iJRASET
International Journal For Research in
Applied Science and Engineering Technology

INTERNATIONAL JOURNAL
FOR RESEARCH
IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 9 Issue: IX Month of publication: September 2021

DOI: https://doi.org/10.22214/ijraset.2021.37973

Encrypted e-Voting System using IoT

Mahidhara Reddy Kankara[1], Maanvik Thodupunuri[2], Lakshmi Srikanth Yechuri[3], N. V. Koteswara Rao[4]
[1, 2, 3]B.E. VIII Semester, Chaitanya Bharathi Institute of Technology, Hyderabad, India
[4]Professor, Dept. of ECE, Chaitanya Bharathi Institute of Technology, Hyderabad, India

Abstract: Elections make a fundamental contribution to democratic governance but a lack of trust among citizens on their electoral system is a hindrance to satisfy the legal requirements of legislators. Even the world's largest democratic countries suffer from issues like vote rigging, election manipulation and hacking of the electronic voting machines in the current voting system. To provide data security for e-Voting systems, the advanced encryption standard (AES) algorithm has been proposed, but traditional AES gives the same ciphertext for every similar pair of key and plaintext. So, to eliminate these disadvantages, AES in Galois-counter mode (GCM) has been used to obtain different ciphertexts all the time by using Initialization Vector. The fingerprint data from each user is verified using Internet of Things (IoT) based Biometric system which also helps to avoid Plural Voting. The whole data is encrypted and stored in the cloud, and it can be decrypted by authorized personnel to obtain the final vote count. So, the proposed model will enhance transparency and maintain anonymity of the voters alongside providing an easily accessible secured voting system.

Keywords: Advanced encryption standard, initialization vector, additional authenticated data, galois-counter mode, biometrics, security, ciphertext, authtag

I. INTRODUCTION

Election is the symbol of democracy and people have the complete freedom to select a visionary leader. The process of election involves people voting for a candidate and the candidate who obtains a major share of votes is declared to lead the people. Paper ballots have been the most common form of voting, but there were cases of vote rigging and plural voting in this method. About 10% of the countries have implemented electronic voting machines (EVM), but it is found out that the EVM's are prone to tampering [6]. So, to secure the data generated from the voting process, the Data encryption standard (DES) algorithm, with a 56-bit key length, was used to encrypt the data. But, with the rapid growth in technology and computational resources, it was confirmed that the DES algorithm is no longer reliable to handle immensely valuable data that is used in the voting process. The successor algorithm for DES, the Advanced encryption standard (AES) algorithm ensures additional layers of security, and thus, the algorithm is preferred for encrypting the data. It is a symmetric block cipher that takes plain text in blocks and converts them into ciphertext. It is found that AES is at least six times faster than triple DES and allows choosing a key of variable length (up to 256-bits) as compared to a fixed key length of 56-bits in DES, thus making it highly robust.

The issue with using a traditional AES algorithm is that it produces the same output for a given pair of input and secret key. This is an undesirable feature, especially when voting is limited to a finite number of parties and it is highly likely that the same party can be voted multiple times. For example, if 10 persons have casted their vote for "Party-1", the input plaintext is the same for all these 10 cases. As the key also remains the same, the output of traditional AES algorithm is the same. It becomes easy for an intruder to analyze ciphertexts and any repetition of certain text in the ciphertexts can lead to attacks. So, it is found necessary to obtain different ciphertexts even if the pair of plaintext and secretkey remain the same. AES algorithm operating in Galois-Counter mode (GCM) has been proved to be suitable in this context. It provides authenticated encryption of data. It also supports pipelining and parallelism which makes it a suitable choice for using in the e-Voting process.

II. LITERATURE SURVEY

Alex Halderman et al. [1] has stated that the electronic voting machines lack the security features and are prone to serious attacks which can be potentially difficult to trace. The main drawback of EVM is that it does not provide information as of when the tampering is done or if the EVM has been tampered or not. So, the paper states the need for encryption which can ensure proper security of data in the EVM's. Also, the need for regular software updates for EVM's is stated.

Shilpa C Venugopal et al. [2] has proposed the use of a customer identification number along with biometrics to validate the voters. The biometrics are stored in the database against the identification number during enrollment process. During verification, the user's biometrics are matched with the existing template in the database. The results of election are stored in the cloud using IoT.

Lalit Kumar Gupta et al. [3] has proposed that the current DES encryption algorithm that is being used for encrypting data generated by the election process has become more vulnerable because of its smaller degree of encryption.

As the decryption is quite easy, it is stated that an advanced algorithm is required for encrypting the data to make it stronger to decrypt and AES is proposed to eliminate these challenges.

