Cryptographic Solutions for Data Security in Cloud Computing: A Run Time Trend-based Comparison of NCS, ERSA, and EHS

John Kwao Dawson[1,*], Frimpong Twum[2], James Benjamin Hayfron Acquah[3] and Yaw Marfo Missah[4]

[1]*Computer Science Department, Sunyani Technical University, Sunyani, Ghana*
[2]*Computer Science Department, Kwame Nkrumah University of Science and Technology, Kumasi, Ghana*
E-mail: Kwaodawson1@yahoo.com; twumf@yahoo.co.uk; jbha@yahoo.com; ymissah@gmail.com
[*]*Corresponding Author*

Received 31 May 2023; Accepted 27 October 2023;
Publication 13 February 2024

Abstract

Due to the recent explosion in the amount of data being created by various social media platforms, e-commerce websites, and other businesses, a paradigm shift from on-site data centers to the cloud is required. Concerns about privacy and secrecy have been a major obstacle to the mainstream adoption of cloud computing. The best approach to protect the confidentiality and privacy of cloud data is by using cryptographic techniques. Researchers have developed several cryptographic algorithms, but they all have lengthy, linear, predictable, memory-intensive execution times. The performance of the CPU, memory, run-time trend, and throughput of the three cryptographic schemes: Enhanced RSA (ERSA), Non-Deterministic Cryptographic Scheme (NCS), and Enhanced Homomorphic Scheme (EHS) are compared using RAsys. The experiment's results demonstrated that NCS and EHS produced

Journal of Cyber Security and Mobility, Vol. 13_2, 265–282.
doi: 10.13052/jcsm2245-1439.1324

non-linear and non-deterministic run times. Again, NCS and EHS produced the lowest throughput and memory consumption for text and numeric data types when data sizes of $5n * 10^2$ $(KB\ (\in 1, 2, 4, 10, 20, 40)$ were processed. However, ERSA produced a run-time trend that was deterministic, linear, and predictable

Keywords: Non-deterministic, cryptography, execution time, encryption, decryption, throughput.

1 Introduction

In light of the intricacy of communication brought on by people's expanding activities, data security is increasingly required [1]. A paradigm change in data storage from on-site data centers to the cloud has been required in order to safeguard the large amounts of data created by different social media platforms, including Facebook, Twitter, Instagram, and others, as well as e-commerce websites [2, 3].

According to Maeser, a study by analytics company Cyence, discovered that a four-hour Amazon cloud computing outage in 2017 cost S&P 500 firms $150 million [4]. A network traffic monitoring company called Apica also predicted that the 54 most well-known e-commerce sites would see a reduction in activity of at least 20% [4]. In addition, Maeser predicted that while the Internet of Things would generate large amounts of data, the need for cloud computing will rise steadily between 2013 and 2020, amounting to about 266% [4]. Again, Maeser emphasized that, in accordance with RightScale's predictions [4], the infrastructure-as-a-service component of cloud computing will result in an increase in demand of 85% or more.

Cloud computing continues to surpass conventional on-site data centers in popularity because of the advantages of agility, scalability, and availability as well as the capacity to speed up the creation of work and reduce operational expenses via the effective use of pay-as-you-use services [5, 6]. This has compelled technology behemoths to spend vast sums of money on cloud computing in order to supply cloud services as contrasted to prior. These benefits have encouraged companies to offload their data to the cloud using cloud service models including Software-as-a-Service (SaaS), Infrastructure-as-a-Service (IaaS), Platform-as-a-Service, and Container-as-a-Service (CaaS) [7, 8], and [9].

The usage of cloud computing raises a number of security concerns, including data confidentiality and privacy [10, 11], and 12]. The use of

cryptographic algorithms has shown to be a viable and effective technique of ensuring the secrecy and privacy of cloud data [13–16], and [17]. The employment of cryptographic methods helps to maintain the secrecy and privacy of cloud data, much like with Enhanced RSA (ERSA) [18], Non-Deterministic Cryptographic Scheme (NCS) [19], and Enhanced Homomorphism scheme [20].

In order to determine which algorithm is most effective for ensuring the confidentiality and privacy of cloud data, this study compares the execution times, CPU utilization, memory usage, and throughput time of Enhanced RSA (ERSA) [18], Non-Deterministic Cryptographic Scheme (NCS) [19], and Enhanced Homomorphic Scheme (EHS) [20].

