Implementation of Enhanced Data Integrity Algorithm in Cloud Computing

Moloko P. Mothlabeng, Topside E. Mathonsi, Daniel P. du Plessis, and Tonderai Muchenje
Tshwane University of Technology, Pretoria, South Africa
Email: pmothlabeng@yahoo.com, {Mathonsite, DuPlessisdp, Muchenjet}@tut.ac.za

***Abstract*—Cloud computing storage services may allow users to simply preserve and manage a huge amount of data at a minimal cost. However, it cannot ensure data integrity. Data is sent to remote cloud servers which might be unsecure and untrustworthy because unauthorized users or the service provider may inadvertently modify data, as a result, data may be lost or altered. Therefore, protecting data from hackers has become crucial to maintaining its integrity. To improve data integrity in cloud computing, this study presents an Implementation of Enhanced Data Integrity algorithm. The proposed algorithm is created by integrating Moving Target Defense (MTD) and New Lightweight Cryptographic Algorithm (NLCA). MTD is used by the EDIE algorithm to dynamically modify network configurations and regularly mislead attackers, reducing the number of attacks and improving data integrity. NLCA provides secure encryption and decryption of data with a faster execution time. In comparison to EDSA and SAKAS data integrity techniques, the results demonstrate that EDIE consumes 6% less energy, encounters only 0.2 percent of all launched man-in-the-middle assaults, and reduces server computing time by 21%.**

***Index Terms*—cloud computing, data integrity, Enhanced Data Integrity Encryption (EDIE) algorithm, Moving Target Defense (MTD)**

I. Introduction

Security flaws in cloud computing have caused users to lose trust in the service. Data integrity is a critical concern in cloud storage since cloud computing application software and databases are relocated to centralized huge data centers where data and service management may not be completely trustworthy [1]. Cloud computing is established as a standard for providing information technology services via the Internet, including hardware, software, and networking, with added benefits such as on-demand service and autonomous resource pooling [2]. The cloud is a centralized storage service that allows data to be stored, accessed, modified, and deleted from any location at any time [3]. The three service alternatives it offers in the IT business are Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS) [4].

This has sparked the attention of many users and organizations in using cloud computing services [5]. Data integrity guarantees that the information is of high quality, correct, and has not been tampered with. Users lose control over their data after storing it in the cloud, However, they continue to rely on the cloud for more reliable services, if their data will remain steady [6]. Cloud service providers are not always trustworthy because to reduce storage space or preserve fewer copies than necessary, they may delete data that is rarely accessed [7].

As a result, data owners must have faith that their information is being processed properly in the Cloud. Therefore, data integrity on untrusted servers is one of the most significant difficulties with cloud data storage, because data integrity affects the correctness of information stored in the system. The validity, quality, and security of data have an impact on the system's operations and desired outcomes in a cloud computing architecture. Integrity is concerned with the efficiency and performance of the program [8].

As a result, the Enhanced Data Integrity Encryption (EDIE) method was designed to ensure enhanced data integrity in cloud computing, where the user begins by attempting to connect to the network. The data owner delegated data administration to a trustworthy auditing server as the initial step. This is to verify that the data is correct, complete, and it is the most recent version uploaded to the cloud storage server. Follows step number two which is login phase, if the user is permitted, it is granted access. Otherwise, access is denied. Thereafter, the MTD phase will be processed to dynamically modify network configurations and regularly mislead attackers, reducing the number of attacks and improving data integrity.

The following are the contributions of this paper: 1) EDIE algorithm will enhance data integrity and reduce delay while increasing the execution speed by using NLCA because it provides a high security level and a significant enhancement in the decryption and/or encryption process, assuring high security while also maintaining low computing costs and consume less energy 2) EDIE algorithm will reduce data integrity attacks by using MTD to dynamically change network configurations and frequently confusing the attackers. 3) The existing solutions are the focus of the data integrity literature. Therefore, there is very little or no scholarly literature on the use of a NLCA integrated with the MTD to solve data integrity issues in cloud computing. As a

Manuscript received April 6, 2022; revised June 13, 2022; accepted June 20, 2022.

result, other academics will use this research to broaden their research into cloud data integrity. 4) By advancing research in the field of information systems and data integrity, this initiative will add to the body of knowledge in this field.

The following is the order in which this paper is organized: Section I, introduces Cloud computing and data integrity applications, as well as the study's goal, the problem it is seeking to resolve, and the paper's contributions. Section II highlighted the related work on cloud computing data integrity solutions. The flowchart of the proposed Enhanced Data Integrity Encryption Algorithm was shown and described in Section III. The proposed algorithm's implementation was discussed in Section IV of this study. The chosen simulation tool in this paper, NS-2, was discussed in Section V. In this section, the simulation results of the comparing algorithms SAKAS, EDSA, and EDIE were also presented. Finally, Section VI brings the paper to a close and makes recommendations for further research.

II. Literature Review

Cloud security continues to receive increased attention in the literature, particularly in terms of data integrity in the cloud. As a result, this section explores different existing solutions proposed to address data integrity issues, thereafter the paper presents the limitations of these existing solutions.

To ensure that attackers cannot plan and execute cyber-attacks, Ahalya *et al*. [9] introduced the MTD algorithm, which allows for dynamic security configurations in a network as well as dynamic encryption schemes. Attackers are unable to launch attacks because of the MTD system. MTD used an entropy-based method for attack detection in the network. According to the results of the experiments, the proposed technique can protect networks from cyber-attacks. The results were simulated using a custom Java simulation tool. The results showed that the achieved throughputs imply security even in the face of threats. The computational time spent on the server was not considered in this analysis. As a result, the number of attack attempts rises. However, the proposed approach uses NLCA to speed up server processing speed.

An NLCA was proposed by Thabit *et al*. [10]. It uses symmetric cryptography to encrypt data. The algorithm is a 128-bit (16 byte) block cipher that encrypts data with a 128-bit (16 byte) key. The encryption and decryption algorithms are both simple and secure. It was influenced by Feistel and SP architectural ideas for increasing encryption complexity. A variety of factors were evaluated to see how well DES, AES, HIGHT, Blowfish, and LED performed, including block size, key length, potential key, mathematical operations, cipher type, and security power. This is a strategy that is like the one we proposed in this study. Therefore, the NLCA is looking into Feistel and SP architectural solutions for reducing attacks and improving data integrity while also speeding up cipher execution.

