UNIVERSITY OF ONTARIO INSTITUTE OF TECHNOLOGY

MASTER THESIS

A Relational Temporal Database Using Blockchain With Optimal Snapshots

---

Author:

Mohammadamin

BEIRAMI

Supervisors:

Dr. Ken PU

Dr. Ying ZHU

A thesis submitted in fulfillment of the requirements

for the degree of Master of Science in Computer Science

August, 2018

Copyright © Mohammadamin Beirami, 2018

Declaration of Authorship

I, Mohammadamin BEIRAMI, declare that this thesis titled, "A Relational Temporal Database Using Blockchain With Optimal Snapshots" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

_____

Date:

_____

Abstract

This thesis deals with the development of a Blockchain-based framework which is built on top of a Relational Database Management System (RDBMS) to discover and report authenticity of the stored records as well as malicious data manipulation on the database system. To do this, we proposed a mechanism to document and analyze the historical records of a database. Historical records are gathered by auditing the executive activities on the tables of the database system and storing these activities in the tamper-evident temporal log tables. The temporal log tables utilize Blockchain technology for two purposes: first, to analyze and testify the authenticity of the records stored in the normal tables and second, to make the log tables tamper-evident. This thesis argues that the Blockchain technology makes it extremely difficult for the privileged users or outsiders to conceal their activities on a database. Although logging the transactions is very useful for forensic purposes, the fact that they require linear time to query the historical data of records, is seen as a challenging problem. To address this issue, optimally creation of multiple snapshots for the materialization is suggested to lower the time complexity of querying for the historical records on the log tables. The proposed system supports all RDBMS systems and to achieve its objectives, the system solely rely on verifiable evidences and not on putting trust on the users of the system.

Acknowledgements

[24] [31] [37] [21] [6] [35]

Contents

List of Figures

List of Tables

Chapter 1

Motivation and Problem Definition

1.1 Motivation

In data science and engineering, historical data is widely analysed for a lot of purposes such as making data-driven decisions [30]. Historical data could be financial reports, project data, emails, audit logs or any similar documents which contain past occurrences in an organization. Although keeping some historical data could seem burdensome and impractical due to the size they may get into [3], but current big data technologies such as cloud storages has made it possible to store these information in an easier and more affordable way than before [36]. A few possible examples of using the historical data could be to evaluate the performance of an enterprise and make relative decisions to improve the performance [11], to analyze user experiences with a product in order to explore the ways to improve the service [20] and to be used for forensic purposes [39]. Regardless of the use of historical data, what matters the most is the trustworthiness of such data [18] as they may be used for very sensitive purposes.

With the ever-increasing complexity in cyber security attacks, guaranteeing the trustworthiness of data is a very difficult task to do. Many organizations store their highly confidential data in relational databases but Relational Database

Management Systems' (RDBMSes) preventive or detection mechanisms is proven to be not always capable of securing data from being compromised [38]. For a better understanding of the complexity of identifying attacks, consider a scenario in which a database super-user who acts as a root can perform a lot of administrative tasks on a relational database. For the sake of more profit, the super-user decides to alter and increase the number of hours he worked in the past month in the database. Since the super-users can bypass all the security requirements, he could alter the records without any red-flags showing this action as a malicious attempt on the database. Such attacks results in the maliciously altered data that are extremely hard to identify because there are no evidences to contradict their legitimacy.

Since preventive methods are not always effective in preventing malicious activities in a relational database [1], a lot of cyber-security professionals [23][28][39][33] as well as international and domestic computer-security standards [8][26][15] [4]recommended to audit the system and keep the transactions in a log file for forensic purposes. A simple example of audit logs is in a financial institution where every withdrawal or deposit of funds by users are recorded in a log file which later on if a user's balance didn't match with the spent or credited funds, log files could be checked to look for the possible mismatches. In a relational database, an audit log file could be seen as a historical data document which contains the executive transactions performed, the timestamp of the transaction and the identity of the user who performed the transaction on the relational tables. In practice, not only these log files are useful assets that could be analyzed to detect maliciously manipulated data by outsider adversaries, but they can also give a valuable insight into the activities of the privileged users on a relational database system [33].

Securing the audit log files themselves from being tampered is also difficult as

they are not immune from being altered[39]. For example in the super-user's attack example, it is possible that the user has the privilege to alter the audit log records in order to remove his footprint from the log table. Needless to say that the result of log file analysis is reliable until it is assured that the log records were submitted legitimately and were not tampered by any users, privileged or unprivileged, intentionally or unintentionally [22]. Therefore there is a need of a system which assumes that such attacks are unpreventable but guarantees that any attempt to tamper normal relational table as well as the log tables is evident and investigatable.