1.1 Problem Statement and Contribution

Data security has proven to be the biggest obstacle to the adoption of cloud computing [21]. To protect cloud data, researchers have proposed several cryptographic schemes. Execution times of the proposed schemes are predictable because of the linear run time trend these systems provide and the inverse connection between run time and data volume. This gives hackers knowledge to calculate runtime and get ready to hack on systems. The linear run time has the effect of raising the amount of data bandwidth required for the transmission and download of huge data volumes. Due to the increasing CPU activity, both the cloud service provider and the subscriber's equipment deteriorate and wear out. The main objective of this study is to experiment with Enhanced RSA [18], Non-Deterministic Cryptographic Scheme [19], and Homomorphism Scheme [20] in order to ascertain the computational statistics of the best-performing cryptographic scheme. Once more, the study provides a well-informed framework that practically and conceptually combines all of the recognized algorithms into a potent system named RAsys. The primary contribution of this study is the proposal of a comprehensive cryptographic system or schemes that may be employed to safeguard the confidentiality and privacy of cloud data.

2 Related Works

In an effort to protect the confidentiality and privacy of cloud data, many cryptographic methods have been developed. An overview of the review of past studies has been given in this section.

ALmarwani et al. [22] proposed a novel tagging approach called Tagging of Outsourced Data (TOD) to secure the privacy of data stored in the cloud. By employing verification, their plan supported cloud data. Because of the short run time of their algorithm, mobile devices might make great use of it. Despite how innovative their method appeared, it was linear. Tahir et al. proposed a genetic algorithm named CryptoGa in their study [23] to enable Almarwani et al. achieve data privacy. When compared to cutting-edge algorithms such as AES, RSA, and DES, their algorithm performed quicker. Nonetheless, the execution time was linear.

Shen et al. [24] recommended the usage of proxy re-encryption and Oblivious Random Access Memory (ORAM) to attain confidentiality and privacy of cloud data. Their technology was developed to allow several users to share cloud data. Members can control access using the ciphertext created via proxy re-encryption, ensuring data privacy. Adee and Mouratidis' [25] work, which encompassed cryptography and steganography, offered proof to back up their results. Although the algorithm's efficiency seemed promising, its execution time was linear.

Thabit et al.'s [26] approach for a Lightweight Cryptographic Algorithm ensured the confidentiality and privacy of cloud data. Using Feistel and substitution algorithms, they raised the complexity of encryption. Although their approach had a linear execution time, it was extremely efficient in terms of run time.

According to the linked research, all of the recommended algorithms ensured the confidentiality and secrecy of cloud data, but their execution times were inversely related to the amount of data processed, making them predictable.

3 Methodology

3.1 Algorithms Used in This Work

3.1.1 Enhanced RSA

By integrating traditional RSA with the Gaussian interpolation formula, Enhanced RSA improved the security of conventional RSA to the fifth degree. The ASCII values of the message are encrypted initially using Gaussian Forward interpolation, and the second and third layers are encrypted and decoded using the standard RSA. As seen in Figure 1, the last stage uses Gaussian First Backward interpolation to re-decode the data. The integration aids in overcoming the well-known RSA factorization challenge [18].

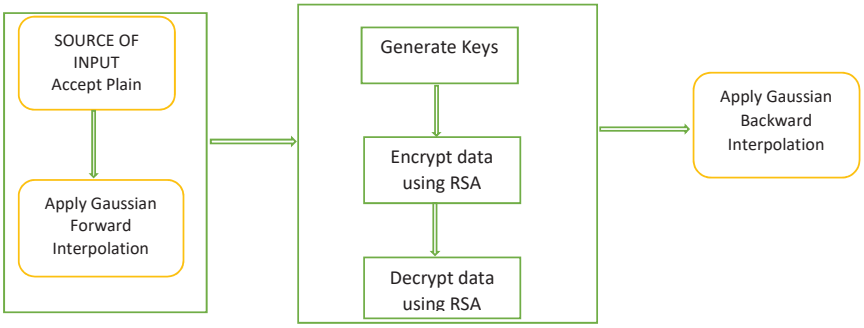

Figure 1 Work flow diagram of enhanced RSA [18].

3.1.2 Non-deterministic cryptographic scheme (NCS)

For producing keys, encrypting data, and decrypting data, NCS employs three phases. The three layers of key creation are intended to give concealed keys that increase the algorithm's security [19]. These include the Fixed Sliding Window Algorithm, Good Primes, the Linear Congruential Generator, and XORing the output to plaintext. Following the use of the Fixed Sliding Window [19], as shown in Algorithm 1, twelve numbers are generated in NCS. A sub-array of $(\frac{n(a[i])}{4})$ is computed on these twelve numbers.

Algorithm 1 The proposed NCS algorithm

Procedure NCS
Compute H = P *Q ▷ $H : P, Q \in Good\ Prime$
$X_Z = k(X_{x-1}) + r\ mod\ n$ ▷ $(n > 0,\ 0 < k < n,\ 0 \leq r < n,\ Compute\ the\ CLG$
$(X_{x-1}) = H$
$X_Z \in 1 \ldots 100,000$
for $i = 0, i < 12, i++$) **do**
$\{\boldsymbol{a}[\boldsymbol{i}] = Rand\ (1, 100000)\}$
end for
$Apply\ Fixed\ Sliding\ Window\ (FSD)\ on\ 12\ arrays\ susing\ sub\text{-}array\ of\ 3$
$a_y = a_i + a_{i+1} + a_{i+2}$
$a_{y1} = a_{i+3} + a_{i+4} + a_{i+5}$
$a_{y2} = a_{i+6} + a_{i+7} + a_{i+8}$
$a_{y3} = a_{i+9} + a_{i+10} + a_{i+11}$
$s_j = \max(a_y, a_{y1}, a_{y2}, a_{y3})$
$y_i = \boldsymbol{x_i} \oplus \boldsymbol{s_i}\ mod\ \left(\left(\frac{n(a[i])}{4}\right)\right)$ ▷ *ENCRYPTION*
$x_i = \boldsymbol{y_i} \oplus \boldsymbol{s_i}\ mod\ \left(\left(\frac{n(a[i])}{4}\right)\right)$ ▷ *DECRYPTION*
End Procedure

3.1.3 Enhanced homomorphism scheme

The Enhanced Homomorphism Scheme (EHS) is a combination of the Linear Congruential Generator (LCG), Fixed Sliding Window Algorithm (FSWA), Good Prime Numbers (GPN), and Gentry's Algorithm. This strategy considers two stages: the development of keys and the execution of the homomorphism scheme. Two of the three methods used to generate the keys are the generation of two great prime numbers and the use of the product as a seed for the Linear Congruential Generator to produce twelve numbers. The twelve integers are subjected to the sliding window technique utilizing a sub-array of three ($\frac{n(a[i])}{3}$).

As observed in Equation (1) for data encryption [20], the first value is s_i, the second value is s_j, the third value is s_k, the fourth value is s_l, and with M the plaintext, as shown in Algorithm 2.

$$C = M + s_i * s_j + s_k * s_l \tag{1}$$

Algorithm 2 Proposed algorithm

Procedure EHS
Start of algorithm
  Compute H = P *Q ▷ $H : P, Q \in Good\ Prime$
$X_Z = k(X_{x-1}) + r\ mod\ n$ ▷ $(n > 0,\ 0 < k < n,\ 0 \leq r < n,\ Compute\ the\ CLG$
$(X_{x-1}) = H$
$X_Z \in 1 \ldots 100,000$
for $i = 0, i < 12, i++$ **do**
  $\{\boldsymbol{a}[\boldsymbol{i}] = Rand\ (1, 100000)\}$
end for
Apply Fixed Sliding Window (FSD) on 12 *arrays susing sub-array of* 3
$s_i = \sum_{n=0}^{2} a_{yn}$
$s_j = \sum_{n=3}^{5} a_{yn}$
$s_k = \sum_{n=6}^{8} a_{yn}$
$s_l = \sum_{n=9}^{11} a_{yn}$
$C_I = \boldsymbol{M_t} + \boldsymbol{s_i} * \boldsymbol{s_j} + \boldsymbol{s_k} * \boldsymbol{s_l}$ ▷ *Encryption*
$C_n+ = C_I$ ▷ Addition homomorphism
$\boldsymbol{C_t}* = \boldsymbol{C_i}$ ▷ Multiplication
$\boldsymbol{M_d} = \boldsymbol{C_t}\ \boldsymbol{mod}\ \boldsymbol{s_k}$ ▷ *Decryption*
End

4 Experimentation

The Enhanced RSA [18], Non-Deterministic Cryptographic Scheme [19], and Enhanced Homomorphism Scheme [20] were compared using a C#

programming language and an i7 Lenovo computer with a 2.10 GHz CPU. As demonstrated in a NET C# programming language where it was used to analyze the execution of the AES algorithm, resulting in 300MB/seconds whereas OpenSSL C simulation obtained an average performance of 960MB/seconds [39], C# programming language is preferable since it influences data execution time.

4.1 Description of Dataset Used in This Work

The dataset for this study was obtained from the Kaggle database [27]. The dataset provides an English-to-French translation using a combination of text and numbers, allowing the algorithms' strength to be evaluated in terms of an execution time trend. This is noteworthy since Loyka et al.'s study yielded different findings when only text and numbers were utilized [28]. The encryption time trend for Loyka et al. when only text was tested was non-linear, but linear for decryption time when only numbers were run [28]. The suggested methods were tested on data sets of $5n * 10^2$ $(KB$ $(\in 1, 2, 4, 10, 20, 40))$.

5 Results and Discussion

5.1 Architectural Design for the Proposed Approach

This section explains the implementation of the four phases in the proposed RAsys, with Figure 2 providing a visual perspective of data sharing and storage. Cloud computing provides storage as part of Infrastructure-as-a-Service. This allows for data storage and sharing in the cloud using a web browser on any electronic device [29].

5.2 The Proposed Framework of the System

This section presents a high-level summary of the improved RSA, enhanced homomorphism scheme, and non-deterministic cryptographic scheme comparison. The diagrammatic representation (RAsys) of the framework is shown in Figure 3. The ERSA, NCS, and EHS structures each include five phases: key generation, encryption, decryption, memory use, and throughput.

How a user registers with a cloud service provider is shown in Figure 3. The registered client uploads the ciphertext onto the cloud after using ERSA, NCS, or EHS to encrypt the plaintext and from which the encryption time, memory utilization, and throughput timings are generated. After downloading

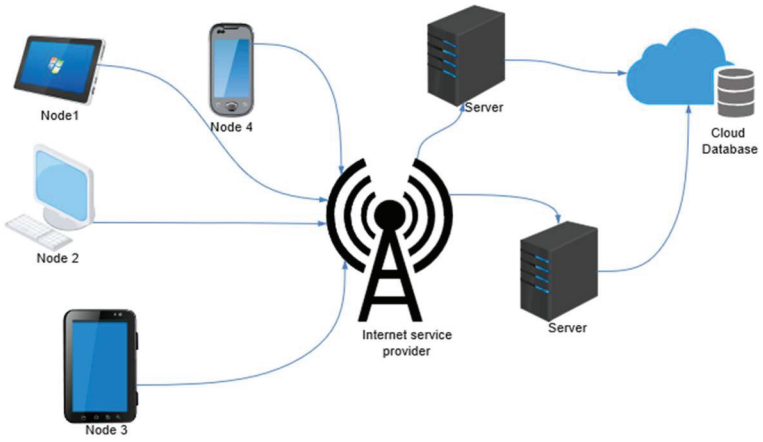

Figure 2 Architectural design for the proposed RAsys.

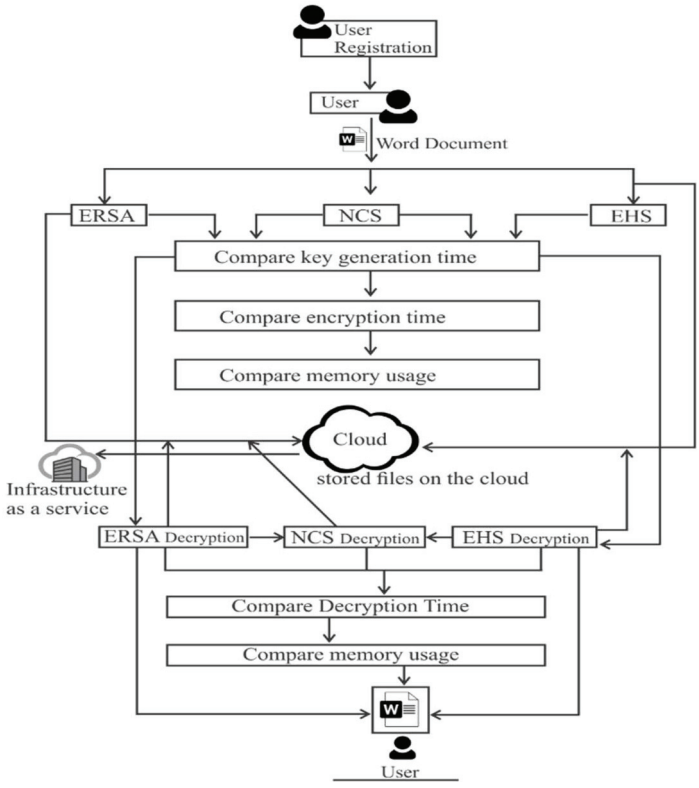

Figure 3 Framework for the proposed RAsys.

the Ciphertext, the decryption, memory storage, and throughput times are determined.

5.3 Computational Results

Tables 1 and 2 show the results of running the algorithms fifty times and calculating their averages.

5.3.1 Encryption time

According to Table 1, EHS had the fastest encryption time for data with a size of 500 KB (0.5 MB), clocking in at 381 milliseconds, followed by NCS at 614 milliseconds. When the data amount was raised to 1000 KB, the ERSA encryption procedure took longer, increasing from 648 milliseconds to 685 milliseconds. However, the encryption times for NCS and EHS decreased from 614 milliseconds to 438 milliseconds and 381 milliseconds to 257 milliseconds, respectively, when the data size was raised from 500 KB to 1000 KB.

