IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received September 11, 2021, accepted September 28, 2021, date of publication October 4, 2021, date of current version October 13, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3117733

Using Blockchain in Cloud Computing to Enhance Relational Database Security

RUBA AWADALLAH AND AZMAN SAMSUDIN
School of Computer Sciences, Universiti Sains Malaysia, Penang 11800, Malaysia

Corresponding authors: Ruba Awadallah (awadallah.rn@gmail.com) and Azman Samsudin (azman.samsudin@usm.my)

This work was supported in part by Universiti Sains Malaysia under Grant LRGS/MRUN: 203.PKOMP.6777005.

ABSTRACT Cloud computing has now become a very standardised concept in our society. However, many modern applications need a better level of security that includes saving data from internal breaches. Thus, cloud databases need effective security mechanisms to keep track of data modifications. This paper will introduce the enhanced structure of cloud relational database (RDB) based on blockchain technology (BC) named BC over cloud-RDB. Through a self-verification mechanism, it enables the client to detect and prevent erroneous RDB manipulation. We proposed two systems to improve cloud-RDB namely, agile BC-based RDB and secure BC-based RDB. Both are distributed among several cloud service providers based on the Byzantine Fault Tolerance consensus. Additionally, both rely on linking records to each other using the SHA-256. At the same time, secure BC-based RDB uses a proof-of-work consensus to make data offensive operation impossible. On the basis of performance of both systems' and security analysis, the agile BC-based RDB is highly suggested for the high throughput database. On the other hand, the secure BC-based RDB is recommended for RDB that contains sensitive data and low throughput performance. The improved RDB is flexible and can be operated according to the data owner's specifications.

INDEX TERMS Blockchain, cloud computing, confidentiality, homomorphic encryption, integrity, privacy, relational database.

I. INTRODUCTION

Metastructure is a crucial difference between cloud computing and classical IT infrastructure. It includes network-enabled and remotely accessible management control components [1]. The cloud service provider (CSP) manages cloud computing systems remotely and runs the cloud smoothly. Furthermore, cloud computing contributes to double up on each layer of its infrastructure. It includes a physical layer responsible for creating the cloud and the virtual layer used and managed by the end-user (client) [2]. Therefore, cloud computing provides on-demand unlimited virtual storage and various resources to afford better computing power. Since cloud computing is based on a pay-as-use system, it is financially viable [3]. In addition, the distributed design of data storage promises to save data from loss by frequently producing backups.

Typically, there is an agreement between the cloud users and CSPs on the type and quality of the service provided by CSPs [4]. Cloud computing platforms keep the client data inside cloud databases, whether structured or unstructured. The structured data is attributed to being inside the cloud relational databases (RDB), the most famous renewed databases available [5]. A relational database management system (RDBMS) uses different languages, usually the Standard Query Language (SQL) is employed, which can store interlinked data in tabular forms, thereby allowing applications to access the data. These tabular databases are also called relations, and they are collections of records with similar attributes [6]. Thus, RDB offers several advantages in documentation, simplification, consistency, location integrity and reliability [7]. [8] demonstrated that for processing database applications in the cloud, SQL can execute more complex queries than other alternatives. It also enables transactions, ensuring that atomic modifications are made to the data while keeping one complete copy from which all other data is copied [9]. The most trending cloud products based on RDB are Google BigQuery, Azure SQL Database, Amazon Redshift, and many more [10]. This study focuses on data security through the RDB stored in cloud computing servers.

The associate editor coordinating the review of this manuscript and approving it for publication was Vlad Diaconita.

Even with all the advantages of cloud computing, it still faces some challenges. According to the Cloud Security Alliance (CSA) [11], the most prominent of which is data security. [12] revealed that there is possibility of client data being compromised through different cloud computing layers, that may affect different data security requirements such as; privacy, confidentiality, integrity, and loss in some cases [13]. The cloud metastructure is where data are processed in the cloud and could be physically located at any data centre worldwide, and that is the main reason behind this dilemma [14]. In other words, the responsibility for data is transferred entirely from the client to the CSP.

Moreover, cloud computing management is centrally controlled, which creates a fear in clients of losing control over their data. Since internal threats are the leading cause of more than 60% of data breaches [15], searching for a trusted CSP is not an easy task as it cannot guarantee the reliability of its employees [16]. Also, [17] indicates that the right to amend terms and conditions lies with the CSPs that may lead them to influence client data whenever they want. Another factor is poor data protection policies and the growing number of devices that can access sensitive data. Furthermore, according to the Computer Security Institute (CSI) survey [18], it was found that the internal breaches are more severe than their foreign counterparts because insiders know the system and can attack valuable records, while outsiders only tend to steal what they can access. Internal attacks are also costly as stated in [19], the current average annual cost of an internal attack comes in at US $13.7 million per year for risks that take more than 90 days to resolve. In contrast, those that take less than 30 days to fix cost an average of US $7.12 million.

A. RELATED WORKS

In conventional RDBMS, the data owner is responsible for managing and tracking the database. These tasks are passed to CSP when switched to the cloud servers. Keeping the local services accountable for these tasks violates the principle of cloud computing. However, insiders and the relationships between data items can play a critical role in launching internal attacks. At the same time, the knowledge base of the insider includes the values of the data items that the insider has previously accessed such as, the history accesses to data records by insiders [20], [21]. Therefore, several studies such as [22]–[25] exploited the knowledge bases to track the accumulated knowledge of insiders. [22], [23] addressed some internal attacks in cloud RDB, and suggested similar models to mitigate these attacks and improve the data availability. Whereas [24], [25], presented risk estimation models depending on Mobile Edge Computing (MEC). In comparison, [25] suggested the use of MEC to create knowledge graphs and dependency graphs for each insider.

Among various methods applied for preventing attacks, [26] suggested a manageable model, which catches and blocks internal threats at policy enforcement points and also provides minimal cost on the performance of policy application points and policy decision points. Also, [27] linked the effective detection of internal attacks that occur by impersonating user behaviour in different environments.

Meanwhile, [28]–[31] relied on a combination of encryption schemes to prevent data breaches, but also emphasized that no single cryptosystem can support all types of queries. Despite the different proposed methods suggested in this field, they unanimously espoused using homomorphic encryption (HE).

Some researchers have adopted blockchain technology (BC) to raise security efficiency in the cloud. For example, [32] formulated a blockchain-based secret-data sharing model that allows personal health record system users to access their records from cloud storage after verifying their identities through BC. In the same vein, in health, [33] presented a similar framework for securing outsourced electronic health records using BC technology. Several papers have also applied the distribution architecture of BC technology in the Internet of Things (IoT). [34] proposed a hierarchical and scalable BC-based trust management protocol, which is resilient against known malicious attacks, like bad-mouthing, ballot-stuffing and cooperative attacks. Also, [35] proposed using the tensor train in cloud-fog computing for industrial data applications, which promises to keep user's data private and secure against clouds and fogs. [36] used BC to tackle the reliability challenges in cyber-physical-social systems, which resulted in a decentralised model for a data-sharing platform between multiple cyber-physical-social data providers.

Also, databases have embraced BC technology in the industry. For example, Hyperledger Fabric [37] debuted to provide no-tamper data storage features. However, the Fabric state database supports a narrower range of SQL queries than other RDBs. Another example of databases based on BC technology is BigchainDB [38] that uses MongoDB [39] as the primary database, thus demonstrating that it only performs NoSql queries and excludes the high capacity features of RDBs. In addition, LedgerDB [40] offers BC services that assure data integrity and non-repudiation at several levels. However, unlike typical RDBs, this platform only allows a single data table which focuses on documenting transactions between users rather than keeping different sorts of information. Also, EthernityDB [41] relied on maintaining all the database data on the Ethereum network. Accordingly, any process of queries conducted, will be through smart contracts. Nevertheless, this system faces several limitations, including restricting query operations and their efficiency.

Previous studies have sought to prove their efficiency in ensuring the availability of client data against internal breaches. However, none of these studies was concerned with the integrity of the computations applied to the data. The CSP can exploit a requested tampering query authorised by the client to violate that query. The seriousness of these violations lies in the client's inability to detect them even if the data is encrypted, because it occurs with a legal appearance and through an entity who is authorised to make the change. This work contributes to the decentralisation of CSP authority on client data. Also, it examines how the client can verify

the CSP's implementation of the requested queries without any alteration.

[42] is the most relevant work to our research goal as they are trying to develop database security using BC. They proposed a TDRB middleware that relies on hashing critical data in the RDB and migrating it to the BC synchronously. Although the study demonstrated its ability to detect tampering with data, it has many gaps. First, this proposal was limited to detecting data manipulation and did not promise a radical solution to prevent internal violations. Second, the suggestion can be applied to stored data but is not compatible with data exposed to external processes as in cloud computing.

Lately, [43] generated a model of distributed CSPs to frequently release their masterhashes within the Bitcoin and Ethereum network. In the model the client performs a self-verification process after a specific time, which ensures safety of the computed data from different breaches and attacks. We will develop this proposal by adopting the components of BC within the cloud RDB. Hence, tracking and restricting CSP changes over the database and determining the records that are affected by the internal attack.

According to [1], pointed out the basic security requirements, which the utilised countermeasures must provide to preserve data from internal violations, some are as follows:

- *Data Privacy:* Ensures that the client has the right to control how their data is handled and processed.
- *Data Confidentiality:* Ensures that client data is not disclosed to any unauthorised entity.
- *Data Integrity:* Ensures that the new processing operations are not random, are requested by the client and guarantee data consistency throughout its entire lifecycle.