In this project, our objective was to design an auditing mechanism that is implemented on top of a relational database system that stores historical records in a transparent way. By doing that the system could detect malicious data manipulation and testify the trustworthiness of the stored historical data as well as authenticity of the records stored in the normal relational tables by analyzing the evidences provided. To achieve these objectives as well as creating a fully functional system, there were the following functional requirements that needed to be addressed.

- ***Immutability:*** To do so, all the transactions in a database system needs to be transparent so that any attempt to maliciously modify records in a normal relational table as well as tampering the historical records are evident.

- ***Fast querying:*** The relational database logs require a linear storage and time to store and fetch historical data of the records [3]. That means that as relational tables grow in size, querying for the historical records in those tables become more expensive [2]. In order to be functional, the proposed system needs to query for the historical records in an optimal and less expensive manner.

- ***Fast appending:*** The system needs to audit and append all executive transactions to the log file in a transparent way. In a system with thousands of transactions in a time constant, the system needs to attach the proof of work to the audited transactions, and append it to the log files computationally cheap.

- ***Identifying malicious activities:*** Given a historical data table, the system should be able to analyze the provided evidences and identifying any data manipulation both on a normal table and the historical data table.

We believe that our proposed system is powerful in the sense that it removes the user-based trust and testifies the credibility or invalidity of the records by analyzing verifiable evidences, ensures that privilege misuses such as altering the auditing mechanism is evident and finally it eliminates the data privacy concerns raised by utilizing third-party auditing tools. Our system utilizes many native to RDBMS tools such as audit loggings and cryptographic techniques, therefore it could be supported by all RDBMSes available today with least additional requirements.

1.2 Problem Definition

This needs to be mathematically written using formal logic.

Chapter 2

Background and Related Work

2.1 Overview

In this chapter, the concepts and definitions which are needed to implement the proposed system are identified. In the first section, the tools and concepts used from relational database management system and in the second section, the concepts in Blockchain technology are introduced. third section is also dedicated to discuss about the related work and the researches which have been done in this field and compare it with our proposed system.

2.2 Temporal Databases

Temporal database is any type of database that requires some aspect of time [9]. Since historical records have a timestamp which indicates the time of a transaction occurrence, in the proposed system, temporal database is a way to store and manage historical records of the transactions.

Definition 1. (Temporal Database): Let $r_i = \{r_1, r_2, ..., r_n\}$, be n number of tables in the relational database D. Denote the attributes of each table as $attr(r_i)$. A temporal table denoted r_i^T is a table with attributes $atr(r_i^T) = \{(updated, deleted) \in \mathcal{T}\} \cup attr(r_i)$ where $\mathcal{T} = \{t_0, t_1, ..., t_n\}$ is the time domain in which transactions

TABLE 2.1: Normal Relational Table r_1

id	item	value
22	Pencil	7.50$
23	Notebook	12.0$

TABLE 2.2: Temporal Table r_1^T

id	item	value	updated	deleted
21	Ruler	3.25$	2018-02-10	-
21	Ruler	3.25$	-	2018-02-20
22	Pencil	8.0$	2018-03-21	-
22	Pencil	7.50$	2018-03-30	-
23	Notebook	12.0$	2018-04-01	-

on r_i happened. A temporal database denoted D^T is the result of augmenting D by r_i^T:

$$D^T = D \cup \{r_i^T : r_i \in D\}$$

The temporal database D^T contains the entire history of the records ever existed in D.

Example 1. Given a normal relational table r_1 (Table 2.1) and a temporal table r_1^T (Table 2.2), the $attr(r_1) = (id, item, value)$ and $attr(r_1^T) = (id, item, value, updated, deleted)$. The result of query $q_1 = \sigma_{id=22}(r_1)$ is $\{(22, Pencil, 7.50\$)\}$ and the result of same query on the temporal table $q_2 = \sigma_{id=22}(r_1^T)$ is $\{(22, pencil, 8.0\$, 2018-03-21, NULL), (22, pencil, 7.50\$, 2018-03-30, NULL)\}$. Also the query $q_3 = \sigma_{id=21}(r_1)$ results in $NULL$ however, the same query on the temporal table $q_4 = \sigma_{id=21}(r_1^T)$ has the history of record with id = 21: $\{(21, ruler, 3.25, 2018-02-10, NULL), (21, ruler, 3.25, NULL, 2018-02-20)\}$.

