ORACLE

MySQL Enterprise Edition

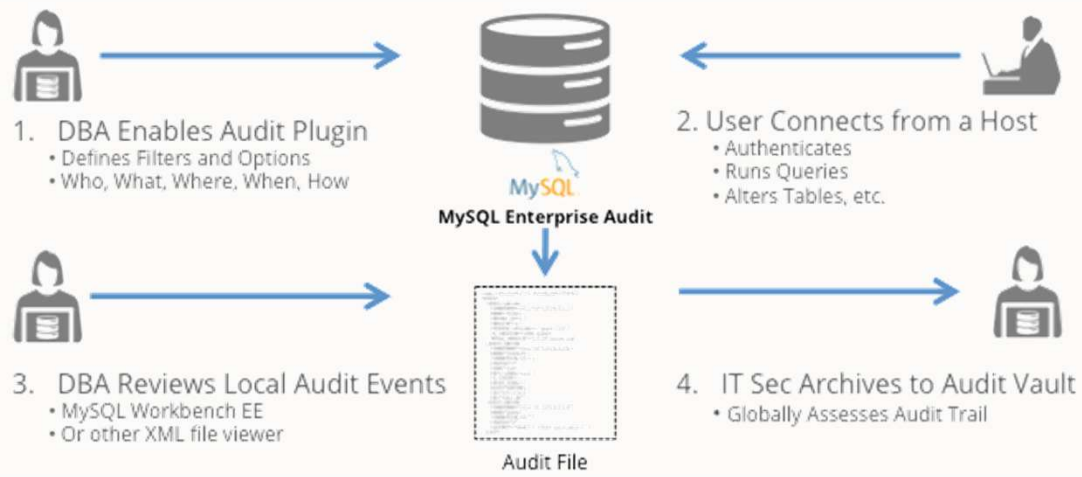
Enterprise Audit

Stuart Davey
MySQL Principal Soultions Engineer
EMEA MySQL GBU
30 January 2024

MySQL Enterprise Audit Overview

Easy to use, policy based to provide a high degree of granularity

Encryption and compression are easily applied



Considerations

Auditing can produce a lot of output

- Consider only logging actions that are associated with sensitive information
  - Use **filters** to achieve the correct granularity for your requirement
- Consider log rotation
  - Manual or Automatic
  - Automatic log rotation is not circular – new files will be added
- Consider moving closed audit log files to a secure store or vault (AWS S3 Glacier?)
- Consider compression

Audit logs should only be read by authorized personnel

- Set appropriate permissions on the audit log directory
- Limit MySQL users who have `AUDIT_ADMIN` privilege (for example, only allow `root@localhost`)
- Consider encryption

Audit logs often need to be retained for compliance

- Consider format: XML (old or new style) or JSON
- Consider moving closed audit log files to a secure store or vault (AWS S3 Glacier?)

Enabling Audit

Audit is enabled by loading a plugin

- This is done by using a script
  - **`audit_log_filter_<OS-TYPE>_install.sql`**
  - For example: **`audit_log_filter_linux_install.sql`**
  - Found in the MySQL share directory or folder
- The script installs audit functions as well as the plugin
  - Audit should only be accessed by these functions and not by the underlying tables using SQL
- Loading is done dynamically
  - it does not require a server restart
  - …but you may want to restart to ensure you use the following variables (in my.cnf)

```
audit-log=FORCE_PLUS_PERMANENT            # Prevents unloading audit, server will not start without audit
audit_log_file=/var/lib/mysql-audit/audit.log # DBA defined directory (otherwise audit.log will be
                                              # written to the data directory.
                                              # Ensure correct ownership & permissions for
                                              # directory  or folder – on Linux
                                              #  mysql:mysql and rwx -rw ---
```

Creating, Applying and Removing Filters

https://dev.mysql.com/doc/refman/8.0/en/audit-log-filter-definitions.html

We create a filter using the function:

```
SELECT audit_log_filter_set_filter('<user-defined-filter-name>', {"filter": { … } });
```

