National College of Ireland

A comparative study of AES encryption modes and hashing for blockchain applications

MSc Research Project
Cloud Computing

Debashish Sarangi
Student ID: 22162844

School of Computing
National College of Ireland

Supervisor: Dr Giovani Estrada

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Debashish Sarangi
Student ID:	22162844
Programme:	Cloud Computing
Year:	2024
Module:	MSc Research Project
Supervisor:	Dr Giovani Estrada
Submission Due Date:	25/04/2024
Project Title:	A comparative study of AES encryption modes and hashing for blockchain applications
Word Count:	6962
Page Count:	23

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:	
Date:	27th May 2024

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):	

A comparative study of AES encryption modes and hashing for blockchain applications

Debashish Sarangi
22162844

Abstract

This paper gives a complete examination of multiple modes of Advanced Encryption Standard (AES) algorithms, concentrating on their performance features and security consequences. A comparative analysis was conducted on all modalities to determine the fastest and most efficient option in terms of throughput. The Electronic Code Book (ECB) mode has been determined to be the fastest one thorough experimentation and analysis. However, it is important to note that this mode may have security weaknesses due to its block-independent nature. Conversely, Cipher Block Chaining (CBC) mode offers greater security through chaining techniques and competitive throughput. the findings underline the significance of balancing speed and security factors when selecting an AES encryption option. A comparative analysis was conducted on all modalities to determine the fastest and most efficient option in terms of throughput. Integration of Simple Mail Transfer Protocol (SMTP) with the Gmail API provides secure key distribution, boosting secrecy and integrity in email communication. This research gives significant insights for enhancing AES encryption and strengthening data security against evolving cyber threats.

1 Introduction

1.1 Comparison Between On and Off-chain Blockchain Systems

1.1.1 Scalability Concerns

On-chain blockchains like the Bitcoin and Ethereum allow for scalability only to some extent because of their fundamental limitations regarding the majority rule and block size. With health care data accumulation that is growing at astronomical rates, a challenge of the volume of transactions on the main blockchain is getting more difficult to manage. The scalability problem may sooner or later causes the transaction process to be delayed, makes transaction costs high, and as a result the overall system performance will decrease. Through employing off-chain techniques, where transactions are performed without the involvement of the core blockchain, scalability limits can be avoided, ensuring that the flow of PHR is carried out promptly and safely. (Kim et al.; 2018).

1.1.2 Privacy Requirements

The primary aim of protecting patient privacy and personal information in healthcare sector is already regulated by such regulations as GDPR. Whilst the majority of on chain

blockchains provide transparency and immutability, they also place exposed all transactions to the public and hence compromising the patient privacy. Offchain solutions offer a possibility to execute anonymous operations. This is done in an environment without the eyes of the public that is the blockchain (Miyachi and Mackey; 2021). Encrypting and securely handling PHR data off-chain provides privacy protection for patients and checks the compliance of the blockchain platform with GDPR regulations, and in the end, such a platform will be trusted and supported by various stakeholders.

1.1.3 Regulatory Compliance

The main element to be remembered in the healthcare as medical records involves the strong regulatory standards such as GDPR and proper handling and sharing of personal data are a must. These guidelines should be strictly adhered to and they emphasize the need for robust security protections to secure the important data. Off-chain solutions of blockchain are more effective in defining access control methods that are consent-based, and they also help maintain GDPR compliance.(Mannan et al.; 2019) Through implementation of off-chain technologies, patients gain the liberty to either let or disapprove access to their PHR data, hence giving individuals the power to take the lead to control their own data and at the same time stand for norms in regulations.

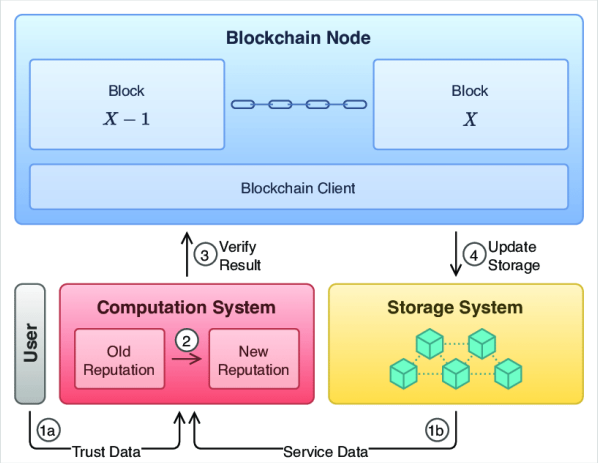

Figure 1: On-chain and Off-chain computation and storage model. From: (Battah et al.; 2021)

1.2 Cryptography

In modern network communication, cryptography serves vital function being a factor of protecting data secrecy, consistency, and integrity. At the base of cryptographic protocols IKE stands (Nagalakshmi et al.; 2011), the TCP/IP implementation that manages the creation of essential keys used to protect data exchanges between clients and servers. IKR underpins the security layers' strengthening which is a primary area of focus, in private networks which deliver move of sensitive data . Therefore, the role of IKR in modern communication set up cannot be overlooked and this further emphasizes the value of IKR in the modern day communication set up. The primary theme of this thesis is to do an

in -depth analysis of the practices and procedures of safeguarding the data through using cryptographic techniques as the tool (Maqsood et al.; 2019).

IKE placed on top of itself in the stack of operating system, due to the fact being the hybrid protocol, is in charge of keys management and generation within the private networks. It should provide the basis for such critical functions as the key exchange during encryption and decryption through a non-secure medium like the Internet, whereby special encryption technologies may be improved to make them more appropriate for use in difficult and risk-prone situations. In mid-to-late 1990s, IKE which is the term for the most widely used protocol for making secure key exchanges, was developed as a key exchange mechanism, it aimed to mitigates untrusted system vulnerabilities, laying a solid foundation for data transmission security.

Cryptography foundations include the principle of key exchange that allows encryption and decryption procedures to take place, making sure that the resultant keys remain shared only between the servers and clients. Also, cryptography employs such things as forward secrecy, perfect security, and deniable authentication among others that ensures the data is safe from potential threats or flaws (Jaswal et al.; 2017).

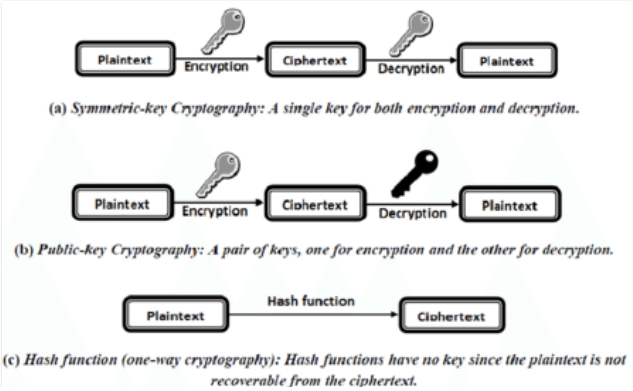

Figure 2: Three types of cryptography: Symmetric-key, public-key, and hash function. (Rahman et al.; 2016)

However, among those are cryptographic algorithms which are further subdivided into secret key cryptography, public key cryptography and hash functions. These very algorithms are right at the centre of all encryption and decryption processes as they ensure the safety and integrity of sent data.

