Securing Financial Transactions with Hybrid ECC and AES Encryption Algorithm

MSc Research Project
Programme Name

Anuja Moruskar
Student ID: X22232362

School of Computing
National College of Ireland

Supervisor: Prof. Niall Heffernan

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Anuja Mahadev Moruskar		
Student ID:	X22232362		
Programme:	MSc in Cybersecurity	**Year:**	2023-24
Module:	MSc Research project		
Supervisor:	Prof. Niall Heffernan		
Submission Due Date:	12/08/2024		
Project Title:	Securing Financial Transactions with Hybrid ECC and AES Encryption Algorithm		
Word Count:	6485 **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: Anuja Moruskar

Date: 12/08/2024

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):	

Securing Financial Transactions with Hybrid ECC and AES Encryption Algorithm

Anuja Moruskar
X22232362

Abstract

Cloud computing has achieved good attention among organization and user. By using it users can easily store their data directly on third party vendors and can enjoy the services, application and storage. This report is going to present the development and analysis of the SecureFinance system, which is a complex solution, and it has been designed to manage financial data in a secure way. The study will focus on the implementation of a hybrid cryptographic model that merges Elliptic Curve Cryptography (ECC) with Advanced Encryption Standard (AES). The primary aim is to increase data protection and performance in managing sensitive financial information as compared to traditional encryption methods. The report is going to find and evaluate the performance of ECC+AES against ECC+DES which shows its good performance and novelty as well. ECC provides strong security with smaller key sizes while AES gives strong encryption capabilities by making the hybrid approach highly good. The performance evaluation showed that ECC+AES delivers faster encryption and decryption times across various file sizes which also shows its practical advantages. The study includes deploying the SecureFinance system on Microsoft Azure in order to ensuring scalability and reliability in a cloud environment. It examines how well the system supports large-scale data management user access. By solving these key research questions related to security, performance and usability the report is going to contribute good data into the implementation of hybrid cryptographic techniques in financial systems.

1 Introduction

The motivation behind this project comes from the increasing need for robust security which do measures in financial systems due to the growing thing of cyber threats. As financial transactions and data management increasingly shift to digital platforms and cloud environments the risk of unauthorized access and data manipulation grows. There are some traditional encryption methods which is good but also face limitations in balancing security with performance which is mainly as data volumes and transaction complexities grow. This project aims to solve these challenges by implementing a hybrid type of encryption model combining ECC and AES. ECC gives strong security with smaller key sizes which also increasing computational performance and scalability while AES gives strong encryption capabilities for protect data. The combination of these techniques aims to deliver a more secure and good solution for managing sensitive financial information. By deploying this hybrid approach in a cloud-based banking system this project is going to look to set a new standard for data protection in financial environments for ensuring both high levels of security and operational performance.

1.2 Aim of the study

The aim of this study is to develop and evaluate a secure banking system name as SecureFinance by using some advanced cryptographic techniques to secure the protection of sensitive financial data. This system has been designed to solve the important need for secure and good handling of customer information and transactions within a modern banking environment. The primary objectives of the study include implementing a hybrid cryptographic approach that combines ECC with AES to increase data security while maintaining high performance. For financial officers the system is going to provide some kind of functionalities to manage customer accounts, financial records and view shared files while securing data confidentiality and integrity with encryption. For bank clients the application is giving features to view and share financial records securely by of course increasing user experience and maintaining privacy. The study focuses on deploying the SecureFinance system on a cloud platform which is mainly using Microsoft Azure to secure scalability, reliability and accessibility. The chosen cryptographic methods which is ECC for good key exchange and AES for secure data encryption. They have been combined to create a secure communication channel and data protection mechanism. This hybrid approach is used to reduce the risks which is associated with data things and unauthorized access which is thereby increasing overall security. Also the research aims to evaluate the performance of the implemented cryptographic techniques in real-world things by analysing their effect on encryption and decryption times having different file types and sizes. By doing so the study will look to provide a good type of solution for secure financial data management in the banking sector by solving both security and performance challenges.

1.3 Research Objectives

The research objectives of this report are:

1. To design and implement the SecureFinance which can handle both customer and financial officer functionalities while providing strong security measures.
2. To combine ECC and AES to develop a hybrid encryption model that will increase data security and performance.
3. To evaluate the performance of ECC+AES vs ECC+DES in terms of encryption and decryption times across so many different file sizes and types.

1.4 Research Questions

How does the hybrid cryptographic model combining ECC and AES affect the security and performance of the SecureFinance system, and how does it compare to the ECC+DES model in terms of encryption and decryption efficiency?

2 Related Work

This chapter is presents an overview of different techniques used for increasing data security and integrity in cloud computing environments. A hybrid approach combining symmetric AES, and asymmetric (ECC, ElGamal) encryption methods has been mostly explored to secure data confidentiality and secure key management. There are some techniques like SHA-256 and Merkle Hash Tree (MHT) which have been used for verifying data integrity. Also, some other methods include genetic algorithms for key generation, steganography for hidden data transmission. These studies are going to show the performance of combining cryptographic and data verification techniques to solve security challenges in cloud computing.

2.1 Use of AES and ECC

In the research done by Bakro et al. (2022) a framework to increase the security of data stored in the cloud by with the help of blockchain technology combined with cryptographic algorithms which includes ECC and AES is proposed. The proposed approach aims to solve the weaknesses and data security challenges in IoT and cloud computing environments which are mainly used in different fields like medical, agricultural, educational, military and environmental sectors. The key challenge which has been faced in this approach is the combination and optimization of blockchain technology with ECC and AES to create a strong security mechanism. The results have showed that the proposed framework offers good security and performance by making it a good solution for securing cloud-stored data in the rapidly growing IoT landscape. Similarly, Rao and Sujatha (2023) have proposed a public cloud security technique using hybrid ECC to increase the security and performance of data storage and access in cloud environments. Their proposed approach generates keys with the help of lightweight Edwards curve and then it is followed by changing the private keys through Identity Based Encryption. To further optimize the process there is a key reduction method which has been used to shorten the keys which is thereby increasing the speed of AES encryption process. This proposed effective and secure hybrid approach offers a simple and efficient design that reduces the need for heavy computing power and resources. Another innovative research done by Rashid et al. (2022) proposed system using AES, RSA and ECC for data stored into insensitive, sensitive and highly sensitive category. They have stored each category on separate cloud servers by using different algorithms for each category for encryption and decryption.

2.2 Key Management Systems in Cloud Security

Ahmad et al. (2023) proposed a Hybrid Cryptographic approach and implemented the Key Management System (HCA-KMS) in a cloud environment. Aim of their research was to resolve cybersecurity and operational issues associated with large-scale data storage. The proposed HCA-KMS model solves the key exchange issues associated with AES and gives simpler and more reliable solution than ECC alone. Performance evaluation has been showed that HCA-KMS gives good and high levels of security for healthcare information. Also, their proposed model achieved good results in terms of speed compared to other algorithms. Another study given by Orobosade et al. (2020) presented a hybrid encryption algorithm which is ECC+AES. The main problem found in this approach is combining AES and ECC to maximize security while maintaining good performance. First, they used the AES algorithm for encryption, and then the ECC algorithm is used to encrypt the AES key in the cloud. The results have showed that the proposed hybrid encryption model successfully gives a secure environment for cloud data by increasing both privacy and security as compared to existing methods

2.3 Data Integrity Verification Techniques

Anwar et al. (2021) proposed the use of the SHA-256 hash function which is coupled with the Merkle-Damgård construction method to increase the performance and security of certificate data security. Similarly, Gangadharaiah and Shrinjayasacharya, (2024) have proposed a hybrid cryptographic approach to increase data integrity and security in cloud storage environments. The proposed system has combined the AES and ECC algorithms with Merkle Hash Tree (MHT) to protect user data stored in the cloud. This method found the growing problems of unauthorized access, hacking and malicious activities in cloud storage. The data is first encrypted using a hybrid AES-ECC encryption technique that generates both private and public keys. This encrypted data is then divided into so many blocks and tags for these blocks are created using SHA-256 which gives the hash value for each segment. Simulation analysis has done by the researcher which showed that the proposed approach achieved very good performance metrics like encryption time of 190 seconds, decryption time of 124 seconds, key generation time of 5.9 seconds and total execution time of 440 seconds.