Poorvadevi *et al*. [11] developed a technique in which a Third-Party Auditor (TPA) mechanism examines the confidentiality of data stored in the cloud on behalf of the data owner. TPA checks the integrity of the data by examining the hash and script. Integrity Verification is provided by the TPA, which saves the smartphone user a lot of time. The data owner has two keys under this scheme, one of which is exclusively accessible to him and is known as the private key, while the other is known as the public key. The message/file is encrypted twice using the owner's private key and TPA's public key. As a result, the information of smartphone users is kept private. The suggested approach uses the RSA algorithm to encrypt and decrypt messages, resulting in message authentication. The hash function of the message is also determined here to ensure data integrity. However, because RSA is computationally costly and slower, performance was hampered.

The AES (Advanced Encryption Standard) scheme proposed by Sajay *et al*. [12] operates consistently well in both hardware and software platforms in a wide range of environments, the key establishment time is rapid, and the key agility is good. Its implementation requires less memory, making it suited for circumstances when space is constrained. The structure stands a good chance of exploiting instruction-level parallelism. Even after several test cases, any block and key sizes that are multiples of 32 are supported (higher than 128-bits), and statistical analysis of the encrypted text was not possible. However, verifying the algorithm's strength in terms of assaults was not considered, but our proposed approach will frequently confound attackers by employing MTD defense.

To resolve the security issues found in prior versions schemes, Odelu *et al*. [13] suggested a Secure Authenticated Key Agreement Scheme (SAKAS). The author developed this approach after several previous systems were vulnerable to server impersonation attacks and failed to achieve secure mutual authentication. Most existing systems, they stated, couldn't provide session-key security (Security) and robust user credential privacy if session-specific secret information was mistakenly disclosed to an attacker. They tested their approach with NS-2 and claimed that it achieved SKsecurity and strong credential privacy. Even if session-specific temporary information was mistakenly given to an adversary due to a session exposure attack, they claimed that their system maintained secure mutual authentication and good user credential privacy. However, it was discovered that it would not be able to improve privacy preservation and efficiency. Furthermore, SAKAS ensured the integrity and security of data stored in the cloud. As a result, our research bridges the gap by using NLCA to ensure data integrity in cloud computing environments.

Kavin and Ganapathy [14] proposed presented the Enhanced Digital Signature Algorithm (EDSA) to assure data integrity in cloud storage systems. This strategy was primarily developed to ensure that data is stored securely in the cloud and is available at any time and from any

location. The suggested EDSA was created utilizing Elliptic Curves that were generated because of algorithm enhancements. The improved equation proposed in their work is used to construct two elliptic curves in the EDSA. These elliptic curve points were used as a public key to execute the signing and verification steps to assure data integrity. The algorithm compares the signing and verification processes' consequent values and guarantees that they are correct. However, to improve privacy and efficiency, more research might be done by introducing new digital signature algorithms. Furthermore, the issue of mobile device energy usage is a major source of concern.

The Dynamic Auditing Protocol scheme was presented by Mishra *et al*. [15] to improve data integrity and enable complicated data operations on cloud servers. The linear combination of data blocks must be sent to the auditor in this manner for verification. To assure data integrity, the plan employs a Third-Party Auditor (TPA). It also simplifies data dynamics by allowing the execution of the most basic data operations such as block modification, insertion, and deletion. The method could reveal data content to the auditor, according to the author, because it requires the server to present linear combinations of data blocks to the auditor for verification, and the scheme's performance is unknown.

Cryptographic accumulator proven data possession is a robust deterministic data integrity verification scheme proposed by Ren *et al*. [16]. The basic design of the CAPDP is based on an upgraded RSA-based cryptographic accumulator, which provides the following advantages: 1) It allows the data owner to run as many data integrity checks as they want; 2) It aids data flow; 3) It is cost-effective in terms of transmission, compute, and storage for both the data owner and the cloud storage provider; 4) The proposed scheme's verification operation is unaffected by the number of blocks being verified; 5) It makes the vetting process easier and less expensive. The approach, according to the author, has to be optimized because it does not prevent replay attacks. This study, however, supports the authors' proposed strategy because it will help to lower the load and cost of data integrity verification in the cloud, but it does not address the issue of replay assaults. The technique also adds a considerable performance overhead to the authorized dictionary, making it too sluggish to employ in practice. Critical infrastructure industries that deal with very sensitive data deem this method to be insufficient (critical data). They used MATLAB to simulate their results.

Cash *et al*. [17] proposed a PoR (proof of retrievability) method that uses spot-checking and error-correcting codes to ensure that data files on archive service systems are both “possession” and “retrievable”. For detection reasons, some special blocks known as “sentinels” are randomly implanted in data files, and the file is encrypted to protect the positions of these special blocks. However, it only works with static data sets and only allows a limited number of queries as a challenge; it also only manages a limited number of check blocks; and it does not safeguard files stored on the cloud provider's servers.

According to a review of the literature, previous research did not propose any method that may successfully enhance data integrity in cloud computing considering performance in terms of the number of attacks, Server computing time, and energy consumption. To address the identified gaps, this paper proposed an Enhanced data integrity algorithm in cloud computing. The proposed algorithm was designed by integrating NLCA and MTD. MTD is used by the proposed EDIE algorithm to dynamically modify network configurations and regularly mislead attackers, reducing the number of attacks and improving data integrity.

III. Enhanced Data Integrity Encryption Algorithm Design

The EDIE algorithm was designed by merging NLCA and MTD to ensure that users' private data is secured in Cloud computing by ensuring data integrity. The suggested EDIE algorithm is set up so that the data owner starts the process by pre-processing the data before sending it to the cloud. MTD is to ensure that the integrity of data is maintained by frequently observing the attackers and changing the system configurations to curb out the issue of those attacks. Encryption process where keys are generated for each block of the file and all the hashes get concatenated. The audit server then requests the encrypted data from the cloud storage for verification purposes. The encryption key sent by the auditor is stored for verification matching, then the encryption key is verified by being matched to the initial key generated, if the key matches, then data is assumed not to have been tampered with.

Based on the literature review, data could be manipulated by unauthorized users or the service provider inadvertently, resulting in data loss or modification. This is not the case with the proposed technique because the Moving Target Defense will secure the user’s data by preventing attacks.

