MySQL Enterprise Edition - Implementation Essentials Bootcamp | Introduction

Introduction

About this Workshop

Securing the data stored in your MySQL Server is key towards avoiding breaches and achieving Regulatory Compliance. This workshop covers the installation, configuration and testing of 4 of the MySQL Security Based Enterprise Features. We will go through how to setup and run Enterprise Audit, Enterprise Transparent Data Encryption, Enterprise FireWall and Enterprise Data Masking.

Estimated Workshop Time: 1 hours 30 minutes (This estimate is for the entire workshop - it is the sum of the estimates provided for each of the labs included in the workshop.)

Your Free Tier server should be accessible for a couple of days after this workshop so what you do not finish when following instruction will be able to be covered later at your own pace.

Prerequisite

For working on this workshop you will need a clean computer to run on. We should have provided you with a link on setting up a free account on Oracle Cloud Infrastructure (OCI).

If you have not set it up, then please go through the following steps:

Introduction

About this Workshop

Securing the data stored in your MySQL Server is key towards avoiding breaches and achieving Regulatory Compliance. This workshop covers the installation, configuration and testing of 4 of the MySQL Security Based Enterprise Features. We will go through how to setup and run Enterprise Audit, Enterprise Transparent Data Encryption, Enterprise FireWall and Enterprise Data Masking.

Estimated Workshop Time: 1 hours 30 minutes (This estimate is for the entire workshop - it is the sum of the estimates provided for each of the labs included in the workshop.)

Your Free Tier server should be accessible for a couple of days after this workshop so what you do not finish when following instruction will be able to be covered later at your own pace.

Prerequisite

For working on this workshop you will need a clean computer to run on. We should have provided you with a link on setting up a free account on Oracle Cloud Infrastructure (OCI).

If you have not set it up, then please go through the following steps:

- OCI Free Tier Setup

Objectives

In this workshop, you will learn how to work with

- MySQL Enterprise Edition
- MySQL Shell

MySQL Enterprise Edition - Implementation Essentials Bootcamp | Lab 1: Create Virtual Cloud Network and Related Components

Create your Virtual Cloud Network and Related Components

Introduction

Create your VCN and subnets

Set up a Virtual Cloud Network (VCN) to connect your Linux instance to the internet. You will configure all the components needed to create your virtual network.

Estimated Time: 10 minutes

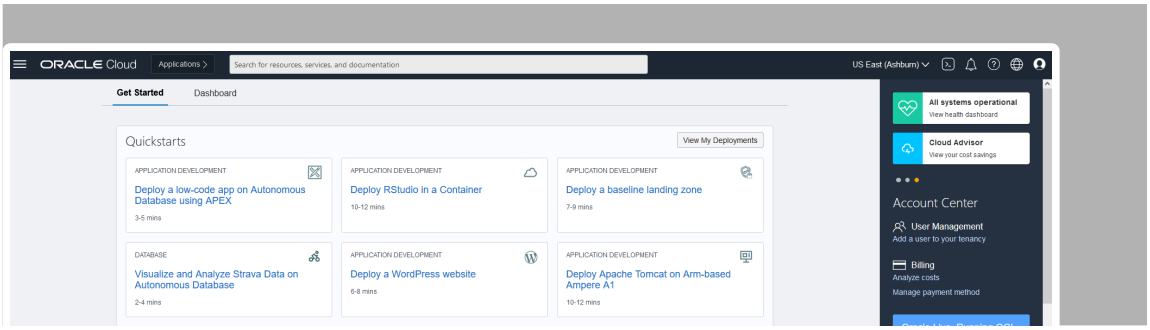
Objectives

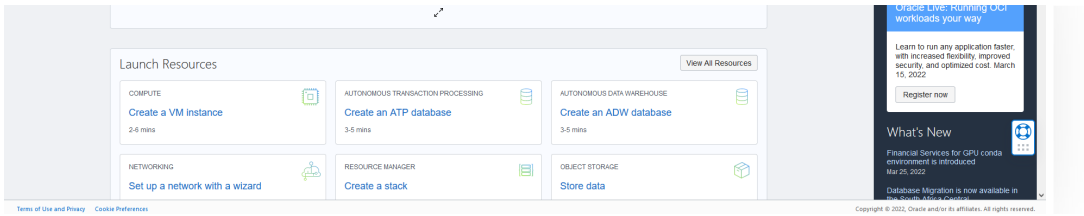
In this lab, you will be guided through the following tasks:

- Create Virtual Cloud Network
- Configure security list to allow MySQL incoming connections

Prerequisites

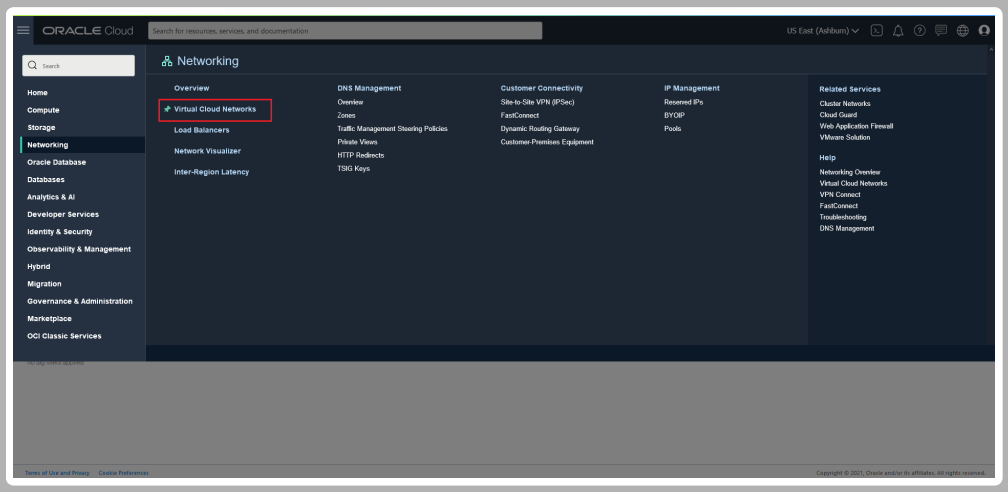
- An Oracle Free Tier or Paid Cloud Account
- A web browser
- Login to OCI to land on OCI Dashboard (This image shows a trial account)



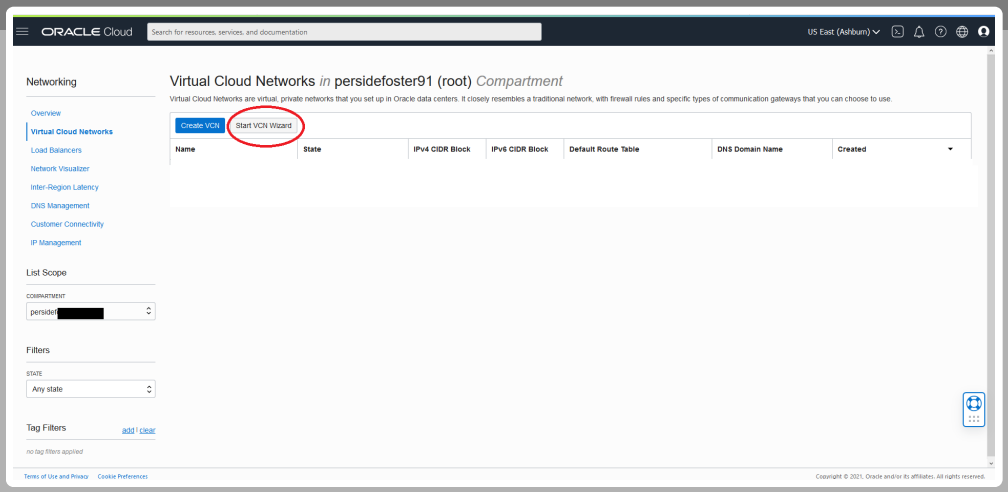


Task 1: Create Virtual Cloud Network

1. Click Navigation Menu
   Select Networking
   Select Virtual Cloud Networks

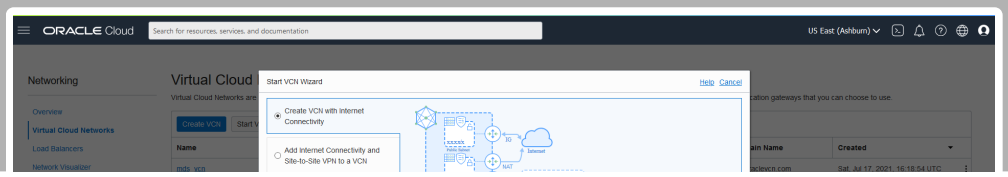


2. Click **Start VCN Wizard**



3. Select 'Create VCN with Internet Connectivity'

   Click 'Start VCN Wizard'



4. Create a VCN with Internet Connectivity

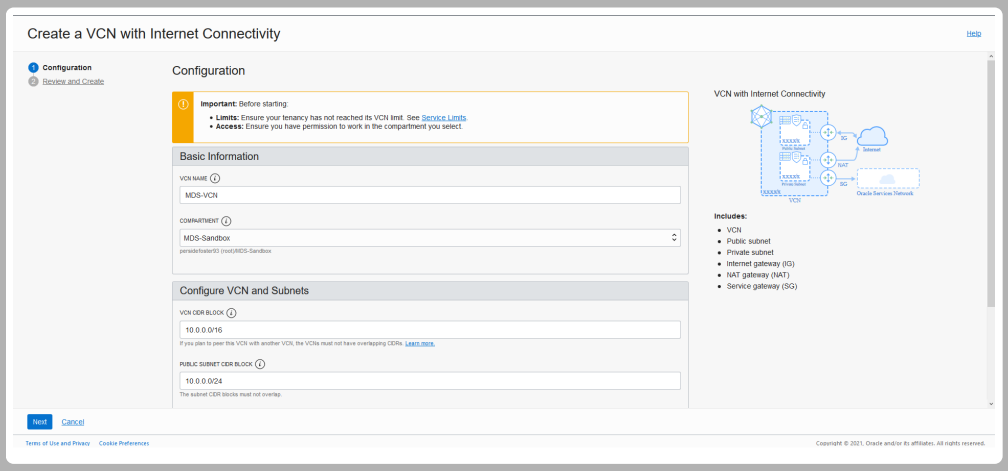
   On Basic Information, complete the following fields:

   VCN Name:

   ```
   myvcn
   ```

   Compartment: Select **(root)**

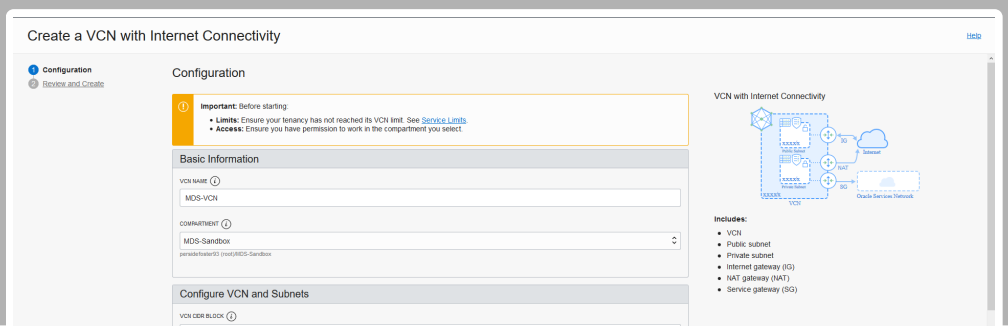
   Your screen should look similar to the following

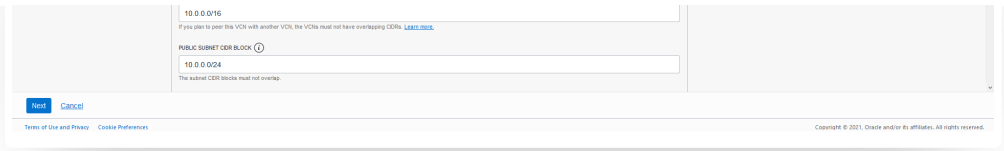


5. Click 'Next' at the bottom of the screen

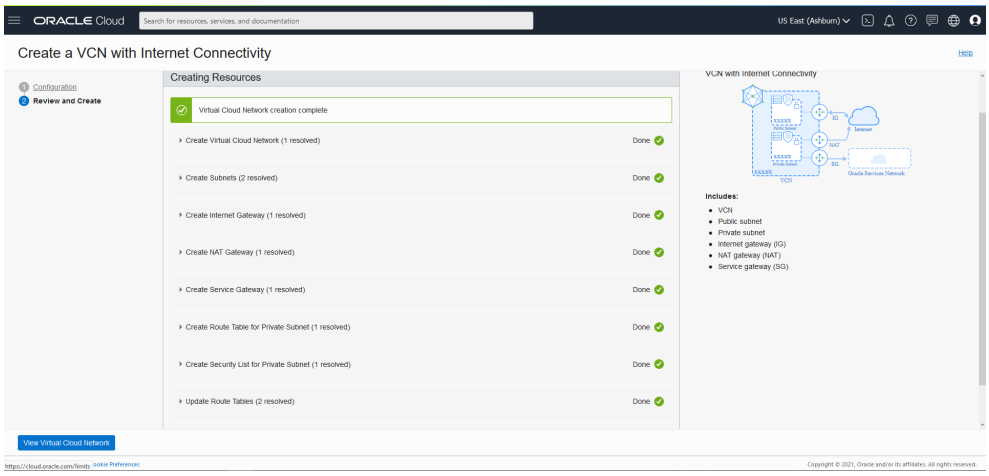
6. Review Oracle Virtual Cloud Network (VCN), Subnets, and Gateways

