Computers, Materials & Continua

Tech Science Press

DOI: 10.32604/cmc.2023.042386

ARTICLE

An Innovative Approach Using TKN-Cryptology for Identifying the Replay Assault

Syeda Wajiha Zahra[1], Muhammad Nadeem[2], Ali Arshad[3,*], Saman Riaz[3], Muhammad Abu Bakr[4], Ashit Kumar Dutta[5], Zaid Alzaid[6], Badr Almutairi[7] and Sultan Almotairi[8]

[1]Department of Computer Science, Alhamd Islamic University, Islamabad, 44000, Pakistan

[2]Department of Computer Science and Technology, University of Science and Technology Beijing, Beijing, 100083, China

[3]Department of Computer Science, National University of Technology, Islamabad, 44000, Pakistan

[4]Department of Electrical Engineering, National University of Technology, Islamabad, 44000, Pakistan

[5]Department of Computer Science and Information Systems, College of Applied Sciences, AlMaarefa University, Riyadh, 13713, Saudi Arabia

[6]Department of Computer Science, Faculty of Computer and Information Systems, Islamic University of Madinah, Medinah, 42351, Saudi Arabia

[7]Department of Information Technology, College of Computer Sciences and Information Technology College, Majmaah University, Al-Majmaah, 11952, Saudi Arabia

[8]Department of Natural and Applied Sciences, Faculty of Community College, Majmaah University, Majmaah, 11952, Saudi Arabia

*Corresponding Author: Ali Arshad. Email: alli.arshad@gmail.com

Received: 28 May 2023 Accepted: 08 November 2023 Published: 30 January 2024

ABSTRACT

Various organizations store data online rather than on physical servers. As the number of user's data stored in cloud servers increases, the attack rate to access data from cloud servers also increases. Different researchers worked on different algorithms to protect cloud data from replay attacks. None of the papers used a technique that simultaneously detects a full-message and partial-message replay attack. This study presents the development of a TKN (Text, Key and Name) cryptographic algorithm aimed at protecting data from replay attacks. The program employs distinct ways to encrypt plain text [P], a user-defined Key [K], and a Secret Code [N]. The novelty of the TKN cryptographic algorithm is that the bit value of each text is linked to another value with the help of the proposed algorithm, and the length of the cipher text obtained is twice the length of the original text. In the scenario that an attacker executes a replay attack on the cloud server, engages in cryptanalysis, or manipulates any data, it will result in automated modification of all associated values inside the backend. This mechanism has the benefit of enhancing the detectability of replay attacks. Nevertheless, the attacker cannot access data not included in any of the papers, regardless of how effective the attack strategy is. At the end of paper, the proposed algorithm's novelty will be compared with different algorithms, and it will be discussed how far the proposed algorithm is better than all other algorithms.

KEYWORDS

Replay attack; malware; message attack; file encryption; cryptology; data security

1 Introduction

Cloud computing is becoming a necessity for the world [1]. The robust physical or virtual infrastructure, accessible across several devices and servers, enables users to retrieve data via internet means rather than relying on an internal hard disk. Data in cloud computing is dispersed over several servers located in various places. The distributed system notion is connected to cloud computing [2]. The use of the cloud is growing as technology advances. On average, up to 81% of businesses have migrated their operations to the cloud. More than 1200 services are now offered through cloud computing, which has expanded twice since 2013 [3]. But if businesses adopt cloud computing more quickly, the rate of attacks also rises. A virtualized data storage technology known as cloud computing allows data to be stored on several servers and accessed from any location.

Many organizations turn to online storage instead of physically storing their data [4]. Data security is a significant concern whenever information is physically kept, regardless of whether the security issue results from legitimate or anonymous users [5]. The most important benefit of keeping data online for an organization is protecting it from data crashes and malware attacks. Cloud architecture consists of two components: one is hardware, while the other is software. Cloud hardware relies on various physical devices, including standard networking equipment, such as routers, switches, load balancers, and firewalls [6]. At the same time, the functionality of all this hardware is performed and controlled with the help of cloud software. Different service providers provide different type of cloud services to end users. Different services includes data storage sizes, infrastructure and databases to store the data [7]. A cloud network is created by utilizing all these services, as shown in Fig. 1.

Figure 1: Structure of cloud network

Before cloud computing, two emerging technologies were introduced to store data, among which the first was client-server computing and the second was distributed computing systems [8]. Client-server computing was a centralized data storage system in which all data was stored on a single server, and services were provided to end users from the server [9]. After client-server computing, various servers were connected to keep data further, and a distributed system was introduced. A practical concept was introduced based on a distributed system called cloud computing. Cloud computing provides many services to users, including Data Backup, Security, Mobility, Unlimited Data Storage, and Low Maintenance [10].

1.1 Various Attacks and Prevention Techniques

With the growing prevalence of cloud computing, several firms are emphasizing data storage in online environments. Similarly, there has been a rise in the frequency of cyber assaults against cloud computing systems. To gain access to the cloud servers, the attacker either applies keylogging techniques to the cloud hardware or uses hacking mechanisms to the cloud software [11]. With the help of these techniques, attackers get easy access to the cloud servers. It is challenging for an attacker to perform a physical attack because data manipulation may be impossible unless any physical cloud component is available [12]. Attacks in cloud computing are always software-based, for which the attacker develops various algorithms. In cloud computing, the attacker carries out two types of attacks. When attempting to gain unauthorized access to a cloud server, attackers often use external attack methods, including but not limited to brute force and pattern-matching assaults. Various techniques are used to identify external threats, including conventional Intrusion Detection and Prevention Systems (IDPS) and firewalls. Using these methodologies makes it possible to avert assaults on cloud infrastructure. However, it is essential to note that safeguarding cloud data remains a challenging task. Insider attacks are often seen on cloud systems, including Distributed Denial of Service (DDoS), Ransomware, Rootkit, Advertising, and Replay assaults [13]. With the help of these attacks, users are tricked, and their data is havoc by accessing their accounts.

1.2 Cryptography

Implementing a cryptographic technique is regarded as the most effective approach for ensuring the safe storage of data. Cryptography is a very efficient method for safeguarding data by transforming it into an encoded structure resistant to comprehension and decryption [14]. Numerous techniques have been devised to protect data from potential attackers. When a malicious actor attempts to target the cloud server via a malicious mechanism, many methods may be used to identify such assaults. However, it is essential to note that complete data protection from attackers remains a challenge [15]. Cryptography is often regarded as a very effective method for safeguarding data against unauthorized access and mitigating the risk of abuse by malicious actors.

Cryptography is a better method of encapsulating the original data using various techniques and algorithms [16]. Within the field of cryptography, the process of transforming data involves converting it into a format that has the characteristic of being both resistant to comprehension and challenging to decipher [17]. The same set-off technique is implemented to bring the data back to its original state, which was used at the time of encryption. A complete cryptographic system is shown in Fig. 2. Cryptography uses two different techniques to convert data into readable and non-readable formats. The first technique is encryption, while the other technique is decryption. In encryption, the original data is converted into a form that cannot be easily cracked using various algorithms. It is necessary to use the same decryption methods used during the encryption process to transform non-readable data into readable data. The process of converting data from an unreadable format to a readable one is sometimes referred to as decryption. The use of identical approaches for both data encryption and decryption is vital.

1.3 Replay Attack

A replay attack is a data-snatching technique in which the attacker captures an authentic message from the network and broadcasts the news again, as shown in Fig. 3. Whenever an attacker intercepts a message and transmits it over the web, they can insert malicious code into the message. It may have a front end similar to a standard transmission, but a back end may be malicious. A replay attack on

a network can be made in two ways. The first is a partial message replay attack, while the other is a complete message replay attack. Detecting a partial message replay attack is more complicated than a whole message replay attack.

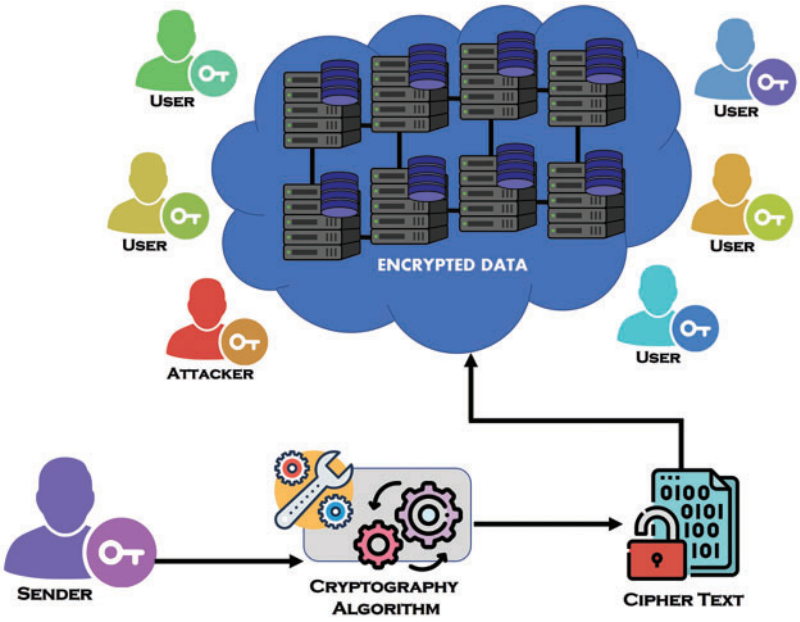

Figure 2: Cryptographic system

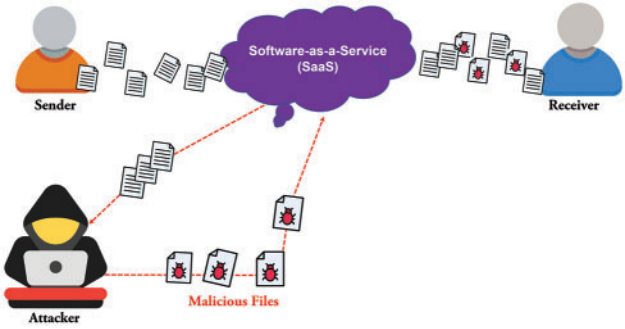

Figure 3: Replay attack mechanism

1.4 Problem Formulation

Researchers have developed numerous tactics and procedures to protect data against replay assaults. Several researchers looked at many pieces of literature to identify replay attacks and selected the most effective strategies from them. None of the articles provided any methods for preventing replay attacks on cloud networks, whether the replay attack is partial or complete. In this paper, a secure TKN-cryptographic algorithm will be developed to protect the network from partial and full message replay attacks.

