UKR Journal of Multidisciplinary Studies (UKRJMS)
Homepage: https://ukrpublisher.com/ukrjms/
Email: submit.ukrpublisher@gmail.com
Volume 1, Issue 4, 2025
ISSN: 3049-432X (Online)

A Stratified Data Security Model Based On Data Priority Using The Advanced Encryption Standard (Aes) And An Enhanced Diffie-Hellman Algorithm

Ofem Ajah Ofem[1], Moses Adah Agana[2*], Obono Iwara Ofem[3], John Adinya Odey[4], David Oboboho Egete[5], Chiamaka Okpalanochikwa[6]

[123456]Department of Computer Science, University of Calabar

***Corresponding Author**: Moses Adah Agana

DOI: *https://doi.org/10.5281/zenodo.17101338*

Article History
Original Research Article
Received: 03-06-2025
Accepted: 20-06-2025
Published: 27-06-2025

Copyright © 2025 The Author(s): This is an open-access article distributed under the terms of the Creative Commons Attribution **4.0 International License (CC BY-NC)** which permits unrestricted use, distribution, and reproduction in any medium for non-commercial use provided the original author and source are credited.

Citation: Ofem Ajah Ofem, Moses Adah Agana, Obono Iwara Ofem, John Adinya Odey, David Oboboho Egete, Chiamaka Okpalanochikwa, (2025), A Stratified Data Security Model Based On Data Priority Using The Advanced Encryption Standard (Aes) And An Enhanced Diffie-Hellman Algorithm, UKR Journal of Multidisciplinary Studies (UKRJMS), volume 1(4), 138-157.

Abstract

The rate of infringements and modifications by cybercriminals to computer documents and data connected to the Internet is immeasurable in our contemporary time. One common form of security breach involves embedding spyware or malicious viruses that secretly monitor user activities and transmit sensitive documents to hackers. Such threats pose serious challenges for computer users and can have devastating effects on individuals, businesses, governments, and other institutions. This study seeks to develop a stratified data security framework that prioritizes data according to its importance, employing the Advanced Encryption Standard (AES) and the Diffie-Hellman algorithm to strengthen existing data protection measures. The system was implemented using Javascript with MySQL at the back end. The testing was done using hypothetical banking transaction details which are classified into three main priority levels: most sensitive, medium sensitive and least sensitive and was test-run on a local server. The results showed that the AES combined with the enhanced Diffie-Hellman can be used to encrypt and protect organizational data based on their importance and can deter unauthorized access. It is recommended that the system be adopted and put into practice to strengthen data integrity and safeguard against unauthorized access to personal files as well as networked computer data across different levels of priority.

Keywords: *stratified; security; priority; encryption; sensitive*

1. Introduction

Because computers connected to the Internet are vulnerable to various infringements and alterations such as the implantation of spyware and viruses that track and transmit user information to hackers—users encounter significant challenges in safeguarding their data. These threats can have severe consequences for individuals, businesses, governments, and other institutions. The major challenges include:

i. Unauthorized intrusion into personal files and data
ii. Inadequate protection measures for sensitive information
iii. Reliance on unverified software for securing valuable data
iv. Failure of existing security systems to provide a unified link for file transfers between two parties
v. Weak security and encryption standards found in files created by most applications.
vi. Session hijacking.
vii. Data and identity theft (Adrian et al., 2017).

Most existing data security techniques such as passwords, firewalls, intrusion detection systems, etc. are easy to maintain, understand, design and are cheap to afford, but one major problem with them is that they can be easily compromised. For instance, passwords can be guessed, user IDs can be stolen, sessions can be hijacked, etc. In addition, intrusion detection systems can only detect intrusion, they cannot prevent intrusion. More so, in such techniques, it is difficult to differentiate authorized users from unauthorized ones.

As reported by Rich (2015) there are even some limitations in the use of biometrics in data security. For instance, people are scared of tampering with their eyeballs, finger prints can be lifted or snatched by a hacker, and this cannot be changed once it has been compromised. In order to make security of systems more effective, it is suggested that biometrics should be combined with other techniques such as passwords, firewalls, encryption, etc.

Given the expansive nature of data communication networks, regulating access points can be challenging, as the coverage of such networks often extends beyond the physical boundaries of an organization (Geir, 2020). In light of the limitations of existing data security measures, the design and implementation of a stratified encryption-based security model that prioritizes data levels represent a significant advancement toward strengthening data protection.

These problems necessitate the design of a stratified data security model to protect data based on the priorities attached to such data. Classifying the data in their order of importance can guide the security protocol to assign to each data stratum.

A strategic approach towards securing data both in stand-alone systems and in networks is expedient due the importance attached to data at all times as well as the ever increasing threats to such data by unscrupulous persons (Kire, 2016). The traditional method adopted by most computer users to secure documents is to lock them with passwords in most applications. This approach is susceptible to compromise by criminals and is therefore almost akin to no protection.

If confidential details regarding a company's customers, financial records, or new product lines are exposed to competitors, the resulting security breach could cause loss of business, legal actions, or even bankruptcy. Safeguarding sensitive information is therefore not only a core business necessity but also, in many instances, a legal and ethical obligation (Hutchinson & Sawyer, 2000). On a personal level, information security plays a crucial role in protecting privacy, though the perception of privacy varies widely across different cultural contexts.

One fundamental method of safeguarding data is encryption. This technique protects information from unauthorized users by converting it into an unreadable form. However, in today's digital environment, cybercriminals possess advanced skills and sophisticated tools capable of intercepting and decrypting files containing sensitive information. This underscores the need for more robust and logically intensive approaches to ensure that data remains secure throughout its life cycle. Strengthening encryption mechanisms not only enhances the confidence of data originators and recipients but also provides assurance that transmitted information is accessible solely to authorized parties (Jeevitha et al., 2016).

Encryption has long been adopted by corporate organs such as the military and various governments to facilitate secret communication. It is often used in safeguarding information from unauthorized access, especially when transmitted remotely via networks such as the Internet, mobile phones, Bluetooth devices and bank automatic teller machines (ATMs). Encryption can also be used to protect data "at rest", such as information stored on local disks and secondary storage media of computers (Ahmad et al, 2017).

The valuable nature of data, vis-a-vis the prevalent security threats to stored data necessitates didactic approaches to secure them. One of such approaches adopted in this study is classification of data based on priorities as to determine the level of protection for each stratum of data in the order of importance attached to each stratum.

Access control is crucial in data protection (Tiller, 2009). The complex and dynamic nature of information security requires that not only one approach should be relied on. A combination of access control strategies for instance, and data encryption can offer a better secured system for data. In this scenario, each level of data should define access privileges and should be encrypted accordingly based on the priority attached to it (The UK Data Protection Act (DPA); 1998; Baker and Mckenzie, 2017). As noted by Tiller (2009), the access control strategy adopted by an organization is directly influenced by its overall approach and philosophy regarding information security.

Encryption entails converting the original intelligent message (the plain text) into an algorithm and a key independent of the plain text. The key is shared by both the sender and the receiver to decipher the message (Stallings, 1999; Henry & Pasley, 2009). A change made to the key changes the output to the algorithm because the algorithm basically produces a different output depending on the specific key being used.

In this study, a stratified data security model based on data priorities using the Advanced Encryption Standard (AES) and the Diffie-Hellman key agreement has been proposed. The AES is a specification for the encryption of electronic data established by the US National Institute of Standards and Technology (NIST) to complement existing access control strategies. In the Diifie-Hellman key agreement, two parties A and B will believe that they have agreed on a common key, but in fact they have both actually agreed on different keys with a third party C.

The objectives of the study are to:

i. Classify individual and organizational data based on their level of importance i.e. the most important being tagged as having the highest priority;
ii. Develop a stratified security protocol for sensitive and valuable data files using the Advanced

Encryption Standard (AES) and the Diffie-Hellman key agreement as a one-time pad (the key can only be used once; a new key is generated each time);

iii. Implement the design using hypothetical banking data with existing access control strategies such as password, network intrusion detection systems and monitoring systems to enhance the integrity of files transmitted with each level of user having a different key based on the data priority.

2. The Rationale for Data Stratification

Data stratification entails the division of the members of a population of data into subgroups to make it easier to identify patterns, relationships and differences amongst them. This can as well help data managers to dynamically manage populations, prioritize the components with selected subgroups and track their progress over time based on key metrics (Chaudhuri, et al., 2007).

Stratification implies dividing a population or inference space up into sub-groups or subunits before sampling. With stratification, it is easier to identify the importance of priority attached to the data and the variability within a stratum is minimized. Therefore, to be useful for sampling, data strata are defined such that similar sampling units are grouped into the same strata and based on the importance or priority attached to them. Since there is minimal variability within a stratum, there is improved precision and efficiency of estimates than in simple random selection (Kim et al., 2013; Mina, Beiruti & Abadi, 2015).

With stratification, data security teams can thus gain more visibility over the security of facts, and can offer a more dependable clarity to their clients. Data security prioritization is akin to risk assessment, and can be either qualitative (involving non numeric levels or cadres such as low, medium and high) or quantitative (measured in terms of monetary or time losses) (Ilia, 2018; Ogbu & Agana, 2019).

3. Overview of Data Security using Encryption and Data Prioritization

3.1 Data Encryption

In cryptography, encryption is the process of transforming information or messages into a form that can only be understood by authorized recipients. While encryption does not prevent interception, it ensures that the content of the message remains indecipherable to unauthorized parties. The original message is known as plaintext, which is converted into ciphertext through an encryption algorithm. This ciphertext can only be interpreted when decrypted using a key provided to the legitimate recipient (Aaron, 2019; Nitin et al., 2013; Faiqa et al., 2017; Jyotirmoy, 2014).

A cryptographic protocol or encryption protocol specifies or describes how encryption and decryption algorithms should be used. This usually provides details about the data structures and representations used for encryption and decryption (Aaron, 2018).

The essential elements of a conventional encryption scheme are illustrated in Figure 1.

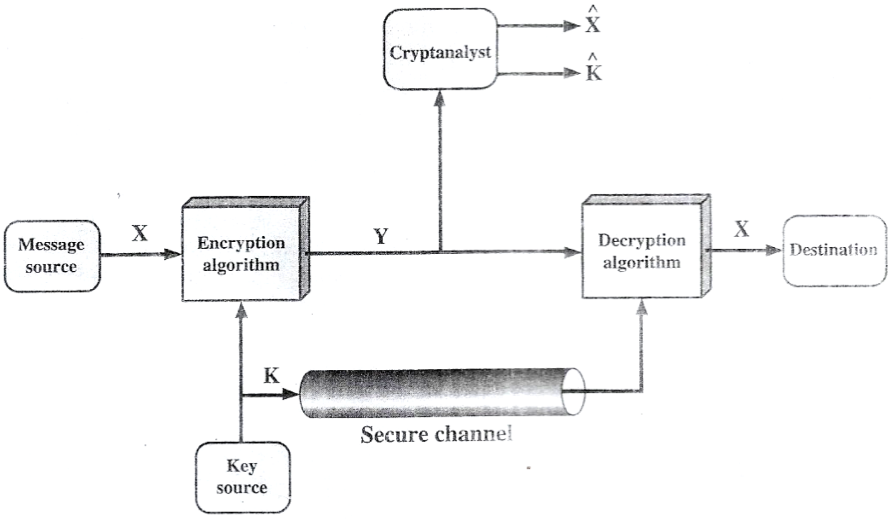


Figure 1: Model of a Conventional Cryptosystem (Stallings, 1999)

According to Salomon (2010), a cryptographic protocol typically includes the following components:

i. Key (K) agreement or establishment, the plaintext (X), and the encrypted message (Y)
ii. Entity authentication
iii. Symmetric encryption and construction of message authentication material
iv. Secure application-level data transport
v. Non-repudiation mechanisms
vi. Secret sharing techniques
vii. Secure multi-party computation

Cryptographic systems arc generically classified along three independent dimensions: namely, the type of operations used for transforming plaintext to cipher text, the number of keys used and the way in which the plaintext is processed (Stallings, 1999; Kessier, 2019).