David A. McGrew et al. [4] has stated that GCM is a robust mode that provides faster outputs with higher degree of security. An IV (Initialization vector) is used as an input alongside Secretkey and Plaintext. The IV is varied for each encryption which has the same secretkey, thus ensuring different outputs all the time. It is also stated that GCM provides data authentication in addition to data encryption.

As seen in the literature survey, some techniques provided security in hardware using fingerprint modules while others have implemented encryption methods for software security. It is found necessary to combine both security in the hardware and an improvement of security in software. This has motivated us to build a model that provides security in both, thus eliminating the drawbacks found in other techniques.

III. TOOLS AND TECHNOLOGIES

In this implementation, a fingerprint module and AES algorithm in GCM mode have been used.

A. Fingerprint Module

Fingerprint verification is one of the well-known and safer techniques to distinctly identify each individual. Every person's fingerprint contains streaks, which are the lines in contact with a surface and valleys, which are the spaces between two streaks. An R305 fingerprint module has been used in implementing the voting system. A Raspberry Pi microcontroller, an SD card and a UART (Universal asynchronous receiver-transmitter) converter are the other components used for the implementation.

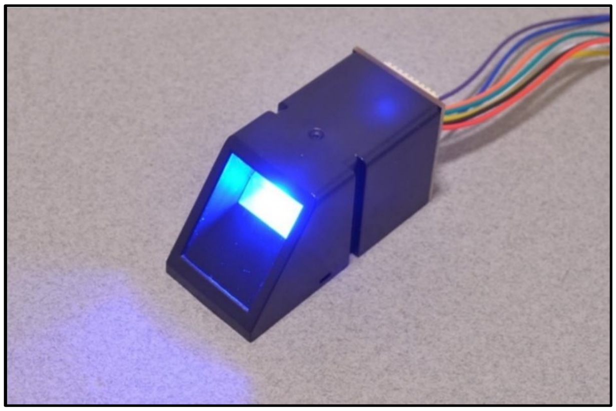


Fig. 1: An image of Fingerprint module

For creating a template during the enrollment process, the user's fingerprint image goes through some image processing techniques like Normalization, Segmentation, Threshold Method, Binarization, Enhancement and Thinning. The created template is stored in a database.

B. AES Algorithm

The Advanced Encryption Standard (AES) is a symmetric block cipher encryption algorithm chosen for securing the data generated in the implementation. The size of input data, also referred to as PlainText is 128-bit, with a SecretKey (or simply key) of variable length. For our implementation, we have chosen the length of secretkey as 256-bits.

Each round contains four steps. They are :

1) Substitute bytes
2) Shift rows (circular left shift)
3) Mix columns
4) Add round key

Initially, the AES algorithm's operations are performed on a 2D array of bytes called the State array, with 4 rows and 4 columns each. At first, the array of bytes from the plaintext is copied into the state matrix. The Secret key undergoes a key schedule process in which the round keys are obtained from the secret key (using Rijndael's key scheduling). A total of 60 words (15 keys) are generated from this process, where 4 words are used in each round. Since the length of secret key used is 256 bit, the algorithm involves 14 rounds, hence 56 words are utilized in these rounds. The remaining 4 words are used in the initial Add round key step.

Before starting the first round, the state array goes through add round step where each column in the state array is XORed with the first 4 words generated from the Rijndael's Key schedule. Now the first round begins and the operations on state array are as follows:

a) Substitution Bytes: It is a non-linear step where the data in each cell is replaced with the data from a lookup table. The lookup table is a 16 x 16 array, indexed by hexadecimal bits [5]. The first 4 bits represent a row, while the last 4 bits represent the column to be looked in the table.

b) Shift Rows: The process involves shifting the rows based on the row number. For example, the zeroth row involves no shifts, first row involves 1 shift, and so on. This is done to prevent the columns in the State Matrix form becoming linearly independent.

c) Mix Columns: Every column from the state matrix is multiplied with the column from a pre-defined matrix. The result is stored in the intermediate array.

4) Add Round Key: The output matrix from the mix columns is XORed with round key. Then the output is given to next round as intermediate array.

The four steps are performed in all the rounds, except the final round. For the final round, the mix columns step is ignored.
The resulting State Matrix is output as the cipher text.

