How Can Homomorphic Encryption Be Used In Healthcare Industries

MSc Research Project
MSc Cybersecurity

Suchin John
Student ID: 23218266

School of Computing
National College of Ireland

Supervisor: Liam Mccabe

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Suchin John		
Student ID:	23218266		
Programme:	MSc Cybersecurity	**Year:**	2025
Module:	MSc Practicum/Internship part 2		
Supervisor:	Liam Mccabe		
Submission Due Date:	29/1/2025		
Project Title:	How Can Homomorphic Encryption Be Used In Healthcare Industries		
Word Count:	7637	**Page Count:**	20

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: SuchinJohn

Date: 29/1/2025

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Table of Content

Table of figures

Table of tables

How Can Homomorphic Encryption Be Used In Healthcare Industries

Suchin John
23218266

Abstract

The healthcare industry faces challenges in protecting sensitive data amidst rising cyber threats. Fully Homomorphic Encryption (FHE) offers a transformative solution by enabling computations on encrypted data without decryption. This ensures privacy throughout data handling. This study explores the practical implementation of FHE in healthcare, leveraging the Microsoft SEAL library to create a secure framework for encrypted data analytics and sharing. The framework integrates BFV and CKKS encryption schemes for integer and real-number computations, respectively. It is tested using synthetic datasets simulating healthcare scenarios. Evaluations demonstrate high accuracy, robust noise management, and scalability for moderate dataset sizes. Results from encrypted computations align closely with plaintext benchmarks validating FHE's effectiveness in privacy preservation. While challenges such as computational overhead and noise depletion in complex operations remain, this research underscores the potential of FHE to secure healthcare data, meeting HIPAA standards and enabling safe, efficient data usage. Future work will focus on scalability and optimization strategies.

Key words – Fully homomorphic encryption (FHE), noise, BFV, CKKS, Microsoft SEAL

1 Introduction

The healthcare industry in particular manages sensitive data including patient health records, financial information and personally identifiable information. These data are classified as Protected Health Information (PHI) Under the HIPAA act. Therefore, ensuring the protection of this data is of utmost priority. While advanced technologies such as Artificial Intelligence and Machine Learning have improved efficiency in handling tasks which include medical imaging analysis, streamlining administrative workflow and faster diagnosis. However, there is an alarming increase in global attacks on data, with healthcare organizations being the major targets. In 2023 alone, 725 data breaches affecting 500 or more health records were reported—more than double the number in 2017 (Alder 2024). Such trends are alarming, and it emphasizes the need to develop not only powerful, fresh solutions that will not only keep sensitive healthcare data safe but also allow for its safe use for essential processes and developments.

This paper investigates the practicalities of using Fully Homomorphic Encryption (FHE) as a transformative technology to address critical privacy and security challenges. Unlike traditional encryption methods, where data must be decrypted before processing, FHE allows computations to be performed directly on encrypted data. This fundamental difference ensures that sensitive information remains encrypted throughout its lifecycle, significantly reducing the risk of data leakage and unauthorized access. FHE has the potential to revolutionize the security of private healthcare data, enabling privacy-preserving data analytics, sharing, and collaboration. Critical health data, such as heart rate and blood pressure—classified as protected health information (PHI) under HIPAA can benefit from this advanced approach to encryption. By maintaining privacy and security without compromising functionality, FHE represents a paradigm shift in the management and utilization of sensitive information.

In 2020, health entities experienced disproportionally large amount of data breaches compared to other industries: the average number of daily breaches in the healthcare industry is 1.76. HIPAA prescribes very strict rules on protecting health records and other healthcare sensitive data from unauthorized access, yet many health entities fail to implement the security controls (Kost, 2022). **The research question that motivates this project is**: How can Homomorphic Encryption be used in Healthcare industries? This work delves into methodologies for designing and deploying FHE, its integration into existing systems, and evaluates its effectiveness in real-world applications.

1.1 Report structure

Literature on research that has been done on existing FHE research, its applications in healthcare and the challenges and benefits that were observed in prior studies has been reviewed. This review provides context and identifies gaps in the current understanding of the potential for FHE in healthcare. The Methodology section details the research approach and steps employed to develop a practical implementation framework for FHE that is not only technically feasible, but also fits within the specific requirements of healthcare organizations. The Implementation and Design Specifications section presents detailed specifications of the artifact created, including the system architecture, properties of encryption protocols, and workflow, describing how FHE could be fully incorporated into healthcare data systems. The Evaluation section evaluates the implementation by identifying different metrics like performance which are crucial to the working of the project and tests to see how efficient they are, and at the end, the Conclusion summarizes the result, assess the potential significance of FHE in healthcare, and suggests possible future research and development directions.

2 Related Work

A cryptographic technique known as Fully Homomorphic Encryption (FHE) makes it possible to perform computations over encrypted data without decryption. F Armknecht published a paper (Armknecht et al., 2015) outlines how FHE can be used in a range of different possibilities and applications, e.g. privacy-preserving advertising, secure cloud processing, medical data analysis as well as financial data securities and applications where

FHE could be implemented. From this I gained an idea on the different platforms I could possibly implement this project. In Bansal's paper (Bansal, 2021) he explains how this technology can be used in different areas. Like in cloud computing, processing can be done securely without exposing plaintext. In the case of electronic voting, HE (Homomorphic Encryption) is used to protect ballot privacy, and integrity while it allows us to have secure ranked choice voting systems like ElGamal. The IoT (Internet of Things) device's massively generated data by interconnected devices is secured in HE. Research into the different areas in healthcare where privacy and data protection takes precedence was carried out. This gave rise to the main research question that is - how can Fully Homomorphic Encryption be used in Healthcare industries.

Marcolla in his paper (Marcolla et al., 2022), discussed the security of FHE and revises concepts such as number theory and lattice based cryptography. It also discusses various generations of FHE. Third generation FHE methods have been incorporated in this project to optimize its performance with batching and modulus switching. The paper also covers opensource tools such as Helib and SEAL to implement FHE. The Microsoft SEAL library was further researched from the paper written by Fransisco (Fransisco Jose, Francisco-Jose Valera Rodriguez, Manzanares Lopez and Cano, 2024). The paper focused on the use of Microsoft SEAL on PC, as an efficient option dependant on dataset size and complexity poly modulus degree encryption. The paper produced by Fawaz (Fawaz et al., 2021), explained the different possibilities and applications where FHE could be implemented, such as privacy-preserving advertising, secure cloud processing, medical data analysis, and financial data security. These papers provided insights on the different platforms the could be adopted for the implementing this project.