3.2. The Advanced Encryption Standard (AES) Cipher

The most widely used symmetric encryption algorithm today is the Advanced Encryption Standard (AES), which is reported to be at least six times faster than Triple DES (Conrad et al., 2019). Originally known as *Rijndael*, AES was developed by Belgian cryptographers Joan Daemen and Vincent Rijmen, whose proposal was selected by the U.S. National Institute of Standards and Technology (NIST) in 2001. Rijndael is a family of ciphers that supports various block and key sizes (Stallings, 1999).

For AES, NIST adopted three versions of the Rijndael cipher, each with a fixed block size of 128 bits but with key lengths of 128, 192, and 256 bits. This led to the standardized variants: AES-128, AES-192, and AES-256. AES is a symmetric-key algorithm, meaning the same key is applied in both encryption and decryption processes (Kessler, 2019). Since its adoption, it has replaced the Data Encryption Standard (DES) of 1977 and has become the global standard for securing electronic data.

Like DES, AES is a symmetric block cipher, but it offers much greater flexibility. While DES restricts its block and key sizes to 64 and 56 bits respectively, Rijndael allows independent selection of block and key sizes from 128, 160, 192, 224, or 256 bits. However, the AES standard limits the block size to 128 bits with only three permissible key lengths (128, 192, or 256 bits). Depending on the chosen key size, the standard is identified as AES-128, AES-192, or AES-256 (Stallings, 1999; Xu et al., 2018).

Figure 2 illustrates the AES byte representation.

$$\begin{bmatrix} b_0 & b_4 & b_8 & b_{12} \\ b_1 & b_5 & b_9 & b_{13} \\ b_2 & b_6 & b_{10} & b_{14} \\ b_3 & b_7 & b_{11} & b_{15} \end{bmatrix}$$

Figure 2: The AES byte representation (Stallings, 1999)

An important feature of AES is that all computations are performed on bytes rather than bits. Consequently, a 128-bit plaintext block is treated as 16 bytes, which are organized into a 4 × 4 matrix for processing, as shown in Figure 2.

Several AES parameters vary with the key length. For instance, when a 128-bit key is used, the algorithm executes 10 rounds of processing, while 192-bit and 256-bit keys require 12 and 14 rounds, respectively. Among these, the 128-bit key remains the most commonly implemented in practice (Stallings, 1999; Conrad et al., 2019; Harshali & Ashok, 2018).

According to Stallings (1999), AES was designed to have the following characteristics:

i. Resistance against all known attacks.
ii. Speed and code compactness on a wide range of platforms.
iii. Design Simplicity.

AES is designed as an **iterative algorithm** that follows the principles of a **substitution permutation network (SPN)**. In this structure, the encryption process consists of a sequence of interconnected operations, where some steps substitute input values with predefined outputs (substitutions), while others rearrange or transform data positions (permutations) (Harshali & Ashok, 2018).

A source produces a message in plaintext:

$$X = [X_1, X_2,........, X_m] \text{ --- } 1$$

The elements of XXX are drawn from a finite alphabet. Traditionally, this alphabet comprised the 26 uppercase letters of the English language. In modern cryptography, however, the **binary alphabet** {0,1}\{0,1\}{0,1} is predominantly used. For encryption, a key of the form K=[K1,K2,…,Km]K = [K_1, K_2, \ldots, K_m]K=[K1 ,K2,…,Km] is generated (Stallings, 1999). If the key is produced at the message source, it must also be securely transmitted to the destination through a protected channel. Alternatively, a trusted third party may generate the key and deliver it securely to both the sender and the receiver.

With the message X and the encryption key K as input, the encryption algorithm forms the ciphertext $Y = [Y_1, Y_2 .... Y_N]$. We can write this as

$$Y = E_K(X) \text{ -- } 2$$

This notation indicates that Y is produced by using encryption algorithm E as a function of the plaintext X, with the specific function determined by the value of the key K.

The intended receiver, in possession of the key, is able to invert the transformation:

$$X = E_K(Y) \text{ --- } 3$$

The schematic diagram of AES structure is illustrated in Figure 3

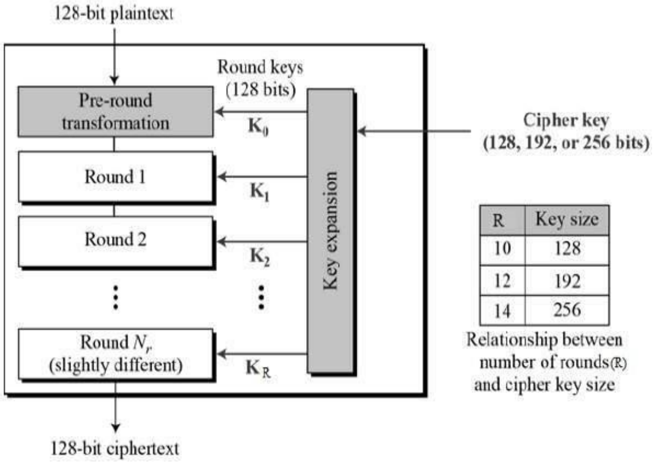


Figure 3: The Schematic Diagram of AES (Harshali and Ashok, 2018)

3.3. The Diffie-Hellman Algorithm

The Diffie-Hellman is believed to have been introduced by Ralph Markel and named after Whitfield Diffie and Martin Hellman as one of the first public-key procedures for exchanging cryptographic keys securely. In this scheme, the sender and receiver make a common secret key in Diffie-Hellman algorithm and then they start communicating with each other over the public channel which is known to everyone (Aryan & Durai, 2017). Most Internet services are secured by Diffie –Hellman.

The scheme is susceptible to attacks such as man-in-the-middle attack, denial of service because the attacker exists in the public channel where he can receive the public key of both sender and receiver and send same which he generates to the sender and receiver. Figure 4 illustrates how this is achieved when Ram and Sita are communicating while Ravan is the man-in-the-middle attacker.

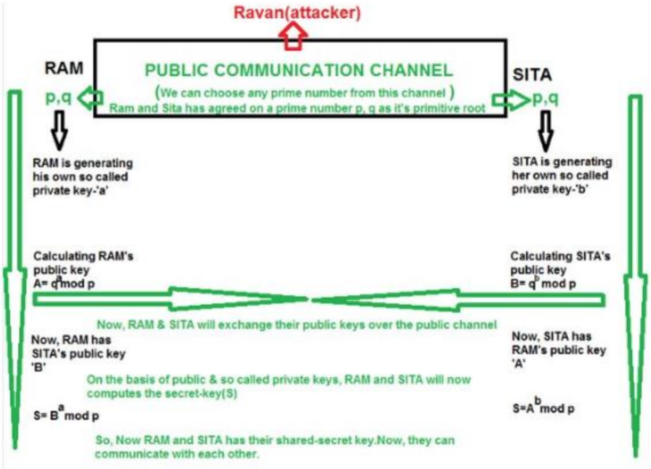

Ravan(attacker)
RAM
PUBLIC COMMUNICATION CHANNEL
(We can choose any prime number from this channel)
Ram and Sita has agreed on a prime number p, q as it's primitive root
SITA
p,q
p,q
RAM is generating his own so called private key-'a'
SITA is generating her own so called private key-'b'
Calculating RAM's public key $A= q^a \text{mod } p$
Calculating SITA's public key $B= q^b \text{ mod } p$
Now, RAM & SITA will exchange their public keys over the public channel
Now, RAM has SITA's public key 'B'
Now, SITA has RAM's public key 'A'
On the basis of public & so called private keys, RAM and SITA will now computes the secret-key(S)
$S= B^a \text{mod } p$
$S=A^b \text{mod } p$
So, Now RAM and SITA has their shared-secret key.Now, they can communicate with each other.

Figure 4: The Diffie-Hellman key generation showing man-in-the-middle attack (Aryan & Durai, 2017)

A similar scenario is illustrated in Figure 5 where Alice and Bob are communicating, while Fred is the man-in-the-middle attacker.

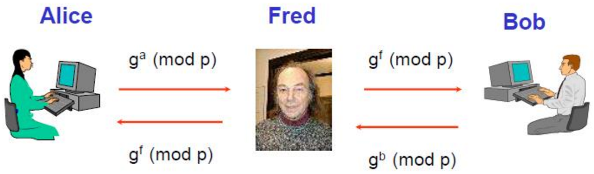


Figure 5: The Diffie-Hellman key generation showing man-in-the-middle attack (Keith, 2006)

3.4. Related Works

Steve (2019) proposed a secure file transfer solution by implementing the Secure File Transfer Protocol (FTP) with Secure Socket Layer (SSL) in a healthcare organization. He emphasized that secure electronic file transfer has become indispensable for business transactions and communication across organizations. In his study, security in FTP was enhanced by integrating the SSL/TLS protocol. The solution adopted was highly customizable and scalable, employing FTP over SSL alongside additional file encryption features to meet the healthcare organization's project-specific requirements.

Aryan and Durai (2017) introduced an enhanced Diffie Hellman algorithm for reliable key exchange. Their approach was aimed at reducing the risks of man-in-the-middle and session hijacking attacks commonly associated with the traditional Diffie–Hellman protocol. The enhancement involved generating a stronger secondary secret key, derived from the primitive root of the first key, which was exchanged between the communicating parties. This ensured that for each transmitted message, a unique shared secret key was established.

In a similar direction, Rasha, Rasha, and Zinah (2020) proposed an improved Diffie Hellman protocol using video entropy. Their method enabled key transfer through public or non-secure channels by leveraging video files as carriers. Keys were extracted based on the entropy values of video frames, providing resistance against man-in-the-middle and discrete logarithm attacks. The system added complexity to key guessing and showed applicability in enhancing the security of communication platforms such as WhatsApp and Viber.

Ahmadi (2011) examined the application of the Advanced Encryption Standard (AES) in different modes. Specifically, he utilized AES with an Initialization Vector (IV) as an IPsec Encapsulating Security Payload (ESP) mechanism, offering confidentiality, origin authentication, and connectionless integrity in secure communications.

Similarly, Harshali and Ashok (2018) proposed an efficient FPGA-based AES implementation with enhanced security features. Recognizing the vulnerability of electronically transmitted data, they introduced a hybrid non-pipelined AES algorithm built upon the traditional AES but with improvements in S-box generation and key initialization. By employing a PN Sequence Generator, they developed a modified S-box and an improved key generation process, which demonstrated notable improvements in encryption strength compared to the standard AES.

Padmaa and Venkataramani (2014) explored a novel approach by combining encryption with image steganography to improve both imperceptibility and capacity. Their method, implemented in the spatial domain, involved embedding a secret message within a carrier color image through dynamic encryption-based steganography. Only authenticated recipients possessing the correct keys could extract the hidden information. However, the limitation of this approach lies in its dependency on image files; if the carrier image is lost or corrupted, the encrypted message becomes irretrievable.

4. Methodology

4.1 Data Sourcing and analysis

The data for the research was primarily collected from commercial banks where customer data and transactions have varying priorities. Some commercial banks were visited and questionnaires were distributed to the bankers and customers to elicit their opinions as well as have a practical experience on how they operate and secure data on their networks. Data was equally gotten from secondary sources (such as available banking records) and from interviews. For instance, some users were interviewed to get information on how they secure their personal data network and the challenges involved. Other secondary sources included relevant journals, books and the World Wide Web.

There are existing data security techniques such as passwords, firewalls, intrusion detection systems, etc. These techniques are easy to maintain, understand, design and are cheap to afford. However, one major problem with these techniques is that the security systems can be easily compromised. For instance, passwords can be guessed, user IDs can be stolen, sessions can be hijacked, and so on. In addition, intrusion detection systems can only detect intrusion, they cannot prevent intrusion. More so, in such techniques, it is difficult to differentiate authorized users from unauthorized ones. The traditional Diffie-Hellman algorithm is also known to be susceptible to some forms of attacks such as man-in-the-middle attack, denial of service and session hijacking.

With such problems with the existing data security techniques, developing and implementing a data security model based on data priority using stratified encryption using the AES and the enhanced Diffie-Hellman algorithms is definitely a step in the right direction.