C. GCM (Galois Counter mode)

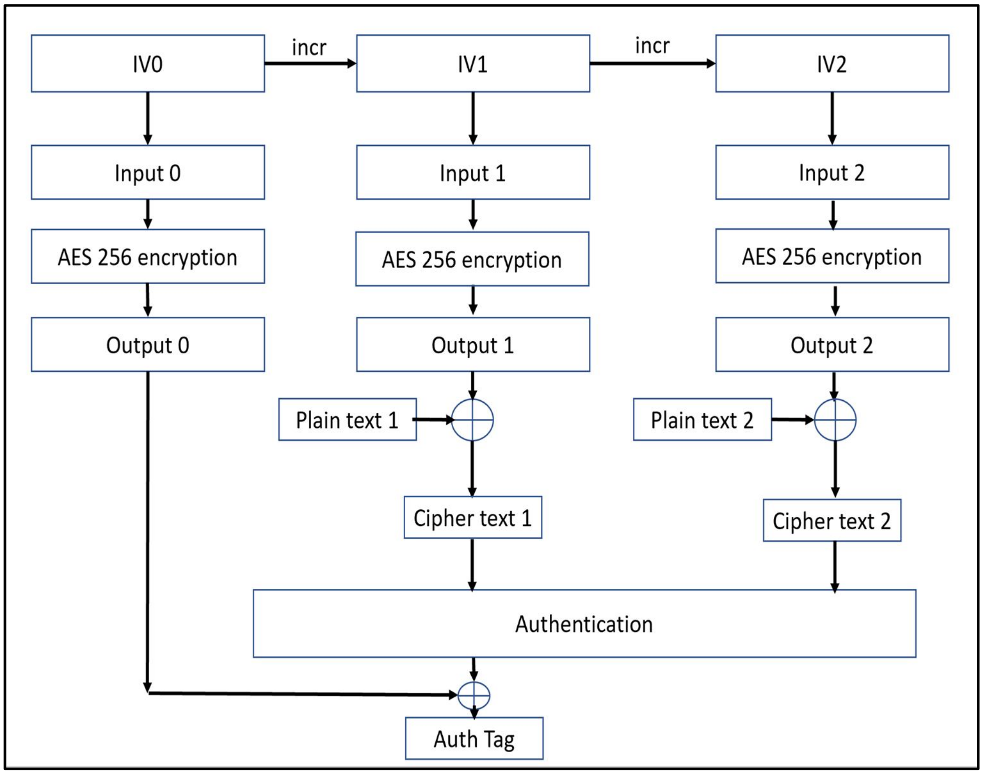


Fig. 2: Flow diagram of AES in GCM mode

In traditional AES, the pair of secretkey and plaintext are used to encrypt the data. The secretkey is to be known to both the sender and receiver in order to encrypt and decrypt the information. If the same plaintext is given as an input while keeping the secretkey constant, the output ciphertext will be the same. This causes issues because the plaintext is nothing but the vote preference, which is highly likely to repeat more than once. For example, if 'n' users vote for "Party-1", the plaintext of these 'n' votes will remain the same, i.e. "Party-1". Now with the same secretkey being used throughout the election process, the same ciphertext is obtained 'n' times. This drawback can allow intruders to compare and analyze the ciphertext and look for certain repetitions in text patterns, find out the secretkey used for the election process and manipulate the votes.

To eliminate this drawback and use multiple blocks of input for encryption, AES in GCM mode has been used in our implementation. GCM provides authenticated encryption of data and is much reliable than the previously used CBC and CFB modes. CBC mode is prone to padding oracle attacks and doesn’t allow parallel encryption of data, which is a performance issue. GCM is resistant to frequency attacks, which is the main reason for using this mode of operation for the implementation.

GCM involves counter mode along with the calculation of Authentication tag during encryption. It is recommended that an IV of length 96 bits is to be used for faster operation, along with a counter of 32-bit length.

1) *Inputs:* SecretKey, IV, Plaintext, AAD
2) *Outputs:* CipherText, AuthTag

The length of ciphertext is equal to the length of plaintext.

The flow diagram of GCM is given in Figure 2. Here, AAD (additional authenticated data) is used to prevent the attacker to analyze the decrypted results when a set of carefully chosen ciphertexts are submitted. AAD helps to recognize and refuse to decrypt any improperly constructed ciphertexts and only allows decryption if a proper encryption of plaintext is done. The IV (Initialization Vector), sometimes referred to as Nonce, is to be made distinct for every invocation of the algorithm which uses the same key so that a different output is generated all the time. So, the implementation uses a sequential IV generation. The data needed to form the IV has to be known to both the ends and hence it is sent along with the ciphertext.

The AuthTag obtained after encryption is sent along with the ciphertext. This is used to compare with the AuthTag obtained after decryption. Only if these tags match, the output of decryption is valid. The plain text is firstly divided into block of 128-bin length. As the IV needs to be different for each block, it is incremented each time. This IV along with counter number is given to AES encryption function. The output generated from AES is XORed with the plain text (except initial block) to obtain blocks of cipher text. These cipher texts undergo authentication process (AAD is used here) and the output obtained is XORed with the initial output block to achieve the Authentication Tag.

