

# MAIL ENCRYPTOR

Mohd Umar (18BCE0196), Atharva Hundare (18BCE0246), Anand  
Kumar(18BCE2225),Ashutosh Singh(18BCE2279)

---

A report submitted for the J component of

**CSE3502** - Information Security Management

**Supervisor:** Dr.D.RUBY



**VIT**<sup>®</sup>  
**Vellore Institute of Technology**  
(Deemed to be University under section 3 of UGC Act, 1956)

School of Computer Science and Engineering

Vellore institute of Technology, Vellore

# Declaration

This report has been prepared on the basis of my own work. Where other published and unpublished source materials have been used, these have been acknowledged.

Word Count: 4105

Student Name: Mohd Umar (18BCE0196), Atharva Hundare (18BCE0246), Anand Kumar(18BCE2225)  
Ashutosh Singh(18BCE2279)

Date of Submission: June 1, 2021

Signature:

Contents

MAIL ENCRYPTOR ..... 1

Declaration ..... 2

Abstract ..... 4

Project Specification ..... 4

1.Introduction ..... 5

1.1 Idea ..... 5

1.2 Scope ..... 5

1.3 Comparative Statement ..... 5

1.4 Expected Results..... 7

2.Literature Review ..... 7

3.Architecture ..... 10

4.Algorithm ..... 12

4.1 AES Algorithm..... 12

4.1.2 Substitution Bytes ..... 13

4.1.3 Shift Rows..... 14

4.1.4 Mix Columns ..... 15

5.Results and Discussions..... 16

6.Conclusion..... 23

Bibliography ..... 23

# Abstract

Email encryption has for long being an active field of research ever since the advent of the field. Companies and institutions providing email services work on end-encryption based on user authentication alone. Other than that, email services have relied on a zero-encryption transmission. This is because most encryption mechanisms including depreciated encrypted techniques and modern techniques like AES and SHA take a certain amount of time to be executed upon the text. This creates inconvenience for the users. However, research suggests that in the era of increasing digital communication, encryption becomes a primary need for every user. For this, most organisations turn to custom security protocols or tools that require additional payment to the email service provider. So, the authors have tried to provide a minimal cost tool that allows three layers of authentication and an additional encryption with little time consumption for securing email transactions.

## Project Specification

**Project Name:** Mail Encryptor

**Date:**

**Project Developers:** Mohd Umar, Atharva Hundare, Anand Kumar, Ashutosh Singh

**Professor:** Ruby D

**Objective:**

To create a safe and secure mail encryptor which can encrypt and decrypt mails to provide security.

**Description:**

Mail Encryptor is a custom text encryptor built on Python-Django framework and uses a simple customized smtp service to allow any user logged in to its system and also to this program to send and receive emails directly through gmail service. The encrypted message itself is encrypted into a string. The software has been built atop Python 3.8.6 and uses Django 3.1.2 framework. The admin of the server can look at all transactions, public keys and the keys involved in encrypting messages and user profiles.

**Dependencies:**

Python, Django, JavaScript

## Interface:

Web Browser (Websites)

# 1.Introduction

In today's digital age, in our everyday life many individuals continually use specialized applications that are texting, person to person communication, web search tools, bookmarking frameworks, partner frameworks, print media and standard mail. These tools are explicitly used as a form or means of internet communication. However as a consequence of widespread availability and utilization of internet, the severity of security problems has increased and its importance is compounded. Two main problems which are tied to the internet is its open nature. The encryption and decryption of information are designed to cope as well as mitigate this security gap. Mail Encryptor is a custom text encryptor built on Python-Django framework and uses a simple customized smtp service to allow any user logged in to its system and also to this program to send and receive emails directly through g-mail service. The encrypted message itself is encrypted into a string. The software has been built atop Python 3.8.6 and uses Django 3.1.2 framework. The admin of the server can look at all transactions, public keys and the keys involved in encrypting messages and user profiles.

## 1.1 Idea

While sending emails on Gmail we cannot be sure whether the information remains confidential since Google can still access the mails. We wanted to make sure that the content is only accessible to the desired recipient and no one else has access to the message shared. The idea of the project is to build a web app named **securemails** which will help users to send emails using Gmail but in encrypted format and the receiver decrypts it back using our web app. We implemented the message encryption by Advanced Encryption Standard (AES) using Stanford Javascript Crypto Library (SJCL).

.

## 1.2 Scope

- A small web server will be developed that shall serve as a means encrypting as well as decrypting emails. The web server will be made using html, JavaScript and built atop Django .
- Also, as a part of our project, the email which is being sent over gmail will be first encrypted using AES Algorithm.
- The web server will also have decryption facility.
- A Django admin module will be created to provide audit functionality.

## 1.3 Comparative Statement

[1]This paper emphasised on the existing problems emails generally face .This paper further implied the technological advancement's effects on emails and how they are getting vulnerable by the minute. The paper

also the expansion of easy access to networks has resulted in obvious rewards for some non-commitment groups capable of stealing information while disseminating it to communities. The email user is given a suggestion to add another security feature such as encryption for email content before it is transmitted using the ESP service. In this paper the AES and 3DES method of encryption has been tested and used successfully to protect email messages. Email messages are pre-encrypted using both algorithms, and then tested and analyzed on the basis of a wide breadth of factors. Test results show that AES is better in terms of integration time, while 3DES is better in terms of message size after the encryption process, but additional changes by bytes are not significant. Therefore, based on test results, email users are recommended to use AES encryption.

2.[4] The authors analyzed various encryption methodologies and tested it on a standard email message while simulating a virus attack. The authors thereby conclude that of all the detection techniques currently in use, Generic signature Scanning is computationally cheapest while also offering a high degree of efficiency. For analysis, they have suggested that heuristic-based techniques are highly accurate. However, they have also suggested that owing to the current computational power available with most hackers and system engineers, heuristic based techniques need to be improved in terms of speed.

3.[6] The authors have used AES-256 and SHA-256 on a cloud server to test the efficiency and compatibility of the same. They have tried to implement integrity and verification of the data stored on the cloud by proposing a new methodology that works towards cloud servers. The method is fairly simple and involves an upload followed by key generation followed by system encryption of the files after a hash key is generated. This encrypted file is then forwarded to a central database. After conducting a reliability control test, the authors concluded that there is little or no bug involved in the project and a strong data integrity is maintained. All these tests were done using Laravel framework.

4.[8] The authors have discussed Random Number Generators, Pseudo-Random number generators and compared them over CPU execution time and computational cost. They then propose the usage of a combined Key derivative 8 function using hashing and random and pseudo random key generators. This has been implemented using Blowfish block cipher and HMA cipher. Argon2id keying material. And finally, entropy is introduced in the hash function of the key generation. This results in the generation of a more secure key generation mechanism.