A. New Lightweight Cryptographic Algorithm

The NLCA achieves a higher execution rate by reducing the number of encryption rounds, which reduces the number of attacks and improves data integrity. The proposed EDIE algorithm is intended to improve data quality by enhancing data integrity in cloud computing during data transfer. The matrix and f-function extensions of keys, rather than a single extension key, are among the advantages of NLCA.

TABLE I. SUMMARY OF THE RELATED WORK

Data Integrity Scheme	Advantages	Limitations/gaps
Moving Target Defence (MTD)	Dynamically modify network configurations and regularly mislead attackers, reducing the number of attacks and improving data integrity. imply security even in the face of threats	Server processing speed was not considered. However, the proposed approach uses NLCA to speed up server processing speed.
Dynamic Auditing Protocol	Offers fully dynamic operation Enabled complex data operations	Because the scheme requires the server to communicate linear combinations of data blocks to the auditor for verification, the scheme may expose data content to the auditor. Proposed auditing protocol is not efficient, due to computational overheads to the third-party auditor. They simulated their results using MATLAB.
New Lightweight Cryptographic Algorithm (NLCA)	Reduce attacks and improve data integrity while speeding up cipher execution.	Requires enhanced defence of attacks
Cryptographic accumulator provable data possession	Support public verification and strong proof of data integrity Unlimited number of data integrity checks. It helps with data flow cost-effective in terms of transmission, compute, and storage The proposed scheme's verification operation is unaffected by the number of blocks being verified It makes the vetting process easier and less expensive.	The scheme also introduces a significant performance overhead that makes the authenticated dictionary too slow for practical use. This scheme is considered inadequate by critical infrastructure sectors that involve highly sensitive data (critical data). They simulated their results using NS-2 simulator. doesn't prevent replay attacks
Secure Authenticated Key Agreement Scheme (SAKAS).	Offer session-key security and robust user credential privacy if session-specific secret information was accidentally given to an adversary.	It was discovered that it would not be able to improve privacy preservation and efficiency.
Third-Party Auditor (TPA) scheme: The proposed method encrypts and decrypts messages using the RSA algorithm	Provides integrity verification and reduce the load from the data owner Uses two encryption keys. Provides authentication structure.	However, the performance was affected due to RSA being computationally intensive and slower. Online burden to the cloud users and cloud servers. Data content may be exposed to TPA during the auditing process
Enhanced Digital Signature Algorithm (EDSA)	Ensure that data is stored securely Uses elliptic curve points as a public key to execute the signing and verification processes.	However, more studies could be done by introducing new digital signature algorithms to improve privacy and efficiency. Furthermore, the issue of mobile device energy usage is a major source of concern.
Efficient and scalable distributed Scheme	Support full dynamic operations The proposed scheme attempts to generate a proof without requiring the server to view or download the entire file from the server	A client is only allowed to perform a certain number of updates and challenges. Only append type insertions are permitted; block insertions are not permitted. Large files provide a problem because each update necessitates re-creating all tasks.
AES (Advanced Encryption standard) scheme	Its implementation takes less memory Quick key setup time	No differential or linear cryptanalysis attacks have yet been proven. The AES cipher, which has a secure of 128 bits, not be suitable for big data applications or other modern huge applications, such as secure cloud storage. As a result, these applications with vast amounts of data may require a larger algorithm with a higher order of mathematical and structural foundations, as well as a trade-off between speed and accuracy.
POR Schemes	Easy to implement	It only works with static data sets and only supports a small number of queries as a challenge. It also only handles a finite number of check blocks. It does not protect files stored on the cloud provider's servers.

Meanwhile, it offers a quick key generation procedure that helps to prevent brute-force attacks. The process of ensuring data integrity through key generation, encryption, and decryption is discussed as follows:

Step 1: Generating the encryption key, the 128-bit encryption key (Kc) is divided into dual sections: 64 bits on each side, left and right.

Step 2: Both left and right 64-bits are then divided into 4-bit segments. Thereafter, in every 4 bits, a shift row is created, and the output of shifting is fed to the f-function.

Step 3: The f-function has four segments, each of which is four bits long (16 bit). As a result, substitution may be used to create cipher key (Kc) using the f-function, as shown in (1) and (2):

$$\mathrm{Kb_i f} = ||_{\mathrm{j-1}}^{5} \mathrm{k_{c4(j-1)+i}} \quad (1)$$

where $i = 5$;

$$\mathrm{Ka_1 f} = \mathrm{f(b_i f)} \quad (2)$$

An XOR operation is performed among the four-round keys to generate the fifth key as shown in (3):

$$kkk = \overset{5}{i=1} \oplus k_i \quad (3)$$

Furthermore, two keys would be produced in this procedure based on the seed given by the user. The same steps for the rounds are repeated to improve execution time because of the limited number of rounds as shown in (4):

$$Ro_{i,j} = \begin{cases} Px_{i,j} \oplus ; j = 1..4 \\ Px_{i,j+1} \oplus Ef_{li} ; j = 2 \\ Px_{i,j-1} \oplus Ef_{ri} ; j = 3 \end{cases} \quad (4)$$

The encoded text is then obtained using equation (5):

$$Ct = R_{51} \# R_{52} \# R_{53} R_{54} \quad (5)$$

As mentioned earlier, the MTD is used to dynamically change network configurations and frequently confuse the attackers. MTD uses a mathematical technique called The Sibson entropy to understand the strategy of the attacker as shown in (6) and (7):

$$Pij(\ mEI) = mEIk(\sum_{k=1}^{N\ fail} mEIk) - 1 \quad (6)$$

$$Ds(Pt - 1\ Src(\ mEI)\ , Pt\ Src(\ mEI)) \quad (7)$$

When the attacker's technique is non-reconnaissance, several reconnaissance approaches such as blind and half-blind can be used to safeguard systems. The attacker's plan is understood, and the full network space is modeled. This also ensures that the attackers, on the other hand, are unable to carry out attacks with MTD in place. It is feasible to learn about the attacker's plans using the entropy method as shown in (8) and (9):

$$= \frac{1}{2}\{D_i[P^{src}\ t - 1(mEI), P^{src}] + D_i[P^{src}\ t - \ 1(mEI), P^{src}]\} \quad (8)$$

$$\left((\frac{Ni\ fail - Nfail/mBmL}{(mBmL)2/12}\right) < \delta chauvenet \quad (9)$$

