JOURNAL OF ENGINEERING RESEARCH & SCIENCES

JENRS

www.jenrs.com
ISSN: 2831-4085

Volume 2 Issue 10
October 2023

EDITORIAL BOARD

Editor-in-Chief

Editorial Board Members

Editorial

In this edition, we focus on a critical area of the digital era: data security. As the sophistication of the internet and the volume of data transmission continue to grow, ensuring the security of files, databases, accounts, and networks has become paramount. Databases, often housing sensitive and private information, are prime targets for cyber attacks. This issue underscores the need for advanced solutions to mitigate threats and reduce risks associated with data processing and storage. The featured research paper introduces a cutting-edge approach to data security using homomorphic encryption for device authentication, emphasizing the

The paper addresses the pressing need to enhance data security in an environment increasingly susceptible to cyber threats. With the continuous evolution of encryption technologies and trends, staying updated is essential for protecting confidential information and sensitive data. The study underscores the significance of adopting robust encryption methods to bolster system security and protect against unauthorized access [1].

The featured paper in this edition presents a groundbreaking approach to data security, addressing a critical need in the digital age. By proposing a homomorphic encryption-based device authentication method, the research offers a robust solution to safeguarding template data and enhancing overall system security. The use of the CKKS technique further demonstrates the potential of advanced encryption methods in protecting sensitive information. We are pleased to share these innovative findings with our readers and look forward to further advancements in data security research.

References:

[1] S. Yadav, G. Howells, “Device Authentication using Homomorphic Encryption,” Journal of Engineering Research and Sciences, vol. 2, no. 10, pp. 1–8, 2023, doi:10.55708/js0210001.

Editor-in-chief

Prof. Paul Andrew

JOURNAL OF ENGINEERING RESEARCH AND SCIENCES

Volume 2 Issue 10 **October 2023**

CONTENTS

Special Issue on Multidisciplinary Sciences & Advanced Technology 2023

Received: 02 August, 2023, Revised: 25 October 2023, Accepted: 05 September, 2023, Online: 30 October, 2023

DOI: https://doi.org/10.55708/js0210001

Device Authentication using Homomorphic Encryption

Supriya Yadav [1,*] , Gareth Howells [2]

[1] School of Engineering & Digital Arts, University of Kent, Canterbury, CT2 7NB, United Kingdom
[2] School of Computing, University of Kent, Canterbury, CT2 7NB, United Kingdom
*Corresponding author: Supriya Yadav, University of Kent, +447729973232 & ersupriyayadav@gmail.com

ABSTRACT: In the digital era, data security in files, databases, accounts, and networks is of utmost importance. Due to the sensitive, private, or protected information they contain, databases are a common target for cyber-attacks. To assess threats to data and lower the risk involved with data processing and storage, data security is crucial. Therefore, it is necessary to find solutions to the data security problems. It has become crucial to be up-to-date on different encryption technologies and trends due to the internet's growing sophistication and dependence on internet data transmission. It can help protect confidential information and sensitive data and enhance the security of the system. In this paper, we propose a homomorphic encryption-based device authentication method while safeguarding template data. Homomorphic encryption technology has the capability of computing on encrypted data, making it more difficult for attackers to get their hands on the original template. In this study, the CKKS technique was used, which supports the approximation of real or complex numbers.

KEYWORDS: Data Security, Device Authentication, Homomorphic Encryption, Encrypted Data, CKKS

1. Introduction

The security of data over the internet is crucial, especially if this data is personal or confidential. The transmitted data can be intercepted during its journey from one device to another. For that reason, there was a need to develop a simple method to secure data. Data encryption is one method to secure the messages, but intruders can still try to crack them.

The homomorphic encryption (HE) algorithm allows mathematical operations to be performed on encrypted data. This is one of the extremely beneficial characteristics that led us to decide on using HE for device authentication [1],[2].

Talking about data, there are three different forms of it: stored, transmitted, and operated on. There are many algorithms available today to secure the first two data variations; however, few of these algorithms can operate on encrypted data, which is where homomorphic encryption comes in because it makes it possible to perform calculations on encrypted data. This means that data processing can be outsourced to a third party without the need to trust the third party to properly secure the data. Without the proper decryption key, the original data can't be accessed.

In general, data security must meet the three criteria listed below:

Confidentiality: One of the most important components of data security is confidentiality. Data confidentiality means that only authorized users are permitted access to the data. Data confidentiality guarantees that unauthorized users are kept at bay.