2.1 Comparison

There exists of many FHE libraries each with their own strengths and weaknesses. Here we compare the different opensource FHE libraries and explain why this project uses the Microsoft SEAL library. Zhu wrote a paper (Zhu et al., n.d.) comparing Microsoft SEAL and OpenFHE and it discusses the reasons why the Microsoft SEAL library was selected. Compared to its counterpart, Microsoft SEAL performed consistently better in all regards to latency. It was better optimized in memory management so the overhead was minimised. But it also performed better for people who were able to tweak the library well enough. After reading this paper, I dug some more into Microsoft SEAL paper which led to another published paper of Zhu et al(Zhu, Suzuki and Hayato Yamana, 2023) comparing Microsoft SEAL, OpenFHE and HElib. Just like this paper, Microsoft SEAL was shown to be more latent than OpenFHE and Helib. I also found that out of the three, SEAL gave excellent performance with more threads keeping it most scalable. This further led me to do further research into comparing Microsoft SEAL with other libraries. Microsoft SEAL, Helib, and PALISADE are compared in a paper by Alycia Carey (Carey, 2020). This paper then discusses the underlying encryption schemes used by the 3 libraries—BFV, CKKS and BGV. The thesis highlights the promise of FHE in changing data security by enabling computations on encrypted datasets. This justified my decision to choose Microsoft SEAL as my FHE library. In the final paper that I mentioned in this section, it compared the 2 different types of encryption schemes that Microsoft SEAL utilizes: BFV, and CKKS. BFV is designed to support computations on integers and addition, multiplication, and squaring operations. CKKS is designed to work with approximate, real number computations which is ideal for applications such as data analytics and machine learning. Basic operations (e.g., addition and multiplication) performed by BFV were more time intensive with small to medium vector sizes, however CKKS had faster time for larger vectors at the cost of more computational overhead. I learned from this paper that BFV scheme is very suitable with applications involving integer arithmetic and in contrast CKKS is more suitable with applications with real value data in scientific computations (Fawaz et al., 2021). This allowed me to choose the encryption scheme that works best for this project. As Microsoft SEAL had access to both encryption schemes, it became more evident as how well it can be incorporated into my project which tries to answer the question of how fully homomorphic encryption can be used in healthcare industries.

2.2 Drawbacks and Mitigations

During my research on related work, I found papers that pointed out the limitations of the Microsoft SEAL libraries, and study by Zhiniang Peng (Peng, 2019). In this paper it identified critical issues, such as gaps in circuit privacy that may allow unintentional disclosure of sensitive details involved in encrypted computation.
Moreover, Peng's work proposed practical countermeasures like noise flooding and re randomization, to combat such challenges and enhance security. Motivated by these recommendations, this project provides means to improve privacy against these attacks while explicitly addressing these previously mentioned shortcomings. With these enhancements incorporated, this work shows how SEAL's limitations can be mitigated successfully, and how SEAL can be used with a strong security framework.

Another paper that really stood out was a theoretical paper on how FHE could be deployed in healthcare. The paper was a purely theoretical. Certain tools like Microsoft SEAL make the implementation easier despite the presence of computational overhead, absence of standards, and ethical issues (Fotios Roumpies and Athanasios Kakarountas, 2023). The goal of this project is to give a practical implementation to the above paper.

3 Research Methodology

The research procedure and methodology focus on designing and implementing Fully Homomorphic Encryption (FHE) within the healthcare context. It aims to design a practical and efficient framework for integrating FHE into healthcare systems in a way that guarantees the privacy of sensitive patient data and allows computations to be performed while maintaining privacy.

The research began with defining clear objectives centred on addressing the question: How can Fully Homomorphic Encryption (FHE) can be deployed in the healthcare industry? Specifically, the project scope was concerned with assuring the security of Protected Health Information (PHI) following the HIPAA rules, including patient metrics (e.g. heart rate, blood pressure), and provide encrypted data analytics. First, there was preliminary research, a feasibility study, consisting of reviewing the existing literature and comparing FHE libraries to figure out the best possible tools. It was found that Microsoft SEAL provided optimal performance, having demonstrated scalability, aligning with the requirements of healthcare data processing. The research then moved towards a design for a framework that incorporates FHE into healthcare workflows in general, especially in cases such as encrypted patient data analysis and secure data sharing between providers.

The chosen FHE software library for implementation is Microsoft SEAL in order to complete the goals of the project. Extensive benchmarking led to this decision and showed that the performance of Microsoft SEAL in latency, scalability, and memory optimization was superior to that of solutions such as OpenFHE and HElib. Two encryption schemes within Microsoft SEAL were utilized: for computations working with integer data, such as patient IDs and medical codes, the BFV scheme is used, while for arbitrary precision real number computations in analyzing the trends in patient metrics, CKKS scheme will be utilised. C++ was used to develop the framework so that it integrates smoothly with Microsoft SEAL. Datasets for this project were generated synthetically to produce realistic (but synthetic) datasets mimicking realistic healthcare settings, and also to ensure compliance with relevant ethical standards conducive for a robust testing environment. Numerical metrics, such as blood pressure readings, and heart rate, were included in the data. Each of these datasets were encrypted using Microsoft SEAL, optimized at the configurations for each dataset to optimize size of the dataset and its complexity. Polynomial modulus degree and plaintext modulus are tailored encryption parameters for the application of healthcare data. The data was encrypted end-to-end and encrypted computations were performed directly on the data, such as aggregation and anomaly detection. In the end, the decrypted outputs were compared to plaintext computations to show that all outputs were accurate and faithful.

Based on dimensions like, performance, accuracy, robustness, and security, these metrics were analyzed to evaluate the implementation. Performance analysis involved measuring the time to encrypt, decrypt, and to run encrypted computations, as well as memory usage and total system latency. Validation of accuracy involved comparing the outcome of encrypted computation over encrypted data with the result from plaintext computation by using parameters like Mean Absolute Error (MAE) and Root Mean Square Error (RMSE). Robustness tests to quantify the effect of changing different encryption parameter configurations (e.g. polynomial modulus degree, plaintext modulus) on computation time, accuracy, and noise budget, and suggested parameter configurations optimal for healthcare datasets were run. System performance under increasing dataset size and under multi-threaded configurations was assessed through scalability testing and at the same time, the framework was shown to scale up appropriately with large healthcare datasets.

Security assessments have been carried out for checking the adherence to the privacy principles on the implementation. To protect against vulnerabilities, certain concepts such as the integration of MPC (Multi-Party Computation) principles were discussed. Noise budget was also tracked to verify the system's robustness in performing complex computations. This project was driven by ethical considerations. Synthetic datasets were used to avoid handling of real patient data at all and to guarantee the compliance with HIPAA and other data privacy regulations.

4 Design Specification

The design of the Fully Homomorphic Encryption (FHE) framework for healthcare systems was guided by the need to secure sensitive patient data while enabling privacy-preserving computations.
Existing literature was reviewed and existing FHE libraries had been compared to perform preliminary research and a feasibility study to identify the most suited tool. The performance, scalability, and suitability of Microsoft SEAL proved to be the best option when the criteria above are applied to healthcare data processing. The research then explored how to design a framework integrating FHE into healthcare workflows in situations such as encrypted patient data analysis and secure data sharing between providers.

Microsoft SEAL was the FHE library that was chosen to reach project goals. Based on such extensive benchmarking, it was observed that the performance of this decision in terms of latency, scalability and memory optimization is better than those of any OpenFHE or HElib. Two encryption schemes within Microsoft SEAL were utilized: For computations with integer data — such as patient IDs, medical codes — the BFV scheme is used, and for approximate real number computations — useful for trending patient metrics — the CKKS scheme is used. The framework was developed using C++, and Microsoft SEAL. For this project, the datasets were synthetically generated to conform to realistic healthcare scenarios, as well as to a rigorous set of ethical standards and serve as an ideal testing environment. The numerical metrics were included in the data, such as blood pressure readings and heart rate. The datasets were encrypted using Microsoft SEAL with configurations optimized to the dataset size and complexity. Polynomial modulus degree and plaintext modulus were tailored to the parameters of healthcare data encryption. Encrypted computations were performed on the data, directly, using aggregation and anomaly detection operations, where the data itself stays encrypted across the whole workflow. Ultimately, the outputs were decrypted and compared to plaintext computations to ensure accuracy and fidelity.

