# CSIT115 Data Management and Security

# **Database Auditing**

Dr Janusz R. Getta

School of Computing and Information Technology - University of Wollongong

### **Database Auditing**

#### Outline

Concepts

**Auditing Categories** 

Database Auditing in MySQL

#### **Concepts**

Database auditing involves observing a database so as to be aware of the actions of database users

Database administrators and consultants often set up auditing for security purposes, for example, to ensure that those without the permission to access information do not access it (Wikipedia: https://en.wikipedia.org/wiki/Database\_audit)

Database activity monitoring (DAM) is a database security technology for monitoring and analyzing database activity that operates independently of the database management system (DBMS) and does not rely on any form of native (DBMS-resident) auditing or native logs such as trace or transaction logs. DAM is typically performed continuously and in real-time (Wikipedia:

https://en.wikipedia.org/wiki/Database\_activity\_monitoring)

#### **Concepts**

An audit trail (also called as audit log) is a security-relevant chronological record, set of records, and/or destination and source of records that provide documentary evidence of the sequence of activities that have affected at any time a specific operation, procedure, or event (Wikipedia: https://en.wikipedia.org/wiki/Audit\_trail)

### **Database Auditing**

Outline

Concepts

**Auditing Categories** 

Database Auditing in MySQL

#### Audit logon/logoff into a database

- Record two events: an event for sign-on and an event for sign-off
- Save login name, timestamps, TCP/IP address of client, and a program used to initiate connection
- Record failed login attempts

#### Audit sources of database usage

- It includes which network node a user is connected to (IP address and host name)
- Which application is being used to access a database

6/36

#### Audit database activities outside normal operation hours

- Activities performed outside normal operating hours are usually suspect and maybe the result of unauthorized access to modify data
- Audit off-hours activities include logons/logoffs and all SQL activities
- Audit off-hours activities do not need to include the activities that are always scheduled to run off-hours. e.g. Extract, Transform, Load (ETL) data warehousing activities

#### **Audit DDL activities**

- DDL commands of SQL are potentially the most damaging commands that exists
- Auditing DDL activities is also done to eliminate errors of developers and database administrators
- Auditing database schemas changes can be done by: (1) using database audit features, (2) using external auditing system, (3) comparison of schema snapshots, (4) some system allow for using database triggers

#### Audit database errors

- Auditing database errors is important because in many cases hackers use a technique of "trial-and error" to investigate a structure of a database, a well written and tested database application does not return errors
- Failed logins is a good example of errors that must be monitored
- Audit database errors may also lead to identification of weak points in database applications

#### Audit changes to sources of stored procedures and triggers

- Change to sources of already developed software may mean the attempts to incorporate malicious code
- Audit can be implemented by comparison of source code developed earlier with the present one, e.g. using diff program
- The second option is to use external database security and auditing system, which alerts on any modify or create command used by a database user
- The third option is to use built in database features, for example ability of a system to trace "recompile" events to track changes to stored procedure and triggers

#### Audit changes to privileges, user logins and other security attributes

- Audit deletion and addition of users, logins, and role
- Audit changes to the mapping between logins and users/roles
- Audit privilege changes over user and roles
- Audit password changes
- Audit changes to security attributes at a server, database, statement, or object level

#### Audit creation, changes, and usage of database links and replication

- Simple implementations using daily differences
- Comparing snapshots
- Using database internal mechanisms

Audit changes to sensitive data (data change audit trails to enforce accuracy in financial data)

- Fully recording "old" and "new" values for each DML activity
- Due to the large number of DML operations the audits must be done very selectively
- Carefully chose data objects to be audited
- Use database system capabilities, external audit systems, or database triggers

#### Audit Select statements for privacy sets

- Record where SELECT statements come from
- Who (username) retrieved the data
- What data was actually retrieved
- Identify so called privacy sets, i.e. which associations of data are really important
- Implementation uses database traces and external auditing systems

#### Audit any changes made to the definitions of what to audit

- Audit changes made to the definitions of audit trails and changes to the audit trails themselves
- Use built-in database features and external auditing systems

### **Database Auditing**

Outline

Concepts

**Auditing Categories** 

Database Auditing in MySQL

MySQL Enterprise Audit (plugin not available within Community Edition)

- Dynamically enable/disable audit stream
- Implement policies that log all or selected login or query activities
- Automatically rotate audit log files based on size
- Integrate XML-based audit log stream with MySQL, Oracle and other third party solutions