Then a apply the filter to a user or users (we can use wildcards (e.g. %) in user names)

```
SELECT audit_log_filter_set_user('<mysql-user-name>', '<user-defined-filter-name>');
```

The filter is described in JSON:

- It has the keyword `filter` and then a one or more values / JSON objects
  - These describe what will be logged, e.g. connections, table accesses, etc.
  - The following logs everything (which is probably too much!)

```
{
    "filter": {
        "log": true
    }
}
```

```
SELECT audit_log_filter_set_filter('log-all',{"filter":{"log":true}});
SELECT audit_log_filter_set_user('log-all', '%');
```

We can remove users and remove filters

```
SELECT audit_log_filter_remove_user('<mysql-user-name>');
SELECT audit_log_filter_remove_filter('<user-defined-filter-name>');
```

Blocking Filters

https://dev.mysql.com/doc/refman/8.0/en/audit-log-filter-definitions.html#audit-log-filtering-blocking-events

Prevention is better than cure!

Filter is created as before, but an “abort” key is added to the “event” object

If an attempt is made to access the object by a user to whom the filter applies then an abort is issued by the database (i.e. the request is not executed, but is still logged)

```
"filter": {
   "class": {
     "name": "table_access",
     "event": {
       "name": [ "read", "insert", "update", "delete" ],
       "abort": {
         "and": [
           { "field": { "name": "table_database.str", "value": "taxdb" } },
           { "field": { "name": "table_name.str", "value": "taxpaid_vip" } }
         ]
       }
     }
   }
 }
```

Formats and Reading Audit Logs

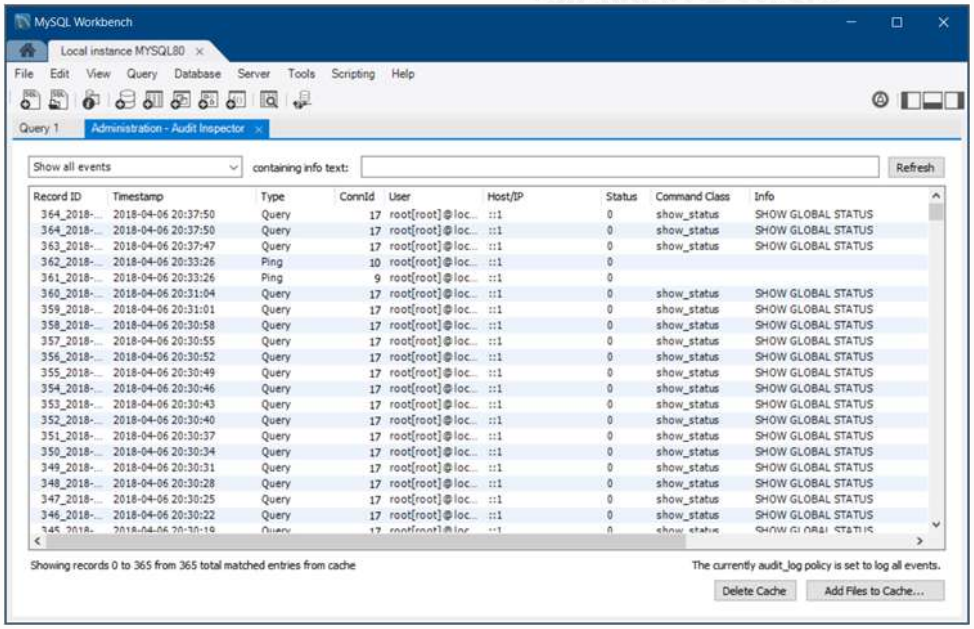
https://dev.mysql.com/doc/refman/8.0/en/audit-log-file-formats.html

- Three formats exist
  1. NEW (new style XML format – the default)
  2. OLD (new style XML format)
  3. JSON
- Easy for programs to parse
- Set the format in my.cnf (below), then restart the server

```
audit_log_format=JSON
```

- The plugin provides a function to allow the reading of JSON formatted logs within MySQL

- Otherwise use Workbench, Oracle Audit Vault or a third party tool to read the log files.



