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 2 of the MySQL Security Based Enterprise Features. We will go through how to setup and run Enterprise Audit and Transparent Data Encryption.

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.

Objectives

Learn More

- MySQL Home Page
- MySQL Enterprise Audit
- MySQL Enterprise Transparent Data Encryption

Acknowledgements

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

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

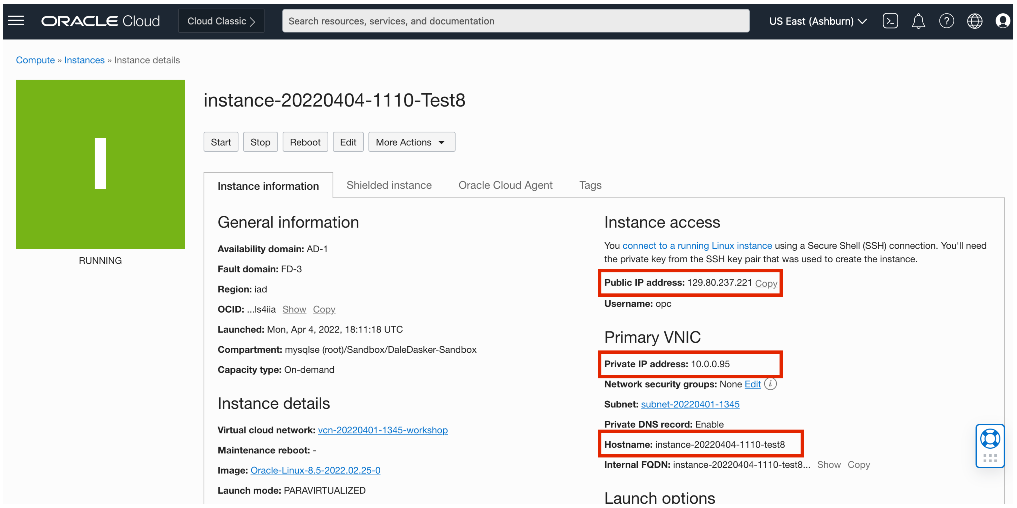
1. lectures pdf
2. lab guide
3. SSH keys to connect labs (it's the same key in two different formats). These keys should have been created when you were creating your Compute Instance.
    - id_rsa in native openssl format. Use it with Workbench
    - id_rsa.ppk in putty format for windows. Use it only 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:



instance-20220404-1110-Test8

General information

Availability domain: AD-1

Fault domain: FD-3

Region: iad

OCID: ...ls4iia Show Copy

Launched: Mon, Apr 4, 2022, 18:11:18 UTC

Compartment: mysqlse (root)/Sandbox/DaleDasker-Sandbox

Capacity type: On-demand

Instance details

Virtual cloud network: vcn-20220401-1345-workshop

Maintenance reboot: -

Image: Oracle-Linux-8.5-2022.02.25-0

Launch mode: PARAVIRTUALIZED

Instance access

You connect to a running Linux instance using a Secure Shell (SSH) connection. You'll need the private key from the SSH key pair that was used to create the instance.

Public IP address: 129.80.237.221 Copy

Username: opc

Primary VNIC

Private IP address: 10.0.0.95

Network security groups: None Edit

Subnet: subnet-20220401-1345

Private DNS record: Enable

Hostname: instance-20220404-1110-test8

Internal FQDN: instance-20220404-1110-test8... Show Copy

Launch options

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 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

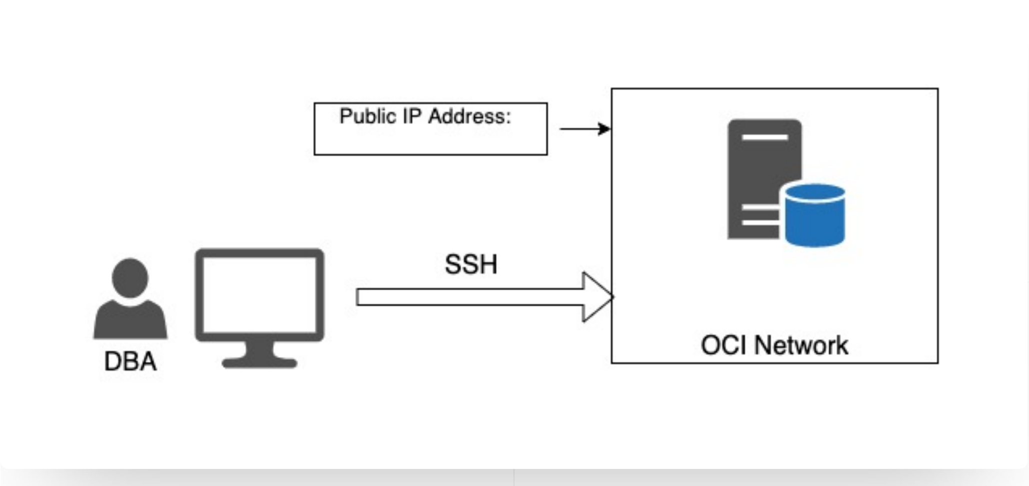
5. Linux ***opc*** user has limited privileges. To work with administrative privileges, use "sudo" like 🟩 shell> sudo su - root