IV. METHODOLOGY

The e-voting process consists of 4 major steps: voter enrollment, connecting Raspberry Pi to cloud, verification process and vote encryption. A Raspberry Pi Model 3 B+ has been used in the implementation. It has four USB ports and a built-in Wi-Fi module.

A. Interfacing Diagram

The fingerprint module is connected to the USB and then the USB connector is connected to one of the ports of Raspberry Pi as shown in Figure 3.

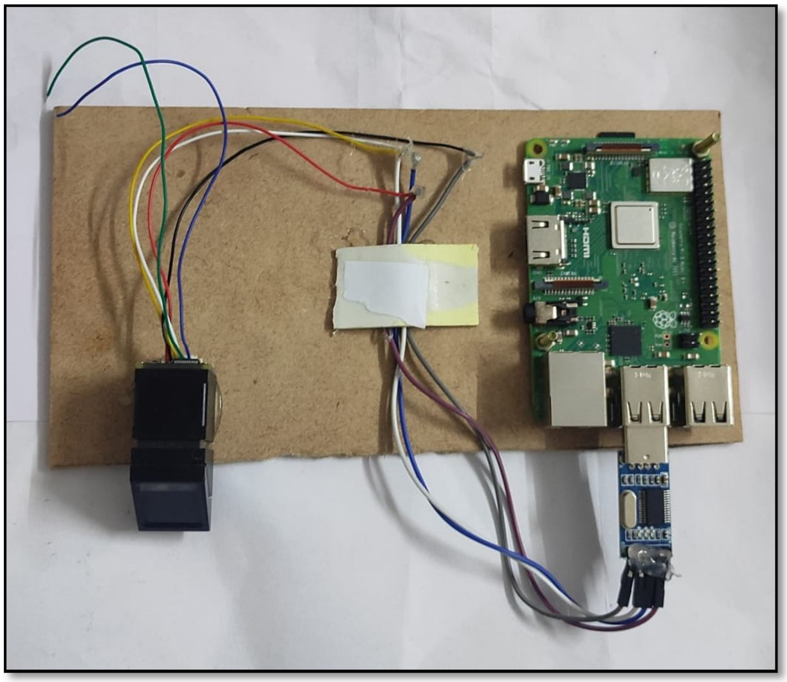


Fig. 3: Interfacing diagram of Raspberry Pi with Fingerprint module

B. Voter Enrollment

Enrollment relates to the process of registering the fingerprints of a voter against their other demographic data as a record of their biometric identity. The enrollment process involves recording the templates against an identity number. The steps involved in the enrollment process are:

1) Firstly, the user is asked to enter an identity number.
2) The user is then prompted to present a biometric template on a scanner (R305) that captures the images.
3) The enrolment system asks the user to present their biometric twice to ensure that the quality of image captured is good for verification.
4) After capturing the image, the input fingerprint goes through image processing techniques like normalization, segmentation, binarization, enhancement and thinning. The final template is derived from the captured images.
5) The template along with the raw image is stored in the biometric server against the customer identity number for later retrieval and verification.

C. Connecting Raspberry Pi to Google Cloud

A project is created in Google cloud and Google Drive API and Google sheets API are enabled. A JSON file gets downloaded which can be used to communicate with the cloud from a remote machine.

D. Verification

The user’s identity number is verified by the system. If a record is found, the system prompts the user to present a live biometric on the scanner. This live biometric is then compared with the biometric template that is stored against the identity number in the biometric server [7] (this process of comparing the two templates is referred to as minutiae matching). In case the verification is successful, the user is prompted to cast a vote, otherwise the user is asked to restart the process [8]. After the verification process, the vote casted is to be encrypted and stored in the cloud.

E. Encryption

The vote casted by the user is given to the AES-GCM encryption function as a plaintext, along with the IV generated and the function outputs ciphertext and AuthTag. These outputs along with IV are stored in the cloud, which can be used for further process. The AAD is used to authenticate the data and in the final step, the AuthTag is generated. This AuthTag can be used by the decryption function to validate the ciphertext. The length of AuthTag is 128 bits and the length of ciphertext is same as of plaintext. The length of IV is fixed at 96 bits.