The proposed model is aimed at optimizing the security of individual and organization data in networks. Stratified encryption is not new in the broad area of data security. However, restrictions of data access based on the priority of the data and users have not been greatly emphasized, so the direct application of the system will be a panacea for enhanced data protection, especially in networks.

The system comprises of various components such as; users, data access level, server and management reporting tools to achieve data security goals. The data security model focuses on the user and data access level, breaking them into strata according to their priorities or levels of importance, and varying the degree of encryption and access privileges based on such priorities per stratum. This should work in real-time and defend the data network or system against majority of intrusions either already known or new kinds of attacks.

The system administrator will define the sensitivity/priority of data types as to determine the encryption level to be assigned to each. Three sensitivity levels defined in the system are as follows:

i. Most sensitive
ii. Medium sensitive
iii. Least sensitive

This was achieved by classifying the data according the importance attached to each category and providing various levels of encryption and decryption keys to each stratum. It is expected that the system will significantly reduce or mitigate hacking in data networks and improve the security level of varying degrees of data.

The inputs to the system were modeled from commercial banking transactions. The input data used include:

i. Account number
ii. Amount deposited/withdrawn
iii. ATM card number
iv. ATM PIN
v. Account name
vi. Bank Verification number (BVN)
vii. Account type, and
viii. Check number

Table I shows the input model, classifying the data according to their priorities.

Table I: The input Model showing Classified Data based on Priorities

DATA PRIORITY	BANK DATA	DATA RECIPIENTS
Most sensitive (internal) LEVEL 1	i. ATM pin, ii. 5-digits confirmation number and iii. BVN	Internal (e.g. management, bank customers, staff)
Medium sensitive (internal)	i. Account name ii. Account number iii. Amount deposited iv. Amount withdrawn v. Loan granted vi. Statements of Accounts vii. Loan disbursement list viii. Annual or quarterly profits	Internal (e.g. management, bank customers, staff)
Least sensitive (external)	i. Adverts ii. Promos	The public (external) and internal recipients too.

The ***data priority*** option of the input allows the administrator to create private keys for the different levels of data security; majorly the most/medium sensitive strata and the least sensitive. The key validation number is provided to enable verification before the private key can be seen by the administrator. To view the private key for each level, the administrator is requested to select the data priority level and provide the key validation code before it can be seen by him or other users.

4.2 Process Design

The process physical model describes the datasets classified based on the priorities attached to them. There is a sensitivity tagging algorithm that encrypts the data and generates decryption keys using the AES at one instance, and the Diffie-Hellman scheme at another instance, based on the data priorities. The data with high priority are assigned a private key while the data with low priority are assigned a public key. The physical model is illustrated in Figure 6.

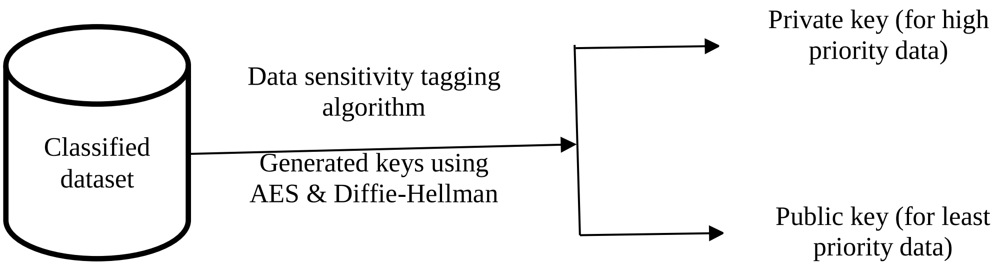

Classified dataset
Data sensitivity tagging algorithm
Generated keys using AES & Diffie-Hellman
Private key (for high priority data)
Public key (for least priority data)

Figure 6: Data Classification Physical Model

The encryption was based on the Advanced Encryption Standard (AES) and the modified Diffie-Hellman algorithm.

The AES Algorithm

The Advanced Encryption Standard (AES) was selected for the system design due to its suitability in scenarios where the encryption key changes across different strata. AES is available in three variants, determined by key sizes of 128, 192, and 256 bits. This study focuses on the 128-bit key schedule, which offers a solid foundation for understanding the extended 192-bit and 256-bit versions (Ahmer et al., 2018). A brief note on how these variants differ from the 128-bit version is provided later in the work.

A typical AES encryption round consists of four sub-processes. As illustrated in Figure 7, the first stage begins with a byte substitution process, where the 16 input bytes are replaced by values from a predefined lookup table, known as the Substitution box (S-box). The output is structured into a 4 × 4 matrix. Following substitution, each of the four rows undergoes a leftward shift operation, in which bytes that move beyond the boundary are reinserted on the right-hand side of the same row.

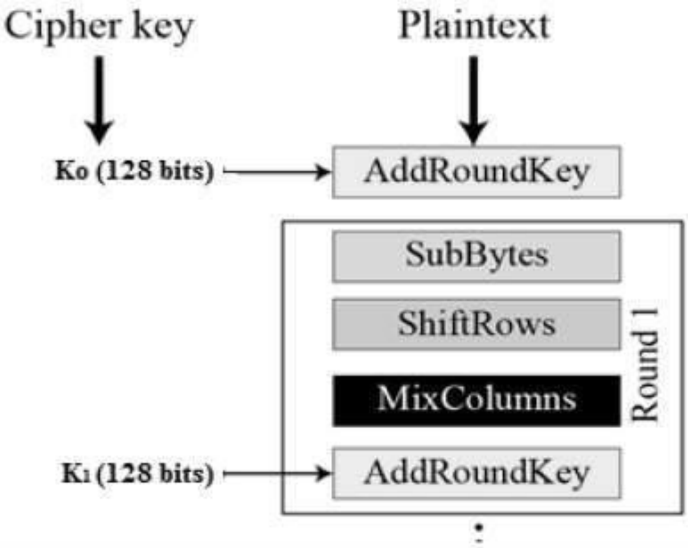


Figure 7: The First Round of the AES Encryption Process

In the next stage of AES, each column of four bytes undergoes transformation through a special mathematical function. This operation takes the four input bytes of a column and produces four new bytes, which replace the original values. The outcome is a new 4 × 4 matrix consisting of 16 updated bytes. It is important to note that this transformation, known as Mix Columns, is omitted in the final round of AES.

Following this, the 16 bytes of the matrix are reinterpreted as 128 bits, which are then combined with the 128-bit round key using the XOR (exclusive OR) operation. If this step occurs in the final round, the output becomes the ciphertext. Otherwise, the resulting 128 bits are reorganized back into 16 bytes, and the process proceeds to the next round.

The total number of AES rounds depends on the key size: 10 rounds for 128-bit keys, 12 rounds for 192-bit keys, and 14 rounds for 256-bit keys. Each round requires a unique 128-bit round key, which is generated from the original AES key through the AES Key Schedule. This schedule derives 10, 12, or 14 round keys, depending on the key length, to serve as inputs for the Add Round Key operations in successive rounds. The process is illustrated in Figure 8.

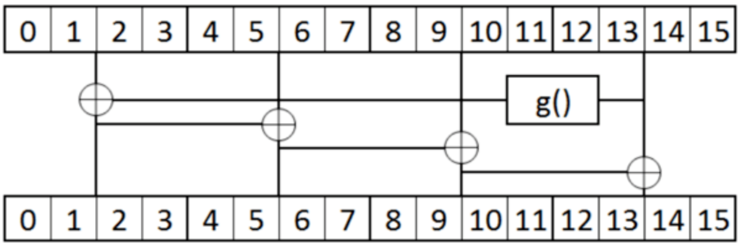


Figure 8: The round key transformation of AES-128

Each word (32 bytes) of the previous round key is exclusive-ored with some value to produce the corresponding word of the current round key. In the case of words 1-3, the value used in the exclusive-or is the previous word (words 0-2) of the previous round key as illustrated in Figure 9.

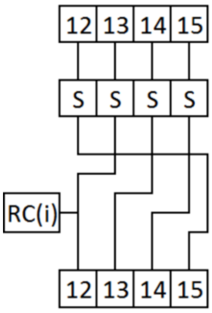


Figure 9: The g function of the AES key schedule

As illustrated in Figure 9, the operation involves three stages: an S-Box transformation, a permutation, and an exclusive-OR (XOR) operation. The output of the key schedule function serves as the round key input to the Add Round Key operation in AES encryption. The same transformation applied to the round key is repeated to generate the key for the next round.

The Diffie–Hellman Algorithm

In its traditional form, the Diffie–Hellman algorithm establishes a shared secret for secure communication over a public network. The protocol leverages elliptic curves to generate points, which are then used along with certain parameters to derive the secret key.

For a simple empirical example, consider four variables: a prime number PPP, a primitive root GGG of PPP, and two private values aaa and bbb.

i. Both PPP and GGG are publicly available.
ii. Two users (e.g., Alice and Bob) select their private values aaa and bbb.
iii. Using these, they each generate a public key which is exchanged openly.
iv. Upon receiving the other’s public key, each user computes the shared secret key.

This process ensures that both parties arrive at the same secret key, which can then be used for encryption. The procedure is summarized in Table II.

Table II: Illustration of Secret key generation and exchange using the Diffie-Hellman Algorithm

ALICE	BOB
Public Keys available = P, G	Public Keys available = P, G
Private Key Selected = a	Private Key Selected = b
Key generated = x = G^amodP	Key generated = y= G^bmodP
Exchange of generated keys takes place	
Key received = y	key received = x
Generated Secret Key = k_a=y^amodP	Generated Secret Key = k_b=x^bmodP
Algebraically, it can be shown that k_a=k_b	
Both users now have a symmetric secret key to encrypt	

Due to the susceptibility of the traditional Diffie-Hellman algorithm to some known attacks, the modified algorithm to subvert such attacks was developed in this study as shown in the algorithm that follows

The Modified Diffie-Hellman Algorithm:

1. Start
2. // Sender = X, Receiver = Y
3. p = prime number agreed on by X and Y
4. q = primitive root of p.
5. X and Y choose their so called private key ‘a’ and ‘b’ known to only two of them
6. X’s public key A=q^a mod p.
7. Y’s public key B=q^b mod p.
8. X and Y exchange their public key: such that X now has B and Y now has A.
9. X computes B^a mod p = q^{ba}mod p= S.
10. Y computes A^b mod p= q^{ba} mod p =S.
11. X and Y each get ‘S’ as their shared secret key.
12. X and Y each take ‘e’ as the primitive root of ‘S’.
13. X and Y generate their own so called private key ‘f’ and ‘g’ known to them only
14. X’s second public key C= e^f mod S.
15. Y’s second public key D= e^g mod S

16. X and Y exchange their second public key: such that X now has D and Y now has C.
17. X calculates D^f mod S= e^{gf} mod S= W.
18. Y calculates C^g mod S= e^{fg}mod S= W.
19. Both X and Y get 'W' as their second shared-secret key.
20. X and Y select their random number 'h' and 'i' respectively.
21. X calculates: X= (W*h) and Y calculates: Y= (W*i)
22. X and Y exchange X and Y finally
23. End.

Key exchange is a fundamental application of public key cryptography. In the system design, an asymmetric scheme was also incorporated to achieve non-repudiation and user authentication. Figure 10 illustrates the integration of three cryptographic techniques into a hybrid cryptographic model for secure communication.

The model demonstrates how the hybrid scheme combines multiple functions to enable secure transmission through the use of a digital signature and a digital envelope. The digital envelope consists of two components: an encrypted message and an encrypted session key. The process operates as follows:

i. The sender generates a random session key and uses it with a symmetric encryption algorithm to encrypt the message.
ii. This session key is then encrypted using the receiver's public key.
iii. The encrypted session key and encrypted message together form the digital envelope.
iv. Upon receipt, the receiver uses their private key to recover the session key and subsequently decrypts the message.

This scheme ensures non-repudiation because the digital signature verifies the sender's identity. If the receiver computes a hash value with the sender's public key and confirms that the message has not been altered, only the sender could have produced that signature. At the same time, the receiver's ability to correctly decrypt the session key confirms that they are the intended recipient.