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

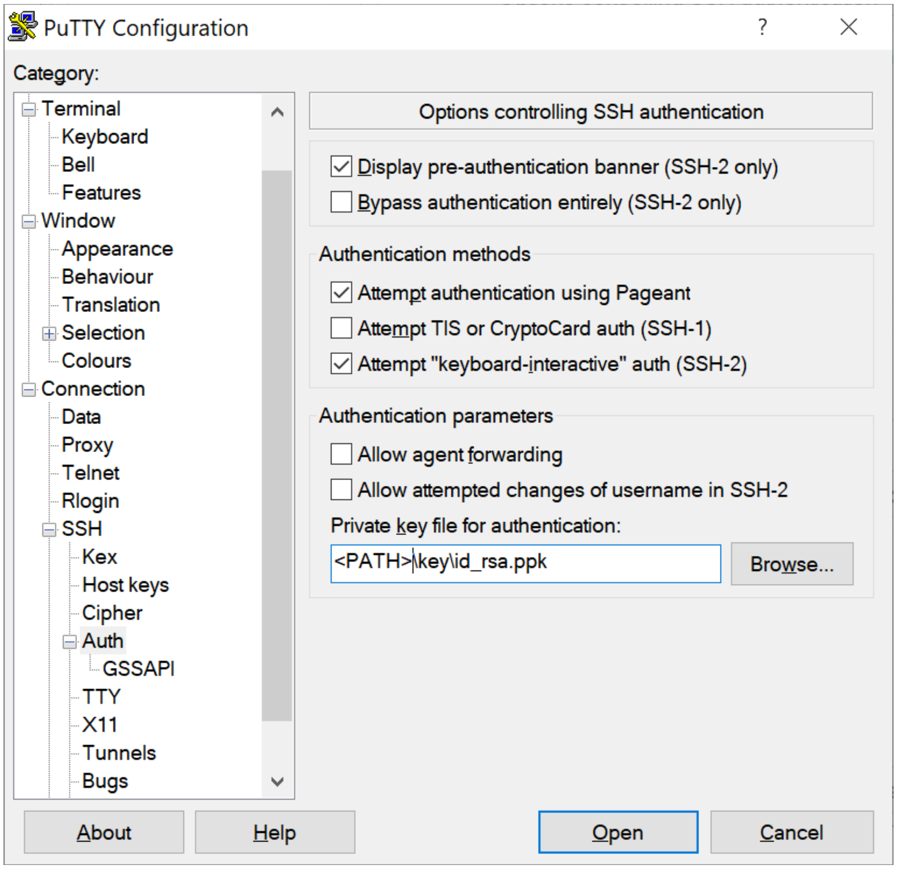
🟩 **shell>**

```
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



PuTTY Configuration

Options controlling SSH authentication

☑ Display pre-authentication banner (SSH-2 only)
□ Bypass authentication entirely (SSH-2 only)

Authentication methods

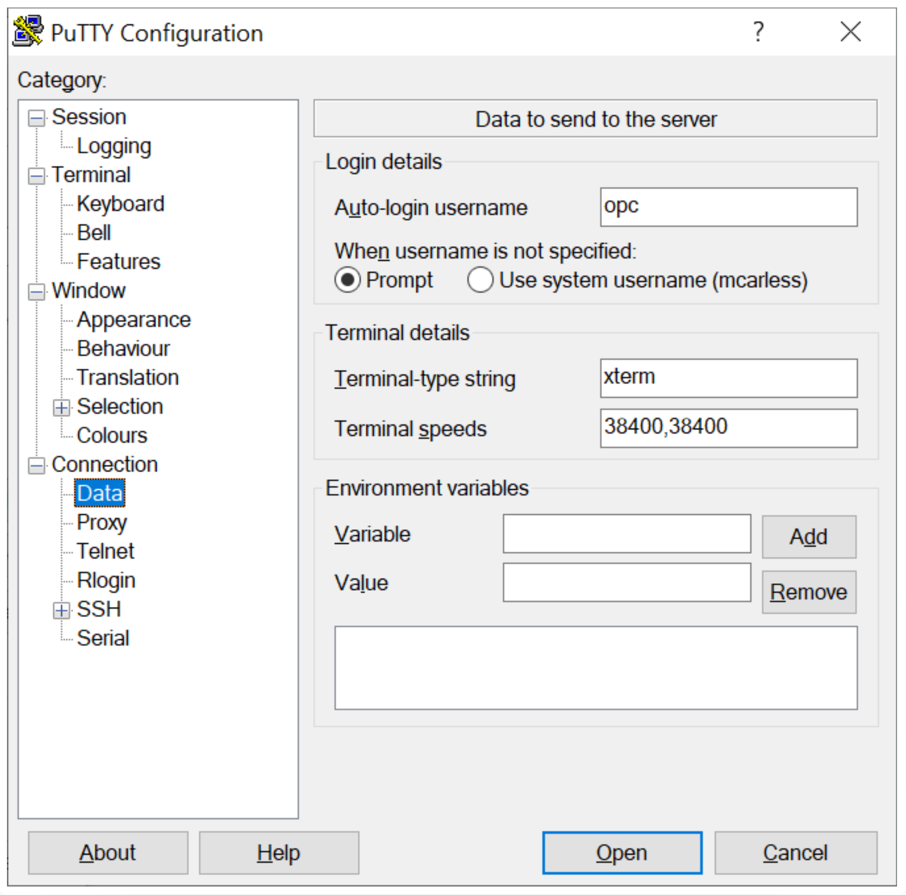
☑ Attempt authentication using Pageant
□ Attempt TIS or CryptoCard auth (SSH-1)
☑ Attempt "keyboard-interactive" auth (SSH-2)

Authentication parameters

□ Allow agent forwarding
□ Allow attempted changes of username in SSH-2