The different metrics analysed across different dimensions are performance, accuracy, robustness, scalability. These were used to evaluate the implementation. Specifically, performance analysis was done through measuring the time for encryption, decryption, and computation on the encrypted data, and measuring the memory usage and overall system latency. Accuracy validation was achieved through the comparison of the results of the encrypted computation operations to the results of the same operations against plaintext data with metrics including Mean Absolute Error (MAE) and Root Mean Square Error (RMSE). The impact of varying encryption parameter configurations (e.g. polynomial modulus degree, plaintext modulus) was also investigated and evaluated with respect to computation time, accuracy and noise budget with regards to robustness testing, and also optimal configurations for healthcare datasets were identified. System performance under increasing dataset sizes and multi-threaded configurations was also evaluated via scalability testing to show the framework's scalability when processing larger healthcare datasets.

This document defines the techniques, architecture, and framework that was used to implement Fully Homomorphic Encryption (FHE) to ensure security of sensitive healthcare data. It offers in-depth description of the system components, the data handling methodology, the encryption framework and computational workflow in ways that meet the strict privacy and security standards that are to be practiced by healthcare industry.

4.1 Framework Architecture

Fully Homomorphic Encryption (FHE) is implemented using the Microsoft SEAL library in an implementation framework. To ensure privacy preserving computations and to give a scalable and seamless integration, this design incorporates the key elements of FHE. For computations on integer data (for example, patient IDs and medical codes), the BFV scheme is used; and for approximate computations with real numbers (for example, trends in heart rate and blood pressure), the CKKS scheme is used.

The flow of data during the whole process is as follows:
Data Collection: Realistic healthcare scenarios are simulated using data from simulated patient reads of systolic blood pressure and heart rate.
Data Encryption: Microsoft SEAL's batch encoding feature is used to encrypt patient data so that the vectorized operations on encrypted data are efficient. This data is then stored to be used later as needed.
Encrypted Computation: Computations, like aggregation, anomaly detection, correlation analysis are typically performed directly on encrypted data.
Decryption: The final result after the encrypted computation is then decrypted and displayed to the doctor who requested it.

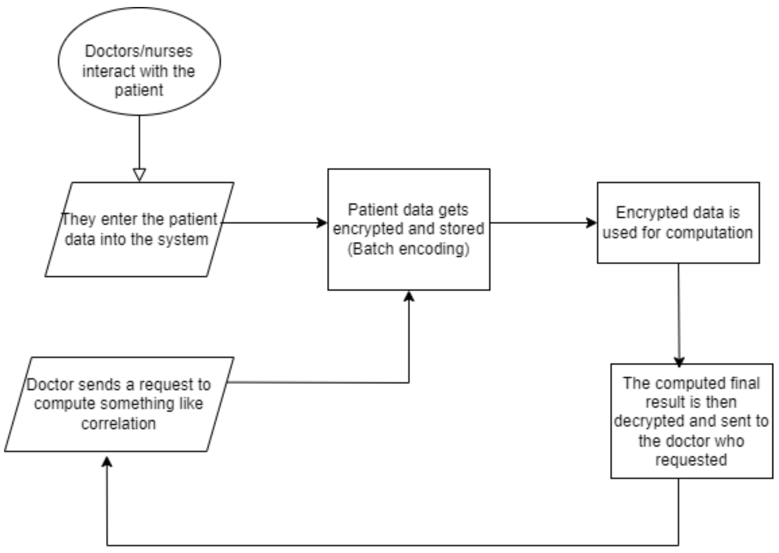

Doctors/nurses interact with the patient
They enter the patient data into the system
Patient data gets encrypted and stored (Batch encoding)
Encrypted data is used for computation
Doctor sends a request to compute something like correlation
The computed final result is then decrypted and sent to the doctor who requested

Fig 1.1 flow of control

4.2 Requirements

In regards to implementing Fully Homomorphic Encryption (FHE) for secure healthcare data computation there are specific hardware needs for efficient performance. Outlined below are the hardware requirements to support FHE's computationally intensive operations, namely encryption, decryption, and homomorphic computation.

Minimum Hardware Requirements for FHE Implementation:

- Processor: Quad-core CPU, 2.5 GHz or higher (e.g., Intel Core i5 8th Generation or newer, AMD Ryzen 5).
- Memory: At least 16 GB RAM to handle large ciphertexts and polynomial operations.
- Storage: Minimum 512 GB, preferably SSD for faster read/write access and handling intermediate computation files.

Recommended Hardware for Optimal Performance:

- Processor: 8-core CPU, 3.0 GHz or greater (e.g., Intel Core i7/i9, AMD Ryzen 7/9).
- Memory: 32 GB or more RAM for enhanced performance.
- Storage: 1 TB or greater NVMe SSD for maximum speed and capacity.
- Parallel Computing: Support for multi-threading and GPUs, preferably dedicated GPUs such as NVIDIA GPUs for accelerated computations.

5 Implementation

The FHE implementation for security of healthcare data is in the form of encrypting patient metric shares such as the blood pressure and heart rate into encrypted shares which can be processed while preserving the strict data confidentiality. Robust cryptographic library Microsoft SEAL was chosen for its support for BFV and CKKS schemes, efficient batch encoding and optimized performance. Both these schemes are considered to be quantum resistant and are widely used in homomorphic encryption schemes designed for privacy preserving computations. For performance reasons, the system was built in C++, leveraging Microsoft SEAL's developed capabilities, while being compatible with healthcare workflows.

Initial step of the implementation consists of data encryption, as synthetic patient datasets obtained from the previous stage are preprocessed to create realistic healthcare scenarios. Using Microsoft SEAL's batch encoder, each share is encoded into vectors and metrics are divided into shares utilizing modular arithmetic. Using, in the setup phase, the public key, the encoded vectors are thus encrypted. Carefully chosen encryption parameters (e.g., polynomial modulus degree and coefficient modulus) are shown to be optimized to balance security and performance, so that the system can perform complex calculations efficiently. After encryption the data is directly subject to homomorphic computations in its encrypted state. Aggregate sums, cross-products, etc., are computed by performing operations such as addition, multiplication, etc., on variance terms and covariance terms. These are the workloads needed to analyze trends, detect anomalies or calculate the correlation between the systolic blood pressure and heart rate to keep the ciphertext sizes and numerical accuracy at manageable levels.
Fidelity validation is performed on the decrypted outputs. The computation results, such as aggregated metrics and computed correlation coefficients, are also compared with those of plaintext computations and are shown to have negligible errors and demonstrate that the system is accurate. The security of circuit output is further increased through the use of noise flooding and re–randomization techniques to protect the circuit from various holes where the leakage of information can take place, thereby meeting the data privacy regulation such as HIPPA.
In order to overcome performance difficulties, the system implements multi-threading to improve computational speed, and scaling to various degrees to deal with large amounts of data sets. The framework's effectiveness is evident in the alignment of results from encrypted computations with plaintext benchmarks. The resulting implementation provides a practical solution for efficient and secure privacy preserving healthcare data analysis.