Additionally, the scheme employs Perfect Forward Secrecy (PFS), meaning that each session uses a unique session key. Even if a session key is compromised, only the communication from that session is exposed, while future sessions remain secure since their keys are independently generated. This process is depicted in Figure 10.

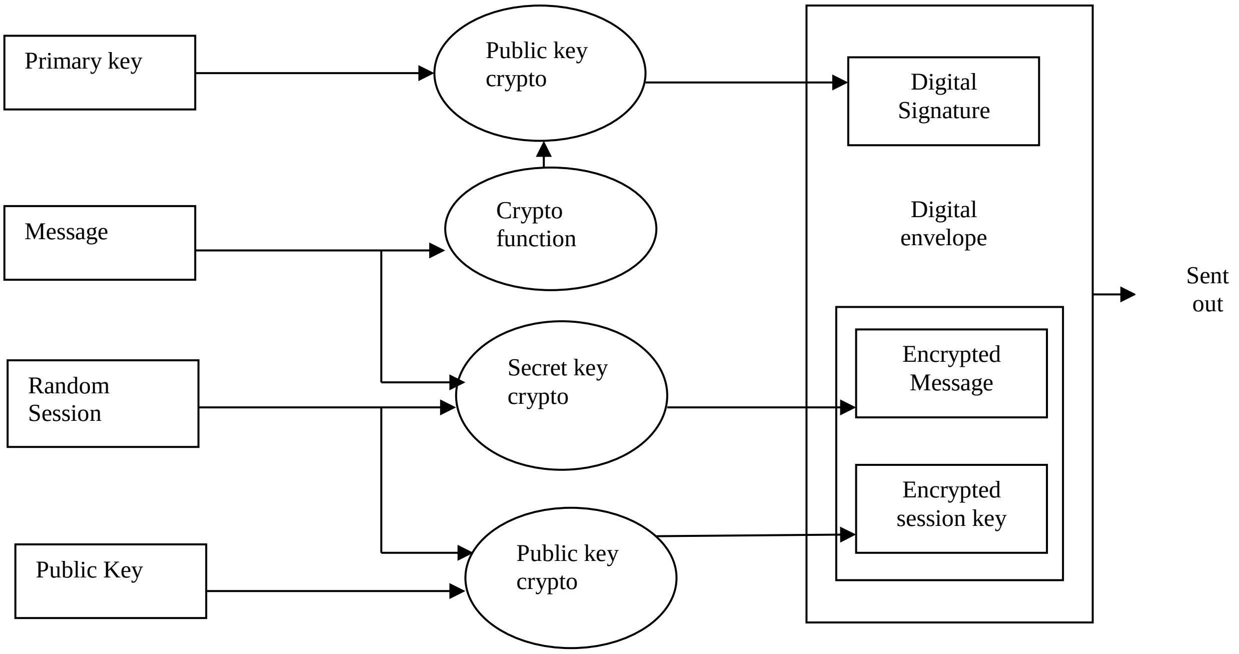


Figure 10: Use of Three Cryptographic Techniques for Secure Communication

The System Flowchart

The system flowchart illustrates the data classification based on priority and the subsequent encryption to protect the data. This is illustrated in figure 11.

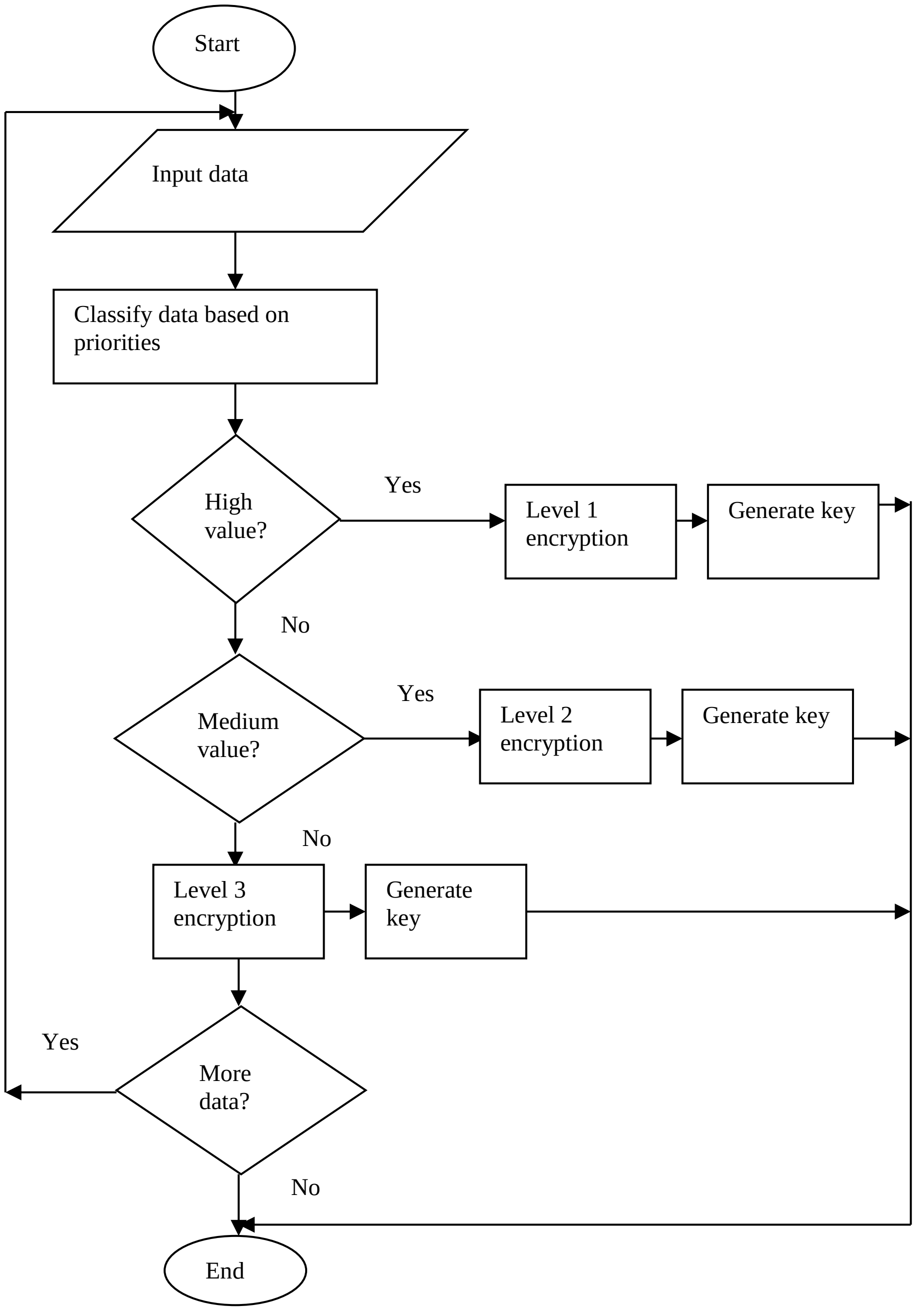


Figure 11: System Flowchart

The Process Activity Model

The process activity model (depicted in the activity diagram shown in Figure 12) illustrates the activities of gathering the data to be classified, the process of encrypting them, generating the decryption keys, and the end of the activity when the data is secured from unauthorized access.

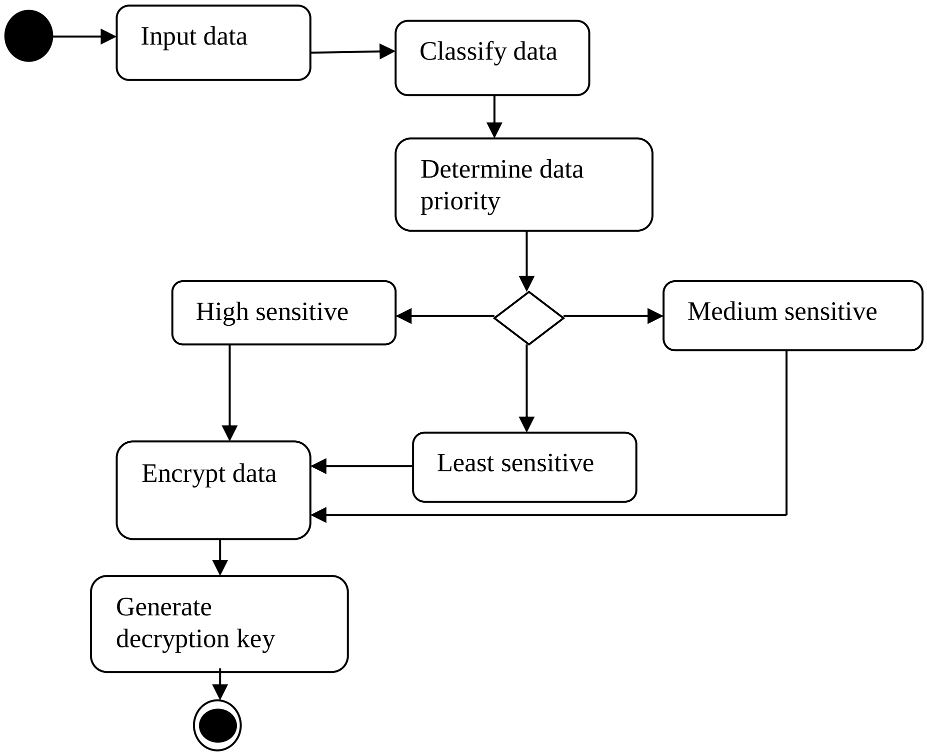


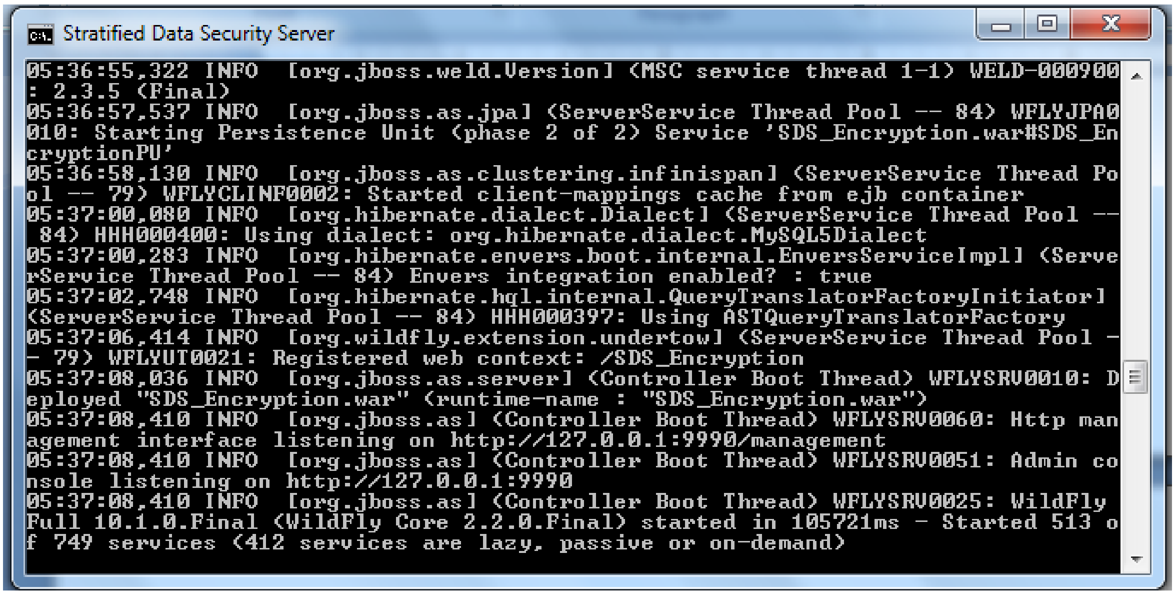
Figure 12: Process Activity Diagram of the Stratified Data Security Model

5. Results

The implementation of the system was done in WildFly environment. The code was written in Javascript. MySQL query browser was used for the implementation of the database.

The system was tested using hypothetical banking data such as, account number, ATM PIN, Bank Verification Number (BVN), etc. The test yielded a result that depicts the expected outcome of the research.

