Journal of Information Security and Applications 61 (2021) 102908

Contents lists available at ScienceDirect

Journal of Information Security and Applications

journal homepage: www.elsevier.com/locate/jisa

Secure mobile internet voting system using biometric authentication and wavelet based AES

Ajish S. [a,*], K.S. AnilKumar [b]

[a] Department of Future Studies(Research Centre), University of Kerala, Kariyavattom Campus, Thiruvananthapuram, Kerala, India
[b] K S M D B College, Sasthamcotta, Kollam, Kerala, India

ARTICLE INFO

Keywords:
Internet voting
Fingerprint template
Iris code
AES encryption
Wavelet based AES encryption

ABSTRACT

The number of mobile phone users increases daily, and mobile devices are used for various applications like banking, e-commerce, social media, internet voting, e-mails, etc. This paper presents a secure mobile internet voting system in which a biometric method authenticates the voter. The biometric image can either be encrypted at the mobile device and send to the server or process the biometric image at the mobile device to generate the biometric template and send it to the server. The implementation of biometrics on mobile devices usually requires simplifying the algorithm to adapt to the relatively small CPU processing power and battery charge. This paper proposes a wavelet-based AES algorithm to speed up the encryption process and reduce the mobile device's CPU utilization. The experimental analysis of three methods(AES encryption, wavelet-based AES encryption, and biometric template generation) exhibits that wavelet-based AES encryption is much better than AES encryption and template generation. The security analysis of three methods shows that AES and wavelet-based AES encryption provides better security than the biometric template's protection. The study of the proposed internet voting system shows that biometric authentication defeats almost all the mobile-based threats.

1. Introduction

Mobile devices have an essential role in the daily activities of the human being. The mobile phone is the commonly used device to make calls and for mobile internet services among mobile devices. Mobile devices are used for various applications like banking, e-commerce, social media, internet voting, e-mails, etc.

The mobile devices are frequently connected to the internet using Wi-Fi or LTE, or other broadband connections [1]. The total mobile phone users in India are over 500 million, and over 77% of Indians are now accessing mobile internet. The number of mobile users is booming due to the cheap rate of mobile phones and mobile internet access. Almost all mobile phone users doesn't know the security threats. The mobile internet contains many risks, and appropriate countermeasures should be taken to overcome the threats [2].

This paper proposes a secure wavelet-based AES internet voting(i-voting) system, which uses the QR code and the biometric authentication of the Aadhaar National ID card of India [3]. The QR code of the Aadhaar card is used for reading the details of the voter like UID, Name, Date of Birth, Address, Gender, etc. The voters are authenticated using the fingerprint or iris biometric authentication method provided by India's Unique Identification Authority (UIDAI) [3]. After the voter's successful authentication, the appropriate electoral district's candidate list is sent to the voter for the vote cast. The i-vote is encrypted using the server's public key and digitally signed using the voter's private key, and transmitted to the server.

The implementation of biometrics on mobile phones usually requires simplifying the algorithm to adapt to a cellular phone's relatively small CPU processing power. The biometric image of the voter can either encrypted at the mobile device and sent to the server or process the biometric image at the mobile device to generate the biometric template and sent it to the server. The fingerprint template is protected using symmetric hashing, and the iris template is protected using bloom filter-based feature transformation before it is sent to the server.

A new wavelet-based image encryption is proposed to adapt to the small CPU processing power and battery charge of the cellular phone. In the wavelet-based AES encryption, Haar DWT [4] is applied to concentrate most of the signal energy to the LL part, the encryption is performed only on LL(1/4) part of the image to speed up the encryption process. The performance and security of AES encryption, wavelet-based AES encryption, and the processing of the biometric image are analyzed.

* Corresponding author.
E-mail address: ajishs2014@gmail.com (Ajish S.).

https://doi.org/10.1016/j.jisa.2021.102908

Available online 17 June 2021

The rest of the paper is organized as follows. Section 2 describes the internet voting system used in Estonia. Section 3 describes the proposed secure mobile internet system using wavelet-based AES. Section 4 analyzes the performance of AES encryption, wavelet-based AES encryption, and template generation at the mobile device. Section 5 examines the security of AES, wavelet-based AES encryption, fingerprint template hashing, and the bloom filter-based feature transformation. Section 6 analyzes the security of the mobile internet voting system. Section 7 point out and discuss the results, and Section 8 concludes the paper.

2. Existing methodologies

2.1. The voting system

The Estonian i-voting system uses public-key cryptography to generate the digitally signed encrypted vote called "double envelop" of the ballot. The digital signature (outer envelope) is used to verify the voter's identity, and it is generated by using the private key of the voter. The encrypted vote(inner envelope) is used to maintain the vote's secrecy, and it is generated by using the server's public key. The server verifies the voter's identity by verifying the digital signature and decrypt the vote before counting them [5].

2.1.1. Vote casting process

The voters in the Estonian i-voting system first download the client voting application published by the voting officials on the website https://valimised.ee. The voter then opens the client voting application and insert the national ID card. The voter authenticates to the server by using the PIN1 of the national ID card and establishes a TLS client authenticated connection to the Vote Forward Server. The server checks the voter's eligibility, and the candidate list of the appropriate electoral district is sent back to the voter [5,6].

The voter picks her candidate 'c' and enters the PIN2 of the national ID card that is used for digitally signing the vote. The voter application generates a random number 'r' and pad it with 'c' and encrypts it using the RSA algorithm using the 2048 bit public key of the server and digitally sign it using the private key(PIN2) of the voter. In [7] the voting protocol is modified by using an OTP instead of using the random number 'r'. The signed and encrypted vote is called double envelope in which the outer envelope is the digital signature and the inner envelope is privacy protected encrypted ballot. The client application sends the double envelope to the server and the server assigns a voter reference 'vr' to the ballot. The server generates a QR code that contains the random number 'r' and vote reference 'vr' and send the QR code to the voter for the verification of vote.

The Estonian i-voting system allows the voter to cast a vote multiple times to protect the coercion, and only the last vote cast is taken into account. All the previous ballots cast by the voter are revoked, but it is logged in the log server. The client voting application indicates whether the voter cast vote previously. The voter can cast a vote on the poll day using a paper ballot and revoke the electronic vote cast by her.

2.1.2. Verification process

The voter can verify the vote using the verification application provided by the election officials [4,8] through a mobile phone. The verification application scans the QR code displayed at the client voting machine using a mobile phone to retrieve the vote reference 'vr' and random number 'r'. The verification application sends the vote reference 'vr' to the server, and the server sends back the corresponding encrypted vote $Encry_{vote}$ and the candidate list CL.

The verification application encrypts each possible candidate in the candidate list CL padded with the random number 'r' using the server's public key and compares it with the encrypted vote $Encry_{vote}$ received from the server. If there is any match, the corresponding candidate is displayed to the voter, and the voter can verify the vote cast's correctness. The voter can verify the vote up to 3 times per vote and within 30 min of the vote cast.

2.1.3. Tabulation

After the online voting ends, the Vote Storage Server verifies each ballot's digital signature(outer envelope) and eliminates the revoked or invalid votes. The digital signatures are strip off from the double envelope during the counting phase, and the encrypted votes are separated. The encrypted votes are transferred to the counting server using DVDs [5,6].

The HSM module attached to the counting server contains the private key of the server. The counting server decrypts the votes using HSM and counts the votes of each candidate. The result of online voting is combined with the in-person voting at polling stations and declare the overall result of the election.