The Data Encryption Standard (DES) and the Advanced Encryption Standard (AES) algorithms, invented by IBM in the 1970s and 2001 respectively, epitomize the growth and importance of cryptographic techniques. DES, building block cipher, works on 64-bit blocks and needs a 56-bit key to perform its encryption and decryption process. Spite its historical significance, DES has been increasingly replaced by AES as a result of the small key size of DES being vulnerable to brute force attacks (Smid; 2021).

In contrast, AES has emerged as a game changer in the cryptographic protocols family by bringing more security and performance to the table than DES. AES operates on key lengths that are changeable, either as 128, 192 or 256 bits, providing firm protection against advanced crypt analysis. AES has been widely employed for different sectors as well as applications. It is, in fact, the most preferred standard used for the security of data transmission in the modern communication networks.

AES implementations give a profusion of cipher possibilities, posing the question of which one is ideal for use. Selecting the appropriate mode of operation is crucial for strengthening the security of the software, as certain modes conceal vulnerabilities exploitable by malevolent actors (Almuhammadi and Al-Hejri; 2017).

In the area of AES encryption mechanisms, the Electronic Code Book (ECB) stands out for its straightforwardness, encrypting each plaintext block independently. However, its vulnerability comes in the same encryption of similar blocks, potentially revealing patterns. Cipher Block Chaining (CBC) increases security by XORing each plaintext block with the previous ciphertext, preventing pattern repetition albeit at the cost of higher processing time. Cipher Feedback (CFB) mode offers an alternate solution, XORing a portion of the encryption output with plaintext, assuring security without the need for padding and aiding error prevention. Output Feedback (OFB) mode updates a shift register with encryption output, decreasing error propagation and increasing security by direct feedback from the encryption module. Finally, Counter Mode (CTR) leverages a counter and nonce to build a key stream, enabling rapid encryption ideal for parallel processing but lacks some data integrity characteristics.

1.3 Research Question

There are many encryption algorithms; however, AES is the current encryption standard. Other algorithms exist beyond AES, such as Blowfish and CAST-128. AES is indeed a family of algorithms with variants that will be reviewed in the next sections. For blockchain purposes, little is known about their suitability in terms of throughput and CPU encryption time. The research question is:

1. How do different AES encryption modes impact the security, efficiency, and performance of blockchain applications?

In this study paper, a comparison of a number of AES variants (AES-ECB, AES-CBC, AES-CFB, AES-OFB, AES-CTR, AES-GCM) will be undertaken, alongside an inquiry of why integrating a hashing algorithm with encryption techniques is necessary to enhance the security of transactions.

1.4 Research objectives

A number of steps have to be taken to answer the abovementioned questions. The research objectives are:

1. Literature review of blockchain and encryption techniques
2. Implementation of blockchain
3. Implementation and comparison of AES variants
4. Implementation of hashing techniques to strength AES
5. Implementation of use case of health records

1.5 Outline

The report proceeds with Section 2 discussing blockchain and encryption theory, while Section 3 discusses the recommended solution. Section 4 explains the approaches adopted, followed by Section 5's implementation phase. Section 6 assesses the approach through case studies for research issues, and Section 7 concludes the findings.

2 Literature Review

The literature review addresses two main areas: the incorporation of blockchain technology in healthcare systems and encryption strategies. It emphasizes blockchain's promise to transform data storage, sharing, and security in healthcare through decentralization and immutable recordkeeping. Additionally, it addresses encryption systems such as symmetric and asymmetric encryption, highlighting their applications and limitations. The paper also covers password management utilising hashing algorithms and the difficulty of combining individual privacy with health data consumption for research purposes, providing solutions to avoid re-identification hazards. This section is aligned to Research Objective 1.

2.1 Blockchain Technology Applications in Conventional Healthcare Systems

The incorporation of blockchain technology into traditional healthcare systems signifies a substantial revolution, offering a fundamental change in the way health-related data is stored, shared, and safeguarded. The decentralised nature of blockchain technology upsets the conventional centralised method to managing healthcare information by eliminating the necessity for middlemen in data verification (Stafford and Treiblmaier; 2020). This development ultimately enhances confidence among parties involved in the healthcare system. The main features of this integration are distributed control, unchangeable recordkeeping, authentication of data source, improved reliability and availability, and strong data security. The combination of these characteristics together guarantees the preservation, availability, and protection of healthcare information, promoting cooperation and facilitating enhanced healthcare provision.

2.2 Encryption Overview

In a suggested thesis, (Kelsey et al.; 1996) offers a hybridized technique that amalgamates cloud system characteristics, seeking to boost authorized users' accessibility to virtual resources from varied geographical locations. The adaptability and varied characteristics of this technique enhance the capacity of networking and decrease latency. Kelsey and colleagues additionally propose a security-oriented methodology meant to harness cloud infrastructure capabilities, thereafter deploying them successfully. In a second research by (Koehler; 2001), the need of preserving data in the cloud and ensuring its anonymity within the idea of cloud computing is underlined. This emphasis is vital to enable main users to access computer services without limitations. Koehler's paper painstakingly investigates potential data security and privacy risks across the data life cycle, providing several tactics and protocols to limit the risk of data breaches via sensitive communication channels. However, the system model adopted in this technique has intrinsic limitations,

particularly the inability to accommodate a considerable quantity of keys necessary for data management across communication channels. Nonetheless, the discovery of supplemental approaches subsequent to the creation of AES and RSA algorithms has proven important in overcoming these problems.

In a study by (Mare et al.; 2011), the RSA technique was recommended for creating public keys to enhance data security in storage files. Legitimate users were provided access to these public keys, contributing to the protection of sensitive data. Meanwhile, binary files within the configuration file required secure treatment to enable proper functioning. To disclose potential attacks on web servers, the AES algorithm was implemented, leveraging unique combinations to identify ideal solutions and enhance data protection against such threats.

In another paper by (McEliece; 2012), the use of symmetric and asymmetric algorithms was proposed to convey communication issues among IoT devices. Leveraging improvements in the RSA method, the encryption process was streamlined to save processing time. However, worries over overall system integrity continued. To solve this, a hybrid solution presented by (Paar and Pelzl; 2009) integrated the strengths of the RSA and AES algorithms to secure communication over insecure media. This hybrid technique displayed better accuracy and efficiency compared to individual methods, giving enhanced protection for data saved in important cloud storage systems.

2.2.1 Asymmetric Key Encryption

In a paper by (Singh; 2013), an asymmetric key encryption technique based on linear geometry principles was proposed. This approach attempted to facilitate covert communication via unsecured internet platforms by utilizing both substitution and transposition techniques. Unlike earlier systems focusing on text, this model prioritized image protection during transmission. A random matrix generated by a random number generator was applied to cipher data from files. Also Asymmetric key cryptography, implemented through the public key technique, uses two keys: a public key and a private key. The private key, known only to the user, and the public key, visible to all, shared a mathematical relationship but could not be inferred from each other, boosting security. Depending on the programme, these keys functioned differently. While data encryption utilized the public key, decryption required the private key. Additionally, private keys were occasionally used to construct digital signatures, confirmed using public keys.

A typical asymmetric key cryptography method's workflow is shown in figure 8.