Figure 13 shows the launch screen of the server. The server is activated when the status of the services are “lazy”, “passive” or “on demand”.



```
05:36:55,322 INFO  [org.jboss.weld.Version] (MSC service thread 1-1) WELD-000900
: 2.3.5 (Final)
05:36:57,537 INFO  [org.jboss.as.jpa] (ServerService Thread Pool -- 84) WFLYJPA0
010: Starting Persistence Unit (phase 2 of 2) Service 'SDS_Encryption.war#SDS_En
cryptionPU'
05:36:58,130 INFO  [org.jboss.as.clustering.infinispan] (ServerService Thread Po
ol -- 79) WFLYCLINF0002: Started client-mappings cache from ejb container
05:37:00,080 INFO  [org.hibernate.dialect.Dialect] (ServerService Thread Pool --
 84) HHH000400: Using dialect: org.hibernate.dialect.MySQL5Dialect
05:37:00,283 INFO  [org.hibernate.envers.boot.internal.EnversServiceImpl] (Serve
rService Thread Pool -- 84) Envers integration enabled? : true
05:37:02,748 INFO  [org.hibernate.hql.internal.QueryTranslatorFactoryInitiator]
(ServerService Thread Pool -- 84) HHH000397: Using ASTQueryTranslatorFactory
05:37:06,414 INFO  [org.wildfly.extension.undertow] (ServerService Thread Pool -
- 79) WFLYUT0021: Registered web context: /SDS_Encryption
05:37:08,036 INFO  [org.jboss.as.server] (Controller Boot Thread) WFLYSRV0010: D
eployed "SDS_Encryption.war" (runtime-name : "SDS_Encryption.war")
05:37:08,410 INFO  [org.jboss.as] (Controller Boot Thread) WFLYSRV0060: Http man
agement interface listening on http://127.0.0.1:9990/management
05:37:08,410 INFO  [org.jboss.as] (Controller Boot Thread) WFLYSRV0051: Admin co
nsole listening on http://127.0.0.1:9990
05:37:08,410 INFO  [org.jboss.as] (Controller Boot Thread) WFLYSRV0025: WildFly
Full 10.1.0.Final (WildFly Core 2.2.0.Final) started in 105721ms - Started 513 o
f 749 services (412 services are lazy, passive or on-demand)
```

Figure 13: The Stratified Data Security Server Launch Screen

Upon the activation of the server, the login prompt is displayed for the user to key in the username and password as illustrated in Figure 14.

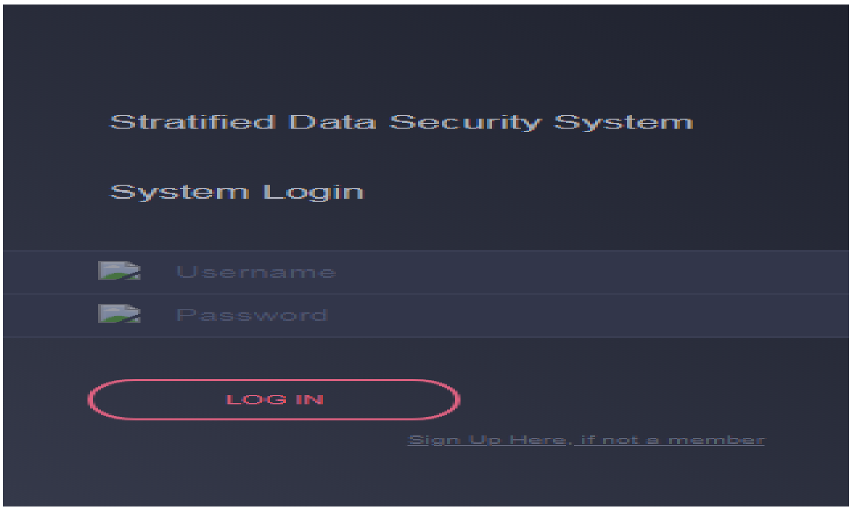
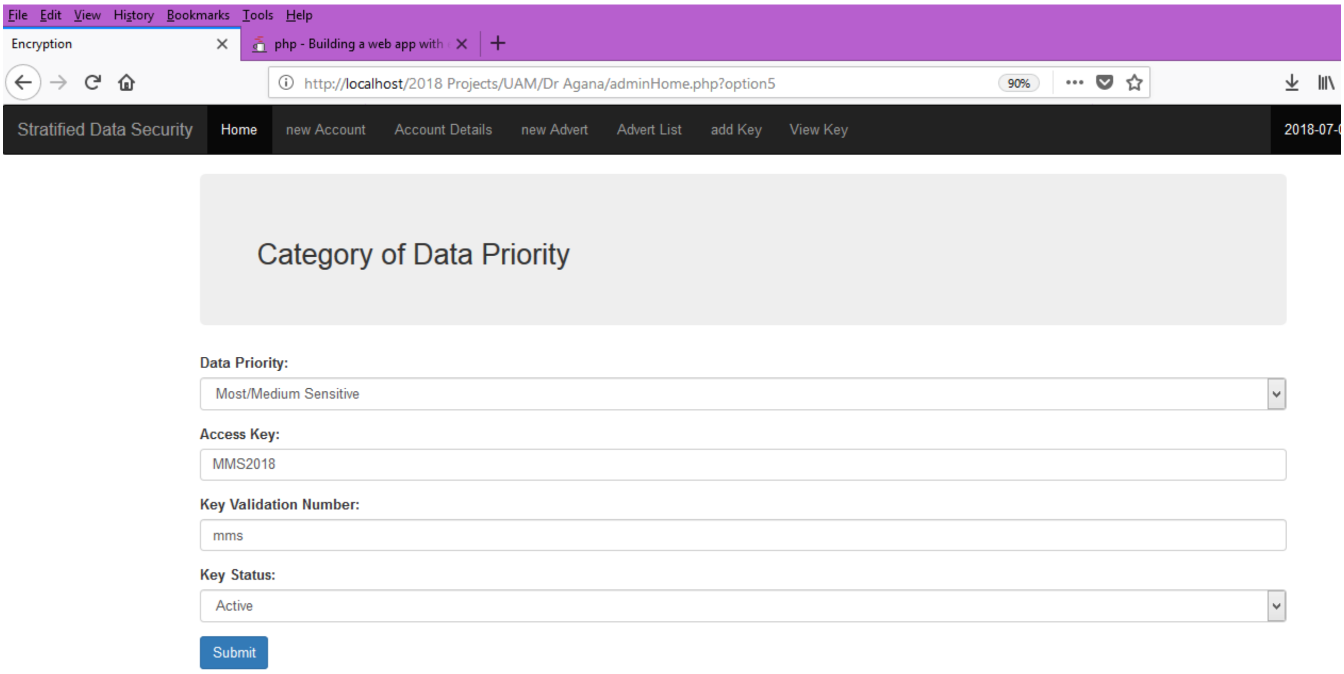

Stratified Data Security System
System Login
Username
Password
LOG IN
Sign Up Here, if not a member

Figure 14: System Login Interface

The home page enables the administrator to add the private encryption key of the data strata. He can also add new account details and use the created private key to view the encrypted account details.

Figure 15 shows the data categories priority page illustrating data priority, access key, key validation number and key status as subcomponents.


File Edit View History Bookmarks Tools Help
Encryption
php - Building a web app with
http://localhost/2018 Projects/UAM/Dr Agana/adminHome.php?option5
90%
Stratified Data Security
Home
new Account
Account Details
new Advert
Advert List
add Key
View Key
2018-07-

Category of Data Priority

Data Priority: Most/Medium Sensitive

Access Key: MMS2018

Key Validation Number: mms

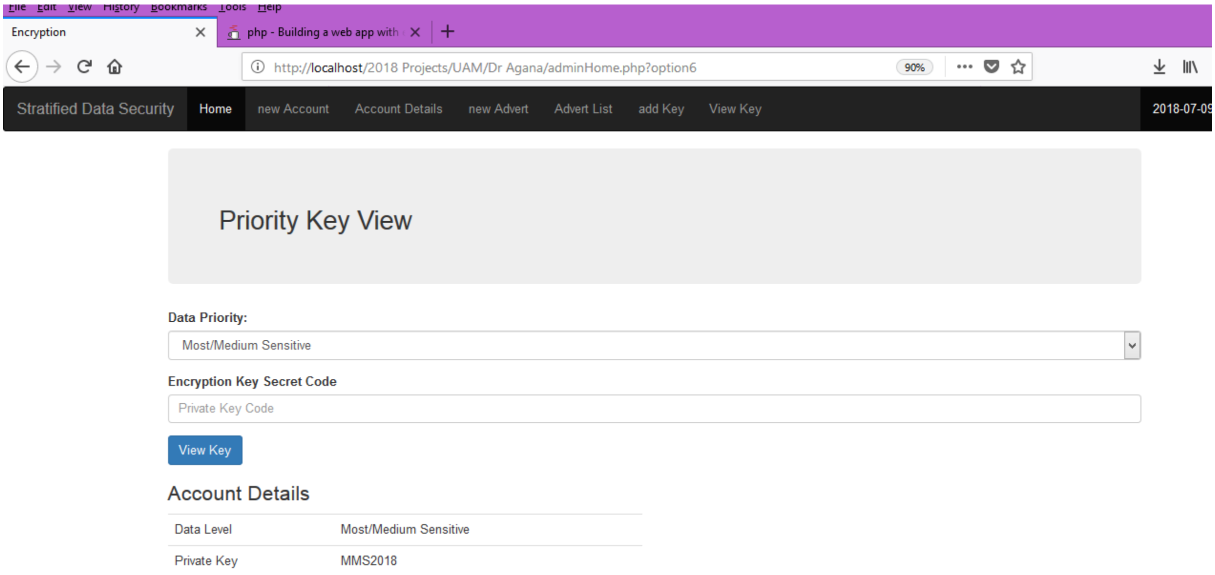
Key Status: Active

Submit

Figure 15: Categories of Data Priority Page

The key validation number is provided to enable verification before the private key can be seen by the admin.

To view the private key for each level, the admin is requested to select the data priority level and provide the key validation code before he can see it. This also displays the priority level of the customer account details. The priority key view is illustrated in Figure 16.


File Edit View History Bookmarks Tools Help
Encryption
php - Building a web app with
http://localhost/2018 Projects/UAM/Dr Agana/adminHome.php?option6
90%
Stratified Data Security
Home
new Account
Account Details
new Advert
Advert List
add Key
View Key
2018-07-09
View Key