   Click 'Create' to create the VCN

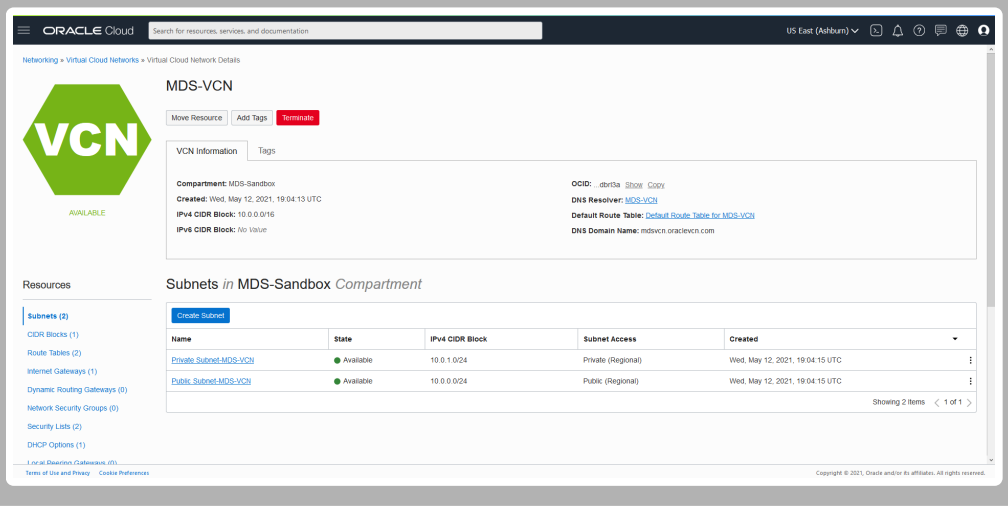




7. The Virtual Cloud Network creation is completing

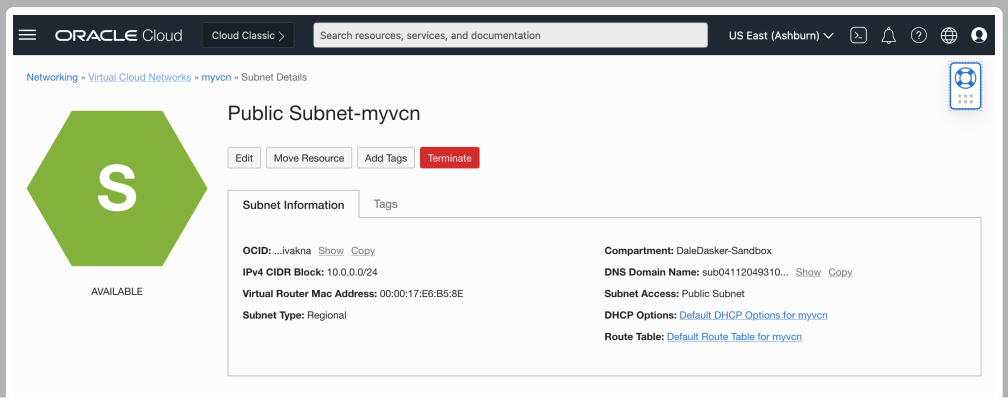


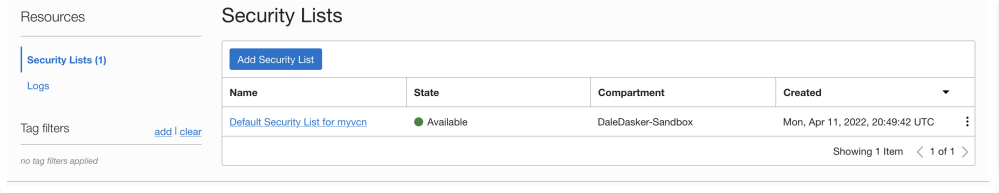
8. Click 'View Virtual Cloud Network' to display the created VCN



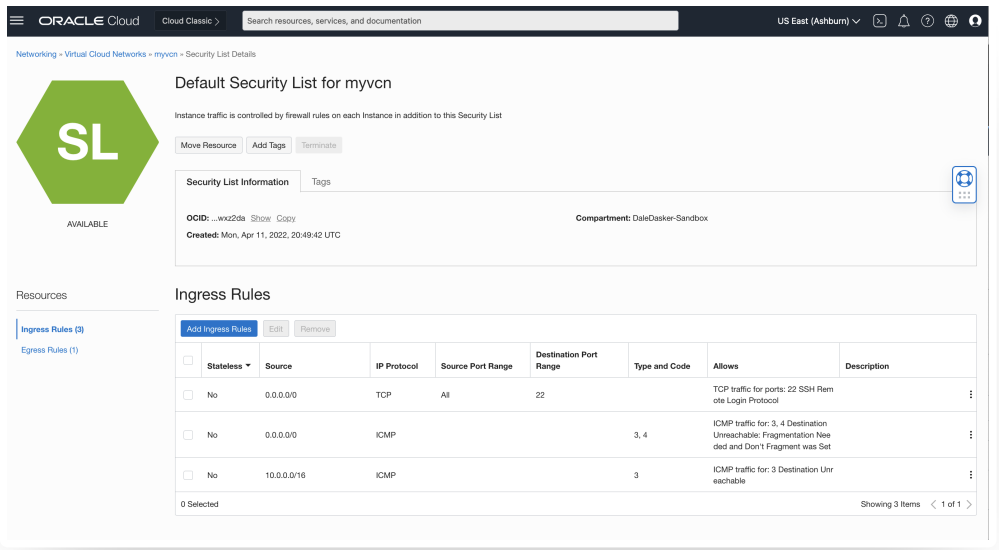
Task 2: Configure security list to allow MySQL incoming connections

1. On myvcn page under 'Subnets in (root) Compartment', click '**Public Subnet-myvcn**'

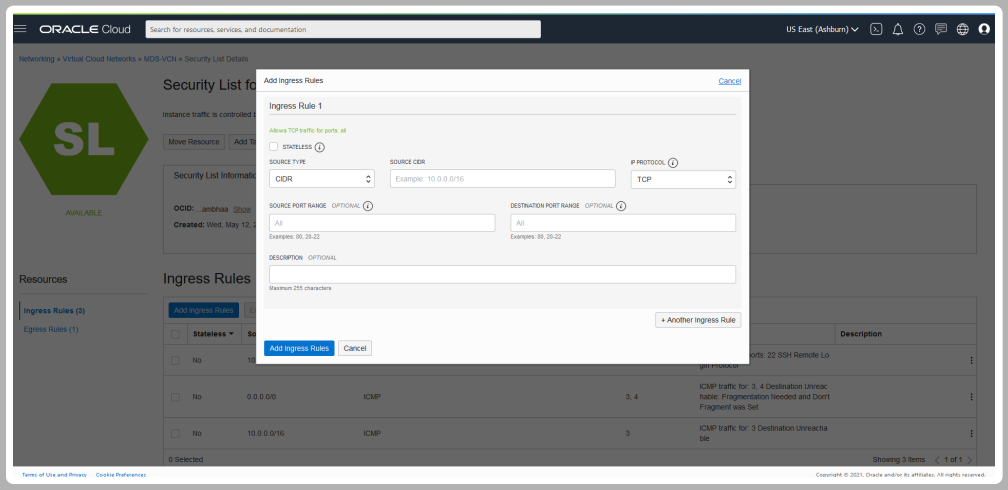




2. On Public Subnet-myvcn page under 'Security Lists', click '**Security List for Public Subnet-myvcn**'



3. On Security List for Public Subnet-myvcn page under 'Ingress Rules', click '**Add Ingress Rules**'



4. On Add Ingress Rules page under Ingress Rule 1

   Add an Ingress Rule with Source CIDR

```
0.0.0.0/0
```

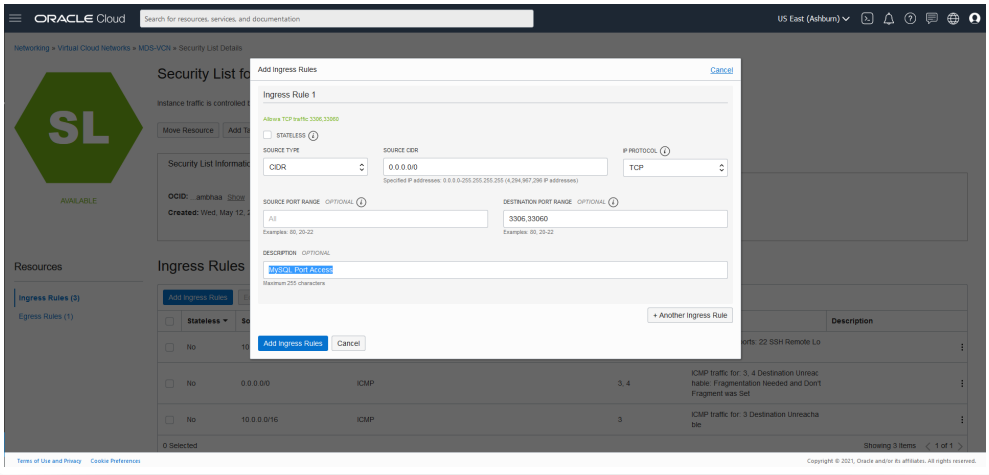
   Destination Port Range

```
3306,33060
```

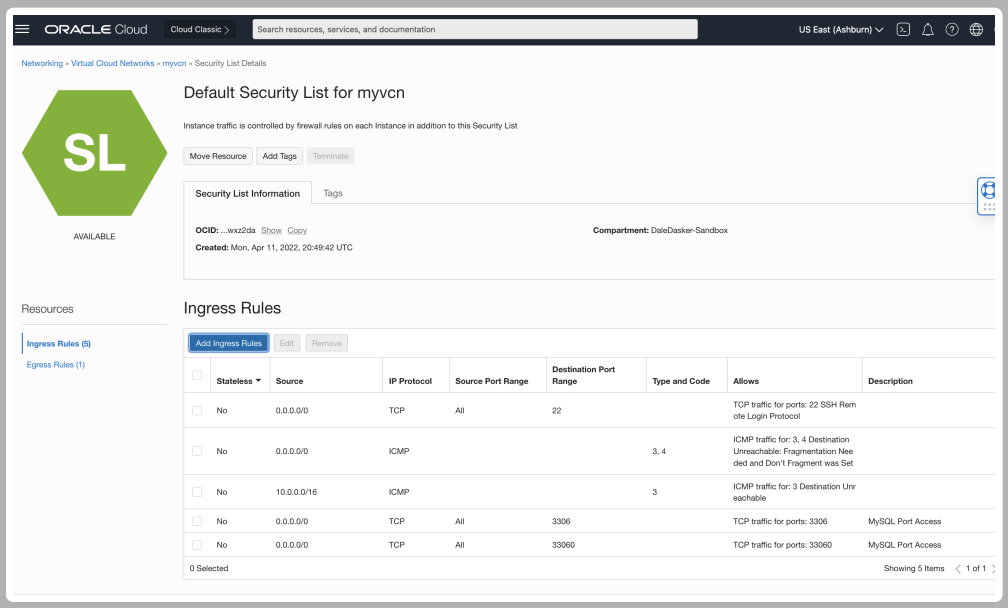
Description

```
MySQL Port Access
```

Click 'Add Ingress Rule'



5. On Security List for Public Subnet-myvcn page, the new Ingress Rules will be shown under the Ingress Rules List



You may now proceed to the next lab

Acknowledgements

MySQL Enterprise Edition - Implementation Essentials Bootcamp | Lab 2: Create Linux Compute Instance

Create Linux Compute Instance

Introduction

Oracle Cloud Infrastructure Compute lets you provision and manage compute hosts, known as instances . You can create instances as needed to meet your compute and application requirements. After you create an instance, you can access it securely from your computer or cloud shell.

Create Linux Compute Instance

In this lab, you use Oracle Cloud Infrastructure to create an Oracle Linux instance.

Estimated Time: 10 minutes

Objectives

In this lab, you will be guided through the following tasks:

- Create SSH Key on OCI Cloud
- Create Compute Instance

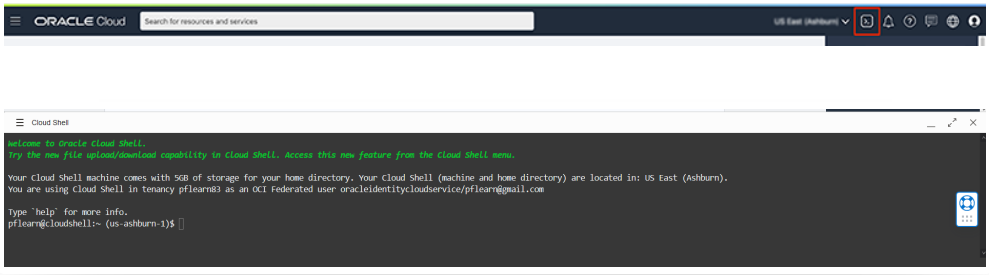
Prerequisites

- An Oracle Free Tier or Paid Cloud Account
- A web browser
- Should have completed Lab 1

Task 1: Create SSH Key on OCI Cloud Shell

The Cloud Shell machine is a small virtual machine running a Bash shell which you access through the Oracle Cloud Console (Homepage). You will start the Cloud Shell and generate a SSH Key to use for the Bastion session.

