

Scenario-Based Digital Forensic Investigation of Compromised MySQL Database

Taurai HUNGWE¹, Hein. S. VENTER², Victor R. KEBANDE³

¹Department of Computer Science, Sefako Makgatho Health Sciences University,
Molotlegi Street, Ga Rankuwa, 0208, South Africa

²Dept. of Computer Science, University of Pretoria, Lynwood Road, Hatfield, 0002, South Africa

³Department of Computer Science and Media Technology, Malmö University,
Nordenskiöldsgatan 1, 211 19 Malmö, Sweden

¹Tel: +27 64902 7964, Email: taurai.hungwe@smu.ac.za,

²Email: hventer@cs.up.ac.za, ³Email: victor.kebande@mau.se

Abstract: Insider and outsider database threats have more often than not posed a greater challenge as far as integrity and investigation of databases is concerned. Database forensic investigation is a process through which scientifically proven methods can be used to create a hypothesis that can prove or disprove the occurrence of a potential security incident. This paper explores the techniques that can be used to conduct forensic investigations of a compromised MySQL database. The authors have simulated investigative scenarios that have aided to conduct forensic investigative processes and the results are promising.

Keywords: Database forensics, Database management system, Forensic configuration, Database, MySQL, MySQL forensic investigation process.

1. Introduction

The existence of the Internet and the proliferation of digital devices mainly as a result of the integration of applications that exist of mobile phones, Internet of Things (IoT) and other business applications [1] have meant that there is need to store data in database systems. Consequently, database systems have become a core component that is mainly needed by a majority of computing systems for purposes of storing critical and sensitive data [2]. Tampering with the content or configuration of a Database Management System (DBMS) may bring integrity concerns in most cases. More so, it is possible for a disgruntled database administrator to easily tamper with content of the database and hide the traces of evidence that may be required to incriminate him.

To proactively solve a number of these concerns and to solve this problem, it is necessary to enforce strong security mechanisms in databases. It is also important to note that maintaining database security in any system is quite an important aspect given that the existence of loopholes can easily allow illegal access to a database [3]. Given that a database investigator will need a lot of time to excavate or prove the causality, it would be imperative for one to timely collect and extract data that can be deemed evidential for investigative purposes. This mainly involves forensically extracting the logs that can help to conduct the audit capability in DBMS in order to detect suspicious events [4] [5] [6].

This paper has been presented in three folds. Firstly, the authors present three simulation scenarios, which are followed by the procedure used to arrive at each of the experiment that was conducted. Next, a critical evaluation of the propositions is given.

The rest of the paper is organised as follows: Section II presents the objectives and Section III, the background of this study. Next, Section IV gives related work. Section V discusses the methodology and Section VI gives the proposed Scenario-Based Digital Forensic Investigation of Compromised MySQL Database. After this, Section VII gives a critical evaluation of the proposed concepts. Section VIII draws the conclusion of the study and gives avenues for future work.

2. Objectives

The overall objective of this study is how forensic investigation process can be conducted in a MySQL database. A scenario-based approach has been used to plan a digital forensic investigation on a compromised MySQL database. The approach assists in the reconstruction of the affected layers and consequently restoration of the database.

The specific objectives of the study are to:

- Present four abstract layers based on the ANSI/SPARC model on a MySQL DBMS.
- Modify each layer to simulate compromised database so that we can see and record the cause and effects of unauthorised modifications.
- Reconstruct a compromised MySQL DBMS to see if we can forensically find evidence.
- Construct forensic process by introducing configurations in a MySQL database.

3. Background

This section provides background information on the following areas: Digital Forensics, 4-Level Database System Architecture and MySQL database.

3.1 Digital Forensics

Digital Forensics (DF) is “the use of scientifically derived and proven methods toward the preservation, collection, validation, identification, analysis, interpretation, documentation and presentation of digital evidence derived from digital sources for the purpose of facilitating or furthering the reconstruction of events found to be criminal, or helping to anticipate unauthorized actions shown to be disruptive to planned operations” [7].

DF processes are also comparable with other forensic branches such as file system forensics. There are similarities with file system forensics, which has the following, four phases: acquisition, identification, evaluation and admission [8], [9]. On the other hand, Digital Forensics Investigations (DFI) aim to correctly identify, collect, validate, analyse, interpret, preserve, and present digital evidence.

