IJoFCS (2017) 1, 8-17

e-forensic PRESS

The International Journal of FORENSIC COMPUTER SCIENCE

www.IJoFCS.org

DOI: 10.5769/J201701001 or http://dx.doi.org/10.5769/J201701001

A Methodology to Test the Richness of Forensic Evidence of Database Storage Engine: Analysis of MySQL Update Operation in InnoDB and MyISAM Storage Engines

James O. Ogutu[1], Elisha O. Abade[2]

(1) University of Nairobi, Email: ogutorengo@gmail.com
(2) University of Nairobi, Email: elisha.abade@uonbi.ac.ke
Website: www.uonbi.ac.ke

Abstract: Digital forensic investigation requires forensic evidence data to prove a claimed crime. With the possibility of performing database forensic as a file system coupled with the fact that there are several storage engines that can be implemented in a database, there is need to know the forensic implication of using a particular storage engine with focus on how much forensic footprint it leaves behind. This work investigated the impact of MyISAM and InnoDB storage engines in generation of persistent forensic data in MySQL DBMS system. A comparison was done on the number of logs and files affected by an update operation in MySQL DBMS implementing either of the storage engines by comparing file metadata before and after UPDATE operation. It was found that more files were affected in InnoDB than in MyISAM implementation.

Key words: Artifacts, Digital evidence, Database Management System, Relational Database Management System, Metadata

I. Introduction

The need for an organized manner of storing information belonging to an organization was the motivation for coming up with the concept of a database. Databases have evolved from simple lists of items to more complex transactional databases that are run on complex computer systems. A database can be defined as: "persistent, logically coherent collection of inherently meaningful data, relevant to some aspect of the real world" [1].

Databases play a pivotal role in businesses as they store variety of data from asset inventory to orders and financial transactions. With the up surge of databases in many business applications, there comes the need to maintain integrity, consistency and reliability of the data stored in the databases. Some of these functions have been integrated in the software systems that are used to run database applications which restrict access by requiring user to authenticate using passwords. These applications are called database management systems (DBMS). However, DBMSs can only ensure integrity, consistency and reliability with the assumption that the system is configured properly and authorized users use the system in the authorized way. But because this may not always happen in real world, there has been need to provide extra measures to mitigate the effect of unauthorised

Paper submitted on: March 26th, 2017

manipulation of the database by authorized or authorized database user acting maliciously[2, 3]. The role of database security expert then comes in to ensure that the data contained in the database is reliable and protected from unauthorized access and manipulation, and in the event that there is a disputed or unlawful manipulation, then an acceptable procedure and evidence can be used to prove that the claimed violation actually took place. The act of proving a past occurrence in database using acceptable data evidence constitutes database forensics. The aim of database forensics is to find out what happened when and to prevent unauthorized data manipulation [2].

Analysis of architectural components of a database system is critical for the understanding of database forensic analyst in which a forensic examiner identifies areas to get evidence about a past activity in a database. Some of the areas in a database where evidence about past activities can be found include metadata, data files; redo logs, transaction logs and storage engines [5]. With the constant security threat to databases and the possibility of data being altered, there has been a deliberate effort in the research community to try to resolve some breaches and challenges in the database world. An overview of some common database systems is given below.

- MS Access-Was developed by Microsoft Corporation and stores data in its own format based on the Access Jet database engine
- MySQL-Is an open-source DBMS developed by MySQL AB Company and uses multiple storage engines. It is the most popular DBMS among open source DBMSs.
- Oracle- An object oriented DBMS Developed by Oracle Corporation.
- Microsoft SQL Server – Is a relational database server developed by Microsoft Corporation.
- Other DBMSs include Ingress, Postgress, and InterBase.

MySQL was chosen for this research because of its popularity amongst users. Reference [4] estimated that there were 500,000 downloads per day and 7.5 billion active user of MySQL world-wide. In addition to this, MySQL is open source.