6 Evaluation

The evaluation of the proposed approach was conducted across four critical dimensions: performance, accuracy, noise budget, and scalability. They capture each part of a system's overall efficacy and resilience, but from a distinct yet interconnected perspective. To assess the system's performance, performance tests were designed to measure its efficiency under different operational conditions and metrics captured include: execution time. It is crucial to know about this criterion, in order to understand how the system works under normal and high demand situation. The accuracy of the system was evaluated on the basis of its ability to deliver the correct results with high precision. This analysis compares the system's output with ground truth or reference benchmarks to illuminate the system's consistency and to inform whether it is meeting its expected performance in real applications. Analysis of the noise budget in the system was used to explore cryptographic integrity of the system, including analysis of the noise budget within the Microsoft SEAL Fully Homomorphic Encryption (FHE) framework. In this evaluation we examined how the system treats the FHE noise budget during operations by assessing its ability to perform encrypted operations correctly while accumulating noise. The scalability assessment examined the system in its ability to handle rising workloads, greater complexity, or more data volume. This criterion helps to determine how the system will perform and work when it comes to dealing with larger or more complex tasks that ensure the feasibility in wider areas of applications.

6.1 Experiment 1 / Performance Analysis

The performance evaluation centers on the computational efficiency of the operations of the Fully Homomorphic Encryption (FHE) pipeline using Microsoft SEAL. Performance characteristics are evaluated by measuring the amount of time required for key aspects such as encryption, homomorphic evaluation and decryption. Data sets encoded manually representing systolic blood pressure and heart rate values are used to assess encryption performance. In encryption, plaintext data is converted into ciphertexts using the public key, in an extremely secure way. while this guarantees security in FHE, it is a computationally intensive step. The operation is performed and the amount of time this takes is measured to understand the overhead of securing data at the input stage. Homomorphic evaluation performance is one of the key foci of this program. The encrypted data points are summed up here completely in the encrypted domain. It is then rotated using the Galois keys and adds the values iteratively to aggregate the data as the encrypted rows. In this step, the FHE scheme is simulated in real world encrypted computation, and demonstrates the efficiency and scalability of the FHE scheme. Evaluating the performance of decryption is done by decrypting the computed sums of systolic blood pressure and heart rate values. In this step, the time needed to decode the ciphertexts into plaintexts with the secret key is measured, plus the decoding overhead is used for accessing the final result. To see the results from encrypted computation quickly, it is vital to understand the decryption time. The lapsed time captured is in miliseconds to see what the overall performance of the encryption pipeline looks like. The evaluation analyzes these metrics to reveal that Microsoft SEAL is a practical tool to handle real world encrypted data processing jobs, and to highlight where it might need to be optimized in order to scale FHE based applications.

6.2 Experiment 2 / Accuracy

For this evaluation criteria, the code performs accuracy testing where results from computation in encrypted domain are rigorously compared with those in plaintext domain. The program calculates three key metrics: average systolic blood pressure (BP), average heart rate and average correlation coefficient between these two datasets. The final results are obtained by decryption, resulting from homomorphic computation of these metrics via encrypted summation, squared, and cross product operations. At the same time, the same metrics are computed on the plaintext data as regular benchmarks. The program measures absolute errors between encrypted and plaintext results for each metric to evaluate accuracy of encrypted computations. To give a more detailed and standardized evaluation, two error metrics widely used among researchers—Mean Absolute Error (MAE), Root Mean Square Error (RMSE)—are computed. A simple measure of the closeness of the encrypted result to the plaintext benchmark is provided by average magnitude of errors. The other, RMSE, squares, and then averages the differences, which penalizes big inaccuracies. Together, these metrics gives us a complete picture for fidelity of the FHE computations. The addition of MAE and RMSE helps because it provides more than just a measure of how far off the average is. It also provides an idea of how widely distributed the deviations are and the range of deviants. It is especially critical in cases where the smallest errors made in a sensitive computation like in healthcare data would have far-reaching implications. These metrics are then combined with absolute and relative error analysis to show that encrypted computations compare very favorably in terms of accuracy and precision. To achieve practicality of the system for privacy preserving applications without sacrificing the integrity of the results, this is guaranteed. These evaluations establish the FHE framework's reliability and provide a stepping stone for its use in real world especially in privacy sensitive domains.

6.3 Experiment 3 / Noise analysis

A vital metric in homomorphic encryption is noise budget which describes the amount of noise that the system can tolerate from the encrypted data for computation, and is also crucial for achieving correct decryption. In the BFV encryption scheme for instance, noise (an arbitrarily small and unlikely to be picked random number) is introduced in order to secure the plaintext by obfuscating it, but every operation (such as addition, multiplication, rotation or relinearization) consumes one out of the limited noise budget, leaving a lower amount remaining to work with. It is necessary to evaluate the noise budget to the reliability of encrypted computations because if the noise level exceeds a critical threshold, decryption becomes impossible or introduces errors. Monitoring the noise budget serves a considerable number of purposes, including computing how deep the system can go computationally, ensuring that the intermediate result still stays under the cover of the lattice, and also allowing us to analyze how well techniques such as modulus switching and relinearization work. In addition, it offers recommendations for adjusting encryption parameters like the polynomial modulus degree, coefficient modulus, and plaintext modulus, so that security, performance, and function are reasonable. The decryptor.invariant_noise_budget(ciphertext) method displays the noise budget in the implementation at key points along the way in the computation process. As an example, this experiment show the noise budget after encryption, after the squaring operations, after re-linearization, after multiplications, and accumulate the sums within operations such as rotations. Real-time insights into the contributions of these operations in the noise budget are displayed, not only notifying you if operations are not within safe limits but also indicating what operation is having an impact and its contribution of that operations to the overall noise budget. You continuously validate that encryption parameters are sufficient. We also keep checking the computations remain robust and exact, by constantly reporting the noise budget where noise is introduced. In particular, this is critical for practical applications like privacy preserving analysis of medical data. Possible operations like the correlation analysis needs to be done but not at the cost of losing the ability to be decrypted. A cornerstone of secure and efficient homomorphic encryption is noise budget analysis: these evaluations ensure correctness, and provide an avenue for performance optimization.

