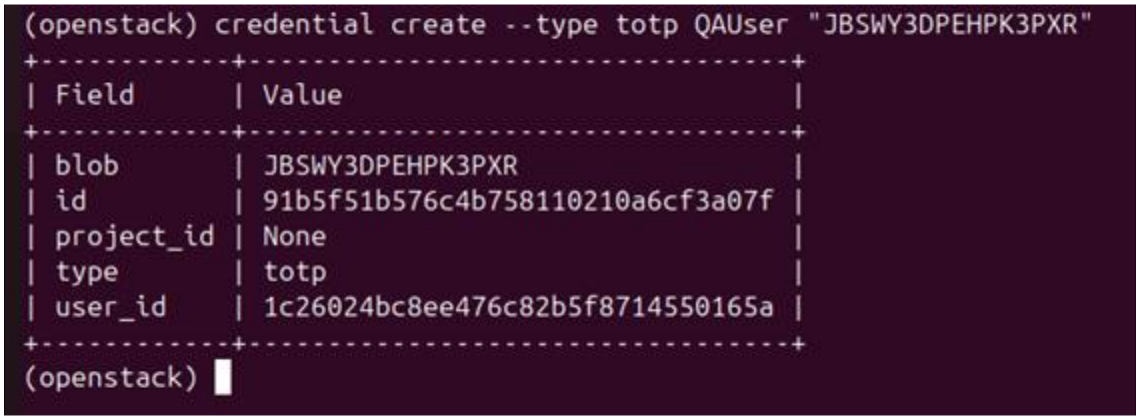
```
(openstack) credential create --type totp CloudAdmin "JBSWY3DPEHPK3PXP"
+------------+----------------------------------+
| Field      | Value                            |
+------------+----------------------------------+
| blob       | JBSWY3DPEHPK3PXP                 |
| id         | bd0b9c4c379d4a66a8c8c48efafc0b01 |
| project_id | None                             |
| type       | totp                             |
| user_id    | e78b082d97324e25861a01df512b740b |
+------------+----------------------------------+
(openstack) 
```

Step 3: Create TOTP credentials for DevUser



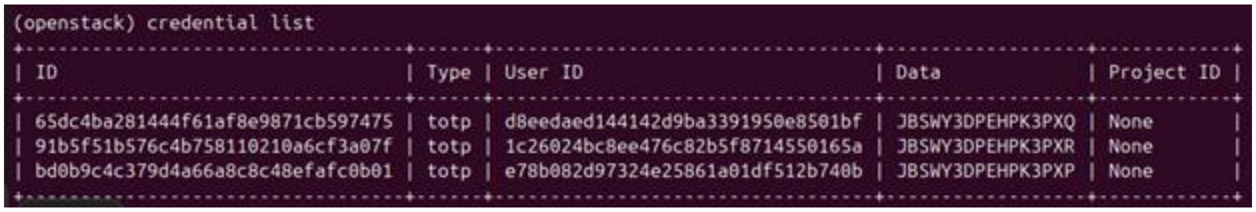
```
(openstack) credential create --type totp DevUser "JBSWY3DPEHPK3PXQ"
+------------+----------------------------------+
| Field      | Value                            |
+------------+----------------------------------+
| blob       | JBSWY3DPEHPK3PXQ                 |
| id         | 65dc4ba281444f61af8e9871cb597475 |
| project_id | None                             |
| type       | totp                             |
| user_id    | d8eedaed144142d9ba3391950e8501bf |
+------------+----------------------------------+
(openstack) 
```

Step 4: Create TOTP credentials for QAUser



```
(openstack) credential create --type totp QAUser "JBSWY3DPEHPK3PXR"
+------------+----------------------------------+
| Field      | Value                            |
+------------+----------------------------------+
| blob       | JBSWY3DPEHPK3PXR                 |
| id         | 91b5f51b576c4b758110210a6cf3a07f |
| project_id | None                             |
| type       | totp                             |
| user_id    | 1c26024bc8ee476c82b5f8714550165a |
+------------+----------------------------------+
(openstack) 
```

