**Server Side:**

How to configure MySQL server to start automatically on boot?

sudo update-rc.d mysql defaults

How to check and display the status of the MySQL service?

sudo systemctl status mysql

**Task 1: Advanced User and Group Configuration**

**1.1 Nested Group Creation:**

How to create a group?

sudo groupadd group\_name

How to assign a user to a group?

sudo usermod -aG group\_name username

//-aG: append the user to a group without removing them from other groups

What is sudoers group?

-We assign a user to sudoers group to allow them to execute commands with root privileges using sudo command

How to assign a user to a sudoers group?

sudo usermod -aG sudo username

How to list the groups:

cat /etc/group

How to check user’s groups:

groups username

How to delete a group?

sudo groupdel group\_name

**1.2 Shared Directories with ACLs:**

What is the difference between chmod and ACL (Access control list)?

-chmod is only able to give permission to user owner, group owner, and others, while ACL is able to give permission to special users or special groups.

To set ACL for a file or directory:

setfacl -m u:name:permission file\_name (we can change the ‘u’ to ‘g’ to set for groups)

-m: to modify or add ACL

u: user

name: user or group name

permission: r for read, w for write, x for execute

To view the ACL for a file or directory:  
getfacl file\_name

**Task 2: Security Hardening:**

**2.1 SSH Key-Based Authentication:**

How to disable password authentication?

1. Open the SSH configuration file: sudo nano /etc/ssh/sshd\_config
2. Disable password authentication for the user by uncommenting PasswordAuthentication and PubkeyAuthentication and set their values to PasswordAuthentication no, PubkeyAuthentication yes
3. Save changes and reload the SSH using: sudo systemctl restart sshd

Why do we disable password authentication?

**1. Prevent Brute Force Attacks**

* Password-based authentication is vulnerable to brute force attacks, where an attacker repeatedly tries different password combinations.

**2. Enforce Secure Authentication Methods**

* Forcing the user to use SSH keys, which are cryptographically stronger than passwords.

**3. Restrict Access for Specific Users**

* If some users require SSH access while others don’t, you can control who can log in and how they authenticate.

**4. Prevent Unauthorized Access**

* If an attacker steals a user’s password, they still can’t access the server if password authentication is disabled.

How ssh works?

When we generate ssh key, 2 types of keys are generated, Public key and Private key, the Private key cant be shared, it stays inside the device itself, while the Public key can be shared to everyone ( will be shared to the server). So lets say I want to access the Server\_VM using Client\_VM, I have to log in Client\_VM and generate an ssh key (Public key and Private key) then share Client\_VM public key to the server, so whenever Client\_VM want to access the Server\_VM, the Server\_VM will send a random message to Client\_VM, then the Client\_VM encrypt this message using his Private key and send it to the Server\_VM, the Server\_VM will verify the authenticator of the message using the stored public key, if valid then the authentication is successful and now Client\_VM can access the Server\_VM.

How to generate and deploy SSH keys for dev\_lead1 to access VM1?

Log in using Client\_VM and generate ssh key using: ssh-keygen -t rsa -b 4096

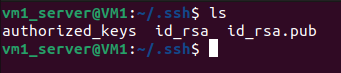
-t rsa: specify the RSA Algorithm

-b 4096: Sets the key size to 4096 bits for stronger security.

After generating the key, it will ask where to save the key 🡪 Don’t change anything, save it in the default file.

Then it will ask to set passphrase (Optional) to protect it using password 🡪 you can skip it

Once you are done, the private and public keys will be generated and saved inside /home/user/.ssh



Authorized\_keys: stores the **public keys** of users who are allowed to log in via SSH key authentication.

id\_rsa: stores the private key of the user

id\_rsa.pub: stores the public key of the user which will be shared

Now we have to share the public key of the Client\_VM to the Server\_VM, we have many ways to share it but the best one using the ssh-copy-id command. How?

In the Client\_VM, write: ssh-copy-id dev\_lead1@Server\_IP\_ADDRESS

Server\_IP\_ADDRESS: the remote host that you would like to connect to

Dev\_lead1: the account where your public key will be shared to

* To know the Server IP you can use the command: hostname -I or ip a

To connect to the user using SSH:

Inside Client\_VM write: ssh dev\_lead1@Server\_IP\_ADDRESS

* For the first time, your local device doesn’t recognize the remote host, so it will prompt whether you want to continue or not: type yes and enter

**For more details: take a look at** [**How to Create an SSH Key in Linux: Easy Step-by-Step Guide | DigitalOcean**](https://www.digitalocean.com/community/tutorials/how-to-configure-ssh-key-based-authentication-on-a-linux-server#step-2-copying-an-ssh-public-key-to-your-server)

* 1. **Automatic Security Updates:**

How to enable automatic security updates using unattended-upgrade?

* Log in to the server and install the package using : sudo apt-get install unattended-upgrades -y
* Also install the update dash notifier common package for automatic reboots using sudo apt install update-notifier-common -y
* Edit the 50 unattended dash upgrade file using:

sudo nano /etc/apt/apt.conf.d/50unattended-upgrades

* Remove the // and change the false to true from the line that contains: Unattended-Upgrade::Automatic-Reboot “false”;
* Add the following line to log updates:

Unattended-Upgrade::LogFile "/var/log/security\_updates.log";

Issue the command Cat /var/run/reboot-required to see if a reboot is required, if yes, reboot the machine and that’s it

* 1. **Configure Message of the Day (MOTD):**
* Open the motd file using the command: Sudo nano /etc/motd
* Add the custom message “Welcome to the Ubuntu administration Lab”

**Task 3: System Monitoring & Automation:**

**3.1** **Metric Collection Script:**

**3.2 File Activity Monitor:**

**Task 4: MySQL User Management and Database Exploration:**

After installing the mysql-server, you can open it using:  
mysql -u root -p

It will ask for password, by default it will not let you in unless you rewrite the code using sudo:

Sudo mysql -u root -p

To change the password of the user:

ALTER USER 'root'@'localhost' IDENTIFIED BY 'NewPassword';

Then exit

IDENTIFIED BY: ASSIGN PASSWORD TO THE USER

To show all the databases stored:

show databases;

If you want to use a certain database:

Use DATABASE\_NAME;

To show all the tables inside a database:

show tables;

To create a new database:

Create database DATABASE\_NAME

To create a table inside a database:

Similar commands to SQL-Developer (we can use similar syntax for queries, updating, deleting ….)

**4.1 MySQL User Creation:**

**4.2 User Authentication and Verification:**

**4.3 Database and Table Exploration:**

**4.4 Logging and Audit Trail:**

**Task 5: Network Connectivity Monitoring:**

**Client Side:**

**Task 1: Development Team (VM2):**

**1.1 Login Attempt Monitoring:**

**1.2 Permission Cleanup Script:**

**Task 2: Operations Team (VM3):**

**2.1 Resource Reporting Script:**

**2.2 Quota Enforcement:**