5.3.2 Decryption time

The trends in the average decryption times for ERSA, NCS, and EHS are compared in Table 2. The quickest was EHS, which had a decryption time of 392 milliseconds with a data size of 500 KB. Decryption times for data sizes of 500 KB, 1000 KB, 2000 KB, 5000 KB, 10 000 KB, and 20 000 KB rose as well, moving from 563 milliseconds to 590 milliseconds, 693 milliseconds, 740 milliseconds, 763 milliseconds, and 840 milliseconds, respectively. However, NCS required 643 milliseconds to decode 500 KB of data. The decryption time dropped to 564 milliseconds when the data size was increased to 1000 KB. It increased to 93 milliseconds when the data size was increased to 5000 KB, then again dropped to 8 milliseconds with a 2000 KB data size. The decryption time for NCS increased to 725

Table 1 Comparing the average encryption time of ERSA, NCS, and EHS

Data Size (KB)	ERSA (ms) [18]	NCS (ms) [19]	EHS (ms) [20]
500	648	614	381
1000	685	438	257
2000	701	692	999
5000	753	805	668
10000	780	406	766
20000	832	158	479

Table 2 Comparing the average decryption time of ERSA, NCS, and EHS

Data Size (KB)	ERSA (ms) [18]	NCS (ms) [19]	EHS (ms) [20]
500	563	643	392
1000	590	564	268
2000	693	8	69
5000	740	93	725
10000	768	725	782
20000	840	534	496

Table 3 Encryption throughput (KB/ms)

Data Size (KB)	ERSA (ms) [18]	NCS (ms) [19]	EHS (ms) [20]
500	0.77	0.81	1.31
1000	1.46	2.28	3.89
2000	2.85	2.89	2
5000	6.64	6.21	7.49
10000	12.8	24.63	13.05
20000	24.04	126.58	41.75

Table 4 Decryption throughput (KB/ms)

Data Size (KB)	ERSA (ms) [18]	NCS (ms) [19]	EHS (ms) [20]
500	0.89	0.78	1.28
1000	1.70	1.77	3.73
2000	2.89	250	28.99
5000	6.76	53.76	6.9
10000	13.02	13.79	12.79
20000	23.81	37.45	40.32

milliseconds when a data size of 10,000 KB was run, but it decreased to 534 milliseconds when a data size of 2,000 KB had been processed.

5.3.3 Throughput

In Table 3, EHS had the throughput time that was the fastest at 1.31 KB/ms when 500 KB of data was executed. The same thing happened when the data size was extended from 500 KB to 1000 KB and 5000 KB. However, NCS had the fastest encryption throughput time of 126.56 KB/ms, followed by EHS with a throughput time of 41.75 KB/ms when data size of 20000 KB was executed.

With 500 KB of data, EHS had the fastest decryption throughput time in Table 4, clocking up at 1.28 KB/ms. EHS once more had the quickest

Table 5 Encryption memory usage for ERSA, NCS, and EHS (MB)

Data Size (KB)	ERSA (ms) [18]	NCS (ms) [19]	EHS (ms) [20]
500	17	13	18
1000	17	9	18
2000	17	9	18
5000	18	9	18
10000	19	9	17
20000	19	9	18

Table 6 Decryption memory usage for ERSA, NCS, and EHS (MB)

Data Size (KB)	ERSA (ms) [18]	NCS (ms) [19]	EHS (ms) [20]
500	17	13	18
1000	17	9	18
2000	17	9	18
5000	18	9	18
10000	19	9	17
20000	19	9	18

throughput time for decryption at 40.32 KB/ms even when the data size was increased to 20000 KB.

The NCS model exhibited the lowest average memory consumption from Tables 5 and 6, with data sizes ranging from 500 KB to 20000 KB.

6 Discussion

Due to the enormous amounts of data that social media platforms and e-commerce companies create every day, encryption techniques with fast run times are crucial in cloud computing systems [30]. It is possible to infer from Table 1 that the amount of the data executed, which came about as a result of the size of the key selected and the size of the data, is proportional to the length of the ERSA encryption process. The run times is deterministic, predictable, and patterned as a result hackers are able to attack such system based on the trend of the execution times [31–33, 38, 41], and [42]. As indicated in Table 1, the only factor affecting the run times of NCS and EHS is the secret key size, hence these encryption times are neither patterned, non-deterministic, nor unpredictable.

According to the trend in run times shown in Table 1, encryption times for NCS and EHS is only reliant on secret key size, not data size, however encryption time for ERSA is affected by both data size and secret key size [34–36, 38, 41], and [42].

Additionally, the ERSA decryption time is proportionate, making the trend in decryption time predictable and linear [37–39], and [40]. However, the Fixed Sliding Window Algorithm, which disintegrates the enormous numbers acquired by choosing the good prime numbers as the beginning keys and XORing the secret key and the Ciphertext, as used by NCS and EHS, results in non-deterministic and nonlinear decryption times. This demonstrates how much less expensive data transfers to and from the cloud are, as well as how much less wear and tear is placed on the computer hardware when NCS and EHS is employed. Tables 2 and 4 when compared inferred that a low throughput time corresponds to a high decryption time.

7 Conclusion

This research provides an in-depth analysis of three cryptographic algorithms with the goal of achieving the quickest linear, non-linear execution times with the least amount of memory usage and throughput time. Encryption time, decryption time, memory use, and throughput were the performance parameters used to compare the various algorithms. The comparison's findings showed that the execution times trends provided by ERSA was linear, patterned, predictable, and deterministic. Again, they required more memory and had longer execution times. However, it has been demonstrated that NCS and EHS had the advantage of producing a non-linear execution time trend, non-patterned execution time trend, non-deterministic execution time trend, lowest execution time, and consumed less amount of memory during execution capable of resisting side-channel attack. Additionally, this makes NCS and EHS eliminate the need for high bandwidth to transmit and download large amounts of data as well as minimize ripping and wearing of hardware due to excessive CUP consumption.