2.4 Hybrid Encryption Techniques

Tahir et al. (2021) combined genetic algorithm (GA) with cryptographic techniques to solve data integrity and privacy issues in cloud computing. The main challenge in this approach is to combine the GA with cryptographic methods to increase security while maintaining performance. CryptoGA's performance was evaluated with use of known parameters such as execution time, throughput, key size and the avalanche effect. Malar et al. (2022) have been proposed an advanced approach to increase the protection of cloud data using a combination of cryptography, steganography and hash functions. The research uses the Blowfish algorithm for encryption which is actually an Embedded Least Significant Bits (E-LSB) algorithm for steganography and the SHA-256 for good control. The proposed approach starts by encrypting data using the Blowfish algorithm which is known for its strong security features. The encrypted data is then hide within an image using the E-LSB steganography technique. This dual-layered protection confirms that the data remains secure even if one layer was compromised. The SHA-256 hashing algorithm has been used to maintain the trust of the data by confirming that any unauthorized alterations can be detected. One of the main challenges found in this approach is securing the security of the steganography process which is mainly against data identification and data attacks. To evaluate the security of the steganography the researchers have done attacks in order to test the strengths of the E-LSB technique

Lai and Heng (2022) proposed a cloud storage system using a hybrid cryptography approach that combines the advantages of both symmetric and asymmetric key cryptographic techniques to solve important type of security issues such secrecy, authentication and data integrity. In this proposed system the AES is used to encrypt the data while the ElGamal algorithm been used to encrypt the keys before uploading the data to cloud storage. To further increase security a hash function SHA-256 is applied before the encryption process and after the decryption process to verify data trust by matching the derived hash values.

2.5 Comparative Analysis

Study	Proposed Approach	Challenges Faced	Results
(Bakro et al., 2022)	Hybrid encryption using AES and ECC	Securing data on the cloud, managing large data volumes	High security, improved performance over AES alone

(Rao and Sujatha, 2023)	Hybrid encryption using AES for encryption and Edwards curve for key generation	Ensuring privacy and security in the cloud	Enhanced data confidentiality, reduced encryption complexity
(Rashid et al., 2022)	Hybrid cryptography with AES for data encryption and ElGamal for key encryption, SHA-2 for data integrity	Protecting data integrity, handling lost or stolen accounts	Improved data confidentiality and integrity, Two-Factor Authentication for additional security
(Ahmad et al., 2023)	Hybrid Cryptographic Mode of Key Management System (HCA-KMS) in a Cloud Environment using AES and ECC	Ensuring security of data stored on cloud.	Improved security with good performance over AES, ECC.
(Tahir et al., 2021)	CryptoGA model using genetic algorithm for key generation and encryption	Ensuring data integrity and privacy in cloud computing	Improved integrity, privacy, performance over DES, 3DES, RSA, Blowfish, and AES
(Lai and Heng, 2022)	Hybrid cryptography with AEs for encryption, ElGamal for encrypting keys and SHA-256 hash function for verifying data integrity.	Efficiently calculating hashes, ensuring data integrity	Effective integrity and authenticity verification
(Orobosade et al., 2020)	Hybrid cryptography with AES for encryption and ECC for second level of encryption	Protecting user data stored on cloud	Good performance compared to AES and ECC
(Anwar et al., 2021)	Hybrid AES-ECC encryption and Merkle Hash Tree for data auditing	Maintaining data integrity and security in cloud storage	Efficient encryption and decryption times, improved data security in blockchain environment
(Malar et al., 2022)	Combination of cryptography (Blowfish) and steganography (E-LSB), SHA-256 for data integrity	Protecting against data identification and destruction attacks, ensuring data security	Enhanced security for data identification attacks, improved PSNR value

(Gangadharaiah and Shrinjayasacharya, 2024)	Hybrid AES-ECC encryption and MHT for data integrity in cloud storage	Protecting data against unauthorized access, maintaining data integrity	96% accuracy, efficient encryption and decryption times, improved data security and integrity in blockchain environment

3 Research Methodology

Cloud computing technology evolution has advanced user's data storing and accessing capabilities. However, this convenience brought challenges with it and data security is one for the key challenge. As financial transactions and data management are moving rapidly to digital platforms and cloud environments and the risk of unauthorized access and data manipulation also increases. Even though conventional encryption methods are effective but they still have shortcomings when it comes to balancing security with performance. This issue becomes more challenging when data volumes and transaction complexities are growing. This project aims to solve these challenges by implementing a hybrid method of encryption model by combining ECC and AES. ECC enhances computational performance and scalability by providing small but strong keys for encryption. AES provides strong encryption capabilities to safeguard the data. Deploying this hybrid encryption model in a cloud-based banking system will ensure both high level of security and operational performance.

3.1 AES

The Advanced Encryption Standard (AES) is a symmetric key encryption algorithm which is been established by the National Institute of Standards and Technology (NIST) in 2001. Designed to replace the older Data Encryption Standard (DES) AES has become the standard for securing sensitive data across different applications due to its performance. AES operates on blocks of data by encrypting each block individually to give high levels of security. AES uses the same key for both encryption and decryption which do secure key management practices to confirm the key remains confidential. The algorithm supports key sizes of 128, 192, and 256 bits by giving different levels of security. AES operates on a fixed block size of 128 bits and processes data with the help of multiple rounds of transformation by depending on the key size. For a 128-bit key AES performs 10 rounds of encryption and for a 192-bit key it performs 12 rounds also for a 256-bit key it does 14 rounds (Smid, 2021).

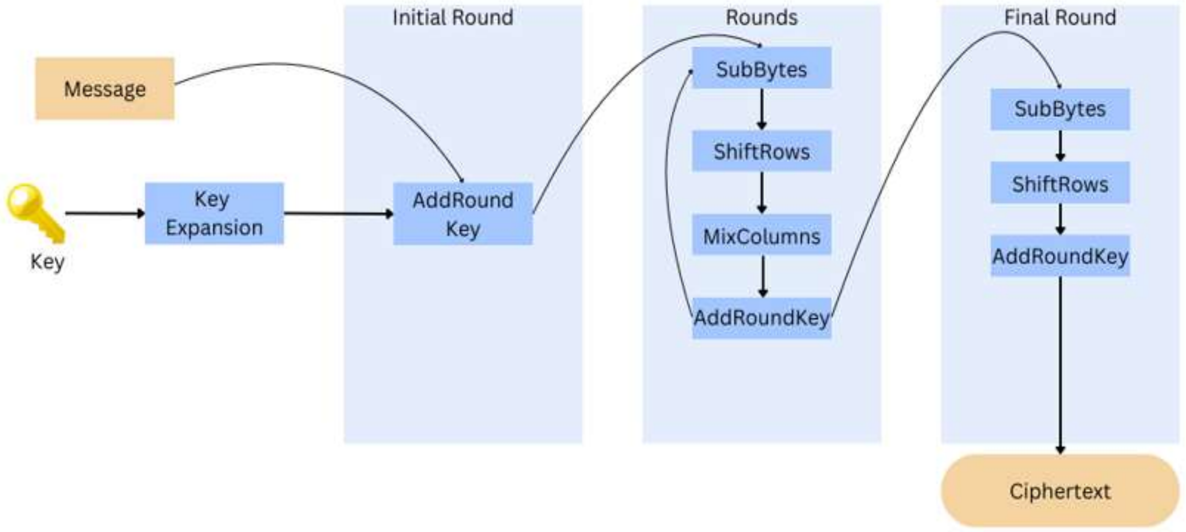


Figure 1: AES Encryption Process

3.2 ECC

ECC is a public key cryptography approach that uses the algebraic structure of elliptic curves over finite fields. It gives the same level of security as traditional cryptographic systems like RSA but with much smaller key sizes. This actually makes ECC mainly good for environments with constrained resources which includes mobile devices and embedded systems. In ECC the key pair consists of a private key and a corresponding public key. The private key is a randomly chosen integer while the public key is a point on the elliptic curve generated by multiplying the private key with a predefined point on the curve called the generator point. The security of ECC trust on the difficulty of the ECDLP which makes it impossible to reduce the private key from the public key. ECC's strength lies in its smaller key sizes as compared to other cryptographic methods. For example, a 256-bit key in ECC gives good security to a 3072-bit key in RSA. This reduction in key size leads to faster computations, lower power consumption and reduced storage requirements which are very important for performance which is mainly in secure communications over the cloud.