Data integrity: is the term used to describe safeguarding data from unauthorized change. Data integrity must be implemented in the cloud to prevent unauthorized data modifications.

Data accessibility: Accessibility is a crucial component of data security. Data availability aims to provide clients with secure network access to their data at any time, from any location.

Data processing on encrypted data in cloud environments is a novel method for securing data. To create such a solution, encryption techniques known as homomorphic encryption have been introduced [1], [3].

This technique permits the operation of encrypted data while guaranteeing data confidentiality during processing.

Since device data was collected from various sources, a data compromise would result in a spoofing attack, and the bad actor could impersonate the device. To ensure the data on this device was not compromised, it had to be encrypted. Now, the question was, "How to protect the data on the device or server?" The most viable option was to leverage file encryption tools.

To achieve this, the following are the alternative techniques that were considered:

Password-based: When files are encrypted, the user specifies the password. The tool will ask for the password during decryption. Because malware can read the file and conduct an offline brute-force attack, it is practical but not very safe [4].

Token- or dongle-based: The hardware token or dongle that contains the decryption private key must be used to perform the decryption. The token can only try the PIN ten times before initializing. However, this method needs a token and is a lot safer than passwords.

HE does not require password or private key protection, resulting in lower overhead to manage and a better solution. Thus, it was the most viable choice for this purpose.

The techniques used to solve this problem are homomorphic encryption (HE), which is data processing delegation without granting access to it. HE enables operations to be performed directly on encrypted data without ever using the decryption key. Using HE, data is encrypted on the client side, pushed into the cloud, securely processed, and the results are sent back to the client for decryption [2], [3].

Homomorphic encryption (HE) offers several advantages over password- or token-based approaches. Password management is not necessary [4]. As a result, it offers a higher level of security. The management of dongles is not required. As a result, the costs associated with hardware and the challenges of managing logistics can be avoided.

Homomorphic encryption techniques offer creative ways to support calculations on encrypted data while protecting the content of private data. These methods do, however, have certain drawbacks, such as high computational costs and the requirement for custom adjustments for every case study [5], [6].

Along with authentication, homomorphic encryption can be leveraged for device identification by performing mathematical operations on encrypted data, which will ensure the protection of the original data. This is the focus of discussion in this paper. Features used for this study exhibit non-standard and multimodal distributions, which present a significant challenge to model and characterize [7]. The details of the mathematical functions were investigated to see if there is a way to test samples and compare them for closeness to the training models for device identification using encrypted modelled data.

The rest of the paper is organized as follows:

Section II focuses on related works; Section III describes homomorphic encryption; Section IV explains the proposed system; Section V focuses on experiments; and Section VI concludes the paper.

2. Background

According to the related work described below, data security has been the subject of substantial investigation.

For monitoring chronic diseases, the study in [8] established an IoT-based architecture with homomorphic encryption to protect against data loss and spoofing assaults. The findings imply that homomorphic encryption offers simple, affordable protection for private health information. For the protection of medical data, blockchain technologies are also used in conjunction with homomorphic encryption.

In 2012, the author introduced the SDC scheme, a straightforward FHE developed from the Gentry cryptosystem, to guarantee cloud storage anonymity [9]. In 2014, the author suggested an enhancement to Gentry's second strategy to enhance the security of applying Fully Homomorphic Encryption (FHE) in cloud computing. Various academics have suggested diverse applications of homomorphic encryption in the field of cloud computing [10]. In 2014, the author introduced a secure image retrieval technique for cloud computing that relies on the homomorphic characteristics of the Paillier scheme [11]. In 2015, the author introduced a technique for safeguarding privacy in medical cloud computing by employing completely homomorphic encryption [12]. In January 2016, the author introduced an XOR homomorphism encryption technique designed to facilitate secure keyword searching on encrypted data for cloud storage [13].

To prevent potential password breaches, the author [14] proposes a novel authentication system based on the password authentication protocol, which uses two servers modified to store passwords. The El Gamal algorithm and DH are employed in this paper. Backup services are offered to maintain the service. Client data from server one is retained as a backup on server two, and vice versa. If one of the two servers were to shut down for whatever reason, the client would still need to get services from another server. This protocol offers protection from both active and passive assaults [14]

In 2016, the author proposed a banking application for data security. The bank contains a large amount of confidential customer information that must be protected from unauthorised access, so the data must be kept confidential. In this paper, paillier HE is used to apply operations to encrypted banking information because it enables performing calculations on ciphertext without the use of a secret key. This plan provides data security and confidentiality [15].