A number of tools are currently available which can be used to carry out a DFI. The tools include the ones mainly used for detection in order to identify risks, e.g., Ontrack and DriveSpy for data recovery [9]. Digital forensic tools can be classified according to usage [10], for example for slack space and data, recovery tools such as Ontrack and DriveSpy may be used. These tools can be used in various environments and for particular reasons such as to meet the legal requirements for the process to stand the test in litigations.

3.2 4-Level Database System Architecture

The 4-level Database System Architecture (4-LDSA) defines the database system in terms of the database components and interactions among database components [11]. The earlier model of database structures was mainly one-dimensional which did not provide for data independence. This evolved to two-dimensional where data storage was separated from data presentation. The third dimension was necessitated by the application program independence. The structural principles are defined and specified by the ANSI/SPARC

architecture published in 1975 providing for the three-dimensional model [12]. However, the architecture of concern to this research is the four-layer model.

The four-layer model, also known as the ANSI/SPARC intentional-extensional model, divides a database into layers of data and metadata. The four abstract layers are the data model, data dictionary, application schema and application data layers in their hierarchical order. This model is still a suitable basis for database forensic examination of modern databases as it looks at data and metadata.

Database Forensics (DBF) should also consider metadata apart from the data itself. Metadata determines how the data should be interpreted. The four abstract layers can be used for a top down or bottom up analysis in the forensic investigation. It can also be used to come up with various configurations that can be exploited in providing answers to the DBF investigation. Figure 1 represents the hierarchical top-down view of the DBMS layers.

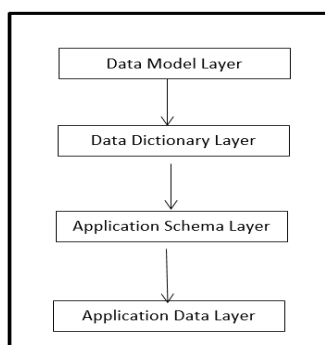


Figure 1: The hierarchical structure of four DBMS abstract layers

The three top layers apart from the application data layer can be viewed as a set of lenses through which one can view data in a database.

3.3 MySQL Database

MySQL database is one of the popular databases in use [13], [14]. Research on one of the other major databases, PostgreSQL has been carried out which gives the impetus of a follow-up research to fill the gap on MySQL DBF [15]. The small size of a MySQL database makes it easier to run on smaller IT resources. This is why it is used for embedded systems. The other aspect is MySQL database, as an open source database, makes a good candidate to work on.

MySQL database is an implementation of relational database model, which means that the database structure is organised as tables or relations and based on mathematical foundations of logic, relational algebra, calculus and operators. The logical model of MySQL database comprises of objects such as tables, views, rows and columns, and offer flexible programming environment. A MySQL database implements rules that govern relationships between data fields, for example, 1-to-1 (one-to-one) relationships. MySQL enforces rules such that applications do not see inconsistent, duplicate, orphan, out-of-date or missing data. To access MySQL database, a structured query language (SQL) is used.

A major key element of MySQL database is the INFORMATION_SCHEMA, which provides database metadata such as databases or tables names. The INFORMATION_SCHEMA is an information database within a MySQL installation and stores all the information for all the database instances maintained in a MySQL server. INFORMATION_SCHEMA comprises of read-only tables or views and does not provide base tables. Since the associated files are views, they are read-only tables [16].

4. Related Work

The section concentrates in giving work that has somewhat been used as related work in this research study. Research by [17] has explored the impact of triggers on forensic analysis and acquisition of databases for forensic purposes. In this study, the authors have been able to ascertain that triggers could easily be used for purposes of identity and attribution. This could also be used before and after in commissioning actions. Another research by [18] has illustrated the FDCC framework that is able to collect forensic information using NoSQL database. This data is easily imported into a graph database where data analysis is easily facilitated. Apart from that, research by [2] has also stated that Database Forensics (DBF) specifically attempts to conduct investigations on threats that targets DBMS across organisations. Notable process that have been identified in this phase include identification, collection, preservation, analysis and reconstruction of evidences. Lastly, another research by [3] has shown that in order to enforce security of databases, Database Forensic Analysis (DFA) provides a technique that can be used to prevent the illegal access of security systems. The authors of this paper would like to acknowledge research that has been accomplished by the other authors because it has given more insights on DBF.