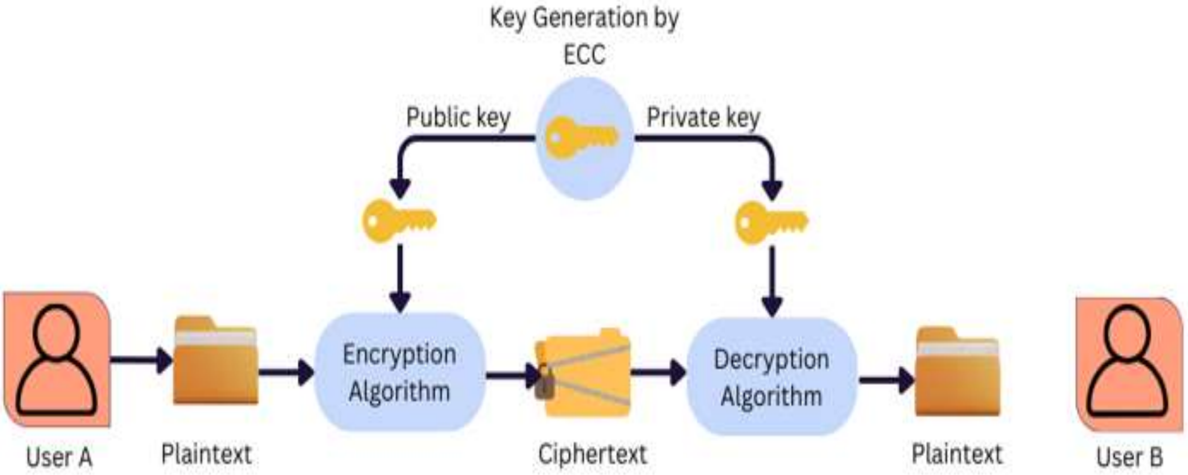


Figure 2: ECC Key generation.

3.3 Encryption Phase

The encryption phase is a very important component of the cryptographic process where plaintext data is transformed into ciphertext to protect it from unauthorized access (Jyothi et al., 2020). This phase does include application of an encryption algorithm to the original data using a specified key. During the encryption phase the algorithm uses the key to manipulate the data in a structured manner. In symmetric encryption the same key is used for both encryption and decryption. Popular symmetric algorithms include AES and DES which encrypt data in fixed-size blocks using a series of substitution and permutation operations. In asymmetric encryption there are two keys which use one is a public key for encryption and another is private key for decryption. Algorithms like RSA and ECC fall into this category by giving secure key exchange mechanisms. The encryption phase secures data confidentiality by making it unreadable to unauthorized parties. The strength of the encryption mainly depends on factors like key size and algorithm complexity. Good encryption is very important for securing sensitive information whether it’s stored data or data transmitted over networks by protecting against any threats like and cyber-attacks and all.

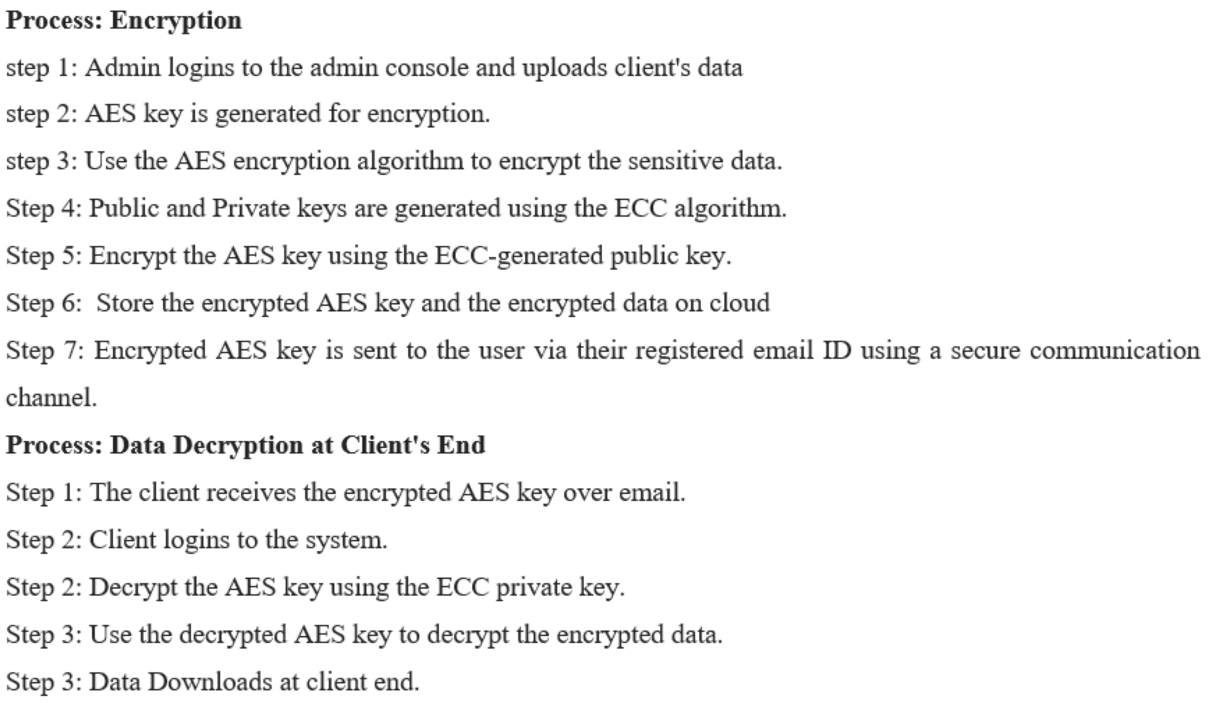
3.4 Decryption Phase

The decryption phase is the reverse process of encryption where ciphertext is transformed back into its original plaintext form by making the data readable again (Nadeem et al., 2023). This phase is very important again for retrieving the original information that was encrypted to protect it from unauthorized access. In the decryption phase the appropriate decryption algorithm and key are applied to the encrypted data. In symmetric encryption systems like AES and DES the same key used for encryption is also used for decryption. The decryption algorithm processes the ciphertext with the help of a series of inverse operations by obviously undoing the substitutions and permutations applied during encryption. In asymmetric encryption systems which includes RSA and ECC which is a pair of keys is used. The private key is used for decryption whereas public key used for encryption. This ensures that even if the public key is widely distributed and known only the holder of the private key can decrypt the message by maintaining the confidentiality of the communication. The decryption phase is important for data security by securing that only authorized parties with the correct key can access the original information. It allows encrypted data to be safely transmitted or stored and then retrieved in its original form when needed by giving a strong defence against unauthorized access and cyber threats.

4 Design Specification

The proposed model for securing financial data management contains a dual-layered architecture which focusses on robust security and efficient data management. This project creates platform for financial institutions and their clients to securely transfer data using Microsoft Azure cloud. The data involved in this exchange is mostly sensitive financial information like transaction details, tax documents, investment data and account balance statements etc. The admin account is created initially to manage administrative function of the system. The admin registration process includes creating a highly secure and authenticated account with multi-factor authentication (MFA) and strong password policies.