The method described in this study has a unique characteristic in that it generates a greater number of cipher texts compared to the original text. When an adversary attempts to engage in cryptanalysis or execute an attack, they will regard each cipher value as the true value, resulting in the acquisition of data that is unrelated to the original data, rather than obtaining the original data itself. The produced value resulting from the suggested technique remains inaccessible for decryption until the implementation of the proposed decryption algorithm.

1.5 Motivation

This research aims to design an algorithm that ensures comprehensive data protection inside cloud networks, explicitly targeting the prevention of replay attacks on cloud data. To safeguard the data, it is necessary to use approaches that provide a high degree of resistance against unauthorized access, rendering them impervious to both simple breaches and the application of cryptanalysis methods. The suggested technique facilitates the detection of partial and complete message replay attacks, enabling easy identification of the attacker's actions. The linkage of each value serves the purpose of safeguarding the data from potential replay assaults. Modifying a single value will have a consequential impact on the whole text. When the method mentioned above is used for data security, the efficacy of the attacker's data-snatching technique will be rendered ineffective. Furthermore, even if an alternative approach is constructed, it will yield results without any meaningful correlation to the original values.

The remaining sections of the document will be disseminated using the following approach. In Section 2, an overview of the prior research will be provided. In Section 3, the proposed work will be discussed. Section 4 will empirically demonstrate the validity of the planned study. In Section 5, a comparative analysis will be conducted between the proposed study and past research. The paper will be concluded in Section 6.

2 Literature Review

To determine the challenges posed by replay assaults, Singh et al. [18] reviewed numerous articles. They also gathered all the methods, datasets, and processes that may be used to safeguard data from replay attacks. After that, various reasons for replay attacks were discussed, including lack of cryptographic techniques, use of insecure architecture, and use of insecure protocols. After surveying multiple papers and collecting various methods, It was discussed that if all these methods were combined to create a practical algorithm, cloud data might be secured against replay attacks.

Tadapaneni surveyed various papers [19] and explained how the lack of an initial level of security in the infrastructure is the primary cause of attacks on cloud networks. Researchers highlighted several security difficulties that might be employed to protect against assaults to address this problem. According to researchers, every technology consists of two stages. One stage is helpful for reliable transmission on the cloud network; the other stage is when the attacker tries to block the information of clients. After that, various techniques and algorithms were combined and discussed how the attacker ratio could be reduced if all efficient methods were used on the cloud server.

According to Zaman et al. [20], protecting the data from these attacks is challenging whenever an attacker performs replay attacks on a control system network. A LQG controller system was developed to detect and mitigate replay attacks on the control system network.

A robust network architecture was initially devised, including dropout-packet characteristics, using the Kullback-Leibler divergence algorithm. The occurrence of replay attacks on a cloud network often results in a much larger ratio of successful attacks compared to the rate at which these attacks are detected.

According to Thirumavalavasethurayar et al. [21], whenever data is broadcast on a cloud network, it can be accessed on every node connected to it when an unauthorized person tries to capture the original data packets and modify them after broadcasting these malicious packets to the network nodes. It is challenging to distinguish malicious packets from authentic packets. Replay attacks were conducted using three distinct nodes in a control area network (CAN) to address this issue, and faulty nodes were identified. Two types of replay attacks were performed to detect faulty nodes, in which the complete message was first captured, and this message was modified and broadcast to the entire network. After that, the complete message was captured, and some portion of it was broadcast. It was then discussed that whenever the data is broadcasted in the cloud network, it is easier to detect the partial replay attack than the complete replay attack message.

In 2021, a tool was developed by researchers [22] to use intrusion detection systems to safeguard cloud networks against DDoS, brute force, and pattern-matching assaults. Furthermore, the solution protects the cloud network from internal and external attacks. The first development of architecture was undertaken with the primary objective of enhancing the security of cloud networks. Subsequently, the intrusion detection system outside the cloud server identified instances of brute force and pattern-matching assaults. Consequently, the focus shifted towards devising strategies to safeguard the cloud network against such attacks. The Cloudflare technique involving static or dynamic IP address blocking to thwart attackers was implemented as a defensive measure to protect the cloud network from internal threats. Additionally, whether all devices inside the cloud network were adequately secured was raised and deliberated upon. Subsequently, it was argued that implementing these security procedures inside the network could decrease the frequency of attacks.

An encryption technique was created by Musa et al. [23] that employed a key to encrypt the sender side of the file before it was sent to the cloud server. The Key used in each text differed from the other readers. Whenever the receiver side file was decrypted, it was interpreted using the same Key. This algorithm was suitable for text-based encryption but not for file-based encryption.

Researchers have used many methodologies and algorithms to safeguard the data [18–22]. Applying algorithms developed for several articles to a certain extent is a viable approach. Every individual sheet of paper has both advantages and disadvantages. There is currently no existing research article that has been able to identify assaults and provide comprehensive security for cloud data successfully. Table 1 displays the methodologies used in various reports, methods, and problems.

Table 1: Comparative analysis

Sr#	1	2	3	4	5	Proposed work
Reference	[18]	[19]	[20]	[21]	[22]	
Proposed work	Surveyed various papers and discussed best techniques	Combines efficient replay attacks detections	Designed LQG controller for detecting replay attacks	Created three nodes CAN and found out the faulty node	Encrypted the file and broadcast file with Key on the cloud network	Can detect complete message and partial message replay attacks

(Continued)

Table 1 (continued)

Sr#	1	2	3	4	5	Proposed work
Novelty/ advantages	Secure architecture can provide data security	Discussed efficient techniques for replay attack detection	Dropout features can detect replay attacks easily	Able to detect a complete message replay attack	Used different keys for encryption	Able to differentiate malicious data from authentic data
Research gap/ disadvantages	Required testing and results	Only find out the replay attacks problems	A high ratio of replay attacks is complex to detect	Partial message detection is difficult	An attacker can easily access the data by key that is uploaded with the file	Find out all existing problems
Problem solutions in the proposed work	Developed a tool, tested it, and showed all results	Provide solution for replay attack detection	Detect a high ratio of replay attacks	Detect complete message and partial message replay attacks	Encrypt data with a single key	Solve all problems in the proposed algorithm

3 Proposed Methodology

Several actions will be implemented to safeguard the integrity of cloud data from replay assaults. Initially, a proficient algorithm will be devised to facilitate the transmission of the unencrypted text. After implementing the encryption mechanism, the data will be broadcast. After that, cryptanalysis mechanisms will be implemented on the data, and it will be identified how the proposed algorithm can be used to protect the data and to what extent the data can be protected from replay attacks.

3.1 Network Architecture

First, an architecture has been developed to broadcast the data, as shown in Fig. 4. A SaaS application has been used to promote the data to different users. All the users who will connect to the cloud server can access SaaS applications, whether those users are valid or there is an attack.

When a user commences data transmission to a cloud server, the user will first use an encryption technique. Then, the encrypted data will be broadcast to the SaaS application. When the receiver receives the data, the receiver will implement a decryption mechanism on the data. After the implementation of encryption mechanisms, the data's authenticity will be checked. Text data validity will be reviewed in two different ways. The receiver will get the original file after decryption if the file is authentic. If any change has been made in the data, whether the change has been made to one word or more than one word. The decrypted data will exhibit a distinct relationship compared to the original data due to the encryption process, which results in the encrypted data being twice the magnitude of the actual value. The number of encrypted text obtained with the SaltVar mechanism will be doubled. Then, the Key and Username will be added to the encrypted data using a loop-through algorithm so that the attack occurs when the encryption mechanism is applied to this data; it becomes difficult for an attacker to Figure out which data is the real one and if any change is made in the data, then all the data will be changed due to the SaltVar mechanism.