5.[11] In today's digital age, in our everyday life many individuals continually use specialized applications that are texting, person to person communication, web search tools, bookmarking frameworks, partner frameworks, print media and standard mail. These tools are explicitly used as a form or means of internet communication. However as a consequence of widespread availability and utilization of internet, the severity of security problems has increased and its importance is compounded. Two main problems which are tied to the internet is its open nature. The encryption and decryption of information are designed to cope as well as

mitigate this security gap. In this paper RC4 based encryption algorithm is used to ensure secure Email communications. One major disadvantage of RC4 algorithm is that it does not provide any sort of authentication for the user to verify if the message was sent by the intended sender and not any other malicious actor. The authors also do not discuss any sort of key exchange mechanism over an unsecured environment which is the case for any data travelling over the internet.

## 1.4 Expected Results

- Encrypted message using AES algorithm
- Decryption Facility for the received mail.

## 2. Literature Review

[1] This paper emphasised on the existing problems emails generally face. This paper further implied the technological advancement's effects on emails and how they are getting vulnerable by the minute. The paper also the expansion of easy access to networks has resulted in obvious rewards for some non-commitment groups capable of stealing information while disseminating it to communities. The email user is given a suggestion to add another security feature such as encryption for email content before it is transmitted using the ESP service. In this paper the AES and 3DES method of encryption has been tested and used successfully to protect email messages. Email messages are pre-encrypted using both algorithms, and then tested and analyzed on the basis of a wide breadth of factors. Test results show that AES is better in terms of integration time, while 3DES is better in terms of message size after the encryption process, but additional changes by bytes are not significant. Therefore, based on test results, email users are recommended to use AES encryption.

[2] This paper was based on applying Advanced Encryption Standard (AES) algorithm to text based data mainly .txt files. This paper further explained the sublime simplicity and effectiveness of the algorithm on server based encryption tools and further implied the difficulty of hackers in accessing the actual data when being encrypting by AES algorithm. Till date there is no evidence of defeating this algorithm. AES can manage three distinctive key sizes, for example, AES 128, 192 and 256 digit and every one of these codes has 128 bit block size. This paper gave an outline of the AES calculation and clarify a few pivotal highlights of this calculation exhaustively and show some past research that has been done on it compared to other algorithms such as DES, 3DES, Blowfish etc. This paper also demonstrates a pseudo code for the implementation of the text based encryption which we incorporated in our project.

[3] The authors have discussed the Advanced Encryption Standard (AES) algorithm, one of the most commonly used as well as broadly symmetric block cipher algorithms currently in use. The algorithm employs a unique salt and structure method that makes it one of those algorithms which have no generic means for cracking as of now. In this paper, the authors have provided a summary of AES algorithm and explain its several crucial nuances as well as features of this algorithm while also demonstrating some previous researches that have been done on it with contrasting it to other algorithms such as DES, Blowfish, 3DES etc.

[4] The authors analyzed various encryption methodologies and tested it on a standard email message while simulating a virus attack. The authors thereby conclude that of all the detection techniques currently in use, Generic signature Scanning is computationally cheapest while also offering a high degree of efficiency. For analysis, they have suggested that heuristic-based techniques are highly accurate. However, they have also suggested that owing to the current computational power available with most hackers and system engineers, heuristic based techniques need to be improved in terms of speed.

[5] Using Python as the programming language, the authors have used both ECC and RSA and compared them over a set experimental data. They have further discussed ECC built upon a ElGamal algorithm and this was done over a server-client model. They have concluded that modular mathematics based ECC is highly efficient and faster in key generation. They further theorized on the basis of the end-to-end server encoding using the particular algorithm which we tried and actually incorporated into our key generation, the encryption keys were based of this research paper.

[6] The authors have used AES-256 and SHA-256 on a cloud server to test the efficiency and compatibility of the same. They have tried to implement integrity and verification of the data stored on the cloud by proposing a new methodology that works towards cloud servers. The method is fairly simple and involves an upload followed by key generation followed by system encryption of the files after a hash key is generated. This encrypted file is then forwarded to a central database. After conducting a reliability control test, the authors concluded that there is little or no bug involved in the project and a strong data integrity is maintained. All these tests were done using Laravel framework.

[7] The authors have discussed DH algorithms to create a key exchange protocol on a multi-peer server. The authors have pointed out that DH algorithm is in fact limited in its scope because it is very computation intensive. A modular arithmetic equation based DH algorithm has been proposed with a memory unit to store session keys. The project has very high efficiency and has reduced space and time complexity, while also being computationally cheaper in all respects.

[8] The authors have discussed Random Number Generators, Pseudo-Random number generators and compared them over CPU execution time and computational cost. They then propose the usage of a combined Key derivative 8 function using hashing and random and pseudo random key generators. This has been implemented using Blowfish block cipher and HMA cipher. Argon2id keying material. And finally, entropy



is introduced in the hash function of the key generation. This results in the generation of a more secure key generation mechanism.

[9] As the world is being rapidly digitized an entirely new type of criminal activities has emerged threatening the security and integrity of the system, network, and information. This is becoming one of the most important problems. There is classified information that can have implications on national level. This paper focus on securing e-mail communication on Android OS by using the hybrid cryptosystem which is a combination of symmetric, public key encryption system and hash function. The paper uses Ping Pong family of stream ciphers to achieve data confidentiality. One major drawback of stream ciphers is that they are quite susceptible to insertions and modifications. An active interceptor who breaks the algorithm might insert spurious text that looks authentic.

[10] End To end encryption is the most ideal approach to ensure computerized messages. Truly, end to end encryption has been amazingly hard for individuals to utilize, yet ongoing devices have made it all the more extensively open, generally by utilizing key-index administrations. These administrations penance some security properties for accommodation. The creators needed to see how normal clients consider these tradeoffs. They led a 52-man client study and found that members could figure out how to comprehend properties of various encryption models. Clients additionally made intelligent evaluations about when various tradeoffs may be proper. Members perceived that the less-advantageous trade model was safer generally speaking, yet viewed as the enrollment model's security adequate for most regular purposes. Perhaps the greatest limit of the examination,, is members' close absolute absence of knowledge of encryption assignments. For instance, one member who saw the trade model previously kept on expecting the irritation of dealing with reporters' locks in any event, when managing the enlistment model later. This beginner disarray may add with the impacts found in our and different examinations, making it hard to demonstrate long haul convenience.