MySQL Workbench – audit inspector

https://dev.mysql.com/doc/workbench/en/wb-audit-inspector.html

Log Rotation

For full details: https://dev.mysql.com/doc/refman/8.0/en/audit-log-logging-configuration.html

Manual by default

- The variable `audit_log_rotate_on_size=0`
- To rotate
  1. Rename the current log file (logging to this file will continue),

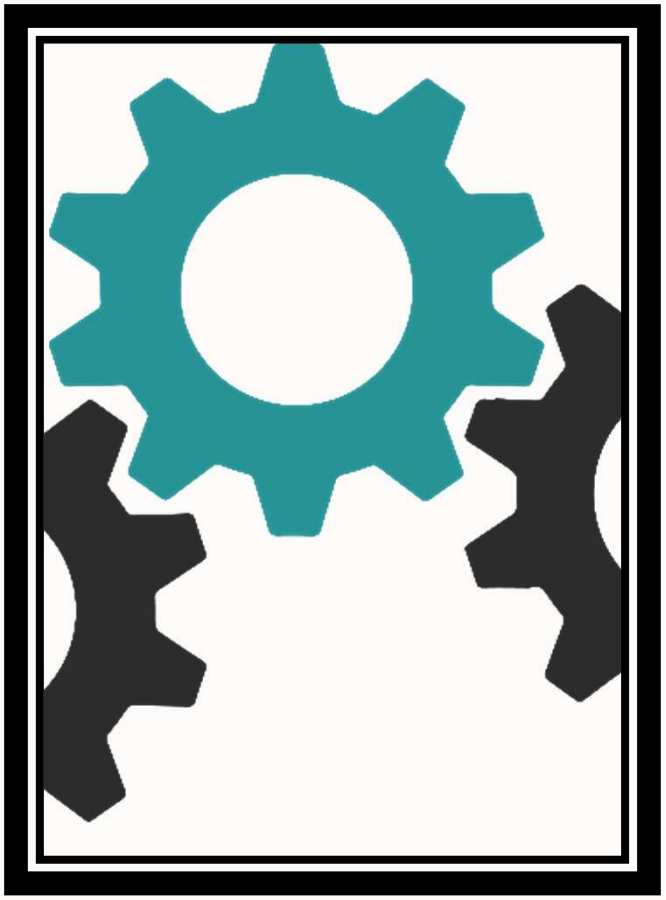
     For example on the OS command line: `mv audit.log audit.1`
  2. Flush the audit log – this will create a new log file and direct all new audit log events to it

     For example in MySQL: `SET audit_log_flush=1;`

     After the flush happens, audit_log_flush is automatically reset to 0; you don't have to do anything

Automatic log rotation

- Set the variable `audit_log_rotate_on_size` to a size between 4096 and 2^{64}
  - The size should be a multiple of 4096 if you set it to ((3 x 4096) – 10) the actual size will be (2 x 4096) = 8192
  - If you set it to less than 4096 it will be truncated to 0 – no automatic log rotation
- Rotation occurs
  - When the rotate size is reached
  - When the `audit_log_encryption_password_set()` function is called to set the encryption password
  - On both plugin initialization and termination



Demonstration: Audit

Compression and Encryption

https://dev.mysql.com/doc/refman/8.0/en/audit-log-logging-configuration.html

By default compression is off

- The variable which governs compression is: `audit_log_compression`
- Default setting is `NONE` (compression off)
- To turn compression on set `audit_log_compression=GZIP` in my.cnf, then **restart the server**

By default encryption is off

- To turn encryption on we must use a MySQL keyring to provide a secure password
- We can use the same keyring that TDE uses
- To configure encryption set `audit_log_encryption=AES` in my.cnf, then **restart the server**
  - Encryption cypher will be AES-256-CBC
- To use encryption we must set a password, in MySQL use the following MySQL audit function

```
SELECT audit_log_encryption_password_set(password);
```

- This will rotate the audit log, the new audit log will then receive encrypted data
- Related function: `audit_log_encryption_password_get()`

Manually Uncompressing and Decrypting Audit Logs

https://dev.mysql.com/doc/refman/8.0/en/audit-log-logging-configuration.html

Caution: only do this on closed logs!