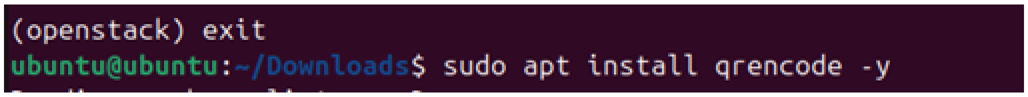
Step 5: Verify all credentials were created.



```
(openstack) credential list
+----------------------------------+------+----------------------------------+------------------+------------+
| ID                               | Type | User ID                          | Data             | Project ID |
+----------------------------------+------+----------------------------------+------------------+------------+
| 65dc4ba281444f61af8e9871cb597475 | totp | d8eedaed144142d9ba3391950e8501bf | JBSWY3DPEHPK3PXQ | None       |
| 91b5f51b576c4b758110210a6cf3a07f | totp | 1c26024bc8ee476c82b5f8714550165a | JBSWY3DPEHPK3PXR | None       |
| bd0b9c4c379d4a66a8c8c48efafc0b01 | totp | e78b082d97324e25861a01df512b740b | JBSWY3DPEHPK3PXP | None       |
+----------------------------------+------+----------------------------------+------------------+------------+
```

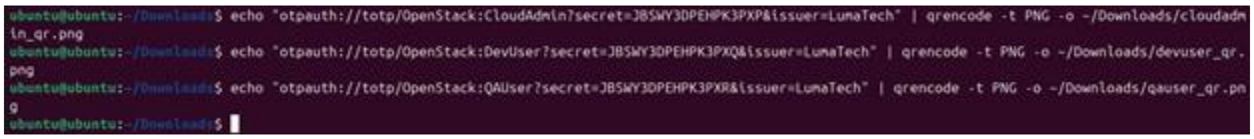
3.3 Generate QR Codes for Authenticator Apps

Step 1: Install QR code generation tool



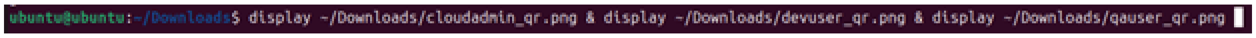
```
(openstack) exit
ubuntu@ubuntu:~/Downloads$ sudo apt install qrencode -y
```

Step 2: Generate QR codes for each user

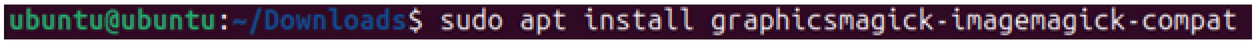


```
ubuntu@ubuntu:~/Downloads$ echo "otpauth://totp/OpenStack:CloudAdmin?secret=JBSWY3DPEHPK3PXP&issuer=LumaTech" | qrencode -t PNG -o ~/Downloads/cloudadm
in_qr.png
ubuntu@ubuntu:~/Downloads$ echo "otpauth://totp/OpenStack:DevUser?secret=JBSWY3DPEHPK3PXQ&issuer=LumaTech" | qrencode -t PNG -o ~/Downloads/devuser_qr.
png
ubuntu@ubuntu:~/Downloads$ echo "otpauth://totp/OpenStack:QAUser?secret=JBSWY3DPEHPK3PXR&issuer=LumaTech" | qrencode -t PNG -o ~/Downloads/qauser_qr.pn
g
ubuntu@ubuntu:~/Downloads$
```

Step 3: Display QR codes for each user



```
ubuntu@ubuntu:~/Downloads$ display ~/Downloads/cloudadmin_qr.png & display ~/Downloads/devuser_qr.png & display ~/Downloads/qauser_qr.png
```