3. Proposed method

Internet voting, also called i-voting, is the voting technology by using electronic devices over the internet. The general architecture of the i-voting system is shown in Fig. 1. The main entities of the internet voting system are [5]

1. Voter: The two main client-side applications are the voter application and the verification application. The voter uses the voter application to cast a vote and the verification application to verify the vote cast by the voter.
2. Central System: Central system is the back end of the voting system, and it consists of the server, database, and vote-counting application.
    - Vote Storage Server(VSS): The VSS stores the i-vote received from the VFS, cancels the votes, and verify the digital signature of the i-vote.
    - Vote Counting Application(VCA): The VCA is an isolated entity in the central system, and it is not connected to the internet. The VCA decrypt the i-vote, counts the vote for each candidate, and publishes the result.
    - Voter List: The voter list database contains the list of eligible voters along with their related consistency.
    - Candidate List: The candidate list database contains the eligible candidates according to the consistency.
    - Aadhaar CIDR Database: The CIDR database contains the fingerprint and iris biometric templates of the voter.
3. Key Management: The key management system generate the private and public key, the public key is combined with the voting application, and the private key is kept confidential in the vote-counting application.
4. Auditing: The audit application audit all activities of the Vote Forward Server, Vote Storage Server, and the counting application to solve any disputes.

3.1. Voting process

The voter should download and install the voting application on their mobile device and opens the app for the cast of votes. The voter first opens the QR code scanner in the voter app and scans the QR code in the Aadhaar card. The Unique Identification Authority of India (UIDAI) [3] is a statutory authority established by the India Government. The officials of UIDAI [3] collect the biometric(finger and iris) and demographic data of residents in India, store it in a centralized database, and issue a Unique Identification Number(UIN) also called the Aadhaar number to each resident.

Voting Protocol

1. The voter V: The voter scans the QR code of the Aadhaar card and sent the details to the Vote Forward Server(VFS).
2. The voter then scans the fingerprint or iris for biometric authentication.

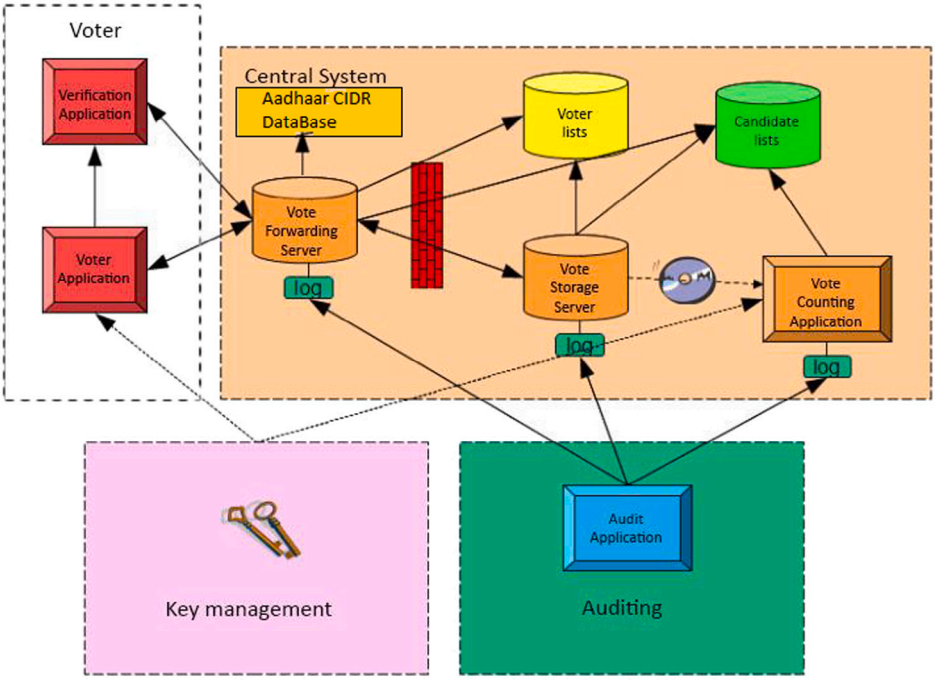

Fig. 1. Internet voting system [5].

3. The voter app encrypts the biometric image using AES or wavelet-based AES and sent it to the VFS or process the biometric image at the mobile device protect it and sent it to the server.
4. Vote Forward Server:
   (a) VFS verifies the authenticity of the voter using fingerprint or iris authentication method.
   (b) Sent the voters list CL = $\{C_1, C_2, \ldots, C_m\}$ to VoterApp, where m represents the total candidates.
5. Voter V: Chooses C_i from CL
6. The VoterApp do the following:
   (a) Using a random number generator, creates a random number r.
   (b) Encrypts the vote C_i and r by using the public key of the server(pkS), $Encry_{vote} = AsymEnc_{pkS}(C_i,r)$.
   (c) Digitally sign the encrypted vote $Encry_{vote}$ by using voter's private key(PIN2) or (skV), SignEncVote = $Sign_{skV}$ ($Encry_{vote}$).
   (d) Sent the SignEncVote to Vote Forward Server.
7. Vote Forward Server do the following:
   (a) Forward the SignEncVote to the Vote Storage Server.
   (b) Generates voteref and sent the voteref to the voter.

The scanned details like UID, name, gender, year of birth, and address are encrypted with the DES encryption algorithm [9]. A session key is used to encrypt the scanned detail. The session key is encrypted with the RSA encryption algorithm [9] using the public key of VFS(PU_{VFS}) integrated into the voting application. The encrypted scanned details and the encrypted session key is sent to the VFS.

The VFS decrypts the encrypted session key using the private key of VFS(PR_{VFS}) and then decrypts the encrypted details of the voter using the session key. The VFS then verifies the eligibility of the voter by checking the voter list database. The next step is the fingerprint or iris biometric authentication of the voter.

3.1.1. Fingerprint biometric authentication

The UIDAI [3] collects the fingerprint and iris templates of residents in India, store them in a centralized database, and issue the Aadhaar number to each resident. The voter should scan the fingerprint or iris using the voter application, and it is sent to the centralized database through the Vote Forward Server. There are two methods for the processing of the fingerprint template at the mobile device. They are (1) the scanned fingerprint image is encrypted at the mobile device and send to the centralized database, (2) process the fingerprint image at the mobile device to generate the biometric template, protect it and send it to the centralized database.

AES encryption of biometric image at the mobile device. The scanned fingerprint/iris image is encrypted using the Advanced Encryption Standard(AES) encryption algorithm [9]. AES is a symmetric bock cipher in which the same key is used for encryption and decryption. The encipher and decipher process consists of 10 rounds, input plain text is converted into an input state of 4*4 matrix, and the state(4*4 matrix) is processed in each round. The four different stages at each round are (1) Substitute Bytes, (2) Shift Rows, (3) Mix Column, and (4) AddRound Key.

1. Substitute Bytes: Each byte in the matrix is substituted with an appropriate byte in a lookup table called S-box.
2. Shift Rows: The last three row of the matrix are shifted left.
3. Mix Column: Each column of the matrix is multiplied by a fixed polynomial.
4. AddRound Key: Each byte of the matrix is xor with the round key value.

Wavelet-based AES encryption of fingerprint/iris image at the mobile device. The implementation of biometrics on mobile devices usually requires simplifying the algorithm to adapt to the cellular phone's relatively small CPU processing power. The wavelet-based AES algorithm is proposed to speed up the encryption process at the mobile device. The Haar DiscreteWavelet Transform(Haar DWT) [4] is applied to the fingerprint/iris image before the encryption process. The Haar DWT consists of two operations: the horizontal operation and the vertical operation.

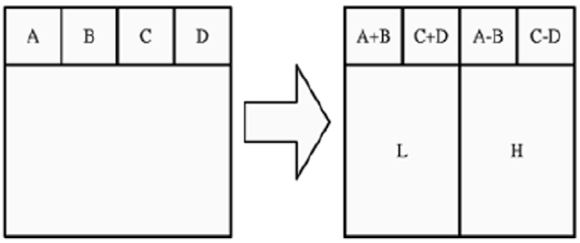

Fig. 2. Horizontal Operation [7].