5. Methodology

Experimentation was used on a MySQL DBMS, to reveal the layers in a MySQL DBMS. The process of identifying the layers was done through inspection of the installed database and guided by documentation from MySQL website. For example, to find the data dictionary layer, the documentation stipulates that the data dictionary is contained in the *ibdata1* file. Listing the files using *ls* command was used to locate the *ibdata1* file. Experiments to illustrate various unauthorised modifications on the different layers were executed. Unauthorised modifications could be carried out directly or indirectly. Direct unauthorised modifications are those modifications carried out when one accesses files at byte level using the hex editor or any other editor or tool. On the other hand, indirect unauthorised modifications are those carried from the OS by use of OS commands such as deleting the file without getting to the byte level structure or altering the path or location of the files. In digital forensics, unauthorised modifications produce digital records that can be used to explain what would have transpired in order to resolve forensic questions. Digital records can be located in the OS, which retains information such as creation date, modification or last update times or timestamps information. The database was reconstructed using uncompromised layers and this led to the construction of forensic configurations in a MySQL database.

6. Scenario-Based Digital Forensic Investigation for Databases based on 4-LDSA

This section presents the approaches that have been used in conducting scenario-based investigation for MySQL database. Mainly, the study focuses on how DBF process can be conducted on a MySQL DBMS installation. DBF perceives the database to be multidimensional and maps the way in which forensics can be carried out on the database [8]. The overall goal was to simulate a compromised DBMS so that forensic processes and methods could be performed.

Exploratory experiments were carried out on a MySQL DBMS. Experiments to illustrate various unauthorised modifications on the different layers were executed. Unauthorised modifications could be carried out directly or indirectly. Direct unauthorised modifications are those modifications carried out when one accesses files at byte level using the hex editor or any other editor or tool. On the other hand, indirect unauthorised modifications are

those carried from the OS by use of OS commands such as deleting the file without getting to the byte level structure or altering the path or location of the files.

6.1 Simulated Attack Scenarios

Experimentation was used on a MySQL DBMS, to reveal the layers in a MySQL DBMS. The process of identifying the layers was done through inspection of the installed database and guided by documentation from MySQL website. For example, to find the data dictionary layer, the documentation stipulates that the data dictionary is contained in the *ibdata1* file. Listing the files using *ls* command was used to locate the *ibdata1* file. Two databases were created, namely, the *Student* and *HR* databases. A copy of the DBMS was kept to provide a backup, which had clean layers, meaning that the DBMS would have the correct or expected data values stored in the database.

6.1.1 Simulated Experiment on Data Model Layer

The data model layer for a MySQL DBMS is located in the */var/lib/mysql* directory structure of the DBMS. All database instances, i.e. *Student*, *Student1* and *HR* were created under this directory, for example, the following directories would have been created:

```
/var/lib/mysql/Student
/var/lib/mysql/Student1
/var/lib/mysql/HR
```

The purpose of this experiment was to simulate unauthorised changes of the DBMS at the data model layer. The procedure for the experiment is shown in Table 1.

Table 1. Procedure for Experiment on Data Model layer

procedure	
1	<i>Mysql</i> directory was accessed by executing the following command: <i>cd /var/lib/mysql/</i>
2	<i>Mysql</i> directory was deleted by executing the following command: <i>del mysql;</i>
3	<i>Student</i> database was selected using the mysql statement: <i>use Student.</i>
4	<i>HR</i> database was selected using the MySQL statement: <i>use HR.</i>
5	<i>Student1</i> database was selected using the mysql statement: <i>use Student1</i>

Mysql directory was deleted. *Student1*, *Student* and *HR* databases were inaccessible. Deletion of *mysql* directory rendered all databases inaccessible. All the databases are affected as they reside in the same directory. The *use* statement would search for non-existence databases as the deletion obliterated the data model layer and as such, the databases were compromised.

6.1.2 Simulated Experiment on Data Dictionary Layer

The location of the data dictionary, */var/lib/mysql/ibdata1* was established. The data dictionary for *Student*, *Student1* (copy of *Student* database) and *HR* databases are contained in the *ibdata1* file. Malicious modification to the *ibdata1* file follows. This experiment was conducted on the data dictionary layer to simulate unauthorised modifications to a MySQL DBMS. The hexedit tool was used to display the contents of the *ibdata1* file, which contains the data dictionary.