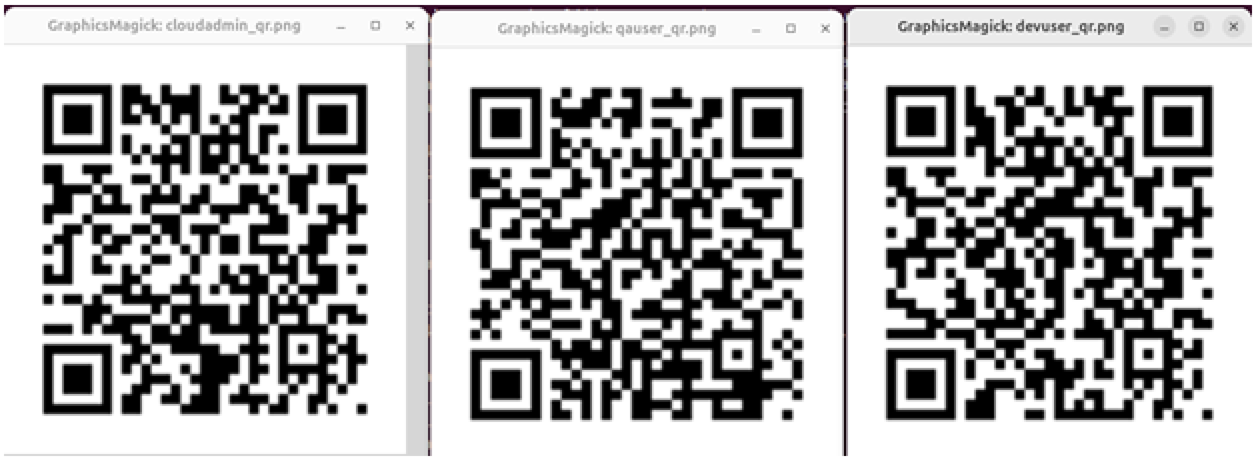


```
ubuntu@ubuntu:~/Downloads$ sudo apt install graphicsmagick-imagemagick-compat
```

Install Graphics Magick – Image Magick application

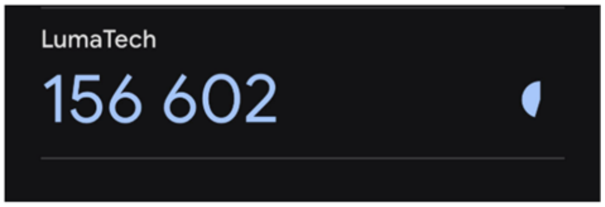
Step 4: Register QR codes in authenticator app

1. Open Google Authenticator or Authy on your smartphone
2. Scan the QR code for CloudAdmin
3. Verify the 6-digit TOTP code appears in your app
4. Note the account name shows as "OpenStack:CloudAdmin"


GraphicsMagick: cloudadmin_qr.png
GraphicsMagick: qauser_qr.png
GraphicsMagick: devuser_qr.png

QR codes for the three users

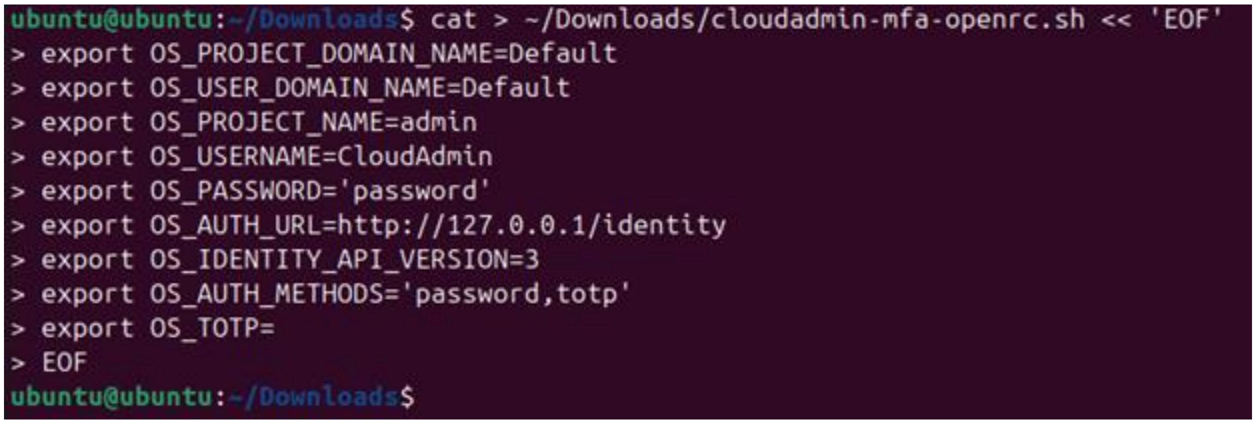
2. Authenticator app with registered TOTP accounts