References

[1] M. Tajammul, R. Parveen and I. A. Tayubi, "Comparative Analysis of Security Algorithms used in Cloud Computing," *2021 8th International Conference on Computing for Sustainable Global Development (INDIACom)*, 2021, pp. 875–880.

[2] I. A. Ibrahim and M. Bassiouni, "Improvement of job completion time in data-intensive cloud computing applications," *Journal of Cloud Computing*, vol. 9, no. 1, Feb. 2020, DOI: 10.1186/s13677-019-0139-6.

[3] B. Seth et al., "Secure Cloud Data Storage System using Hybrid Paillier-Blowfish Algorithm," *Computers, Materials & Continua*, vol. 67, no. 1, pp. 779–798, 2021, DOI: 10.32604/cmc.2021.014466.

[4] R. Maeser, "Analyzing CSP Trustworthiness and Predicting Cloud Service Performance," *IEEE Open Journal of the Computer Society*, vol. 1, pp. 73–85, 2020, DOI: 10.1109/ojcs.2020.2994095.

[5] J. Hassan et al., "The Rise of Cloud Computing: Data Protection, Privacy, and Open Research Challenges-A Systematic Literature Review (SLR)," *Computational intelligence and neuroscience*, vol. 2022, p. 8303504, Jun. 2022, DOI: 10.1155/2022/8303504.

[6] Y.-K. Kim, H.-J. Kim, H. Lee, and J.-W. Chang, "Privacy-preserving parallel kNN classification algorithm using index-based filtering in cloud computing," *PLOS ONE*, vol. 17, no. 5, p. e0267908, May 2022, DOI: 10.1371/journal.pone.0267908.

[7] M. Mustafa, M. Alshare, D. Bhargava, R. Neware, B. Singh, and P. Ngulube, "Perceived Security Risk Based on Moderating Factors for Blockchain Technology Applications in Cloud Storage to Achieve Secure Healthcare Systems," *Computational and Mathematical Methods in Medicine*, vol. 2022, pp. 1–10, Jan. 2022, DOI: 10.1155/2022/6112815.

[8] Y. Al-Issa, M. A. Ottom, and A. Tamrawi, "eHealth Cloud Security Challenges: A Survey," *Journal of Healthcare Engineering*, vol. 2019, pp. 1–15, Sep. 2019, DOI: 10.1155/2019/7516035.

[9] M. Hussein, M. Mousa, and M. Alqarni, "A placement architecture for a container as a service (CaaS) in a cloud environment," *Journal of Cloud Computing*, vol. 8, no. 1, 2019. Available: 10.1186/s13677–019-0131–1.

[10] P. Kulkarni, R. Khanai, and G. Bindagi, "A Comparative Analysis of Hybrid Encryption Technique for Images in the Cloud Environment," *2020 International Conference on Communication and Signal Processing (ICCSP)*, Jul. 2020, DOI: 10.1109/iccsp48568.2020.9182153.

[11] R. K. Nema, A. K. Saxena and R. Srivastava, "Survey of the Security Algorithms over Cloud Environment to Protect Information," 2022 10th International Conference on Emerging Trends in Engineering and Technology – Signal and Information Processing (ICETET-SIP-22), 2022, pp. 1–6, DOI: 10.1109/ICETET-SIP-2254415.2022.9791643.

[12] S. Mittal et al., "Using Identity-Based Cryptography as a Foundation for an Effective and Secure Cloud Model for E-Health," *Computational Intelligence and Neuroscience*, vol. 2022, pp. 1–8, Apr. 2022, DOI: 10.1155/2022/7016554.

[13] P. Kumar, A. Suresh, V. Anbarasu, S. P. Anandaraj, and S. Udayakumar, "A decentralized secured grid integration system using APEBC technique with multi-access AI framework," *Sustainable Computing: Informatics and Systems*, vol. 35, p. 100777, Sep. 2022, DOI: 10.1016/j.suscom.2022.100777.

[14] M. Abu-Faraj, A. Al-Hyari, K. Aldebei, Z. A. Alqadi and B. Al-Ahmad, "Rotation Left Digits to Enhance the Security Level of Message Blocks Cryptography," in IEEE Access, vol. 10, pp. 69388–69397, 2022, DOI: 10.1109/ACCESS.2022.3187317.

[15] P. Chinnasamy and P. Deepalakshmi, "HCAC-EHR: hybrid cryptographic access control for secure EHR retrieval in healthcare cloud," *Journal of Ambient Intelligence and Humanized Computing*, Feb. 2021, DOI: 10.1007/s12652-021-02942-2.