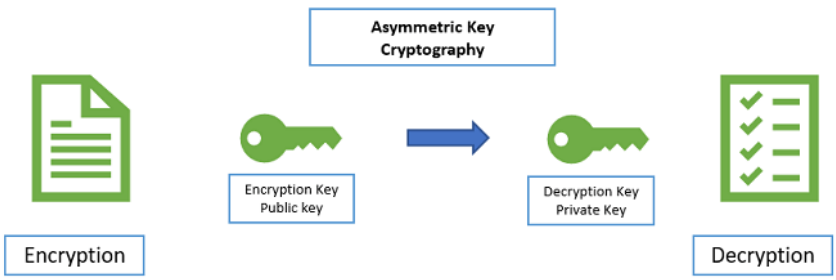

Figure 3: Asymmetric Key Encryption
[1]

2.2.2 Symmetric Key Encryption

In a study by (Rivest et al.; 1978), an AES-based technique was developed to protect information kept in the cloud. The method uses variations for moving data between endpoints, utilizing keys such as AddRoundKey and ShiftRowKey to complete the encryption process. These keys were iterated based on the bit size, guaranteeing that the receiver could decrypt the ciphertext effectively. Despite requiring minimum storage space for key management, the mechanism sustained high encryption requirements, effectively ensuring the confidentiality and integrity of the system during data encryption and decryption procedures. Another work, (Rueppel; 1986) offered an easy cryptographic approach to handle cloud storage difficulties. Subsequently, a hybridized strategy was presented, integrating symmetric and asymmetric techniques for increased cloud data security. This model integrated AES and RSA algorithms, with the RSA implementation proving successful compared to other cryptography techniques (Paradesi Priyanka et al.; 2022).

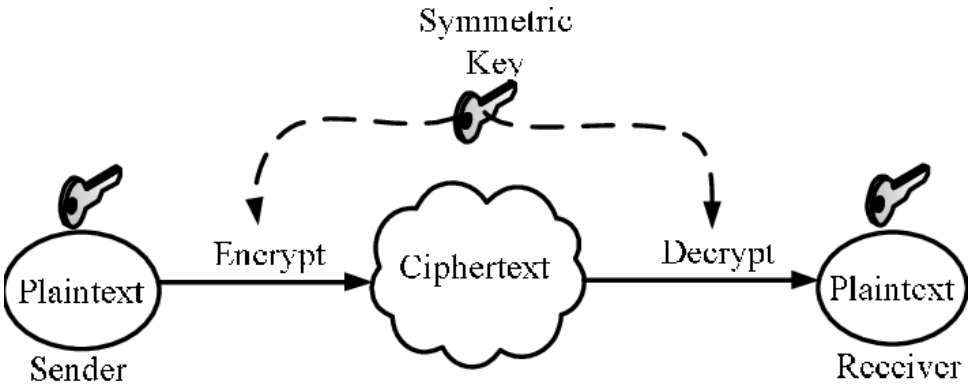

Figure 4: Symmetric Key Encryption (Tahir et al.; 2008)

As noted previously, symmetric encryption techniques typically utilize a single key for both encryption and decryption processes. Algorithms such as DES, AES, and Blowfish are deployed for this purpose, each handling distinct data types based on their block sizes and key sizes. The plaintext, representing the original information intended for transmission, undergoes encryption operations to turn into ciphertext, using a secret key. This ciphertext then acts as input for the encryption method, thereby becoming the new plaintext. This procedure relies on a private key encryption mechanism to generate a single key for both encryption and decryption duties. While straightforward to build, symmetric encryption needs key exchange between communication partners, needing a high level of trust and sharing between parties to provide secure communication channels and prevent unwanted decryption by third parties.

In contrast, asymmetric encryption involves two unique keys—public and private—shared between parties. The public key encrypts data, while the private key solely decrypts messages. This strategy strengthens security by distributing symmetric keys between authorized users and cloud providers. AES and RSA, two thoroughly studied algorithms, have demonstrated reliability due to AES's processing speed and RSA's resilient key length, as observed in comparative studies. Nowadays, AES is the encryption standard of choice and has been ported to hardware across vendors[2].

[2]`https://en.wikipedia.org/wiki/AES_instruction_set`

2.2.3 Comparison Between Different Encryption Algorithms

The comparative study undertaken by (Koukou et al.; 2016) gives a complete analysis of four prominent symmetric key cryptography algorithms: AES, DES, CAST-128, and Blowfish. The research intends to analyse the behavior and performance of various algorithms under varying data loads, concentrating mainly on aspects such as speed, block size, and key size.

AES (Advanced Encryption Standard), proposed by NIST in 2001 to replace DES, stands out for its versatility, enabling varied data and key lengths. Employing 10, 12, or 14 rounds depending on the key length, AES operates on a 128-bit data length structured into a 4x4 matrix, enduring an Add Round Key stage during both encryption and decryption.

Blowfish, created by Bruce Schneider in 1993 (Muin et al.; 2018), offers flexibility with varied key lengths ranging from 32 to 448 bits. Utilizing a 16-round Feistel cipher structure and huge key-dependent S-boxes, Blowfish works well for applications where key changes are few.

CAST-128, invented in 1996 by Carlisle Adams and Stafford Tavares (Boey et al.; 2010), operates as a Feistel network with 12 or 16 rounds, 64-bit blocks, and key sizes ranging from 40 to 128 bits. Employing huge 8x32-bit S-boxes, key-dependent rotations, and XOR operations, CAST-128 offers robust security while being offered royalty-free worldwide.

DES (Data Encryption Standard), The Feistel Structure-based symmetric key algorithm known as DES has a 56-bit key length and a 64-bit plaintext with 16 rounds. DES is thought to be less secure than other algorithms evaluated in the study, despite its widespread use.

Through careful examination, the research reveals that AES delivers the highest security and performance among the analysed algorithms. AES exhibits powerful avalanche effect and integrity checks, notably in ECB mode. Conversely, DES displays significant avalanche effect, while CAST-128 excels in integrity testing, particularly in CBC mode. The findings underline the necessity of selecting the proper encryption technique based on unique application requirements and security considerations.

In recent years, considerable strides have been made in the area of cryptographic research, marked especially by advancements in quantum-resistant cryptography, post-quantum cryptography, and the study of unique encryption approaches customised to specific applications (Garcia et al.; 2023a). These discoveries provide a proactive response to the rising security dangers posed by new technology, particularly quantum computing. Quantum-resistant cryptography strives to reinforce cryptographic systems against the future threat of quantum computers, which contain exponentially more processing power and the ability to render current encryption approaches obsolete.

Conversely, post-quantum cryptography focuses on building cryptographic algorithms immune to quantum attacks, so ensuring the long-term security of sensitive information in a quantum computer era. Concurrently, the exploration of novel encryption algorithms targeted to unique use cases emphasises a holistic approach to solving diverse security concerns . By consistently pushing the boundaries of cryptographic innovation, researchers aspire not only to bolster the resilience of existing systems but also to build the framework for future-proof security solutions capable of withstanding the shifting panorama of cyber threats.

2.2.4 AES Encryption And Its Different Modes

As noted elsewhere (El-Semary and Azim; 2015), AES block ciphers works through five standard modes of operation which are: Electronic Code Book (ECB), Cipher Block Chaining (CBC), Cipher Feedback (CFB), Output Feedback (OFB), and Counter (CTR) modes. To make this report self-contained, a quick review of the main differences are described below.