In our proposal we attempt to enhance RDB security against internal threats and other data breaches, we also suggest encrypting data using HE. A malleable public key cryptosystem maintains data confidentiality and privacy during various external SQL queries without decryption. However, encryption alone does not guarantee all security requirements. To fulfil the integrity of the processed data from internal breaches and eliminate the central authority of CSP, this study relies on BC. It is a trusted technology that uses well-known computer science mechanisms, standard cryptographic features, and record-keeping principles. In this paper, we will simulate the essential components of BC to apply in RDB that is stored and processed in the cloud computing environment. Thus, placing significant restrictions against internal abuses and revoking the central authority of CSP. Furthermore, the proposed system does not interfere with the encrypting process description, therefore it is suitable for all HE cryptosystems.

Next Section II explains the detail of the HE and BC components that were used in establishing our proposal. Then the proposed BC over cloud-RDB design concept is highlighted in Section III. Section IV proves the concept and explains the protocol of running the proposal. Henceforth, Section V discusses the theoretical analysis of the proposed system in terms of security level, implementation, overhead cost and performance. Concluding remarks and an evaluation of the proposal is written in the last section, i.e. Section VI.

II. DESIGN PRELIMINARIES

This paper provides an optimal solution based on encrypting data using HE cryptosystems and simulating BC technology in the cloud RDB structure. In the following Subsections II-A and II-B a further explanation on both is provided.

A. ENCRYPT DATA USING HOMOMORPHIC ENCRYPTION (HE)

HE is a type of asymmetric cryptosystem in which data is encoded into ciphertext via public key, and decryption is done using a private key. Arising from its similarity, the specific character of the HE cryptosystem enables the system to perform operations on the encrypted data even when there is no access to the private decryption key. When the same arithmetic operation is used, the same result is achieved regarding the raw data. In Mathematics, an encryption is homomorphic, if from $Enc(x)$ and $Enc(y)$, it is therefore possible to calculate $Enc(f\ (x, y))$, where f can be: $+$, $\times$, $\oplus$ and without the use of the private key [44].

To use HE cryptosystems, a client should undertake dual operations (keys generation and encryption operations) before outsourcing the data to the cloud. Furthermore, CSP is given the authorisation to undertake evaluation operations on the encrypted data. Once the client wants to return the raw data, a decryption operation is performed. Practically, the client needs to start with *KeyGen* operation to get two keys represented as a public key (*Puk*) and a private key (*Prk*). *Enc* process ensues at the subsequent stage. Next, the client uses the *Puk* to encrypt the data and send the produced ciphertext together with *Puk* to the CSP repository. Each time the client wants to update outsourced ciphertext, they can authorise CSP to carry out *Eval* operation, which produces a new ciphertext. As soon as the client receives the resultant ciphertext, they can operate the *Dec* operation using data' *Prk* to get back the raw data.

The features of HE cryptosystems have evolved, starting with a system that can evaluate one type of process (either addition or multiplication) at a time, called Partial HE [45]. Whereas, an advanced system known as Somewhat HE is developed to implement both arithmetic operations simultaneously, but for a limited number of processes [46]. While in comparison to the above two, a Fully HE is a system where both operations can be functional without limitations [47]. From then onwards, HE cryptosystem has attracted a lot of interest in different fields, and studies conducted have found more specific characteristics and typologies fitting it. Our proposal is perfect for any HE cryptosystem that the client decides to use before sending data to the cloud server.

Works of literature have proven the quality of various HE cryptosystems in providing outsourced storing and computing services with the utmost privacy and

confidentiality [48]–[50]. All HE cryptosystems achieve data confidentiality by giving the *Puk* information to apply the assigned mathematical operations. Also, these cryptosystems have proven their competence in preserving data privacy by not allowing unauthorised parties to modify the data.

However, HE is malleable in nature, making it impossible to achieve IND-CCA2 security notation [51]. Theorem 1 proves that although the adversary cannot distinguish the ciphertext (c_i) information, they can apply an adaptive chosen-ciphertext attack. [52] indicated towards the IND-CCA2 attacks in the cloud environment that CSP could carry out without the client's knowledge.

Theorem 2.1: If a cryptosystem is an HE cryptosystem, then it cannot be IND-CCA2 secure.

Proof: The client begins preparing the keys using *KeyGen* process. Then *Puk* is used to *Enc* the data and send these resulted ciphertexts $Enc_{Puk}(c_i)$ to the cloud servers. As such the CSP will have ciphertexts c_0 and c_1, and has received a challenge $Enc_{Puk}(c_b)$ from which it must be established if $b = 0$ or $b = 1$. The CSP may therefore asks the encryption of some known constant k, and consequently deploy the homomorphic features of the cryptosystem to calculate $Enc_{Puk}(c_b) \times Enc_{Puk}(k) = Enc(c_b + k)$. The CSP then relays a decryption query to the oracle in order to understand $c_b + k$, and can simply establish which of c_0 or c_1 is the challenge. □

Since the cloud is a centralised management system, and HE cannot achieve superior security against data integrity, the client's data is still vulnerable to internal threats that may lead to loss or manipulation in the database without disclosing it. As such, cloud clients cannot rely on the primary relationship structure between them and CSP. To eliminate any internal threats and breaches of the database, we will rely on BC technology. It is a reliable technology since it is related to data immutability, transparency and traceability, and characterised by decentralisation of management. In this paper, we will simulate BC into cloud RDB infrastructure.

B. RELATED BLOCKCHAIN (BC) COMPONENTS

Generally, BC is a type of database that brings information together into groups, known as blocks. Each block contains groups of information within a specific capacity, e.g., Bitcoin blocks have the cryptocurrency exchange information between nodes that are involved in the network communication without a central authority in a peer-to-peer fashion [53].

BC database establishes an irreversible timetable of data when applied in the network. Once the block is completed, it is proven correct and becomes part of the chain. The exact timestamp is given to each block in the chain as it is attached to the chain [54]. The timestamp is a hash code generated by a mathematical function that converts numeric information into a string of numbers and letters. If this information exposes to alteration, then the hash code is also changed [55]. The Secure Hash Algorithm (SHA)-256 has been adopted in various projects built on the BC. For example, it is a significant part of Bitcoin mining. The fact behind the distinction of this algorithm is that it is a deterministic function, fast in computation, resistant to pre-image and second-image attacks, and collision-resistant [56], [57]. This paper relies on the SHA-256 algorithm to take advantage of its security features.

Mining is a network-wide competition among nodes to fix new blocks to the BC. For illustration, in the Bitcoin network, every 10 min on average, Bitcoin develops a mathematical riddle built on SHA-256 [58]. The Proof-of-Work (PoW) is a mathematical mechanism used to find the input solution i of SHA-256 of two variables concatenated: the nonce variable s and the produced record hash P_x, as explained in Equation (1) [59].

$$i = nonce(s) \,||recordhash(P_x) \tag{1}$$

The nonce is the variable used to produce different outputs of a cryptographic function; it is also the key value of adjusting the target difficulty [58]. The target T is a 256-bit value indicating the correct PoW value. So that the value of PoW must be equal to or below the maximum target value. The number of possible solutions reduces with increasing the number of leading zeros, making PoW more difficult. The easiest PoW, which is the highest possible value for T is given as $T_{max} = 2^{224}$.

The target difficulty regulates the speed of creating a new record. It is the ratio between the maximum target and the current target, as shown in Equation (2). After n blocks, the network's average block time is assessed; if it is longer than the estimated block time, the PoW difficulty level is reduced and vice versa.

$$D = \frac{T_{max}}{T_c} \tag{2}$$

Afterwards, each newly mined block distributes across the network, whereby each node verifies it separately before adding it to the BC. After attaching the valid new block, the building of a new block starts on top of this block. As such, a collective effort between nodes through the network means that the BC is continuously developed.

III. PROPOSAL DESIGN

The design of BC over cloud-RDB is based on simulating BC security components over the RDB stored and processed in cloud servers. This study is an improved proposal of [43] to provide a client self-verification system that detects and restricts internal threats applied to data computations in the cloud. The following subsections discuss the generated systems' network structure and system protocol to facilitate the concept proposal.