[16] S. Mittal et al., "Using Identity-Based Cryptography as a Foundation for an Effective and Secure Cloud Model for E-Health," *Computational Intelligence and Neuroscience*, vol. 2022, pp. 1–8, Apr. 2022, DOI: 10.1155/2022/7016554.

[17] C. Mangla, S. Rani, and H. K. Atiglah, "Secure Data Transmission Using Quantum Cryptography in Fog Computing," *Wireless Communications and Mobile Computing*, vol. 2022, pp. 1–8, Jan. 2022, DOI: 10.1155/2022/3426811.

[18] J. K. Dawson, F. Twum, J. B. H. Acquah, Y. M. Missah, and B. B. K. Ayawli, "An enhanced RSA algorithm using Gaussian interpolation formula," *International Journal of Computer Aided Engineering and Technology*, vol. 16, no. 4, p. 534, 2022, DOI: 10.1504/ijcaet.2022.123996.

[19] J. K. Dawson, F. Twum, J. H. Acquah, and Y. M. Missah, "Ensuring Confidentiality and Privacy of Cloud Data Using a Non-Deterministic Cryptographic Scheme," Ploseone, DOI: 10.1371/journal.pone.0274628

[20] J. K. Dawson, F. Twum, J. H. Acquah, and Y. M. Missah, "Ensuring privacy and confidentiality of data on the cloud using an enhanced homomorphism scheme," *Informatica*, vol. 46, no. 8, Nov. 2022, DOI: 10.31449/inf.v46i8.4305.

[21] L. Liu, M. Cao, and Y. Sun, "A fusion data security protection scheme for sensitive E-documents in the open network environment," *PLOS ONE*, vol. 16, no. 12, p. e0258464, Dec. 2021, DOI: 10.1371/journal.pone.0258464.

[22] R. ALmarwani, N. Zhang, and J. Garside, "An effective, secure and efficient tagging method for integrity protection of outsourced data in

a public cloud storage," *PLOS ONE*, vol. 15, no. 11, p. e0241236, Nov. 2020, DOI: 10.1371/journal.pone.0241236.

[23] M. Tahir, M. Sardaraz, Z. Mehmood, and S. Muhammad, "CryptoGA: a cryptosystem based on genetic algorithm for cloud data security," *Cluster Computing*, Jul. 2020, DOI: 10.1007/s10586-020-03157-4.

[24] J. Shen, H. Yang, P. Vijayakumar, and N. Kumar, "A Privacy-Preserving and Untraceable Group Data Sharing Scheme in Cloud Computing," in IEEE Transactions on Dependable and Secure Computing, vol. 19, no. 4, pp. 2198–2210, 1 July–Aug. 2022, DOI: 10.1109/TDSC.2021.3050517.

[25] R. Adee and H. Mouratidis, "A Dynamic Four-Step Data Security Model for Data in Cloud Computing Based on Cryptography and Steganography," *Sensors*, vol. 22, no. 3, p. 1109, Feb. 2022, DOI: 10.3390/s22031109.

[26] F. Thabit, A. P. S. Alhomdy, A. H. A. Al-Ahdal, and P. D. S. Jagtap, "A new lightweight cryptographic algorithm for enhancing data security in cloud computing," *Global Transitions Proceedings*, vol. 2, no. 1, pp. 91–99, Jun. 2021, DOI: 10.1016/j.gltp.2021.01.013.

[27] "Kaggle Datasets", *Kaggle.com*, 2022. [Online]. Available: https://www.kaggle.com/datasets/morriswongch/kaggle-datasets. [Accessed: 27-Sep-2022].

[28] K. Loyka, H. Zhou, and S. P. Khatri, "A Homomorphic Encryption Scheme Based on Affine Transforms," *Proceedings of 2018 on Great Lakes Symposium on VLSI*, May 2018, DOI: 10.1145/3194554.3194585.

[29] Y.-K. Kim, H.-J. Kim, H. Lee, and J.-W. Chang, "Privacy-preserving parallel kNN classification algorithm using index-based filtering in cloud computing," *PLOS ONE*, vol. 17, no. 5, p. e0267908, May 2022, DOI: 10.1371/journal.pone.0267908.

[30] Q. Sun, K. Lin, C. Si, Y. Xu, S. Li and P. Gope, "A Secure and Anonymous Communicate Scheme over the Internet of Things," *ACM Transactions on Sensor Networks*, vol. 18, no. 3, pp. 1–21, 2022. Available: 10.1145/ 3508392 [Accessed 28 September 2022].

[31] M. Papaioannou et al., "A Survey on Security Threats and Countermeasures in Internet of Medical Things (IoMT)," *Transactions on Emerging Telecommunications Technologies*, vol. 33, no. 6, 2020. Available: 10.1002/ett.4049 [Accessed 28 September 2022].