Google Authenticator app showing registered OpenStack accounts with rotating TOTP codes for CloudAdmin

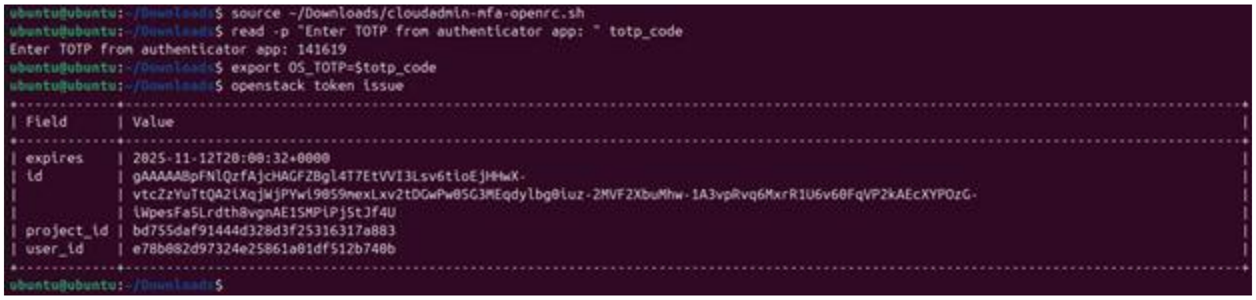
3.4 Test MFA Authentication

Step 1: Create RC files with MFA authentication



```
ubuntu@ubuntu:~/Downloads$ cat > ~/Downloads/cloudadmin-mfa-openrc.sh << 'EOF'
> export OS_PROJECT_DOMAIN_NAME=Default
> export OS_USER_DOMAIN_NAME=Default
> export OS_PROJECT_NAME=admin
> export OS_USERNAME=CloudAdmin
> export OS_PASSWORD='password'
> export OS_AUTH_URL=http://127.0.0.1/identity
> export OS_IDENTITY_API_VERSION=3
> export OS_AUTH_METHODS='password,totp'
> export OS_TOTP=
> EOF
ubuntu@ubuntu:~/Downloads$
```

Step 2: Test MFA login via CLI



```
ubuntu@ubuntu:~/Downloads$ source ~/Downloads/cloudadmin-mfa-openrc.sh
ubuntu@ubuntu:~/Downloads$ read -p "Enter TOTP from authenticator app: " totp_code
Enter TOTP from authenticator app: 141619
ubuntu@ubuntu:~/Downloads$ export OS_TOTP=$totp_code
ubuntu@ubuntu:~/Downloads$ openstack token issue
+------------+-------------------------------------------------------------------------------------------------+
| Field      | Value                                                                                           |
+------------+-------------------------------------------------------------------------------------------------+
| expires    | 2025-11-12T20:00:32+0000                                                                        |
| id         | gAAAAABpFNlQzfAjcHAGFZBgl4T7EtVVI3Lsv6tioEjHHwX-                                                |
|            | vtcZzYuTtQA2iXqjWjPYwi9059mexLxv2tDGwPw05G3MEqdylbg0iuz-2MVF2XbuMhw-1A3vpRvq6MxrR1U6v60FqVP2kAEcXYPOzG- |
|            | iWpesFa5Lrdth8vgnAE1SMPiPj5tJf4U                                                                |
| project_id | bd755daf91444d328d3f25316317a883                                                                |
| user_id    | e78b082d97324e25861a01df512b740b                                                                |
+------------+-------------------------------------------------------------------------------------------------+
ubuntu@ubuntu:~/Downloads$
```