A. SYSTEM NETWORK SETUP

This system follows [43] in hiring at least four CSPs to participate in the network of computations by simulating Byzantine Fault Tolerance (BFT) consensus [60]. The BFT statement indicates that the upper limit on Byzantine faults f should be less than the number of included nodes over 3 as $f < N/3$ [61]. Thus, at least $N = 4$ CSPs act as

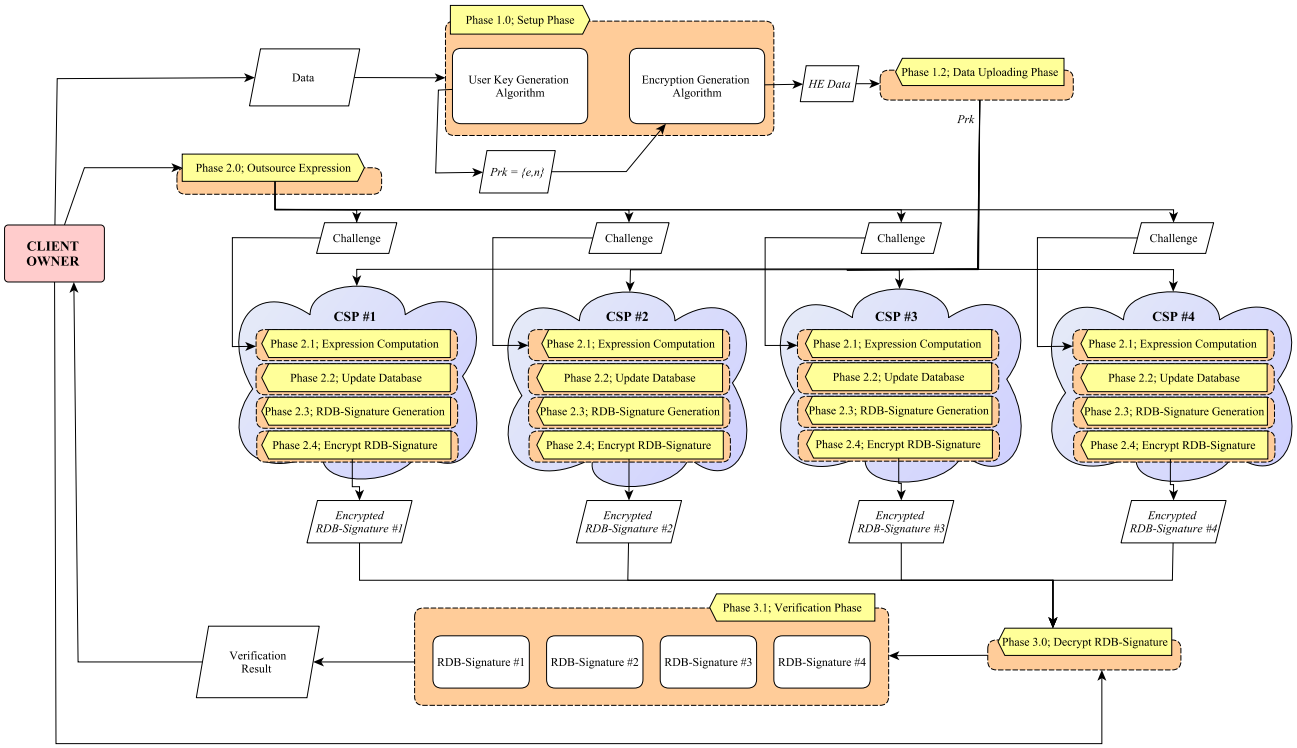

FIGURE 1. BC over cloud-RDB flow diagram.

a solo server on a full copy of the client's database. In other words, the client will privately distribute the database to four CSPs instead of a single central CSP. Autonomously they are responsible for storing, managing, and producing new data records. Moreover, CSPs are connected independently with the client to prevent a 51% attack (when adversaries control over 50% of the network) [62].

This proposal aspires for the client to reach a consensus on the integrity of the applied changes through self-verification. Therefore, during a particular time, the client asks each CSP to send their RDB-signature independently. The received RDB-signature values determine whether each CSP is processing correctly and the data has not been attacked. Accordingly, the client takes advantage of the various cloud computing services with security optimisation. A flow diagram of our proposal BC over cloud-RDB is illustrated in Fig. 1.

Initially, the client uses HE cryptosystems to encrypt the data and store it in the proposed RDB. Then, the client clones this database according to the number of hired CSPs. Finally, the client releases this database confidentially into hired CSPs. The query mechanism is either a request of services over homomorphically encrypted data or a verification request of the previously applied processes.

Once a query is requested, all CSPs must first apply the required computations over the homomorphically encrypted data and then store it in a new record. Then, they have to link the new record with the record that precedes using SHA-256. Thus, forming a series of linked records according to the age of the required query.

After a certain query q_i, the client requests all hired CSPs for the RDB-signature. Which in turn results from five sequential processes; two aggregations, two hashes, and one encryption operation. The multiple hashing allows the client to recognise the affected record if internal abuse occurs. At first, each CSP collects all records' hashes of each table to produce a table hash value. Next, CSP collects all tables' hashes to produce the RDB-signature value. Finally, CSP encrypts the produced RDB-signature result using either the applied HE or any more affordable public key specified by the client. As the purpose of encrypting RDB-signature is only to keep this data from pollution attacks until it reaches the client. In other words, every CSP will do the following operations to send the RDB-signature to the client:

1) Independent aggregation of all records hashes in a specific table.
2) Independent production of each table hash value.
3) Independent aggregation of all the table hashes.
4) Independent production of database hash (RDB-signature).
5) Independent encryption of the RDB-signature using asymmetric cryptography.
6) Independent sending of the RDB-signature to client.

Whenever the client receives the RDB-signature after q_i from all CSPs, the client will use the private key to read the data and compare it with its consorts. The outcome of

a verification process is either a commit (if more than two-thirds return an identical result) or a decision to pre-request the RDB-signature. Table 1 summarised the design protocol.

TABLE 1. BC over cloud-RDB methodology protocol.

1.	Client	: Homomorphically Encrypt Data $c = Enc_{Puk}(m)$.
2.	Client → CSP No.1	: Store [encrypted data $(c_1, c_2, c_3, \ldots, c_n)$] in DB based on BC.
	Client → CSP No.2	: Store [encrypted data $(c_1, c_2, c_3, \ldots, c_n)$] in DB based on BC.
	Client → CSP No.3	: Store [encrypted data $(c_1, c_2, c_3, \ldots, c_n)$] in DB based on BC.
	Client → CSP No.4	: Store [encrypted data $(c_1, c_2, c_3, \ldots, c_n)$] in DB based on BC.
3.	Client → CSP No.1	: query$_1$ $[C_r = c_x \diamond c_y]$.
	Client → CSP No.2	: query$_2$ $[C_r = c_x \diamond c_y]$.
	Client → CSP No.3	: query$_3$ $[C_r = c_x \diamond c_y]$.
	Client → CSP No.4	: query$_4$ $[C_r = c_x \diamond c_y]$.
4.	CSP No.1 → Client	: Digital Signature Hash$_1$ [homomorphically encrypted (DB)].
	CSP No.2 → Client	: Digital Signature Hash$_2$ [homomorphically encrypted (DB)].
	CSP No.3 → Client	: Digital Signature Hash$_3$ [homomorphically encrypted (DB)].
	CSP No.4 → Client	: Digital Signature Hash$_4$ [homomorphically encrypted (DB)].
5.	Client	: Decrypt Digital Signature Hash [homomorphically encrypted (DB)]. : $Hash_{ver} = Hash_1 \oplus Hash_2 \oplus Hash_3 \oplus Hash_4$ $:= \begin{cases} if Hash_{ver} = 0;\ true \\ Hash_{ver} = 1;\ otherwise \end{cases}$

B. SYSTEM DATABASE SETUP

After defining the distributed CSP network's general scope, the system workflow definition in a single CSP is needed. Furthermore, it is necessary to clarify the details database structure of the proposed BC over cloud-RDB.

Two different RDBs are proposed, each with its advantages and disadvantages, which have different impacts based on application results. The system network setup is a common denominator between the two systems. In contrast, the difference lies in the database velocity on which the proposal is based. The agile BC-based RDB proposal is based on the fast-growing or operational RDB, i.e. in about 10 min, it computes more than two queries. In comparison, the secure BC-based RDB depends on a slow-formed RDB. The time interval to the next process is at least 10 min on average. In the following Subsections III-B1 and III-B2, each system overview are explained separately.

1) AGILE BC-BASED RDB

The agile BC-based RDB consists of two types of related tables: chained table and hidden table. The chained table is composed of a set of records; each represents a block in the BC. Consequently, the individual record has the previous record's hash and its unique hash value using SHA-256. The chained table is viewed as an ordered back-linked list of records, each record referring to the previous record in a chain. The structure of the chained table is summarised in Table 2.

TABLE 2. Chained table structure in agile BC-based RDB.

Size	Field	Description
VARCHAR (2)	Primary key	One is for alphabet and one is for data height.
VARIABLE ()	Input	Stored data in homomorphic encrypted form.
VARCHAR (256)	Previous hash	Hash value of parent record.
VARCHAR (256)	Record hash	Hash value of the record attributes value.

Each record within the chained table is identified by a unique primary key which consists of two alphanumeric characters. The prefix character is a constant alphabet determined by the client for each data. While the suffix is a variable integer x, that dictates the current height of the data. Also it expresses the type of data manipulation language (DML) operation applied to the data, i.e. if $x = 0$ means "Delete"; otherwise $x \geq 1$ means "Insert" or "Update". The maximum suffix value is represented by parameter x_n that determines the records' growth to be placed within a specific range of $\mathbb{Z}_n$. After the record reaches x_n, the new record will hold a primary key with a recycled suffix value started again from $x = 1$. The suffix value will indicate activating the garbage collection mechanism to avoid overlapping the primary key and losing the last record update, as will be explained later.

The input attribute indicates the change result of each operation performed on the homomorphically encrypted data. For the existing c_1 and c_2 in the database, the input value is c_3, resulting from applying a query $f(c_1, c_2)$, where $f \in \{+, \times, \oplus\}$. This is the main attribute that the research seeks to verify is not affected by the attackers.

Whereas the previous record hash attribute indicates the SHA-256 value of the single recent previous record. Thus, each record contains its own previous record hash within its own record attributes. The sequence of hashes linking each record to its origin creates a chain going back to the first record ever produced. The first record in BC-based RDB protocol contains a previous hash value equal to zero. Lastly, each record in the chained table also has a current record hash attribute. Which stamped string generated by the SHA-256 applied in the entire record attributes' values.

Therefore, the visualisation of the chained table is a set of records chaining vertically to each other. This vertical stack of records chain is served in a First-In-First-Out (FIFO) manner. The length of the records chain represents the database security parameter (λ) determined by the client.

On the other hand, the hidden table is interpreted as a historical log to guide the processing chained table to the current data state. So, each chained table is indexed by

a hidden table to determine the current data height x_n and apply the next DML. Consequently, the hidden table consists of the chained table primary key field stored into two attributes. Also, the variable integer field keeps only the last height x_n the data have reached in the chained table. The structure of the hidden table attributes are summarised in Table 3.

TABLE 3. Hidden table structure.

Size	Field	Description
VARCHAR (1)	Data Id	Alphabet character in the Primary key of chained table
VARCHAR (1)	Data height	Integer number in the Primary key of chained table

2) SECURE BC-BASED RDB

The secure database architecture is the same as the agile database structure, consisting of two types of tables; chained table and hidden table. Nevertheless, the distinctive contribution to the secure BC-based RDB proposal is that the chained table has attributes that carry the record nonce, the current computation target, and the PoW mathematical result value. Table 4 illustrates the chained table building structure in the secure BC-based RDB.

TABLE 4. Chained table structure in secure BC-based RDB.

Size	Field	Description
VARCHAR (32)	Nonce	Variable number CSPs change to get a hash of the record that is below the target.
VARCHAR (2)	Primary key	One is for alphabet and one is for data height.
VARIABLE ()	Input	Stored data in homomorphic encrypted form.
VARCHAR (256)	Record target	Received goal that accompanying the current computation query.
VARCHAR (256)	Previous hash	Hash value of parent record.
VARCHAR (256)	Record hash	Hash value of the record attributes value.
VARCHAR (256)	PoW solution	Proof of work hash result numerically less than the target.

The nonce attribute indicates the variable value that determines the validity of CSP computations. Whereas, target attribute contains a 256-bit value that the client creates and sends along with the computation request to all CSPs. It is an essential indicator of the correctness of the PoW. The PoW solution's size must be equal to or less than the maximum size of the target added to the database. The PoW attribute contains the resultant valid hash value that CSP found after running a PoW function, as we will explain in Subsection IV-B.

IV. PROOF OF CONCEPT AND PROCESSING PROTOCOL

This section details the processing system and describes the running protocol on every proposed RDB. We simulate the both agile BC-based RDB and secure BC-based RDB structure using MySQL Workbench 8.0 CE [63].

Once the homomorphically encrypted data is released on multiple distributed CSPs, the client outsourced operations to CSPs. Each CSP changes the client's data following the query requirements. The verification mechanism begins when the client requests the RDB-signature value based on SHA-256 from all CSPs at a particular time. This is the consensus value that most CSPs should match. In other words, it is a value that checks that the requested queries match the operations that each CSP has implemented on the client's data. In this way, the client himself verifies the consensus of CSPs on the correctness of the applied processes. The client abides by the honesty of consensus when more than half of the CSPs have identical RDB-signature values. If the similarity of the RDB-signature values is less than two-thirds of the number of CSPs, the client must repeat the verification process. The formal specification for the client's self-verification of the consensus of CSPs is as follows.

Let $Consensus := \prod_{i=1}^{h} f_i$ represent a consensus protocol over a set of h hired CSPs, each executing an exclusive set of computations, f_i. Client state $r_{sig} = v, sig$; v is a strictly increasing verification order, and sig is a RDB-signature produced of one verification order. When the majority of CSPs give a same value c_{sig} of received signature values r_{sig} to the client the client commit the result. Otherwise pre-request p_{sig} the signature value (see Algorithm 1).

Algorithm 1 Client Verification Protocol Based on BFT Consensus $:= \prod_{i=1}^{h} f_i^{0,c_{sig}}$

1: **for** $f_i^{v,sig}$ **do**
2: **set** $i = CSP(d)$
3: **set** $signature_i!(r_{sig})$
4: **if** $|sig_v^1| > \frac{2}{3}h$ **then**
5: **Commit** (v, r_{sig})
6: **else if** $sig \neq c_{sig}$ then
7: $p_{sig_i}^{v,sig}$
8: **end if**
9: **end for**

The slightest alteration in a data record will affect the validity of the RDB-signature values consensus. The following subsections will explain how the client can detect even the trivial attack in the data record.

A. AGILE BC-BASED RDB

To mimic the chaining process in the chained table, two functions are created:

- `RecordHash()` to produce the current record hash.
- `PreviousHash()` to call the previous record hash values based on the data primary key variable.

The scripts shown in Fig. 2 and Fig. 3 are the function creation codes to produce both hashes in one record. Fig. 4 imitates the CSP application of the different DML operations in the agile BC-based RDB.

All DML will be treated to insert statements. The deletion request will assign the primary key suffix zero, where the

```
CREATE FUNCTION RecordHash (inId VARCHAR(   ))
RETURNS VARCHAR(   ) DETERMINISTIC
BEGIN

DECLARE inputs VARCHAR( );
DECLARE phash VARCHAR( );
DECLARE Con VARCHAR( );

SELECT
    Input
FROM
    Chained_Tab
WHERE
    Primary_Key = inId INTO inputs;
SELECT
    Previous_Hash
FROM
    Chained_Tab
WHERE
    Primary_Key = inId INTO phash;
SELECT CONCAT(inId, inputs, phash) INTO Con;
RETURN (SELECT SHA2(Con,256));
END;
```

FIGURE 2. Function creation code to produce the current record hash in the agile BC-based RDB.

```
CREATE FUNCTION PreviousHash (inId VARCHAR(   ))
RETURNS VARCHAR(   ) DETERMINISTIC
BEGIN

DECLARE prehash VARCHAR( );

SELECT
    Record_Hash
FROM
    Chained_Tab
WHERE
    Primary_Key = inId - 1 INTO prehash;
RETURN (prehash);
END;
```

FIGURE 3. Function creation code to produce the previous record hash.

```
CREATE TABLE Chained_Tab (
   Primary_Key VARCHAR(),
   Input VARCHAR(),
   Previous_hash VARCHAR(),
   Record_hash VARCHAR(),
   PRIMARY KEY (Primary_Key),
   INDEX (Primary_Key));


INSERT INTO Chained_Tab()
   VALUES (Primary_Key, Computed Input,
   Previous_Hash(Primary_Key),
   Record_Hash(Primary_Key));
```

FIGURE 4. Simulation of CSP application of DML query in the agile BC-based RDB.

added request will initiate a new record-holding a suffix with a value started from one. Moreover, the update statement will increase the maximum height of data by one.

This layout would allow the data to increase exponentially in a short period; unlike BC technologies, massive data growth is inconsistent with the database's versatility.

Therefore, this paper proposes a garbage collection function to clean up the old and deleted records database. Algorithm 2 illustrates the process of garbage collection function. Two conditions must be fulfilled to activate the garbage collection mechanism. Otherwise, the table continues inserting a new record from a different DML. Which they are:

1) Length (l) of chaining the produced records is equal to or greater than database security parameter (λ).
2) Primary key-suffix is reaching the max limit (x_n).

The first condition ensures that the records chain is not weakened or lost. Whereas, the second condition is to recycle and refresh suffix data for every certain prefix. If the requirements are met before the new DML request, the garbage collection process will omit the following:

1) Record(s) with a suffix value equal to zero.
2) Record(s) before the current suffix with an identical prefix.

However, garbage collection will inadvertently disrupt the chain of records, especially if many different data are manipulated. Thus, many orphan records will be randomly dispersed in the table. A re_chaining function is a refreshing approach that always follows the garbage collection function where is a need to update the records chain. Therefore, saving the stability of the records chain and preparing the database for inserting a new record. In more depth, the fundamental purpose of this method is to reorganise the records chain structure, thus preventing it from being lost. re_chaining operation depends on the solution perspective of the BC fork problem. Its rule is as follows "the longest chain of blocks is considered is the active version of the BC" [58]. In compliance, after garbage collection operations, the longest chain will be the leading chain, and all the orphan records followed using FIFO manner. Algorithm 2 explained the re_chaining function that occurs after the garbage collection process has been triggered.

B. SECURE BC-BASED RDB

The fundamental structure for secure BC-based RDB is the agile BC-based RDB. Also, the data computation process follows the specifications of agile system data (see Fig. 5). However, this proposal differs in its reliance on PoW that shielded database records. Each hired CSP has to make PoW mathematical efforts to produce a new record in the database. The CSP must find the hash value below the current target value to add a new record to the database. In this assumption, the client is the one who adjusts the difficulty of the target, therefore, controls the database performance for record in proportion to the database security. Depending on the Bitcoin approach, the client controls the difficulty to resolve the PoW on an average of 10 min to accept adding a newly created record to the database. Based on equations in Subsection II-B the client has to calculate the difficulty value and therefore determine the target before any request from CSPs.

Hence in the secure BC-based RDB, once CSP wants to insert a new record, CSP should find the `PreviousHash()`

Algorithm 2 Formal Specification of CSP Computation Query Protocol

if $lim_{Px_i \to Px_n} f(Px)$ **then**
  if $l \geq \lambda$ **then**
    call GARBAGE_COLLECTION( )
    call RE_CHAINING( )
    else if DML *delete* **set** $x = 0$
      DML *insert* **set** $x = 1$
      DML *update* **set** $x++$
  end if
end if
procedure GARBAGE_COLLECTION( )
  for every $Px_{current} = Px_i$ **do**
    Delete $x = 0$
    Delete $x < x_{current}$
  end for
end procedure
procedure RE_CHAINING( )
  if $(l_i == l_{max})$ **then**
    set $l_i \leftarrow chain\{\mathcal{V}\}$
    if $Q \neq 0$ **then**
      for $(Px_i == Px_0\, Px_i \notin Q,\ Px_i++)$ **do**
        return $Px_i \in Q$
        else skip
      end for
    end if
  end if
end procedure

```
CREATE TABLE Chained_Tab (nonce INT(),
  Primary_Key VARCHAR(),Input VARCHAR(),
  Previous_hash VARCHAR(),
  Record_target VARCHAR(),
  Record_hash VARCHAR(),
  Record_POW VARCHAR(),
  PRIMARY KEY (Primary_Key),
  INDEX (Primary_Key));


INSERT INTO Chained_Tab()
  VALUES (Primary_Key, Computed Input,
  Previous_Hash(Primary_Key),
  Record_target,
  Record_Hash(Primary_Key),
  Record_POW(Primary_Key));
```

FIGURE 5. Simulation of CSP application of DML query in the secure BC-based RDB.

and `RecordHash()` first, then determine the record PoW value that below the record target, by inserting a random nonce in `PoW()`. Fig. 6 represents a very simplified `PoW()` function that is called by each CSP. Thus, If the attacker tampers with one record data, he/she has to recalculate the PoW of all following records. Thus, the workload on the attacker constituted a vast budget and required an avalanche of electricity without financial returns.

```
CREATE FUNCTION POW (inId VARCHAR(), nonce INT())
RETURNS VARCHAR() DETERMINISTIC
BEGIN

  DECLARE R_target VARCHAR();
  DECLARE R_hash VARCHAR();
  DECLARE R_POW VARCHAR();
  DECLARE Max_nonce INT();

SELECT
    Record_target
FROM
    Chained_Tab
WHERE
    PrimaryKey = inId INTO R_target;
SELECT
    Record_hash
FROM
    Chained_Tab
WHERE
    PrimaryKey = inId INTO R_hash;
    SET Max_nonce = variable;
    IF nonce < Max_nonce THEN
      SET R_POW = SHA2(CONCAT(nonce, R_hash),256);
      WHILE R_POW > R_target DO
        SET nonce = nonce + 1
        AND nonce < Max_nonce;
      END WHILE;
    END IF;
    RETURN R_POW;
END;
```

FIGURE 6. PoW function code in the secure BC-based RDB.

V. RESULTS AND DISCUSSION

This section aims to analyse our BC over cloud-RDB proposal from a security point of view to ensure that the study objective is consistent with the findings. We also provide cost, performance and implementation analyses of both proposed systems to present to the client the fundamental differences between the two systems. We exclude from our estimates the costs of hiring CSPs.

A. SECURITY ANALYSIS

Proposal BC over cloud-RDB fulfils the research objective by providing the three primary security features of data processed in the cloud: privacy, confidentiality, and integrity. Privacy and confidentiality are provided using asymmetric HE cryptosystems. It allows the desired queries to be applied to the encrypted data without accessing the private decryption key or exposing confidential information. BC components on an RDB validate the integrity of the processes applied to the data and detect and prevent any internal attacks.

The research assessment of data integrity relies on the trust model presented in [64], which is a study based on CSA service challenges to develop a metric to evaluate the strength of data security in various dimensions of cloud computing by measuring access policy and data integrity strength. The power of data integrity for any system was classified according to the system's availability and inclusion of certain conditions. The first requirement is to encrypt the data using a robust public-key cryptosystem. The second is to use the hash function and take advantage of the fixed length of

the digest. The third is to use both cryptography and hash. Finally, the last requirement is the ability of the system to detect and correct. The proposed system offers the maximum data integrity verification against computational tampering attacks by achieving all trust system requirements.

Therefore, the proposal BC-based RDB achieved the objectives of the research as follows:

- *Data Privacy:* The proposed system achieves privacy by taking advantage of the HE cryptosystem characteristics. Keeping the private key with the client prevents any unauthorised use or collection attacks.
- *Data Confidentiality:* The proposed system is confidential due to the use of asymmetric HE cryptosystems. In particular, the private key of the encrypted data owns only licensees. Therefore, unauthorised are not allowed to disclose the content of the data, including the CSP.
- *Data Integrity:* The proposal prevents internal abuse and has the advantage of identifying the record affected by the internal attacker. If an internal threat occurs in one of the records for both proposed databases, it will create a corrupt RDB-signature that the consensus process will expose. In the agile BC-based RDB, If an incorrect operation is applied to the data record, the attacker must regenerate a current hash value. Thus, recalculating all successive records' hashes. On the other side, the client can easily detect the failure through a consensus mechanism. At the same time, the secure BC-based RDB system outperforms its counterpart by making any attack hard and nearly impossible by simulating the PoW function. In particular, If internal threats occur in a record, the attacker has to recalculate the hashes of the following records and the PoW value for each record. Thus, the PoW attack makes the system arduous by its mechanism, which takes one day to recalculate 144 records and wasting more than 56kW [1] [65].

  Compared to previous works, BC over cloud-RDB proposed systems have proven their superiority by introducing a self-verification model for the client. Thus, the proposed systems support the client to keep track of his queries and prevent any random change. Also, if internal breaches occur, they allow detecting damaged data records accurately.

B. IMPLEMENTATION ANALYSIS

In practice, the system application to improve RDB security involves adding extra attributes in the RDB tables. The implementation of the proposed system is straightforward and can be easily applied in any RDB system. Because it does not need database restructuring nor additional system setup or configuration work. Fig. 7 depicts the agile BC-based RDB system and Fig. 8 illustrates the secured BC-based RDB system.

The proposed system is also flexible, as it can be applied to tables that contain sensitive information in RDB and not to all tables. Also, the proposal network can be expanded to make more than four CSPs involved.

C. COST ANALYSIS

This paper studies the overhead cost of the client according to the additional energy consumption factor. The cost analysis omits other factors, such as the cost of hiring CSPs. The agile system relies on hashing, a relatively efficient operation that causes a slight financial burden. According to [66], the energy expended of SHA-256 is equal to 0.4μJ for each input byte using a CPU with a 25MHz clock rate. While the consumed energy that clouds computing engines spend per hashing process is much less than in ordinary computers [67]. The cloud computing capability of various CSPs is characterised by very high performance and memory speeds that take advantage of the most compute-intensive workloads. For instance, Emulating a single SHA-256 function in the c2-standard instance on Google Cloud Compute Engine ™ [68] with a single-core max turbo frequency equal to 3.9GHz will consume $1.73E^{-11}$kW/h. Thus, a new DML operation in the agile BC-based RDB will consume only $3.46E^{-11}$kW/h using the same instance configuration. Accordingly, a billion DML operations will consume between 0.1J to 1J of energy with state-of-the-art technology [69]. Therefore, the overall cost of an agile system is considered inexpensive for the client.

In comparison, the secure BC-based RDB wastes relatively high energy in addition to hashing to conduct the PoW operations. The difficulty is the primary measurement that determines the overhead cost to the client, solely responsible for setting it up. A significant difficulty means that a new record would take more computational power to create, making the data more secure against attacks [70]. While reducing the difficulty contributes to reducing the time interval between records. On the contrary, it contributes to decreasing the PoW wasting energy and thus faster insertion of new records. Table 5 shows three interval options resulting from manipulating the difficulty value.

It turns out that matching the PoW process to the Bitcoin mining will consume about 395.4kW in 10 minutes for a single record [2] [65]. While electricity prices change globally, but the average price of electricity in the world for the business in the second quarter of 2021 is US $1.22 [3] [71]. Therefore, the added cost to the client for PoW per record is US $48.5. Consequently, the system's cost of inserting records every 10 minutes consistently in one hour is US $291. In succession, the overhead cost increases to reach more than US $6, 900 during one day. That is because PoW drained approximately 56.9kW in performing 144 records only. Nevertheless, when the difficulty value is set to produce records every 8 minutes, the consumed energy is reduced by 11kW per day. Also, it decreases by approximately 23kW in processing PoW every 6 minutes per day. Thus, the overhead cost reduces

[1] As of 17th August 2021.

[2] As of 17th August 2021.

[3] As of 15th August 2021.

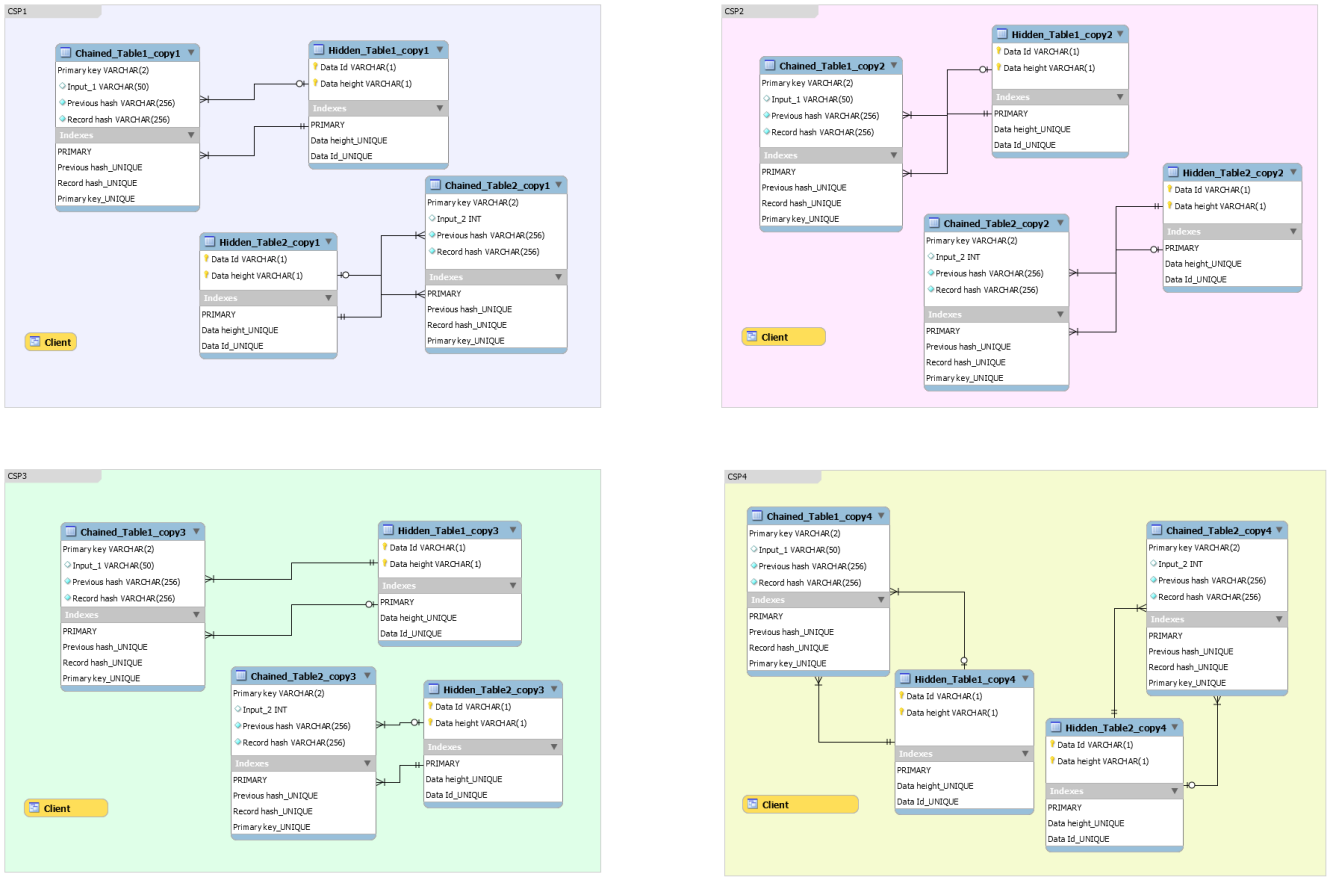

FIGURE 7. Agile BC-based RDB system.

TABLE 5. Overhead costs & system performance for secure BC-based RDB.

Measuring parameters	Proof-of-Work every		
	10 min	8 min	6 min
Number of record (Record)			
1 hour	6	7	10
One day	144	180	240
One week	1008	1260	1680
One month	4380	5475	7300
Power consumption (kilowatts)			
1 hour	2.3724kW	1.8979kW	1.4234kW
One day	56.938kW	45.551kW	34.163kW
One week	398.56kW	318.85kW	239.14kW
One month	1731.9kW	1385.5kW	1039.1kW
Overhead cost (US $)			
1 hour	$291	$233	$175
One day	$6,985	$5,589	$4,192
One week	$48,898	$39,123	$29,342
One month	$212,474	$169,998	$127,499

by US $2,793 in producing records every 6 minutes per day than the actual PoW mechanism. And the performance of record production is increased by 96 records. Furthermore, in one month, the power consumption reaches more than 1700kW in generating one record every 10 minutes. This leads to the client incurring substantial financial losses equivalent to US $200,000. While raising the difficulty target to a record every 6 minutes per month cuts the overhead cost by roughly half. In addition, it increases the proposed system performance to produce records equivalent to 51 days of processing PoW in 10 minutes.

D. PERFORMANCE ANALYSIS AND FUTURE WORK

Database latency and throughput vary according to the nature and requirements of the data owner and the business type [72]. The high-throughput database in which a massive number of new records are generated in a fast manner. Any problem in the performance of the database may affect the business capital. Therefore, to not hinder the efficiency of the database, a lean and flexible security proposal is needed. The agile BC-based RDB is suggested for this type of fast productivity database. However, it will slightly slow down the writing process because CSP processes two hash functions for each new record. As for the secure BC-based RDB, it is utilising PoW function, which affects the system performance as shown in Table 5. The lightest proposed system produces 10 records per hour, i.e. a total of 240 records during the day. Thus, we recommend this proposal for the database with low throughput productivity.

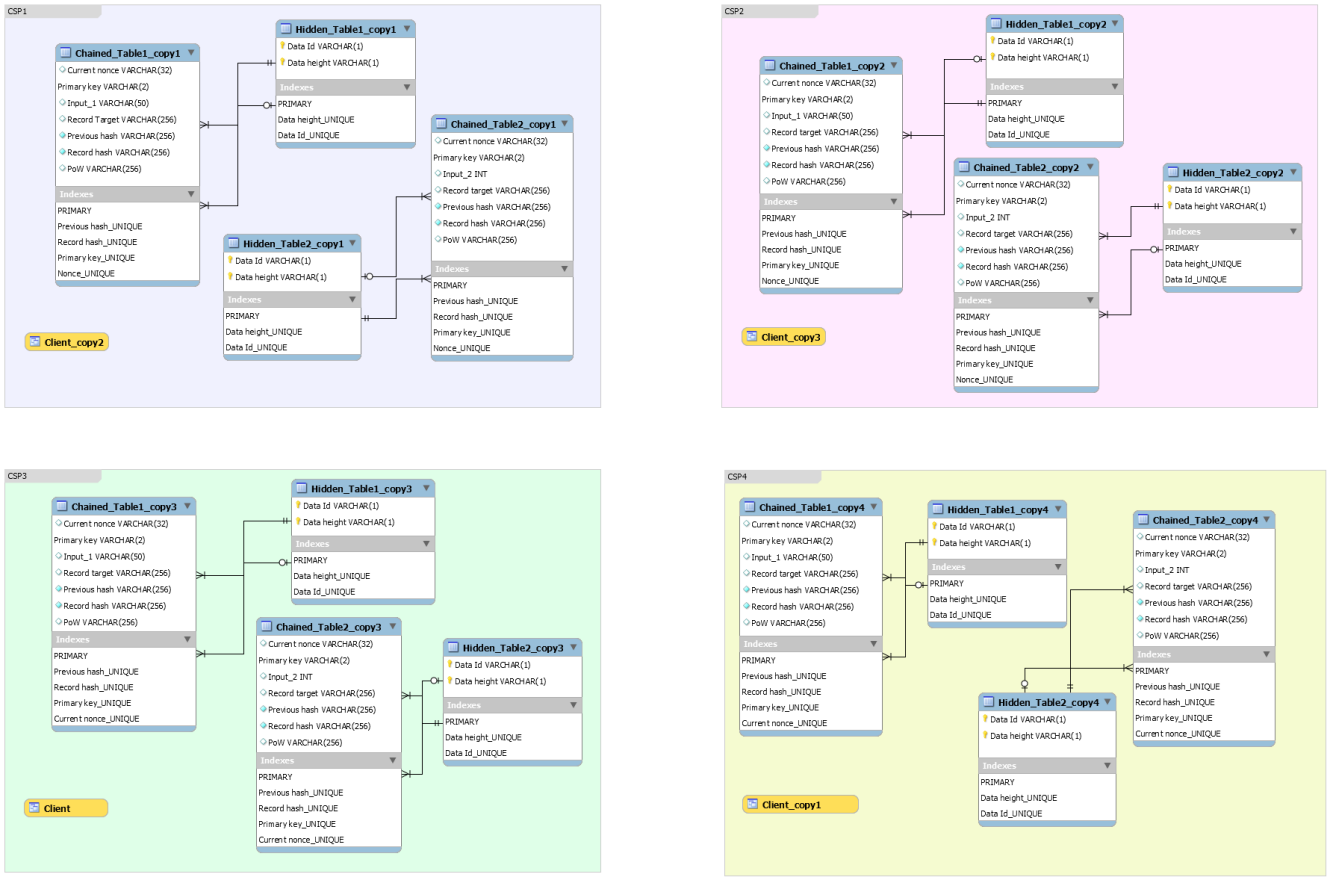

FIGURE 8. Secured BC-based RDB system.

To simulate a working environment with a higher level of security, we recommend merging the two proposed structures into a database. Therefore, the agile structure is applied to tables containing data in permanent processing and is not highly important. On the other hand, the security system is applied to tables containing critical data that do not need continuous processing.

In general, in achieving the natural outsourcing computations, the client uses one CSP to store their data and extract their computations' request results. However, in this proposal, the overhead cost on the client will increase fourfold. This is because the client will use at most four CSP to carry out the same work required from a single CSP. Moreover, when applying the second system that includes the PoW mechanism, a client will be billed extra to achieve the level of data security that the system proposes. In future work, we aim to use IOTA Tangle to improve the proposal's performance to be compatible with the Internet of Things (IoT) environment.