Student database name is represented by the following byte word: 53 74 75 64 65 6E 74 31 annotated by “1” in Figure 2. Malicious modification to change the name of *Student*

database to *ctudent* as highlighted in the Figure 2 below was conducted, *ctudent* byte word representation is 63 74 75 64 65 6E 74 annotated by “2”.

Table 2. Procedure for Experiment on data Dictionary Layer

procedure	
1	<i>Ibdata1</i> file was accessed by executing the following command: <code>cd /var/lib/mysql/</code>
2	The following command was executed: <code>hexedit -m Ibdata1</code> to open the <i>Ibdata1</i> file.
3	Student database name was maliciously modified to <i>ctudent</i> .
4	Student database was selected using the MySQL statement: <code>use Student</code> .
5	Select statement was executed to display the contents of students table in the Student database: <code>Select * from students;</code>
6	HR database was selected using the MySQL statement: <code>use HR</code> .
7	Student database was selected using the MySQL statement: <code>use Student</code> .

The other databases, *Student1* and *HR* were accessible. *Student* database name was maliciously modified to *ctudent*. The unauthorised modifications to the data dictionary abstract layer provided a compromised data dictionary and consequently a compromised MySQL DBMS for the *Student* database, which made it inaccessible. The unauthorised modifications did not affect the other databases: *Student1* and *HR*.

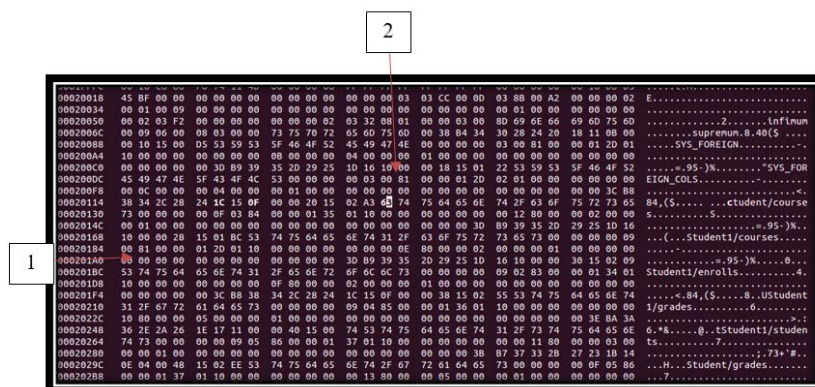


Figure 2: Modifications at data dictionary level

6.1.3 Simulated Experiment on Application Schema Layer

In a MySQL database, the table structures or schemas are defined in the application schema. The application schema is resident in the database files with *.frm* (dot frm) extensions. The *.frm* file contained the table structure or schema file. A corresponding *.frm* file was created for each table. As an example, *students* table will have a *students.frm* file, as shown in Figure 3. These files are located in *Student/* directory for the *Student* database and the location path being */var/lib/mysql/Student/*. Similarly, for the replicated database, *Student1* corresponding *.frm* files are in */var/lib/mysql/Student1/*.

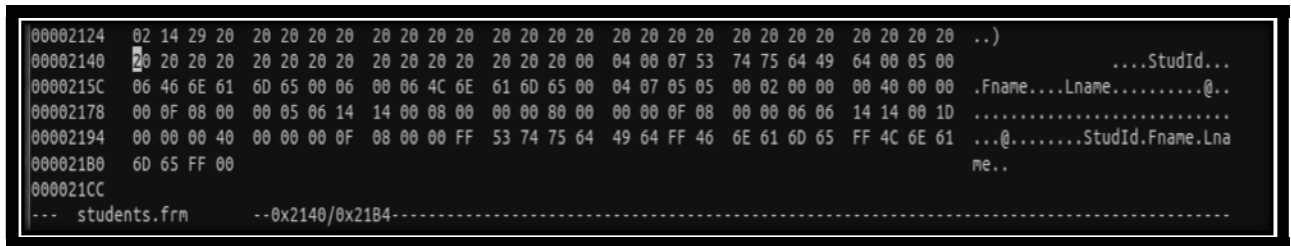


Figure 3: students.frm file structure

This experiment was conducted on the application schema layer to simulate unauthorised modifications to a MySQL DBMS.