The output encrypted message looks like:

```
Message = ‘PARTY2’
Secretkey = b'\x9a\x8e\xef4\xfc<\x9bN\xaeR\xf5\xd8\x7f\xc8\x80\xf6b5@l~\x16\xe9\x0e\xe8\xceD\xe6\xa86u\x1c'
Encrypted_Msg = {'ciphertext': b'19b03e6d3580', 'aesIV': b'4e011b5e9b2617dd8ce79c5b3f70a880', 'authTag':
b'70e519352c2ed0253b43b0c93300b6e4'}
Decrypted_Msg =   'PARTY2'
```

F. Decryption

The process of decryption uses the same key schedule. The whole process of encrypting the data is followed in an inverse order in order to decrypt the data. Before the process begins, the AAD is checked for authentication purpose. If the authentication check gets failed, the decrypt operation halts, and plaintext is discarded instead of processing it further.

Firstly, the encrypted data from cloud is given as an input to the AES-GCM decryption function.. The function returns the output, which is the plain text.

G. Final vote count

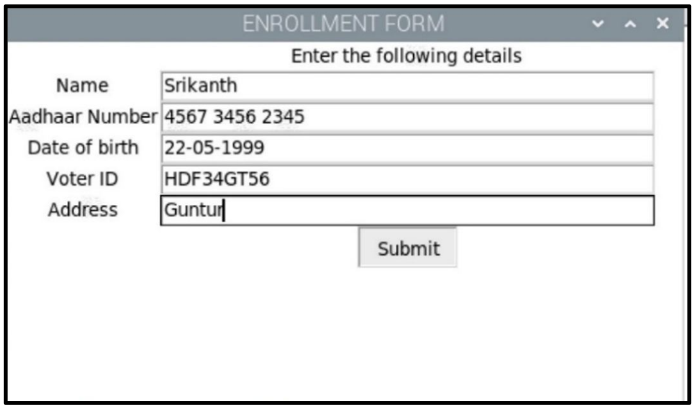
When the authorized personnel want to declare the election result, the encrypted information is collected and decrypted [9]. The input from the Google Spreadsheet is given to a decryption function and the decrypted data is read for counting the number of votes. A dictionary object keeps count of votes and the party with majority votes is displayed.

V. RESULTS AND DISCUSSIONS

The e-Voting process begins with enrollment, where user details are collected and stored in a database.

A. Enrolment Process

The user places the finger on the fingerprint sensor and the system captures the details. The template then gets stored in the database. A GUI for enrollment form appears as shown in Figure 4. The user fills the details and submits the form.



ENROLLMENT FORM

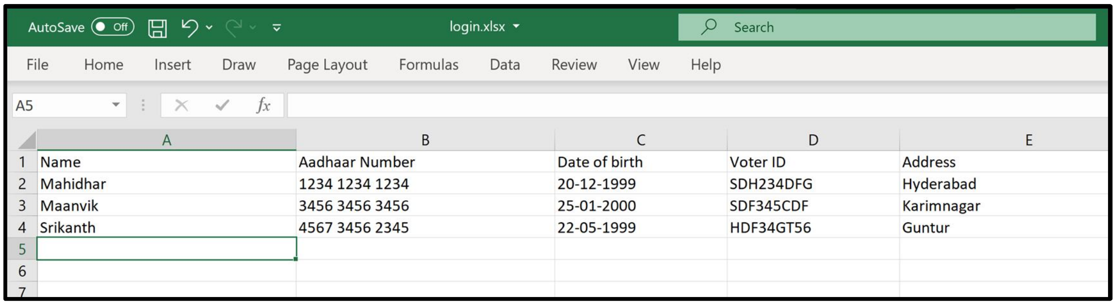
Enter the following details

Name	Srikanth
Aadhaar Number	4567 3456 2345
Date of birth	22-05-1999
Voter ID	HDF34GT56
Address	Guntur

Fig. 4: Voter Enrollment Form

B. Login sheet

The “login.xlsx” sheet gets created and updated with the details (as shown in Fig. 5) that the user enters in the enrollment form.