This will confirm that only authorized personnel can access the admin functions. Once registered the admin has full control over the financial operations which includes the ability to add and manage customer accounts and upload customer's data. Admins are responsible for looking customer registration which includes verifying customer identities and adding them into the system. Customers of the financial institution will have their personal and financial information securely stored and managed in the system. Customers can login to the system to download their financial information. The communication between the two parties should be secure as it involves sensitive information and to ensure this combination of symmetric and asymmetric encryption method is implemented. AES encryption algorithm is used in this project for encryption of the sensitive information. ECC algorithm generates pair of keys, one is private key and other is public key. Public key generated by ECC algorithm is used to encrypt the key of AES. The encrypted key is shared with user over secure communication channel. Key is shared with user over email ID which was collected during the registration process. Private key generated by ECC algorithm is then used to decrypt the AEC key and finally file is decrypted using the decrypted AES key. The Data is uploaded and encrypted at the admin's end while data decryption and downloading occur on the client's side.



Process: Encryption

step 1: Admin logins to the admin console and uploads client's data

step 2: AES key is generated for encryption.

step 3: Use the AES encryption algorithm to encrypt the sensitive data.

Step 4: Public and Private keys are generated using the ECC algorithm.

Step 5: Encrypt the AES key using the ECC-generated public key.

Step 6: Store the encrypted AES key and the encrypted data on cloud

Step 7: Encrypted AES key is sent to the user via their registered email ID using a secure communication channel.

Process: Data Decryption at Client's End

Step 1: The client receives the encrypted AES key over email.

Step 2: Client logins to the system.

Step 2: Decrypt the AES key using the ECC private key.

Step 3: Use the decrypted AES key to decrypt the encrypted data.

Step 3: Data Downloads at client end.

Figure 3: Algorithm of Proposed model.

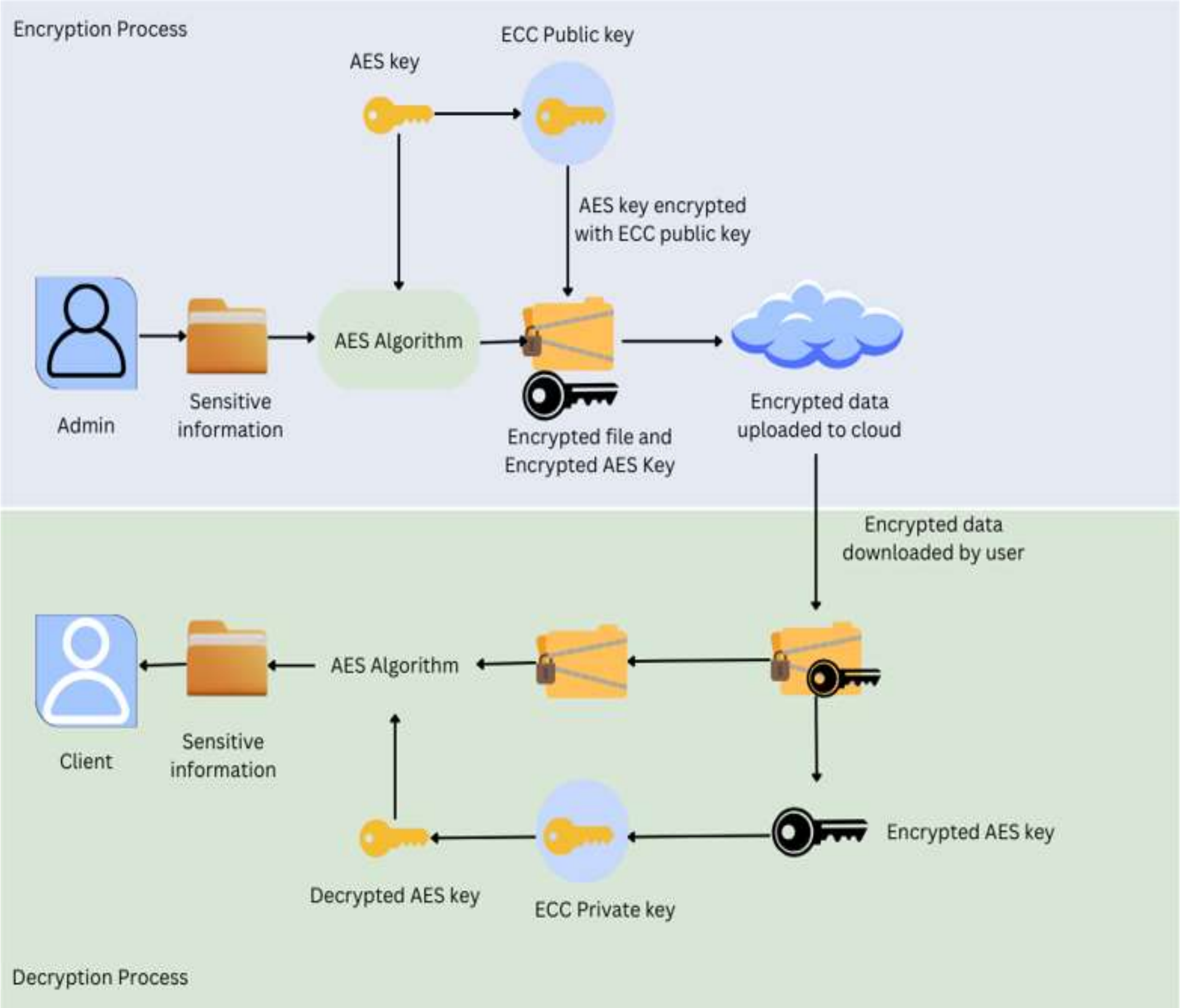


Figure 4: Architecture diagram of Hybrid Approach (ECC and AES).

Hybrid encryption using AES and ECC combines the strengths of both symmetric and asymmetric cryptographic methods and increases the security of financial data on the cloud. This approach is using the robust security features of AES for encrypting large amounts of data, while using speed and efficiency of ECC for key exchange. The flow of entire financial data management system is shown in the above diagram. Random key is generated for AES algorithm, and it is used to encrypt the uploaded data. This key is important for both parties hence it is then secured using the ECC algorithm. The public key generated from the ECC algorithm is used to encrypt AES key. The encrypted AES key is then sent to the recipient through a secure communication channel. After logging into the system successfully, client enter the Key received via email to download the decrypted data. This method helps in order to ensure that only authorized users can access the encrypted financial data.

This hybrid approach ensures data security as AES key cannot be access by user without private key generated from the ECC algorithm. The use of hybrid encryption method in the project provides high level of data security and protects from unauthorized access. Admin and authorized client have access to their dashboard to securely upload and download financial data respectively. This approach greatly improves the security and integrity of the financial system by allowing them to securely manage and exchange financial data

5 Implementation

The project is developed and deployed using multiple languages and tools. Visual Studio 2022 is used for this project as it supports multiple programming languages and framework. It is highly efficient and free to download. The entire model is developed using C# programming language and it is built on the .Net framework. The project uses Microsoft SQL management studio for managing the database. The application built to connect to the local as well as cloud database. The cloud platform used to deploy this application is Microsoft Azure. Microsoft azure cloud platform is easy to use and offers multiple services which makes it an ideal choice for this project. The proposed model has advantages of AES and ECC and allows fast data encryption and decryption along with securely managed key. The process begins with ECC generating a pair of public and private keys. The public key is then used to encrypt a randomly kind of generated symmetric AES key. This encrypted AES key along with the ECC-encrypted public key is securely transmitted to the recipient. Before transmission the newly generated AES key has been used to encrypt the financial data. This ensures that the data is securely encrypted before being sent. Client can then easily decrypt and access their financial data once received encrypted key is decrypted with the use of ECC privet key.