1. To start the Oracle Cloud shell, go to your Cloud console and click the cloud shell icon at the top right of the page. This will open the Cloud Shell in the browser, the first time it takes some time to generate it.



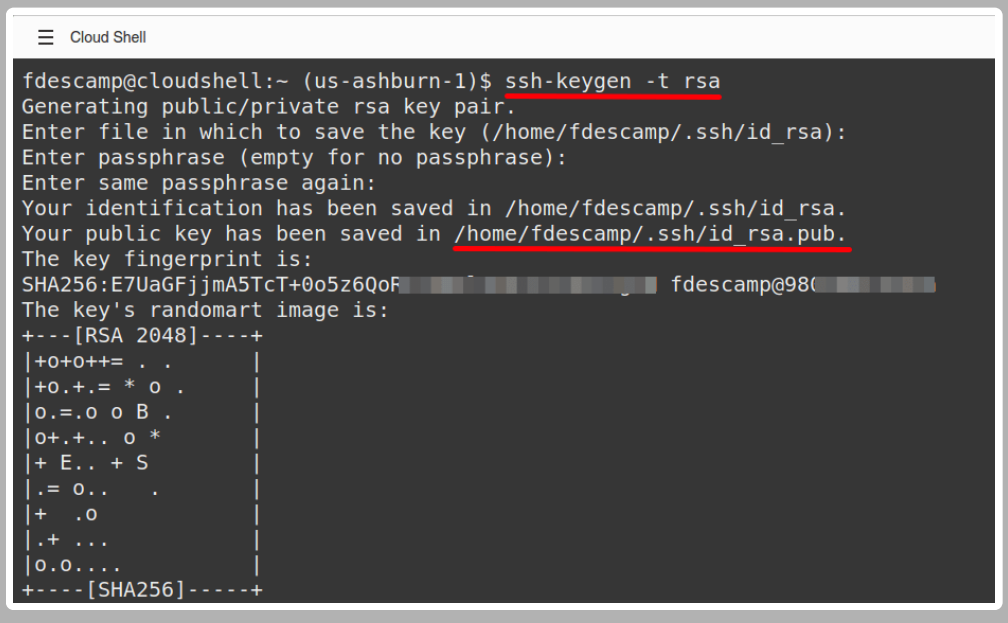
Note: You can use the icons in the upper right corner of the Cloud Shell window to minimize, maximize, restart, and close your Cloud Shell session.

2. Once the cloud shell has started, create the SSH Key using the following command:

```
ssh-keygen -t rsa
```

Press enter for each question.

Here is what it should look like.



3. The public and private SSH keys are stored in ~/.ssh/id_rsa.pub.

4. Examine the two files that you just created.

```
cd .ssh
```

```
ls
```

Note: in the output there are two files, a *private key:* `id_rsa` and a *public key:* `id_rsa.pub`. Keep the private key safe and don't share its content with anyone. The public key will be needed for various activities and can be uploaded to certain systems as well as copied and pasted to facilitate secure communications in the cloud.

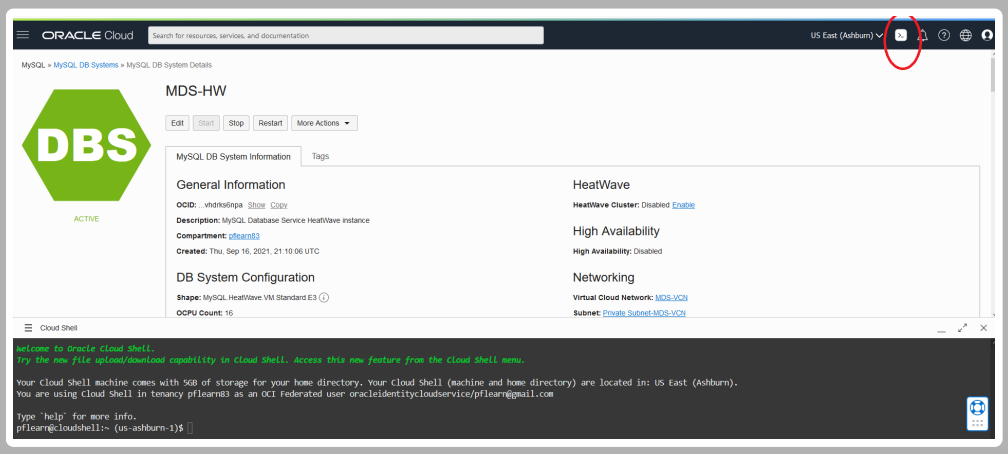
Task 2: Create Compute instance

You will need a compute Instance to connect to your brand new MySQL database.

1. Before creating the Compute instance open a notepad

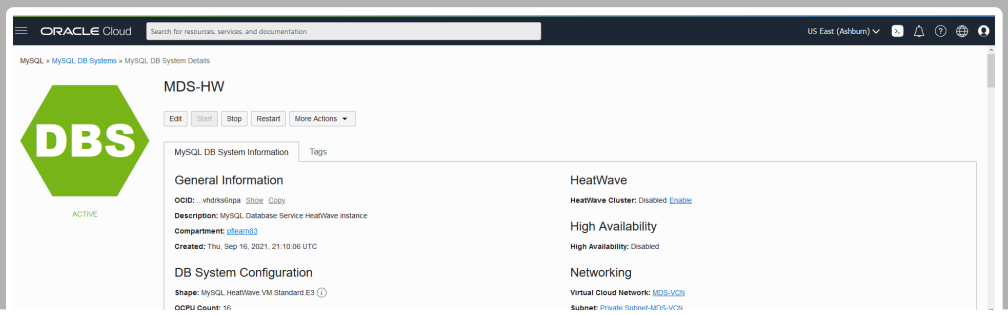
2. Do the followings steps to copy the public SSH key to the notepad

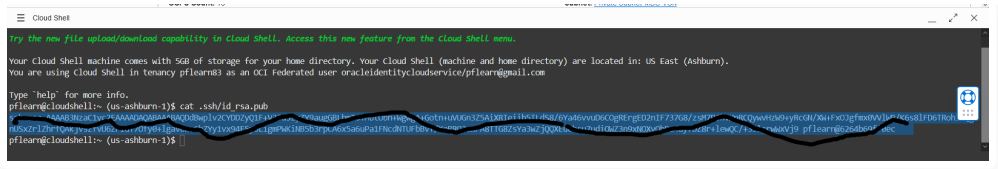
   Open the Cloud shell



   Enter the following command

```
cat ~/.ssh/id_rsa.pub
```





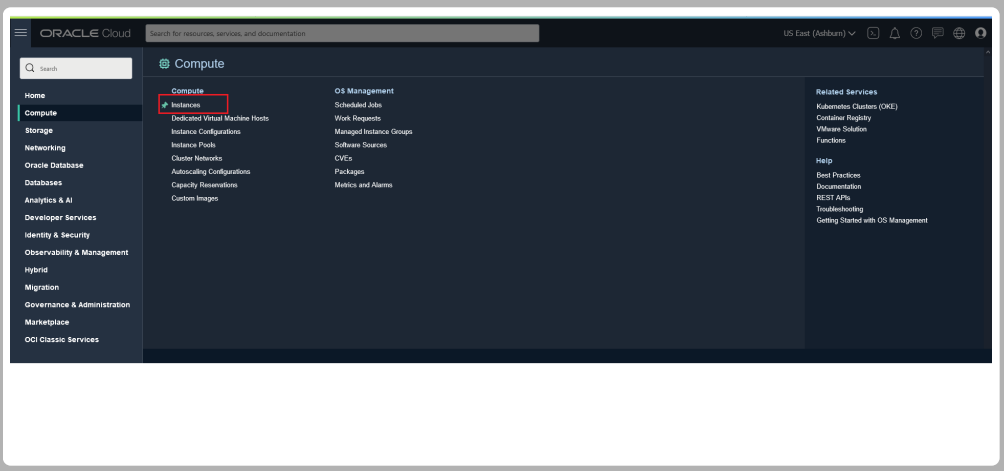
3. Copy the id_rsa.pub content the notepad

   Your notepad should look like this

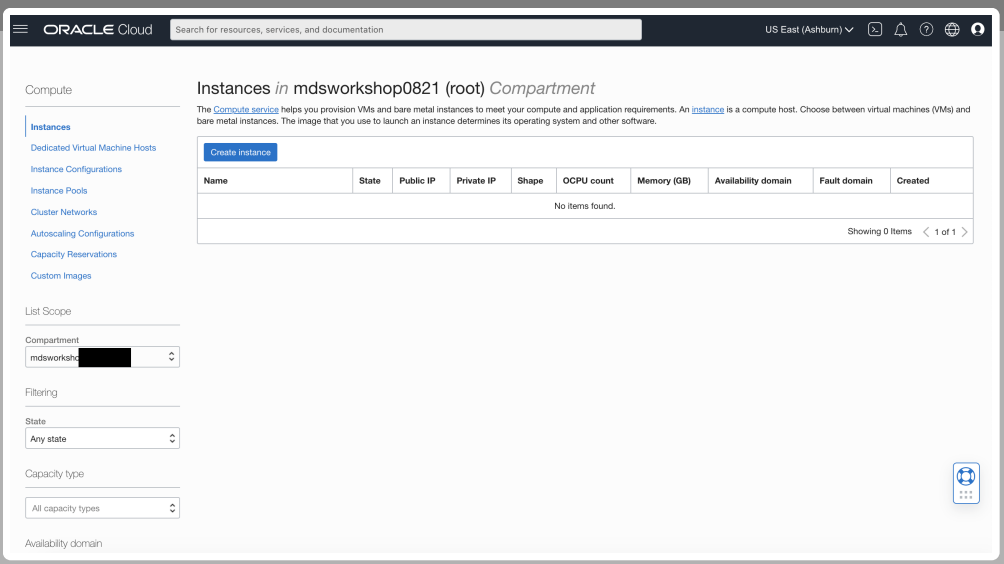
id-rsa.pub

```
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDdBwplv2CYDDZyQ1F+V3Fu5zyZY9augGBLbmT6+MbtObn+WgwgA+Gotn
+uVUGn3Z5AiXRIeiih5ILdS8/6Ya46vvuD6COgRErgED2nIF737G8/zsM7P6hVMnRCQywvHzW9+yRcGN/XW
+FxOJgfmx0VVlWB/K6s81FD6TRohIhQgnUSxZrlZhrfQAkjvsZfvU6ZrIGT7Ofy0+lgav0wu6bZYy1vx94E5y6LigmPWKiNB5b3rpCA6x
5a6uPa1FNcdNTUFbBvY/XoBRQ3amPA8TTG8ZsYa3wZjQQXEu6bsu7udiQWZ3n9xNOXvObDa4ay7DZ8r+lewQC/+35lcrwWxVj9
pflearn@6264b69ff0ec
```