- Electronic Code Book (ECB): According to (Huang; 2013), ECB mode encrypts blocks independently, resulting in identical ciphertext blocks for identical plaintext blocks. Although efficient, it may expose data patterns.
- Cipher Block Chaining (CBC): CBC mode XORs each plaintext block with the preceding ciphertext block before encryption, reducing pattern vulnerability (Hook; 2005). It requires an Initialization Vector (IV) for distinct ciphertexts.
- Cipher Feedback (CFB): In CFB mode, part of the output feeds back into plaintext, enabling an XOR operation . It doesn't require padding and prevents channel noise propagation.
- Output Feedback (OFB): OFB mode updates the shift-register using specific parts of the encryption module's output, handling variable plaintext lengths without padding. It prevents bit errors from spreading.
- Counter Mode (CTR): CTR mode employs a counter to construct a key stream, XORing it with plaintext to generate ciphertext. It's highly parallelizable and doesn't require padding.

2.3 Hashing Algorithms

The article by (Khande et al.; 2021) explores the issues people encounter in managing different passwords for various online accounts and suggests a solution through a safe password manager. It underlines the risks associated with weak passwords and the popularity of cyber-attacks targeting password vulnerabilities. The paper underlines the importance of password security and proposes a cloud-based password management tool combining AES-256 encryption and SHA-256 hashing for increased protection. The process entails constructing a master password, deploying encryption algorithms, and including a random password generator to ensure solid security measures. Practical implementation details and the benefits of the suggested system are discussed, highlighting the relevance of password managers in preserving sensitive information against cyber threats. The essay finishes by underlining the significance of password security and the function of password managers in securing user data in the digital age.

2.4 Privacy Protection and Data Compliance

The paper by (Xiang and Cai; 2021) addresses the challenge of balancing individual privacy protection with the utilization of health data for research and secondary purposes. They advocate for a 'civilized' approach that minimizes re-identification risks while enabling complex data analysis. The study proposes three key strategies for data

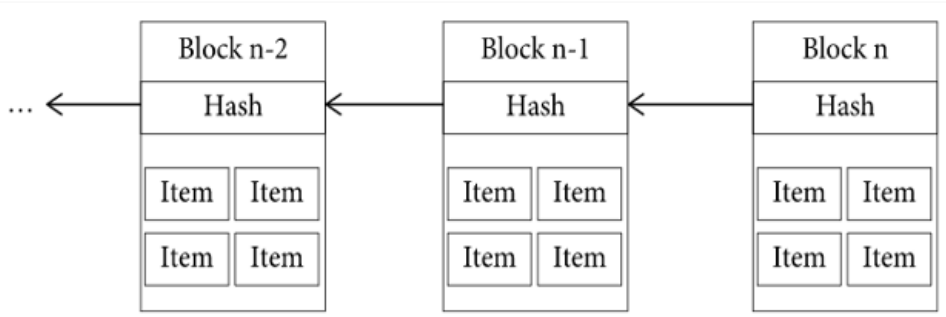

Figure 5: File System of a Blockchain

sharing: mitigating disclosure risks through techniques like data perturbation and suppression, anonymizing data by removing personal identifiers, and adopting collaborative learning models such as secure enclaves and federated learning to develop analytical models without directly sharing raw patient data. These strategies aim to enhance privacy protection while preserving the usability and generalizability of health data for research purposes.

3 Methodology

Identifying any difficulties that could prevent the successful implementation of this project is vital. Key considerations include:

- Release or breach of patient data while the file is being uploaded.
- Missing data that was received by the client.
- Patients' inability to update or change the information they have entered.
- The application's incapacity to retain patients' prior medical records.
- Possibility of involvement from other parties or information-stealing intruders.
- The possibility of computer networking nodes failing.
- Data kept in an unsecured format vulnerable to unauthorized access.
- Challenges experienced by both senders and recipients in accessing the data.

Based on insights from these elements and obstacles faced by previous research, it's necessary to execute the Electronic Health Record (EHR) system securely and reliably. Therefore, merging AES-256 encryption and SHA-256 hashing technique becomes necessary. After a comprehensive assessment of the literature review presented in the preceding sections, the authors of this paper propose the development of a web- based application designed to securely store all data in a SQL database and preserve all transactions occurring within the system. To do this, employing AES-256 encryption and SHA-256 hashing method in conjunction with blockchain technology is proposed to meet the project's objectives. Figure 5 demonstrates the file system in a blockchain.

The suggested thesis intends to meet the following objectives:

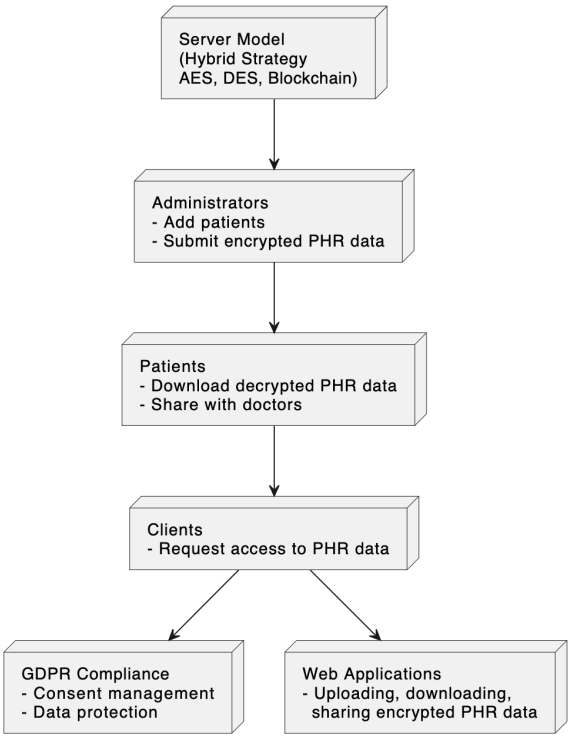

Figure 6: Example of how blockchain can be applied to medical records

- Develop a web-based application to securely store transactional data.
- Generate secret keys and apply cryptography and hashing techniques for safe data access.
- Implement AES-256 encryption and SHA-256 hashing algorithm with a salt value to permit secure data sharing among authorized parties, comparing their efficacy with other AES modes including ECB, CBC, CFB, OFB, and AES-CTR.
- Store shared data on an off-chain blockchain ledger to boost security and transparency.
- Provide users access to keys created by the off-chain blockchain ledger.
- Evaluate the model based on parameters such as time and spatial complexity.
- Compare the results of AES-256 encryption and SHA-256 hashing method with a salt value to determine their efficacy.

Previous systems had limits exposing the server model to dangers, leading to the adoption of a hybrid technique in this thesis. Combining AES-256 encryption, SHA-256 hashing algorithm with a salt value, and off-chain blockchain technology, this hybrid solution operates on a SQL Server, acting as the communication medium between administrators, patients, and customers. This section focuses on the methodologies involved in executing this approach.

The suggested thesis delineates a comprehensive approach for accessing Electronic Health Records (EHRs). The EHR generation process commences with doctors or medical technologists, combining varied medical data types into electronic health records

(EHRs). To alleviate time constraints, a desktop plug-in provides seamless transfer of EHR data to a web application, facilitating efficient communication between users and healthcare providers. Encryption keys developed using AES-256 and SHA-256 enable secure data transit. Patients decrypt requested information using private keys, accelerating responses from healthcare facilities. Blockchain principles bolster system integrity, while transactional ledgers preserve crucial data. Client requests for EHR access are met with patient agreement, guaranteeing respect to privacy rules and creating transparency and responsibility in data handling.