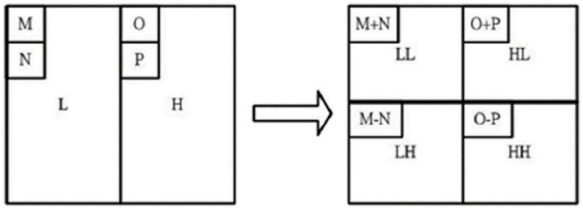

Fig. 3. Vertical Operation [7].

In the horizontal operation, the pixels are scanned from left to right and perform the addition and subtraction operation on nearby pixels. The pixel sum is stored in the left part, and the difference is stored in the right part (Fig. 2). The pixel sum, i.e., the left part denoted by L, is the low-frequency part, and the pixel difference, i.e., the right part denoted by H, is the high-frequency part of the original image.

The pixels are scanned from top to bottom in the vertical operation and perform the addition and subtraction operation on nearby pixels. The pixel sum is stored on the top part, and the difference is stored in the bottom part (Fig. 3). After all the rows and columns of the image are processed, there are four sub-bands LL, LH, HL, and HH (Fig. 3). Most of the signal energy is concentrated on the LL part, and the LL part looks close to the original image.

The layout of wavelet based AES encryption is shown in Fig. 4. The wavelet-based AES encryption transforms the iris/fingerprint image using first-order wavelet decomposition, and only the low-frequency LL part is encrypted using the AES algorithm. The encrypted LL part is then XORed with the high-frequency HL, LH, and HH part. Then the wavelet reconstruction is performed on the encrypted LL part to spread it to the whole image. In the wavelet-based AES encryption, the encryption is performed only on 1/4 part of the original image to speed up the encryption process.

The wavelet-based biometric image encryption consist of the following steps:

Step 1: Input the biometric fingerprint or iris image.

Step 2: Level one Haar wavelet transformation is applied to the biometric image to extract the low (LL) and high frequency(LH, HL, HH) wavelet coefficients.

Step 3: The low-frequency wavelet coefficients (LL) is encrypted using the AES encryption algorithm.

Step 4: The high-frequency wavelet coefficients (LH, HL, HH) is XORed with the encrypted low-frequency part (LL) for the encryption of the whole image.

Step 5: Haar wavelet reconstruction is applied to the encrypted low-frequency part (LL) to spread it to the entire image.

Minutiae generation of fingerprint image at the mobile device. The fingerprint consists of ridge and valley, which form a distinctive pattern on each individual [10]. The fingerprint image is processed at the mobile device to extract the unique feature called minutiae. The three main steps in the fingerprint minutiae extraction are [11–13] (1) Preprocessing (2) Minutiae extraction, and (3) Post-processing.

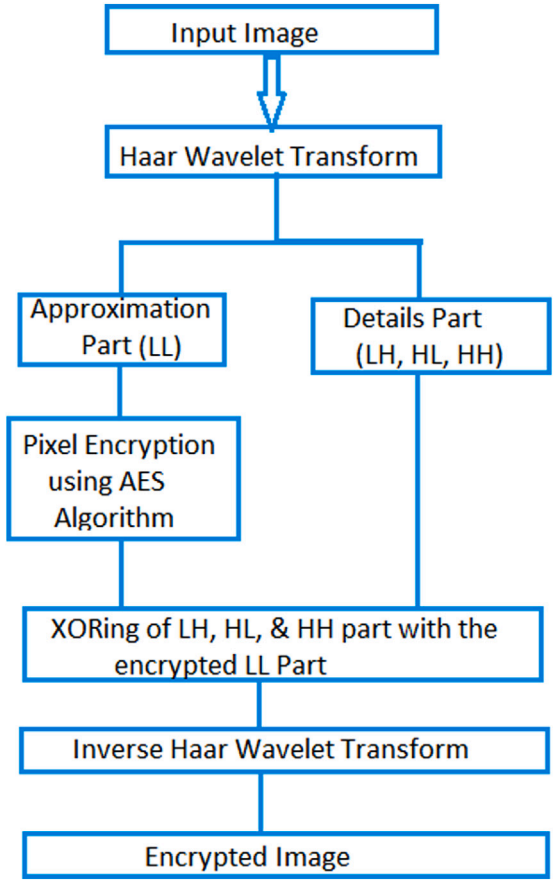

Fig. 4. Wavelet-Based AES Encryption.

3.1.1.0.1. Preprocessing The scanned fingerprint image is first preprocessed to eliminate the redundant information present in the fingerprint image. The different stages in the preprocessing are [13] (1) Image enhancement and (2) Normalization.

3.1.1.0.2. Minutiae extraction The most common fingerprint recognition method is the minutiae matching, and minutiae extraction is the most crucial step in the fingerprint recognition system. The different stages in the binarization based fingerprint minutiae extraction are [11,13] (1) Binarization (2) Image Thinning and (3) Minutiae Extraction.

3.1.1.0.3. Post-processing Due to the presence of sweat, scars, or dirt in the fingerprint image, irregular minutiae called false minutiae might appear along with the genuine minutiae [12,13]. Several examples of false minutiae are spurs, interrupted ridges, bridges, forks, triangles, and structure ladders. The post-processing stage removes the false minutiae from the minutiae set.

3.1.2. Fingerprint minutiae template protection using symmetric hashing

The fingerprint biometric templates are hashed using the symmetric hash function [14,15] before sending it to the server. In the symmetric hash function, the input pattern's order does not have any role in the hashed output. Consider the input $X = x_1x_2x_3 \dots x_n$ and the hash functions in Eq. (1) [14] and Eq. (2) [14].

$$H(X) = k_1x_1 + k_2x_2 + \cdots + k_nx_n, \quad where \quad k_1 \neq k_2 \dots \neq k_n \tag{1}$$

$$H^m_{sym}(X) = x^m_1 + x^m_2 + \cdots . + x^m_m \tag{2}$$

Consider the input X to the hash functions in Eq. (1) and Eq. (2) ($X = x_1x_2 \dots x_n$), if the order of the input X is changed to $X' =$

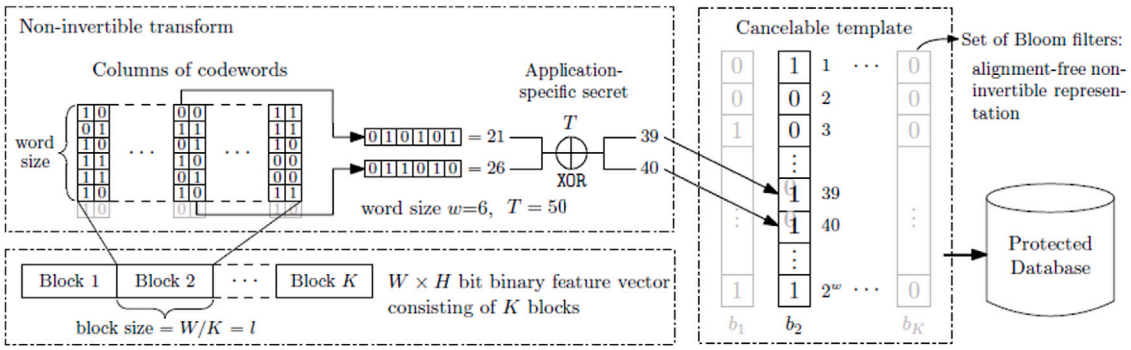

Fig. 5. Bloom Filter Based Feature Transformation [16].

$x_2x_3....x_nx_1$, the first hash function in Eq. (1) outputs an entirely different value because it is not symmetric, the second hash function in Eq. (2) produces the same output for the two inputs X and X'. Different symmetric hash functions can be generated by the combination of one or more symmetric hash function. The symmetric hash function is represented as $H_{sym,f}(X)' = f(H^1_{sym}(X), \ldots, H^n_{sym}(X))$. The symmetric hash function has the property that the order of input pattern doesn't have a role in the hashed output, so symmetric hash functions are used for fingerprint minutiae protection.