4. To launch a Linux Compute instance, go to Navigation Menu Compute Instances



5. On Instances in **(root)** Compartment, click **Create Instance**



6. On Create Compute Instance

   Enter Name

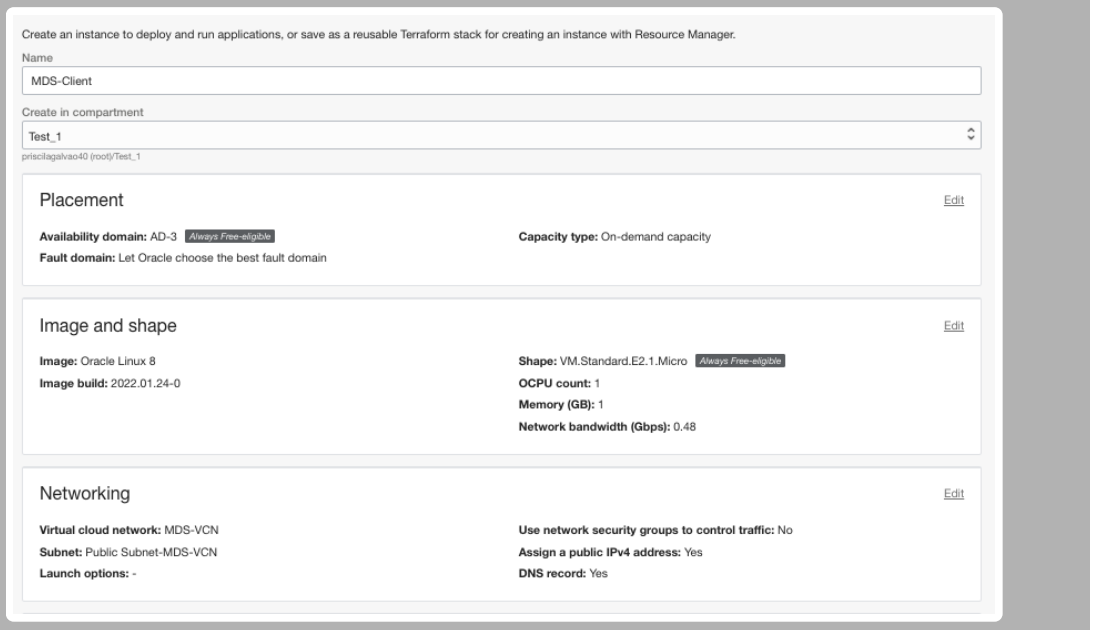
```
myclient
```

7. Make sure **(root)** compartment is selected

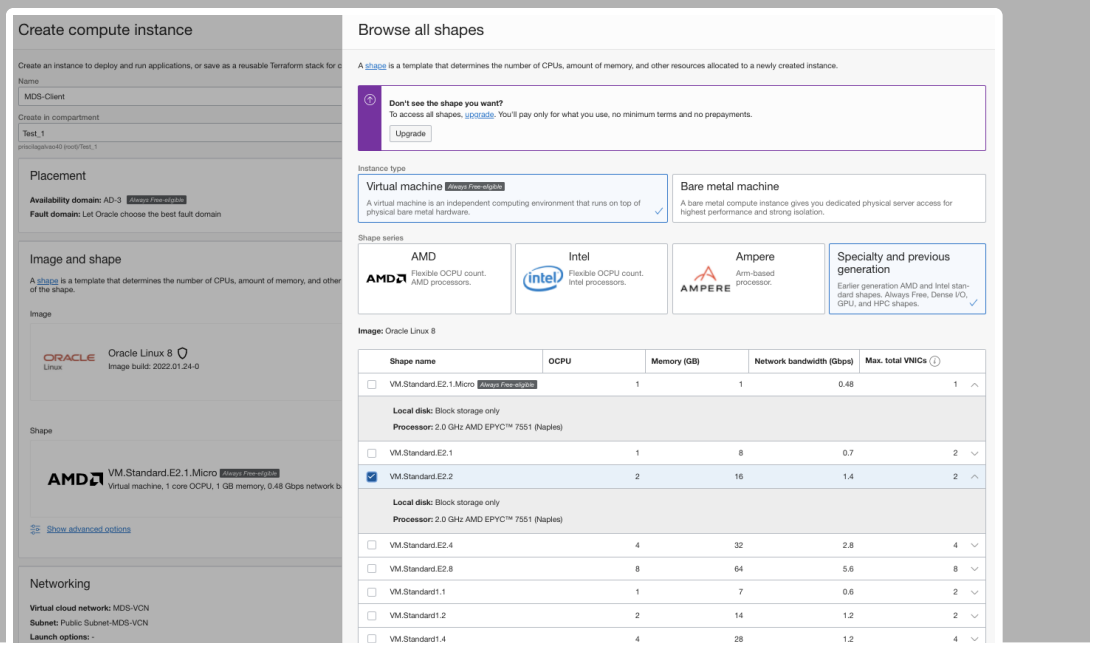
8. On Placement, keep the selected Availability Domain

9. On Image and Shape click the **Edit** link

    - On Image: Keep the selected Image, Oracle Linux 8

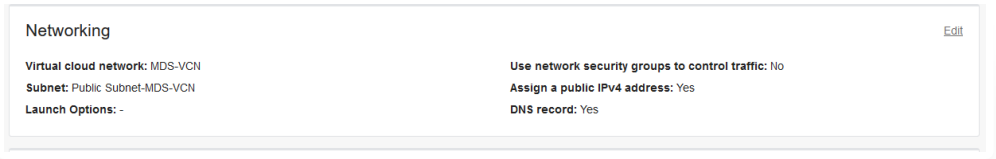


    - On Shape - Click the **change shape** button
    - Select Instance Shape: VM.Standard.E2.2

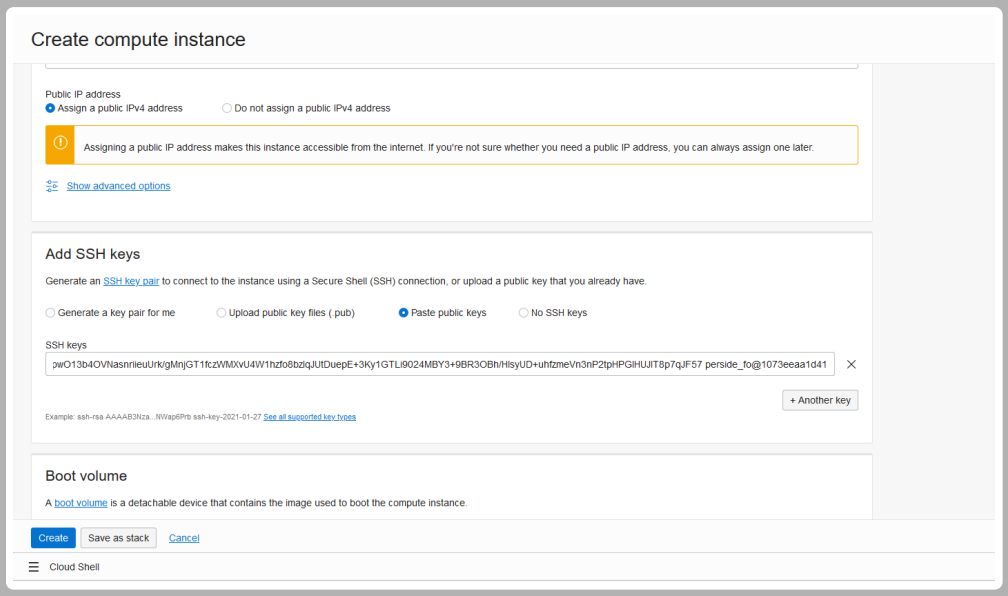


10. On Networking, make sure '**myvcn**' is selected

    'Assign a public IP address' should be set to Yes

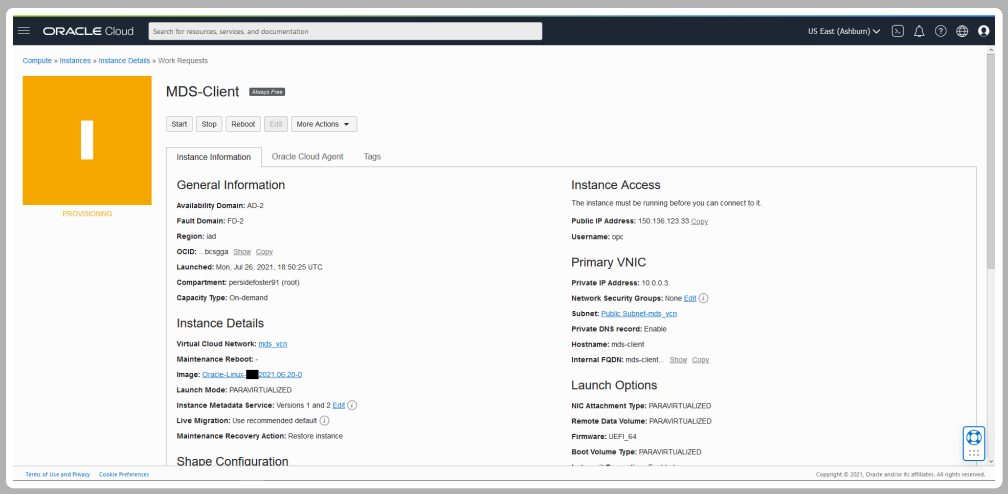


11. On Add SSH keys, paste the public key from the notepad.

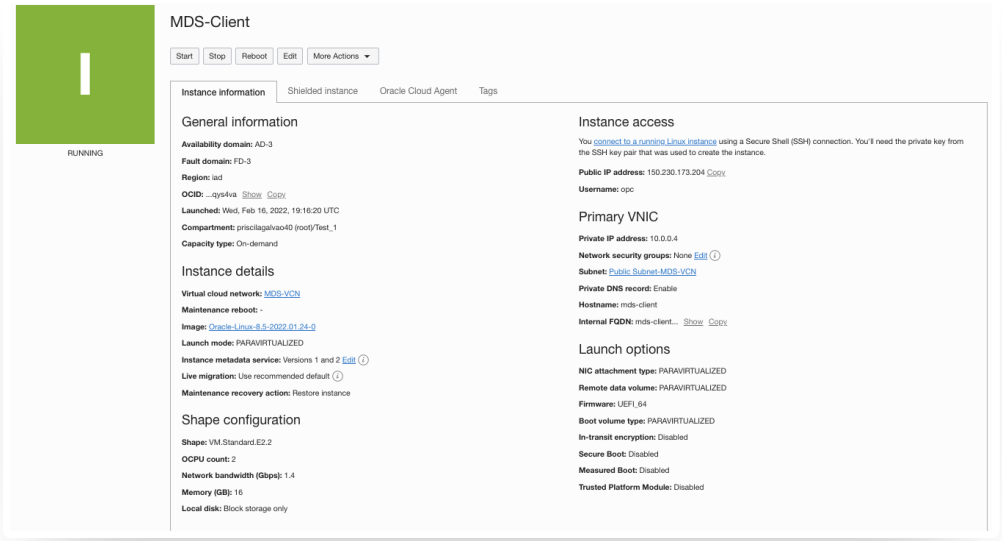


12. Click '**Create**' to finish creating your Compute Instance.

13. The New Virtual Machine will be ready to use after a few minutes. The state will be shown as 'Provisioning' during the creation



14. The state 'Running' indicates that the Virtual Machine is ready to use.

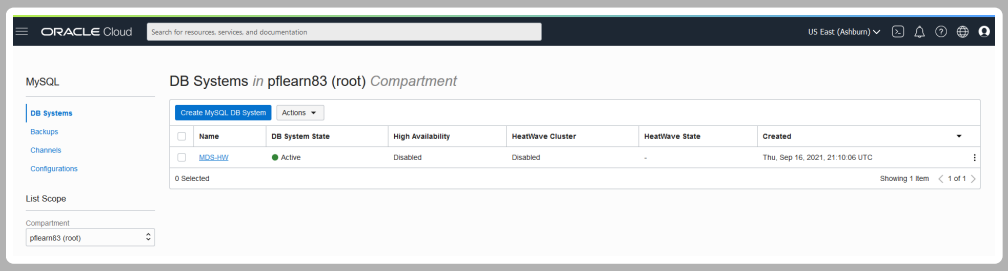


Task 3:Connect to Compute Instance with SSH Key

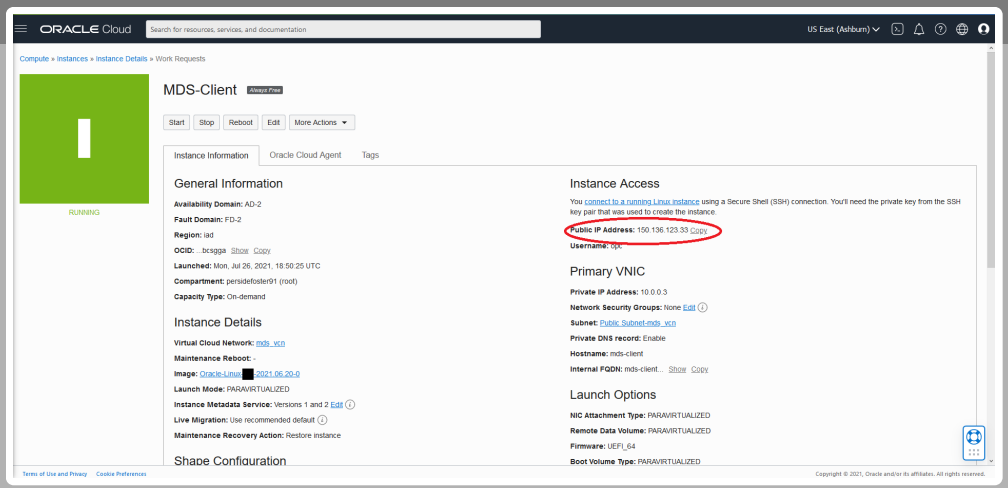
To connect to **myclient** you will need to properly setup your SSH command. Do the following steps:

1. Copy the public IP address of the active Compute Instance to a notepad

   a. Go to Navigation Menu Compute Instances



   b. Click the `myclient` Compute Instance link



   c. Copy `myclient` plus the `Public IP Address` to the notepad

2. Indicate the location of the private key you created earlier with **myclient**.

2. Indicate the location of the private key you created earlier with **myclient**.

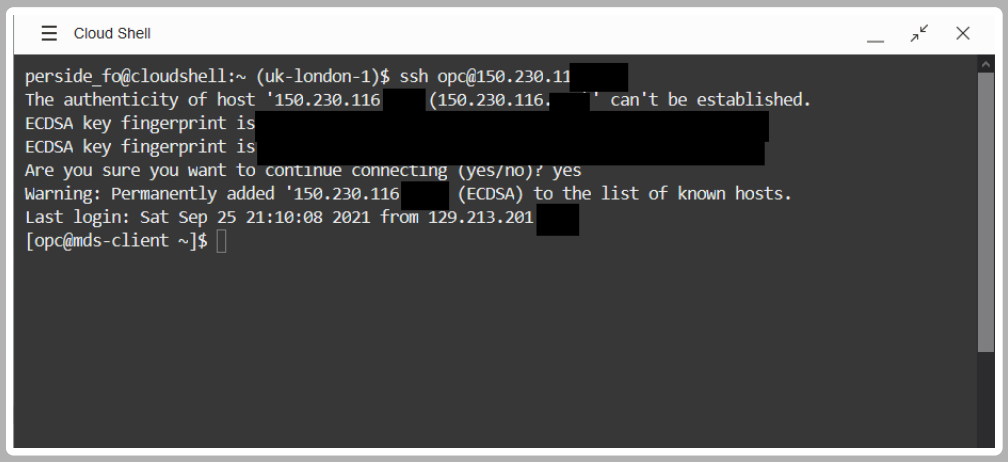
   Enter the username **opc** and the Public **IP Address**.