4 Design Specification

The Advanced Encryption Standard (AES) remains as the prevailing encryption method internationally, however it boasts various versions.A comprehensive investigation was conducted to evaluate various versions of AES in order to determine the algorithm that achieves the highest throughput. Additionally, each version was thoroughly reviewed based on encryption time in this investigation, seeking to find any potential faults within the method. To reinforce perceived vulnerabilities in the method, its security was further enhanced utilizing hashing techniques.

Through this investigation, the aim was to find the optimum AES variation in terms of both performance and security. By assessing encryption time and throughput, insights into the efficiency and resilience of each modification were expected to be gleaned. Moreover, by utilising hashing algorithms to remedy found holes, the overall security posture of the AES algorithm was intended to be enhanced, ensuring its resilience against potential attackers in diverse scenarios.

5 Implementation

The main goal of this thesis is to create a hybrid model that strengthens the server model's security against potential attacks by combining SHA-256 hashing algorithm, AES-256 encryption, and the PBKDF2 key generation method. Building a 32-byte key that will be used in AES-256 encryption operations is the specific objective. This framework also incorporates blockchain technology to guarantee the security of financial transactions within the healthcare sector. Research Objective 2 is covered in this section.

A significant component of this system is key creation, whereby the encryption mechanism generates keys, which are then dispersed across communication participants. It is possible for both parties to exchange and upload data files through a communication channel, but it is vital to safeguard the server model against attacks during this process, particularly if key sharing takes place over an unsecure connection. Nonetheless, the goal of the suggested thesis is to handle these issues without compromising the system's general security.

This research aims to protect a healthcare system's communication channel using the SHA-256 hashing technique and AES-256 encryption. Using these techniques, a random key is generated on the sender's end and sent to the recipient over the communication channel. To guarantee its security, the supplied key is encrypted before to delivery.

The recipient uses an authentication technique to decode the key on their end after receiving it. The parties receive the details of this authentication process through their individual email addresses. In order to enable the safe transfer of Electronic Health

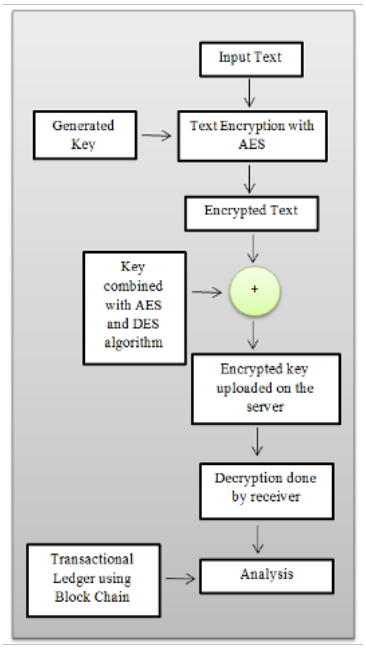

Figure 7: Flowchart of steps taken for the encryption study

Records (EHR), the receiver must first decrypt the encrypted key after receiving it via email together with the related login credentials.

The system design advises the implementation of numerous critical components to meet its objectives:

1. **Hybrid Encryption Model:** The core of the system design is building a hybrid encryption model that includes both AES-256 encryption and SHA-256 hashing algorithm . This technology provides robust encryption of data transmitted across the communication channel, using the benefits of both algorithms to increase security.

2. **Key Generation Mechanism:** A key generation mechanism is developed to create a 32-byte key that will be used in the AES-256 encryption procedure. This system should assure the development of strong, random keys to enhance the security of the encryption process.

3. **Blockchain Integration:** Blockchain technology is integrated into the system design to secure financial transactions inside the healthcare system. This requires building blockchain protocols and ways to maintain the integrity and immutability of transaction records.

4. **Secure Communication Channel:** The concept encompasses the establishment of a secure communication channel over which data files can be transmitted and posted between communicative parties. This channel should be resistant to eavesdropping and tampering to preserve the confidentiality and integrity of transmitted data.

5. **Authentication methods:** Authentication methods are established to authenticate the identity of communication partners and ensure that only authorized users can access encrypted data. This entails the use of login credentials and email verification techniques to verify users throughout key exchange and decryption processes.

6. **Email Communication Protocol:** The system employs email communication protocols to exchange authentication information and encrypted keys between communicating parties. This ensures secure flow of essential information over the communication channel.

7. **Decryption technique:** The design contains a decryption procedure that allows the recipient to decrypt the encrypted key using the authentication information obtained via email. This process should be easy and quick to allow the secure transmission of Electronic Health Records (EHR) between parties.

The system design (Research Objective 5) is accomplished in the following way:

The system features vital components including the Website Server, permitting connection between administrators and users, alongside File-sharing and Downloading Options, enabling numerous communication activities. Sender-Recipient Communication requires inputting queries, encrypted using AES-256 encryption and SHA-256 hashing technique, transforming them into encrypted text files. Encryption, vital for secure data transfer, utilizes AES-256 encryption and SHA-256 hashing. Key Generation and Distribution occur during encryption-decryption, ensuring safe access for administrators and users. Keys are rapidly posted to the Website Server, boosting security and accessibility. Authentication ensures safe data access, with decryption keys integrated inside the agreement. Implementation Tools like Visual Studio IDE expedite development, whereas Transaction Handling employs a transactional ledger for safe data retention. Evaluation and Testing, employing files of varied sizes, insure system robustness and efficiency through comprehensive testing.

5.1 Specifications for AES-256 and SHA-256 Algorithm Implementation

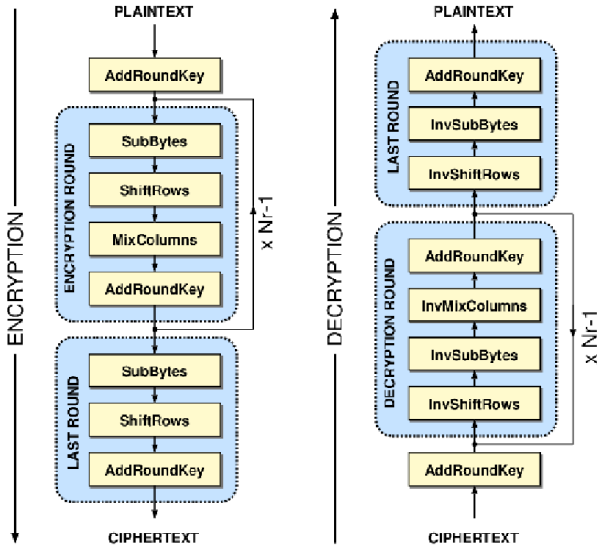

Figure 8: AES encryption architecture (Arrag et al.; 2012)

The AES-256 method has a key size of 256 bits, offering a higher level of security compared to AES-128 (Bi Irie guy cedric; 2018). With a bigger key size, AES-256 offers enhanced resistance to brute force assaults, making it more robust for encryption

purposes. AES-256 is a two-step encryption process largely utilizing substitution and transposition techniques, similar to AES-128. However, the greater key size boosts the security of AES-256, offering stronger protection of critical data. This section aligns with Research Objective 4.

In addition to AES-256 encryption, the implementation contains the SHA-256 hashing technique with PBKDF2 (Password-Based Key Derivation Function 2). SHA-256 strengthens the security of AES-256 encryption by deriving cryptographic keys from passwords or passphrases, making it computationally demanding for attackers to reverse the process and recover the original password or key.