Definition2. (Time domain): The time domain $\mathcal{T}$ consists of discrete timestamps $t_0, t_1, ..., t_n$ in which transactions on tables $r_i \in D$ happened. the range of

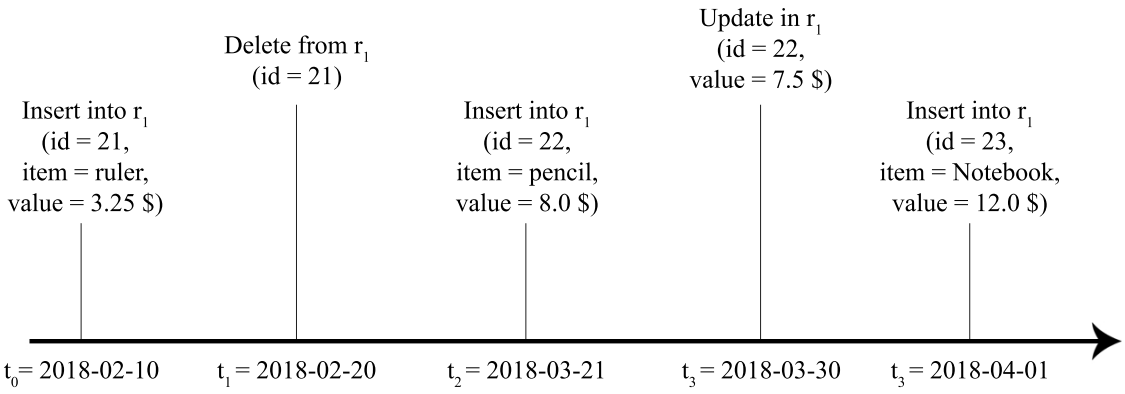

FIGURE 2.1: Timeline.

time domain is: $\mathcal{T} = [t_0, \infty)$ where t_0 is the timestamp in which the first record added to the table r_i. The time domain of a temporal table $r_i^T \in D^T$ is calculated by:

$$\mathcal{T} = r_i^T[updated] \cup r_i^T[deleted]$$

For example in the temporal table r_1^T (Table 2.2), the time domain is: $\mathcal{T} = [2018-02-10, 2018-04-01]$.

Definition 3. (Timestamps): A timestamp $t_i \in \mathcal{T}$ is a particular position in the time domain, in which particular transaction(s) happened. For example in the temporal table r_1^T (Table 2.2), "2018-03-30" is a timestamp in which the record with "id = 22" updated.

Definition 4. (Timeline): Let $u_i(t_j)$ be the total transactions on the tables $r_i \in D$ at timestamps $t_j \in \mathcal{T}$, where $i, j = \{0, 1, ..., n\}$. The $u_i(t_j)$ could be represented as an ordered set points on a vector. This vector is called a timeline for the transactions happened on $r_i \in D$. Figure x illustrates the concept of timeline.

Defionition 5. (Snapshot and snapshot-queries): Given a temporal table $r^T \in D^T$ and a timestamp $t \in \mathcal{T}$, we denote $s(t)$ to be the table instance that obtained by calculating the $\{max(r^T[m])|t : m \in r)\} - r^T[deleted]$ for $\mathcal{T} \leq t$.

Note that $s(t) \in D^T$ but not necessarily $s(t) \in D$. A snapshot is a materialized version of $D(t) = \{s_1(t), s_2(t), ..., s_n(t)\}$. A snapshot-query is an arbitrary relational query on $D(t)$.

We can construct the snapshots using simple windowing functions (as in supported by PostgreSQL [24]).

```
snapshot(r, t) =
 WITH T AS (
  SELECT id, {last_value(x) as x : x ∈ attr(r)} OVER W
  FROM r^T
  WHERE updates ≤ t
  WINDOW W AS PARTITION BY id ORDER BY updates
 )
 SELECT id, {x : x ∈ attr(r)} FROM T
 WHERE NOT T.deleted
```

The query snapshot(r, t) computes the snapshot of r at timestamp t by applying the latest update of each tuple up to timestamp t, while removing tuples that have been deleted.

2.3 Blockchain

In a big picture, Blockchain technology is a distributed trusted public ledger that is tamper-proof and it is open to anyone to inspect [27]. Although today there are a wide range of applications that are using Blockchain [5], but Blockchain is widely known for its application in cryptocurrencies such as Bitcoin [25]. As a matter of fact, Blockchain is the secret why cryptocurrencies are secure, work

in a decentralized network, do not need any central verification system and the users in the system are anonymous [13]. While there are many functionalities within a Blockchain system, such as user anonymity, being public and no need for a central verification system, in this project, for the sake of confidentiality and transparency, not all functionalities within the Blockchain is employed. To understand how Blockchain has been utilized in the proposed system, there are some definitions which needs to be discussed about.