[32] W. Liu, W. Feng, M. Huang, Y. Xu, and X. Zheng, "STEB: A secure service trading ecosystem based on blockchain," *PLOS ONE*, vol. 17, no. 6, p. e0267914, Jun. 2022, DOI: 10.1371/journal.pone.0267914.

[33] R. Masram, V. Shahare, J. Abraham, and R. Moona, "Analysis and Comparison of Symmetric Key Cryptographic Algorithms Based on Various File Features," *International Journal of Network Security & Its Applications*, vol. 6, no. 4, pp. 43–52, Jul. 2014, DOI: 10.5121/ijnsa.2014.6404.

[34] A. Banerjee, M. Hasan, Md. A. Rahman, and R. Chapagain, "CLOAK: A Stream Cipher Based Encryption Protocol for Mobile Cloud Computing," *IEEE Access*, vol. 5, pp. 17678–17691, 2017, DOI: 10.1109/access.2017.2744670.

[35] M. Abd Zaid and S. Hassan, "Proposal Framework to Light Weight Cryptography Primitives," *Engineering and Technology Journal*, vol. 40, no. 4, pp. 516–526, Apr. 2022, DOI: 10.30684/etj.v40i4.1679.

[36] R. Serrano, C. Duran, M. Sarmiento, C.-K. Pham, and T.-T. Hoang, "ChaCha20–Poly1305 Authenticated Encryption with Additional Data for Transport Layer Security 1.3," *Cryptography*, vol. 6, no. 2, p. 30, Jun. 2022, DOI: 10.3390/cryptography6020030.

[37] H. H. Alyas and A. A. Abdullah, "Enhancement the ChaCha20 Encryption Algorithm Based on Chaotic Maps," *Lecture Notes in Networks and Systems*, pp. 91–107, 2021, DOI: 10.1007/978–981-16–0666-3_10.

[38] M. Panda and A. Nag, "Plain Text Encryption Using AES, DES, and SALSA20 by Java Based Bouncy Castle API on Windows and Linux," 2015 Second International Conference on Advances in Computing and Communication Engineering, 2015, pp. 541–548, DOI: 10.1109/ICACCE.2015.130.

[39] M. H. Alrowaithy, "Performance-efficient cryptographic primitives in constrained devices," *theses.ncl.ac.uk*, 2021. http://theses.ncl.ac.UK/jspui/handle/10443/5545 (accessed Oct. 22, 2022).

[40] P. Yadav, I. Gupta, and S. K. Murthy, "Study and analysis of eSTREAM cipher Salsa and ChaCha," 2016 IEEE International Conference on Engineering and Technology (ICETECH), 2016, pp. 90–94, DOI: 10.1109/ICETECH.2016.7569218.

[41] A. Salkanovic, S. Ljubic, L. Stankovic, and J. Lerga, "Analysis of Cryptography Algorithms Implemented in Android Mobile Application," *Information Technology and Control*, vol. 50, no. 4, pp. 786–807, Dec. 2021, DOI: 10.5755/j01.itc.50.4.29464.

[42] P. Singh and K. Deshpande, "Performance evaluation of cryptographic-ciphers on IoT devices," 2018. [Online]. Available: https://arxiv.org/ftp/arxiv/papers/1812/1812.02220.pdf.

Biographies

John Kwao Dawson is a PhD candidate in Computer Science at the Kwame Nkrumah University of Science and Technology. He holds a Master of Philosophy in Information Technology and Bachelor of Education in Information Technology from the Kwame Nkrumah University of Science and Technology and University of Education Winneba, respectively. His area of research is cloud computing, algorithm design, machine learning, artificial intelligence and data and network security.

Frimpong Twum received his BSc (hons.) in Electrical and Electronic Engineering and MSc in Internet and Multimedia Engineering from the London South Bank University, in 2004 and 2007, respectively. He also received his MSc in Information System from the Roehampton University, London, in 2011. He completed his PhD in Computer Science from the KNUST, Ghana, in 2017 with a specialization in Computer Security. He is currently a Senior Lecturer at the Department of Computer Science, KNUST.

James Benjamin Hayfron-Acquah is the Head of the Department of Computer Science. He had his first degree in Computer Science from the Kwame Nkrumah University of Science and Technology (KNUST), in 1991. He then proceeded to have his Master's in Computer Science and Application at the Shanghai University of Science and Technology (SUST), Shanghai, China, in 1996. He obtained his PhD from the Southampton University in the UK, in 2003. He joined the KNUST in March 1996 as a Lecturer. He was promoted to a Senior Lecturer in 2004 and Associate Professor in 2015.

Yaw Marfo Missah is a Lecturer in the Computer Science Department of Kwame Nkrumah University of Science and Technology. He obtained his PhD in Computer Science (DCS) in Enterprise Information System in 2013, Master of Science (MSIT) in Information Technology in 2004, and Bachelor of Science (BSc) in Computer Science in 2000.