One of the most secure authentication methods, according to a paper [16] published in 2016, is the use of biometric validation. The biometric data is saved on the remote server in encrypted form. In this paper's proposed palm print authentication approach, the matching of the user input to the registered biometric data is computed in an encrypted domain based on pailliar homomorphic encryption. This plan is carried out effectively [16].

In 2017, author came up with a security model for biometric verification. In this study, they come up with a new verification scheme based on HE for template protection using multiple biometrics. The Paillier homomorphic encryption scheme is used to encode, process, and decrypt data. By computing the original biometric data and the encrypted template, HE Verification handles the sole cipher text. High accuracy rates can be seen in the results [17].

Encryption methods like homomorphic encryption are also used to protect medical data. Such institutions as hospitals and research institutes are developing technical solutions for sharing patient data in a privacy- preserving manner. Two of these technical solutions are homomorphic encryption and distributed ledger technology. Homomorphic encryption allows encrypted patient data to be shared with other health care service providers while it is encrypted [18].

The authors of [19] suggest a homomorphic encryption-based online safe multiparty computation with patient information sharing to hospitals. In this paper [20], a homomorphic encryption model based on heart rate data was proposed and linked to a personal health information system. The findings show that, despite anticipated storage and network issues, the technique presented was successful in meeting the needs for secure data processing for 500 patients.

The authors of [21] presented a data division scheme-based homomorphic encryption for wireless sensor networks. The findings demonstrate that data security and resource availability are mutually exclusive. By monitoring the patients' vitals with a simple encryption system, [22] demonstrates the applicability of homomorphic encryption. While encryption only occurs in medical facilities, sensor data like breathing and heart rate are encrypted using homomorphic encryption before being transmitted to an unreliable third party.

For monitoring chronic diseases, the study in [23] established an IoT-based architecture with homomorphic encryption to protect against data loss and spoofing assaults. The findings imply that homomorphic encryption offers simple, affordable protection for private health information. For the protection of medical data, blockchain technologies are also used in conjunction with homomorphic encryption. In their article [24], the authors suggested combining blockchain and homomorphic encryption in intelligent transportation systems with autonomous healthcare monitoring to track pandemic infections. In a different work [25], they used homomorphic encryption to create a searchable distributed medical database on a blockchain. The necessity to protect sensitive information is growing, which encourages the integration of several strategies.

The next section provides a detailed explanation of homomorphic encryption.

3. Homomorphic Encryption

According to the problem analysis in Section II and the overview presented above, it is crucial to use a homomorphic encryption approach to safeguard the confidentiality of data.

How organisations and individuals use and manage their data has fundamentally changed thanks to affordable cloud computing and cloud storage. Data can be conveniently saved in encrypted form using traditional encryption techniques like AES, which are incredibly quick. However, the owner of the data must download, decode, and act on the encrypted data locally, which can be expensive and logistically challenging. Alternatively, the cloud server must have access to the secret key, raising security issues. Because the cloud may directly process the encrypted data and only provide the encrypted output to the data owner, homomorphic encryption can greatly simplify this scenario. In more complex application scenarios, many parties with private data may be involved [26]. In these cases, a third party may carry out an operation and then deliver the findings to one or more of the participants for decryption.

Homomorphic encryption is being developed to protect the security of information that must be kept private.

A fully homomorphic cryptosystem is one in which any action on the plaintext may be performed by performing an operation on the corresponding ciphertext (FHE). Gentry [1], [2] made the first such cryptosystem suggestion in a ground-breaking paper. The structure meets the requirements for FHE, but it is found to be computationally inefficient to be used in practice [3], and creating FHE schemes that are computationally feasible is an ongoing field of research [27].

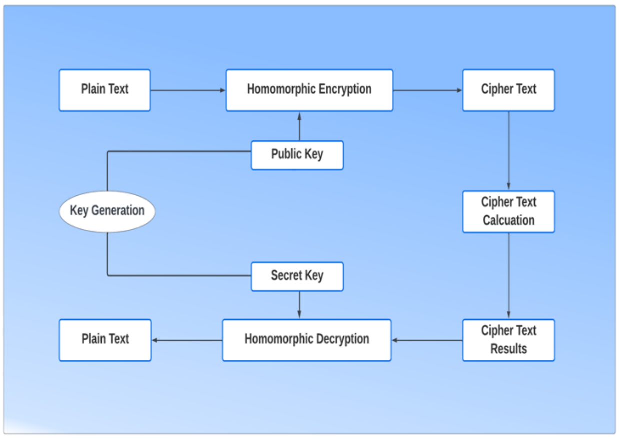


Figure 1: Homomorphic Encryption Process [29]

Two characteristics of the homomorphic encryption technique are additive and multiplicative. An algorithm that can calculate Enc(X1+X2) from Enc(X1) and Enc(X2) without knowing the values of X1 and X2 is said to have an additive property in homomorphic encryption [28]. An algorithm with multiplicative properties can generate Enc(X1*X2) from Enc(X1) and Enc(X2) without needing to know the values of X1 and X2 [28].