A note on file names

Uncompressed, unencrypted	audit.<timestamp>.log
Compressed	audit.<timestamp>.log.gz
Encrypted	audit.<timestamp>.log.<pwd_id>.enc
Compressed and encrypted	audit.<timestamp>.log.gz.<pwd_id>.enc

To uncompress a file

```
gunzip -c audit.timestamp.log.gz > audit.timestamp.log
```

To decrypt a file (with the filename, audit.20190415T151322.log.20190414T223342-2.enc)

1. In MySQL: `SELECT audit_log_encryption_password_get('audit-log-20190414T223342-2');`
2. Take the returned password and on the command line run

```
openssl enc -d -aes-256-cbc -pass pass:<password> -md sha256 \
-in audit.20190415T151322.log.20190414T223342-2.enc \
-out audit.timestamp.log.gz
```

More advanced filters

It is possible to combine classes by creating a JSON array:

```
SET @my_filter='{
  "filter": {
     "class": [{
         "name": "connection",
         …
      },{
         "name": "general",
         …
      }]
    }]
  }
}';
```

And then apply as before

```
# first check that your JSON is valid
mysql> SELECT json_valid(@my_filter);
# make the filter available for use
mysql> SELECT audit_log_filter_set_filter('filter_something',@my_filter);
# assign the filter to users
mysql> SELECT audit_log_filter_set_user('isabel@%','filter_something');
mysql> SELECT audit_log_filter_set_user('daniel@%','filter_something');
```

More advanced filters – logging on and performing DDL

```
SET @log_cnx_ddl_filter='{
  "filter": {
    "class": [{
         "name": "connection",
         "log": true
       },{
      "name": "general",
      "event": {
        "name": "status",
        "log": {
          "and": [{
            "or": [
              {"field": { "name": "general_command.str", "value": "Query" }},
              {"field": { "name": "general_command.str", "value": "Execute" }}
            ]
          },{
            "or": [
              {"field": { "name": "general_sql_command.str", "value": "alter_db" }},
              {"field": { "name": "general_sql_command.str", "value": "alter_db_upgrade" }},
              {"field": { "name": "general_sql_command.str", "value": "alter_event" }},
              {"field": { "name": "general_sql_command.str", "value": "alter_function" }},
              {"field": { "name": "general_sql_command.str", "value": "alter_instance" }},
              {"field": { "name": "general_sql_command.str", "value": "alter_procedure" }},
              {"field": { "name": "general_sql_command.str", "value": "alter_server" }},
              {"field": { "name": "general_sql_command.str", "value": "alter_table" }},
              {"field": { "name": "general_sql_command.str", "value": "alter_table" }},
              {"field": { "name": "general_sql_command.str", "value": "alter_tablespace" }},
```

More advanced filters – logging on and performing DDL (continued)

```
                {"field": { "name": "general_sql_command.str", "value": "create_db" }},
                {"field": { "name": "general_sql_command.str", "value": "create_event" }},
                {"field": { "name": "general_sql_command.str", "value": "create_function" }},
                {"field": { "name": "general_sql_command.str", "value": "create_index" }},
                {"field": { "name": "general_sql_command.str", "value": "create_procedure" }},
                {"field": { "name": "general_sql_command.str", "value": "create_server" }},
                {"field": { "name": "general_sql_command.str", "value": "create_table" }},
                {"field": { "name": "general_sql_command.str", "value": "create_trigger" }},
                {"field": { "name": "general_sql_command.str", "value": "create_udf" }},
                {"field": { "name": "general_sql_command.str", "value": "create_view" }},
                {"field": { "name": "general_sql_command.str", "value": "drop_db" }},
                {"field": { "name": "general_sql_command.str", "value": "drop_event" }},
                {"field": { "name": "general_sql_command.str", "value": "drop_function" }},
                {"field": { "name": "general_sql_command.str", "value": "drop_index" }},
                {"field": { "name": "general_sql_command.str", "value": "drop_procedure" }},
                {"field": { "name": "general_sql_command.str", "value": "drop_server" }},
                {"field": { "name": "general_sql_command.str", "value": "drop_table" }},
                {"field": { "name": "general_sql_command.str", "value": "drop_trigger" }},
                {"field": { "name": "general_sql_command.str", "value": "drop_view" }},
                {"field": { "name": "general_sql_command.str", "value": "rename_table" }}
              ]
            }]
          }
        }
      }]
    }
}';
```