Using stored procedures

Using database triggers

External audit plugin libraries

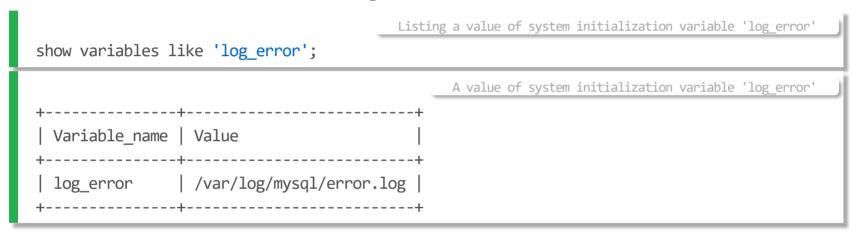
Analysis of database logs

#### Using database logs

- MySQL Server has several logs that contain information about the user activities
- Error log: problems encountered starting, running, or stopping MySQL server
- General query log: Established client connections and statements received from clients
- Binary log: Statements that change data (also used for replication)
- Relay log: Data changes received from a replication master server
- Slow query log: Queries that took more than a given period of time to execute
- DDL log (metadata log): Metadata operations performed by DDL statements

#### Error log

- Error log contains information indicating when database server was started and stopped and also any critical errors that occur while the server is running
- A system variable log\_error determines a location a file error.log with information included in error log



#### Error log

- It is possible to list the contents of a file error.log in the following way

```
Listing the contents of Error log
cat /var/log/mysql/error.log | more
                                                                    Sample contents of error log
151110 14:44:15 mysqld safe Starting mysqld daemon with
                           databases from /var/lib/mysql
2015-11-10T03:44:15.916141Z 0 [Warning] Changed limits: max open files: 1024
(requested 5000)
2015-11-10T03:44:15.916275Z 0 [Warning] Changed limits: table open cache: 431
(requested 2000)
2015-11-10T03:44:16.831947Z 0 [Warning] 'NO ZERO DATE', 'NO ZERO IN DATE' and
 'ERROR FOR DIVISION BY ZERO' sql modes should be used with strict mode.
They will be merged with strict mode in a future release.
2015-11-10T03:44:16.832003Z 0 [Warning] 'NO AUTO CREATE USER' sql mode was not set.
2015-11-10T03:44:16.837216Z 0 [Note] /usr/sbin/mysqld (mysqld 5.7.9) starting as
process 22405.
```

#### General query log

- A system variable general\_log controls logging to general query log

```
show variable like 'general_log';

A value of system initialization variable 'general_log'

+----+

| Variable_name | Value |
+----+

| general_log | OFF |
+----+
```

- To start logging execute the following statement

```
Setting a value of system initialization variable 'general_log'

set global general_log='ON';

A value of system initialization variable 'general_log'

+-----+

| Variable_name | Value |

+-----+

| general_log | ON |

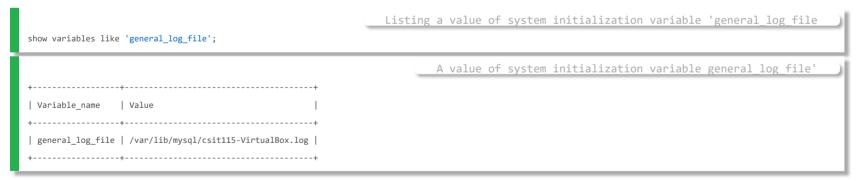
+-----+
```

#### General query log

- General query log is written either to a file or to a relational table
- A variable log\_output determines whether general query log is written either to a file or to a relational table

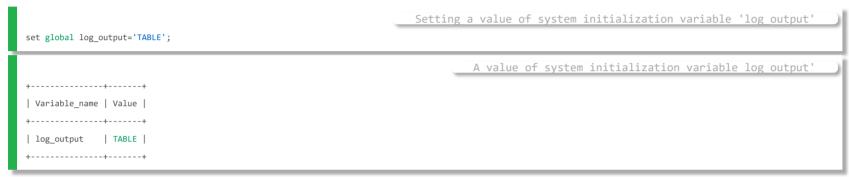


- A name of file for a general query log is determined by a variable general log file

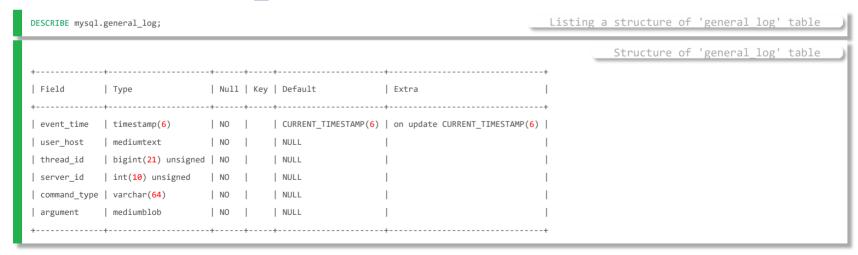


#### General query log

- To redirect general query log to a relational table we change a value of variable log output in the following way:



 General query log is recorded in a relational table mysql.general\_log



#### General query log

To remove old contents of general query log execute the following TRUNCATE TABLE statement

TRUNCATE TABLE mysql.general\_log;

TRUNCATE TABLE statement is surprisingly categorized as DDL statement and it removes the contents of a relational table forever It means that TRUNCATE TABLE statement cannot be reversed with ROLLBACK statement

To find the contents of general query log execute SELECT statement

#### General query log

- Note, that when MySQL server is shutdown and restarted all values of system variables return to its default values or values set up in the system configuration file
- It means that after shutdown and restart the values of variables like general\_log, log\_output, general\_log\_file return to their original values
- To stop writing into general query log execute a statement

#### Slow query log

- The slow query log consists of SQL statements that took more than long\_query\_time seconds to execute and required at least min\_examined\_row\_limit rows to be examined
- The minimum and default values of long\_query\_time are 0 and 10, respectively

#### Slow query log

The value can be specified to a resolution of microseconds

For logging to a file, times are written including the microseconds part

For logging to tables, only integer times are written; the microseconds

part is ignored

#### Slow query log

- To change a value of long query time use the following statement

#### Slow query log

- A system variable slow\_query\_log controls logging to slow query log

```
Listing a value of system initialization variable 'slow_query_log'
show variables like 'slow_query_log';

A value of system initialization variable 'slow_query_log'

+-----+

| Variable_name | Value |
+-----+

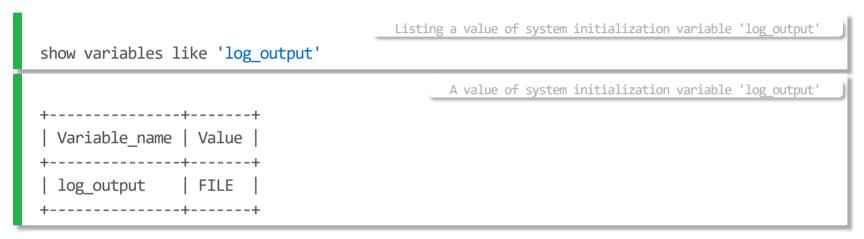
| slow_query_log | OFF |
+-----+
```

#### Slow query log

- To start logging execute the following statement

#### Slow query log

- Slow query log is written either to a file or to a relational table
- A variable log\_output determines whether general query log is written either to a file or to a relational table



#### Slow query log

- A name of file for a slow\_query\_log is determined by a variable slow query log file

#### Slow query log

- To redirect slow query log to a relational table we change a value of variable log output in the following way:

```
Listing a value of system initialization variable 'log_output'

set global log_output='TABLE'

A value of system initialization variable 'log_output

+----+

| Variable_name | Value |
+----+

| log_output | TABLE |
+----+
```

#### Slow query log

- Slowquery log is recorded in a relational table mysql.slow log



#### Slow query log

- To remove old contents of slow query log execute the following TRUNCATE TABLE statement

```
TRUNCATE TABLE mysql.slow_log;

TRUNCATE TABLE statement
```

- TRUNCATE TABLE statement is surprisingly categorized as DDL statement and it removes the contents of a relational table forever
- It means that TRUNCATE TABLE statement cannot be reversed with ROLLBACK statement
- To find the contents of some columns from slow log execute SELECT statement

#### Slow query log

- Note, that when MySQL server is shutdown and restarted all values of system variables return to its default values or values set up in the system configuration file
- It means that after shutdown and restart the values of variables like long\_query\_time, min\_examined\_row\_limit, slow\_query\_log, log\_output, slow\_query\_log\_file return to their original values
- To stop writing into slow query log execute a statement

#### Binary log

- Binary log contains "events" that describe database changes such as table creation operations or changes to table data
- It also contains events for statements that potentially could have made changes (for example, a DELETE which matched no rows)
- Binary log also contains information about how long took the computations of UPDATE statements
- Binary log has two important purposes:
  - it is used for replication, the binary log on a master replication server provides a record of the data changes to be sent to slave servers
  - it is used for data recovery operations
- Binary log is not used for statements such as SELECT or SHOW that do not modify data
- To log all statements (for example, to identify a problem query), use general query log

#### **DDL** log

- DDL log, or metadata log, records metadata operations generated by data definition statements such as CREATE TABLE, DROP TABLE, and ALTER TABLE
- MySQL uses this log to recover from crashes occurring in the middle of a metadata operation
- When executing the statement DROP TABLE t1, t2, we need to ensure that both t1 and t2 are dropped, and that each table drop is complete

#### References

MySQL 5.7 Reference Manual, MySQL Server Administration, MySQL Server logs