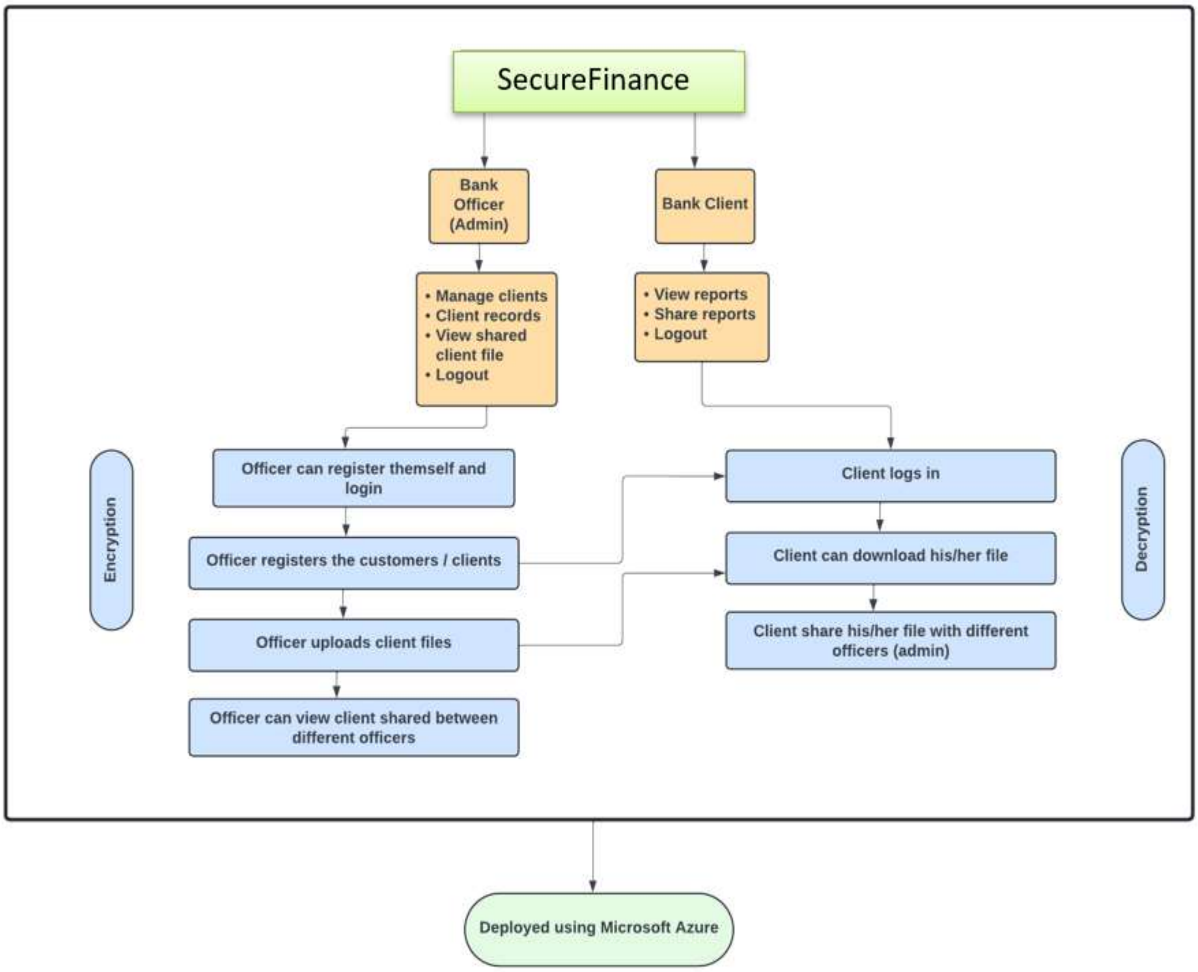

SecureFinance
Bank Officer (Admin)
Bank Client
• Manage clients
• Client records
• View shared client file
• Logout
• View reports
• Share reports
• Logout
Encryption
Officer can register themself and login
Officer registers the customers / clients
Officer uploads client files
Officer can view client shared between different officers
Client logs in
Client can download his/her file
Client share his/her file with different officers (admin)
Decryption
Deployed using Microsoft Azure

Figure 5: System Architecture.

5.1 System Architecture

The architecture of the system has been designed to provide a secure project for managing financial records and client information with a strong focus on data protection and operational reliability. The system architecture includes some important components. This project has been developed using .NET which serves as the core of the application by having interactions between admins and customers. It provides important functionalities for account management, transaction monitoring and data access by securing all operations are user-friendly and secure. The system uses a hybrid cryptographic approach combining ECC and AES. Before transmission the financial data is encrypted using the newly generated AES key by confirming its security before being sent. AES is used to encrypt and decrypt sensitive financial records and client information by securing that data remains confidential and protected from unauthorized access. ECC is used for secure key exchange by enabling the safe distribution of AES keys between the server and users. For deployment of the SecureFinance application is hosted on Microsoft Azure which provides required cloud infrastructure to secure scalability, reliability and high availability. Azure gives the important resources to handle changing loads and makes sure that the application remains accessible to both admins and customers. The backend database management system of the application is SQL Server which is used to store all financial records and client data. The combination of SQL Server with Azure guarantees that the database is both strong and protected against the potential threats. The architecture of this application plays an important role in achieving a high level of its security and performance. ECC and AES provides the good protection of data and use of Microsoft Azure cloud for hosting the application provides the flexibility and reliability. The use of .NET for application development and SQL Server for data management supports a smooth and secure user experience by making SecureFinance a strong solution for managing and protecting financial information.

5.1.1 Web Server Accessed by Financial Officer

The financial officer (admin) signs up in the application by entering their name, email ID, mobile number and creating a secure password. After registration the admin logs into the web application by using the credentials. The admin is then directed to an admin panel that includes a dashboard with key options: Manage Customers, Customer Records, View Shared Files and Logout. The "Manage Customers" option allows for registration and management of customer accounts. “Customer Records" gives options for document uploads and management, "View Shared Files" which gives access to files shared by customers and "Logout" ensures a secure session end.

Front end

The front end of the SecureFinance system gives the bank's financial officers a good interface to manage customer accounts and records. Admins can register new customer accounts by entering important information such as the customer's name, email ID, contact details and address. Once a customer is registered the admin has the capability to manage and update their financial records by securing that all details remain current and accurate. Furthermore, admins can upload important financial documents, transaction records and other important files which are associated with each customer's account. These files are protected by hybrid encryption techniques which includes combination ECC and AES for strong data protection. In order to maintain security, the admin has the option to securely log out of the application after completing their tasks. This structured approach ensures that sensitive financial data is been managed securely while at the same time provides the admin with good tools for looking customer information and interactions.

Back end

When admin registers new customer in the system, automatically notification is sent to the customer's email by giving initial login details along with a secure password. For file uploads which includes financial documents or transaction records the system uses hybrid encryption by combining ECC and AES. An encryption key is generated and stored securely so that only authorized users can decrypt and download the files safely. An encrypted key is generated and sent to the customer whose data is uploaded by the admin. This key serves as a secure access mechanism and makes sure that no other unauthorized person can alter the shared data. This strong backend framework supports secure operations while giving good and confidential management of financial records.

5.1.2 Web Server Accessed by Customer

To access the SecureFinance application, the client first logs in using their email ID and password which was initially sent using email after registration. Upon successful authentication the client is directed to a dashboard featuring some options which includes view Records, Share Records and Logout. The "View Records" option allows the client to access and review their financial records. "Share Records" enables the client to share documents with other authorized parties in a secure way. Finally, the "Logout" option secures the client can securely exit the application by maintaining session security and data integrity.

Front end

After registration by the admin, customers receive login credential through email then they can login to the application securely. Clients can access and review their financial records uploaded by the admin with the use of "view record" option. The "Download Files" feature enables clients to securely download financial documents or reports that the bank has uploaded. This download process only allows authorized clients to securely access their financial records. Finally, the "Logout" option ensures that clients can securely end their session after downloading required financial documents. This easy to use front-end design ensures that clients have easy as well as secure access to their financial information while protecting their privacy and data integrity.

Back end

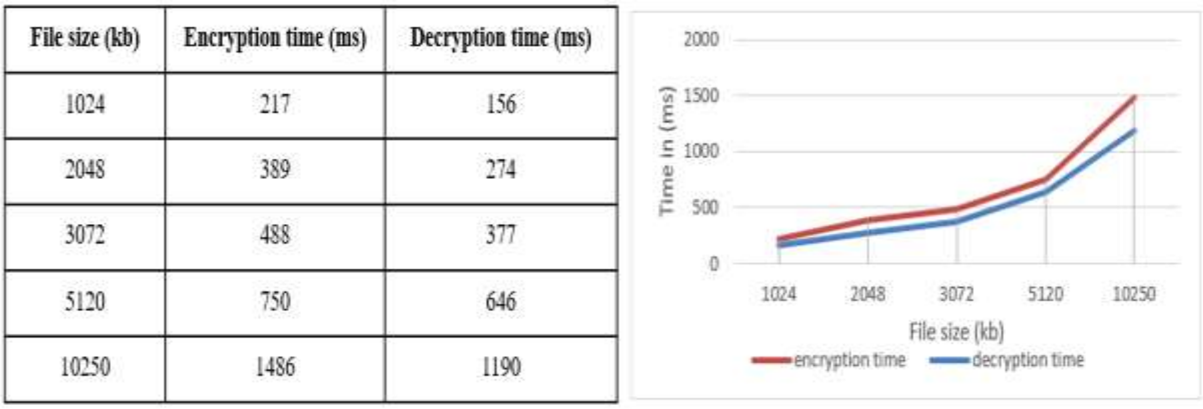
When a client accesses the web application, they are required to enter their details which are verified by secure authentication processes to confirm their identity. For file decryption when clients download their financial documents the system uses hybrid decryption by combining ECC and AES methods. When the client uploads a document the application sends a decryption key to the client's email which enables only the authorized users to access the decrypted data. This strong back-end framework enables secure data management as well as guarantees all client interactions with the web application are protected against unauthorized access and manipulation.

6 Evaluation

This section proves the assessment of the developed ECC + AES hybrid model against ECC+DES Model. The experiments are done by using different file formats (image, video, document) of different file sizes to evaluate the performance of model in terms of encryption time, decryption time, throughput.

6.1 Image Format

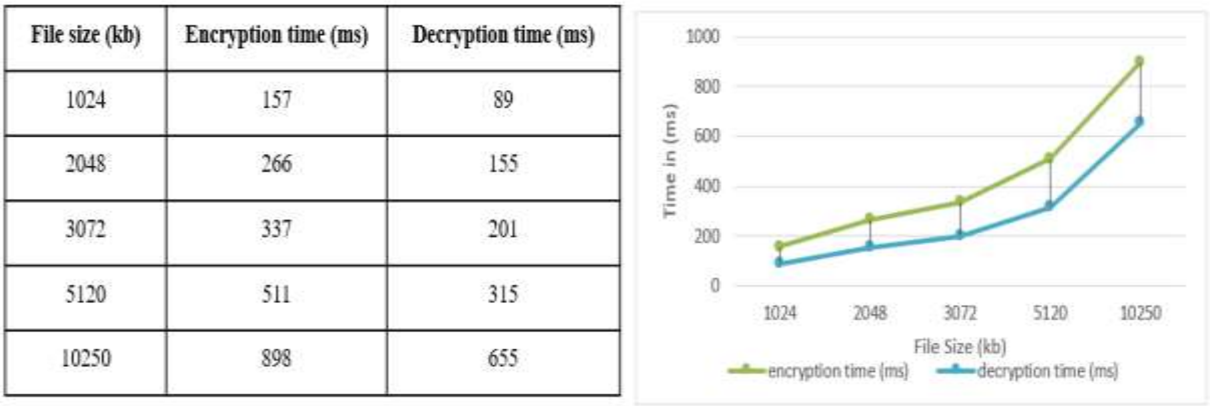
6.1.1 Analysis of ECC+DES with Image format



File size (kb)	Encryption time (ms)	Decryption time (ms)
1024	217	156
2048	389	274
3072	488	377
5120	750	646
10250	1486	1190

Figure 6.1.1: Table and Graph of the encryption and decryption time of ECC+AES of Image

6.1.2 Analysis of ECC+AES with Image format



File size (kb)	Encryption time (ms)	Decryption time (ms)
1024	157	89
2048	266	155
3072	337	201
5120	511	315
10250	898	655

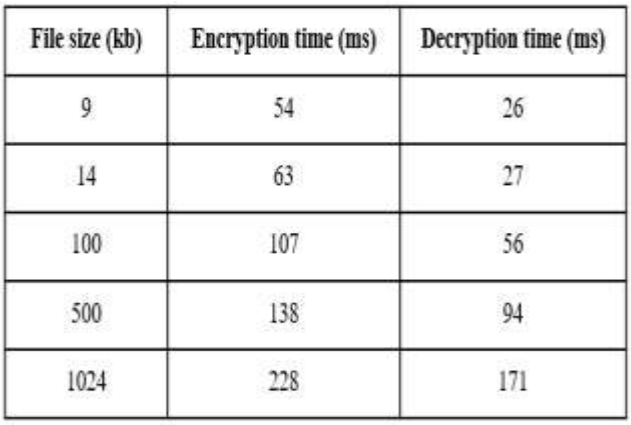
Figure 6.1.2: Table and Graph of the encryption and decryption time of ECC+ AES of Image

The AES+DES and AES+ECC hybrid encryption models perform very differently especially when comparing the encryption and decryption times of image file format. The AES+DES model shows a steady increase in time as file sizes grow. For instance, it takes 217 ms to encrypt a 1024 kb file, and this time escalates to 1486 ms for a 10250 kb file. On the other hand, the AES+ECC model performs more efficiently as encryption time for 1024 kb file it takes 157 ms and increases to 898 ms for a 10250 kb file. This indicates that the AES+ECC hybrid model is comparatively faster in encryption even if the file sizes increase.

Similar pattern is observed when examining decryption times as well. The AES+DES model takes decryption time of 156 ms for a 1024 kb file and 1190 ms for a 10250 kb file. Even though the decryption times are slightly lower than the encryption times of the same model, they still indicate a relatively high computational load. On the other hand, the AES+ECC model takes much lower decryption times when compared to decryption time taken by AES+DES model for images. , It takes 89 ms for a 1024 kb file and increases to 655 ms for a 10250 kb file. Overall, AES+ECC model outperforms AES+DES model in both image file encryption and decryption processes indicating that it is more suitable in processing large volumes of image files.