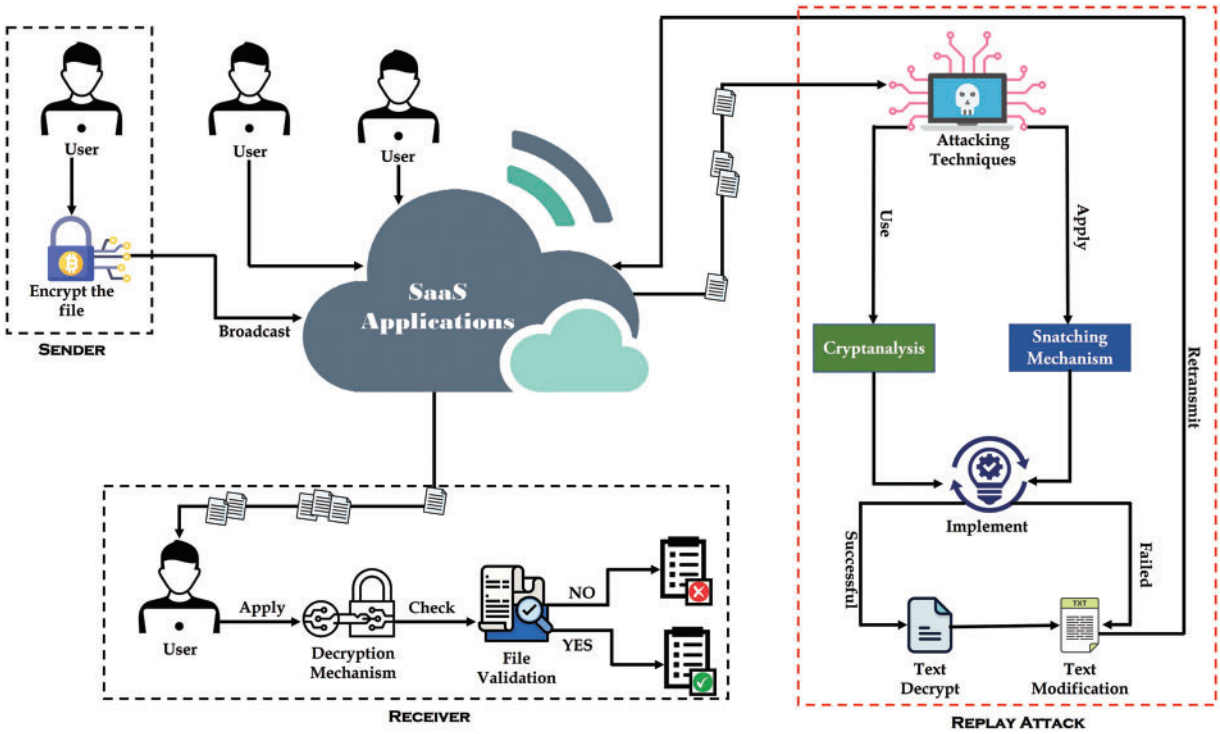

Figure 4: Proposed network architecture

When an attacker deploys any technique to gain entry into the cloud server, their primary objective is to get unauthorized access to the data. When a robust method is in place to safeguard data, unauthorized access to the data is effectively prevented, irrespective of the algorithm's quality. In the context of cloud servers, a replay attack refers to the malicious act of attempting to exploit data by an attacker after its reception. If the data is subjected to encryption, the assailant will exert significant efforts to decipher it. When the perpetrator cannot get unauthorized access to the data, they may resort to altering the data or disseminating compromised data around the cloud network. Differentiating between original and malicious data becomes problematic in the event of a partial or complete message replay assault on the cloud network or when the actual data is replaced with harmful data. This study presents the development of an algorithm that effectively solves the issue by enabling the differentiation between authentic and malicious data.

3.2 Text Encryption System

Initially, plaintext will be used to encrypt the textual data. Subsequently, each character inside the plaintext will be assigned an index value. Once the plaintext has been indexed, all the indexed values will be transformed into 8-bit binary representations. Once the conversion to Binary is completed, the Saltvar mechanism will take further steps. The binary numbers obtained will thereafter undergo conversion into decimal representation by forming pairs of 8 bits. After obtaining the Decimal values, the technique that utilizes a loop will be conducted, as seen in Fig. 5.

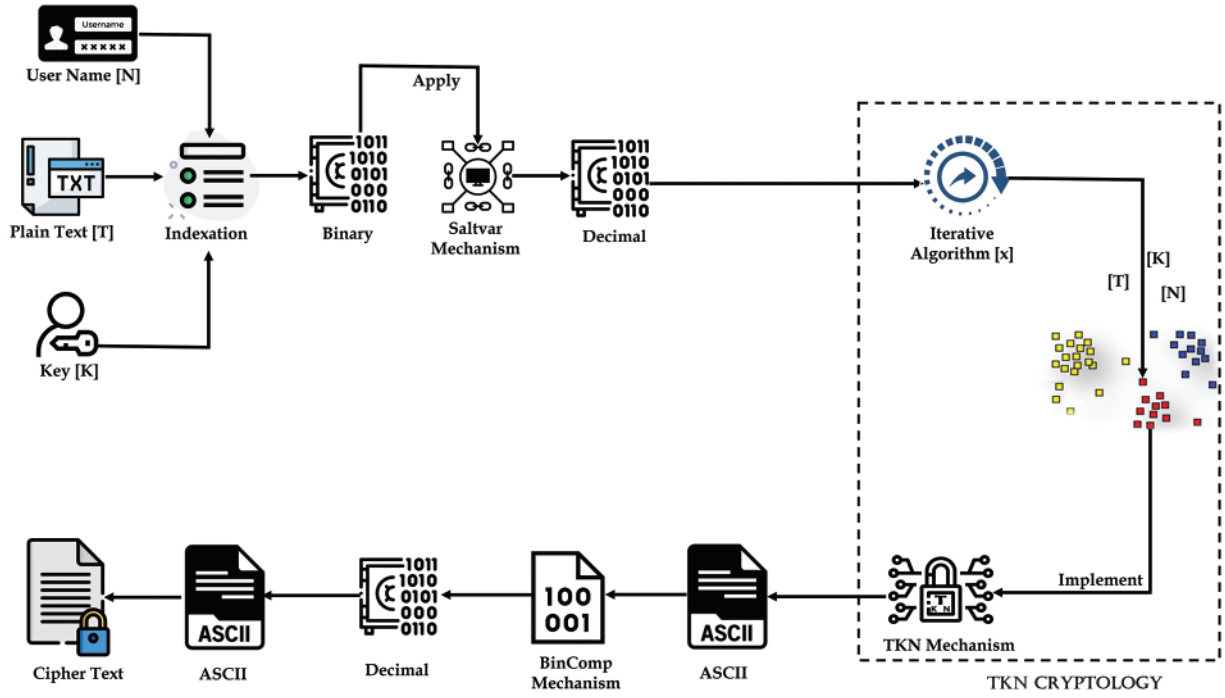

Figure 5: Data encryption system

The loop-based algorithm will perform a number-of-time looping on each decimal value. When the loop-based algorithm is implemented on the received value, the value in the next existing value will be retrieved. When the attacker applies the encryption mechanism or cryptanalysis on these values, the attacker will get values entirely unrelated to the original values. After that, the TKN cryptology will be used. TKN cryptology will combine encrypted values of Text [T], Key [K], and User Name [N], and the arrangement will be provided with the help of TKN cryptology. The same procedure will be applied to the key and secret code while transforming plain text into encrypted text. In cloud networks, when an attacker engages in a replay assault, it might manifest as either a partial or complete message replay attack. Any change will be detected with the help of TKN cryptology. After that, every value obtained from TKN cryptology will be converted into Binary, and the BinComp mechanism will be implemented. The BinComp mechanism will reverse the resulting binary values, from which the obtained values will be unrelated to the actual values. The result obtained from the BinComp mechanism will be converted to decimals to get different decimal values, and then a secure cipher text will be received.

3.2.1 Saltvar Mechanism

This study aims to propose a Saltvar technique for data storage, whereby a total of eight binary bits will be used. Specifically, four binary bits will be appended to the beginning of the plaintext binary, while the remaining bits will be appended to the end. The reason for using the Saltvar mechanism is that when 4-bit of the Saltvar mechanism and 4-bit of plaintext are mixed up with each other, then a new 8-bit pair will be created, from which the decimal values obtained will be completely different from the original values and all the values will be linked to the plaintext.

3.2.2 Loop-Based Algorithm

The loop-forward algorithm is a repetitive algorithm in which the existing values are added, called the loop-forwarding algorithm. The "$x += x$" formula will be implemented in the loop forwarding algorithm. If the value is greater than 255, then a loop backwarding algorithm will be implemented on the matter, in which the formula "$x- = x$" formula will be used. The specific method will be used for the key and secret code to transform plaintext into encrypted text.

3.2.3 TKN Cryptology

The TKN cryptology will be implemented on the loop-based algorithm's values. The Username in the front end will be considered the secret code, but the Username in the back end will play the role of a secret code. Usernames always use three symbols, which are underscore (_), period symbol (.), and asperand (@). While all the characters can be used in the secret code, if an attacker gets hold of the TKN mechanism, the attacker will consider [N] as a username and try to decrypt it, but actually, [N] is a username. There will be a secret code that will be hidden between the two parties.

3.2.4 BinComp Mechanism

The BinComp mechanism is a process that involves reversing each binary value. The value of 0 will be converted to 1, whereas the value of 1 will be converted to 0. The complete steps of Encryption algorithm has been discussed in Algorithm 1.

Algorithm 1: Encryption

Input: Plain Text, Key, Secret Code
Output: Cipher text

1. Get Plain Text.
2. Find indexes of each character of plain text.
3. Convert each index value into 8 bits Binary.
4. The Saltvar technique will be implemented on the binary values acquired in step 3.
5. Convert each 8-bit Binary into Decimal.
6. Apply the Loop Forwarding mechanism on decimal values.
   If decimal > 255, then Apply the Loop Backwarding mechanism
   Otherwise, go to step 7.
7. Convert decimal values into ASCII.
8. Apply all these steps on Key [K] and Sender name [N] to get an encrypted text.
9. Apply the TKN Cryptology mechanism on the encrypted text of Text [T], Key [K], and Sender name [N].
10. Convert values obtained in step 9 into ASCII.
11. Convert each ASCII value into an 8-bit binary.
12. Apply BinComp Mechanism on binary values.
13. Convert binary values into decimals.
14. Convert decimal values into ASCII.
15. Obtained cipher text.