The penetrating process in the proposed system is shown in (10):

$$Ds([P^{Dst}t(mEI), \frac{Nfail}{nBmL} = \frac{1}{2}\{D[P^{Dst}t(mEI), [P^{Dst}t] + D[\frac{Nfail}{mBmL}, [P^{Dst}t]\} \quad (10)$$

Equations (11) and (12) are used to compute the probability of malicious investigation:

$$wmR, mL\ = \ CmR, mL, SmR, mL \quad (11)$$

$$SmR, mL = 1 - f\ (\ P^a j, Pj) \quad (12)$$

To achieve an ideal mutation period, the proposed system used a smoothing coefficient of 0.75. It can be seen in (13) and (14):

$$T\ _{EMP}^{t+1} = \max\ [\alpha T\ _{EMP}^{\ t} \frac{d}{n^t fail} + (1-\alpha)\ T\ _{EMP}^{\ t}, 2RTT] \quad (13)$$

$$T\ _{EMP}^{t+1} = T\ _{EMP}^{\ t} + t'\ d \quad (14)$$

B. *Enhanced Data Integrity Encryption Algorithm Flowchart*

The flowchart demonstrates the logic for processing all the algorithm phases in the EDIE algorithm as shown in Fig. 1.

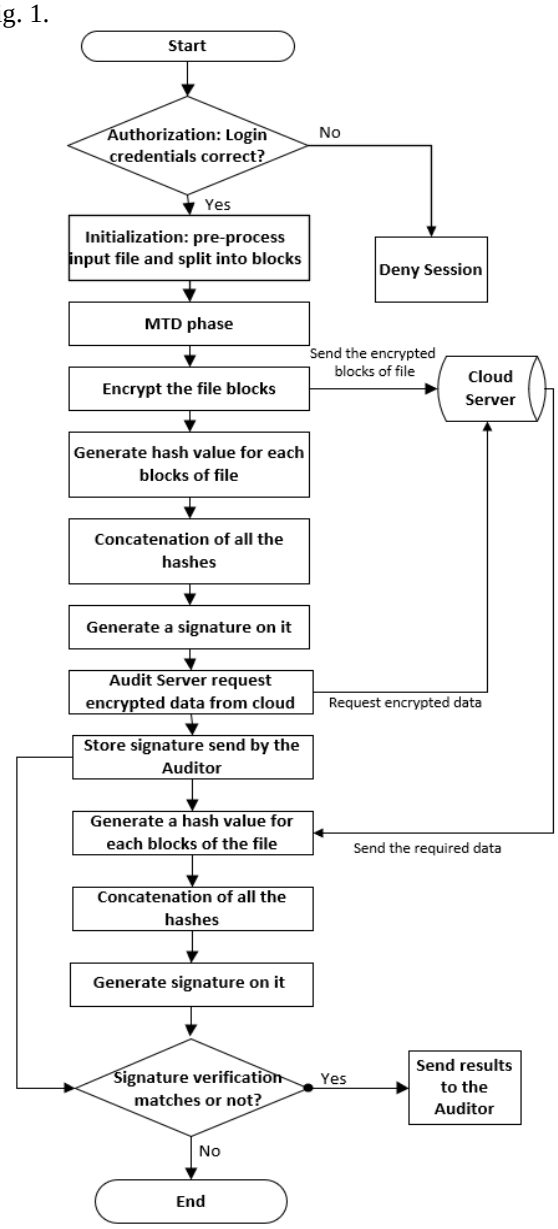

Figure 1. Enhances data integrity encryption algorithm flowchart.

NLCA and MTD were integrated to lower computation time, consume less energy, and lower the level of attacks. According to Thabit et al. [13] Within cloud computing environments, the NLCA algorithm achieves quick data processing efficiency and an acceptable level of security. Furthermore, the NLCA helps to increase the confidentiality and integrity of cloud-stored data while also making it available on demand. The reduced SCT further helps in preventing man-in-the-middle attacks. However, SCT increases with the file size increment because huge files must be computed by the cloud servers. MTD algorithm allows for dynamic security setups in a network, as well as dynamic encryption schemes, to ensure that adversaries

cannot plan and execute cyber-attacks. The MTD system ensures that attackers are unable to initiate attacks.

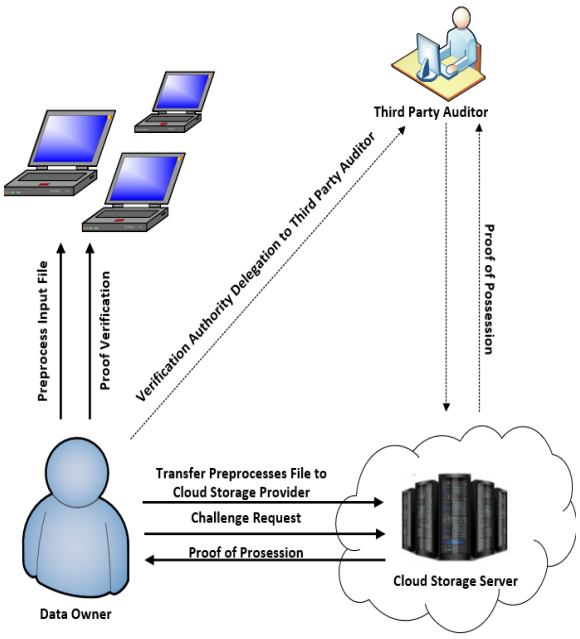

Figure 2. EDIE system architecture design.

The flow of data from the data owner to a cloud storage server and the other way around is shown in Fig. 2.

The first step is that the data owner delegates the data administration to a trusted auditing server. This is to verify that the data is correct, complete, and it is the most recent version uploaded to the cloud storage server.

The second step is where data owner can then send the encrypted data to the cloud storage server, where he or she can edit, amend, or even delete data.

The third step (pre-processing phase) ensures that the data is pre-processed before uploading to cloud storage servers, and some additional metadata is generated.

The fourth step (verification phase) permits auditors to send a challenge request to the cloud storage server, which generates evidence of possession using the data and metadata and receives a response. Following that, the auditor verifies the proof to confirm that the data integrity has not been compromised.