[11] In today's digital age, in our everyday life many individuals continually use specialized applications that are texting, person to person communication, web search tools, bookmarking frameworks, partner frameworks, print media and standard mail These tools are explicitly used as a form or means of internet communication. However as a consequence of widespread availability and utilization of internet, the severity of security problems has increased and its importance is compounded. Two main problems which are tied to the internet is its open nature. The encryption and decryption of information are designed to cope as well as mitigate this security gap. In this paper RC4 based encryption algorithm is used to ensure secure Email communications. One major disadvantage of RC4 algorithm is that it does not provide any sort of authentication for the user to verify if the message was sent by the intended sender and not any other malicious actor. The authors also do not discuss any sort of key exchange mechanism over an unsecured environment which is the case for any data travelling over the internet.

### 3.Architecture

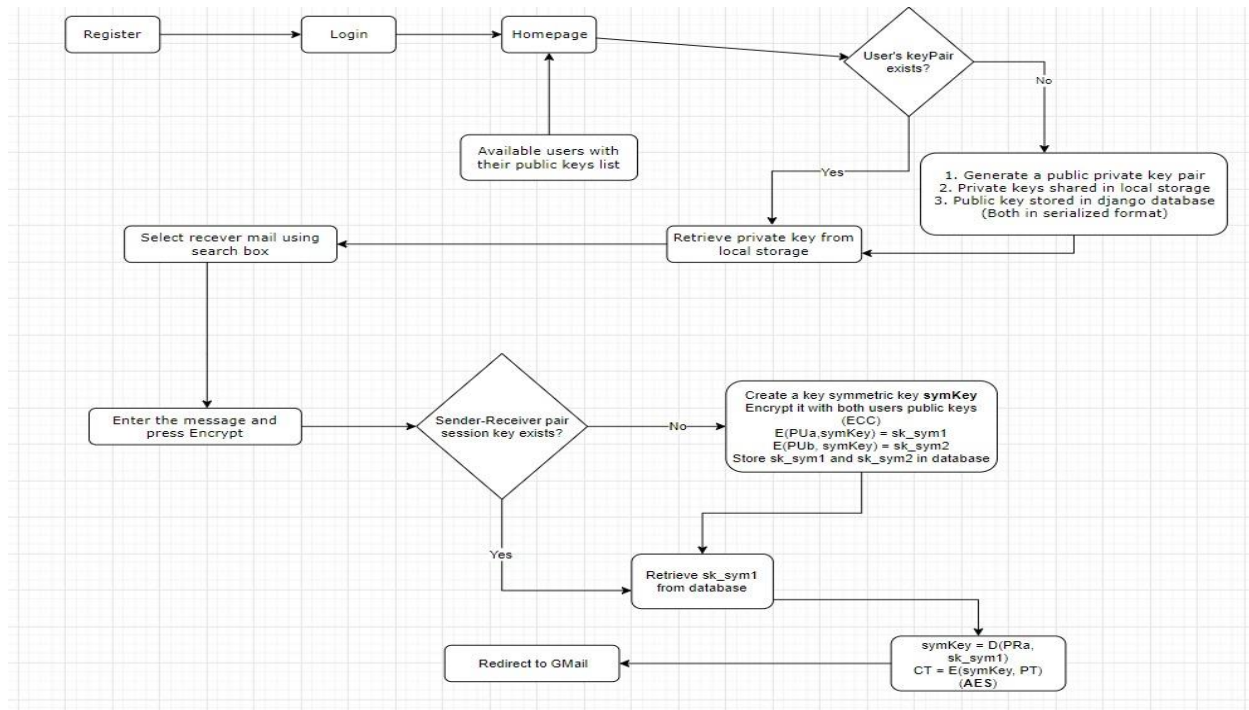


Figure 1 Architecture Diagram

As we are trying to create a mail server and receiver module to demonstrate our encryption method, we'll create our own server in Python. Every step in the encryption can be viewed for experimental purposes. We will further try to model the time complexity of the framework so as to compare it with the present day framework of mails. We will be doing all this with the help of AES Encryption Algorithm. The following are the steps

1. User registers on the portal.
2. User logs into his account and is redirected to the home page.
3. The home page contains a text area for typing in the message to be sent and a list of users already registered on the portal navigable through a search box.
4. If the user's public-private key pair does not already exist, a new public-private key pair is generated. Private key is stored in local storage while the public key is stored in the Django database - both in serialized form.
5. Otherwise, the private key of the user is retrieved from the local storage and public key from Django database
6. User selects the recipient from the list as receivers using search box.
7. User enters the message and presses the 'Encrypt' button.

8. If sender-receiver pair session key does not already exist, a symmetric key **symKey** is created.

It is encrypted with both users' public keys

$$E(PU_a, \text{symKey}) = \text{sk\_sym1}$$
$$E(PU_b, \text{symKey}) = \text{sk\_sym2}$$

sk\_sym1 and sk\_sym2 are stored in the database.

9. Otherwise, retrieve sk\_sym1 (Encrypted public key of sender) from the database.

10. The symmetric key for encrypting the message (session key) is deciphered using

$$\text{symKey} = D(PR_a, \text{sk\_sym1})$$
$$CT = E(\text{symKey}, PT) \text{ (AES)}$$

11. User is redirected to Gmail to send the encrypted message.

12. Receiver logs into the portal and enters in browser the link he receives on GMail and he gets the message decrypted.

## 4. Algorithm

### 4.1 AES Algorithm

The design of AES is based on the idea of substitution permutation and it is efficient and quite fast both in hardware implementation as well as the software implementation. Not at all like its archetype DES, AES doesn't utilize a Feistel network. AES is a variation of Rijndael which has a fixed square size of 128 pieces, and a critical size of 128, 192, or 256 pieces. On the other hand, Rijndael as such is indicated with square and key sizes that might be any numerous of 32 pieces, with at least 128 and a limit of 256 pieces.

AES works on a  $4 \times 4$  segment significant request cluster of bytes, named the state. Most AES estimations are done in a specific limited field.

### Basic Structure of AES

# Rounds  $N_r = 6 + \max\{N_b, N_k\}$

$N_b$  = 32-bit words in the block

$N_k$  = 32-bit words in key

AES-128: 10

AES-192: 12

AES-256: 14

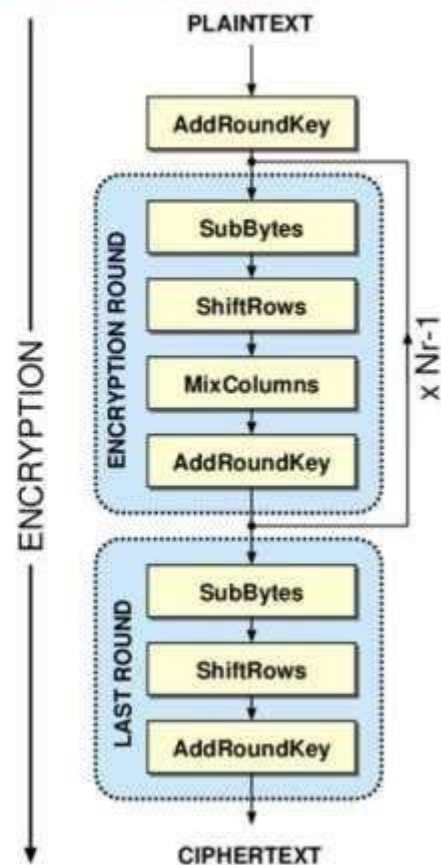


Figure 2 Basic Structure of AES

For instance, if we have 16 bytes, from  $b_0$  to  $b_{15}$  then their representation as a matrix is like :

$$\begin{bmatrix} b_0 & b_4 & b_8 & b_{12} \\ b_1 & b_5 & b_9 & b_{13} \end{bmatrix}$$