Definition.(Cryptography): Cryptography is a way of secure communication between parties in a network while adversaries might also be present. Using Cryptography, messages sent or received are encrypted so that the adversaries cannot read the normal form of the message. This communication is established through various steps such as cryptographic key assignment, encryption and decryption of messages or digitally signing a message and verifying the digital signatures.

Definition.(Cryptographic Keys): Let u be the authenticated user who is using database D. By the creation of the profile of u in the system, a set of strings $< K_{priv}, K_{pub} > \in \mathbf{N}^+$ is generated and assigned to the user where K_{pub} is the public key that is accessible to everyone on the system, and K_{priv} is the private key that is known only to u. These keys are used to encrypt/decrypt messages which is transmitted between the users. **Definition.(Assymetric Encryption)**: Let E be the encryption algorithm, D be the decryption algorithm, m be the message which needs to be encrypted and c be the encrypted message. Given the cryptographic keys $< K_{pub}, K_{priv} >$, An encryption technique is said to be asymmetric if:

$$c = E(K_{pub}, m) \text{ and } m = D(K_{priv}, c)$$

or

$$c = E(K_{priv}, m) \text{ and } m = D(K_{pub}, c)$$

Therefore:

$$D(E(m, K_{pub}), K_{priv}) = m$$

and

$$D(E(m, K_{priv}), K_{pub}) = m$$

Note that, if K_{pub} is known, and $E(K_{pub}, m)$ is also known, in asymmetric encryption method, it is impossible to get m without K_{priv}.

Figure ?? shows the basic steps to send and receive messages between two parties in a secure way by utilizing asymmetric encryption technique. In this scenario, the sender has access to the recipient's public key only. The sender encrypts the document which is to be transferred by using the recipients public key. In asymmetric technique, the only way to decrypt the document is to use the recipient's private key which the is only known to the recipient. This is a powerful technique to ensure that the adversaries cannot read the document in the middle of the connection.

Definition. (Hash function): Assume $m \in \mathcal{A}$ to be the message with an arbitrary size chosen from domain $\mathcal{A}$. $hash(m) \rightarrow sketch$ is a function that maps the m of any size from domain $\mathcal{A}$ to a fixed size string (normally 256 bits) in a smaller domain $\mathcal{B}$.

Definition. (Digital Signature): Digital signature is used to ensure that the digitally transfered data has not altered while transferring. Also it verifies whether or not the transfered data was submitted by a recognized source.
Let m be the document which needs to be digitally signed and transfered. Denote h as hash function, E as encryption algorithm and D as decryption algorithm. In order to digitally sign a document and verify a signature, following steps should be taken:

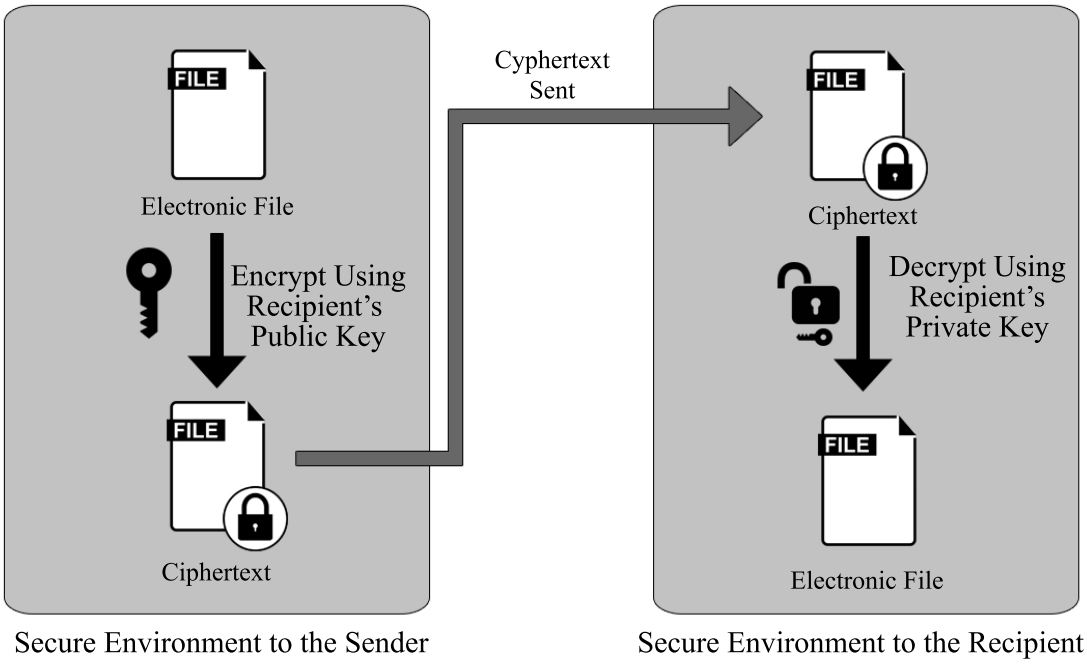

FIGURE 2.2: Asymmetric Encryption Technique.

- **Step 1.** $S_r = h(m)$
- **Step 2.** $c = E(S, K_{priv})$
- **Step 3.** m and c are sent

In order to verify:

- **Step 1.** m and c are received
- **Step 2.** $S_t = h(m)$ is calculated
- **Step 3.** $S_r = D(c, K_{priv})$ is obtained
- **Step 4.** m is valid if $S_t = S_r$ and invalid if $S_t \neq S_r$

Figure ?? depicts the steps that needs to be taken for digitally sign a document and verifying a digital signature.

Definition. (Blockchain for This Project): Let rec_i be the records stored in r_i^T.