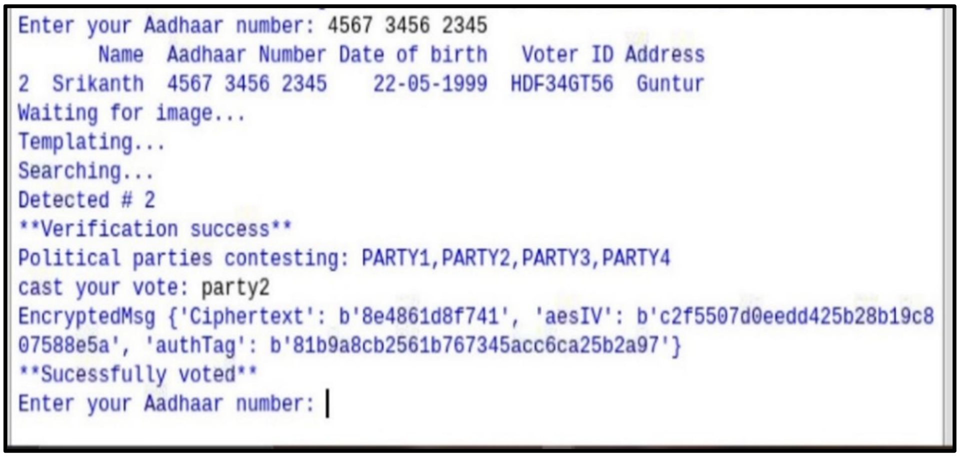


	A	B	C	D	E
1	Name	Aadhaar Number	Date of birth	Voter ID	Address
2	Mahidhar	1234 1234 1234	20-12-1999	SDH234DFG	Hyderabad
3	Maanvik	3456 3456 3456	25-01-2000	SDF345CDF	Karimnagar
4	Srikanth	4567 3456 2345	22-05-1999	HDF34GT56	Guntur
5					
6					
7					

Fig. 5: Login Details

C. Verification

During the verification process, the user is prompted to enter identification number. If the number is found in the login.xlsx file, the user will be asked to verify fingerprints. A fingerprint template is collected and matched to the template that is found in the database. On successful verification, the user is asked to cast a vote (as shown in Fig. 6). If the user enters an invalid identity number or fingerprint, he will be re-prompted.



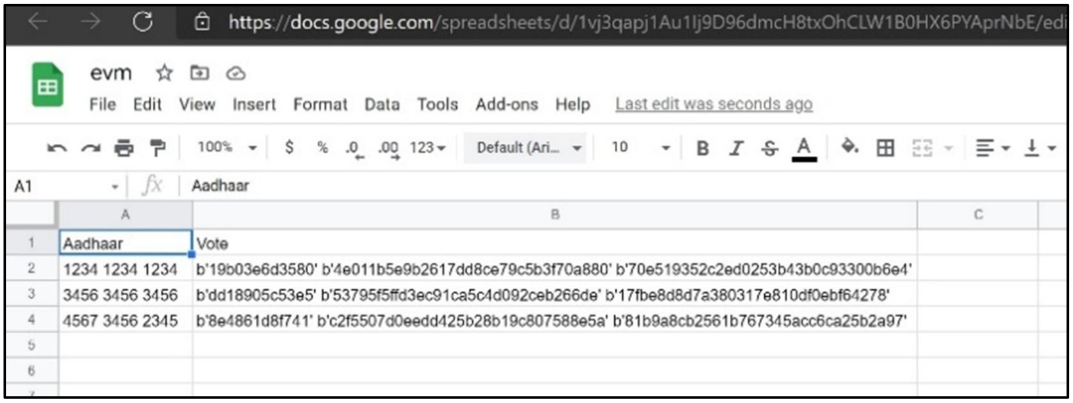
```
Enter your Aadhaar number: 4567 3456 2345
       Name  Aadhaar Number Date of birth   Voter ID Address
2  Srikanth  4567 3456 2345    22-05-1999  HDF34GT56  Guntur
Waiting for image...
Templating...
Searching...
Detected # 2
**Verification success**
Political parties contesting: PARTY1,PARTY2,PARTY3,PARTY4
cast your vote: party2
EncryptedMsg {'Ciphertext': b'8e4861d8f741', 'aesIV': b'c2f5507d0eedd425b28b19c8
07588e5a', 'authTag': b'81b9a8cb2561b767345acc6ca25b2a97'}
**Sucessfully voted**
Enter your Aadhaar number: 
```

Fig. 6: Verification and casting a vote

D. Storing Data in the Cloud

After casting the vote, the data is encrypted and is stored in the cloud. For this process, the “edit.json” file is used to provide credentials and details of the google cloud account. The aadhaar number and encrypted vote are stored in the google spreadsheet (as shown in Fig. 7).

As seen in the figure 7, the ciphertexts obtained for the same pair of secretkey and plaintext are different, which is not in the case of traditional AES. Also, GCM is faster than CBC mode because it supports pipelining and parallelism (as shown in Fig. 9). It also withstands many attacks that CBC cannot.


https://docs.google.com/spreadsheets/d/1vj3qapj1Au1lj9D96dmcH8txOhCLW1B0HX6PYAprNbE/edi
evm
File
Edit
View
Insert
Format
Data
Tools
Add-ons
Help
Last edit was seconds ago
100%
$
%
.0
.00
123
Default (Ari...
10
B
I
A
A1
fx
Aadhaar