Figure 3 Representation of 16 bytes  $b_0$  to  $b_{15}$

The key size utilized for an AES figure determines the quantity of change adjusts that convert the info, called the plaintext, into the last yield, called the ciphertext. The quantity of rounds are as follows

*10 rounds for 128-bit keys.*

*12 rounds for 192-bit keys.*

*14 rounds for 256-bit keys.*

Each round comprises of a few handling steps, including one that relies upon the encryption key itself. A bunch of opposite adjusts are applied to change ciphertext back into the first plaintext utilizing a similar encryption key.

	00	01	02	03	04	05	06	07	08	09	0a	0b	0c	0d	0e	0f
00	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
10	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
20	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
30	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
40	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
50	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
60	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
70	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
80	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
90	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a0	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b0	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c0	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d0	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e0	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f0	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

*Figure 4 S-Box for AES*

#### 4.1.2 Substitution Bytes

Every byte is changed by byte at indexed by row (left 4 bits) & column (right 4 bits) of a 16x16 table bits) of a 16x16 table

EA	04	65	85
83	45	5D	96
5C	33	98	B0
F0	2D	AD	C5

→

87	F2	4D	97
EC	6E	4C	90
4A	C3	46	E7
8C	D8	95	A6

Figure 5 Substitution Bytes

#### 4.1.3 Shift Rows

- The first row is left unaltered.
- The Second row does one byte round shift to the left row and does one byte round shift to the left.
- Third row does a Two byte round shift to the left
- The fourth row does a three byte round shift at the left

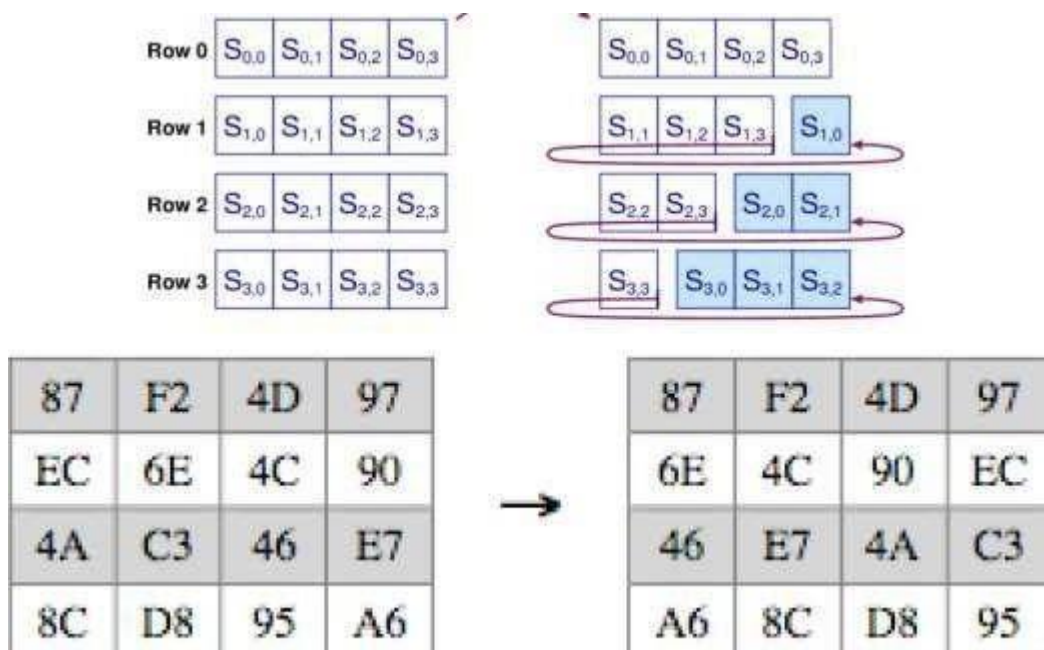


Figure 6 Shift Rows

#### 4.1.4 Mix Columns

Effectively it is a matrix multiplication in GF(2<sup>8</sup>) using prime polynomial  $m(x) = x^8 + x^4 + x^3 + x + 1$

$$m(x) = x^8 + x^4 + x^3 + x + 1$$

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{bmatrix} = \begin{bmatrix} s'_{0,0} & s'_{0,1} & s'_{0,2} & s'_{0,3} \\ s'_{1,0} & s'_{1,1} & s'_{1,2} & s'_{1,3} \\ s'_{2,0} & s'_{2,1} & s'_{2,2} & s'_{2,3} \\ s'_{3,0} & s'_{3,1} & s'_{3,2} & s'_{3,3} \end{bmatrix}$$

87	F2	4D	97
6E	4C	90	EC
46	E7	4A	C3
A6	8C	D8	95

→

47	40	A3	4C
37	D4	70	9F
94	E4	3A	42
ED	A5	A6	BC

$$(\{02\} \cdot \{87\}) \oplus (\{03\} \cdot \{6E\}) \oplus \{46\} \oplus \{A6\} = \{47\}$$

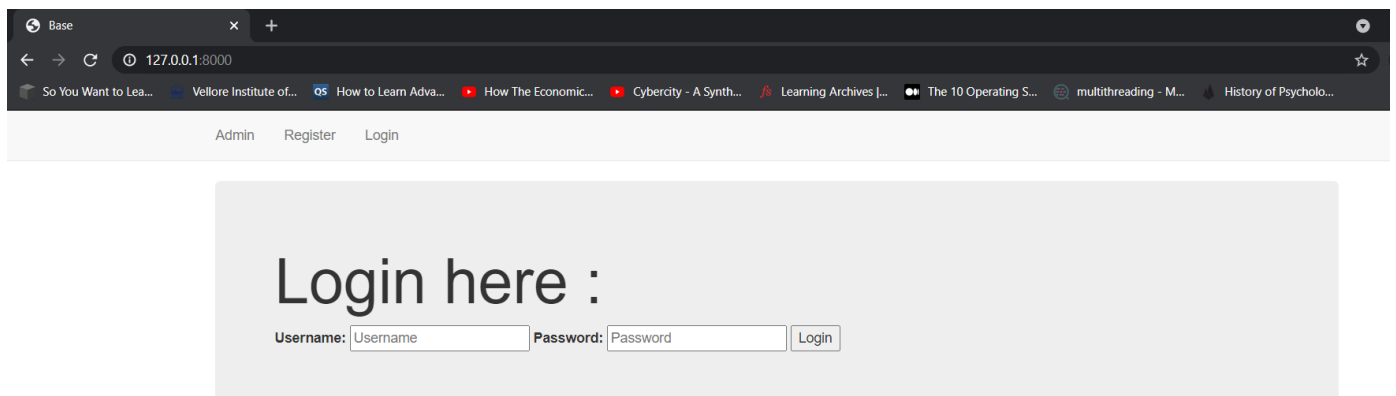
Figure 7 Matrix Multiplication



## 5.Results and Discussions

Chrome was found to be the most compatible browser of all the browsers we tested. Firefox, Edge and Chromium were close second, however firefox required some simple additional customisations to function properly.