ORACLE

Demonstration

Test data: auditdemo.sql – page1 of 2

```
# Copy the code in this page and the next and then paste it into a file called auditdemo.sql
DROP DATABASE IF EXISTS db1;
CREATE DATABASE db1;
USE db1;
CREATE TABLE t1 (id INT AUTO_INCREMENT PRIMARY KEY, name VARCHAR(20));
INSERT INTO t1 (name) VALUES ('Fred');
INSERT INTO t1 (name) VALUES ('Wilma');
INSERT INTO t1 (name) VALUES ('Betty');
INSERT INTO t1 (name) VALUES ('Barney');
DROP DATABASE IF EXISTS taxdb;
CREATE DATABASE taxdb;
USE taxdb;
CREATE TABLE taxpayers (nino INT AUTO_INCREMENT PRIMARY KEY, name VARCHAR(20) NOT NULL);
CREATE TABLE taxpaid_genpub(id INT AUTO_INCREMENT PRIMARY KEY, nino INT NOT NULL, paid INT DEFAULT 0, owing INT DEFAULT 0, comment VARCHAR(30) DEFAULT '',
FOREIGN KEY (nino) REFERENCES taxpayers(nino));
CREATE TABLE taxpaid_vip(id INT AUTO_INCREMENT PRIMARY KEY, nino INT NOT NULL, paid INT DEFAULT 0, owing INT DEFAULT 0, comment VARCHAR(30) DEFAULT '',
FOREIGN KEY (nino) REFERENCES taxpayers(nino));
```

Demonstration

Test data: auditdemo.sql – page 2 of 2

```
INSERT INTO taxpayers (name) VALUES ('Stuart');
INSERT INTO taxpayers (name) VALUES ('Claire');
INSERT INTO taxpayers (name) VALUES ('HM the Queen');
INSERT INTO taxpayers (name) VALUES ('Ken Dodd');
INSERT INTO taxpayers (name) VALUES ('James Bond');
INSERT INTO taxpayers (name) VALUES ('Jim');
INSERT INTO taxpaid_genpub (nino,paid,owing) VALUES(1,100,0);
INSERT INTO taxpaid_genpub (nino,paid,owing) VALUES(2,200,0);
INSERT INTO taxpaid_genpub (nino,paid,owing) VALUES(6,200,0);
INSERT INTO taxpaid_vip (nino,paid,owing,comment) VALUES(3,0,2000000,'Granted royal pardon');
INSERT INTO taxpaid_vip (nino,paid,owing,comment) VALUES(4,0,100000,'Beyond our reach (for now)');
INSERT INTO taxpaid_vip (nino,paid,owing) VALUES(5,5000,0);
DROP USER IF EXISTS 'stuart'@'%' ;
CREATE USER 'stuart'@'%' IDENTIFIED BY 'Welcome1!';
DROP USER IF EXISTS 'claire'@'localhost' ;
CREATE USER 'claire'@'localhost' IDENTIFIED BY 'Welcome1!';
GRANT SELECT,CREATE,INSERT,UPDATE,DELETE ON db1.* TO 'stuart'@'%';
GRANT SELECT,CREATE,INSERT,UPDATE,DELETE ON taxdb.* TO 'stuart'@'%';
GRANT SELECT,CREATE,INSERT,UPDATE,DELETE ON taxdb.* TO 'claire'@'localhost';
```