By implementing SHA-256 , the security of AES-256 encryption is substantially increased. overall, the combination of AES-256 encryption and SHA-256 hashing algorithm provides comprehensive protection for sensitive data stored on the server. This strategy assures that even if an attacker has access to the encrypted data, the usage of AES-256 encryption and SHA-256 hashing algorithm considerably enhances the difficulty of deciphering the information, hence boosting data security and integrity.

6 Evaluation

Three use cases were constructed to answer the three research questions.

6.1 Use case 1: which AES variant is the fastest

A performance evaluation of operation modes was undertaken utilising the C-Sharp programming language in the inquiry. The implemented architecture utilises managed wrappers for AES-128, making use of classes from the System.Security.Cryptography and System classes. The AES class in C-sharp is responsible for providing cryptographic capabilities, encompassing encryption and decryption. It acts as the fundamental component for performing cryptographic operations.

To execute the experiment, a computer with an Apple M1 processor, including a 2.30 GHz CPU and 6 GB of RAM has been used . The computer was employed for the purpose of encrypting files with a varying size range of 500 KB to 200 MB. The value of s for the CFB and OFB modes is set to 16 bits.

The evaluation of performance was carried out based on the metrics of encryption time and throughput, which were specified as follows:

1. **Encryption Time:** This refers to the duration required for an encryption algorithm to transform a plaintext into a ciphertext.

2. **Throughput:** The calculation involves dividing the encryption time (KB/sec) by the total encrypted plaintext in kilobytes. In the context of encryption, throughput reflects the speed of encryption. An increase in throughput is associated with a decrease in power consumption.

6.1.1 Evaluating Performance Based on Encryption Time

Based on the performance evaluation of encryption time, as depicted in below table and chart, the following observations were made:

1. **ECB Mode**: The encryption time for ECB mode was found to be the lowest compared to other modes. This suggests that ECB mode performs encryption more efficiently than the other modes evaluated.

2. **Other Modes (CFB, OFB, CBC, CTR)**: The encryption times for CFB, OFB, CBC, and CTR modes were observed to be similar. This indicates that these modes exhibit comparable performance in terms of encryption time.

3. **Negligible Differences in Small Files**: For small files, particularly those less than 10 MB in size, the differences in encryption time between the various modes were found to be negligible. This suggests that the choice of encryption mode may have less impact on performance for smaller files.

Overall, these findings provide insights into the performance characteristics of different encryption modes, with ECB mode demonstrating superior efficiency in terms of encryption time. This section covers Research Objective 3.

Table 1: Encryption Time Comparison

File Size (KB)	ECB (ms)	CBC (ms)	CFB (ms)	OFB (ms)	CTR (ms)
531	94.4	97.8	109.8	107.7	95.4
2,116	99.5	105.0	115.8	115.7	104.9
10,480	184.5	210.8	217.1	217.1	215.7
52,383	613.8	713.5	732.2	735.8	735.1
108,096	1204.5	1401.2	1419.9	1413.0	1422.9
216,190	2348.5	2744.6	2874.3	2788.0	2794.9

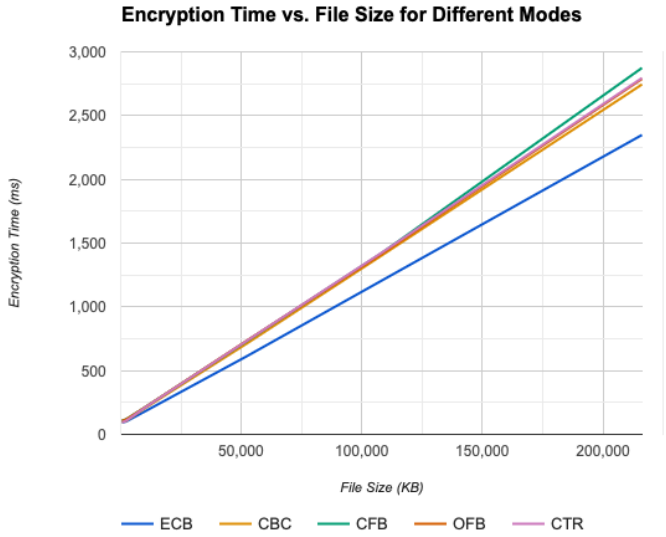

Figure 9: Encryption Time Analysis

6.2 Use case 2: Performance Evaluation Based on Throughput

AES was used in thisexperiment, and the fixed key size was 128 bits. Throughput varies slightly between the different modes of operation, on average. The ECB mode, in particular, showed the lowest throughput when compared to the other modes, suggesting a higher power consumption. The numerical results of the encryption throughput comparison are shown in below Table and chart . It was noted that during encryption, the throughput of the CBC and CFB algorithms was somewhat higher. On the other hand, during encryption, the throughput levels of the OFB and CTR modes were equal.

Table 1: Average Throughput for Each Mode of Operation

Mode	Average Throughput (MB/s)
ECB	7.28
CBC	8.45
CFB	8.74
OFB	8.73
CTR	8.68

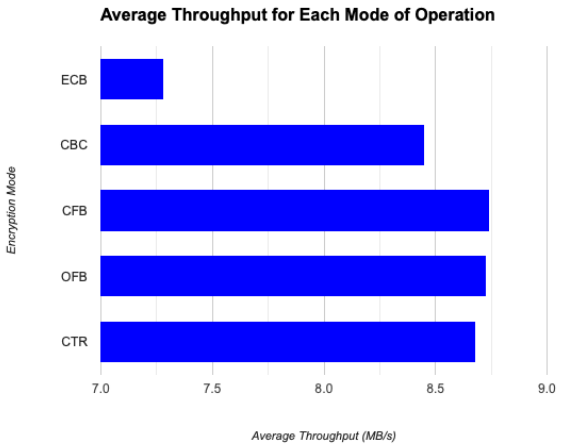

Figure 10: Average throughput for AES modes

6.3 Use case 3: Hashing passwords for more secure implementation of AES (application to medical records)

In AES encryption, the security of the encryption key is crucial, and a typical mistake in coding AES implementations is employing the encryption key directly in plain text. Instead, adopting a hash key obtained from the original encryption key can considerably boost security. By utilising a hash function like SHA-256, the encryption key is turned into a fixed-size hash value, making it more resistant to brute-force attacks and other cryptographic flaws. Without utilising SHA-256 or equivalent secure hash algorithms, the raw encryption key could be subject to numerous attacks, including dictionary attacks and rainbow table attacks, thereby compromising the confidentiality and integrity

of encrypted data. Therefore, employing SHA-256 for key hashing offers an additional degree of security to AES encryption implementations, securing sensitive information from unauthorized access and exploitation. Figure 11 shows the proposed hashing algorithm used in this thesis.

```
private string performHash(string data)
{
    byte[] bytes = Encoding.Unicode.GetBytes(data);
    SHA256Managed hashstring = new SHA256Managed();
    byte[] hash = hashstring.ComputeHash(bytes);
    string hashString = string.Empty;
    foreach (byte x in hash)
    {
        hashString += String.Format("{0:x2}", x);
    }
    return hashString;
}
```