The fifth step (encryption/decryption) allows the encryption/decryption of the data using a symmetric key known only to the receiver to access and protect the user's data. This uses a limited number of encryption or decryption rounds.

The sixth step (cloud storage server) ensures that the user's encrypted data is securely stored in the cloud, ready to be utilized and retrieved at any time and from any location. In this step, the MTD is employed to change the configurations dynamically without disrupting the system, which aids in reducing the risk of data integrity threats.

C. *The Developed Enhanced Data Integrity Encryption (EDIE) Algorithm*

The algorithm phases are carried out, one after the other, to ensure that users' sensitive data is safeguarded, and their integrity is preserved. The session was rejected if the server's response did not match, and the user was instructed to try again. Below is the proposed EDIE designed by integrating NLCA and MTD to enhance data integrity in cloud computing.

Algorithm 1. Enhanced Data Integrity Encryption Algorithm

//Initialization: Pre-processing data before uploading to storage
//Let $G||$ be the storage or cloud,$//||N||$ the total number of servers, //Input $n\ and\ I$ for server and counter, $//G = \{ni, ni = 1, \ldots., (ni - n - 1)\}$,
1. $if(n <= G)\&\&(n == G)$
2. For$(i = 0; i <= n; i++)$
3. $S = i;$
4. $S++;$
5. Else if $(n == R)$ Or $(i = 0; i <= n; i++)$
6. $n++;$

$//Encryption\ with\ MTD$
Input: Secret message(plaintext) denoted s
Output: Encrypted message s'
7. Initialize binary row vector V
8. Initialize low dimension binary matrix M
9. $IV = ConvertIntoBinaryRowVector(s)$
10. $V' = NLCAEncryption(V)$ //Intermediate step 1 using eq (1) to (5)
11. $M' = OuterLayerEncryption(V')$ // Intermediate step 2
12. $C = GenerateFinalCipherText(M)$
13. $C' = DymanicUpdate(c)$
14. Return C //Store encrypted data(c) to the cloud server
15. End //Audit server requests data from cloud storage for verification

//Cloud server level: cloud server sends requested data to the audit server
16. $//G = \{ni, ni + 1, \ldots., (ni - n - 1)\}$
17. $if(n <= G)\&\&(n == G)\{$
18. For$(i = 0; i <= n; i++)$
19. $S = i;$
20. $S++;\ \}$
21. Else if $(n == R)\{$for $(i = 0; i <= n; i++)$
22. $\{16{:}\ n++;\ 17{:}\ \}\ \}$
23. End

$//Encryption\ with\ MTD$
24. V is a binary row vector that needs to be initialized V
25. Initialize low dimension binary matrix M
26. $IV = ConvertIntoBinaryRowVector(s)$
27. $V' = NLCAEncryption(V)$ //Intermediate step 1
28. $M' = OuterLayerEncryption(V')$ // Intermediate step 2
29. $C = GenerateFinalCipherText(M)$
30. $C' = DymanicUpdate(c)$
31. $if\ (Key_{initial==}Key_{final})\{$

//Send the results to the audit server
//For comparing both user and cloud level data, we establish a Boolean value for true and false conditions. Then,
32. Verify (Zi = {G1i(true)G2i(false)}) Check for condition
33. If the condition, G1i = G2i then it is true in the sense that
34. N == nu\ it is set that Z={s(true)} \ n! =nu\
35. Z == {s(false)}}
36. End

The NLCA and MTD ensure low computing costs, low energy consumption, and low attack levels, making the algorithm more efficient and faster while transmitting data on the cloud. Before data is uploaded or downloaded from cloud storage, all phases of the NLCA and MTD will be executed one after the other to enhance their effectiveness.

IV. The Implementation of the Proposed Algorithm

This section explains the implementation of the proposed EDIE algorithm. The algorithm was designed using the NS2 simulator version 2.35 and implemented

using the IEEE 802.11 model and was executed on Linux Ubuntu 16.04.5 LTS. Oracle Corporation has been developing and maintaining Oracle VM VirtualBox Manager since 2014. This Linux OS operates on the most recent version of Oracle VM VirtualBox Manager. OTcl code was used to configure the network topology, while C++ was used to implement the EDIE algorithm. To achieve high-quality and consistent results, the simulations were run multiple times.

When developing and testing systems, the simulator becomes useful. As shown in Fig. 3, NS-2 offers a variety of technical computing applications to scientists, engineers, researchers, and educators.

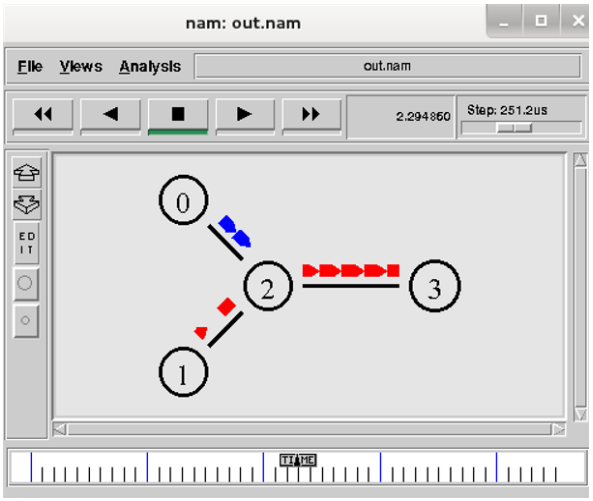

Figure 3. Interface for the NS-2 simulator.

D. Simulation Set-up

The main purpose of modeling the EDIE algorithm was to see if the suggested algorithm improves data integrity in cloud computing. NS-2 is a straightforward event-driven simulation tool that uses software to separate data processing from control.

The advantage of using NS-2 is that it can automatically record statistics such as packet transmission loss, energy consumption, and more. This research study chose NS-2 for the simulation technique for cloud network topology. It offers the required Graphical User Interface (GUI) which provides simplicity when simulating the proposed solution. NS-2 supports IEEE 802.11ac protocol which permits the user to simulate transmission between many Wireless Local Area Network (WLAN) connections using the MAC and physical layers (PHY). NS-2 is more suitable for analyzing and researching the dynamic nature of data integrity communication networks. The constructed simulation topology of NS-2 is demonstrated in Fig. 4.