	A	B	C
1	Aadhaar	Vote	
2	1234 1234 1234	b'19b03e6d3580' b'4e011b5e9b2617dd8ce79c5b3f70a880' b'70e519352c2ed0253b43b0c93300b6e4'	
3	3456 3456 3456	b'dd18905c53e5' b'53795f5ffd3ec91ca5c4d092ceb266de' b'17fbe8d8d7a380317e810df0ebf64278'	
4	4567 3456 2345	b'8e4861d8f741' b'c2f5507d0eedd425b28b19c807588e5a' b'81b9a8cb2561b767345acc6ca25b2a97'	
5			
6			

Fig. 7: Spreadsheet containing encrypted vote information

As shown in the Figure 8, AES in GCM mode has a very high degree of randomness as compared to the traditional AES algorithm. Figure 9 compares the time consumed for encrypting 256 bits of data between AES in CBC mode and AES in GCM mode. It is clear that GCM takes lesser time for both encryption and decryption of data.

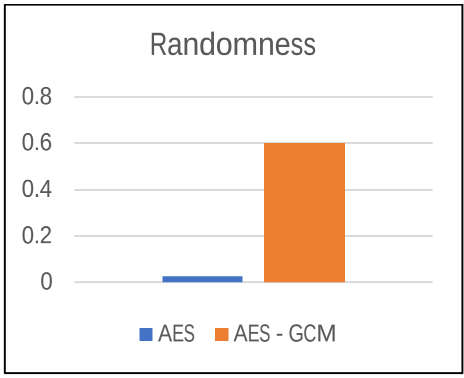


Fig. 8: Randomness of ciphertext for same input

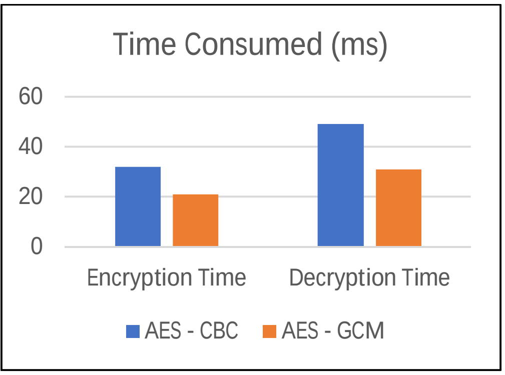


Fig. 9: Time consumption for encryption and decryption

The comparison of encryption time for both the modes against variable data sizes is shown in the figure 10, and it is clear that AES in GCM mode has performed better at all the data sizes ranging from 0.5 MB to 2 MB.

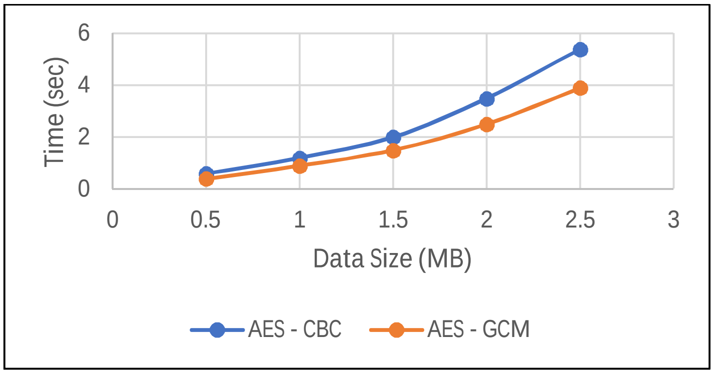
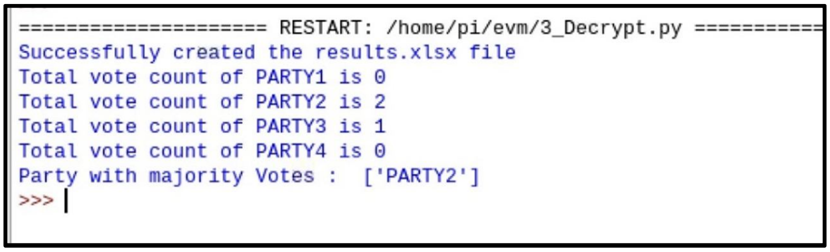


Fig. 10: Time consumed for encryption of data for different data sizes

E. Final Vote Count

The vote count for each party is displayed and the winner is declared as shown in Figure 11.