VI. CONCLUSION

Cloud databases should have a reliable authority control security apparatus to follow data modifications. Specifically, cloud databases are problematic since they can be manipulated even without the acknowledgement of the data owner. As such, blockchain technology should be used to rejig the building structure of the cloud database and arrive in a decentralised manner between the cloud service provider and the client. This paper introduced a scheme for client self-verification which is based on one of two optimised blockchain-based relational database systems; agile BC-based RDB and secure BC-based RDB. The proposed systems developed the cloud relational database structure by adding extra attributes to chain records based on SHA-256. Also, both simulate the decentralisation by distributing the developed database to at least four cloud service providers. Once a query is requested from the client, the hired cloud service providers have to update their database and link the new record(s) to the previous chain of records. When the client requests, they produce a RDB-signature to send to verify their consensus on the same result. The analyses showed that the agile BC-based RDB system is inexpensive and wastes slightly additional energy, with a maximum value of 1 joule. Thus this system has proven its value in high-throughput databases. In comparison, the secure BC-based RDB is superior to the security offered by its counterpart by relying on the proof of work mechanism. However, it is a financially stressful and arduous system. Generally, the adoption of the proposed systems is easy to implement as it does not require

any restructuring of the database or cloud computing system. The proposal systems could be merged or tweaked to satisfy the clients' needs.

REFERENCES

[1] CSA. (2017). *Security Guidance for Critical Areas of Focus in Cloud Computing v4.0*. [Online]. Available: https://cloudsecurityalliance.org/artifacts/security-guidance-v4/

[2] M. Vaishnnave, K. S. Devi, and P. Srinivasan, "A survey on cloud computing and hybrid cloud," *Int. J. Appl. Eng. Res.*, vol. 14, no. 2, pp. 429–434, 2019.

[3] M. Stieninger and M. A. Erskine, "Factors influencing the organisational adoption of cloud computing: A survey among cloud workers," *Int. J. Inf. Syst. Project Manage.*, vol. 6, no. 1, pp. 5–23, Jan. 2018, doi: 10.12821/ijispm060101.

[4] I. Odun-Ayo, O. Ajayi, and A. Falade, "Cloud computing and quality of service: Issues and developments," in *Proc. Int. MultiConf. Eng. Comput. Sci. (IMECS)*, vol. 1, Hong Kong, Mar. 2018, pp. 179–184.

[5] E. S. Kumar, S. Kesavan, and R. C. A. Naidu, "Comprehensive analysis of cloud based databases," *IOP Conf. Mater. Sci. Eng.*, vol. 1131, no. 1, Apr. 2021, Art. no. 012021.

[6] Y. Roh, G. Heo, and S. E. Whang, "A survey on data collection for machine learning: A big data–AI integration perspective," *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 4, pp. 1328–1347, Apr. 2021.

[7] Y. Mansouri, A. N. Toosi, and R. Buyya, "Data storage management in cloud environments: Taxonomy, survey, and future directions," *ACM Comput. Surv.*, vol. 50, no. 6, pp. 1–51, 2017.

[8] Ö. Kasim, "An ensemble classification-based approach to detect attack level of SQL injections," *J. Inf. Secur. Appl.*, vol. 59, Jun. 2021, Art. no. 102852.

[9] A. Memaripour, A. Badam, A. Phanishayee, Y. Zhou, R. Alagappan, K. Strauss, and S. Swanson, "Atomic in-place updates for non-volatile main memories with Kamino-Tx," in *Proc. 12th Eur. Conf. Comput. Syst.*, Apr. 2017, pp. 499–512, doi: 10.1145/3064176.3064215.

[10] Towards Data Science. (2021). *5 Best Cloud Databases to Use in 2021*. Accessed: Aug. 20, 2021. [Online]. Available: https://towardsdatascience.com/5-best-public-cloud-database-to-use-in-2021-5fca5780f4ef

[11] CSA. (2020). *Top Threats to Cloud Computing: Egregious Eleven*. Accessed: Dec. 2, 2020. [Online]. Available:https://cloudsecurityalliance.org/artifacts/top-threats-to-cloud-computing-egregious-eleven/

[12] R. Kumar and R. Goyal, "On cloud security requirements, threats, vulnerabilities and countermeasures: A survey," *Comput. Sci. Rev.*, vol. 33, pp. 1–48, Aug. 2019.

[13] S. Romanosky and A. Acquisti, "Privacy costs and personal data protection: Economic and legal perspectives," *Berkeley Tech. Law J.*, vol. 24, no. 3, p. 1061, 2009.

[14] S. Cates. (2012). *Data Security in the Cloud: Protecting Business-Critical Information in Public, Private, and Hybrid Cloud Environments*. Custom Solutions Group, Vormetric Data Security Simplified. [Online]. Available: https://www.vormetric.com/data-security-solutions/overview/index.html

[15] J. Goldstein. (2020). *What are Insider Threats and How Can You Mitigate Them?*. Accessed: May 11, 2021. [Online]. Available: https://securityintelligence.com/posts/what-are-insider-threats-and-how-can-you-mitigate-them/

[16] N. Anciaux, M. Benzine, L. Bouganim, P. Pucheral, and D. Shasha, "GhostDB: Querying visible and hidden data without leaks," in *Proc. ACM SIGMOD Int. Conf. Manage. Data (SIGMOD)*, 2007, pp. 677–688,

[17] S. Pearson and A. Benameur, "Privacy, security and trust issues arising from cloud computing," in *Proc. IEEE 2nd Int. Conf. Cloud Comput. Technol. Sci.*, Nov. 2010, pp. 693–702.

[18] R. Ichardson. *15th Annual 2010/2011 Computer Crime and Security Survey*. Accessed: Feb. 10, 2021. [Online]. Available: https://cours.etsmtl.ca/gti619/documents/divers/CSIsurvey2010.pdf

[19] Panda Security. *Insider Threat Cost Report*. Accessed: Apr. 7, 2021. [Online]. Available: https://www.pandasecurity.com/en/mediacenter/security/cost-insider-threat-report/

[20] Q. Yaseen and B. Panda, "Organizing access privileges: Maximizing the availability and mitigating the threat of insiders' knowledgebase," in *Proc. 4th Int. Conf. Netw. Syst. Secur.*, Sep. 2010, pp. 312–317.

[21] C. Farkas, T. S. Toland, and C. M. Eastman, "The inference problem and updates in relational databases," in *Database and Application Security*. Boston, MA, USA: Springer, 2001, pp. 181–194.

[22] Q. Yaseen, Q. Althebyan, B. Panda, and Y. Jararweh, "Mitigating insider threat in cloud relational databases," *Secur. Commun. Netw.*, vol. 9, no. 10, pp. 1132–1145, Jul. 2016.

[23] Q. Althebyan, R. Mohawesh, Q. Yaseen, and Y. Jararweh, "Mitigating insider threats in a cloud using a knowledgebase approach while maintaining data availability," in *Proc. 10th Int. Conf. Internet Technol. Secured Trans. (ICITST)*, Dec. 2015, pp. 226–231.

[24] Q. Althebyan, "A mobile edge mitigation model for insider threats: A knowledgebase approach," in *Proc. Int. Arab Conf. Inf. Technol. (ACIT)*, Dec. 2019, pp. 188–192.

[25] Q. Althebyan, "Mitigating insider threats on the edge: A knowledgebase approach," *Int. Arab J. Inf. Technol.*, vol. 17, no. 4A, pp. 621–628, Jul. 2020.

[26] Q. Yaseen, Y. Jararweh, B. Panda, and Q. Althebyan, "An insider threat aware access control for cloud relational databases," *Cluster Comput.*, vol. 20, no. 3, pp. 2669–2685, Sep. 2017.

[27] H. A. Kholidy, "Detecting impersonation attacks in cloud computing environments using a centric user profiling approach," *Future Gener. Comput. Syst.*, vol. 117, pp. 299–320, Apr. 2021.

[28] J. Stephen, S. Savvides, R. Seidel, and P. Eugster, "Practical confidentiality preserving big data analysis," in *Proc. 6th USENIX Workshop Hot Topics Cloud Comput.*, 2014, pp. 1–6.

[29] A. Alsirhani, P. Bodorik, and S. Sampalli, "Improving database security in cloud computing by fragmentation of data," in *Proc. Int. Conf. Comput. Appl. (ICCA)*, Sep. 2017, pp. 43–49.

[30] W. Zheng, A. Dave, J. G. Beekman, R. A. Popa, J. E. Gonzalez, and I. Stoica, "Opaque: An oblivious and encrypted distributed analytics platform," in *Proc. 14th USENIX Symp. Netw. Syst. Design Implement. (NSDI)*, 2017, pp. 283–298,

[31] G. Liu, G. Yang, H. Wang, Y. Xiang, and H. Dai, "A novel secure scheme for supporting complex SQL queries over encrypted databases in cloud computing," *Secur. Commun. Netw.*, vol. 2018, pp. 1–15, Jul. 2018.

[32] T. T. Thwin and S. Vasupongayya, "Blockchain based secret-data sharing model for personal health record system," in *Proc. 5th Int. Conf. Adv. Inform., Concept Theory Appl. (ICAICTA)*, Aug. 2018, pp. 196–201, doi: 10.1109/ICAICTA.2018.8541296.

[33] S. Cao, X. Zhang, and R. Xu, "Toward secure storage in cloud-based eHealth systems: A blockchain-assisted approach," *IEEE Netw.*, vol. 34, no. 2, pp. 64–70, Mar./Apr. 2020, doi: 10.1109/MNET.001.1900173.

[34] D. E. Kouicem, Y. Imine, A. Bouabdallah, and H. Lakhlef, "A decentralized blockchain-based trust management protocol for the Internet of Things," *IEEE Trans. Dependable Secure Comput.*, early access, Jun. 18, 2020, doi: 10.1109/TDSC.2020.3003232.