We employed the Ad-hoc On-Demand Distance Vector (AODV) routing protocol because it permits for dynamic, self-starting routing between communicating MNs on a wireless network when they need to establish and maintain connections. Furthermore, AODV supports the transfer of both unicast and multicast routing packets. AODV builds routes between source nodes and other MNs on a wireless network when they need to establish and maintain connections.

The author of this paper chosen the IEEE 802.11ac protocol because it allows users to model communication between many Wireless Local Area Network (WLAN) nodes using the medium access control (MAC) and physical layer (PHY). 802.11ac provides a 5 GHz bandwidth of up to 1300 Mbps and a 2.4 GHz bandwidth of 450 Mbps and is backward compatible with 802.11b/g/n. Fig. 4 depicts the network topology created for the experimental evaluation of the suggested algorithm.

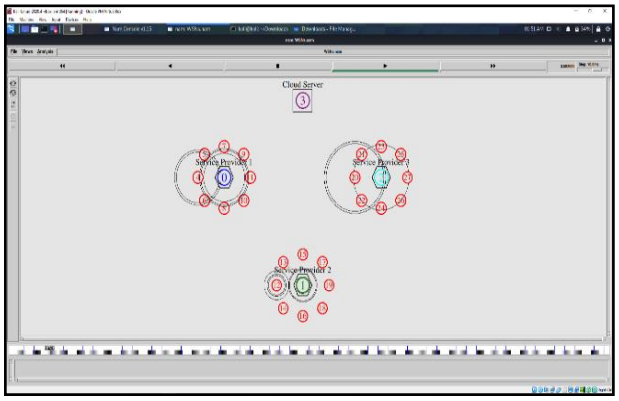

Figure 4. The NS-2 configuration parameters.

V. Simulation Results and Analyses

A simulation is useful because it allows the user to see how efficient the proposed method is before putting it into practice in a real network setting. Researchers can test the suggested algorithm's performance under various network conditions using a network simulator. The simulator can be adjusted to produce more accurate findings for the analysis. This section summarizes and analyzes the results of the simulation used to assess the efficiency of the EDIE algorithm in a range of network configurations.

A. Experimental Evaluation

Experiments were presented to assess the effectiveness of the EDIE algorithm. NS-2 simulator version 2.35 was used and executed on Linux Ubuntu 16.04.5 LTS. This Unix OS runs on the latest version of Oracle VM VirtualBox Manager developed and currently maintained by Oracle Corporation since 2014. The network topology was configured using OTcl code, meanwhile, the EDIE algorithm was implemented using C++. The simulations were performed several times to ensure quality and reliable results. The benefit of using NS-2 is that it can record the data such as packet transmissions, loss, and more automatically. As previously discussed, R Programming was employed to nicely display the analyzed results graphically. Meanwhile, AWK scripts were created to deal with the statistical analysis of the collected data.

B. Simulation Results

To evaluate the EDIE's performance, a simulation setup was utilized to run the simulations, as described in Section IV. The planned EDIE was compared to SAKAS and EDSA using the following performance metrics:

1) Server Computing Time – the measurement of the amount of time it takes the cloud servers to compute the assigned process within the network.
2) Energy Consumption – the measurement of the amount of energy consumed during data transmission within the network.
3) Number of Attacks – the measurement of the number of attacks that were performed successfully with the network.

In this study, the proposed algorithm's performance was compared to that of the best two identified algorithms which is EDSA and SAKAS data integrity techniques.

An EDSA was chosen because it uses a public key to perform the signing and verification processes to ensure data integrity. The algorithm examines the document's uniqueness by comparing the final values of the signing and verification operations. The results of the experimental assessments showed that the suggested EDSA is efficient in terms of key generation time, signing time, and verification time, making it similar to our proposed EDIE. The scholars did say, however, that more study could be done by introducing new digital signature algorithms to improve privacy and efficiency. In addition, considering the issue of energy consumption for mobile devices is also of great concern.

SAKAS was chosen because it provided safe mutual authentication and robust user credential privacy, allowing it to withstand attacks even if session-specific temporary information was unexpectedly given to an adversary because of a session exposure attack. However, it was realized that it cannot enhance privacy preservation and efficiency. Furthermore, the integrity and security of data saved in the cloud were not a problem with SAKAS. Consequently, the proposed algorithm closes that gap by employing NLCA to assure data integrity within cloud computing environments.

1) Server computing time

Server Computing Time (SCT) is the length of time required by the server to perform computational processes. SCT of the proposed algorithm under various network scenarios is given in Fig. 5. The experiment evaluations show that EDIE demonstrated a promising reduced server computing time. This further provides improvements during encryption and/or decryption of the computed data and, thus, providing improved security in terms of data integrity, and low computation cost. As a result, the processing time calculated from the encryption and decryption processes of the proposed algorithm seems to be faster compared to traditional related algorithms. Within cloud computing environments, the NLCA algorithm achieves quick data processing efficiency and an acceptable level of security. Furthermore, the NLCA aids in the improvement of the secrecy and integrity of cloud-saved data while also making it accessible on demand. The reduced SCT further helps in preventing man-in-the-middle attacks. However, SCT increases with the file size increment because huge files must be computed by the cloud servers, see Fig. 5.

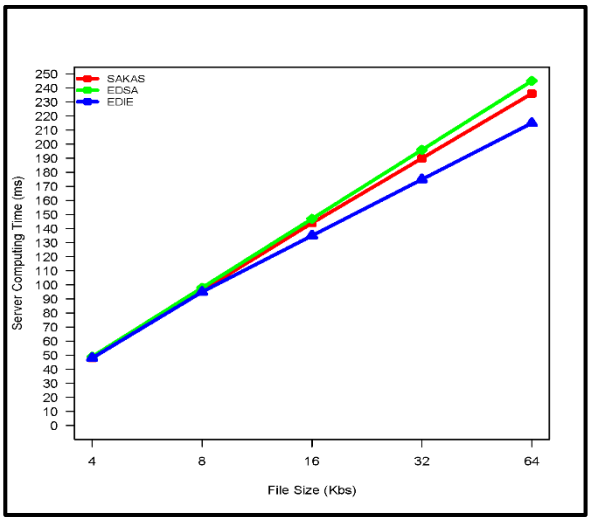

Figure 5. Evaluation of the server computation time performance on the three algorithms.