Demonstration

1. Set up the database for the demo

```
# Load auditdemo.sql into the MySQL database server
mysql -uroot -p < auditdemo.sql
# Now log in and check the data has been loaded, the users are in place and have the desired permissions (grants)
mysql -uroot -p
Simp50n5!
USE db1;
SELECT * FROM t1;
USE taxdb;
SELECT * FROM taxpayers;
SELECT * FROM taxpaid_genpub;
SELECT * FROM taxpaid_vip;
SELECT t.name, t.nino, g.paid, g.owing, g.comment FROM taxpayers t JOIN taxpaid_genpub g ON g.nino WHERE t.nino = g.nino ORDER BY nino;
SELECT t.name, t.nino, v.paid, v.owing, v.comment FROM taxpayers t JOIN taxpaid_vip v ON v.nino WHERE t.nino = v.nino ORDER BY nino;
SELECT t.name, t.nino, g.paid, g.owing, g.comment FROM taxpayers t JOIN taxpaid_genpub g ON g.nino WHERE t.nino = g.nino UNION SELECT t.name, t.nino, v.paid, v.owing,
v.comment FROM taxpayers t JOIN taxpaid_vip v ON v.nino WHERE t.nino = v.nino ORDER BY nino;
SELECT user, host FROM mysql.user;
SHOW GRANTS FOR stuart@'%';
SHOW GRANTS FOR claire@localhost;
quit;
```

Demonstration

2. Install MySQL Audit

```
# Find the MySQL share directory – this is install specific, e.g. for a Linux RPM install the directory will be /usr/share/mysql-8.0
ls /usr/share/mysql-8.0/*.sql
# Load the audit SQL file, then login
mysql -uroot -p < /usr/share/mysql-8.0/audit_log_filter_linux_install.sql
mysql -uroot -p
SELECT PLUGIN_NAME, PLUGIN_STATUS FROM INFORMATION_SCHEMA.PLUGINS WHERE PLUGIN_NAME LIKE 'audit%';
quit;
# In another terminal edit the configuration file, my.cnf, in order to force the load of audit and identify where the log will be written
sudo vi /etc/my.cnf
# Audit - FORCE_PLUS_PERMANENT means audit cannot be unloaded and the server won't start without it.
audit-log=FORCE_PLUS_PERMANENT
audit_log_file=/var/lib/mysql-audit/audit.log
# Perform some OS administration so that the log can be written by MySQL
sudo mkdir -p /var/lib/mysql-audit
sudo chown mysql:mysql /var/lib/mysql-audit
sudo chmod 0750 /var/lib/mysql-audit
# Restart the server to pick up the changes in the my.cnf file.
sudo systemctl restart mysqld
```

Demonstration

3. Setup basic filtering and enter data

```
# Log back into mysql and set up logging to log everything for every user:
mysql -uroot -p
SELECT audit_log_filter_set_filter('log_all', '{ "filter": { "log": true } }');
SELECT audit_log_filter_set_user('%', 'log_all');
# Do some work as various users
USE taxdb;
SELECT * FROM taxpayers;
quit;
mysql -ustuart -p
USE taxdb;
SELECT * FROM taxpayers;
SELECT t.name, t.nino, g.paid, g.owing, g.comment FROM taxpayers t JOIN taxpaid_genpub g ON g.nino WHERE t.nino = g.nino ORDER BY nino;
SELECT t.name, t.nino, v.paid, v.owing, v.comment FROM taxpayers t JOIN taxpaid_vip v ON v.nino WHERE t.nino = v.nino ORDER BY nino;
quit;
```

Demonstration

3 (continued). Enter more data then check the audit log

```
mysql -uclaire -p
USE taxdb;
SELECT * FROM taxpayers;
SELECT t.name, t.nino, g.paid, g.owing, g.comment FROM taxpayers t JOIN taxpaid_genpub g ON g.nino WHERE t.nino = g.nino ORDER BY nino;
SELECT t.name, t.nino, v.paid, v.owing, v.comment FROM taxpayers t JOIN taxpaid_vip v ON v.nino WHERE t.nino = v.nino ORDER BY nino;
quit;
# In another terminal, check the log file
sudo ls -l /var/lib/mysql-audit
sudo cat /var/lib/mysql-audit/audit.log                # Log is currently in the clear; it can be encrypted and compressed
# Perform a manual log rotation
sudo mv /var/lib/mysql-audit/audit.log /var/lib/mysql-audit/audit.log.1
# Log back into MySQL and flush the log, then do some more work
mysql -uroot -p
SET GLOBAL audit_log_flush = ON;
quit;
# Back in the linux terminal - look at the logs and their timestamps audit.log is the new file, audit.log.1 is the older file
sudo ls -l /var/lib/mysql-audit
sudo cat /var/lib/mysql-audit/audit.log
sudo cat /var/lib/mysql-audit/audit.log.1
```