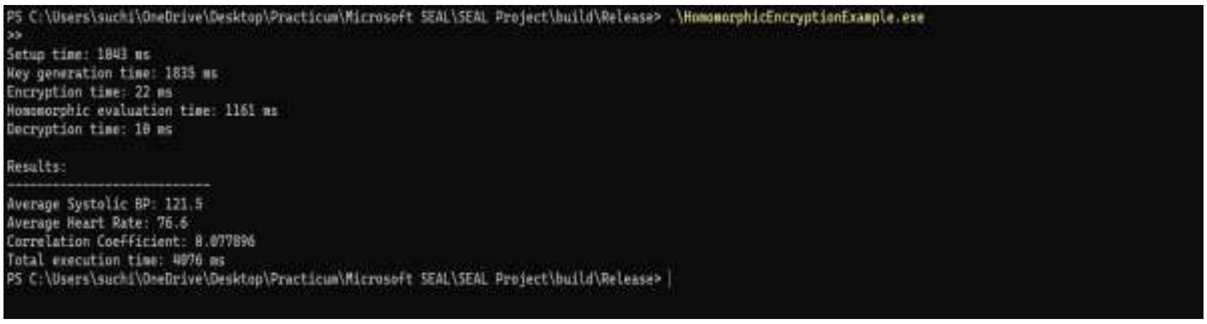
6.4 Experiment 4 / scalability

Scalability means that a system can handle increasing workloads or dataset sizes without considerable degradation with respect to accuracy and performance. In the homomorphic encryption context, scalability must be evaluated since it decides whether the system can compute secure operations within reasonable time when data volume increases. Scalability testing detects bottlenecks in the most important operations (which, in this model includes encryption, evaluation, and decryption) and guarantees that the system will be viable for real world use with massive datasets. A system without scalability may work well for small datasets but is likely to choke under heavier, more complex workloads. This is because the times for computation, systems total available resources, and noise budget are not dataset dependent. To analyze scalability, performance and noise budget tests were performed on datasets with different sizes (which correspond to 8, 30, 60, 90 and 120 elements) in this project. Setup time, key generation time, encryption time, homomorphic evaluation time, decryption time, total execution time were measured during performance testing. The experimental results confirmed that the encryption and decryption times as well as setup and key generation times scale well and are not dependent on the size of the dataset. However, it was found that the times for homomorphic evaluation grew significantly as dataset sizes increased, indicating a scalability bottleneck. In particular, rotations, summations and multiplications were proved to take the majority share of this overhead due to their computational intensity. Insights into the noise efficiency of the system were obtained via noise budget testing of the algorithms in increasing dataset size and computational complexity. Overall, it was found that the initial noise budget stayed the same across all datasets, but that operations like squaring or summation consume more noise with larger datasets. Deeper computations such as summation of squares suffered much greater noise depletion while final noise budgets for simpler operations such as summation remained stable. These tests show there are significant benefits to optimizing across operations to improve performance while maintaining acceptable noise levels for complex workflows. Through objective performance and noise evaluation metrics, the project demonstrates the system’s strengths in scaling initialization and simple operations, and identifies areas for improvement in initialization of larger datasets.

6.5 Discussion

PERFORMANCE

A performance test was run on a dataset of 20 elements, which gave a good indication that the homomorphic encryption system works efficiently and is scalable. In the test with 20 elements, the setup phase was completed in 1043ms and the key generation phase in 1835ms, similar to the results in the test. The time spent in these phases, is significantly less than that of the overall time running on 4120ms. Encryption was completed in 22ms, and the decryption stage was very efficient at 10ms. Operations such as summation, multiplication, and rotations are considered as homomorphic evaluation phase that consumed 1161ms proportional to the smaller dataset. An average systolic blood pressure of 121.5, an average heart rate of 76.6 and a correlation coefficient of 0.077896 was obtained as the results produced from the encrypted computations. This show that the system's performance in setup and key generation times is consistent and insensitive to dataset size. However, we can see that the increase in homomorphic evaluation time can be quite significant, which highlights the cost of working with larger datasets. Both rotations and re-linearizations form a significant part of this cost, and as datasets become larger, such operations become more cumbersome. Although the amount of computation is increased, the correlation coefficient was accurate and dependable, showing how the scheme handled the expanded dataset without any loss of computational correctness. The system appears to be able to securely and accurately scale to modestly larger datasets, and the increasingly expensive homomorphic evaluation time presents a worry regarding its scalability for much larger datasets or more complex operations. The positive aspects of a static runtime, with its consistent setup and key generation times, comes at the cost of having a high runtime contribution for reinitializations and dynamic updates of the application. As the cost of homomorphic evaluation of large datasets increases, scalability can be adversely affected for real world applications with hundreds of thousands of data points. Furthermore, the key generation time is high (though consistent) and hence a bottleneck, particularly for use scenarios where key changes have to occur dynamically. Optimizing costly operations like rotations and multiplications through advanced techniques, such as lazy relinearization or efficient rotation key management, could significantly reduce evaluation time. Exploring parallelization or GPU acceleration for evaluation operations may further enhance performance. Experimenting with alternative parameter configurations, such as adjusting plaintext moduli or coefficient moduli, could help balance precision and computational efficiency.

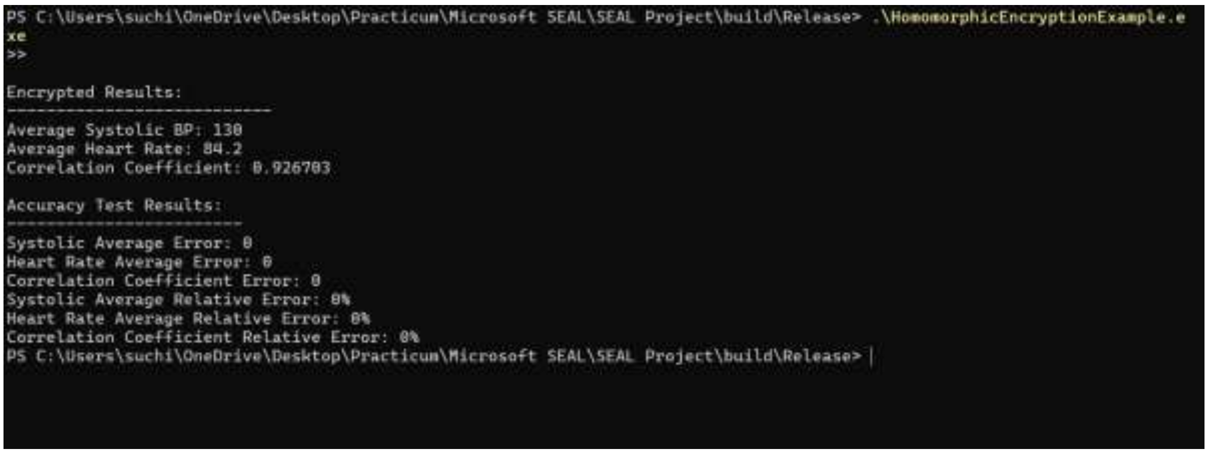


```
PS C:\Users\suchi\OneDrive\Desktop\Practicum\Microsoft SEAL\SEAL Project\build\Release> .\HomomorphicEncryptionExample.exe
>>
Setup time: 1043 ms
Key generation time: 1835 ms
Encryption time: 22 ms
Homomorphic evaluation time: 1161 ms
Decryption time: 10 ms

Results:
----------------------------
Average Systolic BP: 121.5
Average Heart Rate: 76.6
Correlation Coefficient: 0.077896
Total execution time: 4076 ms
PS C:\Users\suchi\OneDrive\Desktop\Practicum\Microsoft SEAL\SEAL Project\build\Release> |
```

(Fig. 2) Performance

ACCURACY

The homomorphic encryption system proved to be very reliable and precise in producing the accuracy test. It tested the results of encrypted computations for a dataset of twenty readings, and compared them with the results of plaintext computations, in terms of average systolic blood pressure, average heart rate, and correlation coefficient. When the encrypted results were produced, the average systolic blood pressure was 130, the average heart rate was 84.2, and the correlation coefficient was 0.926703. Resulting from the results on the plaintext calculations being identical, the accuracy was extremely high. The test demonstrates that the system can do its computations accurately on encrypted data. The absence of any errors both absolute and relative for all metrics indicates that the encryption scheme -encoding, encryption, evaluation, decryption and decoding steps- is being implemented correctly. Secure operations for summations, multiplications and rotations which constitute operations required for computing the correlation coefficient were performed while not affecting the computational noise. The accuracy seen here indicates that the workload could be handled with sufficient precision, and the encryption parameters, including a polynomial modulus degree of 16,384 and a plaintext modulus of 65,537, were appropriate for further operation. Further datasets are required to fully characterize the ability of the system to scale and be precise under heavier computational loads. The first test was basic statistical computations. A more complete evaluation of the accuracy of the system could be had with more diverse or more complex operations.