Figure 11: Typical usage of SHA-256 hashing algorithm

In addition to employing SHA-256 for key hashing, SMTP (Simple Mail Transfer Protocol) has been utilised in conjunction with the Gmail API to securely transmit the hashed key password directly with the user. This solution ensures that the encrypted key password reaches the intended recipient securely and reliably through email communication. By leveraging the Gmail API, Google's sophisticated security measures and encryption technologies are applied, increasing the secrecy and integrity of the key password transmission process. This integration of SMTP and the Gmail API adds an extra layer of protection to the AES implementation, simplifying secure key distribution and boosting the overall security posture of the system.

```
SmtpClient SmtpServer = new SmtpClient();
MailMessage mail = new MailMessage();
SmtpServer.Credentials = new System.Net.NetworkCredential("debashishsarangi0@gmail.com", "ycuoiqqagkmvfpgz");
SmtpServer.Port = 587;
SmtpServer.EnableSsl = true;
SmtpServer.Host = "smtp.gmail.com";
mail = new MailMessage();
mail.From = new MailAddress("debashishsarangi0@gmail.com");

mail.To.Add(email);
mail.Subject = "Authentication  Details";
mail.Body = "File has been shared to your account to acces that file your Key is "+ key;
SmtpServer.Send(mail);
```

Figure 12: Example of SMTP protocol using Google API to share the encryption key. It is part of a sample application for the secure handling of medical records (Use Case 3)

6.4 Discussions

Here, the most prevalent AES block cipher modes of operation were evaluated side by side, analysing their encryption times and throughput. Following NIST recommendations for block ciphers, this research is focused on the following modes of operation: ECB, CBC, CFB, OFB, and CTR. From what can be inferred, ECB has the fastest encryption time among the modes investigated. Compared to large files, the differences between the modes are small. On the other hand, when dealing with files of extremely large sizes, the performance of different modes varies significantly.

6.4.1 Use case 1: encryption speed

It was found that the ECB encryption method displayed the highest speed, possibly due to its simplicity and efficiency, as it encrypts each block separately without considering past blocks. This streamlined procedure leads in speedier processing compared to other options. It was found that the ECB encryption method displayed the highest speed, possibly due to its simplicity and efficiency, as it encrypts each block separately without considering past blocks. This linearity shows that regardless of the mode employed, the encryption time grows proportionally with the complexity of the data being encrypted. This observation emphasises the consistent computational behavior of AES encryption across different modes while handling image data, which was chosen as a typical data type for its relevance to medical records. The linearity found in the encryption time for all AES algorithms while encrypting photos can be traced to the inherent features of image data. Images often consist of a huge number of pixels, each requiring encryption individually. As a result, the encryption time grows linearly with the size and complexity of the image. Unlike text or other data types where the encryption time may vary based on the content, the uniform structure of photos leads to a continuous increase in encryption time as the image size expands. This linear connection demonstrates the regular computing performance of AES encryption while processing image data, regardless of the encryption technique chosen.

6.4.2 Use case 2: throughput

In this situation, Cipher Feedback (CFB) mode emerged as the most efficient alternative, exhibiting the best throughput across the encryption modes investigated. The success of CFB mode can be due to its unique feedback mechanism, where the output of each encryption iteration is sent back into the algorithm for the encryption of subsequent blocks. This strategy not only permits parallel processing, optimizing throughput, but also ensures a high level of diffusion in the ciphertext, limiting the possibility of patterns in the encrypted data. Given these advantages, CFB mode is proven to be well-suited for real-time processing circumstances, such as those experienced in the project, where effective encryption of streaming data is crucial.

ECB has however the lowest throughput and this low performance may arise from its distinct encryption algorithm. ECB encrypts each block independently without considering the relationship between neighbouring blocks, resulting in a lack of diffusion in the ciphertext. This absence of dispersion might lead to patterns in the encrypted data, rendering it susceptible to particular attacks and lowering overall throughput. In contrast, some modes like CBC introduce feedback mechanisms or chaining, boosting security but potentially slowing down encryption due to the increased computational burden. Thus, whereas ECB promotes simplicity, its vulnerability to attacks may lead to reduced throughput compared to more secure options.

6.4.3 Use case 3: hashing

The introduction of SHA-256 (or longer) for key hashing considerably strengthens the security of AES encryption by transforming the encryption key into a fixed-size hash value, rendering it more resilient against brute-force attacks and cryptographic flaws. Without adopting powerful hash algorithms like SHA-256, the raw encryption key remains open to numerous assaults, such as dictionary attacks and rainbow table attacks, posing a

threat to the security and integrity of encrypted data. Integrating SHA-256 ensures an additional degree of security, keeping critical information from illegal access and misuse. Moreover, employing secure SMTP in conjunction with the Gmail API permits secure sharing of the hashed key password directly with the user, leveraging Google's comprehensive security procedures to enhance confidentiality and integrity throughout email communication. This combination method increases key distribution and overall security measures, hardening the AES implementation against potential attackers.

In addition to the adoption of SHA-256 for key hashing, users are encouraged to strengthen the security of their encryption keys by adopting best practices for key management. Instead of storing the key in their email, where it may be vulnerable to illegal access, users are urged to delete the email containing the key and utilize a password management software. Password manager software offers a secure way to store and manage passwords by encrypting them using strong encryption techniques. This double layer of encryption ensures that even if the password manager's data is compromised, the encryption key remains protected, thus preserving important information from future intrusions. By adopting this strategy, users can reinforce the security of their encryption keys and decrease the possibility of illegal access to their protected data.

7 Conclusion and Future Work

This research endeavor resides in its detailed assessment of the performance, security, and efficiency of several AES encryption modes and hashing approaches specifically adapted for blockchain applications. The specific use case in mind was the handling of medical records, as described in Use Case 3. While prior publications have studied particular features of encryption or hashing in isolation, this study presents a comprehensive analysis that includes their combined impact on security and performance. Furthermore, the research adds fresh insights into the practical implementation and design of encryption techniques within real-world blockchain infrastructures, addressing a gap in the existing literature

Ultimately, a thorough analysis of many AES encryption variations reveals insights into their performance characteristics and security implications. Furthermore, the paramount importance of hashing algorithms, specifically AES ECB and SHA-256, in enhancing additional encryption security is emphasised. However, this speed benefit comes at the tradeoff of diminished security due to probable patterns in the encrypted data. Conversely, modes like CBC offer greater security through chaining methods but may display slightly reduced throughput due to higher computing complexity. The results emphasise the importance of considering both performance and security factors when choosing an AES encryption mode. Each version has its own set of trade-offs that need to be thoroughly evaluated according to the specific needs of the application.

Furthermore, this research underlines the crucial importance of hashing algorithms, notably SHA-256, in enhancing the security of AES encryption implementations. By translating encryption keys into fixed-size hash values, SHA-256 mitigates the danger of brute-force attacks and cryptographic flaws, preserving sensitive information from unauthorized access. Furthermore, the incorporation of SMTP with the Gmail API ensures secure key distribution, utilising Google's strong security protocols to enhance the confidentiality and integrity of email communications. Overall, this work offers important insights into how to optimise AES encryption for a variety of use cases, highlighting the

significance of putting in place thorough security measures to fend off evolving cyber-threats and maintain the integrity of encrypted data in practical applications.

References

Almuhammadi, S. and Al-Hejri, I. (2017). A comparative analysis of aes common modes of operation, *2017 IEEE 30th Canadian Conference on Electrical and Computer Engineering (CCECE)*, pp. 1–4.