3.3 Text Decryption System

Various steps will be implemented to decrypt the text, taking a cipher text. The cipher text will be indexed first, then the indexed cipher text will be converted to Binary, and BinComp will be

implemented on the binary text. After implementing the BinComp mechanism, 8-bit binary pairs will be created, and then each pair will be converted to decimal and converted to equivalent ASCII. TKN Segregation mechanism will be implemented after converting to Equivalent ASCII, illustrated in Fig. 6.

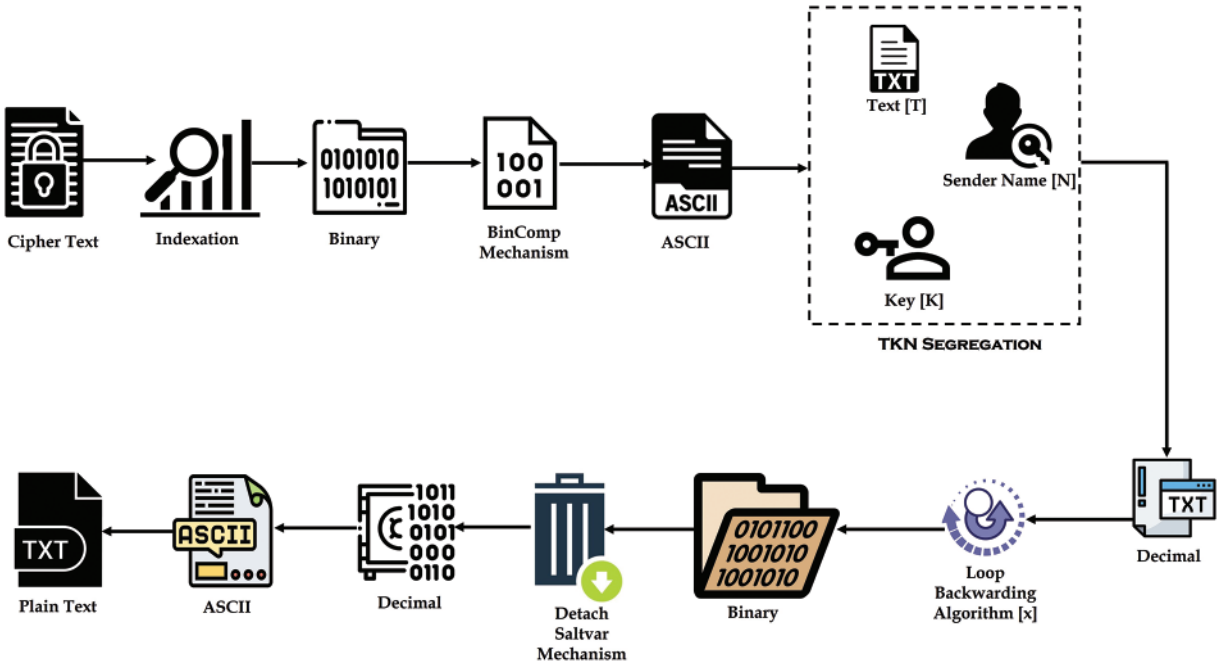

Figure 6: Data decryption system

In TKN segregation, the ASCII text will be divided into three parts: the first part is Text [T], the second part is the Sender Name [N], and the third part is the Key [K]. After implementing TKN Segregation, each value will be converted to a decimal value, and a loop backwarding algorithm will be implemented on decimal values. In the loop backwarding algorithm, the "x− = x" formula will be applied to each value, and the result will be converted into Binary. Upon converting the values to Binary, the Saltvar process will be concluded, subsequently converting the values to decimal and their respective ASCII equivalents. A plain text will be acquired. The complete steps of Decryption algorithm has been discussed in Algorithm 2.

Algorithm 2: Decryption

Input: Cipher text, Static Key
Output: Plain text, Key, Secret Code

1. Get Cipher Text.
2. Find ASCII indexes of cipher text.
3. Convert step-2 values into 8-bit Binary.
4. Apply *BinComp* Mechanism on step-2 values.
5. Convert decimals from binary numbers.
6. Convert decimal values into ASCII.

(Continued)

Algorithm 2 (continued)

7. Separate the Key and sender name from obtained ASCII values by adding 1 in the original length of the Key and sender name.
   The remaining ASCII values will be of text.
8. Convert ASCII values into decimals.
9. Apply the Loop Backwarding mechanism on decimal values.
10. Convert decimal values into Binary.
11. Detach the bits after removing the salting bits from the beginning and end of the binary values.
12. Convert obtained binary values into decimals.
13. Convert Decimal into ASCII.
14. Obtained Plain text.

4 Testing

4.1 Text Encryption System

Step 1: First, a simple text including various letters, numbers, and symbols is taken. All plain text is encrypted using multiple characters, alphabets, and numerals. In Fig. 7, the simple text is shown.

N@deem$123

Figure 7: plain text

Step 2: Upon getting the plaintext, a process is initiated wherein each character inside the plaintext is assigned an index. Subsequently, each character is transformed into its corresponding ASCII representation, as seen in Fig. 8.

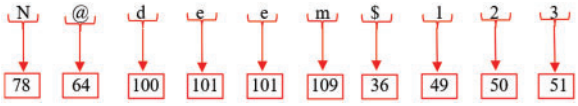

Figure 8: Decimal to ASCII conversion

Step 3: After the conversion of plain text to ASCII, it can be seen that each ASCII value is subsequently transformed into binary, as shown in Fig. 9.

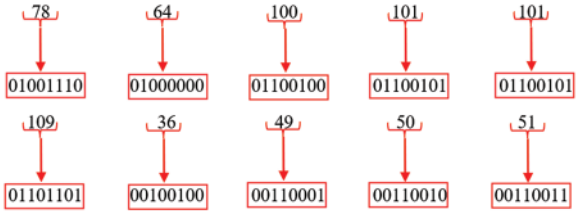

Figure 9: ASCII to binary conversion

Step 4: After converting the various ASCII values into binary representations, each Bit undergoes a reversal process, as shown in Fig. 10.

Figure 10: Bit reversing mechanism

Step 5: The SaltVar mechanism is used after the reversal of the bits. In the Saltvar technique, the Bit reversing result is augmented by "1101" bits at both the beginning and end, as seen in Fig. 11.

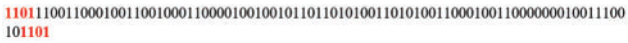

Figure 11: Saltvar mechanism

Step 6: Upon implementation of the SaltVar mechanism, the 4-bit plain text and 4-bit Saltvar mechanism are combined to create an 8-bit entity. Similarly, each preceding 4-bit and subsequent 4-bit are united to generate 8-bit pairs, as seen in Fig. 12.

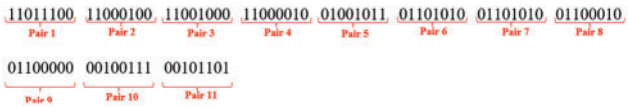

Figure 12: Bits pairing

Step 7: After forming distinct pairs, each 8-bit pair converts to Decimal. The resulting decimal values are seen in Fig. 13.

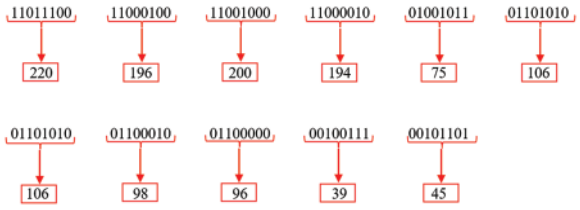

Figure 13: Binary to decimal conversion

Step 8: Following the gathering of ASCII values from various pairings, a loop-forwarding algorithm is executed on each decimal value based on the contextual circumstances of the values. Each decimal value is substituted in this process with the subsequent value, as shown in Fig. 14.

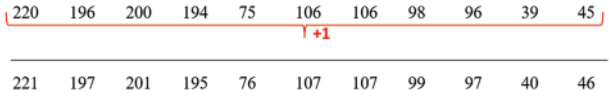

Figure 14: Loop forwarding algorithm

Step 9: After the loop forwarding mechanism is effectively implemented, a conversion procedure is triggered wherein each decimal number is converted into its respective ASCII representation. The process of converting the original text yields a set of encrypted values, which may be seen in a visual representation shown in Fig. 15.

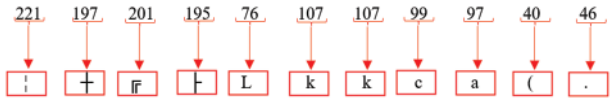

Figure 15: Encrypted text

Step 10: The plain text is encrypted using the TKN cryptology technique, which is then applied to both the key and secret code. Table 2 displays the encrypted representation of the password, key, and secret code.

Table 2: Encrypted plain text

Plain text	Encrypted plain text [T]	User defined key	Encrypted key [K]	Sendername/ secret code	Sendername/ secret code encrypted text [N]
N@deem$123	¦┼╔╞Lkkca(.	Cloud@5%9	┌╦K┴#k¬dm.	W@jia_12	ı╔ð®jfa/«

Step 11: After receiving the encrypted text, TKN cryptology is implemented to concatenate the encrypted text by first combining the encrypted text [T], then the Key [K], and then the Secret Code [N], and a secured text has been obtained. Complete message and partial message replay attacks can be detected using TKN. The encrypted text of the text, Key, and Secret Code is shown in Fig. 16.