Consider the n minutiae $\{c_1, c_2, \ldots, c_n\}$ and m hash functions described by Sergey Tulyakov in [14] which are as follows.

$$\begin{aligned} h_1(c_1, c_2, \ldots, c_n) &= \frac{c_1 + c_2 + \cdots . + c_n}{n} \\ h_2(c_1, c_2, \ldots, c_n) &= \frac{c_1^2 + c_2^2 + \cdots . + c_n^n}{n^2} \\ &\ldots\ldots \\ h_m(c_1, c_2, \ldots, c_n) &= \frac{c_1^m + c_2^m + \cdots . + c_n^m}{n^m} \end{aligned} \tag{3}$$

In the above equations, m denotes total hash functions and n denotes the total minutiae points as input, if m<n then it is impractical to produce the original minutia template from the hashed template.

3.1.3. Iris biometric authentication

The iris biometric system is a highly secure and most accurate biometric system with less False Acceptance Rate [10]. The unique characteristics of iris, such as epigenetic pattern and the stability over time, are desirable for use as biometric authentication and identification of individuals. John Daugman [17,18] proposed the preliminary version of an iris recognition system, and the different stages of the iris recognition system are (1) segmentation, (2) normalization (3) feature encoding.

Segmentation. The iris region is embedded between the iris-sclera and the iris pupil. The iris region is isolated from the sclera and pupil in the first stage of segmentation. Usually, the eyelid obstructs the lower and the upper part of the iris. The second stage of segmentation is to remove the eyelid and eyelash part and to exclude other artifacts to uncover the circular iris region.

Iris normalization. In the normalization process, the iris image's Cartesian coordinate is transferred into a fixed-sized rectangular, polar coordinate to attain better comparison results. The varying distance between the camera and the iris affects the dimension of the iris image. The most significant and commonly used iris normalization techniques are the Daugman rubber sheet model [19] and Wildes [20] image registration.

Feature extraction. The Feature extraction [19] stage extracts the iris image's discriminating information and generates a biometric template for accurate recognition. The significant features present in the iris are extracted using texture analysis techniques and included in the biometric template for precise matching. Most feature extraction methods use the bandpass decomposition of the normalized iris image to create a biometric template.

3.1.4. Bloom filter based feature transformation

The bloom filter [21] is designed as a binary array of length 2^w; initially, all the 2^w bits are set to zero. Usually, a set of bloom filters(k bloom filters) are used to represent the protected template and k hash functions $h_1, h_2, \ldots, h_k$ that maps hashed output to the range $[0, 2^w - 1]$. The bit at position p of the bloom filter b_i is set to 1 if the hash output $h_i(x)$ is p, where x ε S is the input of the bloom filter. A bit of the bloom filter b_i may set to 1 multiple times. If the bit position $h_i(y)$ is set in bloom filter b_i, which indicates the presence of the element y in S. If the bit position $h_i(y)$ is set to 1, which means that the element y is present in S only with a particular probability of false positive.

The feature vectors extracted from the normalized iris image is usually represented as a two-dimensional binary feature vector of width W and height H. As shown in Fig. 5, the iris code of width W is divided into K equal size blocks, where the size of each block= W/K = l and each column contain $w \leq H$ bits. As shown in Fig. 5, there is one separate bloom filter for each block, a total of K bloom filters [16] each of length 2^w with template size K*2^w. The columns in each block are successively mapped to the assigned bloom filter's corresponding bit location. The transform of an iris feature vector to bloom filter b_i is implemented by converting each column of the block i into a decimal value. The decimal value is then XOR with an application-specific secret T, the result is used as the index of bloom filter b_i, and it is set to one. The mapping function of each column x $\epsilon(0,1)^w$ is defined as

$$\text{b[h(x)]} = 1 \text{ with h(x)} = \Big(\sum_{j=0}^{w-1} x_j.2^j\Big) \oplus \text{T}$$

The bloom-filter based feature transformation reduces the size of the protected iris template.

3.1.5. Encryption and digital signature generation of vote

After the voter's successful biometric authentication, the server checks the voter's eligibility, and the candidate list of the appropriate electoral district is sent back to the voter. As shown in Fig. 6, the candidate list CL=$\{C_1, C_2, \ldots, C_n\}$ contain the candidate number and the name of the candidate. The voting application generates a random number r and concatenates it with the voter selected candidate number C_i. The vote cast(C_i, r) is encrypted using the RSA algorithm with the public key integrated into the voting application. The public key used for encryption $PU_{server} = \{e_s, n_s\}$ and the private key used for decryption $PR_{server} = \{d_s, n_s\}$. The private key is kept secret in the Hardware Security Module attached to the vote-counting application. The vote is encrypted by using the method described below:

$Encrypt_{vote} = (C_i, r)^{e_s} \bmod n_s$

Fig. 6. Candidate List and Casting of Vote.

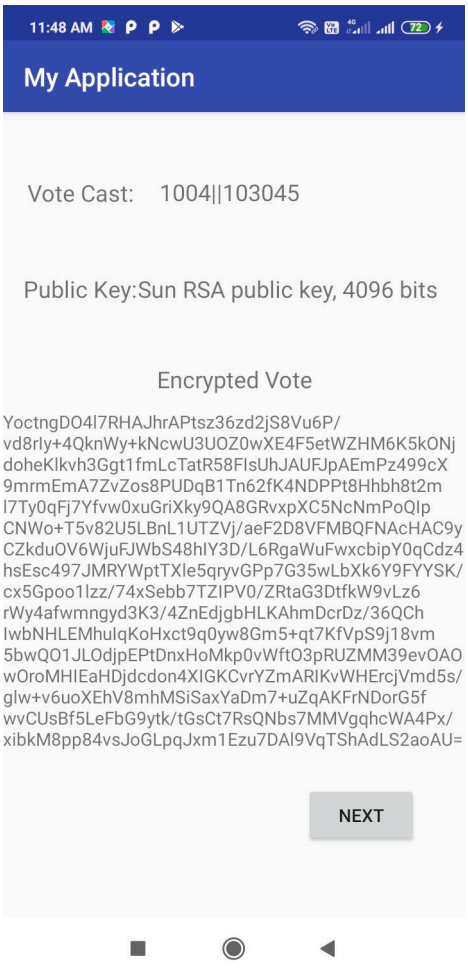

Fig. 7. Encrypted Vote.

The encrypted vote is shown in Fig. 7. The encrypted vote can be decrypted only by the vote-counting application using the private key PR_{server}.

The encrypted vote is digitally signed by the voting application using the private key generated by using a mobile ID [22]. The private key is used for digitally signing the vote $PR_{voter} = \{d_v, n_v\}$ and the public key to verify the digital signature $PU_{voter} = \{e_v, n_v\}$. The encrypted vote is digitally signed by using the method described below:

$$Sign_{vote} = (Encrypt_{vote})^{d_v} \bmod n_v$$

The voter should enter the private key generated by mobile ID [22], and the encrypted vote is digitally signed using the voter's private key. The digitally signed vote is verified by the Vote Storage Server using the public key PU_{voter}. The voting application sent the signed and encrypted vote to the Vote Forward Server(VFS), the VFS forward it to the Vote Storage Server.

The Vote Storage Server verifies the digital signature; for that, the vote forward server decrypts the $Sign_{vote}$ using the Public Key PU_{voter} as described below:

$$Encrypt'_{vote} = (Sign_{vote})^{e_v} \bmod n_v$$

If the decrypted signed vote $Encrypt'_{vote}$ matches the $Encrypt_{vote}$ sent by the voter application the digital signature verification is a success; otherwise, the voter should revote. The Vote Storage Server generates a QR code that contains the random number r and a vote reference(voteref) and sends it to the voter through the VFS.

3.2. Verification process

The voter first downloads and installs the verification application on the mobile phone and verifies the application's authenticity. Verify application then scans the QR code displayed by the voter application, which contains the random number 'r' and the vote reference 'voteref'. The verify app then send the voteref to the Vote Storage Server(VSS), and the VSS sends back the encrypted vote and the candidate list CL=$\{C_1, C_2, \ldots, C_n\}$ to the verify application. Verify application encrypt all the candidates in the candidate list CL using the public key of server PU_{server} and compare it with the encrypted vote send by the VSS, and identify the candidate number that the voter cast a vote and display it on the screen.

3.3. Counting

The digital signatures are removed from the double envelope during the counting phase, and the encrypted votes are separated. The encrypted votes are transferred to the counting server using DVDs. The HSM module attached to the counting server contains the private key of the server. The counting server decrypts the votes using HSM and counts the votes of each candidate. The result of online voting is published on the website.

4. Experimental implementation

The fingerprint or iris image of the voter can either be encrypted at the mobile device by using AES or wavelet-based AES encryption and

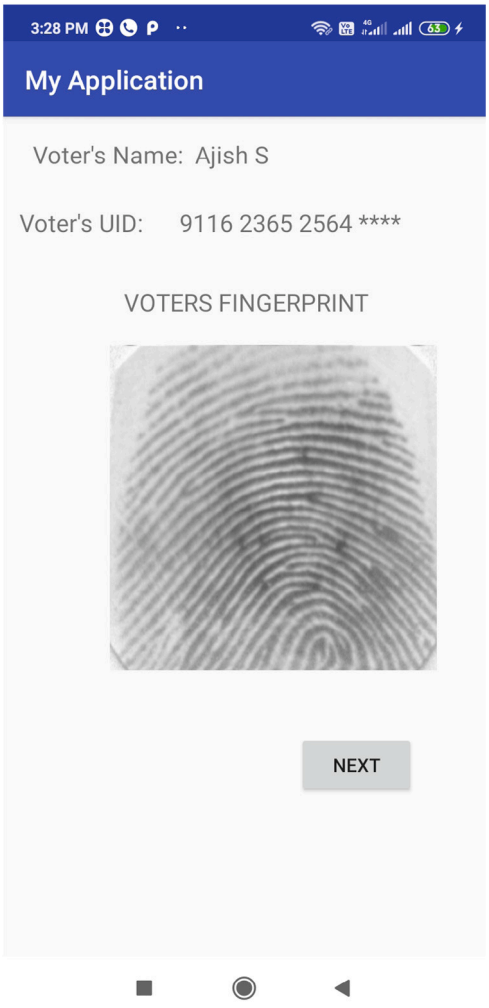

Fig. 8. Fingerprint image of the Voter.

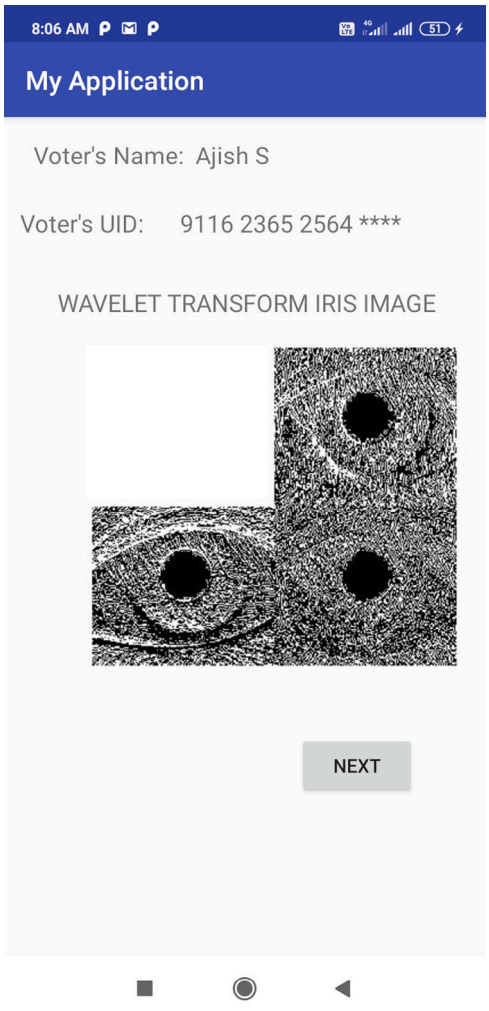

Fig. 9. Fingerprint image of the Voter.

Table 1
AES encryption time of fingerprint images.

Sl No	Fingerprint image	Image size (KB)	Encryption time (s)	CPU utilization (%)
1	101–1	88.3	0.841	24.2
2	101–2	88.3	0.882	26.1
3	101–3	88.3	0.878	25.2
4	101–4	88.3	0.871	25.1
5	101–5	88.3	0.825	24.1
6	101–6	88.3	0.832	24.3
7	101–7	88.3	0.817	24.1
8	101–8	88.3	0.801	23.9

Table 2
Wavelet transformation time of fingerprint images.

Sl No	Finger print image	Image size (KB)	Wavelet transformation time (s)	CPU utilization (%)
1	101–1	88.3	0.096	10.5
2	101–2	88.3	0.068	9.6
3	101–3	88.3	0.097	10.3
4	101–4	88.3	0.077	9.8
5	101–5	88.3	0.077	9.9
6	101–6	88.3	0.065	8.5
7	101–7	88.3	0.088	9.6
8	101–8	88.3	0.093	10.2

send to the server or process the fingerprint or iris image at the mobile device to generate the biometric template, protect it, and send it to the server. The performance of the three methods are analyzed using the Android/Java program.

4.1. AES encryption of fingerprint image at the mobile device

The fingerprint image of the voter is shown in Fig. 8. The eight fingerprint images of the FVC2004's DB3 fingerprint image database is encrypted using the AES algorithm, the encryption time, and the CPU utilization is reported in Table 1. The average encryption time of the eight fingerprint images is 0.843 s and the average CPU utilization is 24.625%.

4.2. Wavelet-based AES encryption of fingerprint image at the mobile device

The eight fingerprint images of the FVC2004's DB3 fingerprint image database is encrypted using a wavelet-based AES algorithm, the wavelet transformation time and CPU utilization are reported in Table 2. The average time for wavelet transformation of eight fingerprint images are 0.082 s, and the average CPU utilization is 9.8%.

The encryption time and the CPU utilization of the wavelet transformed fingerprint image are reported in Table 3. The wavelet transformed fingerprint image's average encryption time is 0.662 s, and the CPU utilization is 17.537.

Table 3
Encryption time of wavelet transformed fingerprint images.

Sl No	Finger print image	Image size (KB)	Encryption time (s)	CPU utilization (%)
1	W101–1	52.4	0.670	18.6
2	W101–2	51.4	0.648	16.4
3	W101–3	52.8	0.680	18.7
4	W101–4	53.3	0.662	17.5
5	W101–5	52.6	0.665	17.6
6	W101–6	51.2	0.647	16.4
7	W101–7	53.1	0.660	17.5
8	W101–8	51.6	0.664	17.6

Table 4
Minutiae generation time of fingerprint images.

Sl No	Finger print image	Image size (KB)	Minutiae generation time (s)	CPU utilization (%)
1	101–1	88.3	2.110	46.2
2	101–2	88.3	2.188	47.3
3	101–3	88.3	2.121	47.1
4	101–4	88.3	2.201	48.2
5	101–5	88.3	2.134	47.5
6	101–6	88.3	2.163	46.7
7	101–7	88.3	2.101	47.3
8	101–8	88.3	2.112	45.9

Table 5
Hashing time of fingerprint template.

Sl No	Fingerprint template	Template size (KB)	Template hashing time (s)	CPU utilization (%)
1	T101–1	1.29	0.062	8.5
2	T101–2	1.18	0.059	7.6
3	T101–3	0.836	0.065	8.2
4	T101–4	1.06	0.058	7.2
5	T101–5	1.18	0.057	7.1
6	T101–6	1.40	0.059	7.4
7	T101–7	1.46	0.058	7.6
8	T101–8	1.07	0.057	7.3

4.3. Minutiae generation of fingerprint image at the mobile device

The minutiae points of the eight fingerprint images of the FVC2004's DB3 fingerprint image database are generated, and the minutiae generation time is reported in Table 4. The average minutiae generation time of the eight fingerprint images is 2.141 s, and the CPU utilization is 41.775%.

4.4. Hashing of fingerprint minutiae template at the mobile device

The fingerprint minutiae templates are hashed using the symmetric hash function before it is sent to the server. The eight fingerprint minutiae templates are hashed using a symmetric hash algorithm [14], the hashing time, and CPU utilization reported in Table 5. The eight fingerprint template's average hashing time is 0.059 s, and the CPU utilization is 7.6%.

4.5. Encryption of iris image at the mobile device

The seven iris images of the CASIAv1 iris image database are encrypted using the AES algorithm; the encryption time and CPU utilization are reported in Table 6. The average encryption time of the seven iris images is 0.744 s, and the CPU utilization is 18.45%.

4.6. Wavelet-based iris image encryption at the mobile device

The seven iris images of the CASIAv1 iris image database is encrypted using a wavelet-based AES algorithm, and the wavelet transform time and CPU utilization are reported in Table 7. The wavelet transformed iris image is shown in Fig. 9. The average wavelet transform time is 0.031 s, and CPU utilization is 8.5%.

Table 6
AES encryption time of iris images.

Sl No	Iris image	Image size (KB)	Encryption time (s)	CPU utilization (%)
1	0011–1	11.6	0.716	17.8
2	0011–2	11.4	0.723	18.2
3	0011–3	11.8	0.755	18.6
4	0011–4	11.3	0.759	18.7
5	0011–5	12.2	0.797	19.2
6	0011–6	10.6	0.726	18.4
7	0011–7	10.9	0.736	18.3

Table 7
Wavelet transformation time of iris images.

Sl No	Iris image	Image size (KB)	Wavelet transformation time (s)	CPU utilization (%)
1	0011–1	11.6	0.031	9.8
2	0011–2	11.4	0.031	9.6
3	0011–3	11.8	0.032	10.3
4	0011–4	11.3	0.032	10.1
5	0011–5	12.2	0.032	9.7
6	0011–6	10.6	0.030	9.6
7	0011–7	10.9	0.030	9.4

Table 8
Encryption time of wavelet transformed iris images.

Sl No	Iris image	Image size (KB)	Wavelet based encryption time (s)	CPU utilization (%)
1	0011–1	47.4	0.621	11.3
2	0011–2	47.2	0.613	11.1
3	0011–3	47.7	0.604	11.6
4	0011–4	47.2	0.611	12.2
5	0011–5	47.9	0.632	13.1
6	0011–6	46.8	0.596	11.1
7	0011–7	47.1	0.604	12.6

Table 9
Iris code generation time.

Sl No	Iris image	Image size (KB)	Minutiae generation time (s)	CPU utilization (%)
1	0011–1	11.6	1.341	39.9
2	0011–2	11.4	1.326	40.3
3	0011–3	11.8	1.379	41.5
4	0011–4	11.3	1.329	40.6
5	0011–5	12.2	1.302	40.2
6	0011–6	10.6	1.346	41.7
7	0011–7	10.9	1.226	39.6

The wavelet transformed iris images are encrypted using the AES algorithm, and the encryption time of wavelet transformed iris image is reported in Table 8. The average time for wavelet encryption is 0.611, and the CPU utilization is 11.87%.

4.7. Iris code generation at the mobile device

The iris code of the seven iris images of the CASIAv1 iris image database is generated, and the iris code generation time and the CPU utilization is reported in Table 9. The average iris code generation time of the eight fingerprint images are 1.321 s, and CPU utilization is 40.54%.

4.8. Boom filter based iris template protection at the mobile device

The iris temple is protected using bloom filter-based feature transformation [21] before it is sent to the server. The transformation time

Table 10
Bloom filter transformation time.

Sl No	Iris template	Template size (KB)	Bloom transformation time (s)	CPU utilization (%)
1	T0011–1	1.42	0.160	18.2
2	T0011–2	1.44	0.166	19.6
3	T0011–3	1.45	0.168	18.5
4	T0011–4	1.47	0.154	17.4
5	T0011–5	1.43	0.164	18.6
6	T0011–6	1.45	0.166	18.4
7	T0011–7	1.44	0.168	17.9

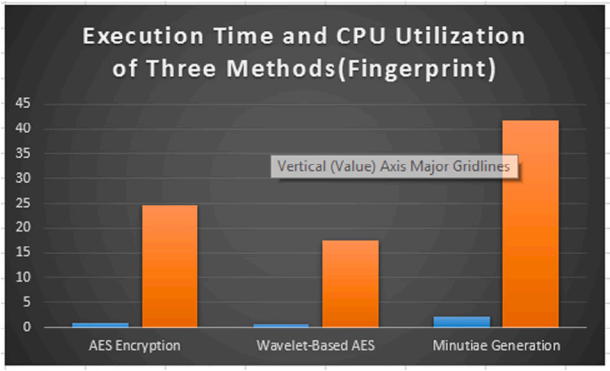

Fig. 10. Execution Time and CPU Utilization of AES Encryption, Wavelet based AES Encryption and Minutiae Generation of Fingerprint Image.

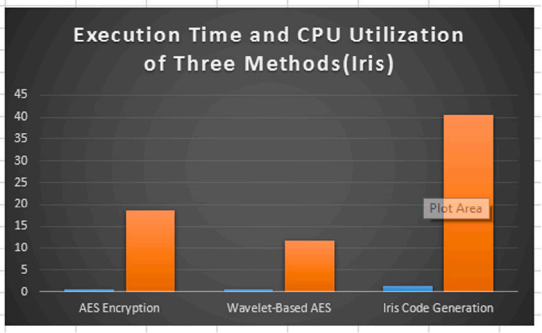

Fig. 11. Execution Time and CPU Utilization of AES Encryption, Wavelet based AES Encryption and Iris Code Generation of Iris Image.

and CPU utilization are reported in Table 10. The average bloom filter-based feature transformation time is 0.163 s, and the CPU utilization is 18.371%.

4.9. Comparison of performance of encryption and minutiae/iris code generation at mobile device

The execution time and CPU utilization of the three methods are reported in Table 11. The execution time of AES encryption, wavelet-based AES encryption, and the minutiae generation of fingerprint images is shown in Fig. 10. In fingerprint biometric authentication, the best performer is the wavelet-based fingerprint image encryption at the mobile device with an encryption time of 0.744 s and CPU utilization of 17.573%. Fig. 11 shows the execution time of AES encryption, wavelet-based AES encryption, and the iris code generation of iris images. In iris biometric authentication, the best performer is the wavelet-based iris image encryption at the mobile device with an encryption time of 0.642 s and CPU utilization of 11.87%.

5. Security analysis of biometric protection methods

5.1. AES and wavelet-based AES encryption

5.1.1. Brute force attack

The brute force attack [27] try all possible keys until the plain text is correctly decoded from the ciphertext. The maximum key size used in the AES algorithm is 256-bits. A high-performance 4-core intel i7 machine can process 230 bytes per second or 1024MiB/sec. The AES encryption algorithm process blocks 16(2^4) bytes, so the high-performance intel i7 machine can encrypt $2^{30-4} = 2^{26}$ blocks per second.

The total seconds in a year is 60 * 60 * 24 * 365.25 = 31,557,600 and the total keys that can be search for decryption in a year is 31,557,600 * 226, or 2,117,794,686,566,400 or 2117.8 trillion keys. The brute force attack on AES-256 needs to try 2^{256} possible keys. The total time to complete the brute force attack in years is 2^{256}/2117.8, nearly equal to 2.73 * 10^{61} years. i.e., Twenty-seven trillion trillion trillion trillion trillion trillion years is needed for the completion of brute force attack using a single high-performance intel core i7 machine.

5.1.2. Side-channel attacks

The side-channel attack [25] exploits the weakness in implementing the AES encryption and decryption process. The side-channel attack is a timing attack that records and analyzes the time taken for the AES encryption. Form the encryption time, the attacker analyzes the CPU clock cycle used for the encryption process and tries to derive the AES encryption key. The AES implementation prevents the side-channel attack by providing constant time for the encryption process by adding extra time to maintain the encryption time constant. Thus the AES algorithm should defeat the side-channel attack.

5.2. Fingerprint template hashing

The non-invertibility of the protected template is measured as total calculations utilize by an attacker to regenerate the minutiae through any attack procedure. The attacker can easily go through the brute force attack in the case of the hash method proposed by Sergey Tulyakov in [9], because the hashed minutiae points and the original minutiae points are nearby values.

The non-invertibility of the protected template is measured as total calculations utilize by an attacker to regenerate the minutiae through any attack procedure. Suppose there were n minutiae points in the transformed or protected minutiae template. The attacker first guesses various n-tuples called the pre-image of the minutiae. Suppose the advisory checks only the 2^{H_i} most expected pre-images of minutia v_i where i = 1...n. Here H_i the entropy is calculated using Eq. (4) [19]

$$H_i = -\sum_{r=1}^{m_i} P(l_{v_i} = r|\vec{v}_i) log_2(P(l_{v_i} = r|\vec{v}_i)) \quad (4)$$

where m_i is the total pre-images of v_i, and l_v is the variable representing the pre-image of $\vec{v}$, which is true one [19].

In the hash method proposed by Sergey Tulyakov in [13], the attacker can quickly identify the pre-image set because the original minutiae and the hashed minutiae are nearby values. Also, the total minutiae points in the pre-image set for each transformed minutiae are less. So the attacker can quickly reverse the transformed minutiae(hashed minutiae) to the original minutiae.

The irreversibility analysis of the hash function proposed by Sergey Tulyakov in [13] is conducted by considering the hashing of the triplet, i.e the hash function ($h(c1, c2, c3) = \frac{c1+c2+c3}{3}$) with three input minutiae (c1,c2,c3) (Fig. 12).

Consider (x,y) coordinates of the first minutiae and its two nearest neighbors. The x and y coordinates of the minutiae points are c_1=(222,54), c_2=(262,55) and c_3=(263,64). The hashed point h(c_1, c_2, c_3)=($\frac{222+262+263}{3}$, $\frac{54+55+64}{3}$)=(249, 57.666).

Table 11
Comparison of different methods.

Sl No	Method	Average encryption time (s)	Minutiae/Iris generation time (s)	Hashing/Bloom transformation time (s)	Total time (s)	CPU utilization (%)
1	Fingerprint encryption at the mobile device	0.843	–	–	0.843	24.625
2	Wavelet based fingerprint encryption at the mobile device	0.744	–	–	0.744	17.573
3	Minutiae generation and hashing at the mobile device	–	2.141	0.059	2.2	41.775
4	Iris encryption at the mobile device	0.744	–	–	0.744	18.45
5	Wavelet based Iris encryption at the mobile device	0.642	–	–	0.642	11.87
6	Iris code generation and bloom filter transformation at the mobile device	–	1.321	0.163	1.484	40.54

Table 12
Different i-voting methods and the possible attacks.

Method	Vote Modifying Malware (VMM) Attack	Re-voting Malware (RVM) Attack	Self-voting Malware (SVM) Attack	Bad verification attack	Violation of vote secrecy
Internet Voting in Estonia [23]	VMM attack possible	RVM attack possible	SVM attack possible	–	–
Verifiable Internet Voting in Estonia [24]	VMM attack possible	RVM attack possible	SVM attack possible	Bad Verification Attack possible	Violate the secrecy of vote
Improving the verifiability of the Estonian Internet Voting scheme [24]	VMM attack possible	RVM attack possible	SVM attack possible	Bad Verification Attack not possible	Not Violate the secrecy of vote
Estonian Voting Verification Mechanism Revisited [25]	VMM attack possible	RVM attack possible	SVM attack possible	Bad Verification Attack possible	Violate the secrecy of vote
Estonian Voting Verification Mechanism Revisited Again [26]	VMM attack possible	RVM attack possible	SVM attack possible	Bad Verification Attack possible	Violate the secrecy of vote
I-voting system using Aadhaar Card	VMM attack not possible	RVM attack not possible	SVM attack not possible	Bad Verification Attack not possible	Not Violate the secrecy of vote

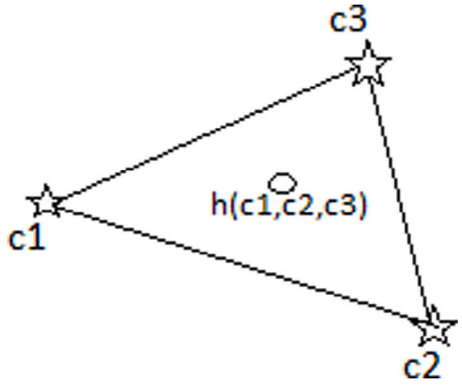

Fig. 12. Three minutiae(stars) participate in the creation of the triplet center(circle).

The reversibility attack is when the attacker attempts to find the minutiae points (c_1, c_2, c_3) from the hash value i.e., h(c_1, c_2, c_3). According to the hash method proposed by Sergey Tulyakov in [13], the x and y coordinates are nearby values of the x and y coordinates in the hashed template, so the attacker can easily go through the brute force attack and can easily identify the x and y coordinates of c_1, c_2, and c_3.

5.3. Bloom filter based feature transformation

For a bloom filter 'b' of length n, the probability that any bit set to 1 is 1/n, i.e., the probability that the bit remains 0 is 1-1/n. For a total of l code words in each block, the total number of bits set to 1 in each bloom filter is $1 - (1 - 1/n)^l$. A particular number of codewords are mapped to the same position in the bloom filter, even for a small value of l. Suppose $|b|$ bits are set to 1 after the mapping of l code words into the bloom filter. Hence the number of bits remapped is $1 - |b|/l$. The regeneration of the original iris block by the attacker involves $|b|$ codewords' arrangement into l positions(for the entire iris code, K times). For $|b| \leq l$, the total number of the possible sequence is analyzed by the function f$(|b|, l)$ [16] where

$$f(|b|,l) = \begin{cases} 1 & if \quad |b| = 1 \\ |b|^l - \sum_{i=1}^{|b|-1} \binom{|b|}{i} . f(i,l) & \text{otherwise} \end{cases} \tag{5}$$

The function f($|b|$,l) is the result of all possible sequence minus the sequences which contain less than b code words appear. The attacker endeavors to guess the matrix M^*, and the probability of success is 1/f$(|b|, l)$.

The total number of blocks in the bloom filter based transformation K= 15 and the number of codewords in each block l=32. For the irreversibility analysis, each bloom filter based transformed iris template is taken, and the number of ones in each bloom filter is counted. The f$(|b|, l)$ of each bloom filter is calculated using Eq. (5), and the average value of f($|b|$,l) for the bloom filter based transformed iris template is calculated.

The average value of the number of ones in each bloom filter in the transformed iris template with word size w=10 is $|b|$=22, and the number of codewords in each block l=32. The average value of f($|b|$,l) for the bloom filter based transformed iris template with word size w=10 is f$(|b|, l)$=$3.4501e^{+39}$. The probability of success of the irreversibility

analysis of the Bloom-Filter based transformed iris template with word size w=10 is $1/3.4501e^{+39}$.

6. Security analysis of internet voting system

The different i-voting methods and the possible attacks are discussed in Table 12. In the Internet Voting in Estonia by Priit [23], the voter's authentication is by using the PIN of the national ID card, so the Vote Modification Malware [5] can easily change the vote cast. The Revoting Malware [5] and Self Voting Malware [5] can easily cast a vote in the background. This method doesn't verify the ballot using the vote verification application.

The Verifiable Internet Voting in Estonia by Sven and Jan [24] provides both the voting and verification application. The bad verification attack [6] can easily bypass the verification protocol, and the vote's secrecy is violated during the verification stage.

Sven Heiberg [28] improve the Estonian internet voting scheme; the vote's secrecy is preserved in this method. But the Vote Modification Malware [5] can easily change the vote cast, and the Re-Voting Malware [5] and Self Voting Malware [5] cast a vote in background without the voter's knowledge.

In the Estonian Voting Verification Mechanism Revisited [29] and Estonian Voting Verification Mechanism Revisited Again [8] the Vote Modification Malware, Re-Voting Malware, and Self Voting Malware [5] can change the vote and cast vote in background. The Bad Verification Attack generates a dissimilarity between the verified vote and the actual vote in the server. These two methods violate the secrecy of the ballot during the verification stage.

The i-voting system using the Aadhaar card uses biometric methods instead of the PIN to authenticate voters. The use of biometric authentication with the modified verification protocol defends the Vote Modification Malware. The private key of the voter is derived from the fingerprint biometric template, and it is difficult for the re-voting and self-voting malware to derive the private key of the voter.

7. Research findings and discussion

The implementation of biometrics on mobile devices usually requires simplifying the algorithm to adapt to the relatively small CPU processing power and to reduce the cellular phone's battery usage. The state of art comparison of three methods is reported in Table 11. The data from Table 11 indicates that the best performer in terms of CPU utilization and execution time is the wavelet-based encryption of fingerprint images at the mobile device.

AES and wavelet-based AES's security analysis exhibits better security than the security of symmetric hashing and bloom filter-based feature transformation. The generation of the minutiae template and iris code requires more CPU utilization and processing time. The attacker can easily reserve the symmetric hashed minutiae points to the original minutiae points, and from the minutiae points, the attacker can regenerate the fingerprint image of the voter.

The proposed internet voting system with biometric authentication exhibits better security than the internet voting system in Estonia. The mobile-based internet voting system using biometric authentication protects the Vote Modification Malware, Re-Voting Malware, and Self Voting Malware attacks [5].

8. Conclusion

In this paper, a secure mobile phone-based internet voting system is implemented in which biometric methods are used to authenticate the voter. The three methods used for implementing biometric authentication at the mobile device are (1) AES encryption of the biometric image, (2) wavelet-based AES encryption of the biometric image, and (3) template generation of the biometric image. The wavelet-based AES algorithm speeds up the encryption process and reduces the mobile device's CPU utilization. The experimental analysis of three methods exhibits that the performance of wavelet-based AES encryption is much better than the AES encryption and template generation. The security analysis of three methods shows that AES and wavelet-based AES encryption shows better security than the protection of the biometric template. The analysis of the implemented i-voting system shows that biometric authentication defeats almost all the mobile-based threats.

CRediT authorship contribution statement

Ajish S.: Conceptualization, Methodology, Software, Data curation, Writing - original draft, Investigation, Resources, Validation, Writing - review & editing. **K.S. AnilKumar:** Visualization, Supervision, Project administration.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Funding

No funding was received for this work.

Intellectual property

We confirm that we have given due consideration to the protection of intellectual property associated with this work and that there are no impediments to publication, including the timing of publication, with respect to intellectual property. In so doing we confirm that we have followed the regulations of our institutions concerning intellectual property.

References

[1] Wahl Harald, Mense Alexander, Kaufmann Christian, Eckkrammer Florian, Gollner Helmut, Albayrak Ümit, et al. Mobile internet security threats and the security awareness of smart phone users. In: Las vegas international academic conference las vegas.
[2] M. Sujithra, Padmavathi G. Mobile device security: A survey on mobile device threats, vulnerabilities and their defensive mechanism. Int J Comput Appl (0975 – 8887) 2012;56(14).
[3] UIDAI. Aadhaar Enabled Services whitepaper aadhaarenabled service delivery. 2010.
[4] Chen Po-Yueh, Lin Hung-Ju. DWT based approach for image steganography. Int J Appl Sci Eng 2006.
[5] Torn Tanel. Security analysis of estonian I-voting system using attack tree methodologies (Master thesis). Tallinn University of Technology, Faculty of Information Technology, Department of Computer Science.
[6] Springall Drew, Finkenauer Travis, Durumeric Zakir, Kitcat Jason, Hursti Harri, MacAlpine Margaret, et al. Security analysis of the estonian internet voting system. ACM; 2014, 978-1-4503-2957-6/14/11.
[7] Ajish S. Enhanced ATM security using biometric authentication and wavelet based AES. MATEC Web Conf 2016;42:06003.
[8] Kubjas Ivo, Pikma Tiit, Willemson Jan. Estonian voting verification mechanism revisited again, smartmatic. Cybernetica Centre of Excellence for Internet Voting. Ulikooli, 51003 Tartu, Estonia.
[9] Stallings William. Cryptography and network security principles and practice. 5th ed. Prentice Hall.
[10] Wayman J. Handbook of iris recognition. Biometr IET 2014;3:41–3.
[11] http://shodhganga.inflibnet.ac.in/bitstream/10603/24394/9/09-chapter4.pdf.
[12] Bansal Roli, Sehgal Priti, Bedi Punam. Minutiae extraction from fingerprint images - a review. IJCSI Int J Comput Sci Iss 2011;8(5). 3.
[13] Cao Letian, Wang Yazhou. Fingerprint image enhancement and minutiae extraction algorithm (Master thesis). Linnaeus University. Sweden.
[14] Tulyakov Sergey, Farooq Faisal, Mansukhani Praveer, Govindaraju Venu. Symmetric hash functions for secure fingerprint biometric systems. Pattern Recognit Lett 2007;28:2427–36.
[15] Kumar Gaurav, Tulyakov Sergey, Govindaraju Venu. Combination of symmetric hash functions for secure fingerprint matching. In: 20th international conference on pattern recognition. IEEE Xplore; 2010.

[16] Rathgeb C, Breitinger F, Busch C. Alignment-free cancelable iris biometric templates based on adaptive bloom filters. In: Proceedings of ICB. 2013.
[17] Daugman JG. High confidence visual recognition of persons by a test of statistical independence. IEEE Trans Pattern Anal Mach Intell 1993;15:1148–61.
[18] Daugman J. Biometric personal identification system based on iris analysis. Google patents. 1994.
[19] Daugman John. How iris recognition works. IEEE Trans Circuits Syst Video Technol 2004;14(1):21–30.
[20] Wildes Richard P. Iris recognition: An emerging biometric technology. IEEE Proc 1997;85(9):1348–63.
[21] Rathgeb Christian, Breitinger Frank, Busch Christoph, Baier Harald. On the application of bloom filters to iris biometrics. IET Biometr 2013.
[22] Mobile identity, estonia's mobile-ID: driving today's e-services economy, mobile identity, GSMA.
[23] Priit Vinkel. Internet voting in estonia nordic conference on secure IT systems. Springer; 2011.
[24] Sven H, Jan W. Verifiable internet voting in Estonia. In: 6th international conference on electronic voting, verifying the vote. p. 1–8.
[25] Comparative study on AES 128-bit and AES 256-bit, https://shodhganga.inflibnet.ac.in/bitstream/10603/249377/3/ch-apter%203.pdf.
[26] Ajish, AnilKumar KS. Secure I-voting system with modified voting and verification protocol. Lecture notes in electrical engineering, vol. 672, Springer; 2020.
[27] Scrambox. How long would it take to brute force AES-256, https://scrambox.com/article/brute-force-aes/.
[28] Heiberg Sven, Martens Tarvi, Vinkel Priit, Willemson Jan. Improving the verifiability of the Estonian Internet Voting scheme. In: International joint conference on electronic voting. Springer; 2016, p. 92–107.
[29] Mus Koksal, Kiraz Mehmet Sabir, Cenk Murat, Sertkaya Isa. Estonian voting verification mechanism revisited. 2017.