Demonstration

4. Create a blocking filter

```
# Create a filter that will expose users who are illegally checking on the tax affairs of VIPs
mysql -uroot -p
SET @filter = '{
  "filter": {
    "class": {
      "name": "table_access",
      "event": {
        "name": [ "read", "insert", "update", "delete" ],
        "abort": {
          "and": [
            { "field": { "name": "table_database.str", "value": "taxdb" } },
            { "field": { "name": "table_name.str", "value": "taxpaid_vip" } }
          ]
        }
      }
    }
  }
}';
```

Demonstration

4 (continued). Triggering the blocking filter

```
SELECT @filter;
SELECT audit_log_filter_set_filter('block_all_on_table_taxpaid_vip',@filter);
SELECT audit_log_filter_set_user('stuart@%','block_all_on_table_taxpaid_vip');
quit;
# Log in as stuart and run some queries against the taxdb tables
mysql -ustuart -p
USE taxdb;
SELECT * FROM taxpayers;
SELECT * FROM taxpaid_genpub;
SELECT t.name, t.nino, g.paid, g.owing, g.comment FROM taxpayers t JOIN taxpaid_genpub g ON g.nino WHERE t.nino = g.nino ORDER BY nino;
SELECT t.name, t.nino, v.paid, v.owing, v.comment FROM taxpayers t JOIN taxpaid_vip v ON v.nino WHERE t.nino = v.nino ORDER BY nino;
SELECT * FROM taxpaid_vip;
quit;
# Perform a manual log rotation then check the audit file
sudo mv /var/lib/mysql-audit/audit.log /var/lib/mysql-audit/audit.log.2
# Log back into MySQL and flush the log, then do some more work
mysql -uroot -p!
SET GLOBAL audit_log_flush = ON;
# Now go and have a look at the archived file in another terminal – it should show stuart accessing confidential data!
sudo vi /var/lib/mysql-audit/audit.log.2
```

Demonstration

5. Automatic log rotation

```
# add audit_log_rotate_on_size=8196 to config file (my.cnf) - unrealistically small in order to quickly demonstrate rotation!
sudo vi /etc/my.cnf
audit_log_rotate_on_size=8196
# restart the server
sudo systemctl restart mysqld
# log back in and do some more work
mysql -ustuart -p
USE db1;
SELECT * FROM t1;
INSERT INTO t1 (name) VALUES ('Homer');
INSERT INTO t1 (name) VALUES ('Marge');
INSERT INTO t1 (name) VALUES ('Bart');
INSERT INTO t1 (name) VALUES ('Lisa');
INSERT INTO t1 (name) VALUES ('Maggie');
SELECT * FROM t1;
DELETE FROM t1 WHERE id > 4;
quit;
```

Demonstration

5 (continued) Automatic logging

```
# do some more work as another user
mysql -uclaire -p
USE taxdb;
SELECT * FROM taxpayers;
SELECT * FROM taxpaid_genpub;
SELECT * FROM taxpaid_vip;
SELECT t.name, t.nino, g.paid, g.owing, g.comment FROM taxpayers t JOIN taxpaid_genpub g ON g.nino WHERE t.nino = g.nino UNION SELECT t.name, t.nino, v.paid, v.owing, v.comment FROM taxpayers t JOIN taxpaid_vip v ON v.nino WHERE t.nino = v.nino ORDER BY nino;
quit;
# Now look at the log files – you should see new files created
sudo ls -l /var/lib/mysql-audit
```