Successful OpenStack CLI authentication using CloudAdmin account with TOTP multi-factor authentication. The token was issued only after providing both password and valid TOTP code.

3.5 Outcome

Summary of MFA Implementation:

- Keystone configured to support TOTP-based multi-factor authentication
- TOTP secrets registered for all user accounts (CloudAdmin, DevUser, QAUser)
- QR codes generated and registered in authenticator applications
- MFA login tested via both CLI and web interface
- Authentication failures verified for invalid TOTP codes

Security Benefits:

1. Defense Against Credential Theft: Even if passwords are compromised, attackers cannot access the system without the second factor
2. Compliance: Meets regulatory requirements for strong authentication in cloud environments
3. Audit Trail: MFA events are logged, providing detailed authentication records
4. User Accountability: Individual TOTP secrets ensure non-repudiation of user actions
5. Risk Mitigation: Significantly reduces the attack surface for unauthorized access

References

[1] OpenStack Foundation, “OpenStack Security Guide,” OpenStack Documentation, Latest ed., 2025. [Online]. Available: https://docs.openstack.org/security-guide/. [Accessed: Nov. 11, 2025].

[2] OpenStack Foundation, “OpenStackClient Command List – Identity (Keystone),” OpenStack Docs, Latest ed., 2025. [Online]. Available: https://docs.openstack.org/python-openstackclient/latest/cli/command-list.html. [Accessed: Nov. 11, 2025].

[3] OpenStack Foundation, “Sample policy.yaml Configuration — Nova,” OpenStack Documentation, Latest ed., 2025. [Online]. Available: https://docs.openstack.org/nova/latest/admin/configuration/samples/policy.yaml.htm l. [Accessed: Nov. 11, 2025].

[4] OpenStack Foundation, “Multi-Factor Authentication (MFA) in Keystone,” OpenStack Identity Service Documentation, Latest ed., 2025. [Online]. Available: https://docs.openstack.org/keystone/latest/admin/mfa.html. [Accessed: Nov. 11, 2025].

[5] National Institute of Standards and Technology (NIST), “Security and Privacy Controls for Information Systems and Organizations,” Special Publication 800-53 Rev. 5, 2020. [Online]. Available: https://doi.org/10.6028/NIST.SP.800-53r5. [Accessed: Nov. 11, 2025].

[6] National Institute of Standards and Technology (NIST), “Framework for Improving Critical Infrastructure Cybersecurity,” Version 1.1, 2018. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/CSWP/NIST.CSWP.04162018.pdf. [Accessed: Nov. 11, 2025].

[7] Cybersecurity and Infrastructure Security Agency (CISA), “Multifactor Authentication,” *CISA*, 2025. [Online]. Available: https://www.cisa.gov/topics/cybersecurity-best-practices/multifactor-authentication. [Accessed: Nov. 12, 2025].

[8] OpenStack Foundation, “Multi-Factor Authentication — Keystone,” *OpenStack Documentation*, ver. 28.1.0, Jan. 2019. [Online]. Available: https://docs.openstack.org/keystone/latest/user/multi-factor-authentication.html. [Accessed: Nov. 12, 2025].

[9] OpenStack Foundation, “Time-based One-time Password (TOTP),” *OpenStack Documentation*, Mar. 2022. [Online]. Available:

https://docs.openstack.org/keystone/latest/admin/auth-totp.html. [Accessed: Nov. 12, 2025].