The basic architecture of homomorphic encryption is shown in Figure 1 [29].

The main algorithm utilized to represent the system is defined in the next section.

4. Proposed System

This section gives an overview of the proposed homomorphic encryption-based device identification system. To uniquely identify each device when data from different devices is generated and sent to the proposed model, which uses fully homomorphic encryption. This means that the data from these devices will always be in encrypted form throughout the computation. This makes the process of identifying devices safe and secure.

Additionally, in this work, Microsoft Seal is leveraged, which is an open-source and highly optimised HE library developed by the Cryptography Research Group at Microsoft Research [30].

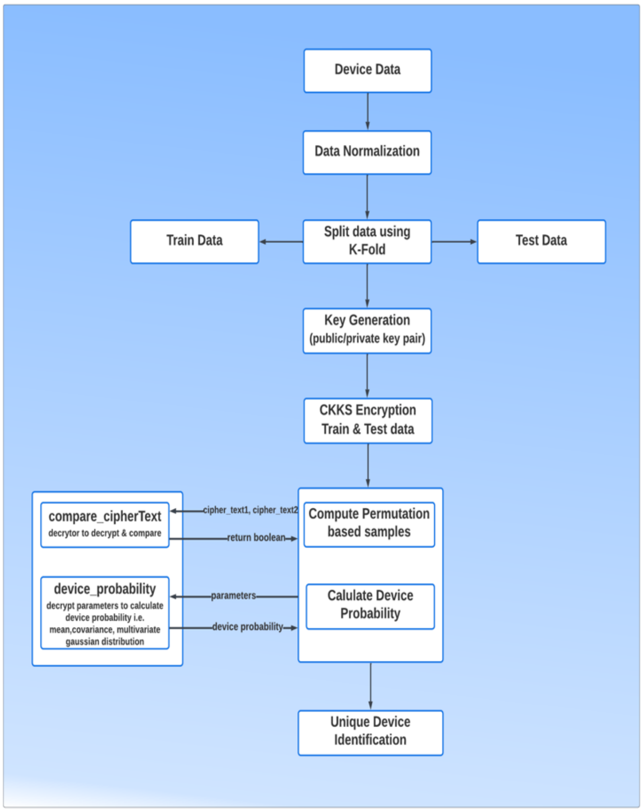

Device Data
Data Normalization
Train Data
Split data using K-Fold
Test Data
Key Generation (public/private key pair)
CKKS Encryption Train & Test data
compare_cipherText
decrytor to decrypt & compare
cipher_text1, cipher_text2
return boolean
Compute Permutation based samples
device_probability
decrypt parameters to calculate device probability i.e. mean,covariance, multivariate gaussian distribution
parameters
device probability
Calulate Device Probability
Unique Device Identification

Figure 2: Proposed system process

The homomorphic encryption library Microsoft SEAL, which also enables additions and multiplications on encrypted integers or real numbers, supports the BFV and CKKS algorithms. Most of the time, using this technology, it is not possible to evaluate other operations on encrypted data, such as encrypted comparison, encrypted sorting, or regular expressions. Therefore, Microsoft SEAL should only be used to build cloud computation components of projects that require privacy. The BFV schemes allow modular arithmetic to be performed on encrypted integers [30].

Figure 2 shows the step-by-step proposed system process. The proposed system consists of three main processes, which are described below: feature extraction, homomorphic encryption-based data protection, and device identification matching results.

4.1 Feature Extraction

This process has been explained in detail [7], where all data was pulled from devices and then used criteria for good features and shortlisted features that have high inter-sample variance and low intra-sample variance, and then normalized the data and split the data using k-fold (k = 10) into training and test data.

4.2 Homomorphic encryption-based data protection

To achieve the goal of data protection from spoofing attacks, HE is leveraged, whereby test and training sample data is encrypted and, at the time of a cryptographic operation (for example, authentication), the encrypted data is used. Thus, the advantage is that spoofing and impersonation attacks can be avoided.