Priority Key View

Data Priority:
Most/Medium Sensitive

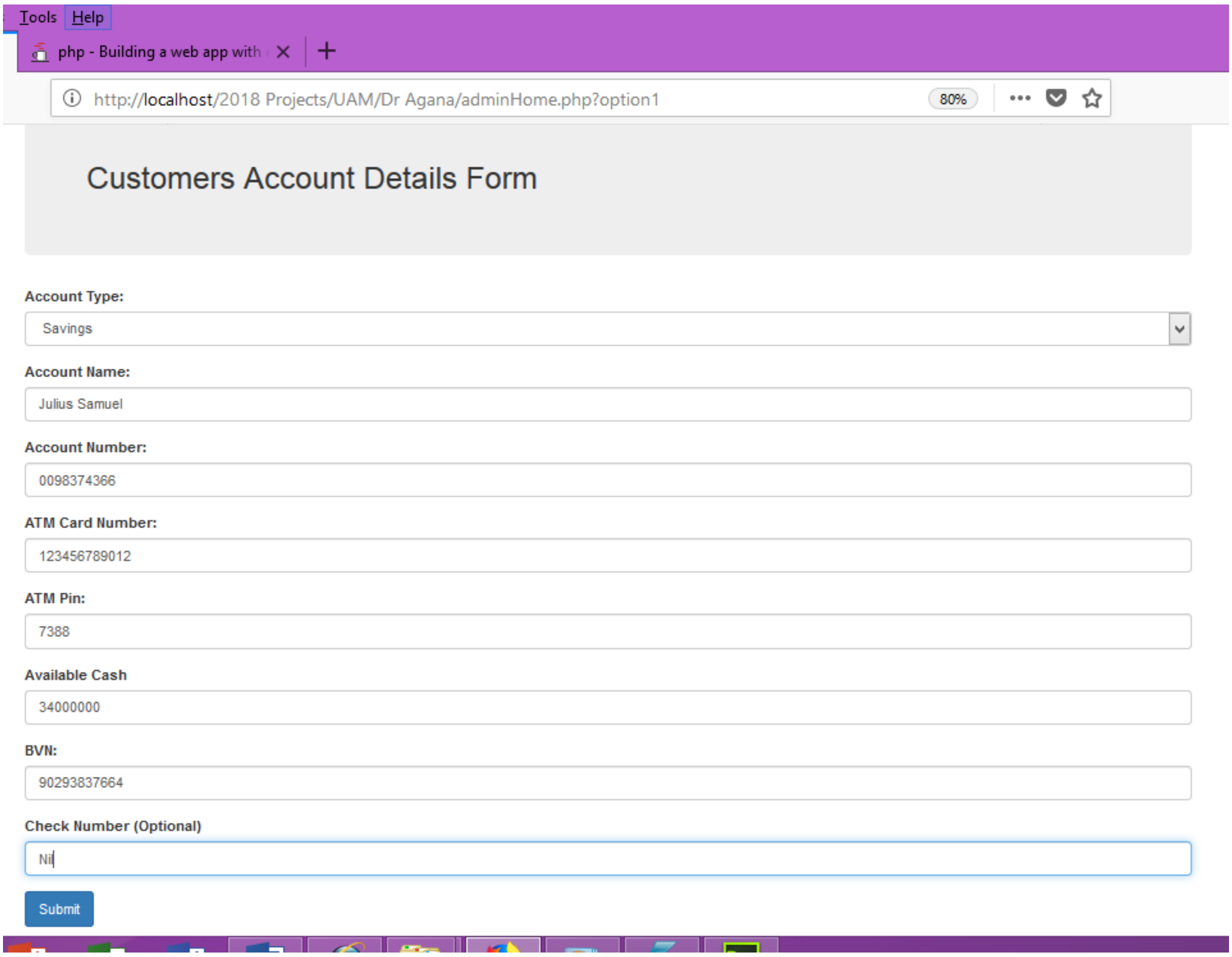
Encryption Key Secret Code
Private Key Code

Account Details

Data Level	Most/Medium Sensitive
Private Key	MMS2018

Figure 16: Priority Key View

The system automatically selects the private key registered for the Most or Medium Sensitive Class of Data and uses it to encrypt the new account details for every time a new account detail is added. The private key will then be used in decrypting it. The account details form illustrating this is shown in Figure 17.


Tools Help
php - Building a web app with
http://localhost/2018 Projects/UAM/Dr Agana/adminHome.php?option1
80%
Submit

Customers Account Details Form

Account Type:
Savings

Account Name:
Julius Samuel

Account Number:
0098374366

ATM Card Number:
123456789012

ATM Pin:
7388

Available Cash
34000000

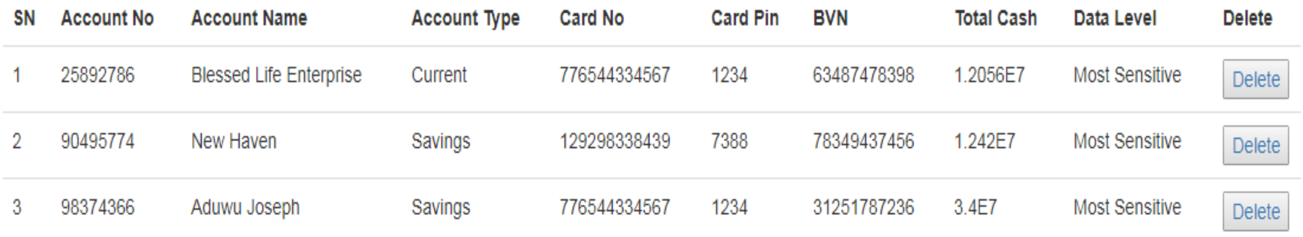
BVN:
90293837664

Check Number (Optional)
Nil

Figure 17: Customer's Account Details Form

Table III shows account details with specified priorities.

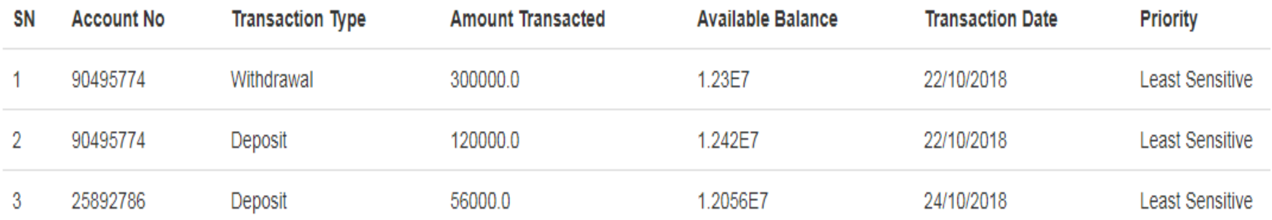
Table III: Account Details showing Priority Levels



SN	Account No	Account Name	Account Type	Card No	Card Pin	BVN	Total Cash	Data Level	Delete
1	25892786	Blessed Life Enterprise	Current	776544334567	1234	63487478398	1.2056E7	Most Sensitive	Delete
2	90495774	New Haven	Savings	129298338439	7388	78349437456	1.242E7	Most Sensitive	Delete
3	98374366	Aduwu Joseph	Savings	776544334567	1234	31251787236	3.4E7	Most Sensitive	Delete

Table IV shows transaction logs specifying the priorities of the various transactions.

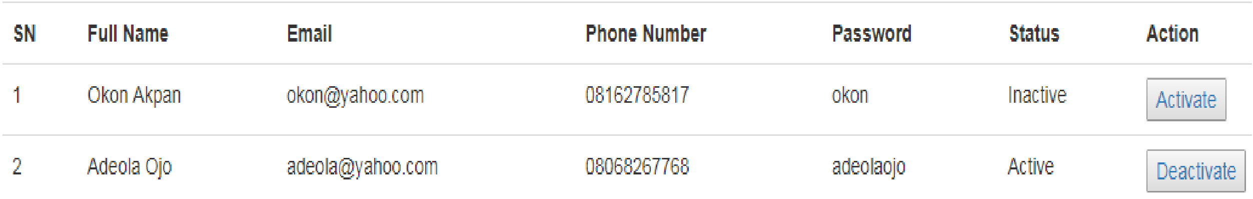
Table IV: Transaction Log



SN	Account No	Transaction Type	Amount Transacted	Available Balance	Transaction Date	Priority
1	90495774	Withdrawal	300000.0	1.23E7	22/10/2018	Least Sensitive
2	90495774	Deposit	120000.0	1.242E7	22/10/2018	Least Sensitive
3	25892786	Deposit	56000.0	1.2056E7	24/10/2018	Least Sensitive

As a monitoring strategy, the administrator regulates which staff member has access to some certain data type and who does not. He can thus activate or deactivate a staff. Table V illustrates the activation status of staff as assigned by the administrator.

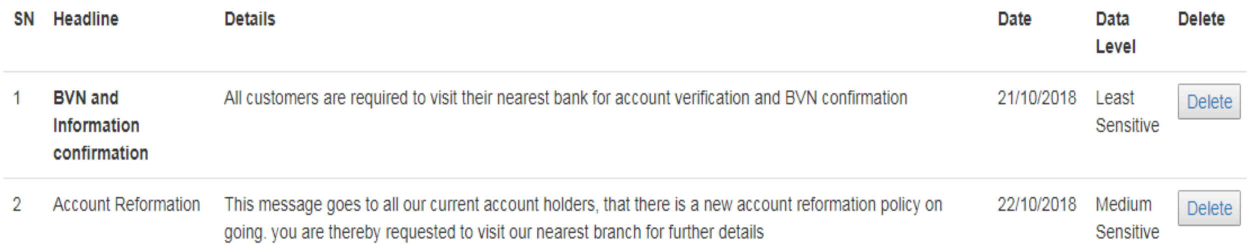
Table V: Staff Activation Status



SN	Full Name	Email	Phone Number	Password	Status	Action
1	Okon Akpan	okon@yahoo.com	08162785817	okon	Inactive	Activate
2	Adeola Ojo	adeola@yahoo.com	08068267768	adeolaojo	Active	Deactivate

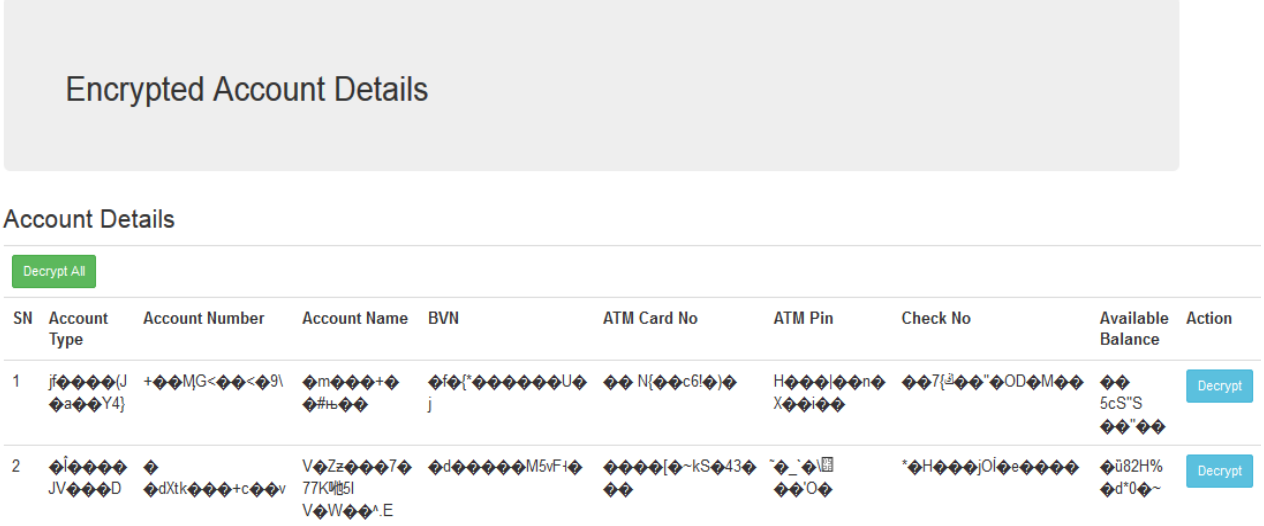
Information dissemination to customers is also prioritized based on sensitivity. Table VI shows some information made available to the customers from the system and their sensitivity levels.

Table VI: Information to Customers based on Sensitivity



SN	Headline	Details	Date	Data Level	Delete
1	**BVN and Information confirmation**	All customers are required to visit their nearest bank for account verification and BVN confirmation	21/10/2018	Least Sensitive	Delete
2	Account Reformation	This message goes to all our current account holders, that there is a new account reformation policy on going. you are thereby requested to visit our nearest branch for further details	22/10/2018	Medium Sensitive	Delete

The details of all encrypted accounts can also be viewed by the administrator. Figure 18 shows all encrypted customers' accounts; and as the administrator, one can decrypt all by entering the private key or decrypt individual accounts by entering the same private encryption key.