6.2 Doc Format

6.2.1 Analysis of ECC+DES with Doc format



File size (kb)	Encryption time (ms)	Decryption time (ms)
9	54	26
14	63	27
100	107	56
500	138	94
1024	228	171

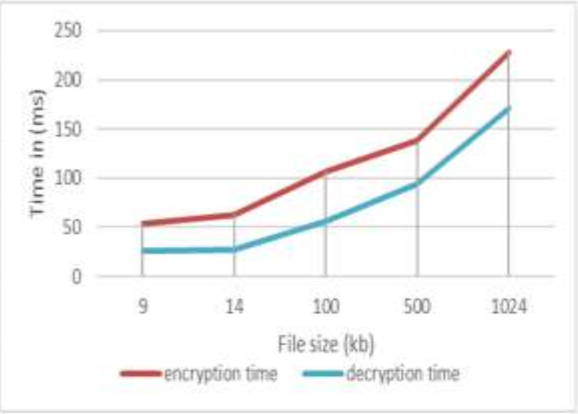
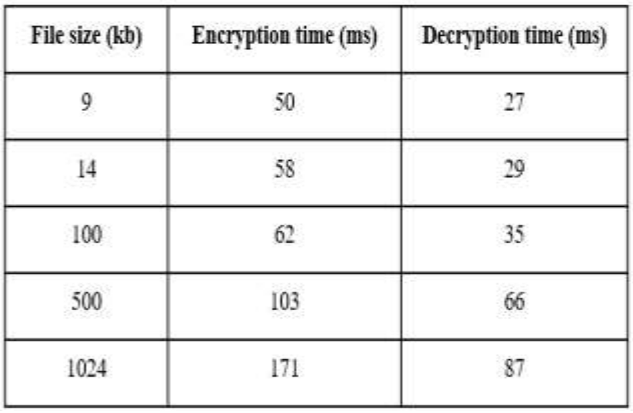


Table 6.2.1: Table and graph of the Encryption and Decryption Time of ECC+DES (Doc)

6.2.2 Analysis of ECC+AES with Doc format



File size (kb)	Encryption time (ms)	Decryption time (ms)
9	50	27
14	58	29
100	62	35
500	103	66
1024	171	87

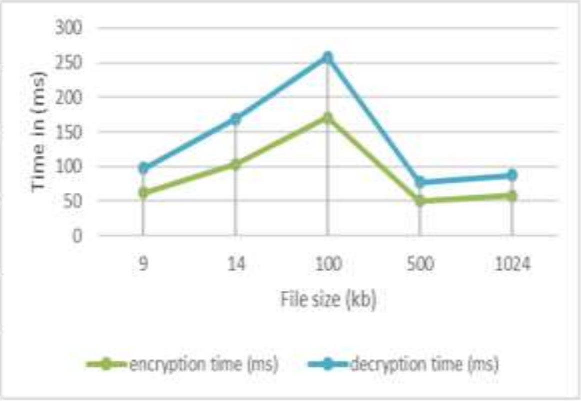


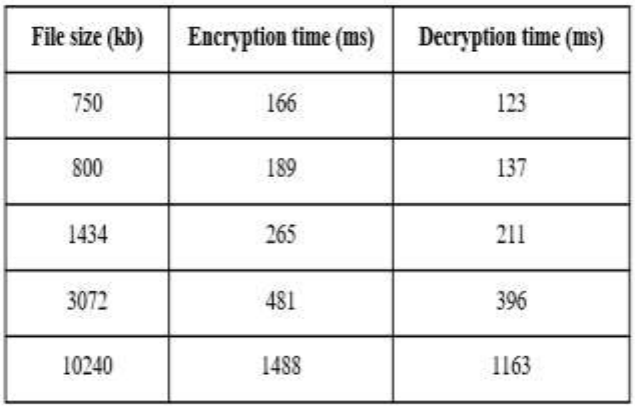
Table 6.2.2: Table and graph of the Encryption and Decryption Time of ECC+AES (Doc)

The further experiment on hybrid models shows that ECC+AES model consistently outperforms the ECC+DES model in terms of doc encryption and decryption speed. For instance, for a document of size of 1024 kb, ECC+AES takes 171 ms to encrypt the data, whereas ECC+DES requires 228 ms. This pattern continues for all documents with different file sizes which shows that ECC+AES demonstrating faster encryption times. This indicates that AES when used with the ECC produces strong encryption without the significant overhead observed when DES combined with ECC algorithm. As a result, ECC+AES offers a more efficient solution for scenarios where quick encryption is crucial, especially when the file size increases.

Moreover, ECC+AES also shows better performance in decryption of document type files. The ECC+AES also has better results in decryption of document type files. When the file size is 9 kb and 14 kb, ECC+AES decryption time for both cases is 27 ms and 29 ms, respectively, which are slightly greater than ECC+DES decryption times of 26 ms and 27 ms, for the same sizes. However, ECC+AES is able to maintain lower decryption times when the size of the file increased. For example, for 1024 kb, ECC+AES takes 87 ms to decrypt the data while ECC+DES takes 171 ms. This result indicates that ECC+AES is not only faster at encrypting document type data but also at decrypting it which makes it more suitable for environments where both operations need to be performed frequently.

6.3 Video Format

6.3.1 Analysis of ECC+DES with Video format



File size (kb)	Encryption time (ms)	Decryption time (ms)
750	166	123
800	189	137
1434	265	211
3072	481	396
10240	1488	1163

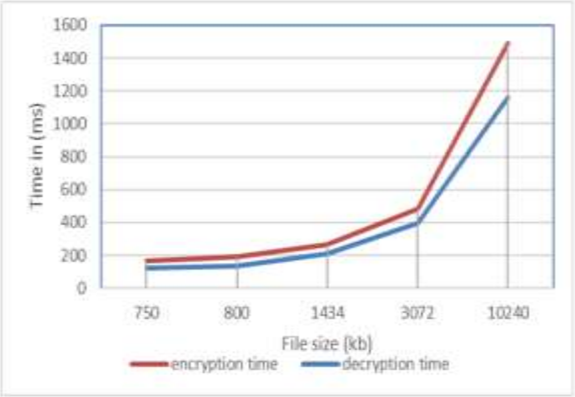
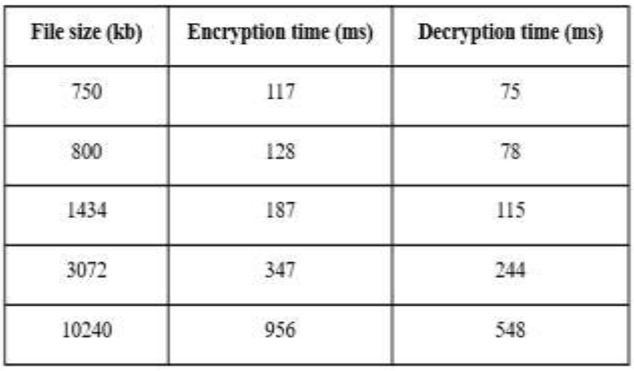


Table 6.3.1: Table and graph of the Encryption and Decryption Time of ECC+DES (Video)



File size (kb)	Encryption time (ms)	Decryption time (ms)
750	117	75
800	128	78
1434	187	115
3072	347	244
10240	956	548

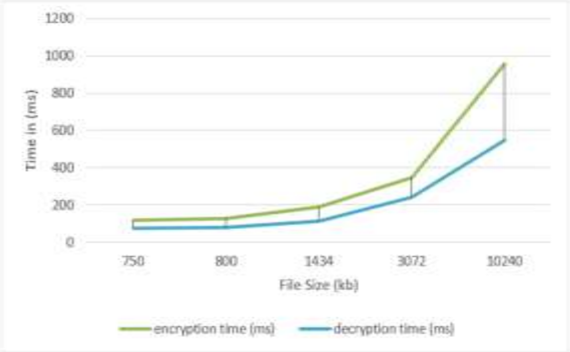


Table 6.3.2: Table and graph of the Encryption and Decryption Time of ECC+AES (Video)

Video files are used in order to further evaluate the performance of the hybrid models, when comparing the Encryption and Decryption results of the both hybrid encryption models applied on video files, it was evident that the ECC+AES encryption model is much more efficient in the encryption and decryption process than the ECC+DES regardless the file size.
Result observed from this experiment shows that the ECC+AES model takes less time for the encryption compared to the ECC+DES model. For example, it only took 956 ms to encrypt the largest video file size of 10240 kb through ECC+AES, while it took 1488 ms to encrypt the same video through ECC+DES. This indicates that AES is very fast and effective when used with ECC. As shown in the above results, ECC+AES performs better in encryption time even if the video file size increase, which makes it a better option for situations where fast encryption is requirement.
The decryption times also add up to the benefits of the ECC+AES model. For instance, it takes only 548 ms for the ECC+AES model to decrypt 10240 kb size video while the ECC+DES model takes 1163 ms, which means that the former is faster at decrypting videos from end-user perspective, especially when the user needs to access the video content immediately. In this case the comparison of the time taken for the ECC+AES and ECC+DES hybrid encryption models when applied in videos it was evident that the ECC AES was efficient and faster in the encryption and decryption processes of videos of different sizes.