Microsoft offers a variety of APIs to aid in computation and support proposed model. The concept is briefed below.

The first step is to generate a key pair (public and private key) using:

1) **get_seal ->** return the encryptor and encoder. Then encrypt training and test data using the public key [25].
2) **compare_ciphertext ->** decrypt the ciphertext with private_key, compares it, and returns a boolean, i.e., TRUE if ciphertext1 > ciphertext2, False otherwise.
3) **device_probability** -> decrypt the permutation test and training samples, computes the mean, covariance, and multivariant using the private_key and return the probability as a result.

This model makes use of these APIs for data comparison on encrypted values. This helps in achieving data security since the data computation model is only known to work with encrypted values and is described in detail.

4.2.1 CipherText API Comparison

Boolean comparison on cipher text is not straightforward and is not supported in the Microsoft SEAL library either; however, the probability model uses comparison while calculating probability based on input values. One way to implement comparison is to encrypt messages bit by bit and write a comparison circuit; however, this can be very inefficient from both a running time and data expansion point of view. So, our own logic has been enhanced and implemented to build this comparison circuit, which has enhanced the capability of the model to be more secure and robust.

Microsoft SEAL allows additions and multiplications to be performed on encrypted integers or real numbers. A comparison algorithm was built by leveraging the ciphertext value addition capability, and proposed algorithm works as follows:

4.2.1.1 Comparison Algorithm

1. X and Y are two values that are encrypted.
2. Produce a new random number and encrypt it.
3. Perform the calculation.
4. Decrypt the result.
5. Subtract a random number from the result of the decryption.
6. If the results from Step 5 > 0, then X > Y; otherwise, Y > X.

ICMetrics-specific flow is described below.

4.3 Device Identification Results

In the last stage, the results were compared based on the highest probability and used to identify the device uniquely.

4.4 Algorithm for the proposed system

The algorithms below introduce the process of a fully homomorphic encryption-enabled model for device identification.

- Read the data from all the devices.
- Select features based on criteria for good features.
- Normalize the data and then split it using k-fold.
- Calculate the column threshold for training and testing and encrypt the threshold using CKKS FHE.
- Encrypt test and training data.
- Pass this encrypted training and test data to the model, which will return the device probability.
- ○ The model will compute the permutation samples.
- ○ Compute the device probability based on permutation samples after calculating the mean, covariance, and multivariate Gaussian distribution.
- Save the probability produced for each device.
- Repeat the process for all devices to determine the maximum probability generated, which serves as prediction data for device identification.

5. Experiment

The experiment's goal is to assess the proposed system for device identification using homomorphic encryption based on its accuracy and performance.

5.1 Experiment Setup

The section offers a comprehensive examination of the experimental results connected to the suggested model, MVGD. The experimental dataset contains features discussed [7]. The hardware of the MacBook Air and MacBook Pro served as the source of this data (memory, CPU and hard drive). Eight devices running current software were employed. This study made use of data acquired from the MacBook Air and Pro, Python programming, and Microsoft Excel. And each device has 1,000 samples in this research.

The experiment makes use of 17 features that were pre-processed, gathered from common computing devices, and then supplied to the model for training. The selected features were subsequently divided into sets to increase operational robustness. These feature sets offer more natural obfuscation, are more reliable than individual features, and generate a stronger base for applying the ICMetrics system. There are three feature sets: 8F, 6F and 3F. The process of feature selection is explained in detail [7].

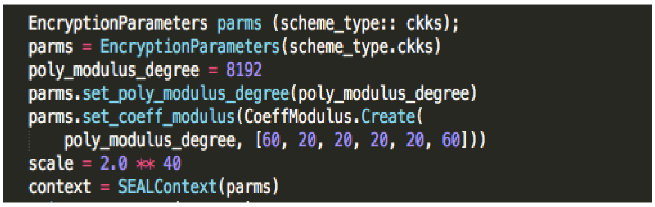
For holdout accuracy estimations, the data were split into two sections: 80% and 20%, respectively. 20% was used for testing after 80% was used for training the model. For validating the results, k-fold cross-validation for k = 10 was employed. For k-fold cross- validation, data was divided into k sections, one of which was used as test data and the remaining k-1 as train data.

In the encryption stage of this experiment, the CKKS homomorphic encryption algorithm provided by the open-source Microsoft SEAL library [30], [31] was utilised to encrypt the data.