Private key file for authentication: <PATH>\key\id_rsa.ppk

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



Data to send to the server

Login details

Auto-login username: opc

When username is not specified:
● Prompt ○ Use system username (mcarless)

Terminal details

Terminal-type string: xterm

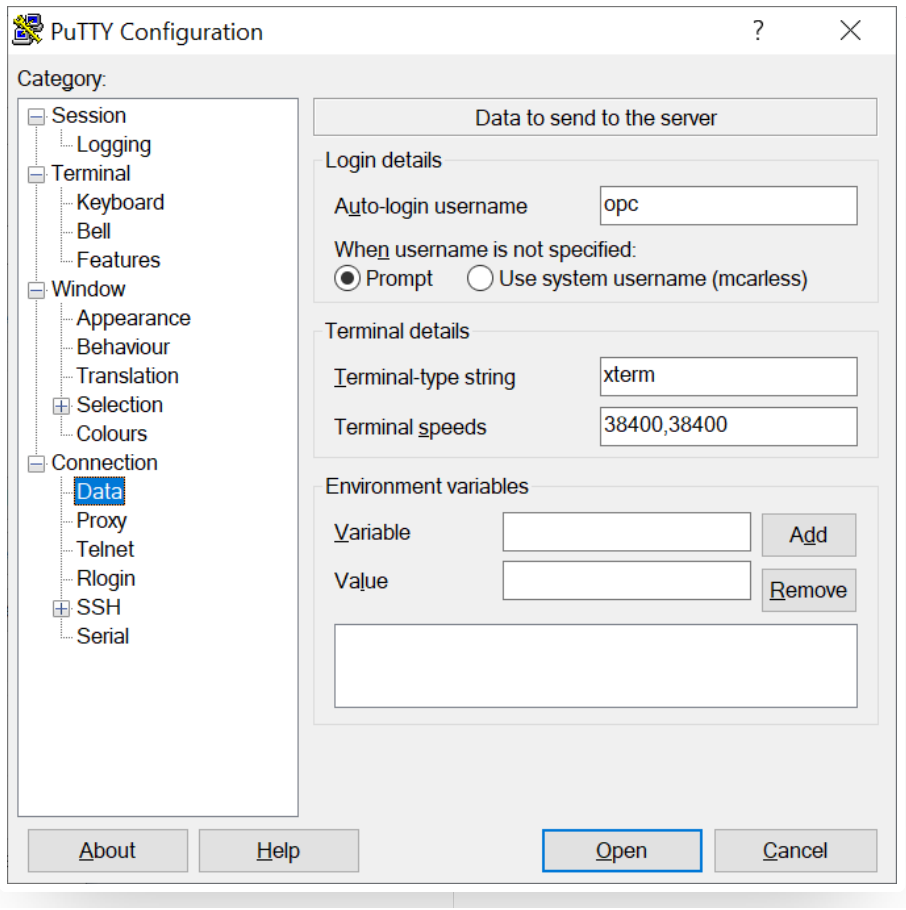
Terminal speeds: 38400,38400

Environment variables

Variable:

Value:

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



Data to send to the server

Login details

Auto-login username: opc

When username is not specified:
● Prompt ○ Use system username (mcarless)

Terminal details

Terminal-type string: xterm

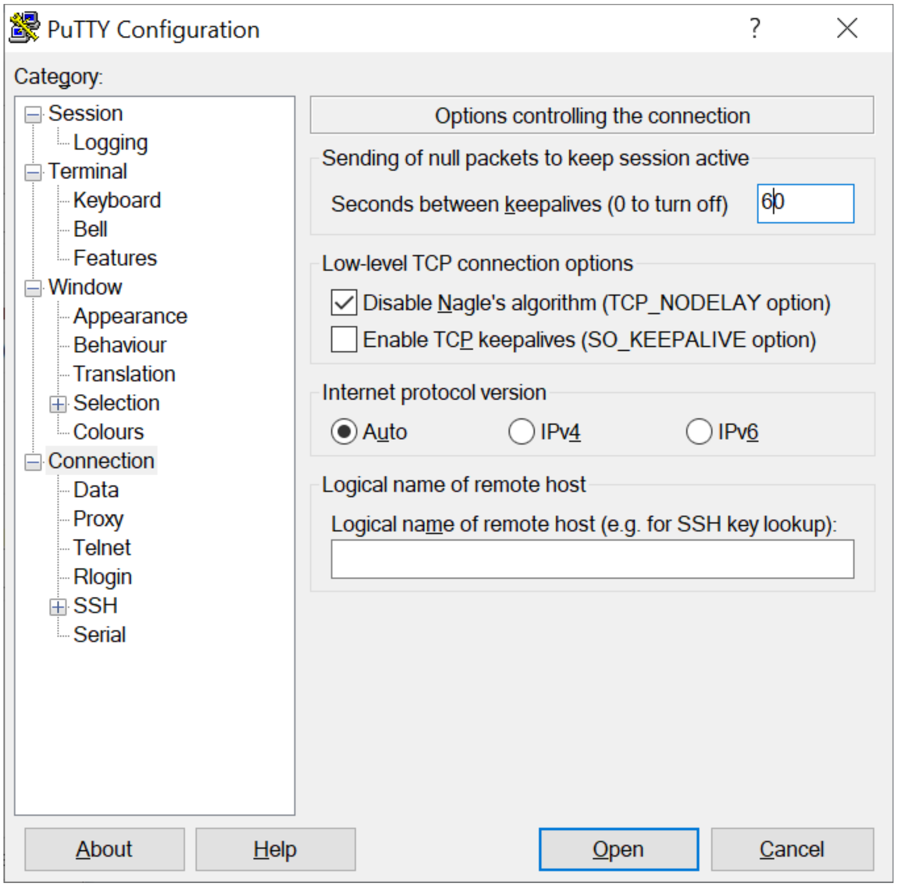
Terminal speeds: 38400,38400

Environment variables

Variable:

Value:

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



Options controlling the connection

Sending of null packets to keep session active

Seconds between keepalives (0 to turn off) 60

Low-level TCP connection options

☑ Disable Nagle's algorithm (TCP_NODELAY option)

☐ Enable TCP keepalives (SO_KEEPALIVE option)

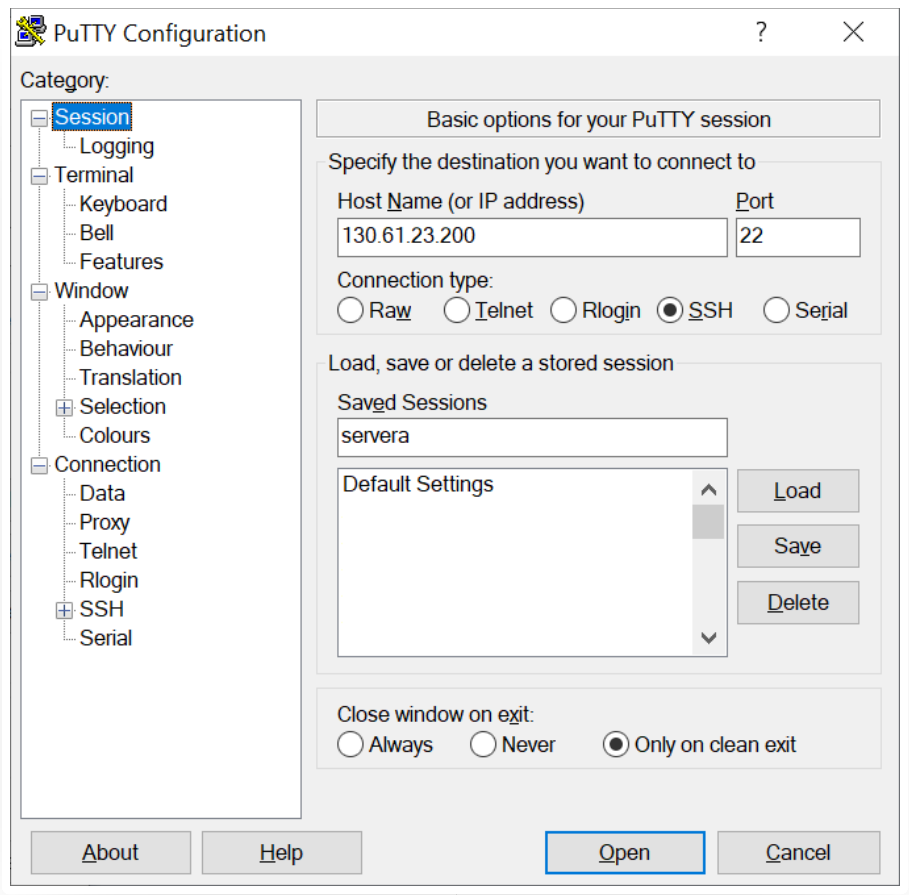
Internet protocol version

● Auto ○ IPv4 ○ IPv6

Logical name of remote host

Logical name of remote host (e.g. for SSH key lookup):

6. Return to "Session" and click save



Specify the destination you want to connect to

Host Name (or IP address): 130.61.23.200

Port: 22

Connection type: ○ Raw ○ Telnet ○ Rlogin ● SSH ○ Serial

Load, save or delete a stored session

Saved Sessions: servera

Default Settings

Close window on exit: ○ Always ○ Never ● Only on clean exit

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 curl -u vsftp_user:vsftp_Pa33w0rd  -o
workshop.tar.gz ftp://129.158.254.37
/workshop.tar.gz
```