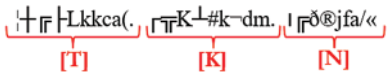

Figure 16: TKN result

Step 12: The result derived from TKN cryptology is transformed into the Decimal format, as seen in Fig. 17.

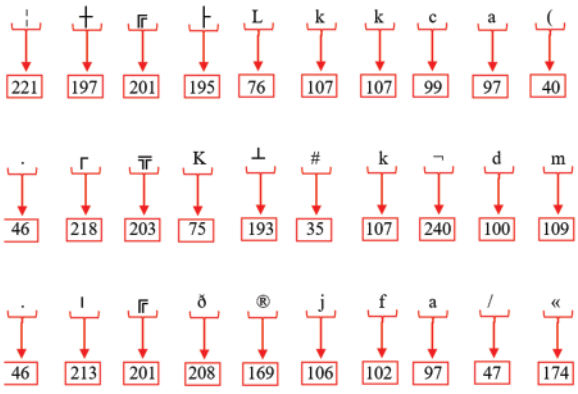

Figure 17: The decimal representations of all encrypted text values

Step 13: Following the acquisition of the various decimal values, each decimal value is transformed into an 8-bit binary representation, as seen in Fig. 18.

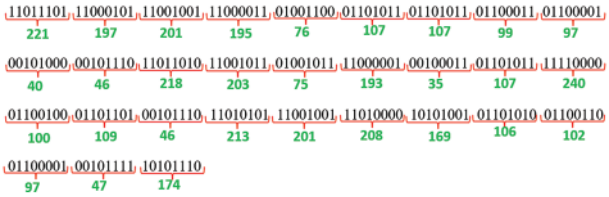

Figure 18: Binary values of each decimal

Step 14: Upon obtaining the binary values, the Bit is reversed, resulting in distinct decimal values, as seen in Fig. 19.

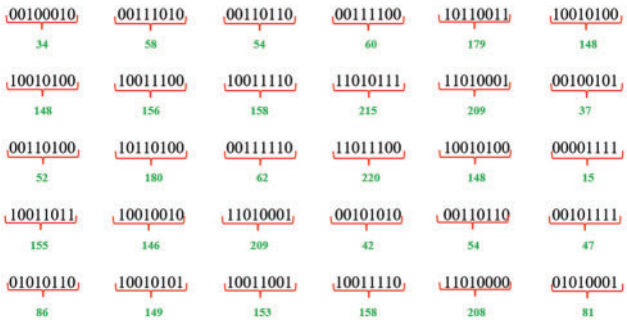

Figure 19: Invert decimal values

Step 15: Upon acquiring several decimal values, each of these numbers is then associated with its corresponding ASCII value, as seen in Fig. 20.

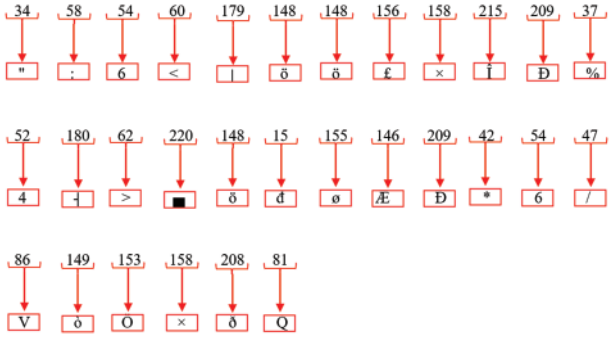

Figure 20: Equivalent ASCII value of decimal

Step 16: The acquisition of many equivalent values, as seen in Fig. 21, generates a safe ciphertext.

":6<¦ õö£×ÎÐ%4┤>■ðđøÆÐ*6/VòÖ×ðQ

Figure 21: Cipher text

4.2 Text Decryption System

Various procedures have been used in deciphering encrypted text and transforming it into its original, intelligible form.

Step 1: The given text represents a cipher text, as seen in Fig. 22.

Figure 22: Cipher text

Step 2: Upon acquisition of the ciphertext, the conversion of each cipher value to its corresponding ASCII representation, as seen in Fig. 23.

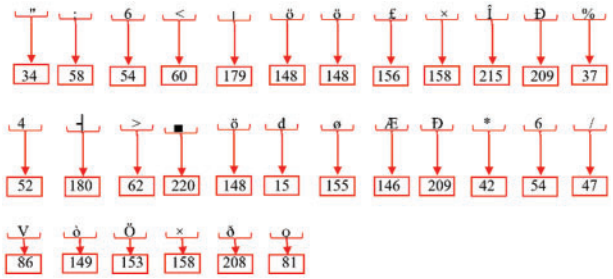

Figure 23: Decimal value of each ASCII

Step 3: After obtaining a range of decimal values, each value is converted into an 8-bit binary representation, as seen in Fig. 24.

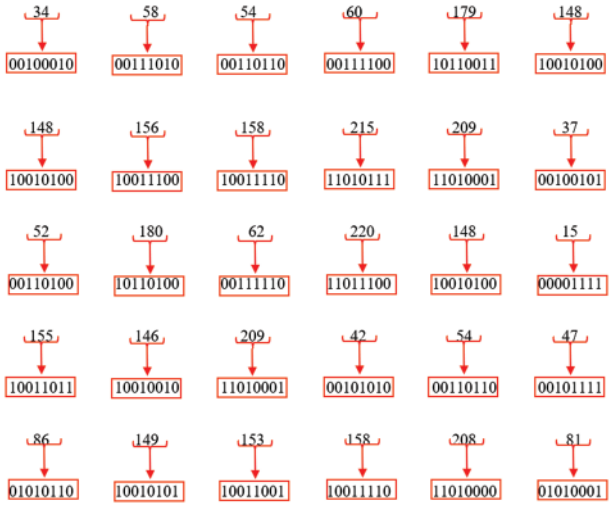

Figure 24: 8-bit binary values of each decimal

Step 4: Once the 8-bit binary values have been acquired, a technique for inverting the bits is applied to all the binary values. This process results in the generation of various decimal values, as seen in Fig. 25.

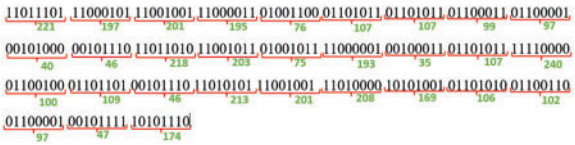

Figure 25: Decimal values of each 8-bit binary

Step 5: After collecting the various decimal numbers, each one is converted into its ASCII equivalent, as seen in Fig. 26.

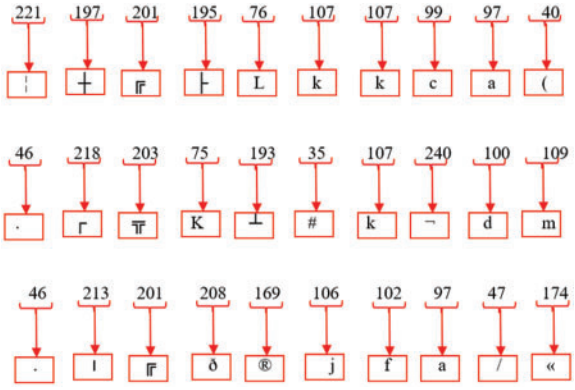

Figure 26: Equivalent ASCII of each decimal value

Step 6: Upon acquiring the distinct ASCII values, the value is partitioned into three segments. Consequently, the decrypted plaintext, Key, and Secret Code are derived, as seen in Fig. 27.

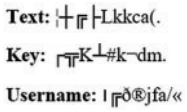

Figure 27: Values split results

Step 7: Once the values have been divided, the decryption process is executed, resulting in a set of distinct decimal values. In Fig. 28, the textual data is transformed into numeric representations.

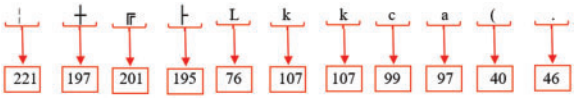

Figure 28: Decimal values of each text

Step 8: Following the obtaining of decimal values, a reverse loop algorithm is executed on each decimal value, resulting in the derivation of distinct decimal values, as seen in Fig. 29.

Figure 29: Loop backwarding results

Step 9: The process involves converting each decimal value received from backward looping into an 8-bit binary representation. This results in the generation of distinct binary values, as seen in Fig. 30.

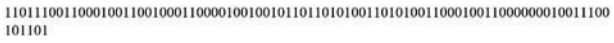

Figure 30: Decimal results of each decimal

Step 10: Once the binary data have been acquired, the Saltvar mechanism is eliminated, as seen in Fig. 31.

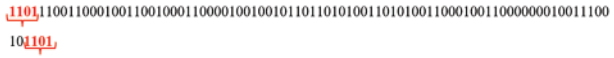

Figure 31: Elimination of SaltVar mechanism

Step 11: Fig. 32 demonstrates that the binary values undergo inversion after removing the Saltvar values.

Figure 32: Bit inversion

Step 12: The formation of 8-bit pairs occurs after the inversion of the bits. Following this, the decimal values of each 8-bit pair are translated, as seen in Fig. 33.

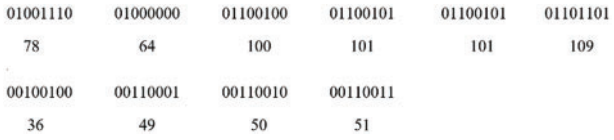

Figure 33: Decimal values of each 8-bit pair

Step 13: Following the conversion of binary numbers to decimal numbers, each decimal digit is transformed into its corresponding ASCII character, as seen in Fig. 34.