### Login Page



Base x +

127.0.0.1:8000

So You Want to Lea... Vellore Institute of... OS How to Learn Adva... How The Economic... Cybercity - A Synth... JS Learning Archives [...] The 10 Operating S... multithreading - M... History of Psycholo...

Admin Register Login

# Login here :

Username:  Password:

*Figure 8 Login Page*

### USER INTERFACE





Figure 9 UI

### Filling Message in the message box



Figure 10 Message

## Password input for Encryption Prompt

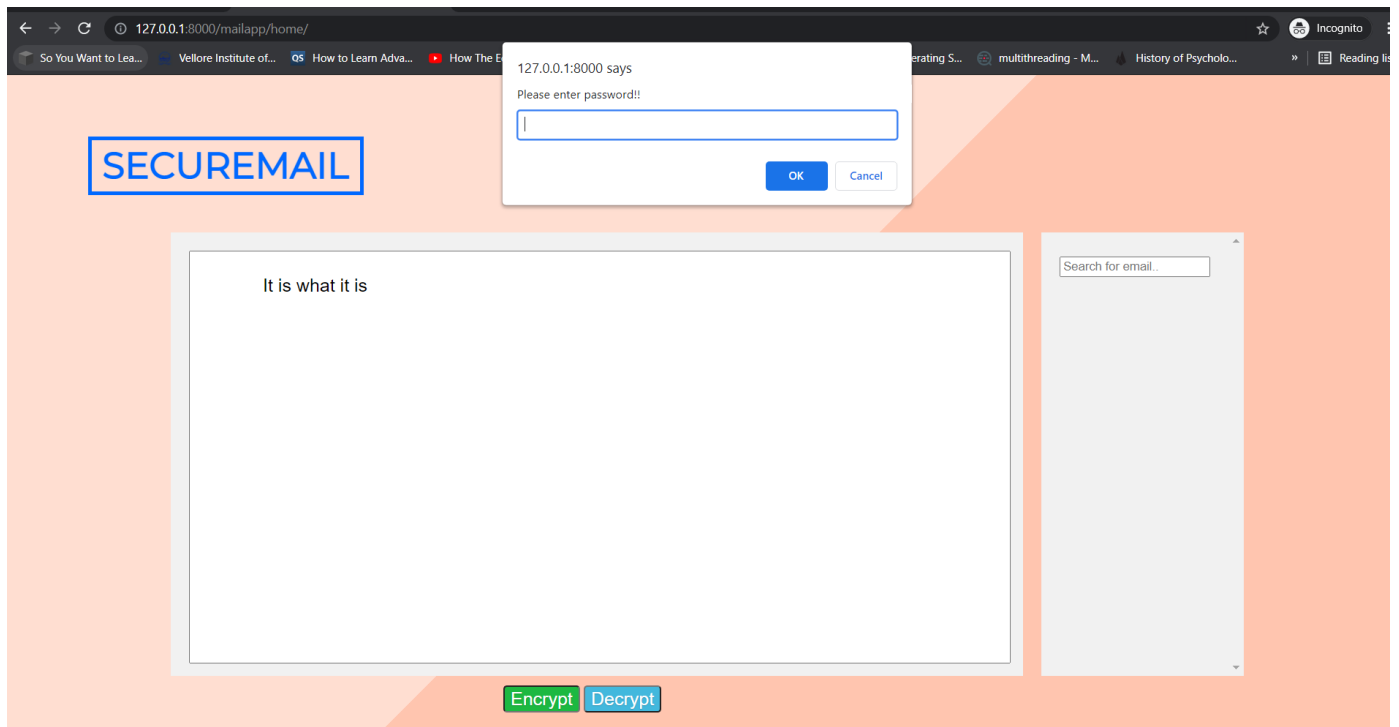


Figure 11 Password Input

## Encrypted message

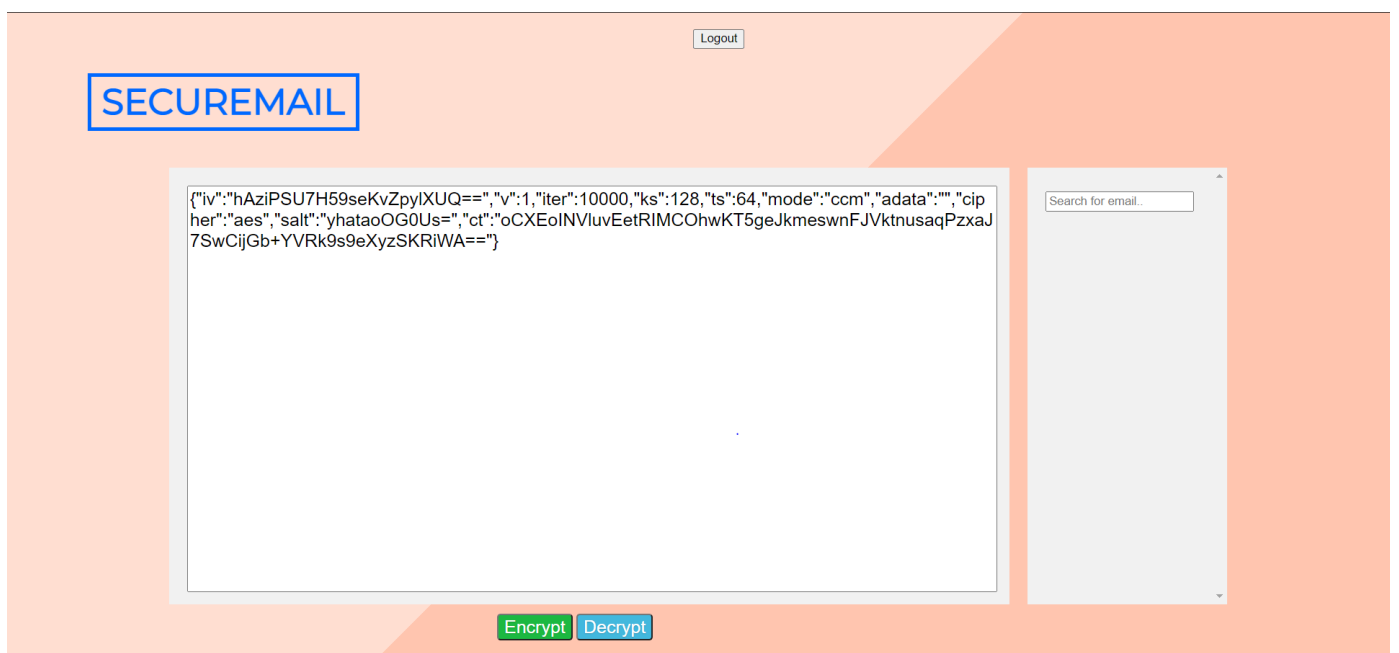


Figure 12 Encrypted Message

# The Gmail Tab opens

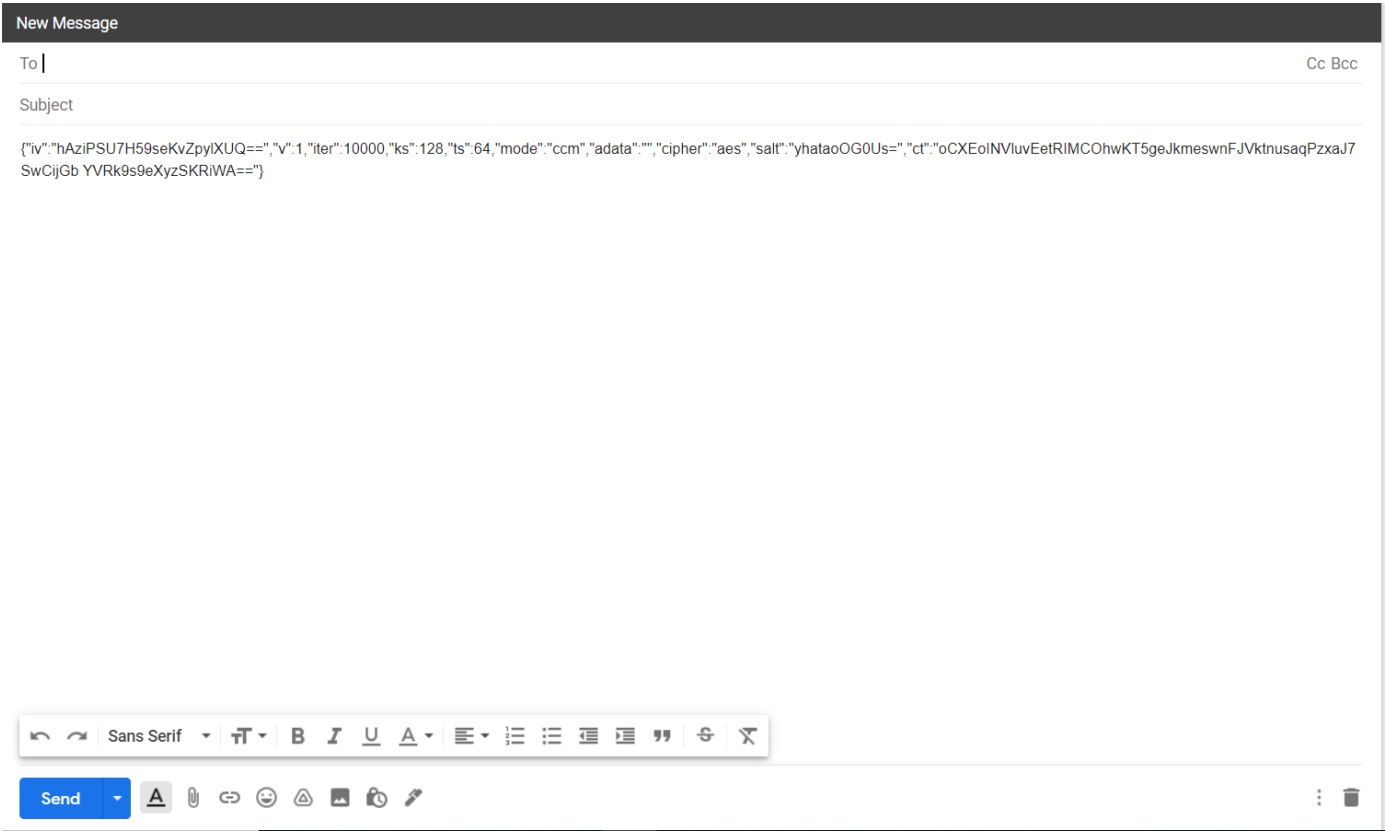


Figure 13 The Gmail Tab Opens

# Decryption

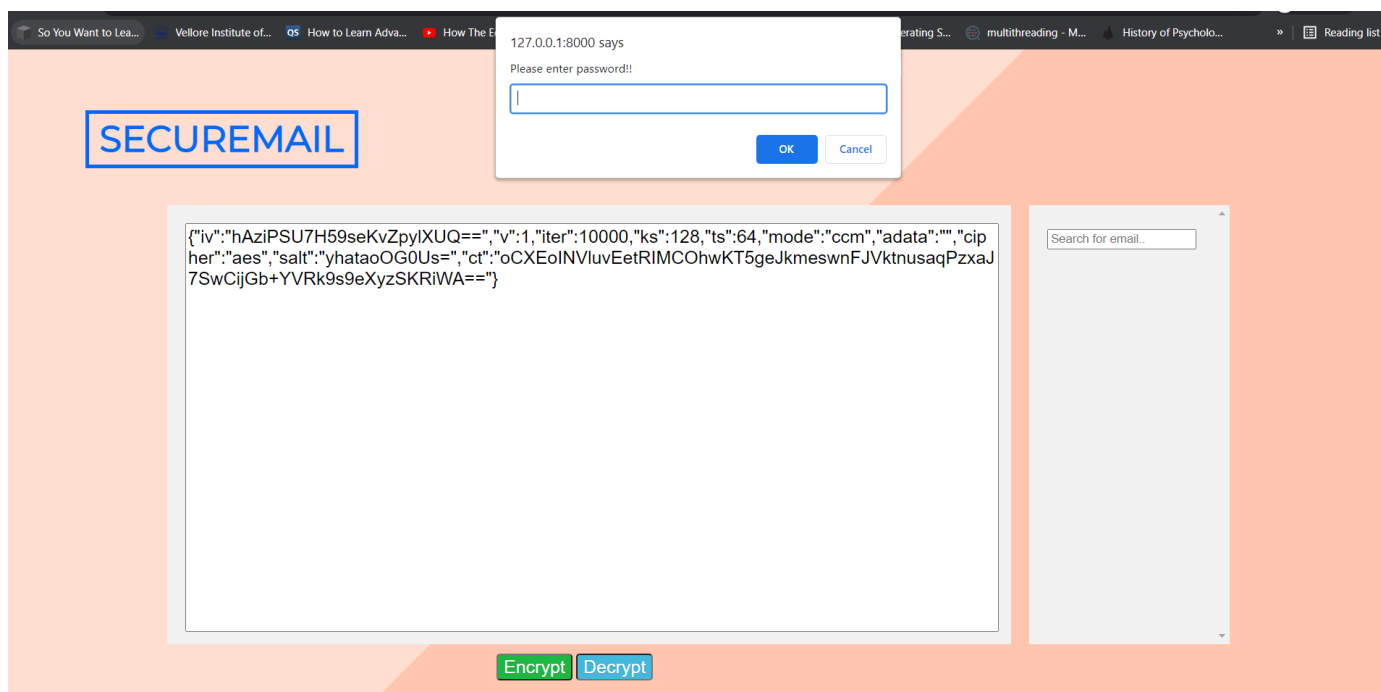


Figure 14 Password Input



Figure 15 Decrypted Message

# FOR ANALYSIS AND AUDIT

## Logging into Admin account

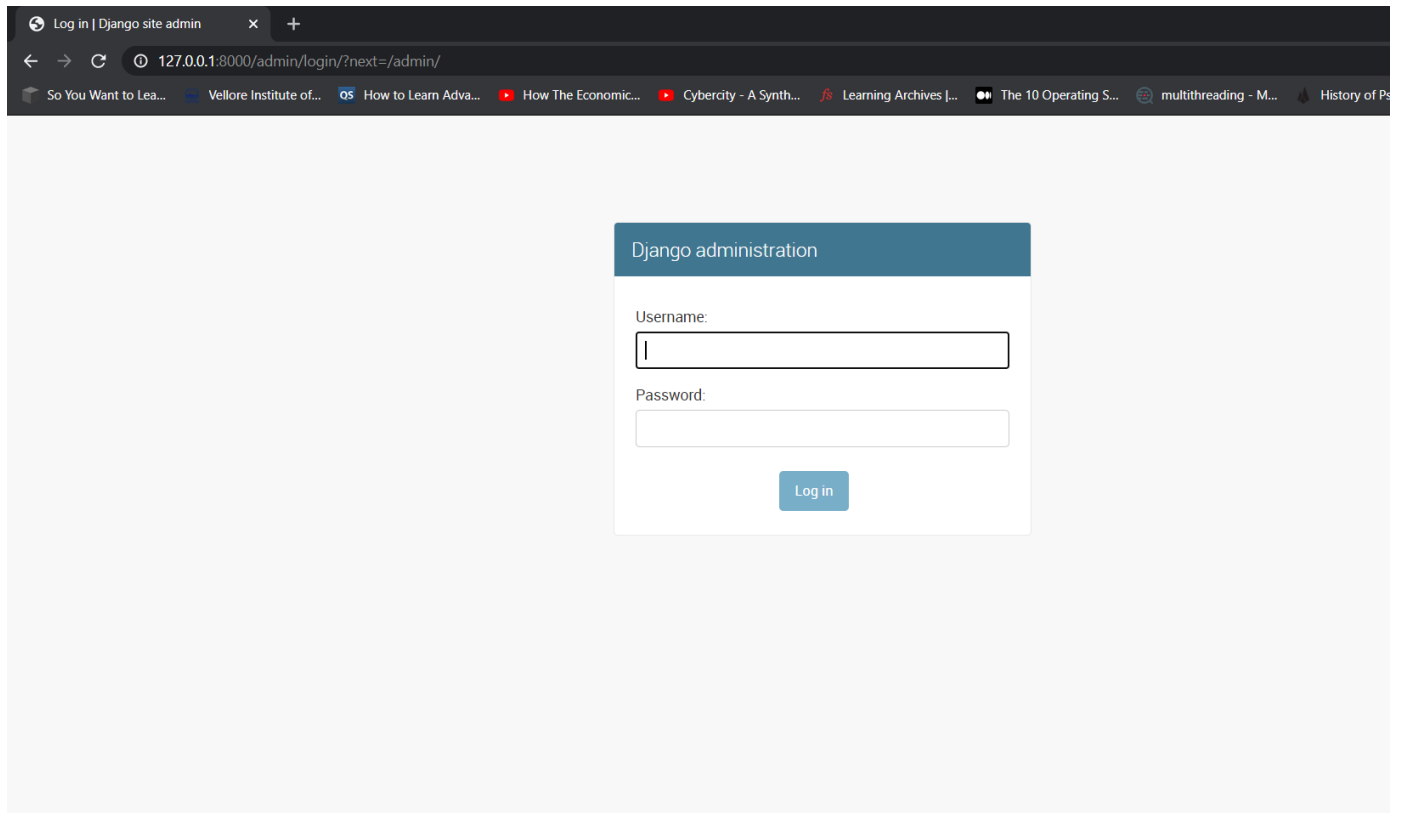


Figure 16 Admin Login

## Users data as visible to the admin

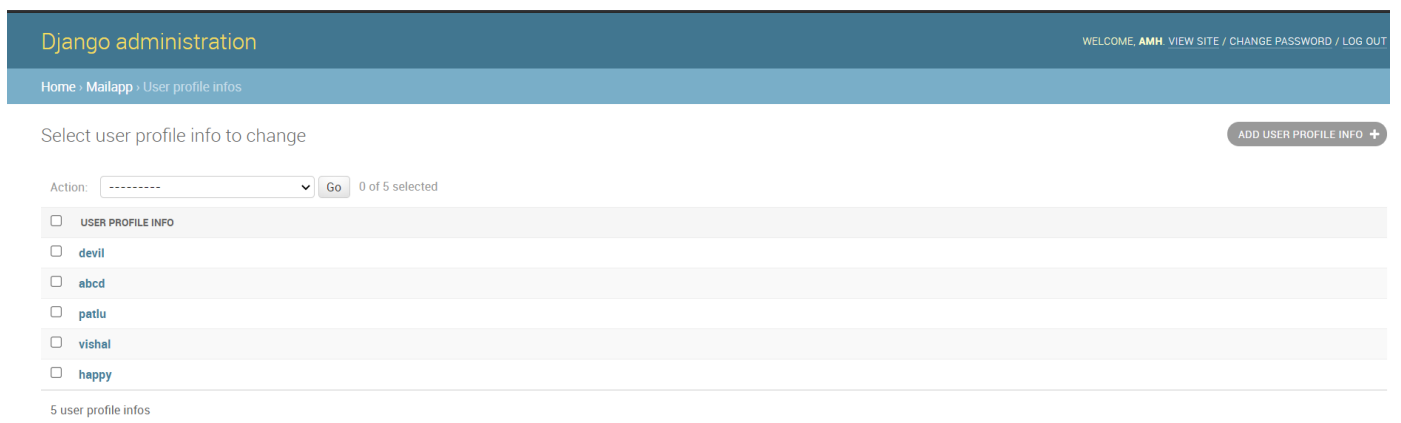


Figure 17 Users Data Visible To admin

# Message and keys as visible to the admin

Django administration

WELCOME, **AMH** / VIEW SITE / CHANGE PASSWORD / LOG OUT

Home / Mailapp / Msgs datas / ipM1tTURJ3sT2jP1ip01qA=-l@lrAIEIZCZbD8-

Change msgs data

HISTORY

Key:

ipM1tTURJ3sT2jP1ip01qA=-l@lrAIEIZCZbD8-

Value:

"  
{\\"iv\\":\\"ipM1tTURJ3sT2jP1ip01qA=-l\\",\\"iv\\":1,\\"iter\\":10000,\\"ks\\":128,\\"ts\\":64,\\"mode\\":\\"ccm\\",\\"adata\\":\\"\\",\\"cipher\\":\\"aes\\",\\"salt\\":\\"yAIEIZCZbD8-V\\",\\"ctf\\":\\"BGtfKCv+Ck2Sxyc9481SFLs4Xx8S7JSWAjOoEt  
sHlyITl+TDLf02a0yC5QsmiMu3M/GRfmAO8N6mBm9Gzzfx6sS4XHUMyUxMvT0e+IsApDzP59r8u8L\\"}"

Delete

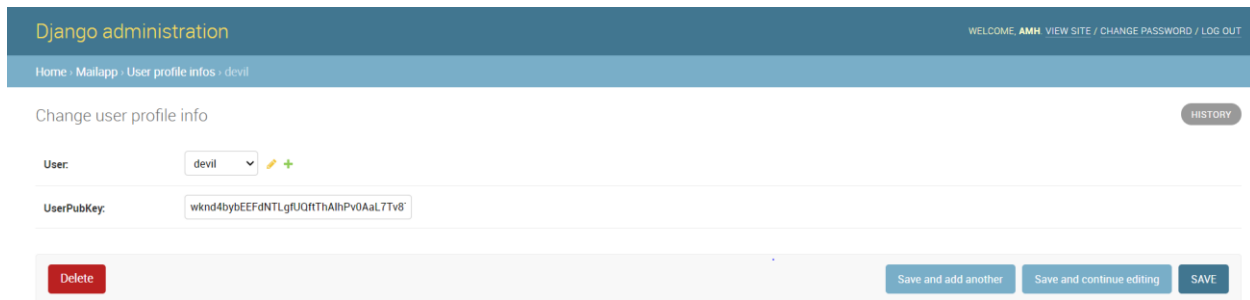
Save and add another

Save and continue editing

SAVE

Figure 18 Message and Keys Visible to Admin

## User and its public key



Django administration

WELCOME, AMH. VIEW SITE / CHANGE PASSWORD / LOG OUT

Home > Mailapp > User profile infos > devil

Change user profile info

History

User: devil

UserPubKey: wknd4bybEEFdNTLgflUQRThAihPv0AaL7TV8

Delete Save and add another Save and continue editing SAVE

Figure 19 User and Its Public Key

## 6. Conclusion

The team has built a novel service for allowing multiple layers of encryption while also attempting to counter the chief problem that email service providers cite to avoid encryption - user convenience and time consumption. Javascript is highly integrable to any web application. That has certainly helped a great way through this project.

For future works, the project could incorporate a larger scale of users on a cloud based server or possibly a hardware accelerated server, which will lead the project to deliver the same efficiency as a traditional email service.

## Bibliography

1. R. Indrayani, Subektiningsih, P. Ferdiansyah and D. A. Satria, "Effectiveness comparison of the AES and 3DES cryptography methods on email text messages," *2019 International Conference on Information and Communications Technology (ICOIACT)*, 2019, pp. 66-69, doi: 10.1109/ICOIACT46704.2019.8938579.
2. D. Fischer, B Markscheffel and K Scherr "Secure e-mail communication - Comparison and selection of encryption solutions using an utility value analysis approach"

*12th International Conference for Internet Technology and Secure Transactions, IEEE, 2018.*

3. R. Mazumder, A. Miyaji and Chunhua Su, "A simple construction of encryption for a tiny domain message," *2017 51st Annual Conference on Information Sciences and Systems (CISS)*, 2017, pp. 1-6, doi: 10.1109/CISS.2017.7926080.
4. A. P. BHATT and M. SHARMA, "E-Mail Security Framework Through Various Virus Encryption Techniques," *2019 International Conference on Intelligent Computing and Control Systems (ICCS)*, 2019, pp. 134-138, doi: 10.1109/ICCS45141.2019.9065809.
5. B. S. B. Gowda, "Implementation of Elliptic Curve Cryptography over a Server-Client network," *2020 5th International Conference on Devices, Circuits and Systems (ICDCS)*, 2020, pp. 116-119, doi: 10.1109/ICDCS48716.2020.243562
6. M Husmi, H T Ciptaningtya, W Suadi, R M Ijihadie, R Anggoro, M F Salam and S Arifiani "Security audit in cloud-based server by using encrypted data AES-256 and Sha-256" , *IOP Conference Series: Materials Science and Engineering, IOP Publishing, Scopus*
7. A. S. Rawat and M. Deshmukh, "Efficient Extended Diffie-Hellman Key Exchange Protocol," *2019 International Conference on Computing, Power and Communication Technologies (GUCON)*, 2019, pp. 447-451.
8. K. S. Charan, H. V. Nakkina and B. R. Chandavarkar, "Generation of Symmetric Key Using Randomness of Hash Function," *2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, 2020, pp. 1-7, doi: 10.1109/ICCCNT49239.2020.9225280.
9. S. Purevjav, T. Kim and H. Lee, "Email encryption using hybrid cryptosystem based on Android," *2016 18th International Conference on Advanced Communication Technology (ICACT)*, 2016, pp. 426-429, doi:



10.1109/ICACT.2016.7423418.

10. W. Bai, D. Kim, M. Namara, Y. Qian, P. G. Kelley and M. L. Mazurek, "Balancing Security and Usability in Encrypted Email," in *IEEE Internet Computing*, vol. 21, no. 3, pp. 30-38, May-June 2017, doi: 10.1109/MIC.2017.57.
11. M K Pehlivanoglu and N Duru ,Email Encryption using RC4 Algorithm by *International Journal of Computer Applications* 2015