SECURING DATA IN CLOUD ENVIRONMENT USING ENIGMA AND DES COMBINATION ENCRYPTION

MOLEENA MOHANTA¹, DHRUMIL SUNIL CHABLANI², DR.M.KOWSIGAN³*(CORRESPONDING AUTHOR)

¹,²,³ Department of Computer Science and Engineering, 
College of Engineering and Technology, 
Faculty of Engineering and Technology, 
SRM Institute of Science and Technology, 
SRM Nagar, Kattankulathur 603203, Kanchipuram, Chennai, TN, India 
¹ mj6874@srmist.edu.in, ² ds7297@srmist.edu.in, ³* kowsigam@srmist.edu.in

ABSTRACT

As physical markets are moving to cyberspace, the requirement for IT infrastructure for data storage purposes is growing exponentially. To secure sensitive data and information from third parties, data security in a system is of paramount importance. Encryption is amongst the most reliable methods to improve security and confidentiality. Encryption algorithms hide the actual representation of data in a manner that the original information is obtained only by using a key known as the decryption method. The goal of encryption is to protect data from hackers during transit and at rest. Even though encryption is practiced to translate information into an unintelligible format, we cannot be sure that classified data would not be accessible to cybercriminals who seem to be getting intelligent by the day.

In this paper, we imbibe previous works in the field of Cloud security and the methodologies used to protect and secure these systems. Our aim is to use Enigma encryption and Data Encryption Standard algorithm for scrambling plain text to secure cipher text. SMTP mail services are used for reliable transmission. Furthermore, this paper gives particular emphasis to enigma encryption and its comparisons to various encryption and decryption algorithms. This project is designed for encrypting human correspondence only; thus, plain texts and cipher text messages consisting solely of the English letters A through Z, plus a few more characters can be encrypted and decrypted.

Keywords: Enigma Encryption, SMTP, DES, Decryption, Data Security, Cloud Computing

1. INTRODUCTION

Many of us communicate in cyberspace despite actually realizing the security of the exponential growth of digital networking and electronic information sharing. In cyberspace, we share a lot of our sensitive documents and secrets. Our online communication is in cyberspace, whether we like it or not. Everything that we exchange is always vulnerable and accessible to cybercriminals [1].

Information and resources on the cloud are usually available and accessible to everyone, and as they are stored in the cloud, cloud users lose their hold on the data. A strict access management policy on confidential data is required in many application components to ensure privacy and ensure stability so that data is accessible only to approved users [2].

A method to data confidentiality is by securing information with a password. As passwords become another very basic user authentication procedure, extracting passwords seems to be a common and easiest attack strategy [3]. The other method to have a password is to scam users by social manipulation or online phishing. The theft of a password may be accomplished by taking a look through an individual's workspace, obtaining an unsecured password link through the Internet, or having access to a database and viruses [4]. Therefore, one such way to use
the technique of authentication is to encrypt the information to prevent other people from reading and removing or manipulating the data.

The main idea of encryption or security of data as a whole relies on three pillars [5]. Often referred to as the “Three Pillars of Authentication”, these entities are used to signal the authenticity of a user. A user may prove his/her identity by using either one or a combination of two or more of the following entities:

- What you know
- What you have
- What you are

These three identifiers in combination can be used to authenticate a user. The example of each could be as: “What you know” entity refers to a thing that only a privileged user would know. This kind of authentication mainly includes passwords. The next identifier - “What you have” talks in regards to something a user might possess. This encompasses everything from a 2FA physical keys, to RFID Identity cards to proprietary software. The third and the final pillar of authentication - “What you are” deals with what a user is. This method covers biometrics primarily, i.e. fingerprints, retina imaging, palm scans, etc. [6][7][8]

The system proposed in this paper combines the first two types of authentication - “What you know” and “What you have” to build a secure and robust system that is both easy to use and complex to break.

The rest of this paper is arranged as follows. Section II talks about the preliminaries of Enigma encryption, Data Encryption Standard (DES), Simple Mail Transfer Protocol (SMTP) and its deployment in the cloud environment. Section III briefly sums up the existing models and their comparisons and limitations. Section IV describes the proposed system and its implementation. Section V explains the experimental study and result. Finally, Section VI and VII wind up the paper with a conclusion and future scope.