Encryption parameters for CKKS are

- n: degree of polynomial modulus
- q: coefficient modulus
- scale: scaling factor for plaintext message inputs

SEAL generates all required parameters using these three parameters.



```
EncryptionParameters parms (scheme_type:: ckks);
parms = EncryptionParameters(scheme_type.ckks)
poly_modulus_degree = 8192
parms.set_poly_modulus_degree(poly_modulus_degree)
parms.set_coeff_modulus(CoeffModulus.Create(
    poly_modulus_degree, [60, 20, 20, 20, 20, 60]))
scale = 2.0 ** 40
context = SEALContext(parms)
```

Figure 3: Encryption Parameters

The first step in setting up the cryptosystem is to select the encryption parameters as outlined in Figure 3. An instance of the class Encryption Parameters contains them all. The three moduli that the encryption algorithm uses are first set: q (coefficient modulus), t (plain modulus), and Xn + 1 (polynomial modulus). The three most vital criteria are these three, and selecting them appropriately is essential for getting the best results.

5.2 Experimental Results

In this section, the device identification performance of the proposed system is evaluated over device feature data. In the proposed system, the encryption is performed on the feature vector.

Table 1: Proposed system's time performance in various operations

Feature set	Key Generation Time	Test Data Encryption Time	Overall computation time
8F	0.463s	50.698s	6376.720s
6F	0.417s	32.934s	1502.186s
3F	0.400s	16.203s	263.586s

In Table 1 below, the time performance of the proposed system is demonstrated. This table captures key generation, test data encryption time, and overall computation time to uniquely identify the device for all three feature sets. It takes about 0.46 seconds to generate the public key and private key for 8F, and the other two feature sets have less key generation time because the other two feature sets contain fewer features than the first one. 8F test data encryption time (50.698s) and overall computation time (6376.720s) are higher than the other two feature sets. The conclusion is that a larger sample value means more information is preserved, which may lead to better device identification accuracy. However, homomorphic encryption, decryption and key generation are the most time-consuming operations of the whole procedure. Therefore, a larger sample size means longer computational time, but data under homomorphic cipher text can have high privacy security. The experiment below was conducted on a MacBook Air with 8 GB of 1600 MHz DDR3 memory and a 1.3 GHz Intel Core i5 processor.

The tables 2, 3, and 4 show results based upon device features for each feature set, respectively. The effects of different parameters on the system's performance in terms of encryption and computational time are explored in the following.

Table 2: 8F HE based encryption and computation time

Devices	Train Data Encryption Time	Computation Time
D0	44.275s	2897.953s
D1	50.160s	1085.930s
D2	40.913s	132.240s
D3	42.024s	91.805s
D4	38.562s	139.640s

D5	37.100s	1006.374s
D6	37.414s	98.050s
D7	37.039s	5831.506s

Table 3: 6F HE based encryption and computation time

Devices	Train Data Encryption Time	Computation Time
D0	28.946s	280.528s
D1	29.636s	196.820s
D2	27.341s	89.374s
D3	27.675s	72.152s
D4	26.612s	44.321s
D5	26.607s	87.765s
D6	26.553s	179.756s
D7	26.549s	297.567s

Table 4: 3F HE based encryption and computation time

Devices	Train Data Encryption Time	Computation Time
D0	14.079s	25.667s
D1	13.554s	20.272s
D2	13.554s	14.751s
D3	13.535s	13.807s
D4	13.601s	13.614s
D5	13.550s	22.755s
D6	13.534s	13.429s
D7	13.536s	13.412s

As can be seen from Tables 2, 3, and 4, the average training encryption time for D0 is 44.275s, which is twice as high as 6F and three times higher than 3F because the multiplication on the cipher text requires more computation time, and in 8F, there is more data and it takes more time to execute. The computation time for D0 is 2897.953s, which is higher than 6F (280.528s) and 3F (25.667s). This is the observation made when test data is verified against training data and test samples are mapped to modes. This multimodal feature process is explained in a paper [7]. Three feature sets were used for analysis, out of which FS1 (8F) train encryption and computation time are the highest as compared to the second and third feature sets. These feature sets include information about disk such as CPU performance and disk read and write operation speeds. This can be unique to different devices, resulting in the highest accuracy achieved from this feature set. Tables 5 and 6 show the HE-based percentage accuracy of 8F and 6F individual devices, respectively. The devices are individually identified according to the findings.