Table 3. Procedure for Experiment on Application Schema Layer

procedure	
1	Student directory was accessed by executing the following command: <code>cd /var/lib/mysql/Student.</code>
2	The following command was executed: <code>hexedit -m students.frm</code> to open the students.frm file.
3	The column heading or field name <i>fname</i> was changed to that of <i>lname</i> by altering the following byte values: 46 6E 61 CD 65 to: 4C 6E 61 CD 65.
4	Student database was selected using the MySQL statement: <code>use Student.</code>
5	Describe statement was executed to display the structure and field names of students table.

Unauthorised editing was carried out on the *students.frm* file, by changing the corresponding hexadecimal value. The column heading or field name *fname* was changed to that of *lname*. After the malicious modification, the edited field, *fname*, reverted to the old column heading as if nothing was maliciously altered. Two inferences were made to why the field names reverted to their original field names. One could be that, in a MySQL database, the information required would be read from the *ibdata1* file, Figure 4. The *ibdata1* file had a copy of the information, which is similar to that contained in *.frm* files.



Figure 4: Ibdata1 File

Apart from containing the data dictionary, the *ibdata1* file, as observed in Figure 3, contained descriptions of the database structures. Thus, there is the database name, as annotated by “1” – *Student* for the *Student* database. “2” – *students*, the table in the *Student* database, annotate the application schema or table name. The other information on fields or columns is also contained in the *ibdata1* file, annotated by a “3” for *Fname* and the annotation “4” showed the *StuId*, which is the field name for student identification number in *students* table. The actual data values for the various columns of a given table in a MySQL database are also duplicated in the *ibdata1* file.

6.1.4 Simulated Experiment on Application data Layer

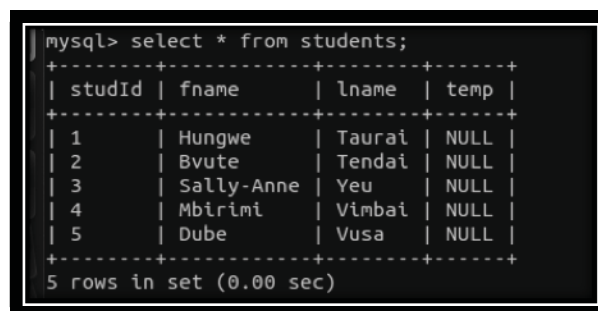
The application data layer presents data to an end user in an understandable way. At this layer, for example, the end user would be able to view the actual datasets. By use of a select statement, the actual datasets such as the surnames of students from the *students* table would be displayed. As such in the viewer's perspective is the important layer. Most of the database queries can be done at this layer to give different views of information.

Unauthorized modifications or alterations to the application data layer can be achieved by using data manipulation language (DML). Statements such as *insert*, *delete* and the like which are used to alter data values in the database belong to DML. Data definition language (DDL) statements can also be used to make malicious modifications. The purpose of this experiment was conducted on the application data layer to interchange or swap the column names in the *students* table.

Table 4. Procedure for Simulated Experiment on Application Data layer

procedure	
1	<i>Student database was selected using the MySQL statement: use Student.</i>
2	<i>A temporary field was created in the students table by executing the following script:</i> <i>ALTER table students ADD</i> <i>Temp varchar (20)</i> <i>AFTER lname;</i>
3	<i>The lname column datavalues were moved to temp column by executing the following script:</i> <i>INSERT INTO students temp</i> <i>SELECT lname</i> <i>FROM students;</i>
4	<i>The fname column datavalues were moved to lname column by executing the following script:</i> <i>INSERT INTO students lname</i> <i>SELECT fname</i> <i>FROM students;</i>
5	<i>The temp column datavalues were moved to fname column by executing the following script:</i> <i>INSERT INTO students fname</i> <i>SELECT temp</i>
6	<i>The temp field was deleted by executing the following script:</i> <i>ALTER table students DELETE</i> <i>temp varchar (20);</i>
7	<i>Select statement was executed to display the contents of students table in the Student database: Select * from students;</i>