4. Extract workshop files

🟩 **shell>**

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

5. Setup for MySQL Client

🟩 **shell>**

```
sudo ln -s /usr/lib64/libtinfo.so.6.1
/usr/lib64/libtinfo.so.5
```

Learn More

- Creating SSH Keys
- Compute SSH Connections

Acknowledgements

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

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: -- 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 Ports 3306

1. 🟩 **shell>**

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

2. 🟦 **mysql>**

   ```
   status
   ```

3. 🟦 **mysql>

```
exit
```

4. 🟦 **mysql>**

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

Task 3: Import Sample Databses

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

   🟩 **shell>**

```
cd /workshop/database/employees/
```

   🟩 **shell>**

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

Task 4: Learn Useful SQL Statements

1. 🟩 **shell>**

```
mysql -uroot -p -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
```

Learn More

(optional - include links to docs, white papers, blogs, etc)

- MySQL Tutorial
- URL text 2

Acknowledgements

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

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 -p -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: 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 -p -h 127.0.0.1 -P 3306
   ```

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

   🟦 **mysql>**

   ```
   USE employees;
   ```

   🟦 **mysql>**

```
SELECT * FROM employees;
```

2. Switch to the administrative connection revoke privilege on city to appuser

🟦 **mysql>**

```
REVOKE SELECT ON employees.* FROM
'appuser1'@'127.0.0.1';
```

🟦 **mysql>**

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

3. Repeat the select on appuser connection for the user. There is a difference?

🟦 **mysql>**

```
SELECT * FROM employees;
```

Task 3: Use appuser1 connection

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

🟦 **mysql>**

```
exit
```

🟩 **shell>**

```
mysql -u appuser1 -p -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 4: Restore user privileges

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';
```

Task 5: 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';
   ```

2. Using the Administrative Connection, create another new user and restrict the user to your "Server" IP

   a. 🟦 **mysql>**

   ```
   CREATE USER 'appuser3'@'127.0.0.1' IDENTIFIED
   BY 'Welcome1!';
   ```

   b. 🟦 **mysql>**

   ```
   GRANT ALL PRIVILEGES ON employees.* TO
   'appuser3'@'127.0.0.1';
   ```

Learn More

- CREATE USER
- MySQL Access Control Lists

Acknowledgements

- **Author** - Dale Dasker, MySQL Solution Engineering

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 stop the instance. In the lab we edit my.cnf to see how to do it in this way

Task 1: Setup Audit Log

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

   a. Edit the my.cnf setting in /mysql/etc/my.cnf

   🟩 **shell>**

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

   b. Change the line “plugin-load=thread_pool.so” to load the plugin

   🟩 **shell>**

   ```
   plugin-load=thread_pool.so;audit_log.so
   ```

   c. below the previous add these lines to make sure that the audit plugin can't be unloaded and that the file is automatically rotated at 20 MB 🟩 **shell>**

   ```
   audit_log=FORCE_PLUS_PERMANENT
   ```

   🟩 **shell>**

   ```
   audit_log_rotate_on_size=20971520
   ```

   🟩 **shell>**

   ```
   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>**

```
mysqladmin -uroot -p -h 127.0.0.1 -P3306
shutdown
```

🟩 **shell>**

```
sudo /mysql/mysql-latest/bin/mysqld
--defaults-file=/mysql/etc/my.cnf $MYSQLD_OPTS
&
```

e. Load Audit functions

🟩 **shell>**

```
mysql -uroot -p -h 127.0.0.1 -P 3306 <
/workshop/audit_log_filter_linux_install.sql
```

f. Using the Administrative Connection, create a Audit Filter for all activity and all users

🟦 **mysql>**

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

🟦 **mysql>**

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

g. 🟦 **mysql>**

```
exit
```

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

🟩 **shell>**

```
tail -f /mysql/data/audit.log
```

Task 2: Use Audit

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

   a. 🟩 **shell>**

   ```
   mysql -u appuser1 -p -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 -p -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_event
s ');
```

🟦 **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 -p -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;
```

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

a. 🟩 **shell>**

```
mysql -u appuser2 -p -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 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');
```

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

   a. 🟩 **shell>**

```
mysql -u appuser1 -p -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;
```

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

   a. Check existing filters:

   🟦 **mysql>**

```
SELECT * FROM mysql.audit_log_filter;
```

   b. Check Users being Audited:

   🟦 **mysql>**

```
SELECT * FROM mysql.audit_log_user;
```

c. Global Audit log disable

mysql>

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

10. 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, March 2022

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:

- References
- https://dev.mysql.com/doc/refman/8.0/en/innodb-data-encryption.html

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 -p -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 /mysql/etc/my.cnf

🟩 **shell>**

```
sudo nano /mysql/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=/mysql/data/mysql-
keyring/keyring-encrypted
```

```
keyring_encrypted_file_password=V&rySec4eT
```

c. Restart MySQL

🟩 **shell>**

```
mysqladmin -uroot -p -h 127.0.0.1 -P3306
shutdown
```

🟩 **shell>**

```
sudo /mysql/mysql-latest/bin/mysqld
--defaults-file=/mysql/etc/my.cnf $MYSQLD_OPTS
&
```

3. "Spy" on employees.employees table

a. 🟩 **shell>**

```
strings "/mysql/data/employees/employees.ibd"
| head -n50
```

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

   a. 🟩 **shell>**

```
mysql -u root -p -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>**

```
strings "/mysql/data/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>**

```
strings "/mysql/data/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>**

```
strings "/mysql/data/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

- Keyring Plugins
- InnoDB Data At Rest