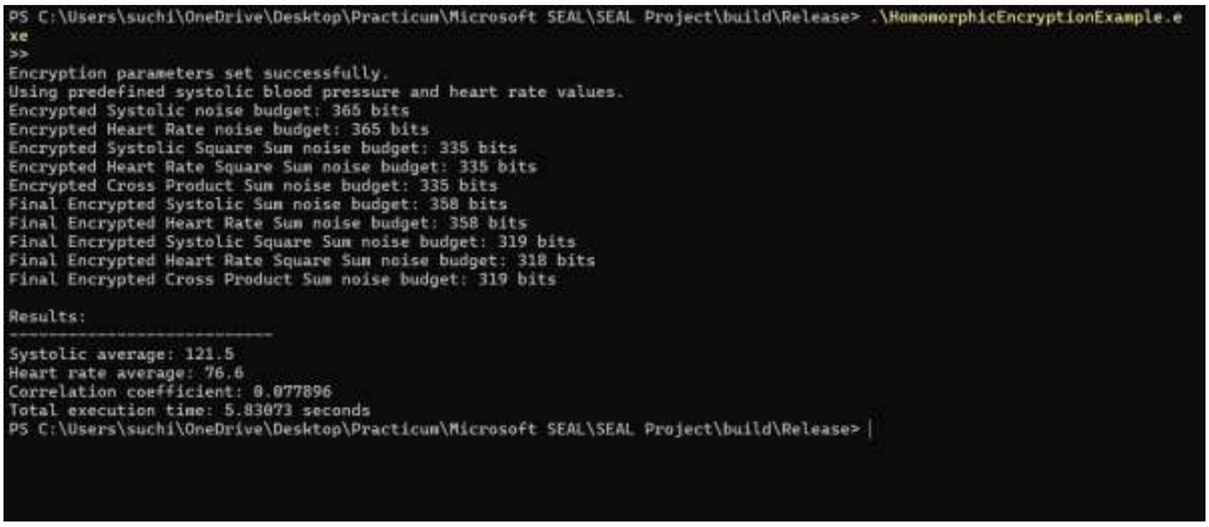


```
PS C:\Users\suchi\OneDrive\Desktop\Practicum\Microsoft SEAL\SEAL Project\build\Release> .\HomomorphicEncryptionExample.e
xe
>>

Encrypted Results:
----------------------------
Average Systolic BP: 130
Average Heart Rate: 84.2
Correlation Coefficient: 0.926703

Accuracy Test Results:
-------------------------
Systolic Average Error: 0
Heart Rate Average Error: 0
Correlation Coefficient Error: 0
Systolic Average Relative Error: 0%
Heart Rate Average Relative Error: 0%
Correlation Coefficient Relative Error: 0%
PS C:\Users\suchi\OneDrive\Desktop\Practicum\Microsoft SEAL\SEAL Project\build\Release> |
```

(Fig. 3) Accuracy test

NOISE BUDGET

To evaluate the behavior of encrypted computations and system capability to maintain sufficient noise levels across computations, a noise budget evaluation test was carried out. Initial encryption parameters were successfully initiated using a polynomial modulus degree of 16,384 and a plaintext modulus of 65,537, giving a large initial noise budget in which to conduct precise computations. Predefined systolic blood pressure and heart rate datasets were processed by the system and the noise budget was tracked throughout the computations at various steps, that is, encryption, squaring, multiplication, rotation and summation. The values of the initial capacities show the encryption scheme's initial ability to handle computations. A slight difference in noise budget when systolic (365 bits) and heart rate (365 bits) are encoded or encrypted is well within acceptable limits and this may be due to a slight difference in what is encoded. The complexity of these operations is the expected squaring and multiplication, so the noise budget was reduced about 29–30 bits. Noise budget values at the final stage indicate only an insignificant reduction compared to earlier stages, indicating good noise management during additional operations such as rotations and summations. This guarantees that results are decryptable with good accuracy. The noise budget stayed within safe limits at all times and proved that the encryption parameters were robust and that techniques like relinearization and rotation were effective. This reduction in noise budget across operations is minor due to the encryption scheme's complex computations with only marginal degradation in noise capacity. These results support the system's capacity for performing secure critical use of sensitive data and providing accurate and decryptable outcomes. The test was conducted on a small dataset relative to the system, thereby possibly limiting understanding of its behavior under larger workloads. The operations tested were mostly statistical. For some more complex computations, the impact on the noise budget may be different.



```
PS C:\Users\suchi\OneDrive\Desktop\Practicum\Microsoft SEAL\SEAL Project\build\Release> .\HomomorphicEncryptionExample.e
xe
>>
Encryption parameters set successfully.
Using predefined systolic blood pressure and heart rate values.
Encrypted Systolic noise budget: 365 bits
Encrypted Heart Rate noise budget: 365 bits
Encrypted Systolic Square Sum noise budget: 335 bits
Encrypted Heart Rate Square Sum noise budget: 335 bits
Encrypted Cross Product Sum noise budget: 335 bits
Final Encrypted Systolic Sum noise budget: 358 bits
Final Encrypted Heart Rate Sum noise budget: 358 bits
Final Encrypted Systolic Square Sum noise budget: 319 bits
Final Encrypted Heart Rate Square Sum noise budget: 318 bits
Final Encrypted Cross Product Sum noise budget: 319 bits

Results:
----------------------------
Systolic average: 121.5
Heart rate average: 76.6
Correlation coefficient: 0.077896
Total execution time: 5.83073 seconds
PS C:\Users\suchi\OneDrive\Desktop\Practicum\Microsoft SEAL\SEAL Project\build\Release> |
```

(Fig. 4) noise Budget

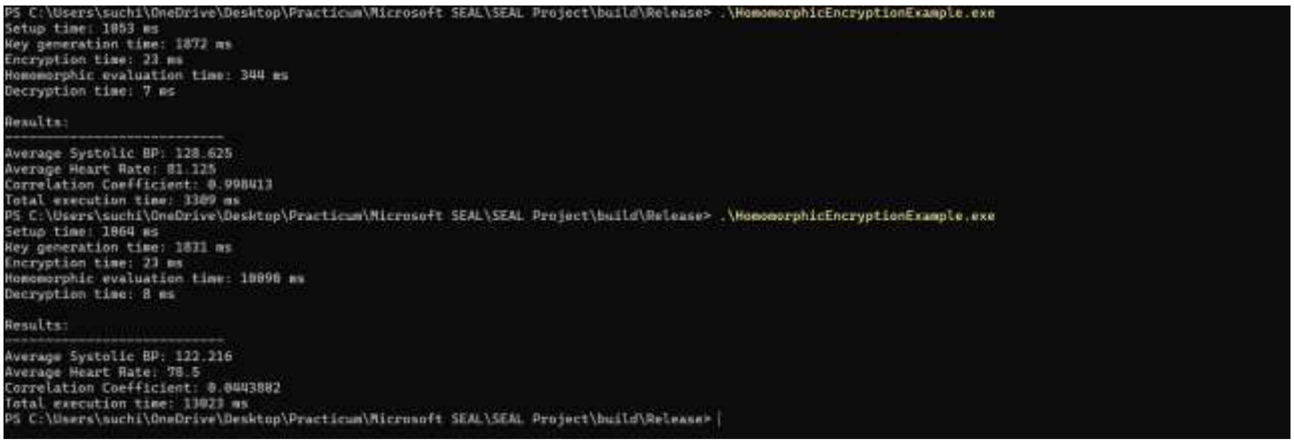
SCALABILITY

The performance test was conducted with datasets of 8, 30, 60, 90, and 120 elements to evaluate the scalability of the homomorphic encryption system. The key metrics measured included setup time, key generation time, encryption time, homomorphic evaluation time, decryption time, and total execution time.