   Note: The **myclient** instance shows the Public IP Address as mentioned on TASK 5: #11

   (Your SSH login command should look like this:

   ssh -i ~/.ssh/id_rsa opc@132.145.170...)

```
ssh -i ~/.ssh/id_rsa opc@<your_compute_instance_ip>
```



   ** You are ready to install MySQL on the Compute Instance**

You may now proceed to the next lab

Acknowledgements

- **Author** - Dale Dasker, MySQL Solution Engineering
- **Last Updated By/Date** - <Dale Dasker, April 2022

MySQL Enterprise Edition - Implementation Essentials Bootcamp | Lab 3: Setup

SETUP

Environment Setup

Objective: Connect Personal Computer to the Oracle Network and the Oracle Cloud Infrastructure (OCI)

In this lab you will Download Lab Materials, plus connect your Personal Computer to the Oracle Network and the Oracle Cloud Infrastructure (OCI)

Estimated Lab Time: -- 10 minutes

Objectives

In this lab, you will:

- Download lab materials
- Setup SSH client
- Record Server information

Prerequisites

In compliance with Oracle security policies, I acknowledge I will not load actual confidential customer data or Personally Identifiable Information (PII) into my demo environment

This lab assumes you have:

- An Oracle account
- All previous labs successfully completed

Task 1: Download Lab Material and SSH client

1. lecture

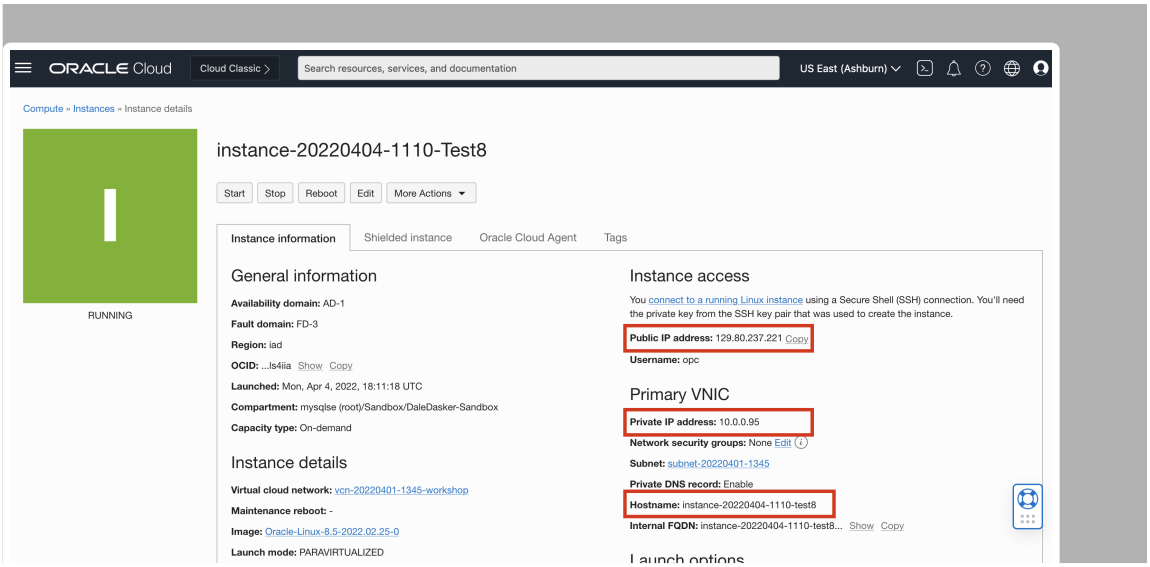
2. lab guide
3. SSH keys to connect labs (it's the same key in two different formats). These keys should have been created when you were creating your Compute Instance.
    - id_rsa in native openssl format. Use it with Workbench
    - id_rsa.ppk in putty format for windows. Use it only with putty
4. If you have not yet installed an SSH client on your laptop, please install one e.g. (windows) https://www.putty.org/

Task 2: Record Lab Server info on Notepad

student###-Server:

- Hostname:
- Hostname FQDN:
- Public IP: (e.g. 130.61.56.195)
- Private IP: (e.g. 10.0.11.18)

Example:



Task 3: Review Misc Lab Information

1. Document standard
    - When in the manual you read **shell>** the command must be executed in the Operating System shell.
    - When in the manual you read **mysql>** the command must be executed in a client like

- When in the manual you read **mysql>** the command must be executed in a client like MySQL, MySQL Shell, MySQL Workbench, etc. We recommend students to use MySQL Shell to practice with it.
- When in the manual you read MySQL **mysqlsh>** the command must be executed in MySQL Shell.

2. Lab standard

    - shell> the command must be executed in the Operating System shell
    - mysql> the command must be executed in a client like MySQL, MySQL Workbench
    - mysqlsh> the command must be executed in MySQL shell

3. The software used for the labs is located on a local /workshop folder within each server.

4. Tip: set the keep alive for SSH connection to 60 seconds, to keep session open during lectures

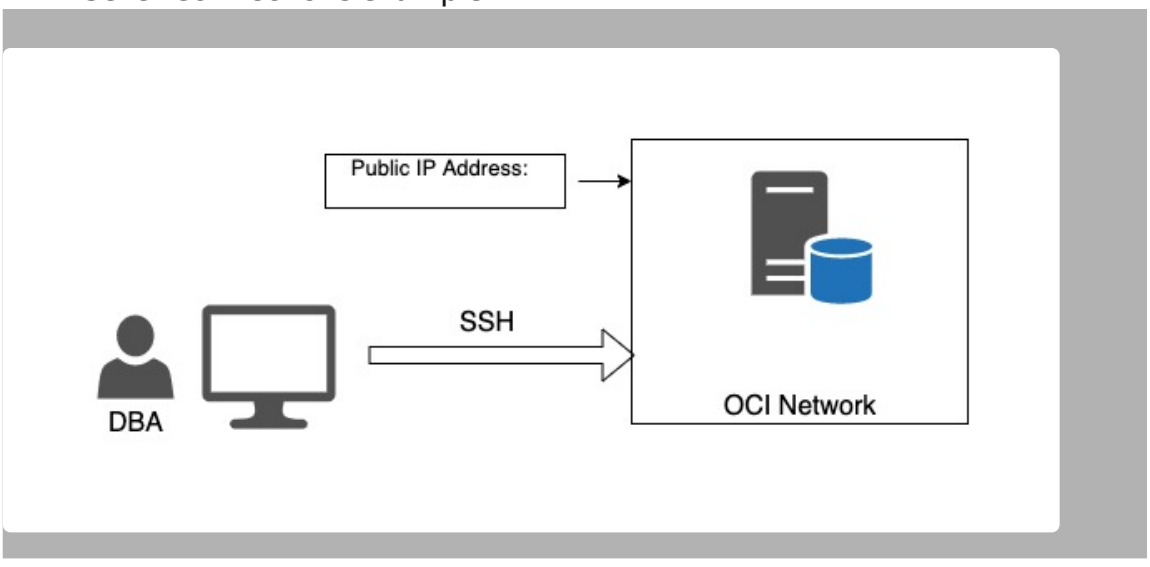
5. Linux ***opc*** user has limited privileges. To work with administrative privileges, use "sudo" like
   shell> sudo su - root

Task 4: Setup Lab Server and Connection

1. Server description **ServerA** will be used to run the full Workshop on. You will:

    - Install MySQL Enterprise Edition 8.0.
    - Install a MySQL Shell as a command line interface for MySQL Enterprise Edition.
    - Install the Sample Employees Database

2. Sever Connections example:



3. Test the connection to your Linux machines from your laptop using these parameters

    - a. SSH connection
    - b. SSH key file named “id_rsa” or "
    - c. username “opc”
    - d. no password
    - e. Public IP address of your assigned Linux VM (serverA, serverB)

4. Examples of connections:

Linux: use “id_rsa” key file

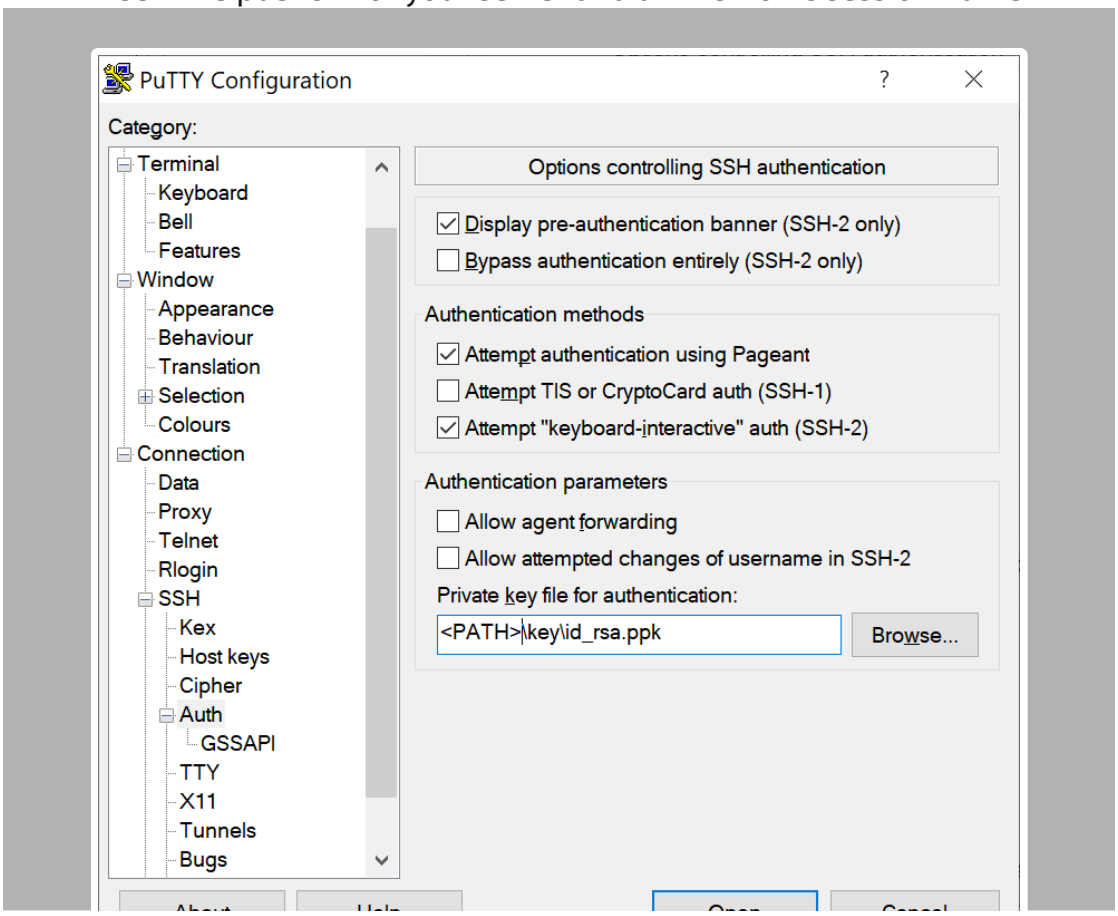
shell>

```
ssh -i id_rsa opc@public_ip
```