2) Energy consumption

The total energy consumed by the network to accomplish transmission, reception, and data aggregation is known as Energy Consumption (EC). The EC of the three algorithms has been evaluated and assessed in various network circumstances. We assessed the average energy consumption for all configured cloud servers using the proposed EDIE algorithm. The EDIE algorithm has shown a low energy consumption of up to 6%, whereas SAKAS and EDSA both yield 8% and 9%, respectively, according to the testing results. Integration of NLCA gave a high level of security and a considerable increase during encryption and/or decryption, as indicated in the previous section, assuring high security while also maintaining low computing costs in terms of energy, see Fig. 6 below:

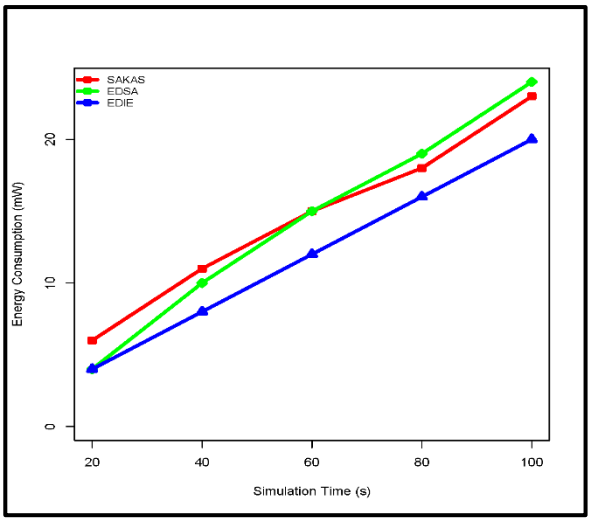

Figure 6. Average energy consumption.

3) Number of attacks

A Network Attack (NA) is an attempt to obtain unauthorized access to a network with the intent of stealing data or engaging in other destructive behavior. Fig. 6 shows the NA of the three schemes observed and

evaluated under various network circumstances. The EDSA and SAKAS algorithms are similar in that they both experience higher attack levels of up to 1.5 percent, where it is clear from the simulation that the EDIE algorithm reached a maximum of 0.2 percent level of attacks and thus offered improvements in terms of data integrity maintenance and assurance. As a result, the proposed method appears to be more effective for cloud computing in terms of data collecting and processing speed see Fig. 7 below:

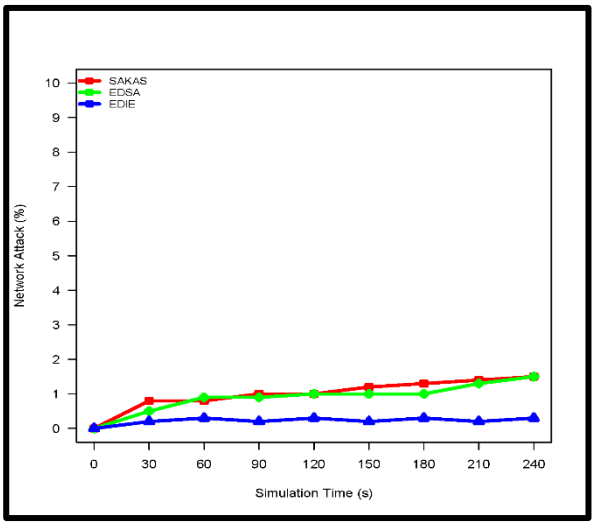

Figure 7. Evaluation of the number of attacks (%).

Based on all the performance matrices analyzed in the three identified algorithms of this study, the proposed EDIE algorithm showed that it outperformed the EDSA and SAKAS algorithms.

Furthermore, the evaluations have shown that EDIE algorithm has a low energy consumption of up to 6%, but SAKAS and EDSA both yield 8 and 9 percent, respectively. As mentioned in the preceding section, NLCA integration provided a high level of security and a significant rise during encryption and/or decryption, ensuring strong security while retaining low computational costs in terms of energy which contributes to the enhancement of data integrity because the higher the level of security, the less the number of attacks.

Attackers could utilize network vulnerabilities to obtain access to your company's data by exploiting flaws or defects in hardware or software. When a cybercriminal has the potential to penetrate your network security, it automatically becomes a data integrity threat. The evaluations have shown that SAKAS and EDSA, on the other hand, both have an increase in attacks of up to 1.5 percent. EDIE only experience a low percent increase of up to 0.2% due to the use of MTD that keeps on changing the network configurations to confuse the attackers, which enhance the data integrity because the less the number of attacks the higher the improvement of data integrity.

Finally, SAKAS has a 21 percent increase in server computing time, while EDSA has a 30 percent increase in server computing time when compared to EDIE. Employing NLCA algorithm achieves a high level of data processing efficiency while also maintaining a reasonable level of security. Furthermore, the NLCA aids in the improvement of cloud-saved data's secrecy and integrity while also making it available on demand. The less the server computing time, the lower the latency and energy consumption, that does not give the attackers enough time to steal the network configurations to build the attacks against the system. Additionally, by employing MTD, the network configurations will keep on changing to confuse the attackers and leads improving data integrity. The evaluations are shown in Fig. 5, Fig. 6, and Fig. 7.

VI. Conclusion

Data integrity is one of the most challenging and crucial security concerns in the cloud computing era. Given the importance of data integrity, this research looked at and analyzed various existing data integrity solutions, as well as their benefits and drawbacks. The EDIE algorithm was designed by combining the NLCA and the MTD in this work. MTD is used by the EDIE algorithm to dynamically modify network configurations and regularly mislead attackers, reducing the number of attacks and enhancing data integrity. Using the NS-2 simulation tool, the performance of the EDIE method was compared to that of the EDSA and SAKAS procedures. The results of the analysis were provided based on the amount of time it took the server to compute, the amount of energy it consumed, and the number of attacks it received. The results showed that the proposed algorithm performed better than EDSA and SAKAS schemes in all performance metrics evaluated. This is because the suggested EDIE algorithm uses less energy, reduces server computing time, and reduces attack levels while improving data integrity.

We intend to assess the EDIE algorithm's security on the Internet of Things (IoT) environment in the future to ensure flawless security.