II. PRELIMINARIES

A. Enigma Encryption

This module is an upgraded implementation of the original Enigma machine used by the Germans in World War II. Enigma is a stream-encryption technique, i.e. it breaks down plaintext into singular bits/characters after which ciphering takes place. The following module is made to undertake human-level, so it is limited to English alphabets and a few other special characters.

The basic working of enigma is based on the translation of each bit/character into a different character as it passes through the plugboard, rotors, and reflector. [9] A character is first passed on to the plugboard where it is swapped with another character based on the plugboard pair settings. Next, the transcoded character passes through a set of three rotors placed adjacent to each other. Each rotor has 26 values from ‘A’ to ‘Z’ and a spin functionality to scramble each bit differently. Post the rotor encoding, the bit passed through the reflector where it is swapped based on the reflector setting, back to the rotors and the plugboard to finally give the resultant bit. [10] The below-shown fig.1 states how a plaintext is encrypted and decrypted using enigma mechanism.

For ease of implementation, the module is conveniently divided into four sub-modules. The sub-modules are as follows:

- **Enigma:** This is the master sub-module and its main function is to make sure the other sub-modules work in tandem. This module is responsible for enigma configuration input which includes rotor setting, reflector setting, ring setting, ring start positions, plugboard pairs, and plaintext message for encoding and decryption key and ciphered message for decoding. Furthermore, this sub-module has the delicate task of checking the validity of user input before passing it to other sub-modules. Finally, this sub-module is also assigned with function call management and access control.

- **Rotor:** The rotor sub-module mimics a single rotor used by the enigma machine. Each rotor has 26 values that represent the English alphabet. The following module includes eight different rotors, each with a different combination of the same 26 alphabets. Out of these eight rotors, only three are used at a time. The selected rotors are used in different combinations to increase the total number of configuration possibilities.

www.turkjphysiotherrehabil.org
• **Reflector:** The reflector is the simplest sub-module with one primary task, to encode and return each bit back to the rotors for further scrambling.

• **Plugboard:** This sub-module of Enigma is in charge of creating alphabet pairs. This means that any two alphabets are grouped together so that swapping may occur during the encoding process. The plugboard provides the most amount of scrambling possibilities when compared to other sub-modules. The plugboard alone is capable of creating about 150 trillion possible enigma configurations.

![Fig 1: Working of encryption and decryption using Enigma](image)

**B. Data Encryption Standard (DES)**

The DES module or the Data Encryption Standard Module is an algorithm that is based on block encryption techniques. As compared to a stream cipher that takes one bit at a time, a block cipher like DES splits the input plaintext into 64-bit blocks, each of which becomes a single input for the DES algorithm. The encoding of each 64-bit block produces a ciphere string of the same length. [11][12]

Data Encryption Standard uses the same key and algorithm for encryption as well as decryption. The key originally used has a length of 64-bits [13], but 8 bits are removed, which are later used as parity. The 8, 16, 24, 32, 40, 48, 56, and 64th bits are removed from the key to finally give out a 56-bit useable key.

Coming to the algorithm itself, DES consists of 16 rounds wherein two basic processes, substitution and transposition take place. In each round, there are two basic inputs, the first is the 64-bit plaintext received from the previous round and a 48-bit key. It must be noted that the 48-bit key for each round is derived from the useable 56-bit key by means of half-split and left circular shift (LCS). For rounds 1, 2, 9, and 16 1-bit LCS occurs whereas for all other rounds 2-bit LCS occurs whereas for all other rounds 2-bit LCS takes place. After the shift has occurred, a permuted choice must be made to reduce the 56-bit key to a 48-bit key. This process is called ‘Key Transformation’.

Once the key transformation process is complete, the 48-bit key is used as an input to each round along with 64-bit plaintext from the previous round. In each round, the 64-bit plaintext input is divided into half, i.e. two plaintext strings of 32-bit each. Next, the right hand plaintext (32-bit) is passed through an Expansion permutator to increase its length to 48 bits. Finally the two 48-bit inputs go through a series of processing including an XOR operation, substitution, permutation and another XOR with the left 32-bit plaintext to give a resultant 64-bit cipher text. This aforementioned process represents a single round and is repeated 16 times to obtain the final result - the DES Cipher Text.