6.4 Throughput

$$\text{Throughput} = \frac{\text{Size of Ciphertext}}{\text{Encryption Time}}$$

File Size	Encryption Time	Decryption Time	Throughput Encryption	Throughput Decryption
9	50	27	0.18	0.333
14	58	29	0.241	0.482
100	62	35	1.612	2.857
500	103	66	4.854	7.575
1024	171	87	5.988	11.77

File Size	Encryption Time	Decryption Time	Throughput Encryption	Throughput Decryption
9	54	26	0.166	0.346
14	63	27	0.222	0.518
100	107	56	1.242	1.785

500	138	94	3.623	5.319
1024	228	171	4.491	5.988

The ECC+AES model displays a progressive improvement in its throughput rates as file sizes rise. For instance, when the file size is 9 kb, the throughput is 0 in encryption. 18, while for a 1024 kb file it takes as much as 5.998. This increase also conveys that ECC+AES has the capability to encrypt larger file sizes with better time efficiency. We see that even in the small file sizes, the decryption throughput is higher than the encryption throughput. The algorithm achieves 0.333 decryption throughput when it comes to a 9 kb file size, and it increases to 11. 77 for 1024kb file size. This makes the ECC+AES model more efficient in decrypting large files since the time taken to extract the decrypted data has a direct impact on decision making processes.

The ECC+DES model also demonstrates a relatively good encryption throughput with smaller file sizes; however, it is always outperformed by the ECC+AES model. For example, the throughput of the 9 kb file size is 0.16 and throughput for the 1024 kb file size is 4.491. This means that although ECC+DES is capable of managing larger file sizes, it lacks the ability to do so with the efficiency when compared to ECC+AES.

7 Conclusion and Future Work

7.1 Conclusion

The comparative analysis of ECC+AES and ECC+DES shows that ECC+AES gives good performance in terms of both encryption and decryption speeds. ECC+AES give good results as compared to ECC+DES across different file sizes and types which includes documents, videos and general data. This is mainly for the larger files where ECC+AES consistently shows faster encryption and decryption times by making it more suitable for applications requiring good handling of bulk data. The performance of ECC+AES is attributed to the speed and strength of the AES algorithm in managing larger datasets. The results have showed that for modern applications needing high-performance encryption solutions ECC+AES is the preferred choice due to its better balance of speed and security. A hybrid cryptographic model is a way of protecting data by combining two different types of encryption methods. The idea is to use the strengths of each method to make the data more secure and efficient to handle. The key discovery was that this ECC and AES combination design was more secure and quicker than the other ECC+DES model for financial transaction.

7.2 Future Work

Future research could explore so many limitations to increase the performance and applicability of hybrid cryptographic techniques. One potential direction is optimizing ECC and AES algorithms further to improve their computational performance which is mainly for very large files or high-throughput environments. Investigating the combination of newer cryptographic algorithms that could provide more security and performance benefits can be done in the future. Also exploring hardware acceleration for ECC and AES could reduce encryption and decryption times by making these techniques better for real-time applications.

References

Ahmad, S., Mehfuz, S. and Beg, J. (2022). Hybrid cryptographic approach to enhance the mode of key management system in cloud environment. *The Journal of Supercomputing*, 79(7377-7413.). doi:https://doi.org/10.1007/s11227-022-04964-9.

Anwar, M.R., Apriani, D. and Adianita, I.R. (2021). Hash Algorithm In Verification Of Certificate Data Integrity And Security. *Aptisi Transactions on Technopreneurship (ATT)*, 3(2), pp.65–72. doi:https://doi.org/10.34306/att.v3i2.212.

E. Esai Malar and B. Paramasivan (2021). Enhancing Security and Privacy Preserving of Data in Cloud Using SHA and Genetic Algorithm. *Advances in intelligent systems and computing*, pp.401–411. doi:https://doi.org/10.1007/978-981-16-2543-5_34.

Esther Jyothi, V., Prasad, Dr.B. and Mojjada, D.R.K. (2020). Analysis of Cryptography Encryption for Network Security. *IOP Conference Series: Materials Science and Engineering*, 981(2), p.022028. doi:https://doi.org/10.1088/1757-899x/981/2/022028.

Lai, J.-F. and Heng, S.-H. (2022). Secure File Storage On Cloud Using Hybrid Cryptography. *Journal of Informatics and Web Engineering*, 1(2), pp.1–18. doi:https://doi.org/10.33093/jiwe.2022.1.2.1.

Mhamad Bakro, Sukant Kishoro Bisoy, Ashok Kumar Patel and M. Adib Naal (2022). Hybrid Blockchain-Enabled Security in Cloud Storage Infrastructure Using ECC and AES Algorithms. *Lecture notes on data engineering and communications technologies*, pp.139–170. doi:https://doi.org/10.1007/978-981-16-9260-4_6.

Mohanty, M.D., Das, A., Mohanty, M.N., Altameem, A., Nayak, S.R., Saudagar, A.K.J. and Poonia, R.C. (2022). Design of Smart and Secured Healthcare Service Using Deep Learning with Modified SHA-256 Algorithm. *Healthcare*, 10(7), p.1275. doi:https://doi.org/10.3390/healthcare10071275.

Nadeem, M., Arshad, A., Riaz, S., Zahra, S., Band, S. and Mosavi, A. (2022). Two Layer Symmetric Cryptography Algorithm for Protecting Data from Attacks. *Computers, Materials & Continua*, [online] 74(2), pp.2625–2640. doi:https://doi.org/10.32604/cmc.2023.030899.

Orobosade, A., Aderonke, T., Boniface, A. and J., A. (2020). Cloud Application Security using Hybrid Encryption. *Communications on Applied Electronics*, 7(33), pp.25–31. doi:https://doi.org/10.5120/cae2020652866.

Ranganatha Rao, B. and Sujatha, B. (2023). A hybrid elliptic curve cryptography (HECC) technique for fast encryption of data for public cloud security. *Measurement: Sensors*, [online] 29(p.100870.), p.100870. doi:https://doi.org/10.1016/j.measen.2023.100870.

Rashid, M.N., Abed, L.H. and Awad, W.K. (2022). Financial information security using hybrid encryption technique on multi-cloud architecture. *Bulletin of Electrical Engineering* *and Informatics*, 11(6), pp.3450–3461. doi:https://doi.org/10.11591/eei.v11i6.3967.

Reyad, O., Mansour, H.M., Heshmat, M. and Zanaty, E.A. (2021). *Key-Based Enhancement of Data Encryption Standard For Text Security*. [online] IEEE Xplore. doi:https://doi.org/10.1109/NCCC49330.2021.9428818.

Shruthi Gangadharaiah and Purohit Shrinivasacharya (2024). Secure and efficient public auditing system of user data using hybrid AES-ECC crypto system with Merkle hash tree in blockchain. *Multimedia Tools and Applications*. doi:https://doi.org/10.1007/s11042-024-18363-0.

Smid, M.E. (2021). Development of the Advanced Encryption Standard. *Journal of Research of the National Institute of Standards and Technology*, [online] 126. doi:https://doi.org/10.6028/jres.126.024.

Tahir, M., Sardaraz, M., Mehmood, Z. and Muhammad, S. (2020). CryptoGA: a cryptosystem based on genetic algorithm for cloud data security. *Cluster Computing*, 24(4). doi:https://doi.org/10.1007/s10586-020-03157-4.

Velmurugadass, P., Dhanasekaran, S., Shasi Anand, S. and Vasudevan, V. (2020). Enhancing Blockchain security in cloud computing with IoT environment using ECIES and cryptography hash algorithm. *Materials Today: Proceedings*, 37(7). doi:https://doi.org/10.1016/j.matpr.2020.08.519