Appendix A Table of Notations

Symbol	***Description***
$\otimes$	XOR Operation
$\odot$	XNOR Operation
$\|\|$	Concatenation
s'	Encrypted message
c'	Dynamic Update
Z	Boolean value
N	The total number of servers
IV	Onvert into binary row vector
M'	Outer layer Encryption
Zi	User identity
M	Initial low dimension binary matrix
V	Initial binary row vector V
$n\ and\ i$	For server and counter
G	Cloud/Storage
R	Formation while verifying for integrity
a	Data counting for n server

CONFLICT OF INTEREST

The authors state that there are no conflicts of interest in the publication of this research.

AUTHOR CONTRIBUTIONS

Moloko P. Mothlabeng carried out the research, have made substantial contributions to the design of the algorithm, simulated, and evaluated the results, wrote the paper. Topside E. Mathonsi have been involved in drafting the manuscript or revising it critically for important intellectual content. Tonderai Muchenje have contributed to the manuscript's creation and critical revision for key intellectual substance. Deon P Duplessis have contributed to proofreading and language editing. All authors approved the final version.

ACKNOWLEDGMENT

The authors appreciate the financial assistance provided by Tshwane University of Technology.

REFERENCES

[1] M. K. Walia, M. N. Halgamuge, N. D. Hettikankanamage, *et al.*, "Cloud computing security issues of sensitive data," *IGI Global*, pp. 60-84, March 2019.
[2] D. Marinescu, *Cloud Computing Theory and Practice*, 3rd ed., Morgan Kaufmann, 2022, ch. 1, pp. 1-10.
[3] K. Zkik, G. Orhanou, and S. E. Hajji, "Secure mobile multi cloud architecture for authentication and data storage," *International Journal of Cloud Applications and Computing*, vol. 7, no. 2, pp. 62-76, April 2017.
[4] M. S. Giri, B. Gaur, and D. Tomar, "A survey on data integrity techniques in cloud computing," *International Journal of Computer Applications*, pp. 122-135, July 2015.
[5] Z. Asadi, M. Abdekhoda, and H. Nadrian, "Cloud computing services adoption among higher education faculties: Development of a standardized questionnaire," *Education and Information Technologies*, vol. 1, no. 25, pp. 175-191, July 2020.
[6] M. S. Carmona, "Is biometric technology in social protection programmes illegal or arbitrary? An analysis of privacy and data protection," *An Analysis of Privacy and Data Protection ESS-Working Paper*, pp. 1-43, 2018.
[7] L. Zhou, F. Anmin, S. Yui, *et al.*, "Data integrity verification of the outsorced bug data in the cloud environment," *Journal of Network and Computer Applications*, vol. 122, pp. 1-15, November 2018.
[8] S. Juare, *Survey on Data Security and Integrity Issues in Cloud Computing*, IOP Publishing, 2019, pp. 440-445.
[9] M. Ahalya, J. Renuka, N. R. Srinivas, *et al.*, "Moving target defence framework using dynamic encryption scheme," *Studia Rosenthaliana Journal for the Study of Research*, vol. 12, no. 5, pp. 75-87, May 2020.
[10] F. Thabit, S. Alhomdy, H. A. Ahdal, *et al.*, "A new lightweight cryptographic algorithm for enhancing data security in cloud computing," *Global Transitions Proceedings*, vol. 2, no. 1, pp. 91-99, January 2021.
[11] R. Poorvadevi, T. Mannuru, and R. Narala, "Enhancing distributed data integrity verification scheme in cloud environment using machine learning approach," in *Proc. 6th International Conference on Trends in Electronics and Informatics*, 2022, pp. 863-867.
[12] K. Sajay, S. S. Babu, and Y. Vijayalakshmi, "Enhancing the security of cloud data using hybrid encryption algorithm," *Journal of Ambient Intelligence and Humanized Computing*, pp. 1-10, July 2019.
[13] V. Odelu, A. K. Das, S. Kumari, *et al.*, "Provably secure authenticated key agreement scheme for distributed mobile cloud computing services," *Future Generation Computer Systems*, vol. 68, pp. 74-88, March 2017.
[14] B. P. Kavin and S. Ganapathy, "A new digital signature algorithm for ensuring the data integrity in cloud using elliptic curves," *The International Arab Journal of Information Technology*, vol. 18, no. 2, pp. 180-190, March 2021.
[15] R. Mishra, D. Ramesh, and D. R. Edla, "Dynamic large branching hash tree based secure and efficient dynamic auditing protocol for cloud environment," *Cluster Comput.*, vol. 24, pp. 1361-1379, 2021.
[16] Y. Ren, X. Liu, Q. Wu, *et al.*, "Cryptographic accumulator and its application," *Security and Communication Networks*, pp. 54-195, March 2022.
[17] D. Cash, A. Küpçü, and D. Wichs, "Dynamic proofs of retrievability via oblivious RAM," *Journal of Cryptology*, vol. 30, no. 1, pp. 22-57, September 2017.

Moloko P. Mothlabeng received her B-Tech Information Technology in 2017, from Tshwane University of Technology, Pretoria, South Africa. She is currently a Systems Administrator at Financial Intelligence Centre, South Africa. Her research interests include Cloud Computing, Data Integrity and IoT.

Topside E. Mathonsi received his B-Tech in 2013, M-Tech in 2015 and DComp in 2020 from the Tshwane University of Technology, Pretoria, South Africa. He is currently the Senior Lecturer at Department of Information Technology, Tshwane University of Technology, South Africa. His research interests include 5G, IoT/IoE, Cybersecurity, and AI.

Daniel P. du Plessis received his BML in 2002 from University of the Free State (South Africa), MIT in 2005 from University of Pretoria (South Africa) and PhD (IT) from North-West University (South Africa) He is currently the Senior Lecturer and Assistant Dean Teaching and Learning at Information and Communication Technology (ICT) Faculty, Tshwane University of Technology, South Africa. His research interests include Wireless Networks, IOT and Cyber Security.

Tonderai Muchenje received his Master of science in 2008 from University Fort Hare, master's in computer Auditing in 2012 from University of Johannesburg (South Africa) and PhD (IT) from Nelson Mandela Metropolitan University (South Africa). He is currently the Senior Lecturer at Department of Information Technology, Tshwane University of Technology, South Africa. His research interests include Internal controls, IOT, wireless Networks and Cyber Security.