Encrypted Account Details

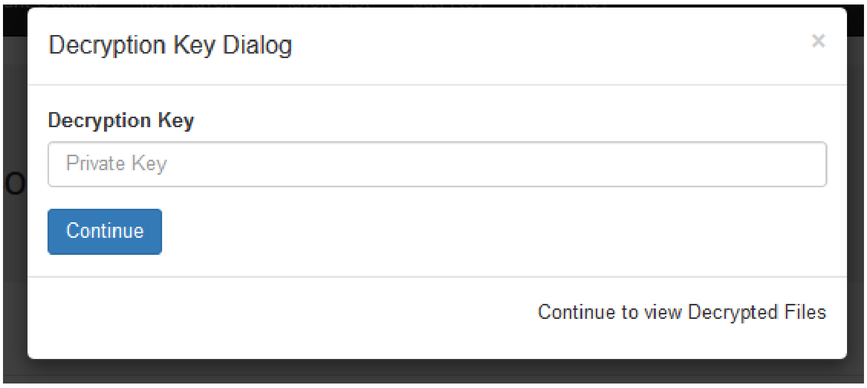
Account Details

SN	Account Type	Account Number	Account Name	BVN	ATM Card No	ATM Pin	Check No	Available Balance	Action
1	jf����(J �a��Y4}	+��MG<��<�9\	�m���+� �# њ��	�f�{*������U� j	�� N{��c6!�)�	H���|��n� X��i��	��7{ﻠ��"�OD�M��	�� 5cS"S ��"��	Decrypt
2	�Î���� JV���D	� �dXtk���+c��v	V�Zz���7� 77K��5I V�W��^.E	�d�����M5vF��	����[�~kS�43� ��	`�_`�\▯ ��'O�	*�H���jOÌ�e����	�ü82H% �d*0�~	Decrypt

Figure 18: Encrypted Accounts Details

As shown in figure 18, each data is encrypted according to its priority level.

A decryption key dialog requesting the private key with which to decrypt all the encrypted accounts prompts the administrator for the key when access is requested before the person requesting can be granted access. This is illustrated in Figure 19.



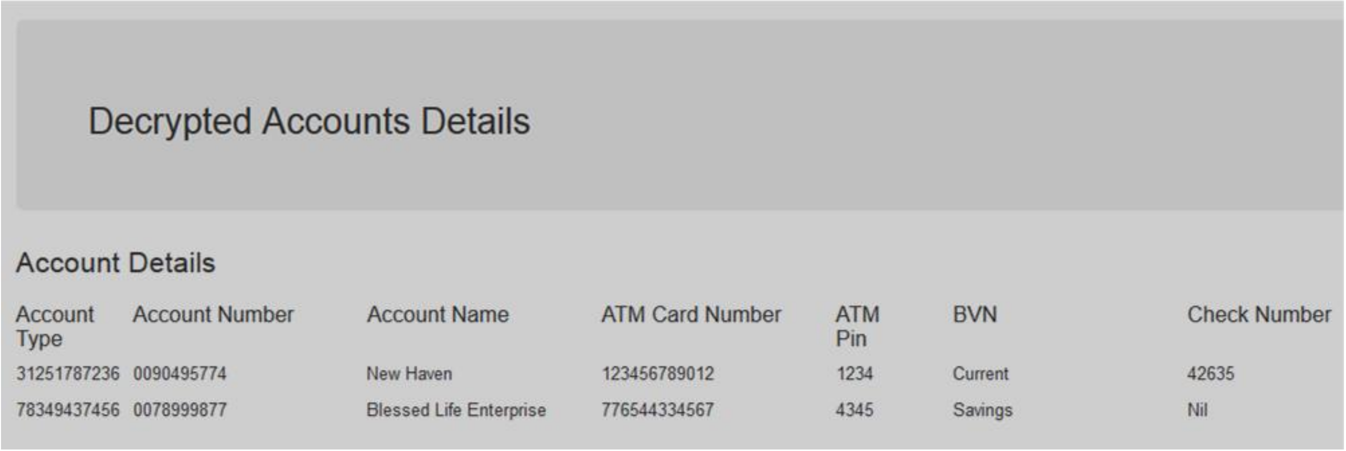
Decryption Key Dialog

Decryption Key

Continue to view Decrypted Files

Figure 19: Decryption Key Dialog Box

Entering the right key gives the decrypted account details as shown in Figure 20.



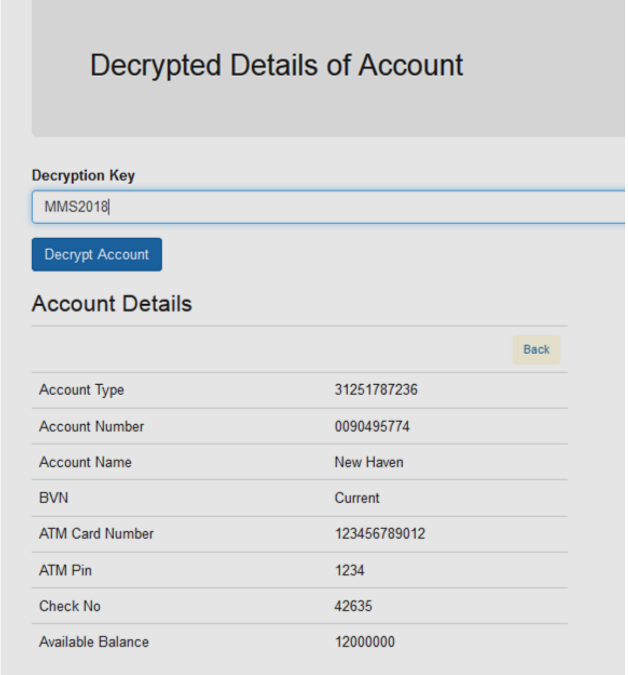
Decrypted Accounts Details

Account Details

Account Type	Account Number	Account Name	ATM Card Number	ATM Pin	BVN	Check Number
31251787236	0090495774	New Haven	123456789012	1234	Current	42635
78349437456	0078999877	Blessed Life Enterprise	776544334567	4345	Savings	Nil

Figure 20: Decrypted Account Details

To decrypt an individual customer’s account requires selecting the decrypt button and the private key must be entered to decrypt it as shown in Figure 21.



Decrypted Details of Account

Decryption Key

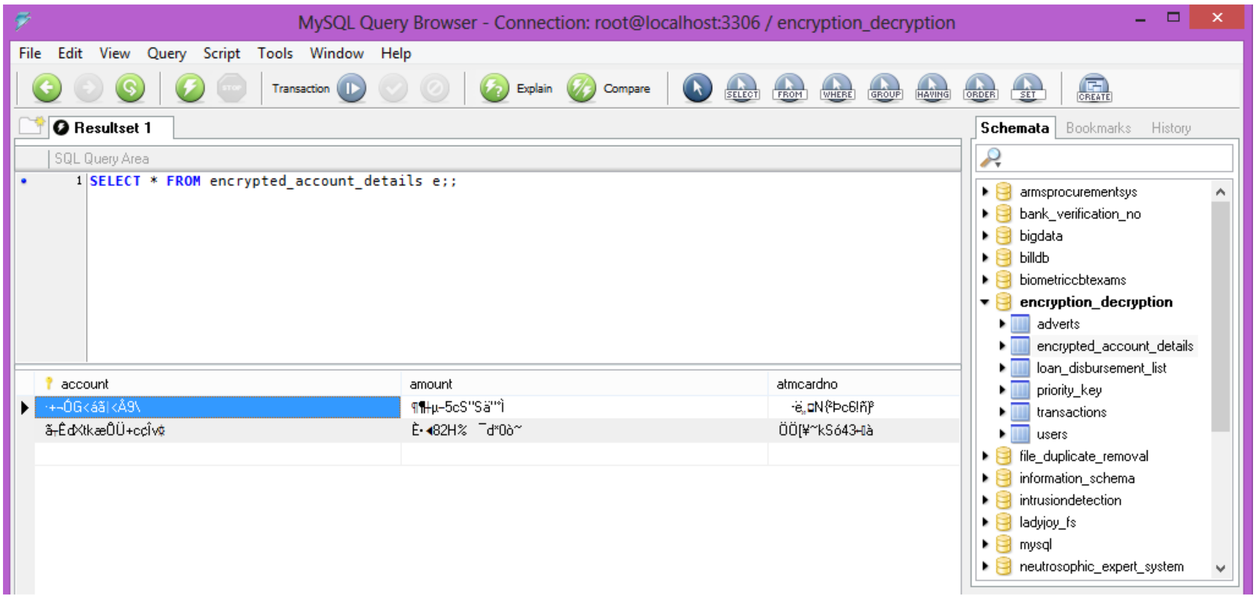
MMS2018

Account Details

Account Type	31251787236
Account Number	0090495774
Account Name	New Haven
BVN	Current
ATM Card Number	123456789012
ATM Pin	1234
Check No	42635
Available Balance	12000000

Figure 21: Decrypted Details of Individual Account Entries

The same approach is used for the entries such as adverts that belong to the least sensitive classification of data. Figure 22 shows the encrypted data entries from MySQL query browser.



MySQL Query Browser - Connection: root@localhost:3306 / encryption_decryption

```
SELECT * FROM encrypted_account_details e;;
```

account	amount	atmcardno
·+¬ÓG<áã|<Å9\	¶¶†µ–5cS"Sä"°Ì	-ë„◘N{²Þc6!ñ)³
ã┬ÊdXtkæÔÜ+cçÎv¢	È·◄82H% ¯d*0ò~	ÖÖ[¥~kSó43·←▯à

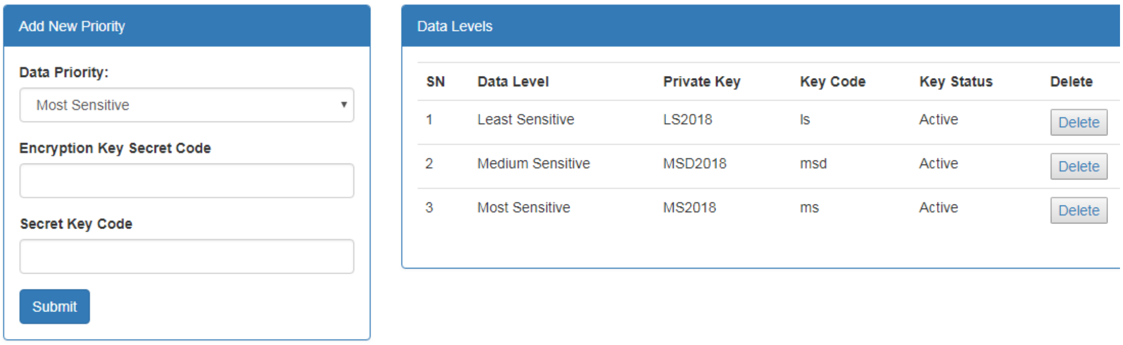
Figure 22: Encrypted Data Entries in the Database

A sample of encrypted account details in the database is displayed in Table VII.

Table VII: Encrypted Account Details

account	amount	atmcardno	atmpin	actname	bvn	acct_type	checkNo
+¬ÓG<áã|<Å9\	¶¶†µ–5cS"Sä"°Ì	-ë„◘ N{²Þc6!ñ)³	HàÖ|¤n¦†X÷²i§ß	mÛÃÁ+ý┘ ø#	├fõ{*¬¿ä÷Uë┤j	jf©ôÈ(J↑ a®Y4}	©7{Ú¬ßÅ"ÑOD¿Mæ
ã┬ÊdXtkæÔÜ+c¢Îv¢	È· ◄82H% ¯d*0ò~	ÖÖ[¥~kSó43· ←▯à	Ö®Ö_`\Ô¯ ýà'	VÚZÆ¶ûÍý7÷←ʔ77Kå'ƒ└5	Ýd¶ù¯°M5vF▯Ë§õ	xÃíë•±Ä◘JVÊ┤▯D	*¶HöjOÄ°òeïÃüÐ

The system administrator defines the sensitivity of data types as to determine the encryption level to be assigned to it. Figure 23 illustrates the interface where the administrator defines the sensitivity of various data items.