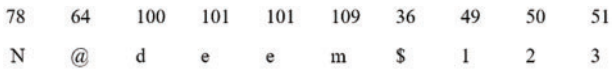

Figure 34: ASCII values of decimals

Step 14: Upon obtaining the distinct ASCII values, the values above are combined, resulting in the collection of plain text, as seen in Fig. 35.

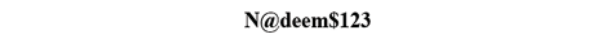

Figure 35: Plain text

4.3 Text Algorithm Testing

After developing the algorithm, the proposed algorithm has been tested at different times for which a cloud shell has been designed. First, a plaintext is taken, which has been encrypted, as shown in Fig. 36. After that, a Key has been used, and then a Secret Code has been used. The secret code is shown as the username in the front end, but actually, it is a secret code.

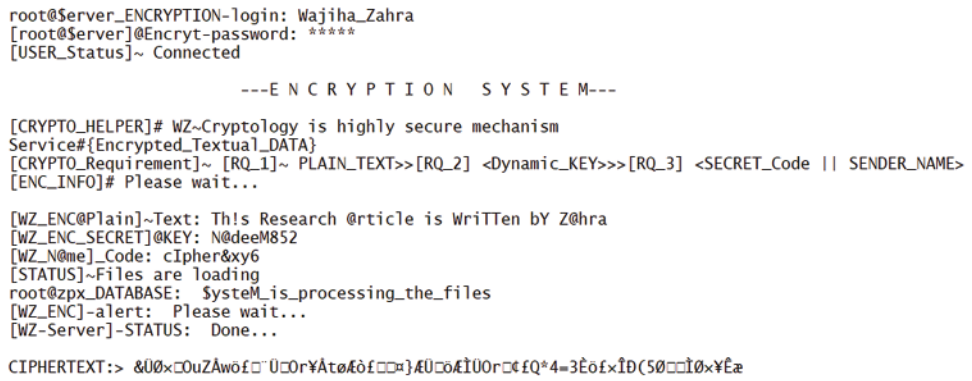

Figure 36: Encrypted text

Various instances of testing have been conducted to evaluate the tool's efficacy. Each instance included the examination of distinct plain texts using multiple keys and secret codes, resulting in diverse outcomes. These outcomes are shown in Table 3.

Table 3: Experimental results

Testing	Plain text [T]	Plaintext length [T_L]	Static key [K]	Static key length [K_L]	Secret code [N]	Secret code length [N_L]	Cipher text	Cipher text length [T_L] [K_L] [N_L]
1	Th!s Research @rticle is WriTTen bY Z@hra	41	N@dee M852	9	cIpher&xy6	10	&ÜƲ×┘OuZÅwö ƒƧUÜÈOrtÅtøÆ òƒƒ¤}ÆÜШöÆ ƎÜOrШ¢ƒQ*4=3 Èöƒ×ÎÐ(5Ʋ‖Ǝ Ø×τÊæ	63
2	mY_b@ck B0ne !s mY POWeR	24	@l!Arsh @d	9	$tuD!O$0ne	10	*TÉOɪOuSÅrШ»t ù×:¤qÆ× ┌ÅUSæ ,×¦Æƒ₩Í╗×Ð$ù ×<»╝ÈƬƬ▒	46
3	@ Secure Crypt0logy P@per	25	Muh@ MmaD NadeEM	15	Su$k!8	6	*Ǝ×ɤOuÊÅo×<ɤ τƒε¤tÜƎεöÆO~ Ð#ÈÈƒûù¤|Íôôɪ ¦ö/Ð-6▒ƒ┤εQ	49
4	W@j!h@ Z@hRa	12	Z@iN!	5	Com%p7	6	&Ü]×┘O~¦û╣× ðQ&Àı×┘Q8>Ǝ ▏ÅÆÐ	26
5	C0mPuter Sc!ence	16	Z@hra	5	Mu$k!@	6	$Æùòû▓ÆOzƎƒ Ǝ»S×2Ð&ÜƲ×┘ Q.i▒ƒ┤/Ð	30
6	Cl0ud seCur!ty	14	VeRn@m	6	C!ph%r@t#	9	%ƬШ‖ƎεÈÆd|ö ɤ;ÆÐ#×ÎÜTÿQ" ╝ɤ┌Ǝ¢×Ш▒Ð	32
7	Re$e@rCh p@pEr	14	SyEd@ Z@hr@	11	Pl@y&F@iR	9	*Ǝɪɤdđ}Æ┌ɤÈƒ┤ ÜQ.┌Ʋ×┘O~▄öɪ εQ*U×ÏÏÁɤ▉×Q	37
8	Th!s I$ iNf0rm@tion $ecurity	28	tHe$!s@5	8	pR0Ject%1@	10	%ƬτÜƎεöƒ»wÅò ƒɤËÜɤ8ùùÅ|Á¤r Ш¢ƒQ$>ÊШ╝┤Ø ▄ɳ.Í4╝εöÖɪ^η	49

Distinct cipher text values may be derived by encrypting the plain text using various keys and secret codes. The quantity of cipher texts is twice as large as the original values. When the amount of cipher text surpasses that of the original text, it becomes arduous for an adversary to decipher each value of the cipher text. When substituting each value using a distinct technique, the actual value will no longer correlate with the cipher value. The plain text was encrypted using a cryptographic tool, and a decryption process was devised to decipher the resulting cipher text. The decryption process involves utilizing a cipher text on a decryption tool to acquire a plain text, representing the original content as seen in Fig. 37.

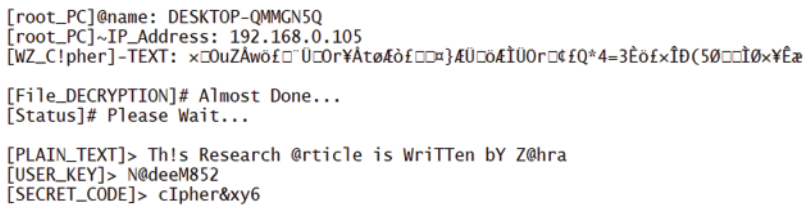

Figure 37: Decrypted text

4.4 Text Prevention from Replay Attacks

After obtaining a cipher text, the attacker employs several decryption techniques to decipher the encrypted message. The innovation of the work lies in the fact that, regardless of the number of distinct methods developed by an attacker, they cannot decode the ciphertext using the suggested approach. In situations when the attacker is unable to exploit the encryption process, their subsequent strategy involves conducting a replay attack on the cipher text data. The perpetrator modifies the values of the encrypted text and disseminates them to the cloud server, resulting in a whole message replay. Complete message and partial message replay attacks have been detected with the help of this tool. The first entire message replay attack has been detected, for which a cipher text was taken, and some values from this cipher text have been replicated with malicious values. Malicious values are those values that have nothing to do with ciphertext or plaintext. After replacing the values, the cipher values were implemented on the tool, and a plaintext result was obtained, which has no link to the plaintext. Due to the change of some values in the cipher text, the link to the original values is lost, as shown in Fig. 38. The proposed approach may effectively identify malicious ciphertext messages resulting from an entire message replay assault, whereby an attacker alters some values or key values.

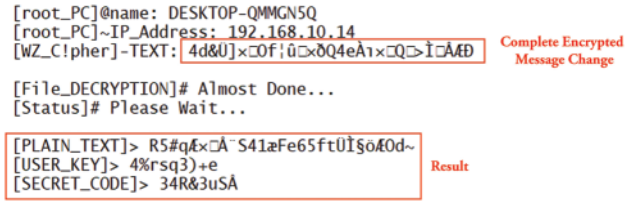

Figure 38: Complete message replay attack

When an attacker removes some values from the encrypted text or broadcasts it by reducing the cipher text length, such attacks are called partial message replay attacks. From different researchers' points of view, a partial message replay attack is more complex to detect than a complete message replay attack. In this paper, a partial message replay attack is detected after detecting a complete message replay attack. In which the attacker has deleted some cipher values and broadcasted the data.

When the attacker employs the broadcast message on the tool, the resulting values gained are not connected to the original plain text. Furthermore, the obtained plain text values change significantly, as seen in Fig. 39.

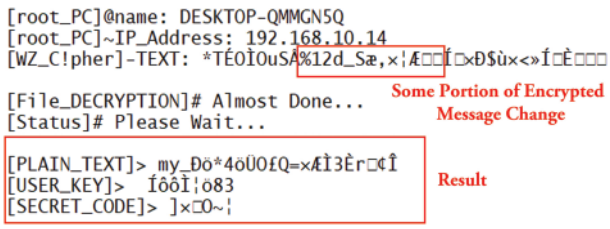

Figure 39: Partial message replay attack

4.5 Cryptanalysis

Cryptography is a technique in which data is converted into a format that is neither easily understood nor broken. In cryptography, each character of plaintext data is converted into cipher format using various mechanisms and techniques. In which a key is also used, whether that Key is static or dynamic. Cryptanalysis is a process in which predictions are made on cipher text using various mechanisms, and the resulting Key is implemented on the cipher text. This study employs two distinct cryptanalysis techniques to determine the feasibility of extracting the Key from the cipher text generated by the proposed method.

4.5.1 Cryptanalysis by Plain Text and Cipher Text

To evaluate the efficacy of the suggested algorithm, the cryptanalysis process is conducted on both the plaintext and ciphertext. This involves determining the length of the plaintext [T] and the ciphertext [T][K][K]. The Key can be obtained from the cryptanalysis mechanism only if the length of the plaintext and cipher text is equal. If both sizes are unequal, no cryptanalysis algorithm can be implemented, and the Key can be predicted. If both lengths are not equal, the key prediction is impossible. Even if the Key is obtained from a cryptanalyst, it cannot be used on the cipher text, nor can the plaintext be obtained.