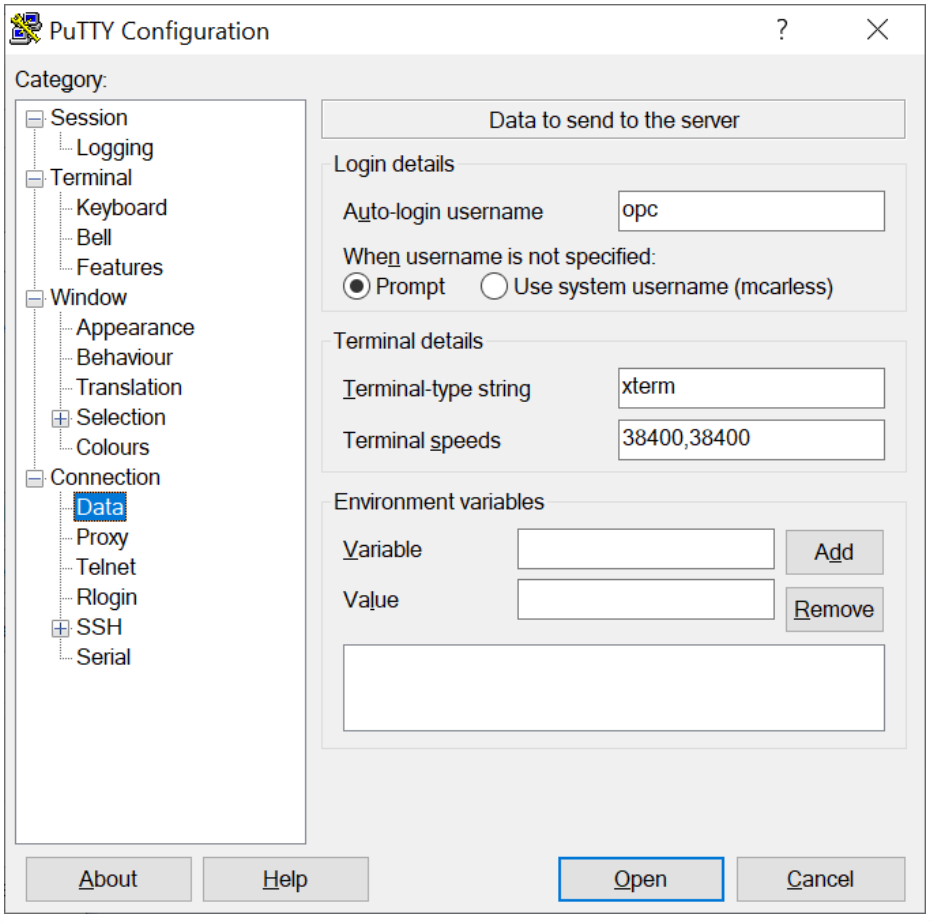
Windows: use “id_rsa.ppk” key file

1. Open putty

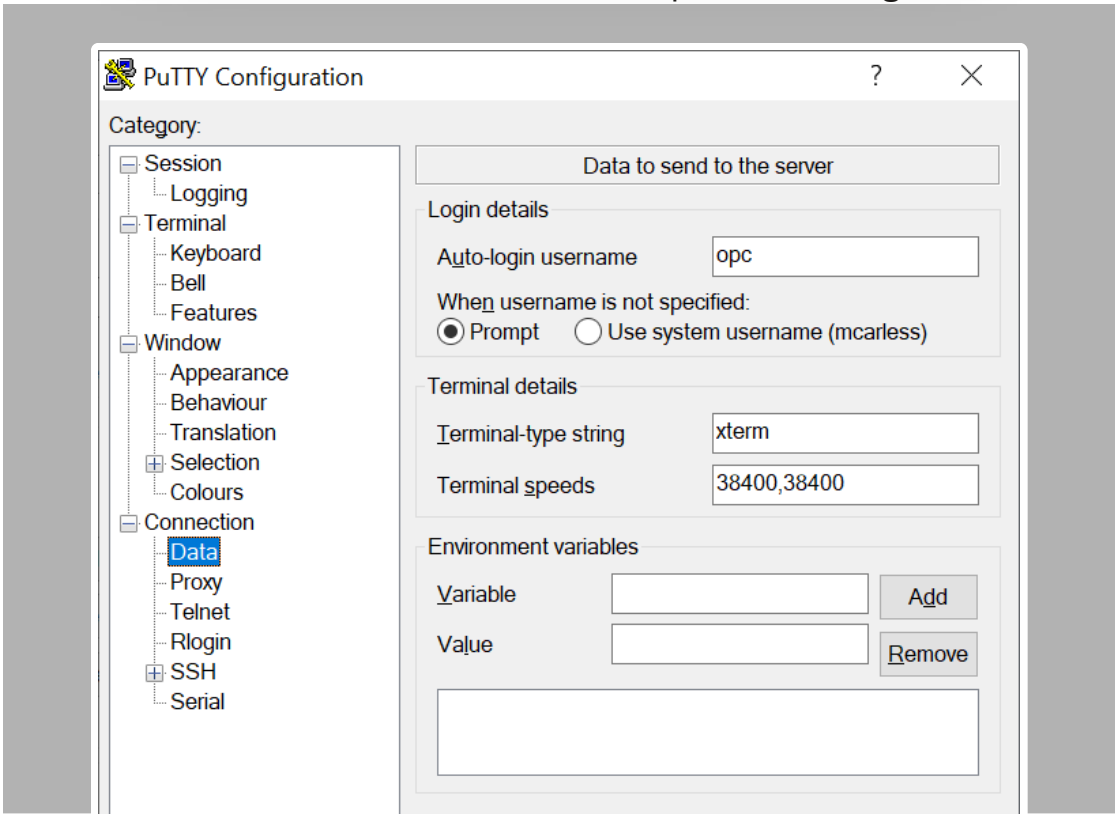
2. Insert the public IP of your server and a mnemonic session name



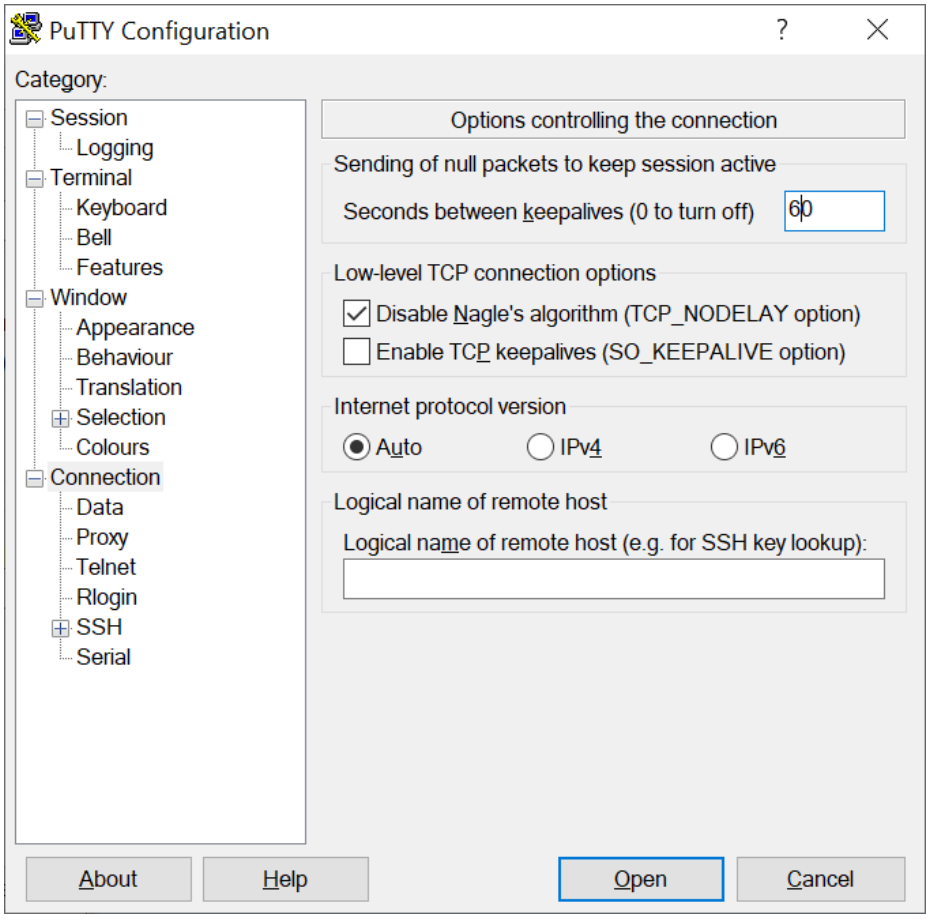
3. Choose "Connection SSH Auth" and provide the id_rsa.ppk path



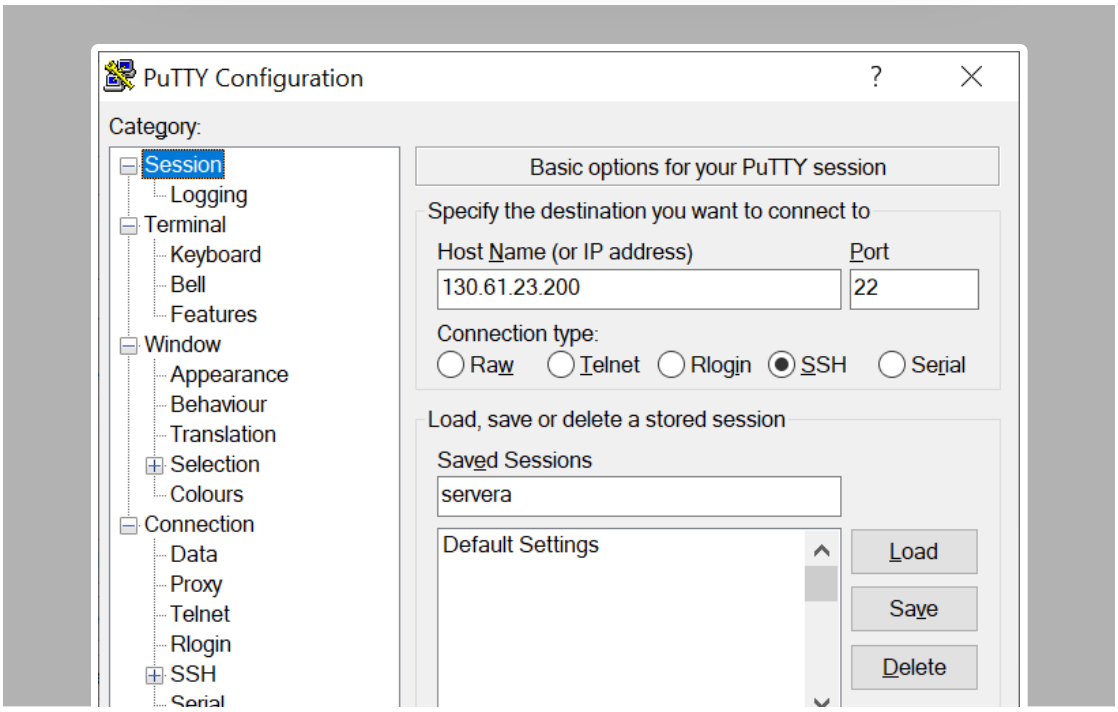
4. Select "Connection Data" and insert "opc" in "Auto-login username"

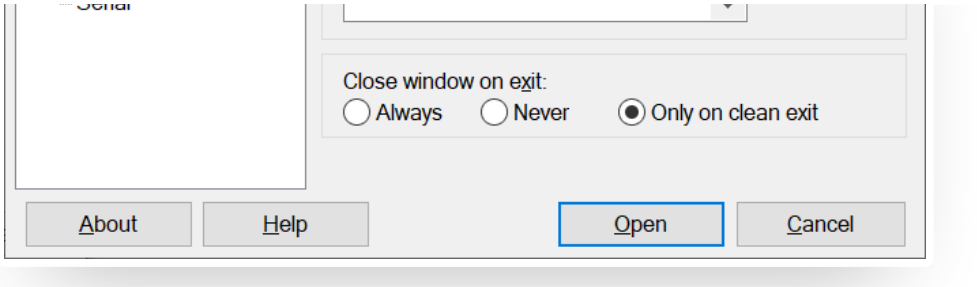


5. e) Choose Connection and insert "60" in "Seconds between keepalives



6. Return to "Session" and click save





Task 5: Setup workshop directory on Server

1. SSH to Server

   shell>

```
ssh -i id_rsa opc@public_ip
```

2. Make /workshop Directory

   shell>

```
sudo mkdir /workshop
```

3. FTP workshop files

   shell>

```
cd /workshop
```

   shell>

```
sudo wget https://objectstorage.us-ashburn-
1.oraclecloud.com/p/votWQhR-wMX-
ro12lv5DZ9XVHvGaNu5FcMEumkeKQ5gSOeEhjkXVKyWHdorYUyIL
/n/idazzjlcjqzj/b/bucket-20230109-1614-
Security_Workshop_01252023/o/workshop.tar.gz
```

4. Extract workshop files

   shell>

```
sudo tar xvf workshop.tar.gz
```

Learn More

- Creating SSH Keys
- Compute SSH Connections

Acknowledgements

- **Author** - Dale Dasker, MySQL Solution Engineering
- **Last Updated By/Date** - <Dale Dasker, January 2023

MySQL Enterprise Edition - Implementation Essentials Bootcamp | Lab 4: Install - MySQL Enterprise Edition

INSTALL - MYSQL ENTERPRISE EDITION

Introduction

Detailed Installation of MySQL Enterprise Edition 8.0 and MySQL Shell on Linux

Objective: RPM Installation of MySQL 8 Enterprise on Linux

RPM Installation of MySQL Enterprise 8 on Linux

Estimated Time: 15 minutes

Objectives

In this lab, you will:

- Install MySQL Enterprise Edition
- Start and test MySQL Enterpriese Edition Install
- Install MySQL Shell and Connect to MySQL Enterprise

Prerequisites

Test code This lab assumes you have:

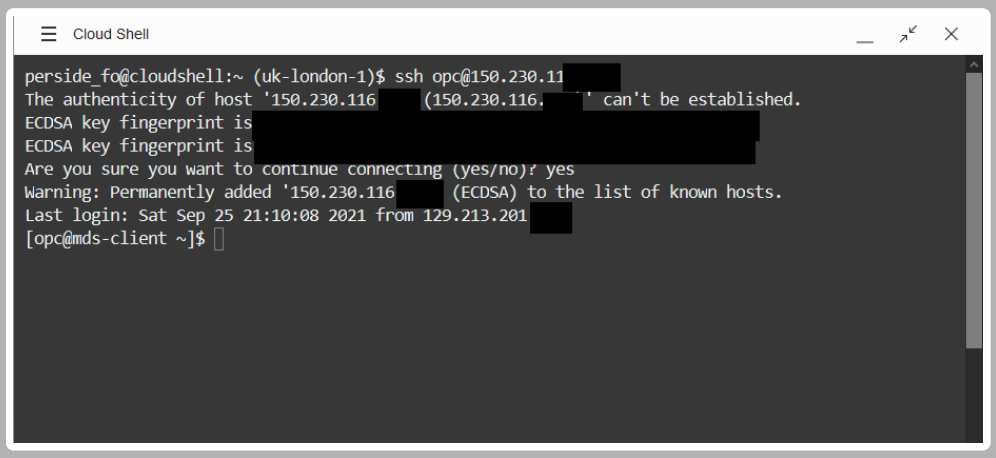
- An Oracle account
- All previous labs successfully completed
- Lab standard
    - shell> the command must be executed in the Operating System shell
    - mysql> the command must be executed in a client like MySQL, MySQL Workbench
    - mysqlsh> the command must be executed in MySQL shell