Add New Priority

Data Priority: Most Sensitive ▾

Encryption Key Secret Code

Secret Key Code

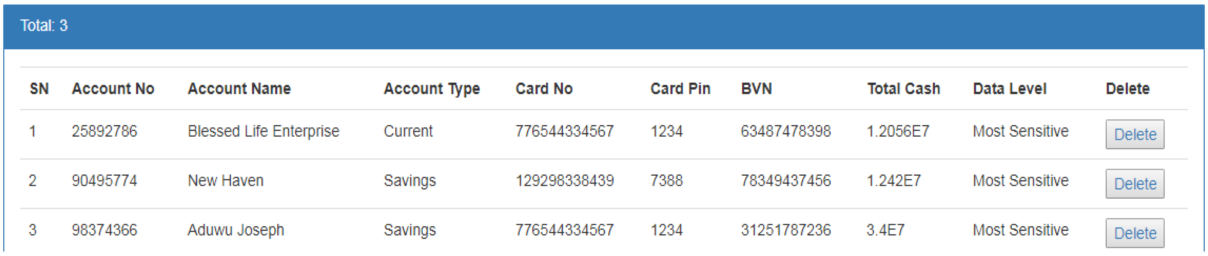
Data Levels

SN	Data Level	Private Key	Key Code	Key Status	Delete
1	Least Sensitive	LS2018	ls	Active	Delete
2	Medium Sensitive	MSD2018	msd	Active	Delete
3	Most Sensitive	MS2018	ms	Active	Delete

Figure 23: Data Sensitivity Definition

Table VIII shows account details with specified priorities

Table VIII: Account Details showing Priority Levels

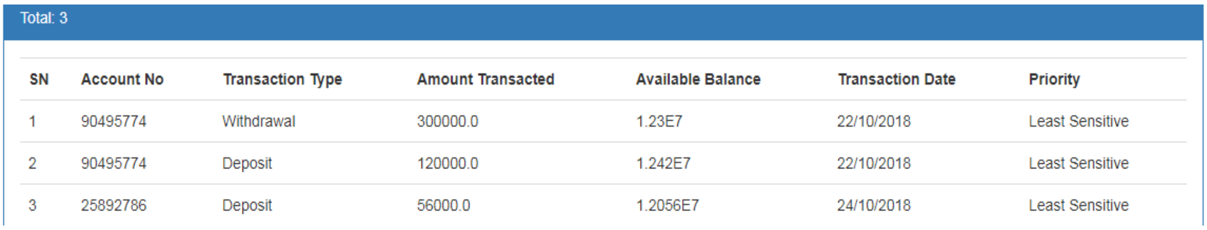


Total: 3

SN	Account No	Account Name	Account Type	Card No	Card Pin	BVN	Total Cash	Data Level	Delete
1	25892786	Blessed Life Enterprise	Current	776544334567	1234	63487478398	1.2056E7	Most Sensitive	Delete
2	90495774	New Haven	Savings	129298338439	7388	78349437456	1.242E7	Most Sensitive	Delete
3	98374366	Aduwu Joseph	Savings	776544334567	1234	31251787236	3.4E7	Most Sensitive	Delete

Table IX shows transaction logs specifying the priorities of the various transactions.

Table IX: Transaction Log



Total: 3

SN	Account No	Transaction Type	Amount Transacted	Available Balance	Transaction Date	Priority
1	90495774	Withdrawal	300000.0	1.23E7	22/10/2018	Least Sensitive
2	90495774	Deposit	120000.0	1.242E7	22/10/2018	Least Sensitive
3	25892786	Deposit	56000.0	1.2056E7	24/10/2018	Least Sensitive

6. Conclusion

This study focused on developing a stratified data security model that prioritizes data protection by integrating the Advanced Encryption Standard (AES) with a modified Diffie–Hellman algorithm, aimed at strengthening the security of organizational data.

The research was designed to provide data security to organizational data in both standalone and networked systems based on the priorities attached to each data item. The implication is that the data item with the highest value is christened as having a higher priority (high sensitivity), and is thus given a greater security consideration, while the data with a lesser degree of importance is said to be of lesser priority or sensitivity, and is as such accorded a lesser degree of security consideration.

The objectives of the system were achieved because the application was tested with hypothetical banking transaction data and was found to be efficacious in providing security for data of various classes of priority via encryption.

This system is meant for all kinds of computers and networks hosting and transmitting sensitive data since it serves a security against unauthorized access to data. It is therefore recommended for implementation in all entities where vary in both type and degree of importance.

The study was limited to the combination of only two algorithms for data encryption and protection based on their priority levels. The implication is that if attackers break this security scheme, the data becomes insecure. The study can be extended to a wider scope covering other encryption schemes and a wider range of data too.

Competing Interests: Not Applicable
Funding Information: Not Applicable

Author contribution
Moses Adah Agana: Lead researchjer and supervisor
Ofem Ajah Ofem: Co-supervisor and system analyst

John Adinya Odey: System designer
David Oboboho Egete: Co-designer and data analyst
Chiamaka Okpalanochikwa: Programmer
Iwara Obono Ofem: Programmer
Informed Consent: All authors have due consent

References

1. Aaron, W. (2018). Cyber Security Trends in 2019: Zero Trust, Biometrics, IoT and GDPR

2. Aaron, W. (2019). What is Encryption? An Overview of Modern Encryption Technology. Retrieved from https://learn.g2.com/what-is-encryption on 23-10-2019.

3. Adrian, P, Pawel, S., Raphael, M. R. and Laurent, C. (2017). Information Security and Cryptography - SCION: A Secure Internet Architecture. Cham, Switzerland: Springer eBook. https://doi.org/10.1007/978-3-319-67080-5.

4. Ahmad, S., Thanikaiselvan, V. and Rengarajan, A. (2017). Data Security through Data Hiding in Images: A Review. *Journal of Artificial Intelligence, 10, 1-21.* **DOI:** 10.3923/jai.2017.1.21, available at https://scialert.net/abstract/?doi=jai.2017.1.21

5. Ahmadi, S. (2011). Mobile WiMax: A Systems Approach to Understanding IEEE 802.16m Radio Access Technology. SienceDirect, Elsevier B.V.

6. Ahmer, K.J, Licheng, W., Tong, L. and Muhammad, A.Z. (2018). Lightweight Cryptographic Techniques for Automotive Cybersecurity. Review Article: Wireless Communications and Mobile Computing. Vol. 2018, pp. 1-15. Article ID 1640167, 15, available at **https://doi.org/10.1155/2018/1640167**

7. Aryan, C.K. & Durai, R.V.P.M. (2017). Enhanced Diffie-Hellman algorithm for reliable key exchange. Proceedings of IOP Conference Series Materials Science and Engineering 263(4):042015. DOI: 10.1088/1757-899X/263/4/042015.

8. Baker, R.D. and Mckenzie, LLP. (2017). Overview of UK Data Protection Regime (DPA1998 Version). Retrieved from https://uk.practicallaw.thomsonreuters.com/7-107-4765?transitionType=Default&contextData=(sc.Default)&firstPage=true&bhcp=1 on 28-10-2019 .

9. Chaudhuri, S., Das, G. and Narasayya, V. (2007). Optimized Stratified Sampling for Approximate Query Processing. ACM TODS 32, 2.

10. Conrad, E. Misenar, S. and Feldman, J. (2019). Advanced Encryption Standard.. In Eleventh Hour CISSP, Third Edition). ScienceDirect, Elsevier, B.V.

11. Faiqa, M., Muhammad, A., Muhammad, M. and Munam, A.S. (2017). Cryptography: A Comparative Analysis for Modern Techniques. International Journal of Advanced Computer Science and Applications, 8(6), 442-448. DOI: 10.14569/IJACSA.2017.080659.

12. Gir, M.K. (2020). A Philosophy of Security Architecture Design. Wireless Personal Communications. 113:1615–1639. https://doi.org/10.1007/s11277-020-07310-5.

13. Harshali, Z. and Ashok, S. (2018). An Efficient AES Implementation using FPGA with Enhanced Security Features. Journan of King Saud University, Article In Press. Retrieved from https://www.sciencedirect.com/science/article/pii/S1018363918302071 on 04-11-2019.

14. Henry, K. and Pasley, K. (2009). Cryptography. In Tipton, H.F. (2009) Ed.). Official (ISC)^2 Guide to the CISSP CBK, 2nd Edition, Pp 309-399.

15. Hutchinson, S.E. and Sawyer, S.C. (2000). Computers, Communication and Information: A User's Introduction. New York: DP Publications.

16. Ilia, S. (2018). Identify and Prioritize Information Security Risks. Blog/Security & Compliance. Retrieved from https://blog.netwrix.com/2018/01/04/identify-and-prioritize-information-security-risks/ on 26-06-2020.

17. Jyotirmoy, D. (2014). A Study on Modern Cryptography and their Security Issues. International Journal of Emerging Technology and Advanced Engineering, 4(10), 320-324. Retrieved from https://pdfs.semanticscholar.org/9418/5705fa1b886542a69538bb0faa7fc293e048.pdf on 23-10-2019.

18. Jeevitha, B. K, Thriveni, J. and Venugopal, K. R. (2016). Data Storage Security and Privacy in Cloud Computing: A Comprehensive Survey. International Journal of Computer Applications (0975 – 8887), 156(12), 16-27. Available at https://pdfs.semanticscholar.org/ae5b/efabeaa314f2068ad6614752aa44b478d5ef.pdf.

19. Keith, M. (2006). Cryptographic Key Management. Information Security Summer

School: Information Security Group, Royal Holloway, University of London.

20. Kessier, G.C. (2019). An Overview of Cryptography. Retrieved from https://www.garykessler.net/library/crypto.html on 29-10-2019.

21. Kim, J.Y, Oh, Y., Park, S. Cho, S. and Park, H. (2013). Stratified Sampling Design Based on Data Mining. Health Information Research, 19(3), 186-195. doi: 10.4258/hir.2013.19.3.186

22. Kire, J. (2016). Security Techniques for Data Protection in Cloud Computing. International Journal of Grid and Distributed Computing (9) 1, 49-56. Available at http://dx.doi.org/10.14257/ijgdc.2016.9.1.05

23. Mina, M., Beiruti, M.A. and Abadi, M.H.S. (2015). Assessment of High and Low Rate Protocol-based Attacks on Ethernet Networks. International Journal of Advanced Computer Science and Applications (IJACSA), 6(7), 144-157. Available at https://pdfs.semanticscholar.org/2fee/f11c5d521e76cd20ce48737c06c46af7d275.pdf

24. Nitin, J., Ajay, S. and Sandip, V. (2013). Review and Analysis of Cryptography. Techniques. International Journal of Scientific and Engineering Research, 4(3), 1-6.

25. Ogbu, H. N. & Agana, M.A. (2019). Intranet Security using a LAN Packet Sniffer to Monitor Traffic. In Natarajan M.(Eds) (2019), 9(8), 57-68: CCSIT, NCWMC, DaKM. DOI: 10.5121/csit.2019.90806.

26. Padmaa, M. and Venkataramani, Y. (2014). Encrypted Secret Blend with Image Steganography for Enhanced Imperceptibility and Capacity. Research Journal of Information Technology, 6(4), 342-355. Available at https://scialert.net/fulltextmobile/?doi=rjit.2014.342.355

27. Rasha, M. M., Rasha I. & Zinah, R. H. (2020). An Improved Diffie-Hellman Protocol Security Using Video Entropy. Joiurnal of Southwest Jiaotong University, 55(6), 1-10. DOI: https://doi.org/10.35741/issn.0258-2724.55.6.5

28. Rich, L. (2015). Are Biometrics the Future of Data Security? Government Technology. Accessed online at www.govtech.com, 15-12-2018.

29. Salomon, D. (2010). Elements of Computer Security. Verlag London: Springer. Available at https://www.springer.com/gp/book/9780857290052. **DOI** 10.1007/978-0-85729-006-9.

30. Stallings, W. (1999). Cryptography and Network Security: Principles and Practice, 2nd Edition. Upper Saddle River, New Jersey: Prentice Hall.

31. Steve, T. (2019). Case Study In Secure File Transfer: Implementing Secure FTP with SSL In a Healthcare Organization. SANS Institute, Information Security Reading Room, pp. 5-33. Retrieved from https://www.sans.org/reading-room/whitepapers/casestudies/paper/1462 on 04/11/2019.

32. Tiller, J.S. (2009). Access Control. In Tipton, H.F. (2009) Ed.). Official (ISC)^2 Guide to the CISSP CBK, 2nd Edition, Pp 1-116.

33. Xu, J., Wei, L., Zhang, Y., Wang, A., Zhou, F., and Gao, C., (2018). Dynamic Fully Homomorphic encryption-based Merkle Tree for lightweight streaming authenticated data structures. Journal of Network and Computer Applications, 107, 113–124.