Table 5: 8F HE based encryption-based devices accuracy

Devices	**Accuracy**
D0	91%
D1	90%
D2	90%
D3	96%
D4	90%
D5	88%
D6	89%
D7	90%

Table 6: 6F HE based encryption based devices accuracy

Devices	**Accuracy**
D0	90%
D1	88%
D2	91%
D3	90%
D4	90%
D5	92%
D6	92%
D7	94%

From the experiment results, the observation is that the proposed model using CKKS-based HE takes longer for computation depending on how many features and samples are factored, as shown in the above results. The objective in this paper is to ensure data protection during the process of analysis and device identification.

6. Conclusion

In this paper, an architecture and an implementation of a device identification system in the HE domain were presented and subsequently evaluated experimentally. The system fulfils the data protection objectives. A cryptographic technique called CKKS homomorphic encryption was executed to secure the device data and uniquely identify the device. The proposed model has multimodal features, and for analysis, comparison was used to identify the devices. In order to construct this comparison circuit, our own logic was used, and we calculated all the model's necessary parameters in Section V, which enhanced the capability of proposed model to be more secure and robust. By utilizing HE, the security objectives of a dataset are achieved.

By using CKKS homomorphic encryption, the computational time and device identification accuracy are studied in this paper. According to the experimental results, homomorphic encryption is time-consuming. The future work will focus on efficient homomorphic encryption algorithms to reduce computational time and explore additional applications that can leverage proposed implementation. This will further help to establish the feasibility of the approach.

References

[1] C. Gentry, “Fully homomorphic encryption using ideal lattices,” in

ACM Symp. Theory of Comput., 2009, pp. 169–178.

[2] C. Gentry, "Toward basing fully homomorphic encryption on worst-case hardness," in *Proc. CRYPTO*, 2010, pp. 116–137.

[3] C. Gentry, "Computing arbitrary functions of encrypted data," *Commun. ACM*, vol. 53, no. 3, pp. 97–105, 2010.

[4] Common Vulnerabilities in Password-Based Login, LoginRadius Blog. [Online]. Available: https://www.loginradius.com/blog/identity/common-vulnerabilities-password-based-login/#:~:text=Passwords%20are%20one%20of%20the. [Accessed: Mar. 31, 2023].

[5] R. Hamza, A. Hassan, A. Ali, M. B. Bashir, S. M. Alqhtani, T. M. Tawfeeg, and A. Yousif, "Towards Secure Big Data Analysis via Fully Homomorphic Encryption Algorithms," *Entropy*, vol. 24, no. 4, p. 519, 2022. doi: https://doi.org/10.3390/e24040519.

[6] K. Munjal and R. Bhatia, "A systematic review of homomorphic encryption and its contributions in healthcare industry," *Complex & Intelligent Systems*, 2022. doi: https://doi.org/10.1007/s40747-022-00756-z.

[7] S. Yadav, P. R. Khanna, and G. Howells, "Device Authentication Using Wavelet Based Features," *International Journal for Information Security Research*, vol. 12, no. 1, pp. 1062–1072, 2022.

[8] M. S. H. Talpur, M. Z. A. Bhuiyan, and G. Wang, "Shared-node IoT network architecture with ubiquitous homomorphic encryption for healthcare monitoring," *International Journal of Embedded Systems*, vol. 7, no. 1, pp. 43–54, 2015.

[9] J. Li, D. Song, S. Chen, and X. Lu, "A Simple Fully Homomorphic Encryption Scheme Available in Cloud Computing," in *Proceeding of IEEE*, 2012.

[10] B. Chen and N. Zhao, "Fully Homomorphic Encryption Application in Cloud Computing," in *Wavelet Active Media Technology and Information Processing (ICCWAMTIP), 11th International Computer Conference*, 2014.

[11] Y. Zhang, L. Zhou, Y. Peng, and J. Zhang, "A secure Image Retrieval Method Based on Homomorphic Encryption for Cloud Computing," in *Proceedings of the 19th International Conference on Digital Signal Processing*, 2014.

[12] O. Kocabas and T. Soyata, "Utilizing Homomorphic Encryption to Implement Secure and Private Medical Cloud Computing," in *8th International Conference on Cloud Computing*, IEEE, 2015.

[13] S. Q. Ren, B. H. M. Tan, S. Sundaram, T. Wang, Y. Ng, and C. Victor, "Privacy-Preserving Palm Print Authentication Using Homomorphic Encryption," in *2016 IEEE 14th Intl Conf on Dependable, Autonomic and Secure Computing*, pp. 878-881, 2016.