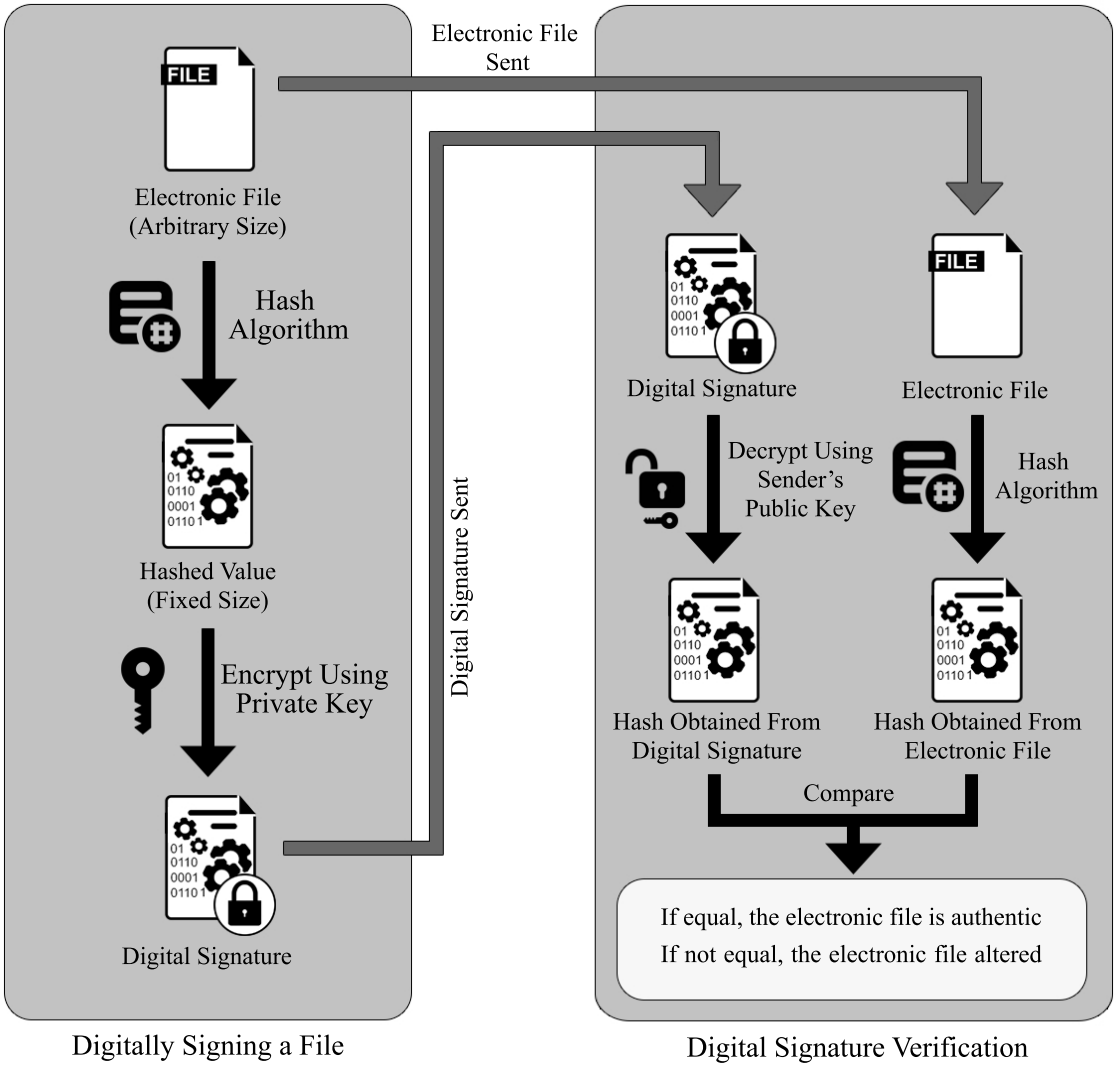

FIGURE 2.3: Digital Signature Creation and Verification.

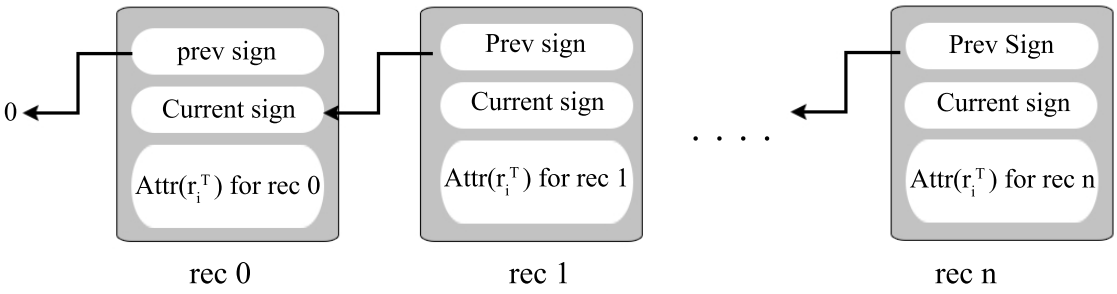

FIGURE 2.4: Idea of the Blockchain for the proposed system.

We denot $Sign$ as $S_r(attr(r_i^T \text{ for } rec_i))$ which is digital signature of attributes of rec_i. We can create a blockchain of historical records by storing the signature of previous record as an attribute of current record. The idea of the Blockchain has been depicted in figure ??.

2.4 Related Work

There has been a wide range of studies on ensuring the trustworthiness of records in a database from different perspectives. These studies ranges from establishing trust between nodes in a real-time distributed systems [19] to secret sharing schemes in cloud databases [7] and utilizing logfiles for forensic purposes [33]. In this project, the assumption is that the preventive models are not able to stop the advarsaries from manipulating the data of a relational database system, therefore a tamper-evident log table has been offered to evaluate whether or not the data has been altered.

Database audit logs contain valuable information such as any insertion, deletion and modification of the records performed in a database along with the timestamp of the performed tasks. The United States department of defense

in its "Trusted Computer System Evaluation Criteria" document and under requirement 4 pointed out the importance of auditing the transactions in a computer system in a secure and efficient manner [4]. In this document also protecting the audit logs from modification or destruction has been stressed out. "ISO/IEC 27001" standard published by International Organization for Standardization and International Electrotechnical Commission [17], "Guide to Computer Security Log Management" standard published by United States National Institute of Standard and Technology [26], eHealth Ontario HER standard [8], European Commission Information System Security Policy C(2006) 3602 [16] and the University of British Columbia's information Security Standard #17 [15] are a few of many other international or domestic regulations that have suggested guidelines to secure computer systems by using historical records such as audit logs.

Analyzing audit logs for forensic purposes has been the topic of research by many scientists however, since the trustworthiness of the results from log table analysis has a direct relationship with the authenticity of the records of the log file, a lot of studies have been carried out to make log tables tamper-proof. Haber *et al.*[12] proposed a basic methodology by utilizing timestamping, hashing and digitally signing techniques in order to make unmodifiable historical records for digital documents. Peha in [29], offered a mthod to detect log tampering by one way hashing and employing multiple trusted notaries to keep track of all transactions. The author argued that if any notary decided to falsify the transaction, the attempt is discovered by other notaries. Snodgrass *et al.* [34] also offered one way hashing mechanism and employed trusted notaries, however in order to enhance the security of the method offered by Peha, they offered to hash the records with a timestamp of previous transaction modification. Schneier *et al.*[32] offered a cryptographic-based mechanism to make the

log files nearly impossible for the attacker to alter. The authors claimed that by utilizing their approach, it is guaranteed that the outsiders cannot modify the log files.