Using the *Student* database, the *students* table field names were swapped by first creating a temporary field in the *students* table. Figure 5 represents the *students* table with the created temporary field.



```
mysql> select * from students;
+-----+-----+-----+-----+
| studId | fname   | lname  | temp |
+-----+-----+-----+-----+
| 1      | Hungwe | Taurai | NULL |
| 2      | Bvute  | Tendai | NULL |
| 3      | Sally-Anne | Yeu   | NULL |
| 4      | Mbirimi | Vimbai | NULL |
| 5      | Dube   | Vusa   | NULL |
+-----+-----+-----+-----+
5 rows in set (0.00 sec)
```

Figure 5: Students Table

The *fname* and *lname* field datavalues were swapped through the *temp* field. Once the *lname* field datavalues were moved to *temp* column, the *fname* field datavalues

were subsequently moved to the field, which was originally occupied by *lname*. The *temp* field was deleted. Selecting datavalues from *lname* gave *fname* datavalues. Thus, giving the wrong datavalues as the datavalues had been swapped.

7. Evaluation of Proposed Concepts

This section gives an evaluation of the proposed concepts based on the 4LDSA point of view. It is important to note that the authors have concentrated on giving the discussion based on the simulated experiments. When a database has been compromised, it remains to the forensic investigator to plan on what should be done, where to start, and what are the most probable scenarios that could have led to such database compromise. The answers lead to the combinations or configurations related to solving the problem of a compromised database. This goes a long way in assisting the forensic investigator.

This section links the resultant scenarios, which have been presented by the experiments on, unauthorised attacks to the four DBMS layers namely data model, data dictionary, application schema and application data layers. The experiments simulated how a MySQL database was compromised through the unauthorised modifications to the layers of the DBMS. If a single layer is compromised such as the data model layer, the data model layer would be represented by a binary value “1”. Consequently, if the layer was not compromised, it would be represented by a binary value “0”. Therefore, a layer can exist in one of the possible states of either compromised (“1”) or the state of not being compromised (“0”). The summary of different states is presented in Table 5. These states are the different configurations or combinations.

Considering the top down hierarchical layer level of the DBMS, the uppermost being data layer, followed by data dictionary, then application schema and the lowest being the application data layer, the binary values will either represent a corresponding compromised or a non-compromised layer. For example, a combination of the following binary values “0110” represents from the leftmost binary value, data model layer not compromised, followed by data dictionary compromised, application schema also compromised and the rightmost binary value representing a non-compromised application data layer.

Table 5: Different Configurations

Data Model	Data Dictionary	Application Schema	Application data
0	0	0	0
0	0	0	1
0	0	1	0
0	0	1	1
0	1	0	0
0	1	0	1
0	1	1	0
0	1	1	1
1	0	0	0
1	0	0	1
1	0	1	0
1	0	1	1
1	1	0	0
1	1	0	1
1	1	1	0

1	1	1	0
---	---	---	---

The configurations would then be used to conduct a systematic MySQL database forensic investigation. These configurations have been used to come up with a MySQL database forensic process.

Let us consider a configuration of “0001” as an example, which represents a compromised application data layer in a student database. Some of the columns in the student database would be student name, surname and semester marks. Let us consider where semester marks could have been maliciously modified. A select statement would reveal that all student qualify to write the final examination. This would be a false representation of the correct status. Some students would not have met the required semester pass mark to sit for the final examination. Within an institution, there would be other databases such as employees’ database. Considering the results of experiments in Section 4, if all the databases are available and are still accessible, the data model layer would not have been maliciously modified. Similar eliminations would be done up to the point of leaving the last layer of a compromised application data layer. The configuration “0001” will direct the investigator to look up of the unauthorised changes in the application data layer. The investigator has to check if the data returned is consistent with the values residing in the database or if there are changes to the data or not. The investigation is guided by the confirmations on what the actual or correct values reside in the database.

8. Conclusion and Future Work

The paper has provided a scenario-based approach that can be used to conduct digital forensic investigation for a compromised MySQL database. The authors have presented three simulated scenarios with their respective results. The paper identified important investigative scenarios that could be used if the integrity of a database is in question. The experiments culminated in MySQL forensic investigation process (MqFIP) and construction of configurations in MySQL database. MqFIP as an add-on subprocess augments the identified investigative processes class, in the Harmonised Digital Forensic Investigation Process ISO/IEC 27403, [19] [20]. For future work, the authors aim to identify more scenarios that are based on live forensic processes.