Task 1: Install MySQL Enterprise Edition using Linux RPM's

Note: If not already connected with SSH

- connect to **myclient** instance using Cloud Shell (**Example:** ssh -i ~/.ssh/id_rsa opc@132.145.17....)

```
ssh -i ~/.ssh/id_rsa
opc@<your_compute_instance_ip>
```



1. Install the RPM's

shell>

```
cd /workshop
```

shell>

```
sudo yum -y install *.rpm
```

Task 2: Start and test MySQL Enterprise Edition Install

1. Start your new mysql instance

shell>

```
sudo systemctl start mysqld
```

```
sudo systemctl start mysqld
```

2. Verify that process is running

   shell>

```
ps -ef | grep mysqld
```

   shell>

```
netstat -an | grep 3306
```

3. Another way is searching the message “ready for connections” in error log as one of the last

   shell>

```
sudo grep -i ready /var/log/mysqld.log
```

4. Retrieve root password for first login:

   shell>

```
sudo grep -i 'temporary password'
/var/log/mysqld.log
```

5. Login to the the mysql-enterprise installation and check the status (you will be asked to change password)

   shell>

```
mysqlsh --uri root@localhost:3306 --sql -p
```

6. Create New Password for MySQL Root

   mysqlsh>

```
ALTER USER 'root'@'localhost' IDENTIFIED BY
'Welcome1!';
```

mysqlsh>

```
status
```

7. Create a new administrative user called 'admin' with remote access and full privileges

mysqlsh>

```
CREATE USER 'admin'@'%' IDENTIFIED BY
'Welcome1!';
```

mysqlsh>

```
GRANT ALL PRIVILEGES ON *.* TO 'admin'@'%' WITH
GRANT OPTION;
```

mysqlsh>

```
quit
```

Learn More

- MySQL Linux Installation
- MySQL Shell Installation

Acknowledgements

- **Author** - Dale Dasker, MySQL Solution Engineering
- **Last Updated By/Date** - <Dale Dasker, January 2023

MySQL Enterprise Edition - Implementation Essentials Bootcamp | Lab 5: Install - Verify MySQL Enterprise Edition

INSTALL - VERIFY MYSQL ENTERPRISE EDITION

Introduction

Goal: Verify the new MySQL Installation on Linux and import test databases

Objectives:

- understand better how MySQL connection works
- install test databases for labs (world and employees)
- have a look on useful statements

Estimated Time: -- 10 minutes

Objectives

In this lab, you will:

- Discuss MySQL Connection
- Connect to Port 3306
- Import Sample Databses
- Learn Useful SQL Statements

Prerequisites

This lab assumes you have:

- An Oracle account
- All previous labs successfully completed
- Lab standard

- shell> the command must be executed in the Operating System shell
- mysql> the command must be executed in a client like MySQL, MySQL Workbench
- mysqlsh> the command must be executed in MySQL shell

Task 1: Discuss MySQL Connection

Please note that now you have an instance on the server on port 3306. To connect to MySQL, always use the IP address, otherwise you may connect to wrong instance. Here we practice connecting to the right one (port 3310 is intentionally wrong). To help you understand "why" these check lines (not all are always available…)

- Current user:
- Connection:
- UNIX socket:
- TCP port:

Task 2: Connect to Port 3306

1. **shell>**

```
mysql -u root -p --protocol=tcp
```

mysql>

```
status
```

mysql>

```
exit
```

2. Check a different port: **mysql>**

```
mysql -uroot -p -h localhost -P3310 --
protocol=tcp
```

Task 3: Import Sample Databases

1. Import the employees demo database that is in /workshop/databases folder.

   shell>

```
cd /workshop/database
```

   shell>

```
mysql -uroot -pWelcome1! -P3306 -h 127.0.0.1 <
./employees.sql
```

Task 4: Learn Useful SQL Statements

1. **shell>**

```
mysql -uroot -pWelcome1! -h 127.0.0.1 -P 3306
```

2. **mysql>**

```
SHOW VARIABLES LIKE "%version%";
```

3. **mysql>**

```
SELECT table_name, engine FROM
INFORMATION_SCHEMA.TABLES WHERE engine <>
'InnoDB';
```

4. **mysql>**

```
SELECT table_name, engine FROM
INFORMATION_SCHEMA.TABLES WHERE engine =
'InnoDB';
```

5. **mysql>**

```
SELECT table_name, engine FROM
INFORMATION_SCHEMA.TABLES where engine = 'InnoDB'
and table_schema not in
('mysql','information_schema', 'sys');
```

6. **mysql>**

```
SELECT ENGINE, COUNT(*), SUM(DATA_LENGTH)/ 1024 /
1024 AS 'Data MB', SUM(INDEX_LENGTH)/1024 / 1024 AS
'Index MB' FROM information_schema.TABLEs group by
engine;
```

7. **mysql>**

```
SELECT table_schema AS 'Schema', SUM( data_length )
/ 1024 / 1024 AS 'Data MB', SUM( index_length ) /
1024 / 1024 AS 'Index MB', SUM( data_length +
index_length ) / 1024 / 1024 AS 'Sum' FROM
information_schema.tables GROUP BY table_schema
;
```

8. The “G” is like “;” with a different way to show results **mysql>**

```
SHOW GLOBAL VARIABLESG
```

mysql>

```
SHOW GLOBAL STATUSG
```

mysql>

```
SHOW FULL PROCESSLIST;
```

mysql>

```
SHOW ENGINE INNODB STATUSG
```

```
exit
```

Learn More

- MySQL Tutorial

MySQL Enterprise Edition - Implementation Essentials Bootcamp | Lab 6: Security - MySQL Users

Web Clip

SECURITY - MYSQL USERS

Introduction

Users management Objective: explore user creation and privileges on a Server

*This lab walks you through creating some users which will be used to Audit.

Estimated Time: 10 minutes

Objectives

In this lab, you will do the followings:

- Connect to mysql-enterprise
- Create appuser

Prerequisites

This lab assumes you have:

- An Oracle account
- All previous labs successfully completed
- Lab standard
    - shell> the command must be executed in the Operating System shell
    - mysql> the command must be executed in a client like MySQL, MySQL Workbench
    - mysqlsh> the command must be executed in MySQL shell

Notes:

- Open a notepad file and your linux Private IP on student###-serverA

- serverA PRIVATE ip: (client_ip)

Task 1: Connect to mysql-enterprise on Server

1. Connect to your mysql-enterprise with administrative user

   shell>

```
mysql -uroot -pWelcome1! -h 127.0.0.1 -P 3306
```

2. Create a new user and restrict the user to your “Server” IP

   a. **mysql>**

```
CREATE USER 'appuser1'@'127.0.0.1' IDENTIFIED BY
'Welcome1!';
```

   b. **mysql>**

```
GRANT ALL PRIVILEGES ON employees.* TO
'appuser1'@'127.0.0.1';
```

   c. **mysql>**

```
SHOW GRANTS FOR 'appuser1'@'127.0.0.1';
```

Task 2: Add additional users

1. Using the Administrative Connection, create a new user and restrict the user to your “Server” IP

   a. **mysql>**

```
CREATE USER 'appuser2'@'127.0.0.1' IDENTIFIED BY
'Welcome1!';
```

b. **mysql>**

```
GRANT ALL PRIVILEGES ON employees.* TO
'appuser2'@'127.0.0.1';
```

Task 3: Connect to a second mysql-enterprise on Server

1. Open a new SSH connection on Server and from there connect to mysql-enterprise with appuser1

   a. connect to mysql-enterprise with appuser1

   shell>

   ```
   mysql –u appuser1 –pWelcome1! –h 127.0.0.1 –P
   3306
   ```

   b. Run a select on the tables e.g.

   mysql>

   ```
   USE employees;
   ```

   mysql>

   ```
   SELECT * FROM employees;
   ```

Task 4: Use appuser1 connection - **** OPTIONAL ****

1. Close and reopen the appuser1 connection for the user, then repeat above commands. There is a difference?

   mysql>

   ```
   exit
   ```

shell>

```
mysql -u appuser1 -pWelcome1! -h 127.0.0.1 -P
3306
```

mysql>

```
USE employees;
```

mysql>

```
SELECT * FROM employees;
```

2. Switch to the administrative connection revoke 'USAGE' privilege using and administrative connection and verify (tip: this privilege can't be revoked…)

mysql>

```
REVOKE USAGE ON *.* FROM 'appuser1'@'127.0.0.1';
```

mysql>

```
SHOW GRANTS FOR 'appuser1'@'127.0.0.1';
```

3. Using the administrative connection revoke all privileges using and administrative connection and verify

mysql>

```
REVOKE ALL PRIVILEGES ON *.* FROM
'appuser1'@'127.0.0.1';
```

mysql>

```
SHOW GRANTS FOR 'appuser1'@'127.0.0.1';
```

4. Close and reopen appuser session, do you see schemas?

Task 5: Restore user privileges **** OPTIONAL ****

1. Using the administrative connection restore user privileges to reuse it in next labs

mysql>

```
GRANT ALL PRIVILEGES ON employees.* TO
'appuser1'@'127.0.0.1';
```

Learn More

MySQL Enterprise Edition - Implementation Essentials Bootcamp | Lab 7: Security - MySQL Enterprise Audit

SECURITY - MYSQL ENTERPRISE AUDIT

Introduction

MySQL Enterprise Audit Objective: Auditing in action…

Estimated Lab Time: 20 minutes

Objectives

In this lab, you will:

- Setup Audit Log
- Use Audit

Prerequisites

This lab assumes you have:

- An Oracle account
- All previous labs successfully completed
- Lab standard
- shell> the command must be executed in the Operating System shell
- mysql> the command must be executed in a client like MySQL, MySQL Workbench
- mysqlsh> the command must be executed in MySQL shell

Notes:

- Audit can be activated and configured without stopping the instance. In the lab we edit my.cnf to see how to do it in this way

Task 1: Setup Audit Log

1. If already connected to MySQL then exit **mysql>**

```
    exit
```

2. Enable Audit Log on mysql-enterprise (remember: you can't install on mysql-gpl). Audit is an Enterprise plugin.

   a. Load Audit functions. If running in a replicated environment, load the plugin no each of the Replicas first and then modify the SQL script to only load the functions. **shell>**

```
mysql -uroot -pWelcome1! -h 127.0.0.1 -P 3306 <
/usr/share/mysql-
8.0/audit_log_filter_linux_install.sql
```

   b. Edit the my.cnf setting in /mysql/etc/my.cnf **shell>**

```
sudo nano /etc/my.cnf
```

   c. Add the following lines to the bottom of the file. These lines will make sure that the audit plugin can't be unloaded and that the file is automatically rotated at 20 MB and format of data is JSON.

   shell>

```
plugin-load=audit_log.so
audit_log=FORCE_PLUS_PERMANENT
audit_log_rotate_on_size=20971520
audit_log_format=JSON
```

   d. Restart MySQL (you can configure audit without restart the server, but here we show how to set the configuration file)

   shell>

```
sudo service mysqld restart
```

3. Connect to your mysql-enterprise with administrative user

shell>

```
mysql -uroot -pWelcome1! -h 127.0.0.1 -P 3306
```

a. Using the Administrative Connection, create a Audit Filter for all activity and all users. Privileges required are AUDIT_ADMIN and SUPER

mysql>

```
SELECT audit_log_filter_set_filter('log_all', '{
"filter": { "log": true } }');
```

mysql>

```
SELECT audit_log_filter_set_user('%',
'log_all');
```

b. **mysql>**

```
exit
```

c. Monitor the output of the audit.log file:

shell>

```
sudo tail -f /var/lib/mysql/audit.log
```

Task 2: Use Audit

1. Login to mysql-enterprise with the user “appuser1”, then submit some commands

a. **shell>**

```
mysql -u appuser1 -pWelcome1! -h 127.0.0.1 -P
```

```
mysql -u appuser1 -p<redacted> -h 127.0.0.1 -P
3306
```

b. **mysql>**

```
USE employees;
```

c. **mysql>**

```
SELECT * FROM employees limit 25;
```

d. **mysql>**

```
SELECT emp_no,salary FROM employees.salaries WHERE
salary > 90000;
```

2. Let's setup Audit to only log connections. Using the Administrative Connection, create a Audit Filter for all connections

a. **mysql>**

```
SET @f = '{ "filter": { "class": { "name":
"connection" } } }';
```

b. **mysql>**

```
SELECT
audit_log_filter_set_filter('log_conn_events',
@f);
```

c. **mysql>**

```
SELECT audit_log_filter_set_user('%',
'log_conn_events');
```

3. Login to mysql-enterprise with the user “appuser1”, then submit some commands

a. **shell>**

```
mysql -u appuser1 -pWelcome1! -h 127.0.0.1 -P
3306
```

b. **mysql>**

```
USE employees;
```

c. **mysql>**

```
SELECT * FROM employees limit 25;
```

d. **mysql>**

```
SELECT emp_no,salary FROM employees.salaries WHERE
salary > 90000;
```