Although the afformentioned researches might have promising results to protect the historical records from being compromised by an outsider, but they have one thing in common which is the role of an insider to carry out malicious attacks is forgotten. Also hiring third-party software/hardware may bring up a lot of privacy concerns. Therefore, unlike Peha [29] and Snodgrass *et al.* [34], our proposed system doesnot require a third-party notary to attest the authenticity of the transactions and unlike Schneier *et al.* [32], our proposed system doesnot put trust on any user on the system.

The role of privileged users in acting maliciousely in a database has been discussed by many researchers [3] [39]. Liu *et al.* offered a network-based auditing mechansim which also keep track of privileged users' activity and performs audit analysis through event correlation. Wagner *et al.* [38] proposed a mechanism to detect database file tampering by looking for inconsistencies in the database's storage. The authors argued that the databases follow patterns in storage which even the privilaged users have no access to. Therefore, if a record is deleted maliciousely in the log file, the inconsistency in the storage is evident for a period of time. Unlike mentioned propsed methods, our system uses inherent to RDBMS tools and doesnot require a network-level-auditing mechansim or having access to the server's storages, therefore it could be easily implemented on remote servers and relational databases on cloud storages. Also, our proposed system not only discovers maliciously deletion of the records but also identifies any malicious modification without any time constraints.

For the sake of gathering verifiable evidences, in our proposed system, any changes to the database regardless of the users' access privilages needs to be

tracked. This action needs to be done by utilizing inherent to DBMS tools. Fabbri *et al.* [10] discussed about SELECT triggers, required database spicifications, efficient implementation techniques and their role for data auditing. Hauger *et al.*[14] also discussed about using the triggers in the database for forensic purposes. Triggers are supported by RDBMS and they are good candiates to be used in the proposed system but it is naive to assume that a simple trigger is secure enough to be used for forensic purposes.

To make the audit logs secure enough and tamper-evident, the proposed system utilizes Blockchain Technology for this purpose. Blockchain is widely known as the backbone of cryptocurrencies such as Bitcoin. [25]

Bibliography

[1] Paul Ammann, Sushil Jajodia, and Peng Liu. "Recovery from malicious transactions". In: *IEEE Transactions on Knowledge and Data Engineering* 14.5 (2002), pp. 1167–1185.

[2] Mohammadamin Beirami, Ken Pu, and Ying Zhu. "Towards Optimal Snapshot Materialization To Support Large Query Workload For Append-only Temporal Databases". In: *2018 IEEE BigData Congress*. 2018.

[3] Scott A. Crosby and Dan S. Wallach. "Efficient data structures for tamper-evident logging". In: *2009 the 18th conference on USENIX security symposium*. USENIX Association. 2009, pp. 317–334.

[4] *Department of Defense Trusted Computer System Evaluation Criteria*. Standard. United States Department of Defense, 1985.

[5] Vikram Dhillon, David Metcalf, and Max Hooper. *Blockchain Enabled Applications*. Apress, Berkeley, CA, 2017.

[6] Jiang Du et al. "DeepSea: Progressive Workload-Aware Partitioning of Materialized Views in Scalable Data Analytics." In: *EDBT*. 2017, pp. 198–209.

[7] Rahul Dutta and Annappa B. "Privacy and trust in cloud database using threshold-based secret sharing". In: *2013 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*. 2013, pp. 800–805.

[8] *eHealth Ontario EHR Standard - Security Logging and Monitoring Standard*. Standard. eHealth Ontario, 2018.

[9] Ramez Elmasri and Shamkant B. Navathe. *fundamentals of database systems, sixth edition*. Addison-Wesley, 2010.

[10] Daniel Fabbri, Ravi Ramamurthy, and Raghav Kaushik. "SELECT triggers for data auditing". In: *2013 IEEE 29th International Conference on Data Engineering (ICDE)*. 2013, pp. 1141–1152.

[11] Maryam Ghasemaghaei, Khaled Hassanein, and Ofir Turel. "Impacts of Big Data Analytics on Organizations: A Resource Fit Perspective". In: *AMCIS*. 2015.

[12] Stuart Haber and W. Scott Stornetta. "How to Time-Stamp a Digital Document". In: *Advances in Cryptology-CRYPT0' 90*. Springer Berlin Heidelberg, 1991, pp. 437–455.

[13] Hanna Halaburda and Miklos Sarvary. *Beyond Bitcoin: The Economics of Digital Currencies*. Palgrave Macmillan, New York, 2016.

[14] Werner K. Hauger and Martin S. Olivier. "The role of triggers in database forensics". In: *2014 Information Security for South Africa*. 2014, pp. 1–7.