Plaintext and ciphertext cryptanalysis include the examination of distinct plaintext values for verification purposes. The method under consideration has been executed on these values, as seen in Fig. 40. Several significant findings have been acquired, as shown by the data presented in Table 4. Upon obtaining varying outcomes, the lengths of both the plain text [T] and encrypted text [T][K][N] have been determined. The length of the cipher text is double that of the plain text, indicating that the cipher text and plain text cannot be of equal length owing to the implementation of the suggested method. In the event of disparate sizes, implementing the cryptanalysis method becomes unfeasible, precluding the possibility of key prediction. To accurately forecast the Key, it is necessary for the length of both the cipher and plain text to be the same.

4.5.2 Cipher Text Cryptanalysis

The Kasiski test is a cryptographic procedure used to generate the Key from the given data. The snatching algorithm is used to get the Key from the Kasiski test, as seen in Fig. 41.

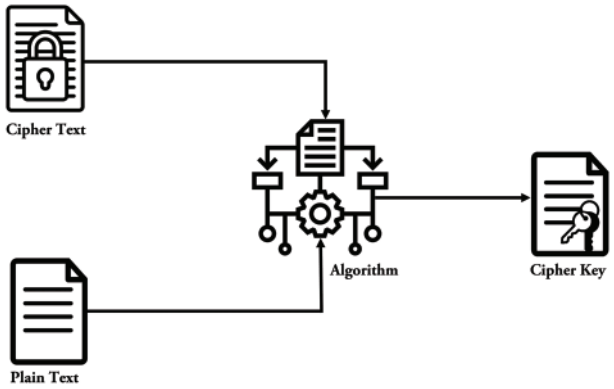

Figure 40: Plain text and cipher text cryptanalysis

Table 4: Results of plaintext and ciphertext cryptanalysis

Sr#	Plaintext	Text length [T]	Cipher text length X = [T][K][N]	Length equalization (T, X)	Cryptanalysis algorithm	Key prediction possibilities
1	Th!s Research @rticle is WriTTen bY Z@hra	41	63	T ≠ X	No	No
2	mY_b@ck B0ne !s mY POWeR	24	46	T ≠ X	No	No
3	@ Secure Crypt0logy P@per	25	49	T ≠ X	No	No
4	W@j!h@ Z@hRa	12	26	T ≠ X	No	No

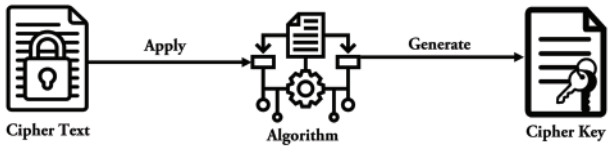

Figure 41: Cipher text cryptanalysis

The first step in the process involves extracting the Key from the cipher text using the Kasiski test. Subsequently, all the recurring values employed in the cipher text are identified. Once the repeating values have been identified, the Index of each repetitive value has been assigned and recorded in Table 5. The Kasiski length method has been used to index the repetitive value, whereby the distance between the initial value and the xth value has been ascertained. The variable "y" has been used to denote distance. Once the distance has been determined, implementing the Greatest Common Division (GCD) has been carried out on the variable "y". The value received from the greatest common divisor (GCD) determines the key length. Implementing an index of coincidence is feasible when the Key

and cipher text lengths are equivalent. Identifying the Key becomes unattainable when there is a discrepancy in the lengths of the key and cipher text. The Index of Coincidence is used to identify both Favorable Cases and Possible Cases. Favorable cases refer to instances when random numbers being equal may be predicted. The phrase "Total Cases" refers to the several different methods of determining the length of the Key. The cipher texts derived by the suggested method exhibit complete dissimilarity to the original text due to the use of a strategy that involves substituting the original values with alternative values and interconnecting all the words. The repetition of a value does not always imply the feasibility of accurately predicting the corresponding Key. The Key may alone be obtained from the cipher text by substituting each value with its corresponding ASCII value. The method suggested in this study has a unique characteristic wherein all attempts to exploit it, regardless of the attacker's proficiency in developing cryptanalysis algorithms, become futile when the values are altered. This attribute distinguishes the algorithm and contributes to its uniqueness. The complete Steps of Kasisiki Test algorithm has been discussed in Algorithm 3.

Algorithm 3: Kasiski test

Input: Ciphertext
Output: Key Identification

1. Ciphertext
2. Identify the repeatable ciphertext values
3. Identify repeatable value indexes.
4. Find the length by the Kasiski Length algorithm.
   Identify the gap between the initial number and with xth number by using the $Y = Y_1 - Y_n$ Formula.
   Identify the GCD of all gaps.
5. Find the Key Length by using step 4.
6. IF CIPHER_TEXT_SIZE = = KEY_SIZE, then go to the next stage.
   ELSE (FIND_KEY) and go to step 9.
7. Apply Index of Coincidence
   $J_c(Y) = (\text{Favorable cases/Total Possible cases})$
   $J_c(Y) = (H_i/T_c)$
8. Kasiski Key
9. Exit

Table 5: Cryptanalysis of cipher text

Proposed algorithm			Kasiski test			
Testing	Cipher text length	Repeatable values distance	GCD	Length equalization (V, K)	Index of coincidence algorithm	Key identification
1	30	30	4	$V \neq K$	No	No
2	63	63	7	$V \neq K$	No	No
3	46	46	6	$V \neq K$	No	No
4	49	49	4	$V \neq K$	No	No
5	26	26	3	$V \neq K$	No	No

4.6 Algorithm Efficiency

A novel technique has been devised to enhance cloud data security by mitigating the risk of replay attacks. The suggested algorithm exhibits originality in its interconnectedness of each value. The suggested approach may effectively detect whole or partial message replay attacks and identify data tampering by an attacker. The suggested approach results in a threefold increase in the number of cipher texts compared to the original text. Additionally, all values are substituted with unrelated values unrelated to the original values. Access to the data will only be feasible upon the same procedure for decrypting the generated data. In the event that an adversary attempts to acquire the Key by methods such as cryptanalysis, the suggested algorithm will effectively prevent the execution of such a methodology. The decryption of the data by the attacker is rendered infeasible due to the attacker's erroneous assumption that each cipher value corresponds to the original value, resulting in an incorrect interpretation of the data. When using such algorithms for cloud security, data may be effectively safeguarded against unauthorized access and data exfiltration methods.

The suggested algorithm tool has been deployed on several servers to evaluate the algorithm's effectiveness. Testing has been conducted on each server using different Operating Systems (OS). The first step included determining the lengths of the plain, Key, and cipher text. Subsequently, the RAM allocation for the plain and encryption text has been calculated, as shown in Table 6. Once the memory allocation has been established, the subsequent step involves calculating the time allocation for the encryption and decryption processes of the data. The algorithm under consideration is a resilient and efficient method for performing data encryption with high reliability and ensuring comprehensive data security.

Table 6: Algorithm efficiency and its runtime performance

Performance					Space complexity		Space complexity	
Servers	Window O. S	Plain text length	Key length	Cipher text length	Plaintext memory allocation	Cipher text memory allocation	Data encryption time (sec)	Data decryption time (sec)
1	7	41	9	63	35.21 KB	62.37 KB	13.63	24.85
2	7	24	9	46	29.54 KB	57.34 KB	2.49	23.47
3	8	25	15	49	30.17 KB	59.81 KB	10.24	22.85
4	10	12	5	26	21.16 KB	43.29 KB	08.65	16.73
5	10	16	5	30	24.43 KB	47.53 KB	10.46	20.41
6	11	14	6	32	23.37 KB	48.72 KB	09.41	19.54
7	11	14	11	37	23.34 KB	46.52 KB	09.46	21.75
8	11	28	8	49	31.75 KB	60.49 KB	10.56	23.85

5 Comparative Analysis

Researchers have investigated various strategies and algorithms to safeguard data against replay assaults. However, it is essential to note that these techniques do not provide absolute security for the data, and it is not possible to exclude the occurrence of such attacks.

Some researchers [18] have collected effective methods for recognizing replay attacks after surveying several articles. The research clarified that the absence of a secure cloud architecture is the primary cause of replay attacks on the cloud. Notably, this study does not provide practical methods for protecting data against replay attacks.

In the article [19], researchers identified the leading cause of replay attacks as cloud security issues. Instead of initial level security, algorithm-based protection could secure cloud data from internal and external threats, which is the paper's challenge.

In their study, the researchers devised an LQG controller system to identify replay assaults inside the control system network. To do this, they developed a secure architecture and included dropout-packet characteristics. Furthermore, the researchers highlighted the observation that the occurrence of replay attacks surpasses the effectiveness of replay detection. This information can be found in reference [20]. The subject addressed in this article pertains to the protection of cloud networks via the development of an effective algorithm. Regardless of the replay attack ratio, implementing such an algorithm ensures the security of the cloud network.

Researchers developed a Control Area Network (CAN) to safeguard data from replay assaults [21]. The experiment included utilizing three nodes, and replay attacks were executed on each node. The complexity of detecting partial message replay attacks was shown to be greater in comparison to full message replay attacks throughout the discussion. The subject addressed in this research pertains to the protection of cloud networks, which may be achieved by developing an algorithm capable of detecting both forms of assaults.