4. Let's setup Audit to only log unique users. Using the Administrative Connection, create a Audit Filter for appuser1

   a. Remove previous filter:

   mysql>

```
SELECT
audit_log_filter_remove_filter('log_conn_events
');
```

   mysql>

```
SELECT audit_log_filter_flush();
```

   b. **mysql>**

```
SELECT audit_log_filter_set_filter('log_all', '{
```

```
"filter": { "log": true } }');
```

c. **mysql>**

```
SELECT
audit_log_filter_set_user('appuser1@127.0.0.1',
'log_all');
```

d. **mysql>**

```
SELECT audit_log_filter_flush();
```

5. Login to mysql-enterprise with the user “appuser1”, then submit some commands

a. **shell>**

```
mysql -u appuser1 -pWelcome1! -h127.0.0.1 -P
3306
```

b. **mysql>**

```
USE employees;
```

c. **mysql>**

```
SELECT * FROM employees limit 25;
```

d. **mysql>**

```
SELECT emp_no,salary FROM employees.salaries WHERE
salary > 90000;
```

e. **mysql>**

```
quit;
```

6. Login to mysql-enterprise with the user “appuser2”, then submit some commands

   a. **shell>**

```
mysql -u appuser2 -pWelcome1! -h127.0.0.1 -P
3306
```

   b. **mysql>**

```
USE employees;
```

   c. **mysql>**

```
SELECT * FROM employees limit 25;
```

   d. **mysql>**

```
SELECT emp_no,salary FROM employees.salaries WHERE
salary > 90000;
```

7. Let's setup Audit to only log table accesss. Using the Administrative Connection, create a Audit Filter for tables

   a. Remove previous filter:

   mysql>

```
SELECT audit_log_filter_remove_filter('log_all
');
```

   mysql>

```
SELECT audit_log_filter_flush();
```

   b. **mysql>**

```
SELECT
audit_log_filter_set_filter('log_table_access_events
', '{ "filter": { "class": { "name": "table_access"
} } }');
```

c. **mysql>**

```
SELECT audit_log_filter_set_user('%',
'log_table_access_events');
```

d. Login to mysql-enterprise with the user “appuser1”, then submit some commands
shell>

```
mysql -u appuser1 -pWelcome1! -h127.0.0.1 -P
3306
```

e. **mysql>**

```
USE employees;
```

f. **mysql>**

```
SELECT * FROM employees limit 25;
```

g. **mysql>**

```
SELECT emp_no,salary FROM employees.salaries WHERE
salary > 90000;
```

8. Let's setup Audit to only log access to salaries tables. Using the Administrative Connection, create a Audit Filter for salaries

   a. Remove previous filter: **mysql>**

```
SELECT audit_log_filter_remove_filter('log_all
');
```

```
);
```

mysql>

```
SELECT audit_log_filter_flush();
```

b. **mysql>**

```
SET @f='

{
"filter": {
   "class":
     {
      "name": "table_access",
      "event":
        {
          "name": [ "insert", "update", "delete" ],
          "log": { "field": { "name":
"table_name.str", "value": "salaries" }}
        }
     }
 }
}';
```

c. **mysql>**

```
SELECT audit_log_filter_set_filter('salary_insert',
@f);
```

d. **mysql>**

```
SELECT audit_log_filter_set_user('%',
'salary_insert');
```

9. Login as 'appuser1' and run a query against the salaries table;

a. **shell>**

```
mysql -u appuser1 -pWelcome1! -h127.0.0.1 -P
3306
```

b. **mysql>**

```
USE employees;
```

c. **mysql>**

```
SELECT * FROM employees limit 25;
```

d. Run updates on salaries table **mysql>**

```
UPDATE employees.salaries SET salary = 74234 WHERE
emp_no = 10001;
```

10. Some Administrative commands for checking Audit filters and users. Log in using the Administrative Connection

shell>

```
mysql -uroot -p -h 127.0.0.1 -P 3306
```

a. Check existing filters: **mysql>**

```
SELECT * FROM mysql.audit_log_filterG
```

b. Check Users being Audited: **mysql>**

```
SELECT * FROM mysql.audit_log_userG
```

c. Reading from Audit Log within MySQL Client **mysql>**

```
SELECT JSON_PRETTY(CONVERT(audit_log_read(audit_log_read_bookmark()) using
utf8mb4))G
```

d. Global Audit log disable **mysql>**

```
SET GLOBAL audit_log_disable = true;
```

e. Check what Audit Functions are available **mysql>**

```
SELECT * FROM  mysql.func;
```

f. Check that the Audit plugin loaded **mysql>**

```
SELECT PLUGIN_NAME, PLUGIN_STATUS FROM INFORMATION_SCHEMA.PLUGINS WHERE
PLUGIN_NAME LIKE 'audit%';
```

11. You can check the documentation about other Log filters & policies

Learn More

- Writing Audit Filters
- Audit Filter Definitions

Acknowledgements

- **Author** - Dale Dasker, MySQL Solution Engineering
- **Last Updated By/Date** - <Dale Dasker, January 2023

MySQL Enterprise Edition - Implementation Essentials Bootcamp | Lab 8: Security - MySQL Enterprise Transparent Data Encryption (TDE)

Web Clip

SECURITY - MYSQL ENTERPRISE TRANSPARENT DATA ENCRYPTION

Introduction

3c) MySQL Enterprise Transparent Data Encryption Objective: Data Encryption in action…

This lab will walk you through encrypting InnoDB Tablespace files at rest

Estimated Lab Time: 20 minutes

Objectives

In this lab, you will:

- Install and encrypt Data Files

Prerequisites (Optional)

This lab assumes you have:

- An Oracle account
- All previous labs successfully completed
- Lab standard
    - shell> the command must be executed in the Operating System shell
    - mysql> the command must be executed in a client like MySQL, MySQL Workbench
    - mysqlsh> the command must be executed in MySQL shell

Notes:

- InnoDB Data At Rest

Task 1: Install and setup TDE

1. Install MySQL Enterprise Transparent Data Encrytption on mysql-enterprise using Administrative MySQL client connections

   shell>

```
mysql -u root -pWelcome1! -P3306 -h127.0.0.1
```

2. Check to see if any keyring plugin is installed and load if not:

   a. **mysql>**

```
SELECT PLUGIN_NAME, PLUGIN_STATUS FROM
INFORMATION_SCHEMA.PLUGINS WHERE PLUGIN_NAME LIKE
'keyring%';
```

   b. Edit the my.cnf setting in /etc/my.cnf

   shell>

```
sudo nano /etc/my.cnf
```

   b. Add the following lines to load the plugin and set the encrypted key file

   shell>

```
early-plugin-load=keyring_encrypted_file.so
keyring_encrypted_file_data=/var/lib/mysql-
keyring/keyring-encrypted
keyring_encrypted_file_password=V&rySec4eT
```

   c. Restart MySQL

   shell>

```
sudo service mysqld restart
```

3. "Spy" on employees.employees table

   a. **shell>**

```
sudo strings
"/var/lib/mysql/employees/employees.ibd" | head -
n50
```

4. Now we enable Encryption on the employees.employees table:

   a. **shell>**

```
mysql -u root -pWelcome1! -P3306 -h127.0.0.1
```

   b. **mysql>**

```
USE employees;
```

   c. **mysql>**

```
ALTER TABLE employees ENCRYPTION = 'Y';
```

5. "Spy" on employees.employees table again:

   a. **shell>**

```
sudo strings
"/var/lib/mysql/employees/employees.ibd" | head -
n50
```

6. Administrative commands

   a. Get details on encrypted key file: **mysql>**

```
SHOW VARIABLES LIKE
'keyring_encrypted_file_data'G
```

b. Set default for all tables to be encrypted when creating them: **mysql>**

```
SET GLOBAL default_table_encryption=ON;
```

c. Peek on the mysql System Tables: **mysql>**

```
sudo strings "/var/lib/mysql/mysql.ibd" | head -
n70
```

d. Encrypt the mysql System Tables: **mysql>**

```
ALTER TABLESPACE mysql ENCRYPTION = 'Y';
```

e. Validate encryption of the mysql System Tables: **mysql>**

```
sudo strings "/var/lib/mysql/mysql.ibd" | head -
n70
```

f. Show all the encrypted tables: **mysql>**

```
SELECT SPACE, NAME, SPACE_TYPE, ENCRYPTION FROM
INFORMATION_SCHEMA.INNODB_TABLESPACES WHERE
ENCRYPTION='Y'G
```

Learn More

MySQL Enterprise Edition - Implementation Essentials Bootcamp | Lab 9: Security - MySQL Data Masking

Web Clip

SECURITY - DATA MASKING

Introduction

Data Masking and de-identification MySQL Enterprise Masking and De-identification provides an easy to use, built-in database solution to help organizations protect sensitive data from unauthorized uses by hiding and replacing real values with substitutes. Objective: Install and use data masking functionalities

Estimated Lab Time: -- 12 minutes

Objectives

In this lab, you will:

- Create sample data with random generation utilites which are part of Enterprise Masking
- Test Masking of Sensitive Data
- Create a View and user which only sees masked data

Prerequisites

This lab assumes you have:

- An Oracle account
- All previous labs successfully completed
- Lab standard
    - shell> the command must be executed in the Operating System shell
    - mysql> the command must be executed in a client like MySQL, MySQL Workbench
    - mysqlsh> the command must be executed in MySQL shell

Notes:

- Data masking has more functions than what we test in the lab. The full list of functions is here
- https://dev.mysql.com/doc/refman/8.0/en/data-masking-usage.html

Task 1: Install masking plugin

1. To install the data masking plugin, execute with statements

   a. **shell>**

```
mysql -uroot -pWelcome1! -h 127.0.0.1 -P 3306
```

   b. **mysql>**

```
INSTALL PLUGIN data_masking SONAME
'data_masking.so';
```

   c. **mysql>**

```
SHOW PLUGINS;
```

2. Look for data_masking and check the status? Is it active?

Task 2: Use masking functions

1. Install masking functions

   a. **mysql>**

```
CREATE FUNCTION gen_range RETURNS INTEGER SONAME
'data_masking.so';
CREATE FUNCTION gen_rnd_email RETURNS STRING SONAME
'data_masking.so';
CREATE FUNCTION gen_rnd_us_phone RETURNS STRING
SONAME 'data_masking.so';
CREATE FUNCTION gen_rnd_ssn RETURNS STRING SONAME
```

```
'data_masking.so';
CREATE FUNCTION mask_inner RETURNS STRING SONAME
'data_masking.so';
CREATE FUNCTION mask_outer RETURNS STRING SONAME
'data_masking.so';
CREATE FUNCTION mask_ssn RETURNS STRING SONAME
'data_masking.so';
```

2. Use data masking functions

a. **mysql>**

```
SELECT mask_inner(last_name, 2,1) FROM
employees.employees limit 10;
```

b. **mysql>**

```
SELECT mask_outer(last_name, 2,1) FROM
employees.employees limit 10;
```

Task 3: Discussion and use Masking functions and random generators

1. Create Table to generate and add masking data

mysql>

```
CREATE TABLE employees_mask LIKE employees;
```

2. Add data to newly created table

mysql>

```
INSERT INTO employees_mask SELECT * FROM
employees;
```

3. Create new column for SSN's

mysql>

mysql>

```
ALTER TABLE employees_mask ADD COLUMN ssn
varchar(11);
```

4. Create new column for emails's

mysql>

```
ALTER TABLE employees_mask ADD COLUMN email
varchar(40);
```

5. Use Functions to generate sample SSN data

mysql>

```
UPDATE employees_mask SET ssn = gen_rnd_ssn() WHERE
1;
```

6. Use Functions to generate sample Email data

mysql>

```
UPDATE employees_mask SET email = gen_rnd_email()
WHERE 1;
```

7. Let's look at the data that we just created

mysql>

```
SELECT * FROM employees_mask LIMIT 5;
```

8. Let's mask the SSN

mysql>

```
SELECT
emp_no,first_name,last_name,mask_ssn(CONVERT(ssn
USING latin1)) AS ssn FROM employees_mask LIMIT
```

```
USING latin1)) AS ssn FROM employees_mask LIMIT
5;
```

9. Let's create a view which only shows the masked data

   mysql>

```
CREATE VIEW masked_customer AS SELECT
emp_no,first_name,last_name,mask_ssn(CONVERT(ssn
USING latin1)) AS ssn FROM employees_mask;
```

10. Let's create a user who only has access to the view with the masked data

    mysql>

```
CREATE USER 'accounting'@'%' IDENTIFIED BY
'Pa33word!';
```

    mysql>

```
GRANT SELECT ON employees.masked_customer TO
'accounting'@'%';
```

11. Log in with new user account and run queries

    mysql>

```
quit;
```

    shell>

```
mysql -uaccounting -pPa33word! -h 127.0.0.1 -P
3306
```

    mysql>

```
SELECT * FROM employees.masked_customer LIMIT 5;
```

12. Try accessing table that is not masked

mysql>

```
SELECT * FROM employees.employees_mask LIMIT 5;
```

Task 4: *** OPTIONAL *** Discussion and use Masking functions and random generators

1. Discuss differences between mask_inner and mask_outer

mysql>

```
SELECT mask_inner(last_name, 1,1, '&') FROM
employees.employees limit 1;
```

2. Use data masking random generators to these statements several times

a. **mysql>**

```
SELECT gen_range(1, 200);
```

b. **mysql>**

```
SELECT gen_rnd_us_phone();
```