[15] *INFORMATION SECURITY STANDARD #17 Logging and Monitoring of UBC Systems*. Standard. University of British Columbia, 2014.

[16] *Information System Security Policy C(2006) 3602*. Standard. European Commission, 2006.

[17] *Information technology – Security techniques – Information security management systems – Requirements*. Standard. ISO/IEC, 2013.

[18] Rohit Jain and Sunil Prabhakar. "Trustworthy data from untrusted databases". In: *2013 IEEE 29th International Conference on Data Engineering (ICDE)*. IEEE. 2013, pp. 529–540.

[19] Grace Khayat and Hoda Maalouf. "Trust in real-time distributed database systems". In: *2017 8th International Conference on Information Technology (ICIT)*. 2017, pp. 572–579.

[20] Bernhard Klein et al. "Analysis of Log File Data to Understand Mobile Service Context and Usage Patterns". In: *International Journal of Interactive Multimedia and Artificial Intelligence* 2.3 (2013), pp. 15–22.

[21] Donald Kossmann and Konrad Stocker. "Iterative dynamic programming: a new class of query optimization algorithms". In: *ACM Transactions on Database Systems (TODS)* 25.1 (2000), pp. 43–82.

[22] Chung-Yi Lin et al. "Secure logging framework integrating with cloud database". In: *2015 International Carnahan Conference on Security Technology (ICCST)*. 2015, pp. 13–17.

[23] Raffael Marty. "Cloud application logging for forensics". In: *Proceedings of 2011 ACM Symposium on Applied Computing*. ACM. 2011, pp. 227–230.

[24] Bruce Momjian. *PostgreSQL: introduction and concepts*. Vol. 192. Addison-Wesley New York, 2001.

[25] Satoshi Nakamoto. *Bitcoin: A Peer-to-Peer Electronic Cash System*. https://bitcoin.org/bitcoin.pdf. 2008.

[26] *NIST Special Publication 800-92: Guide to Computer Security Log Management*. Standard. National Institute of Standards and Technology, 2006.

[27] OECD. *OECD Science, Technology and Innovation Outlook 2016*. OECD, 2016.

[28] Alecsandru Patrascu and Victor-Valeriu Patriciu. "Logging for Cloud Computing Forensic Systems". In: *International Journal of Computers Communication and Control* 10.2 (2015), pp. 222–229.

[29] Jon M. Peha. "Electronic commerce with verifiable audit trails". In: *In Proceedings of ISOC*. 1999.

[30] Doug Rose. *Data Science*. Apress Berkeley,CA, 2016.

[31] Mohammad Sadoghi et al. "Reducing database locking contention through multi-version concurrency". In: *Proceedings of the VLDB Endowment* 7.13 (2014), pp. 1331–1342.

[32] Bruce Schneier and John Kelsey. "Cryptographic Support for Secure Logs on Untrusted Machines". In: *Proceedings of the 7th Conference on USENIX Security Symposium - Volume 7*. SSYM'98. USENIX Association, 1998, pp. 53–62.

[33] Arunesh Sinha et al. "Continuous Tamper-Proof Logging Using TPM 2.0". In: *International Conference on Trust and Trustworthy Computing*. Springer. 2014, pp. 19–36.

[34] Richard T. Snodgrass, Shilong Stanley Yao, and Christian Collberg. "Tamper Detection in Audit Logs". In: *Proceedings of the Thirtieth International Conference on Very Large Data Bases - Volume 30*. VLDB '04. VLDB Endowment, 2004, pp. 504–515.

[35] Mohammad Karim Sohrabi and Vahid Ghods. "Materialized View Selection for a Data Warehouse Using Frequent Itemset Mining." In: *Jcp* 11.2 (2016), pp. 140–148.

[36] Domenico Talia, Paolo Trunfio, and Fabrizio Marozzo. *Data Analysis in the Cloud: Models, Techniques and Applications*. Elsevier, 2015.

[37] Yun Tian, Yanqing Ji, and Jesse Scholer. "A prototype spatio-temporal database built on top of relational database". In: *2015 12th International Conference on Information Technology-New Generations (ITNG)*. IEEE. 2015, pp. 14–19.

[38] James Wagner et al. "Carving database storage to detect and trace security breaches". In: *Proceedings of the Seventeenth Annual DFRWS USA*. Elsevier. 2017, s127–s136.

[39] James Wagner et al. "Detecting Database File Tampering through Page Carving". In: *2018 21st International Conference on Extending Database Technology*. 2018, pp. 121–132.