Dataset Size	Setup Time (ms)	Key Generation Time (ms)	Encryption Time (ms)	Homomorphic Evaluation Time	Decryption Time	Total Execution Time	Correlation Coefficient
8	1053	1872	23	344	7	3309	0.998413
30	1040	1857	21	1907	7	4838	0.277931
60	1044	1790	21	4490	8	7363	0.0102929
90	1021	1775	22	7120	7	9951	0.0596037
120	1064	1831	23	10090	8	13023	0.0443802

(1. Performance Scalability)

The lapsed times stayed relatively constant in setup and key generation, ranging from 1021ms to 1064ms and 1800ms respectively. One-time costs did not affect scalability. The encryption and decryption times were the same for all dataset sizes (roughly 21–22ms for encryption, and 7–8ms for decryption). This shows that these operations scale linearly with small element overheads. Homomorphic evaluation times experienced the greatest performance impact and grew sharply with data set size reflecting the disproportionate increase in evaluation complexity. This is attributed to operations such as rotations, summations, and multiplications, which are computationally intense. Total execution times increased linearly with data set size, with homomorphic evaluation times dominating at large scales. System performance for setup, key generation, encryption and decryption was measured and for the three dataset size showed efficient scaling. With increase in dataset size, homomorphic evaluation times increased, a symptom of scalability bottleneck. Rotations and summations appear to be disproportionate among operations contributing to this overhead. The accuracy of correlation coefficient was subject to some degradation with larger datasets, possibly due to noise accumulation or increased variability in the data. The figure below shows the difference in performance when datasets of 8 elements and 120 elements are used.



```
PS C:\Users\suchi\OneDrive\Desktop\Practicum\Microsoft SEAL\SEAL Project\build\Release> .\HomomorphicEncryptionExample.exe
Setup time: 1053 ms
Key generation time: 1872 ms
Encryption time: 23 ms
Homomorphic evaluation time: 344 ms
Decryption time: 7 ms

Results:
------------------------------
Average Systolic BP: 128.625
Average Heart Rate: 81.125
Correlation Coefficient: 0.998413
Total execution time: 3309 ms
PS C:\Users\suchi\OneDrive\Desktop\Practicum\Microsoft SEAL\SEAL Project\build\Release> .\HomomorphicEncryptionExample.exe
Setup time: 1064 ms
Key generation time: 1831 ms
Encryption time: 23 ms
Homomorphic evaluation time: 10098 ms
Decryption time: 8 ms

Results:
------------------------------
Average Systolic BP: 122.216
Average Heart Rate: 78.5
Correlation Coefficient: 0.0443882
Total execution time: 13023 ms
PS C:\Users\suchi\OneDrive\Desktop\Practicum\Microsoft SEAL\SEAL Project\build\Release> |
```

(Fig. 5) Performance scalability Tests

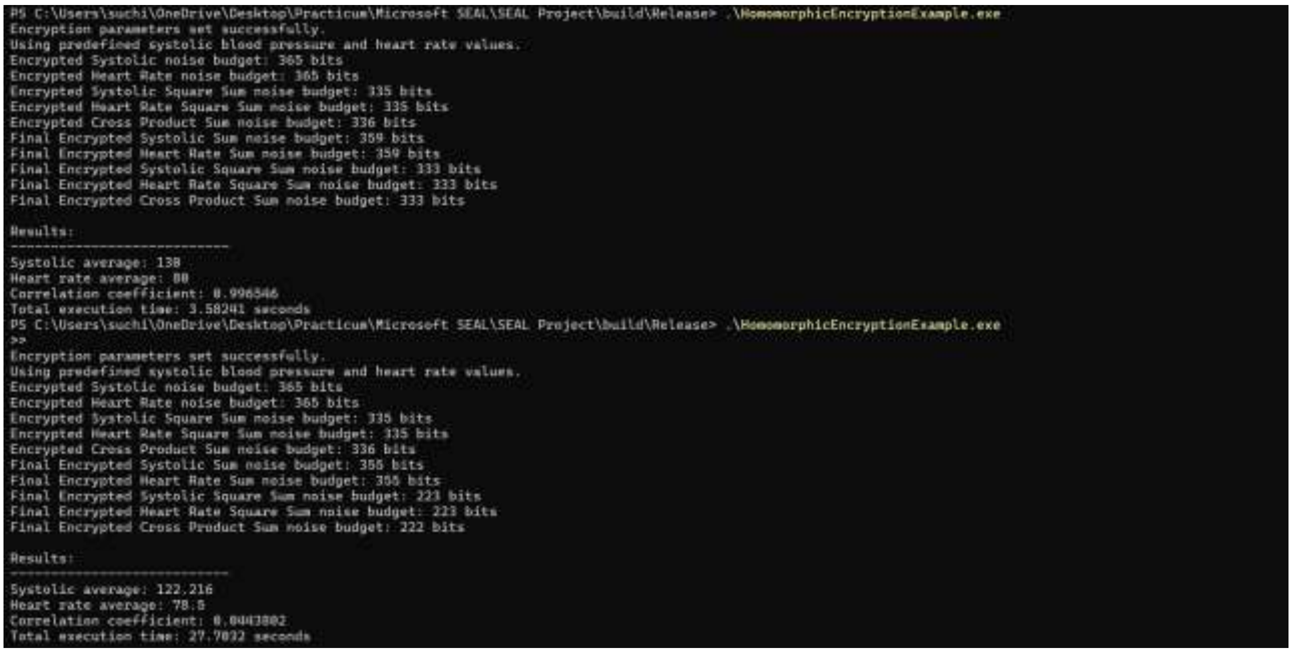
The noise budget tests were conducted with datasets of 8, 30, 60, 90, and 120 elements to analyze how noise levels are affected by dataset size and the complexity of homomorphic operations. Key metrics such as the initial and final noise budgets at various stages (e.g., encryption, squaring, multiplication, summation) were tracked to assess the system's scalability and noise efficiency.

Dataset Size	Initial Noise Budget (Bits)	After Squaring and Multiplication (Bits)	Final Noise Budget (Bits)	Final Square Sum Noise Budget (Bits)	Execution Time (s)
8	365	335	359	333	3.58
30	364	335	356	309	7.78

60	365	335	356	279	14.01
90	365	335	355	249	20.65
120	365	335	355	222	27.70

(2. Noise Budget Scalability)