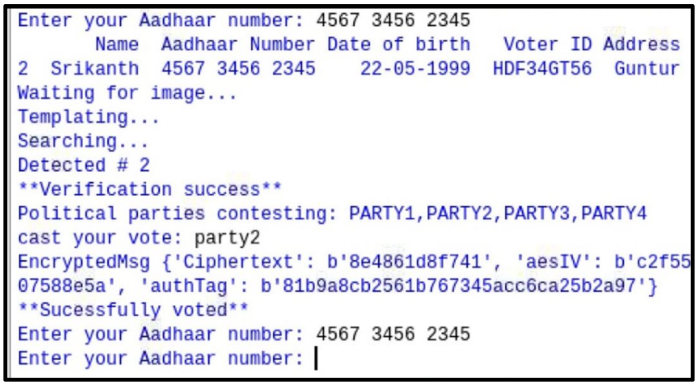


```
==================== RESTART: /home/pi/evm/3_Decrypt.py ==========
Successfully created the results.xlsx file
Total vote count of PARTY1 is 0
Total vote count of PARTY2 is 2
Total vote count of PARTY3 is 1
Total vote count of PARTY4 is 0
Party with majority Votes :  ['PARTY2']
>>> |
```

Fig. 11: Final vote count

F. Plural Voting

If the same person tries to vote multiple times, the system doesn’t accept a second vote and prompts to enter a valid aadhaar number. As shown in the figure 12, the user with ID “4567 3456 2345” has already voted. He has again entered the same number and the system simply re-prompted to enter a valid number.



```
Enter your Aadhaar number: 4567 3456 2345
       Name  Aadhaar Number Date of birth   Voter ID Address
2  Srikanth  4567 3456 2345    22-05-1999  HDF34GT56  Guntur
Waiting for image...
Templating...
Searching...
Detected # 2
**Verification success**
Political parties contesting: PARTY1,PARTY2,PARTY3,PARTY4
cast your vote: party2
EncryptedMsg {'Ciphertext': b'8e4861d8f741', 'aesIV': b'c2f55
07588e5a', 'authTag': b'81b9a8cb2561b767345acc6ca25b2a97'}
**Sucessfully voted**
Enter your Aadhaar number: 4567 3456 2345
Enter your Aadhaar number: |
```

Fig. 12: An instance of Plural Voting

VI. CONCLUSION

A secured e-Voting system has been developed using encryption and incorporating Internet of Things. A Raspberry Pi has been interfaced with a fingerprint module for collecting user details during enrollment. The fingerprint templates are stored against aadhaar number in the database and this template is used to compare with the user’s fingerprint during the verification process. The casted vote is encrypted using the AES algorithm in GCM mode and is stored in the cloud. Authorized personnel can decrypt the encrypted data to obtain the final vote count. The proposed model achieves a higher level of security, faster vote encryption and decryption and also represses plural voting, and hence it can be used for e-Voting processes.

REFERENCES

[1] J. Alex Halderman, Hari K. Prasad and Rop Gonggrijp, “Security Analysis of India’s Electronic Voting Machines”, 17th ACM conference on Computer and communications security, pp. 1-14, October 2010.

[2] Shilpa C Venugopal, Reshmi K Rajan, "Iot Based Voting Machine with Fingerprint Verification ", International Journal of Applied Engineering Research, Vol. 15, pp. 97-102, 2020.

[3] Lalit Kumar Gupta, Utkarsh Tiwari, Ajay Kumar, Soumya Jaiswal, “AES Based Online Voting System”, International Journal of computer sciences and engineering, Issue. 3, Vol. 7, pp. 915-918, March 2019.

[4] D. McGrew and J. Viega, “The Galois/counter mode of operation (GCM)”, Submission to NIST Modes of Operation Process, vol. 20, 2004.

[5] M. O. Yinyeh, David Sanka Laar, "Implementation of Advance Encryption Standard (AES) in Biometric Electronic Voting Software", Communications on applied electronics, Vol 4, No. 3, pp. 32-36, Jan 2016.

[6] Mayur Patil, Vijay Pimplodkar, Anuja R. Zade, Vinit Vibhute, Ratnakar Ghadge, “A Survey on Voting System Techniques”, International Journal of Advanced Research in Computer Science and Software Engineering, Volume 3, Issue 1, January 2013.

[7] V. K. Yadav, S. Batham, M. Jain, S. Sharma, "An approach to Electronic Voting System using UIDAI," *2014 International Conference on Electronics and Communication Systems (ICECS)*, pp. 1-4, 2014.

[8] J. Deepika, S. Kalaiselvi, S. Mahalakshmi, S. A. Shifani, "Smart electronic voting system based on biometrie identification-survey," *2017 Third International Conference on Science Technology Engineering & Management (ICONSTEM)*, pp. 939-942, 2017.

[9] K. Hasta, A. Date, A. Shrivastava, P. Jhade, S. N. Shelke, "Fingerprint Based Secured Voting", *2019 International Conference on Advances in Computing, Communication and Control*, pp. 1-6, 2019.