Arrag, S., Hamdoun, A., Tragha, A. and Khamlich, S. (2012). Design and implementation a different architectures of mixcolumn in fpga, *International Journal of VLSI Design Communication Systems* **3**.

Battah, A., Iraqi, Y. and Damiani, E. (2021). Blockchain-based reputation systems: Implementation challenges and mitigation, *Electronics* **10**(3).
URL: *https://www.mdpi.com/2079-9292/10/3/289*

Bi Irie guy cedric, T. (2018). A comparative study on aes 128 bit and aes 256 bit, *INTERNATIONAL JOURNAL OF COMPUTER SCIENCES AND ENGINEERING* **volume 6**: 30–33.

Boey, K., Lu, Y., O'Neill, M. and Woods, R. (2010). Differential power analysis of cast-128, *2010 IEEE Computer Society Annual Symposium on VLSI*, pp. 143–148.

El-Semary, A. M. and Azim, M. M. A. (2015). Counter chain: A new block cipher mode of operation, *Journal of Information Processing Systems* **11**(2): 266–279.

Garcia, C. R., Aguilera, A. C., Olmos, J. J. V., Monroy, I. T. and Rommel, S. (2023a). Quantum-resistant tls 1.3: A hybrid solution combining classical, quantum and post-quantum cryptography, *2023 IEEE 28th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*, pp. 246–251.

Garcia, C. R., Aguilera, A. C., Olmos, J. J. V., Monroy, I. T. and Rommel, S. (2023b). Quantum-resistant tls 1.3: A hybrid solution combining classical, quantum and post-quantum cryptography, *2023 IEEE 28th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)* pp. 246–251.
URL: *https://api.semanticscholar.org/CorpusID:268725753*

Hook, D. (2005). *Beginning Cryptography with Java*, John Wiley & Sons.

Huang, K. (2013). A novel structure with dynamic operation mode for symmetric-key block ciphers, *International Journal of Network Security Its Applications* **5**: 17–36.

Huang, K.-T., Lin, Y.-N. and Chiu, J.-H. (2013). Real-time mode hopping of block cipher algorithms for mobile streaming, *International Journal of Wireless Mobile Networks* **5**: 127–142.

Jaswal, K., Choudhury, T., Chhokar, R. L. and Singh, S. R. (2017). Securing the internet of things: A proposed framework, *2017 International Conference on Computing, Communication and Automation (ICCCA)*, pp. 1277–1281.

Kelsey, J., Schneier, B. and Wagner, D. (1996). Key-schedule cryptanalysis of idea, g-des, gost, safer, and triple-des, *Advances in Cryptology—CRYPTO'96*, Springer, pp. 237–251.

Khande, R., Ramaswami, S., Naidu, C. and Patel, N. (2021). An effective mechanism for securing and managing password using aes-256 encryption pbkdf2, *INTERNATIONAL JOURNAL OF ELECTRICAL ENGINEERING AND TECHNOLOGY* **12**.

Kim, S., Kwon, Y. and Cho, S. (2018). A survey of scalability solutions on blockchain, *2018 International Conference on Information and Communication Technology Convergence (ICTC)*, pp. 1204–1207.

Koehler, S. C. (2001). Method and system for authenticating digital certificates issued by an authentication hierarchy.

Koukou, Y., Othman, S. and Md Siraj, M. (2016). Comparative study of aes, blowfish, cast-128 and des encryption algorithm, *IOSR Journal of Engineering* **06**: 01–07.

Mannan, R., Sethuram, R. and Younge, L. (2019). Gdpr and blockchain: A compliance approach, *International Journal for the Data Protection Officer, Privacy Officer and Privacy Counsel* **3**: 7. Accessed: 5 April 2024.
URL: *https://heinonline.org/HOL/LandingPage?handle=hein.journals/idpp3div=16id=page=*

Maqsood, F., Ahmed, M., Ali, M. M. and Shah, M. A. (2019). Cryptography: A comparative analysis for modern techniques, *Dept. Computer Science & Information Technology, Superior University, Lahore, Pakistan.* .
URL: *https://pdfs.semanticscholar.org/8331/4e07dfb9d15145fa79734f63e47932866101.pdf/1000*

Mare, S. F., Vladutiu, M. and Prodan, L. (2011). Secret data communication system using steganography, aes and rsa, *Design and Technology in Electronic Packaging (SIITME), 2011 IEEE 17th International Symposium for*, IEEE, pp. 339–344.

McEliece, R. J. (2012). *Finite Fields for Computer Scientists and Engineers*, Vol. 23, Springer Science & Business Media.

Miyachi, K. and Mackey, T. K. (2021). hocbs: A privacy-preserving blockchain framework for healthcare data leveraging an on-chain and off-chain system design, *Information Processing Management* **58**(3): 102535.
URL: *https://www.sciencedirect.com/science/article/pii/S0306457321000431*

Muin, M. A., Muin, M. A., Setyanto, A., Sudarmawan and Santoso, K. I. (2018). Performance comparison between aes256-blowfish and blowfish-aes256 combinations, *2018 5th International Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE)*, pp. 137–141.

Nagalakshmi, V., Babu, I. R. and Avadhani, P. (2011). Modified protocols for internet key exchange (ike) using public encryption and signature keys, *2011 Eighth International Conference on Information Technology: New Generations*, pp. 376–381.

Paar, C. and Pelzl, J. (2009). *Understanding Cryptography: A Textbook for Students and Practitioners*, Springer Science & Business Media.

Paradesi Priyanka, M., Kaur, N., Nazir, N., Ali Khan, A., Vikram Singh, M., Kaur, M., Behera, T., Rakhra, M. and Dahiya, O. (2022). A comparative review between modern encryption algorithms viz. des, aes, and rsa, *2022 International Conference on Computational Intelligence and Sustainable Engineering Solutions (CISES)*, pp. 295–300.

Rahman, M. M., Akter, T. and A, R. (2016). Development of cryptography-based secure messaging system, *Journal of Telecommunications System & Management* **05**.

Rivest, R. L., Shamir, A. and Adleman, L. (1978). A method for obtaining digital signatures and public-key cryptosystems, *Communications of the ACM* **21**(2): 120–126.

Rueppel, R. A. (1986). Stream ciphers, *Analysis and Design of Stream Ciphers*, Springer, pp. 5–16.

Singh, G. (2013). A study of encryption algorithms (rsa, des, 3des and aes) for information security, *International Journal of Computer Applications* **67**(19).

Smid, M. E. (2021). Development of the advanced encryption standard, *Journal of research of the National Institute of Standards and Technology* **126**: 126024.

Stafford, T. F. and Treiblmaier, H. (2020). Characteristics of a blockchain ecosystem for secure and sharable electronic medical records, *IEEE Transactions on Engineering Management* **67**(4): 1340–1362.

Tahir, R., Javed, M. and Cheema, A. (2008). Rabbit-mac: Lightweight authenticated encryption in wireless sensor networks, pp. 573 – 577.

Vaidehi, M. and Rabi, B. J. (2014). Design and analysis of aes-cbc mode for high security applications, *Second International Conference on Current Trends In Engineering and Technology - ICCTET 2014*, pp. 499–502.

Xiang, D. and Cai, W. (2021). Privacy protection and secondary use of health data: Strategies and methods, *BioMed Research International* **2021**: 6967166.
URL: *https://doi.org/10.1155/2021/6967166*