2. Literature Review

With the definition of database as a collection of organised information, [6] explains how this collection of organized data can be managed using a database management system (DBMS) which is an aggregate of data, hardware, software and users. Database hardware is a standard computer system with memory [7]. The statements that are used for manipulating data in a database are structure query language (SQL) which performs retrieval and update of data. Retrieval is the collecting of data from the database for data that match the specification of the user query while updating involves modification, deletion and insertion of data into the database. This work also highlights various database architectures such as; functional, application and logical architectures. Database can also assume two or multi-tier architecture [4]. The logical architecture, also known as ANSI architecture has three distinct layers of data abstraction which are physical, logical and user layer [9].

Relational database model has been discussed by [6] as the foundation of the contemporary database. It consists of tables which are classes of data structures, relational algebra that are the methods used to build a new table from the initial one and constraints that are imposed on the data contained in the tables. Tables in a relational database have three distinguished features; table name, the heading of the content (each as a column entry), and the content of the table as list of rows [8]. Referential integrity ensures that the entity being referred to in the table by users has a meaningful value. Reference [10] states that the aim of database forensics is to find out what happened when and to revert unauthorised data manipulation.

When undertaking digital forensics, [10] stresses that when carrying out database forensic, one needs to ensure that scientifically proven methods are used to gather, process, interpreter digital evidence in order to give true reconstruction of the criminal activity. This work gives methodologies for tamper detection in database using audit logs. It also explores the vulnerabilities of using audit logs to perform

database forensics in that a criminal can change their contents to hide his criminal activities.

This work points at places to look for evidence when undertaking database forensics; which are; system metadata, data files, redo logs, transaction logs and memory and trace files. Some temper detection methodologies are also described and they include: Notarization hash verification, more specialized forensic analysis algorithms such as monochromatic, RGB, Tilled-bitmap and 3D are also described. Finally they categorize artifacts in database forensics as resident or non-resident. Resident artifacts are those that reside within files and memory locations strictly reserved for SQL server while non-resident artifacts reside in files not explicitly reserved for SQL server use.

Progress made in database research can be found in [7]. This work gives the two approaches to a database by forensic experts while performing database forensic analysis; one way is to look at a database as files residing in file system therefore database forensic can just be performed like forensics of other important software applications like emails or web browsers in this aspect, database forensic viewed as a sub discipline of file system forensic and some techniques like imaging and file carving are applicable. The other approach to database forensic is to view a database as complex multidimensional system with numerous interconnected components that should be analyzed together to expose accurate truth, crucial components considered here are data model, data dictionary, application scheme and the application data.

A framework for performing database forensic analysis is given in [16]. This framework gives a guide in how to undertake forensic activities of identifying, preserving, collecting, analyzing, validating and interpreting digital evidence in a database and finally generating a report.

By the virtue that DBMS write data items to multiple location and copies such as in tables' indexes, logs materialized views and temporary relations view then when data is deleted in one location, it is not completely destroyed but its traces are left in some database locations. These data traces can be recovered by performing forensics which will extract data and information from database, logs, cache, data files, and table space e.t.c. The undo logs which show older version of data and the redo log that stores information used during crash to restore the status (recovery) are also high lightened.

What to look at when performing database forensic analysis on MySQL server is described in [3]. The work describes places where artifacts may exist in MySQL server which include server files such as database transaction logs, query logs, query cache and key cache.

InnoDB as common and popular database storage engine is explored by [11].The storage pattern of InnoDB in MySQL is described. InnoDB stores information about each table in the directory of the database as a .frm file with the table name as the file name and the size of the .frm is limited to four GB beyond which is transacted. By default InnoDB stores all data of all tables in a single file. The InnoDB storage format parts are; fill header, page heading,infinum and .supernum, user records, free space, page directory and fill trailer are also discussed. This work finally proposes a tool that reads hexadecimal data from the form then uses it to reconstruct the table and then uses the table information to locate the data in the data storage file.

A database forensic approach using log files is presented in [16]. It highlights the procedure for carrying out digital forensic using log files and applying standard forensic steps of identification, acquisition and presentation, examination and analysis, and finally documentation. However, due to complexity of databases, database centric forensic follow steps of acquisition and presentation, collection and analysis. The logs maintained by MySQL are also discussed, they include error log, general query log, binary log, update log and slow query log. This work gives a two part framework for database forensic analysis. The first part depicts the user performing an action in the database and the second part is collecting and analyzing forensic data from the central database.

The binary log has an update log that contains information needed to recreate the database since the server was last restored or the logs

were flashed. A list of every query that changed the data can also be found by passing the log-bin-option to the SQL server (Mysqld) [12]. Whenever SQL server starts a new session, it opens a new log file in addition to the old one such that if the previous log file was aden-bin.000001 then the new file will be incremented to aden-bin.000002.SQL statements in the binary log file can be viewed by using sqlbinlog command with full path of the binary log file for example, program data/mysql/data/binlog. Procedures for accessing error log, query log, redolog, undo log and index logs are given.

While the existence of log files content in database are important for forensic reconstruction, their existence can also be seen in bad side in that they can be exploited to expose privacy. Reference [12] gives a user defined deletion process for data in MySQL database in order to maintain privacy. As opposed to deletion by overwriting and setting a delete bit, this approach deletes data from all inventories created by the system. It gives a propagation strategy that performs deletion on multiple areas of a database.

2.1 Conceptual Framework

The conceptual framework depicts the instances of InnoDB and MyISAM storage engines interaction with MySQL DBMS and the scenarios of logs artifacts analysis for each storage engine instance [17]. The end results are two file systems analyses for each storage engine implementation. Each corresponding file analysis, that is; before the UPDATE and switch off and then after UPDATE and switch off were compared and conclusion made on which storage engine implementation affects more files out of the listed file targets.

This conceptual framework shows that the storage engine implemented has an influence in how an update operation writes to MySQL internals. With the implementation of either InnoDB or MyISAM storage engine, there was a presumed distinct system that is autonomous in the way it generates and writes forensic data. Each instance of storage engine had the analysis of the artefacts done in comparison to the same artefacts from the other storage engine implementation.

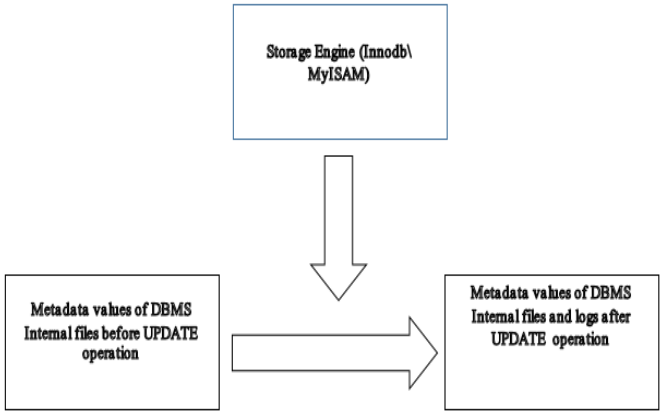

Figure 1. Conceptual framework.

Methodology

The methodology for this research was qualitative. This is because the research entailed exploring and analysing the effect of the used storage engines to MySQL internals. Various research methods elaborated in [13] explains that the application of qualitative methodology is appropriate in research undertakings where the result or observations to be made cannot be expressed in numbers; rather, they are explained and illustrated in words.

While tools were be used to read file information, the outcome is explained in words after observation. The observations made were then be compared appropriately.

3.1 Design

This research assumed experimental design [14, 16]. The subjects of the experiment were the following files; Transaction logs, Redo logs, Index logs, Query logs, Error logs Undo logs and Master Data File. The treatments were the two storage engines (MyISAM and InnoDB). After installing the storage engines to separate instance of MySQL and performing identical update operation on both, the observed changes brought about by respective storage engines were compared. The specific parameters looked

at were; MAC times, size change of the file and the MD5 has value.

3.2 Research Tools

The main forensic tool used in this research was Autopsy Sleuth Kit Version 4.0 for Windows. By using file system forensics approach, all the target files in the data base have their metadata and content changes acquired, analyzed and viewed using the same tool.
Autopsy is an open source forensic tool available for both windows and Linux platforms. In this research we used Autopsy version 4.0 because of its stability and the ability to give full view of files and folders and their metadata such as name, file type, size, modified time, accessed time, created time, and the MD5 hash of the file. These parameters will be the basis for viewing and analyzing the changes that have taken place in the file in question.
Our target files are as follows:

- Transaction log
- Redo log
- Index log
- Query log
- Error log
- Master Data File (MDF)

Applications, Tools and Equipment used in this experiment are as follows:

- MySQL 5.5.
- MySQL Workbench 6.3 CE.
- Access Data FTK imager.
- Autopsy sleuth Kit
- Two desktop computers installed with Windows 7 operating system.

3.3 Procedure

1. Two freshly wiped computers were prepared and installed with window operating system.
2. On each machine, MySQL (Version 5.5) was installed. One installation of MySQL with InnoDB storage engine and the second one with MyISAM storage engine.
3. MySQL Workbench (MySQL graphical user interface) was installed to each computer to be used for database operation.
    1. Identical databases (named 'Customer accounts') were created in each MySQL installation and in each database an identical table (name, records and settings) is created and populated with similar records.
2. MySQL instances were stopped and the machines powered off for a short time. The machines were then powered on and at this time it was assumed that all the volatile evidence data got discarded when the machines were powered off.
3. Using FTK imager activated from USB derive, an image of the logical drive where MySQL was installed was created and the image saved in a clean, wiped external evidence hard drive. This was done to both installations.
4. MySQL instances were started again.
5. With the records in each table known and identical in both MySQL installations, a specific record was updated to a common value in both systems.
6. MySQL instances were stopped and the machines were powered off to discard volatile data.
7. A second set of images was taken using FTK imager and saved as in (6) above.
8. At this point there are four images to be subjected to analysis. Two images based on MyISAM and another two based on InnoDB storage engines. These images were then ingested and analysed one by one using Autopsy analysis procedures.
9. Comparison was then made between the values of metadata before UPDATE and the values after UPDATE operation for identical files in each storage engine instance.

3.4 Measurement Metrics

The measurement metrics for this research was the count of the number of files whose metadata changed after performing the update operation. More counts of affected files meant potentially richer permanent forensic data while less count meant weaker permanent forensic evidence data

4. Results

The sample result presented in this section shows the transformation of metadata values of the file as was presented in autopsy result

window after Ingestion and analysis. It is important to remember that the metadata parameters that were looked at were identical in both storage engine instances. It's also important to note that all the four images analyzed by Autopsy were analyzed using the same procedure and each file present in a particular storage engine instance had its metadata values before UPDATE operation compared to the metadata values of the same file after UPDATE operation.

The presentation of the result starts with the name of the file followed by a brief description of the file. The logical location of the file within the forensic image then follows. The values of the file metadata before UPDATE are given followed by the metadata values after UPDATE. Observations of the transformation of the metadata values for the file are finally given.

4.1 Sample Results

Transaction log

Transaction log keeps log of all queries that have changed something in the database. It is the equivalence of binary log in mysql 5.5(MySQL 5.6 Reference manual) it is located in C:\program files/program data/mysql/mysql server5.5/data/mysql/ndb_binlog. The logical paths to these files in the result may look different from the norm because all are based on the image data is the base drive.

Transaction log before update

Name	/img_myisam1/ProgramData/MySQL/MySQL Server 5.5/data/mysql/ndb_binlog_index.frm
Type	File System
Size	8778
File Name Allocation	Allocated
Metadata Allocation	Allocated
Modified	2016-09-08 13:42:33 EAT
Accessed	2016-09-08 13:43:33 EAT
Created	2016-09-08 13:42:33 EAT
Changed	2016-09-08 13:4:33 EAT
MD5	7c38a874f706c1883bd40fb7b0e72be2

Transaction log after UPDATE

Name	/img_myisam2.E01/ProgramData/MySQL/MySQL Server 5.5/data/mysql/ndb_binlog_index.frm
Type	File System
Size	8778
File Name Allocation	Allocated
Metadata Allocation	Allocated
Modified	2016-09-08 13:42:33 EAT
Accessed	2016-09-08 16:40:18 EAT
Created	2016-09-08 13:42:33 EAT
Changed	2016-09-08 16:40:18 EAT
MD5	8d38a874f706c1883bd40fb7b0e72be2

Observation: The created time and modified time has remained the same aster UPDATE while change time access time and MD5 hash have changed after update. This shows that the operation had affected the transaction log file.

Re-Do log.

This is physically present in the disk by default as a set of files named ib_logfile0 and ib_logfile1. This data structure is used during crash recovery to correct data written by incomplete transactions. During normal operations, the redo log encodes requests to change InnoDB table data that result from sql statements. The two data structures are as bellow.

Ib_logfile0 before update

Name	/img_myisam1/ProgramData/MySQL/MySQL Server 5.5/data/ib_logfile0
Type	File System
Size	14487760
File Name Allocation	Allocated
Metadata Allocation	Allocated
Modified	2016-09-08 13:42:33 EAT
Accessed	2016-09-08 13:43:33 EAT

Created	2016-09-08 13:42:33 EAT
Changed	2016-09-08 13:4:33 EAT
MD5	5c7fbab3c474a8a07c7c7031bc58e1cb

Ib_logfile0 after update

Name	/img_myisam2/ProgramData/MySQL/MySQL Server 5.5/data/ib_logfile0
Type	File System
Size	14487760
File Name Allocation	Allocated
Metadata Allocation	Allocated
Modified	2016-09-08 13:42:33 EAT
Accessed	2016-09-08 16:40:18 EAT
Created	2016-09-08 13:42:33 EAT
Changed	2016-09-08 16:40:19 EAT
MD5	e2adb6b0586e713ae74292c336d5aeee

Observation: There is evidence of the file being affected by the UPDATE operation based on the transformation of the metadata. The MD5, the changed time accessed times are different in the two images.

Ib_logfile1 before UPDATE

Name	/img_myisam1.E01/ProgramData/MySQL/MySQL Server 5.5/data/ib_logfile1
Type	File System
Size	10485760
File Name Allocation	Allocated
Metadata Allocation	Allocated
Modified	2016-09-08 13:42:33 EAT
Accessed	2016-09-08 13:43:33 EAT
Created	2016-09-08 13:42:33 EAT
Changed	2016-09-08 13:43:33 EAT
MD5	c1c9645dbc14efddc7d8a322d85f26e9

Ib_logfile1 after update

Name	/img_myisam2.E01/ProgramData/MySQL/MySQL Server 5.5/data/ib_logfile1
Type	File System
Size	10485760
File Name Allocation	Allocated
Metadata Allocation	Allocated
Modified	2016-09-08 13:42:33 EAT
Accessed	2016-09-08 13:43:33 EAT
Created	2016-09-08 13:42:33 EAT
Changed	2016-09-08 13:43:33 EAT
MD5	c1c9645dbc14efddc7d8a322d85f26e9
Internal ID	182038

Observation: all the values of metadata have remained the same therefore showing that this file was not affected by the UPDATE operation.

4.2 Summary of Findings

By doing the comparison shown above for all the files in the two storage engines, the observations made are summarized in the table below. The table shows the summary of how UPDATE operation affected the target files in InnoDB and MyISAM storage engine implementations.

	TARGET FILES						
STORAGE ENGINES	Transaction log	Re-Do log(ib_0)	Re-Do log(ib_1)	Query log	Error log	Master Data File	.frm
InnoDB	✓	✓	✓	✓	✓	✶	X
MyISAM	✓	✓	X	✓	✓	✶	X

Table 1: Summary of the result

Legend:

✓ -File affected

X - File not affected

✶-File non-existent because of implementation reasons.

From this table, it can be seen that transaction log, redo log(ib_0), redo log(ib_1), query log and Error log are affected in InnoDB implementation while transaction log, redo log(ib_0), query log and Error log are affected in MyISAM implementation. Re-do (ib_01) is however not affected in MyISAM implementation. The .frmfiles of the tables are not affected in both implementations. Master Data File (MDF) is however not analyzed because the implementations were stand alone and could not implement the concept of MDF.

4.3 Proposed Methodology

Based on the steps followed in this research, a methodology is outlined as a summary of the steps that were followed in undertaking the experiment. This methodology can be used to test the footprints of any storage engine on the internal files of a DBMS. This will help in flagging and listing files that have been affected by a particular database operation. These file can then be analysed to interpret the actual content to see the nature of change to determine the worth of the evidence.

It is important to remember that the metadata and file information used in this work were as follows:

- **Modified** -when the content of the file most recently changed
- **Accessed** -when the file was most recently opened for reading
- **Created** -the time of creation of the file
- **Changed** -When the file was first created or had the meta data changed
- **MD5** -the MD5 digest of the file
- **Size** -size of the file

In addition to Modified, Access, Created\Change time information of the files, the forensic tool used, just as in the case of this research, it should give the hash values and size of the file both before and after the operation. In our case, Autopsy gave MD5 hash foe the two file instances. The strength of MD5 hash is that it can detect a change in a file as small as a single bit. The size of the file was also observed to have changed for some files after the operation.

The proposed methodology for testing the forensic richness of a storage engine is illustrated in the following diagram.

5. Conclusion

The following are the conclusions from the study.

- It has been shown that more files are affected in InnoDB than MyISAM implementation hence InnoDB has a higher number of potential forensic evidence locations than MyISAM.
- It is possible to list files affected by an operation in a database system by using a forensic tool to perform a file system analysis
- The methodology shown in this research can be replicated in other file systems to show which files or set of files are affected by an operation
- Because the content of a files are not always the same in different systems, this methodology will help forensic examiners to sieve the files that have potential evidence and then subject them to further analysis therefore eliminating the problem of analysing all files, some of which may not contain any evidence.

5.1 Further work

While the study has given a methodology of how to list potential forensic evidence locations in a file system, hence showing forensic richness, the study does not however scrutinize the individual evidence location to show its content as either operation instruction or data values. While this work gives the methodology of how to list evidence location, more work is required is necessary to verify what type of evidence is present in each listed location.

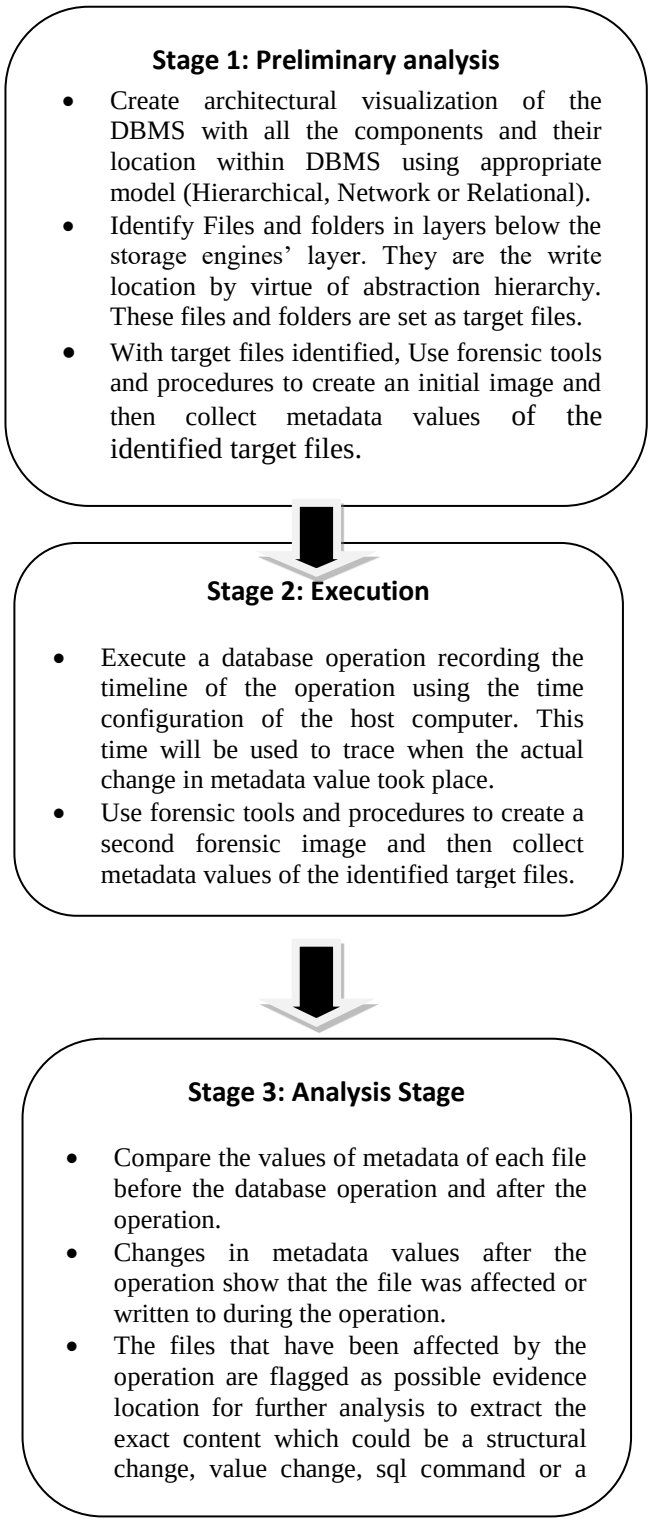

Figure 2. Proposed methodology.

6. Acknowledgment

We acknowledge the support of University of Nairobi School of Computing and informatics C4D Lab for setting up resource for performing the experiment.

References

[1] Robbins, R.J., "Database Fundamentals", Johns Hopkins. University, (1994).

[2] Weippl, E., "Database Forensics." In Proceedings of the 24th IEEE International Conference on Advanced Information Networking and Applications (AINA), (2010).

[3] Alexander, K.B., "Database Forensic *Databas Forensic Analysis Using Log Files"* (2014).

[4] Holck, J., Mahnke, V. and Zicari, R. 'Winning Through Incremental Innovation: The case of MySQ. AB. IRIS." (2008).

[5] Bannon, R., Chin, A., Kassam, F., Roszko, A. and Holt, R., "Mysql conceptual architecture".

[6] Khanuja, H.K. and Adane, D.S., "A Framework for database forensic analysis". Computer Science & Engineering, 2.3 (2012):27.

[7] Hammer, M. and Mc Leod, D., "Database description with SDM: a semantic database model. ACM Transactions on Database Systems (TODS)", 6.3 (1981):351-386.

[8] Hauger, W.K. and Olivier, M.S., "The state of Database Forensic research". In Information Security for South Africa (ISSA),(2015):1-.

[9] Gilfillan, I., Mastering MySQL 4.Sybex (2003).

[10] Flores, D.A., Angelopoulou, O. and Self, R.J., "Combining Digital Forensi Practices and Database Analysis as an Anti-Money Laundering Strategy for Financial Institutions" Third International Conference on Emerging Intelligent Data and Web Technologies. (September,2012):218-224). IEEE.

[11] Fruhwirt, P., Huber, M., Mulazzani, M. and Weippl, E.R.. “Innodb database forensics”. In Advanced Information Networking and Applications (AINA), (2010):1028-1036. IEEE.

[12] Grebhahn, A., Schäler, M. and Köppen, V., “Secure Deletion: Towards Tailor-Made Privacy in Database Systems.” In BTW Workshops, (2013): 99-113.

[13] Hancock, B., Ockleford, E. and Windridge, K., “An introduction to qualitative research”. Nottingham: Trent focus group (1998).

[14] Johnson, B., Toward a new classification of nonexperimental quantitative research. Educational Researcher, 30.2(2001):3-13.

[15] Bricki, N. and Green, J., A guide to using qualitative research methodology, (2007).

[16] Khanuja, H.K. and Adane, D.D., “Database security threats and challenges in database forensic: A survey”. In the Proceedings of 2011 International Conference on Advancements in Information Technology (AIT 2011), available at: http://www.ipcsit.com/vol20/33-ICAIT2011-A4072.pdf.

[17] Naiqi, L., Zhongshan, W. and Yujie, H., “Computer ForenSics Research and Implementation Based on NTFS File System”. In 2008 ISECS International Colloquium on Computing, Communication, Control, and Management 1(2008): 519-523.