In the paper [22], the researchers implemented a security measure to safeguard the cloud network from potential assaults. This included encrypting the file at the sender's end and uploading the encrypted file with the corresponding encryption key to the cloud network. In the process of decrypting the file, the recipient employs the same encryption key that was used to encrypt the data. The acquisition of the Key by the attacker introduces a vulnerability that facilitates the decryption of the file, constituting the central issue addressed in this article.

Numerous scholars have conducted extensive studies on diverse methodologies to safeguard data, specifically focusing on their efficacy in countering replay assaults. The present study examined a range of scholarly articles on safeguarding data from replay assaults. The researchers analyzed and elucidated the many strategies used in these studies, highlighting their efficacy in mitigating the risks associated with replay attacks on data. None of the existing publications have successfully devised methodologies that comprehensively safeguard cloud data from replay attacks, nor have they effectively detected such assaults. After a comprehensive examination of many methodologies, it has been determined that relying just on a single methodology is insufficient to provide total data protection.

Furthermore, no strategy can be deemed entirely dependable when applied to cloud environments. The suggested technique presents a revolutionary approach by effectively detecting all instances of replay assaults on cloud systems. By using this algorithm for cloud data, it offers comprehensive protection for the cloud network.

Novelty of the Proposed Work

This study presents the development of a proficient algorithm for data preservation, wherein the bits are interconnected utilizing the Saltvar mechanism. In the Saltvar mechanism, distinct pairs are created by appending 4 bits at the beginning and 4 bits at the end of the plaintext bits. Altering a single value within these pairs induces a complete modification in cipher text. Subsequently, employing a loop-based algorithm, the values undergo multiple shifts iteratively. When the values are shifted, the

cipher text will be linked to the shifted values, but a loop-based algorithm must be used to retrieve the values. Then, TKN cryptology is implemented on the obtained values to detect replay attacks. In cases where an attacker manipulates the original values or broadcasts malicious values instead of actual values, the interconnection established by the proposed algorithm ensures that each value remains inherently associated.

Consequently, employing the proposed algorithm during decryption facilitates the identification of replay attacks. Numerous symmetric algorithms have been devised to protect the data, exhibiting reliability for cloud data protection, including AES. However, the novelty of the proposed algorithm lies in its capacity to yield cipher text three times longer than the original text, followed by the interconnection of each value. This approach uniquely imparts a pervasive impact on the entire text upon altering a single value, setting it apart from any existing algorithm. When such algorithms are applied to secure cloud data, regardless of the attacker's proficiency, the integrity of the proposed algorithm remains impervious to breach. Furthermore, the novel characteristics of the proposed algorithm render it immune to attacks such as cryptanalysis.

6 Conclusion

Based on the successful development of a robust algorithm and the implementation of safeguards against replay attacks in cloud data storage, it can be inferred that the use of the suggested algorithm inside a cloud network would render the data inaccessible, regardless of the efficacy of any potential algorithm devised by an attacker. If an individual attempts to perform a snatching operation via the cryptanalysis technique, it is essential to note that even if the attacker manages to get the Key, they cannot use such Key on the encrypted data due to alterations made to the data. Several experiments were conducted to evaluate the uniqueness of the suggested method, with diverse outcomes. After acquiring diverse outcomes, the suggested method was compared with a range of distinct couples. The effectiveness and safety of the suggested algorithm were deliberated over, with specific attention given to its ability to identify both entire message and partial message replay assaults. The suggested technique offers a unique approach to detecting both assaults: replay attacks on the data.

In the future, the public key cryptography mechanism will be used to protect the data, in which the data will be protected with the Private Key, and the Public Key will be used for decryption. After that, different techniques will be compared to protect data from replay attacks and DDoS attacks, and an effective strategy will be developed to protect against both types of attacks.

Acknowledgement: Not applicable.

Funding Statement: The author would like to thank Deanship of Scientific Research at Majmaah University for supporting this work under Project Number R-2023-811.

Author Contributions: S.W. Zahra Methodology; M. Nadeem Conceptualization; A. Arshad Software, Supervision; S. Riaz Writing—review & editing; M. Abubakr Editing; A.K. Dutta Formal analysis; Z. Alzaid Proof read, Funding acquisition; B. Almutairi Formal analysis, Funding acquisition and S. Almotairi Editing and Funding acquisition.

Availability of Data and Materials: Not applicable.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] A. Katal, S. Dahiya and T. Choudhury, "Energy efficiency in cloud computing data centers: A survey on software technologies," *Cluster Computing*, vol. 26, pp. 1845–1875, 2022.

[2] G. H. Lokesh and G. BoreGowda, "Phishing website detection based on effective machine learning approach," *Journal of Cyber Security Technology*, vol. 10, pp. 1–14, 2020.

[3] S. Saxena, A. Shrivastava and V. Birchha, "A proposal on phishing URL classification for web security," *International Journal of Computer Applications*, vol. 178, pp. 47–49, 2019.

[4] N. Chaabouni, M. Mosbah, A. Zemmari, C. Sauvignac and P. Faruki, "Network intrusion detection for IoT security based on learning techniques," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 3, pp. 2671–2701, 2019.

[5] S. T. Zargar, J. Joshi and D. Tipper, "A survey of defense mechanisms against distributed denial of service (DDOS) flooding attacks," *IEEE Communications Surveys and Tutorials*, vol. 15, no. 4, pp. 2046–2069, 2013.

[6] Y. Gu, K. Li, Z. Guo and Y. Wang, "Semi-supervised K-means DDoS detection method using hybrid feature selection algorithm," *IEEE Access*, vol. 7, pp. 64351–64365, 2019.

[7] Y. Sharma, H. Gupta and S. K. Khatri, "A security model for the enhancement of data privacy in cloud computing," in *Amity Int. Conf. on Artificial Intelligence*, Dubai, United Arab Emirates, pp. 898–902, 2019.

[8] C. Palanisamy, T. Kumaresan and S. Varalakshmi, "Combined techniques for detecting email spam using negative selection and particle swarm optimization," *International Journal of Advanced Research Trends in Engineering and Technology*, vol. 3, pp. 1102–1106, 2016.

[9] S. Newman, "Under the radar: The danger of stealthy DDoS attacks," *Network Security*, vol. 2, pp. 18–19, 2019.

[10] V. Schlatt, T. Guggenberger, J. Schmid and N. Urbach, "Attacking the trust machine: Developing an information systems research agenda for blockchain cybersecurity," *International Journal of Information Management*, vol. 68, pp. 102470, 2023.

[11] Q. Hu, B. Du, K. Markantonakis and G. P. Hancke, "A session hijacking attack against a deviceassisted physicallayer key agreement," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 1, pp. 691–702, 2020.

[12] P. N. Brown, H. P. Borowski and J. R. Marden, "Security against impersonation attacks in distributed systems," *IEEE Transactions on Control of Network Systems*, vol. 6, no. 1, pp. 440–450, 2019.

[13] S. Muneer, M. B. Alvi and A. Farrakh, "Cyber security event detection using machine learning technique," *International Journal of Computational and Innovative Sciences*, vol. 2, no. 2, pp. 42–46, 2023.

[14] S. S. Roy, A. Sinha, R. Roy, C. Barna and P. Samui, "Spam email detection using deep support vector machine, support vector machine and artificial neural network," in *Soft Computing Applications: Proc. of the 7th Int. Workshop Soft Computing Applications (SOFA 2016)*, Arad, Romania, pp. 162–174, 2016.

[15] I. Idrissi, M. Boukabous, M. Azizi, O. Moussaoui and H. E. Fadili, "Toward a deep learning-based intrusion detection system for IoT against botnet attacks," *IAES International Journal of Artificial Intelligence (IJ-AI)*, vol. 10, no. 1, pp. 110–120, 2021.

[16] M. K. Islam, M. A. Amin, M. R. Islam, M. N. I. Mahbub, M. I. H. Showrov *et al.,* "Spam-detection with comparative analysis and spamming words extractions," in *2021 9th Int. Conf. on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*, Noida, India, pp. 1–9, 2021.

[17] M. L. Mihailescu and S. L. Nita, "A searchable encryption scheme with biometric authentication and authorization for cloud environments," *Cryptography*, vol. 6, no. 1, pp. 8, 2022.

[18] V. Singh and S. K. Pandey, "Revisiting cloud security threats: Replay attack," in *2018 4th Int. Conf. on Computing Communication and Automation (ICCCA)*, Greater Noida, India, pp. 1–6, 2018.

[19] N. R. Tadapaneni, "Cloud computing security challenges," *International Journal of Innovations in Engineering Research and Technology*, vol. 7, pp. 1–6, 2020.

[20] A. Zaman, B. Safarinejadian and W. Birk, "Security analysis and fault detection against stealthy replay attacks," *International Journal of Control*, vol. 95, no. 6, pp. 1562–1575, 2020.

[21] P. Thirumavalavasethurayar and T. Ravi, "Implementation of replay attack in controller area network bus using universal verification methodology," in *2021 Int. Conf. on Artificial Intelligence and Smart Systems (ICAIS)*, Coimbatore, India, pp. 1142–1146, 2021.
[22] M. Nadeem, A. Arshad, S. Riaz, S. S. Band and A. Mosavi, "Intercept the cloud network from brute force and DDoS attacks via intrusion detection and prevention system," *IEEE Access*, vol. 9, pp. 152300–152309, 2021.
[23] A. Musa and A. Mahmood, "Client-side cryptography based security for cloud computing system," in *2021 Int. Conf. on Artificial Intelligence and Smart Systems (ICAIS)*, Coimbatore, India, pp. 594–600, 2021.