**C. Simple Mail Transfer Protocol (SMTP)**

The SMTP module or the Mail module has the primary function of sending the encrypted message and encoded decryption key to the receiver via e-mail service. [14] In this case, we will use Gmail servers as they are readily available in most regions and comparatively more secure than other free mailing services. As a part of this module, a large chunk of work is undertaken by public APIs, inbuilt classes, and external Java Archive (JAR) files.

The entire module can be expressed in three simple steps which are as follows [15]:

[www.turkjphysiotherrehabil.org](http://www.turkjphysiotherrehabil.org)
1. **Getting the session:** For every new mail that needs to be sent, a new session needs to be created. In this case, the session is initiated simply by creating an object for the `javax.mail.session` class and using a set of credentials (username and password) of a Gmail ID that will be used to send the mail. To start the session, the SMTP host is set to `smtp.gmail.com` and the port is set to 465 while keeping SSL and Authentication to a Boolean true value.

2. **Compose the Message:** This is the portion of the module wherein the encrypted message and the decryption key is attached to the mail. Here, the Java Mail API is used to bind the content to the email. To start, the 'From name' and 'subject' is inserted followed by the actual content which includes but is not limited to the encrypted message and decryption key. Finally, the recipient's mailing address is added. It must be noted that the aforementioned details are inserted using an object of the `javax.mail` class and preexisting functions of the class.

3. **Sending the Mail:** The final step consists of mail transmission. For this step, the `javax.mail.Transport` class is used. A straightforward object creation and 'send' function call is all it takes for the successful delivery of the mail to the recipient.

**D. Cloud Environment**

The cloud module is used for data storage and program development. The expansion of the cloud has sparked off the use of various kinds of cloud service deployment models like public, private, hybrid, and community cloud. The proposed framework will be primarily implemented on a private cloud. This project is specifically designed for testing purposes; hence it will be using a cloud stimulator like CloudSim to deploy. In the future, AWS Elastic Beanstalk may be used for deployment purposes. Apart from that, implementation in public and hybrid cloud will require the study of three versions of the cloud sector that are being widely used: Platform as a Service (PaaS), Infrastructure as a Service (IaaS), and Software as a Service (SaaS). [16][17][18]

This module is developed and incorporated into the project for three main reasons: 1. Cloud servers are vast, open, convenient platforms that are used by various institutions and individuals to store and access sensitive data, 2. Software integration happens seamlessly in the cloud, not much effort is required to configure and integrate the software into the cloud and 3. Daily, cloud systems are subjected to a wide variety of attacks from expert hackers to malicious insiders [19]. Therefore, an augmented framework is proposed to secure all the information present in the cloud using multiple encryption techniques. This combination encryption method will not only secure the data efficiently but will also be better in terms of time complexity.

With the increase in technology demand, this module is subjected to change. Hence, all our future works will be based on the structured on-demand computing model.

**III. RELATED WORK**

Over the past few years, data security has gained prominence due to the increasing number of attacks on public and private systems online. Numerous approaches have been introduced to bring cessation to these illegal practices. One of the earliest contributions in this field was done by Sumana [20]. Her research focuses on Proxy Re-encryption. Proxy re-encryption (PR) is a platform that protects and communicates cloud data safely with receptors, offers an effective solution for securing data. Herein, each client sends the data via its public key and preserves its encrypted data with the cloud storage service. So when data in the cloud has to be exchanged, the client calculates the re-encryption key by using a private key and the public key of the intended recipient. Later on, the key is redirected to a third-party proxy server. The proxy server rearranges the ciphertext to re-encrypted ciphertext using the key and transfer of the ciphertext to the recipients. The problem with such a system is that as the recipient number increases the scale of the key and the ciphertext remains constant.

In [21], a contributory broadcast encryption is introduced to an associate encryption system that enables a station to safely transmit decryption keys to all subgroups of members but needs an authorized member to transmit them. A group of people agrees on a shared public encryption key within that recent paradigm when each member has a decryption key. A sender who has the public encryption key will restrict decryption towards a handful of its participants. Anyone can send private information to almost any group of people in the community, and perhaps the system needs no reliable server. The proposed method performs effectively but it fails when design standards
are high for contributory broadcast encryption within short ciphertexts. In addition to that, this method sustains a heavy encryption cost.