[14] Nishikant, S. Burande and Prof. S. A. Kahate, "Design Model for Two Server Password Authentication Protocol," *IJCSET*, vol. 11, no. 5, Nov. 2015.

[15] K. Suveetha and T. Manju, "Ensuring Confidentiality of Cloud Data using Homomorphic Encryption," *Indian Journal of Science and Technology*, vol. 9, no. 8, Feb. 2016.

[16] J. H. Im, J. Choi, D. Nyang, and M. K. Lee, "Privacy-Preserving Palm Print Authentication Using Homomorphic Encryption," in *2016 IEEE 14th Intl Conf on Dependable, Autonomic and Secure Computing*, pp. 878-881, 2016.

[17] M. Gomez-Barrero, E. Maiorana, J. Galbally, P. Campisi, and J. Fierrez, "Multi-Biometric Template Protection Based on Homomorphic Encryption," *Pattern Recognition*, vol. 67, pp. 149-163, Jul. 2017.

[18] J. Scheibner, M. Ienca, and E. Vayena, "Health data privacy through homomorphic encryption and distributed ledger computing: an ethical-legal qualitative expert assessment study," *BMC Medical Ethics*, vol. 23, no. 1, 2022. doi: 10.1186/s12910-022-00852-2.

[19] A. V. Kumar, M. S. Sujith, K. T. Sai, G. Rajesh, and D. J. S. Yashwanth, "Secure Multiparty computation enabled E-Healthcare system with Homomorphic encryption," in *IOP Conference Series: Materials Science and Engineering*, vol. 981. IOP Publishing, 2020, p. 022079.

[20] R. Bocu and C. Costache, "A homomorphic encryption-based system for securely managing personal health metrics data," *IBM Journal of Research and Development*, vol. 62, no. 1, pp. 1–1, 2018.

[21] X. Wang and Z. Zhang, "Data division scheme based on homomorphic encryption in WSNs for health care," *Journal of Medical Systems*, vol. 39, no. 12, pp. 1–7, 2015.

[22] M. Kara, A. Laouid, M. A. Yagoub, R. Euler, S. Medileh, M. Hammoudeh, A. Eleyan, and A. Bounceur, "A fully homomorphic encryption based on magic number fragmentation and El-Gamal encryption: Smart healthcare use case," *Expert Systems*, 2021, p. e12767.

[23] M. S. H. Talpur, M. Z. A. Bhuiyan, and G. Wang, "Shared-node IoT network architecture with ubiquitous homomorphic encryption for healthcare monitoring," *International Journal of Embedded Systems*, vol. 7, no. 1, pp. 43–54, 2015.

[24] H. Tan, P. Kim, and I. Chung, "Practical homomorphic authentication in cloud-assisted vanets with blockchain-based healthcare monitoring for pandemic control," *Electronics*, vol. 9, no. 10, p. 1683, 2020.

[25] A. Ali, M. F. Pasha, J. Ali, O. H. Fang, M. Masud, A. D. Jurcut, and M. A. Alzain, "Deep Learning Based Homomorphic Secure Search-Able Encryption for Keyword Search in Blockchain Healthcare System: A Novel Approach to Cryptography," *Sensors*, vol. 22, no. 2, p. 528, 2022.

[26] M. Albrecht, M. Chase, H. Chen, et al., "Homomorphic encryption standard," *Cryptology ePrint Archive*, 2019.

[27] K. Lauter, M. Naehrig, and V. Vaikuntanathan, "Can homomorphic encryption be practical?," in *Proc. ACM Cloud Comput. Security Workshop*, 2011.

[28] I. Jabbar and S. Najim, "Using fully Homomorphic encryption to secure cloud computing," *Internet of Things and Cloud Computing*, vol. 4, no. 2, pp. 13-18, 2016. doi: 10.11648/j.iotcc.20160402.12.

[29] S. Yue, "Fully Homomorphic Encryption Part One: A Gentle Intro," *Medium*. [Online]. Available: https://stevenyue.medium.com/fully-homomorphic-encryption-part-one-a-gentle-intro-94c3c3850568. [Accessed: Oct. 21, 2023].

[30] Microsoft Research, "Microsoft SEAL: Fast and Easy-to-Use Homomorphic Encryption Library," [Online]. Available: https://www.microsoft.com/en-us/research/project/microsoft-seal/.

[31] Microsoft Research, "Microsoft SEAL (release 3.2)," Microsoft Research, Redmond, WA, 2019. [Online]. Available: https://github.com/Microsoft/SEAL.