[35] J. Feng, L. T. Yang, R. Zhang, W. Qiang, and J. Chen, "Privacy preserving high-order bi-Lanczos in cloud-fog computing for industrial applications," *IEEE Trans. Ind. Informat.*, early access, May 28, 2020, doi: 10.1109/TII.2020.2998086.

[36] J. Feng, L. T. Yang, Y. Zhu, N. J. Gati, and Y. Mo, "Blockchain-enabled tensor-based conditional deep convolutional GAN for Cyber-physical-Social systems," *ACM Trans. Internet Technol.*, vol. 21, no. 2, pp. 1–17, Jun. 2021.

[37] (2018). (Hyperledger Fabric). Accessed: Aug. 10, 2021. [Online]. Available: https://www.hyperledger.org/projects/fabric

[38] Bigchaindb. (2018). *Meet BigchainDB*. Accessed: Aug. 10, 2021. [Online]. Available: https://www.bigchaindb.com/

[39] MongoDB. (2021). *MongoDB: The Application Data Platform*. Accessed: Aug. 10, 2021. [Online]. Available: https://www.mongodb.com/

[40] Ledgerdb. *LedgerDB: A Ledger Database That Provides Powerful Data Audit Capabilities*. Accessed: Aug. 11, 2021. [Online]. Available: https://www.alibabacloud.com/tc/product/

[41] S. Helmer, M. Roggia, N. E. Ioini, and C. Pahl, "EthernityDB–integrating database functionality into a blockchain," in *Proc. Eur. Conf. Adv. Databases Inf. Syst. (ADBIS)*, 2018, pp. 37–44.

[42] J. Lian, S. Wang, and Y. Xie, "TDRB: An efficient tamper-proof detection middleware for relational database based on blockchain technology," *IEEE Access*, vol. 9, pp. 66707–66722, 2021, doi: 10.1109/ACCESS.2021.3076235.

[43] R. Awadallah, A. Samsudin, J. S. Teh, and M. Almazrooie, "An integrated architecture for maintaining security in cloud computing based on blockchain," *IEEE Access*, vol. 9, pp. 69513–69526, 2021, doi: 10.1109/ACCESS.2021.3077123.

[44] S. Tan and A. Samsudin, "A survey of homomorphic encryption for outsourced big data computation," *KSII Trans. Internet Inf. Syst.*, vol. 10, no. 8, pp. 3826–3851, 2016, doi: 10.3837/tiis.2016.08.022.
[45] O. Hanash, "Homomorphic encryption of text documents," Ph.D. dissertation, Dept. Comput. Sci., Middle East Univ., Amman, Jordan, 2020.
[46] D. Boneh, E. Goh, and K. Nissim, "Evaluating 2-DNF formulas on ciphertexts," in *Theory of Cryptography*. Berlin, Germany: Springer, 2005, pp. 325–341, doi: 10.1007/978-3-540-30576-718.
[47] C. Gentry, "A fully homomorphic encryption scheme," Ph.D. dissertation, Dept. Comput. Sci., Stanford Univ., Stanford, CA, USA, 2009.
[48] D. Das, "Secure cloud computing algorithm using homomorphic encryption and multi-party computation," in *Proc. Int. Conf. Inf. Netw. (ICOIN)*, Jan. 2018, pp. 391–396, doi: 10.1109/ICOIN.2018.8343147.
[49] R. Shrestha and S. Kim, "Integration of IoT with blockchain and homomorphic encryption: Challenging issues and opportunities," in *Advances in Computers*, vol. 115. Amsterdam, The Netherlands: Elsevier, 2019, pp. 293–331.
[50] A. A. Badawi, L. Hoang, C. F. Mun, K. Laine, and K. M. M. Aung, "PrivFT: Private and fast text classification with homomorphic encryption," *IEEE Access*, vol. 8, pp. 226544–226556, 2020, doi: 10.1109/ACCESS.2020.3045465.
[51] N. Döttling, J. Müller-Quade, and A. C. Nascimento, "IND-CCA secure cryptography based on a variant of the LPN problem," in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Secur.* Berlin, Germany: Springer, 2012, pp. 485–503.
[52] R. Awadallah and A. Samsudin, "Homomorphic encryption for cloud computing and its challenges," in *Proc. IEEE 7th Int. Conf. Eng. Technol. Appl. Sci. (ICETAS)*, Dec. 2020, pp. 1–6.
[53] D. Yaga, P. Mell, N. Roby, and K. Scarfone, "Blockchain technology overview," 2019, *arXiv:1906.11078*. [Online]. Available: http://arxiv.org/abs/1906.11078
[54] X. Burri, E. Casey, T. Bollé, and D.-O. Jaquet-Chiffelle, "Chronological independently verifiable electronic chain of custody ledger using blockchain technology," *Forensic Sci. Int., Digit. Invest.*, vol. 33, Jun. 2020, Art. no. 300976.
[55] M. J. Dworkin. (Aug. 2015). *SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions*. Federal Inf. Process. Stds.(NIST FIPS). [Online]. Available: https://csrc.nist.gov/publications/detail/fips/202/final
[56] N. Kishore and B. Kapoor, "Attacks on and advances in secure hash algorithms," *IAENG Int. J. Comput. Sci*, vol. 43, no. 3, pp. 326–335, 2016.
[57] M. Marx, E. Zimmer, T. Mueller, M. Blochberger, and H. Federrath, "Hashing of personally identifiable information is not sufficient," SICHERHEIT, Gesellschaft für Informatik, Bonn, Germany, Tech. Rep., 2018, pp. 55–68, doi: 10.18420/sicherheit2018_04.
[58] A. M. Antonopoulos, *Mastering Bitcoin: Unlocking Digital Cryptocurrencies*, 1st ed. Sebastopol, CA, USA: O'Reilly Media, 2014.
[59] A. Magnani, L. Calderoni, and P. Palmieri, "Feather forking as a positive force: Incentivising green energy production in a blockchain-based smart grid," in *Proc. 1st Workshop Cryptocurrencies Blockchains Distrib. Syst.*, Jun. 2018, pp. 99–104.
[60] M. Castro and B. Liskov, "Practical byzantine fault tolerance and proactive recovery," *ACM Trans. Comput. Syst.*, vol. 20, no. 4, pp. 398–461, Nov. 2002.
[61] E. Buchman, "Tendermint: Byzantine fault tolerance in the age of blockchains," Ph.D. dissertation, Dept. Eng. Syst. Comput., Univ. Guelph, Guelph, ON, Canada, 2016.
[62] S. Sayeed and H. Marco-Gisbert, "Assessing blockchain consensus and security mechanisms against the 51% attack," *Appl. Sci.*, vol. 9, no. 9, p. 1788, Apr. 2019.
[63] *MySQL Workbench 8.0*, Workbench, MySQL, USA, 2019.
[64] R. Shaikh and M. Sasikumar, "Trust model for measuring security strength of cloud computing service," *Proc. Comput. Sci.*, vol. 45, pp. 380–389, Jan. 2015.
[65] DIGICONOMIST. *Bitcoin Energy Consumption Index*. Accessed: Aug. 17, 2021. [Online]. Available: https://digiconomist.net/bitcoin-energy-consumption
[66] S. Sinha and R. M. Gaudar, "Energy consumption for cryptographic algorithms with different clocks on smart cards in mobile devices," *Int. J. Comput. Appl.*, vol. 66, no. 19, pp. 1–4, 2013.
[67] R. Martino and A. Cilardo, "A flexible framework for exploring, evaluating, and comparing SHA-2 designs," *IEEE Access*, vol. 7, pp. 72443–72456, 2019, doi: 10.1109/ACCESS.2019.2920089.
[68] Google. (2019). *Google Cloud Compute Products*. Accessed: Aug. 13, 2021. [Online]. Available: https://cloud.google.com/compute/docs/disks/?authuser=3&hl=ru#localssds
[69] R. Martino and A. Cilardo, "SHA-2 acceleration meeting the needs of emerging applications: A comparative survey," *IEEE Access*, vol. 8, pp. 28415–28436, 2020, doi: 10.1109/ACCESS.2020.2972265.
[70] (2020). *Network Difficulty*. [Online]. Available: https://www.blockchain.com/charts/difficulty
[71] (2021). *Global Petrol Prices*. [Online]. Available: https://www.globalpetrolprices.com/data_electricity_download.php
[72] D. J. Teece, "Explicating dynamic capabilities: The nature and microfoundations of (sustainable) enterprise performance," *Strategic Manage. J.*, vol. 28, no. 13, pp. 1319–1350, Dec. 2007.

RUBA AWADALLAH received the B.Sc. degree (Hons.) in network and communication engineering from Al Ain University, United Arab Emirates, in 2012, and the M.Sc. degree in engineering management from Abu Dhabi University, United Arab Emirates, in 2014. She is currently pursuing the Ph.D. degree with the School of Computer Sciences, Universiti Sains Malaysia (USM). She is also a Graduate Research Assistant with USM. Her research interests include data security, cryptography, blockchain, and parallel computing.

AZMAN SAMSUDIN received the B.Sc. degree in computer science from the University of Rochester, Rochester, NY, USA, in 1989, and the M.Sc. and Ph.D. degrees in computer science from the University of Denver, Denver, CO, USA, in 1993 and 1998, respectively. He is currently a Professor with the School of Computer Science, Universiti Sains Malaysia (USM). He has published more than 100 articles over a series of books, professional journals, and conferences. His research interests include cryptography, switching networks, and parallel computing.