Wang and Wu [22], introduced some changes to the already existing homomorphic encryption technique proposed by Vaikuntanathan and Brakerski [23]. The following technique is different because it allows to perform computation without first decrypting the encrypted data. Moreover, this methodology allows authoritative public key encryption to multiple users with less estimating overhead in comparison to other pre-existing systems. There are primarily four fragments of this system model: Data Users, Data Owners, Cloud Servers, and Key Generation Centre. If a data owner wishes to externalize the data to a cloud server, he first encodes his file with a public key algorithm. The data owner then sends a message to the cloud server after adding PEKS (Public-key encryption with keyword search) to the ciphertext of each keyword. The message can be represented as:

\[ F(\text{File}) \parallel \text{PEKS}(x_1,k) \parallel \cdots \parallel \text{PEKS}(x_n,k) \]

This model proposed in the paper was significantly better in every aspect, but it is time-consuming and difficult to implement in normal use case scenario.

Another technique was proposed by Chen, Liu and Zhang [24], which uses an Advanced Encryption Standard (AES) and Elliptic Curve Cryptography (ECC) in a way that the key can be transmitted and administered without a secure hub to safeguard the confidentiality of the AES key and the privacy of the user. This procedure enhances the cryptographic speed and reliability of common data. Hence, the system is highly stable. The only disadvantage it has is that the system doesn't provide monitoring control to managers who handle the secure center which leads to problems like collision and data manipulation.

A system administrator normally encrypts the data to preserve the integrity of data in the cloud in order to decrypt data by those designated data users. This poses a grave concern if more users than the data holder originally appointed have to share the encrypted information. To deal with this challenge, Zhou and Deng [25] implement and formalize an IDE model through the cohesive incorporation of two well-established encryption mechanisms, respectively, IDE and ID-based broadcast encryption (IBBE). Within that permitted limit ‘n’ number of users who could view the same information, the computational cost of the installation algorithm is linear. Also, it allows users to choose the first secure identity-based encoding systems to ensure security and later convert encrypted information so that they can provide access to information from another encoding scheme. Between each process this system involves coordination between owner and authority, which might lead to an efficiency crisis.

Tang [26] suggested an arrangement for encrypting user data depending on all questions to satisfy the user's complex query requirements. A dual AES encryption approach is also used to address the intensity of attack in deterministic encryption. A competitor row sharing approach is built into the system to protect the privacy of the users when their data is modified. Furthermore, when modifying data on encrypted databases, the nearby rows sharing system will safeguard user's privacy and security. The relative findings reveal that the proposed method has decent success on the user proxy’s encryption costs, but it is lagging for user aggregation query costs.

Encryption is indeed an effective way of delivering customer data protection security to the public cloud computing providers while enabling consumers to scan their keywords using their encrypted data [27]. This methodology enables the search of several keywords in which any search keyword can comprise zero and every keyword location can contain a wildcard and display any set of symbols. Thus it provides customizable user permission and cancellation so that search and decryption rights are easily managed. It also boasts a strong level of privacy, as the cloud service in the testing stage is ignorant of matching data. Wildcard encrypts ciphertext securely into multiple encrypted pieces, as needed. Additional processing is required to improve the accuracy so that spelling mistakes can be avoided.

The newly introduced strategy in the field of information security is reversible information protection via encryption. In [28], the image encryption scheme and the hiding data function are merged into one machine with reversible data concealed by encryption strategy. During the encryption process, the main concept is to include three various encryption keys and the recipient can make a block-based image encryption. The hidden message bit that is to be inserted into it decides the collection of the pseudo-alternative bytes for encrypting a certain picture frame. The only drawback of this technique is that the transmitter can only incorporate a single bit into an image.
frame, with the frame size indicated by randomized controlled trials being 9x9 pixels. The only downside is that a bit of error happens most of the time because the vector model is incorrectly labeled.

Kowalski and Zyczkowski [29] suggested a fiber transmission system. The method of encryption depends on pseudo-random amplification of spatial light, the blend of two encryption keys, and the compressed sensing substructure. The needed encryption output is a linear combination of light waves and pseudo-random sequences. For transmitting encrypted information via optical fiber, the proposed architecture is used. The obtained experimental structure is planned to regulate 16-cell SLM data and transfer encrypted information via a single optical fiber. The findings of the observational work show that this is an efficient way to encrypt data. The approach involves additional research, explicitly because of its computational complexity.