References

- [1] Al-Dhaqm, A., Razak, S., Othman, S. H., Choo, K. K. R., Glisson, W. B., Ali, A., & Abrar, M. (2017). CDBFIP: Common Database Forensic Investigation Processes for Internet of Things. IEEE Access, 5, 24401-24416.
- [2] Al-Dhaqm, A. M. R., Othman, S. H., Razak, S. A., & Ngadi, A. (2014, August). Towards adapting metamodeling technique for database forensics investigation domain. In Biometrics and Security Technologies (ISBAST), 2014 International Symposium on (pp. 322-327). IEEE.
- [3] Bria, R, Retnowardhani, A, Utama, D.N. (2018). Five Stages of Database Forensic Analysis: A Systematic Literature Review. In 2018 International Conference on Information Management and Technology (ICIMTech). (pp. 246 – 250).IEEE.
- [4] Wright, C. “SANS Blog: Forensics and Data Access Auditing,” 15 March 2009a. [Online]. Available: <http://computerforensics.sans.org/blog/2009/03/15/forensics-and-data-accessauditing>. [Accessed 21 April 2012].
- [5] Stahlberg, P, Miklau, G and Levine, B.N. “Threats to Privacy in the Forensic Analysis of Database Systems,” in ACM International Conference on Management of Data, 11-14 June. Beijing, China, 2007.
- [6] Flores, D. A., Angelopoulou, O., & Self, R. J. (2012, September). Combining digital forensic practices and database analysis as an anti-money laundering strategy for financial institutions. In Emerging Intelligent Data and Web Technologies (EIDWT), 2012 Third International Conference on (pp. 218-224). IEEE.
- [7] Palmer, G. (2001). A Road Map for Digital Forensic Research, Report from DFRWS 2001, First Digital Forensic Research Workshop, Utica, New York, August 7 – 8, pp. 27–30.

- [8] Olivier, M. S. (2009). On metadata context in database forensics. *Digital Investigation*. Vol. 5 (3-4), pp. 115 – 123
- [9] Deterdeing, B. (2002). Tools. In: Marcella, A.J. and Greenfield, R.S. (Eds), *Cyber Forensics. A Field Manual for Collecting, Examining, and Preserving Evidence of Computer Crimes*. London: Auerbach Publication
- [10] Wiles, J and Reyes, A. (2007). *The Best Damn Cybercrime and Digital Forensics Book Period*. Massachusetts: Syngress
- [11] Date, C.J. (2004). *An Introduction to Database Systems*. (8ed). New York: Pearson Education INC
- [12] Hay, D. (2011). *UML and Data Modelling: A Reconciliation*. New Jersey: Technics Publication
- [13] Bridgwater, A. (2012). *Open Source Insider*. [Online] Available at <http://www.computerweekly.com> > [Accessed 11 January 2018]
- [14] Hess, K. (2010). Top 10 Enterprise Database Systems to Consider. ITBusinessEdge. [Online] Available at < <http://www.serverwatch.com/article.php/3883441/Top-10-Enterprise-Database-Systems-to-Consider.htm> > [Accessed 11 January 2018]
- [15] Beyers, H., Olivier, M. and Hancke, G. (2011). Assembling Metadata for Database Forensics. In: Peterson, G. and Sheno, S. (Eds), *Advances in Digital Forensics VII*. 7th IFIP WG 11.9 International Conference on Digital Forensics, FL, USA, January 31 - February 2, 2011. Revised Selected Papers, pp. 89 - 99. Orlando: Springer
- [16] Oracle Corporation. (2014). *MySQL 5.0 Reference Manual*. [Online] Available at < <http://dev.mysql/doc/refman/5.0/en/what-is-mysql.html> > [Accessed 11 January 2018]
- [17] Hauger, W. K., & Olivier, M. S. (2015). The impact of triggers on forensic acquisition and analysis of databases.
- [18] Thorpe, S., & Bernard, M. (2017, March). Graph mining for forensic databases. In *SoutheastCon, 2017* (pp. 1-10). IEEE.
- [19] ISO/IEC 27043. (2015). *Information technology – Security techniques – Incidents investigation principles and process*. Geneva: ISO.
- [20] Mumba, R. and Venter, H.S. (2014). Testing and Evaluating the Harmonised Digital Forensic Investigation Process in Post Mortem Digital Investigations. *ADFSL Conference on Digital Forensics, Security and Law*.