The systolic and heart rate noise budgets were robust across all dataset sizes with an initial noise budget of 364–365 bits. Operations of squaring and multiplication consistently left the noise budget at about 29–30 bits, independent of size of the dataset. This shows predictable noise depletion during these computationally intense steps. Noise budget for systolic and heart rate sums stayed quite flat at ~355 bits for all dataset sizes leading to efficient noise management during rotations and summations. As dataset size increased, the decrease in constraint size on the noise budget for square sums was greater. It was observed that deeper computations with larger data sets have more noise, especially for square sum and cross product operations. Wider datasets exhibited more noise depletion in operations such as squaring and summation, most notably for square sums and cross products. However, it may restrict the computational depth available for more complex or extended workflows. Furthermore, the execution time increased dramatically as dataset size increased, and operations such as rotation and summation required optimization. The difference in noise budget after computing two datasets with 8 and 120 elements respectively are shown below.



```
PS C:\Users\suchi\OneDrive\Desktop\Practicum\Microsoft SEAL\SEAL Project\build\Release> .\HomomorphicEncryptionExample.exe
Encryption parameters set successfully.
Using predefined systolic blood pressure and heart rate values.
Encrypted Systolic noise budget: 365 bits
Encrypted Heart Rate noise budget: 365 bits
Encrypted Systolic Square Sum noise budget: 335 bits
Encrypted Heart Rate Square Sum noise budget: 335 bits
Encrypted Cross Product Sum noise budget: 336 bits
Final Encrypted Systolic Sum noise budget: 359 bits
Final Encrypted Heart Rate Sum noise budget: 359 bits
Final Encrypted Systolic Square Sum noise budget: 333 bits
Final Encrypted Heart Rate Square Sum noise budget: 333 bits
Final Encrypted Cross Product Sum noise budget: 333 bits

Results:
----------------------------
Systolic average: 138
Heart rate average: 80
Correlation coefficient: 0.996546
Total execution time: 3.58241 seconds
PS C:\Users\suchi\OneDrive\Desktop\Practicum\Microsoft SEAL\SEAL Project\build\Release> .\HomomorphicEncryptionExample.exe
>>
Encryption parameters set successfully.
Using predefined systolic blood pressure and heart rate values.
Encrypted Systolic noise budget: 365 bits
Encrypted Heart Rate noise budget: 365 bits
Encrypted Systolic Square Sum noise budget: 335 bits
Encrypted Heart Rate Square Sum noise budget: 335 bits
Encrypted Cross Product Sum noise budget: 336 bits
Final Encrypted Systolic Sum noise budget: 355 bits
Final Encrypted Heart Rate Sum noise budget: 355 bits
Final Encrypted Systolic Square Sum noise budget: 223 bits
Final Encrypted Heart Rate Square Sum noise budget: 223 bits
Final Encrypted Cross Product Sum noise budget: 222 bits

Results:
----------------------------
Systolic average: 122.216
Heart rate average: 78.5
Correlation coefficient: 0.0443802
Total execution time: 27.7032 seconds
```

(Fig. 6)Noise budget scalability Tests

7 Conclusion and Future Work

The primary research question addressed in this study was: How can Fully Homomorphic Encryption be used in the healthcare industry. The study set out to design and implement a practical FHE framework for healthcare applications, which would be secure in protecting

Protected Health Information (PHI), and to support encrypted data analytics. This project successfully implemented a comprehensive FHE framework using the Microsoft SEAL library that satisfies the FHE architecture presented. Major contributions were to assess the BFV and CKKS encryption schemes, propose workflows for secure computation on encrypted healthcare data, and characterize performance, accuracy, noise budget, and scalability. The system proved robust at preserving data security and accuracy with very high fidelity in encrypted computations compared to plaintext benchmarks. The project also analysed performance evaluation to show consistent efficiency of encryption and decryption times, as well as homomorphic evaluation times that scale with dataset size. The system was subjected to noise budget testing, and it was found that noise levels were managed effectively across computations, so that complex operations did not produce results that could not be decrypted. The results of this research have key implications for the use of FHE for healthcare. This work shows that FHE is feasible as a means to support secure, end-to-end encrypted computations such as patient data aggregation, trend analysis, and anomaly detection while meeting strict privacy standards such as HIPAA. Nevertheless, scalability limitations were found such as homomorphic evaluation size limitations due to bottlenecks (and hence scalability limitation in homomorphic evaluation time for larger datasets). Noise depletion in deeper computations were also found to affect the system's capabilities in performing the workflows that are more complex and beyond the current one.

These challenges could be addressed in the future works by including principles from Multi Party Computation (MPC). In this approach the data can be stored in Virtual Shares to multiply the data in separate pools (buckets) and even parts (shares) and ensure that the entire dataset is never consolidated in one part of it. In this way it would cut down security risks by reducing keys in circulation, especially in cases where decryption is needed before application of the operation such as division. The absence of complete data at a single point enhances privacy and decreases the susceptibility. Since MPC can introduce substantial noise that would make final decryption results inaccurate, however, this project did not implement MPC. Addressing these accuracy challenges would be required to integrate virtual shares and to keep computations accurate and reliable. Hybrid models fusing FHE and MPC may greatly contribute to tackle the apparent conflict and to propose a secure and scalable solution for the analysis of healthcare data. Research into optimizing noise management techniques (e.g., advanced re-randomization and noise reduction strategies) to achieve acceptable accuracy while implementing virtual shares could be done in future. Additionally, the adaptive parameter configurations can be designed specific to certain healthcare workflow. It can also accommodate hardware acceleration such as GPUs to make system more efficient. This would make privacy preserving computation on a larger scale possible and thus open up the way for real world deployments of FHE in secure healthcare systems.

References

Alder, S. (2024). December 2023 Healthcare Data Breach Report. [online] HIPAA Journal. Available at: https://www.hipaajournal.com/december-2023-healthcare-data-breach-report/.

Armknecht, F., Boyd, C., Carr, C., Gjøsteen, K., Jäschke, A., Reuter, C.A. and Strand, M. (2015). A Guide to Fully Homomorphic Encryption. [online] ePrint IACR. Available at: https://eprint.iacr.org/2015/1192.

Marcolla, C., Sucasas, V., Manzano, M., Bassoli, R., Fitzek, F.H.P. and Aaraj, N. (2022). Survey on Fully Homomorphic Encryption, Theory, and Applications. Proceedings of the IEEE, 110(10), pp.1572–1609. doi:https://doi.org/10.1109/jproc.2022.3205665.

Francisco-Jose Valera-Rodriguez, Manzanares-Lopez, P. and Cano, M.-D. (2024). Empirical Study of Fully Homomorphic Encryption Using Microsoft SEAL. Applied Sciences, [online] 14(10), pp.4047–4047. doi:https://doi.org/10.3390/app14104047.

Zhu, H., Suzuki, T., Huang, H. and Yamana, H. (n.d.). Performance Comparison of Homomorphic Encrypted Convolutional Neural Network Inference between Microsoft SEAL and OpenFHE. [online] Available at: https://proceedings-of-deim.github.io/DEIM2023/5b-9-2.pdf.

Zhu, H., Suzuki, T. and Hayato Yamana (2023). Performance Comparison of Homomorphic Encrypted Convolutional Neural Network Inference Among HElib, Microsoft SEAL and OpenFHE. 2021 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE), [online] 13092, pp.1–7. doi:https://doi.org/10.1109/csde59766.2023.10487709.

Carey, A. (2020). On the Explanation and Implementation of Three Open-Source Fully Homomorphic Encryption Libraries. [online] ScholarWorks@UARK. Available at: https://scholarworks.uark.edu/csceuht/77/ [Accessed 26 Nov. 2024].

Fawaz, S.M., Belal, N., ElRefaey, A. and Fakhr, M.W. (2021). A Comparative Study of Homomorphic Encryption Schemes Using Microsoft SEAL. Journal of Physics: Conference Series, 2128(1), p.012021. doi:https://doi.org/10.1088/1742-6596/2128/1/012021.

Peng, Z. (2019). Danger of using fully homomorphic encryption: A look at Microsoft SEAL. [online] arXiv.org. doi:https://doi.org/10.48550/arXiv.1906.07127.

Fotios Roumpies and Athanasios Kakarountas (2023). A Review of Homomorphic Encryption and its Contribution to the Sector of Health Services. doi:https://doi.org/10.1145/3635059.3635096.

Bansal, V. (2021). *Survey on Homomorphic Encryption*. [online] IEEE Xplore. doi:https://doi.org/10.1109/ISCON52037.2021.9702486.

Kost, E. (2022). Biggest Cyber Threats in Healthcare (Updated for 2022) | UpGuard. [online] www.upguard.com. Available at: https://www.upguard.com/blog/biggest-cyber-threats-in-healthcare.