In [30], six methodologies were introduced for safer and efficient access and processing of database information. The CIA triad is a model developed to enable information security management systems within the database [31], [32], [33],[38][39]. Many encryption methods are developed, ECC is among the most successful. But this paper primarily uses AES symmetric encryption. The very first phase of this encryption process is to place the data in two-dimensional (2D) array, after which multiple encryption rooms repeat cipher transitions. This system requires more attention as data can be changed by unauthorized people outside of the system [34], [35][36][37]. Moreover, when an attacker tries to acquire the key either from row or column of the 2D array, the database collapses. Hence, not an efficient way to support data integrity.

IV. PROPOSED SYSTEM

The proposed system mentioned in this paper is based on encryption techniques that are primarily used for human conversation encoding. Thus, this paper uses encoding techniques that facilitate the ciphering of English alphabets along with a few special characters. The entire system has been coded in Java 11 with Netbeans 12 LTS as the IDE. Java was used in this particular case because of its ease in implementation and better integration with existing platforms where secure encryption is required. These may include everything from cloud platforms to stand-alone systems in both public and private environments. The IDE Netbeans was chosen for its customizability and external jar importation features which make it simple to add complex tasks without a significant increase in the number of lines of code.

The main system is divided into three modules that handle offline encoding and transmission and a module to deploy the system to various environments. The modules are split based on individual tasks that are undertaken by each module. The following are the working and description of each module:

1. Enigma:

As the name suggests, this module has the primary task of encrypting plaintext to enigma cipher. To perform this function, this module has further divided into four sub-module as a java class for precise control over each method. The sub-modules are “Enigma”, “Rotor”, “Plugboard” and “Reflector” that mimic the actual working parts of the German military enigma machine.

The enigma module starts with taking inputs from the user. These include rotor order, reflector, ring settings, ring start position, plugboard pairs, and the initial plaintext message. The rotor order input takes three roman numerals (out of eight) as input in left to right order for the rotor sub-module. Next is the reflector input wherein alphabets A, B, or C are used to select the type of reflector to be employed. After this comes the ring setting where the rotor setting for each previously selected rotor is entered and this is taken in as a set of three numbers ranging from 01 to 26. Post this, the input for ring start positions is acquired by accepting three alphabets as starting indexes. The last configuration setting is plugboard pairs; this input is variable as the user may or may not elect to use swap pairs. If the user so chooses, the input is secured as multiple alphabet pairs separated by space. In the end, the user is prompted to enter the plaintext message that is to be encoded. For easier understanding, the following table shows the input format for each configuration variable.

<table>
<thead>
<tr>
<th>Sr. No</th>
<th>Setting input</th>
<th>Format</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Rotor Order</td>
<td>Three roman numerals separated by space</td>
<td>II IV VI</td>
</tr>
</tbody>
</table>
Reflector
A single alphabet (A/B/C)

Ring Setting
Three numbers each of range 01-26 separated by space
01 02 26

Ring Start position
Three alphabets separated by space
X Y Z

Plugboard pairs
Alphabet pairs separated by space [min 0 pairs | max 13 pairs]
AQ BZ LM

Plaintext
Regular English expression
Encode this message

Once the inputs are all accepted, a decryption key is generated in the following format:

(Rotor order) - (Reflector) - (Ring Setting) - (Ring Start Position) - (Plugboard Pairs)

Thus, for the inputs provided in the table above, the decryption key will look like this:

(II IV VI)-(C)-(01 02 26)-(X Y Z)-(AQ BZ LM)

Next, coming to the working of the entire machine, the following algorithm will showcase the operation very clearly:

Algorithm I: Encrypting the plaintext using Enigma

```
declare & initialize rotor
declare & initialize reflector_swap_pairs
declare & initialize normal_alphabet and scrambled_alphabet
declare & initialize plugboard_swap_pairs
declare & initialize plaintext_msg
for each character in plaintext_msg :
    // Apply plugboard transcoding
    for each pair in plugboard_swap_pairs :
        if (character included in swap pair) :
            swap (original value, pair value)
        end if
    end
    // Apply forward rotor transcoding
    for each rotor alphabet :
        normal_alphabet = indexOf(character)
        scrambled_alphabet = new character at that index
        swap (character, new character)
    end
    // Apply reflector transcoding
    for each pair in reflector_swap_pairs :
        if (character included in swap pair) :
            swap (original value, pair value)
        end if
    end
    // Apply reverse rotor transcoding
    for each scrambled_alphabet :
        scrambled_alphabet = indexOf(input character)
        normal_alphabet = new character at that index
        swap (character, new character)
    end
    // Apply plugboard transcoding
```
for each pair in plugboard_swap_pairs:
    if (character included in swap pair):
        swap (original value, pair value)
    end if
end

ciphertext = ciphertext + transcoded input

// Rotor turn
for each rotor [L => R]:
    left alphabet = indexOf(left notch)
    right alphabet = indexOf(right notch)
    if (left index == right index):
        cycle left alphabet onwards by 1
    end if
end

cycle right-most alphabet onwards by 1

The following is a class diagram representing class distribution, class interaction, member functions and variables used in the Enigma module:

Fig 2: Class diagram of the proposed Enigma module

This module will output the description key in plaintext format and the enigma ciphered message.

2. Data Encryption Standard (DES)

This the second module of the system and its duty is to encode the plaintext decryption key generated by the enigma module. The reason for the second layer of encryption is to make sure that if the key is captured by an attacker, it remains in an unreadable format. Data Encryption Standard (DES) is a block cipher, meaning that instead of working like a stream cipher taking one bit at a time, DES splits the entire plaintext into 64-bit blocks and then encrypts each block. DES algorithm is based on a key that remains the same for both encoding and decoding processes.
The following diagrams fig3 and fig4, show the working of DES. Fig 3 shows how the entire encoding inclusive of 16 rounds works with a given initial key and a plaintext input. Fig 4 on the other and aims to describe a single round using Expansion, Permutation and XOR processes.

The input for this module is the decryption key in plaintext format generated by the enigma module. A function call from the enigma module is responsible for passing the plaintext key to the DES module. The simplified working of DES has been previously explained in this paper under preliminaries. The following algorithm lays out the detailed implementation of DES.

The output of this module is a cipher decryption key. To keep this key secure and away from alterations during transmission, it is stored in a text file.

3. Simple Mail Transfer Protocol (SMTP) Mailing

This is the last and final module of the encryption and transmission process. The reason for the existence of this module is for the secure transmission of the key and the encrypted message. This module receives its input from both the previously mentioned modules. The Enigma module provides the enigma encoded message and the DES module provides the double encoded decryption key. Furthermore, the mailing module requires the recipient’s email address to send the message and key.

www.turkjphysiotherrehabil.org
Once all three inputs are received, the module starts its main process. The mailing process occurs in three stages—Session Start, Message Composition, and Mail Send. The following table describes each step briefly:

<table>
<thead>
<tr>
<th>Sr. No</th>
<th>Module</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Session Start</td>
<td>In this step, a new session window is opened using the host name and port and establishing a connection to the host. For this system, Gmail services are used because of their reliability. To create a session and establish a connection, inbuilt methods from classes like “javax.mail.Session”, “javax.mail.Authenticator”, and “javax.mail.InternetAddress”.</td>
</tr>
<tr>
<td>2</td>
<td>Message Composition</td>
<td>This step involves message drafting by using “javax.mail.MimeMessage” and “javax.mail.MimeBodyPart” class functions. The primary method used is MimeMessage which helps in adding sender name, subject, mail content as well as attachments. This is particularly useful as message transmission requires content space while the decryption key needs to be sent as an attachment.</td>
</tr>
<tr>
<td>3</td>
<td>Mail Send</td>
<td>The final step occurs after a connection has been established with the host and the mail has been composed. With the use of the “javax.mail.Transport” class functions, the mail is sent to the recipient.</td>
</tr>
</tbody>
</table>

V. EXPERIMENTAL STUDY AND RESULT ANALYSIS

Every aspect of human life has been gravely impacted by the pandemic situation. Prior to the coronavirus outbreak, infrastructure management providers, as well as companies, were looking for migration of their primary software, storage and computing services to a variable cost model. The obvious choice was to migrate to the pay-as-you-go model, an integral idea of cloud infrastructure. The decision for migration was made easier due to the increasing web traffic caused by the pandemic. As a result of schools shutting down, physical boundaries closing and official work from home requirements, cloud data centers have undergone tremendous changes to provide for the changing supply in computational power. As per the statistical data available, last year, only 10% of cloud users reduced their usage while 31% show a slight increase from their projected usage and 26% of cloud users demanded for significantly higher computation power and storage than planned.

**Fig 6:** Cloud services used for enterprises from 2019-2021
Cloud platforms are being widely used by companies to speed up processes, scale up operations and increase productivity of cloud-based workloads. Looking at different technologies that have moved to cloud environments, it is evident that the migration to cloud is exponential. In fig.6, it is evident how technologies are moving to a more secure and reliable cloud infrastructure.

However, the vastness of cloud services and the sheer amount of data stored on the cloud is also inviting attackers alike.

The need for a secure platform rises as more users rush to use the cloud environment. Some cloud users deploy private security algorithms for their data, while a major chunk of cloud users rely on security measures provided by cloud providers. Cloud providers employ a variety of security measures, one such measure is encryption. Encoding techniques are a must for any environment to secure data in transit as well as data at rest.

The stream cipher encoding used in this paper is based on German Enigma used during World War II. The original enigma had 159 quintillion combinations while the upgraded version of enigma used here has the following number of possible combinations.

Possible combinations of Enigma:

- Rotors = $8 \times 7 \times 6 = 336$
- Start Positions = $26 \times 26 \times 26 = 17576$
- Plugboard combinations = $\frac{26!}{6!10!2!10!} = 150,738,274,937,250$

TOTAL COMBINATIONS = 890,190,309,219,827,616,000 (890 quintillion approx.)

With the possible 890 quintillion options, if a brute force attack had to take place on a supercomputer in 2019 with a speed of 100 petaflops (100 Trillion keys per second), it would take around 103.03 days without taking DES encoding into consideration.

During the course of this paper, encryption algorithms like AES, DES and 3DES were also compared for their time complexity based on the input size. The following graph shows the comparison of DES, AES and 3DES as simulated on an Intel Pentium 4 (2.4 GHz) machine in ECB mode.

![Execution Time Comparison Chart](chart.png)  
**Fig 7:** Execution time comparison of DES, AES and 3DES
The findings show that DES has an edge in terms of performance over other algorithms. There are certain flaws in every process. Although the execution time of the DES for large kilobytes of plaintext is less, it is incapable of providing adequate security. Therefore, the proposed system uses a double encryption model wherein the main encryption is undertaken by Enigma encoding whereas DES encrypts the decryption key for better security.

VI. CONCLUSION

Cloud security is essential as it has become a core requirement for most businesses. The paramount importance of cloud computing is putting compulsion on cloud service providers to keep their client data secure. This paper discussed various encryption techniques used for securing data in the cloud environment. Remarkable attention was given to Enigma encryption and decryption methodology and its elements. Based on the significance and comparison of various models a system was proposed. The proposed system incorporates Enigma, DES, and SMTP that enhances the efficiency of data protection in the cloud. The following system will constitute a new path for safe broadcasting platforms. The whole mechanism provides improved robustness. The detailed evaluation confirms the accuracy and reliability of our system. Organizations dealing with highly confidential details should opt for this system for securing their cloud databases, as this system is easily attainable in any cloud platform.

VII. FUTURE SCOPE

The following paper put forward a system to secure data transmission in cloud. This section of the paper discusses some of the future advancements in this field.

Cloud computing bestowed a whole new sphere of platforms and services which aid to generate IT solutions and provide infrastructure to the same. Collaborating with IoT and Cloud, large amounts of data can be stored in the cloud. Multiple investigators have proved that within the next few years Cloud Computing will be a leading technology.

Contrary to the demand, one must oversee the security aspects of it. Poor initialization can cause data breaching. Protocols exist to limit this but they are not structured properly to give efficient results. Additional efforts are needed in this section to ensure reliability.

Our defined model is comparatively a cut above the ones widely used these days. The future is uncertain. Hence, periodical changes must be introduced in any system to keep it relevant.

REFERENCES:


