International Journal of Engineering Trends and Technology
ISSN: 2231-5381/ https://doi.org/10.14445/22315381/IJETT-V74I4P106

Volume 74 Issue 4, 69-86, April 2026

Original Article

A Cryptographic Approach to Privacy-Preserving Credit Card Security Using PFYHD-CVV and KYM-ECC

Venkatesh Kumar M[1], C. Lakshmi[2]

[1]*Department of Computer Science & Engineering, SRMIST, Chennai, Tamil Nadu, India.*
[2]*Department of Computational Intelligence, SRMIST, Chennai, Tamil Nadu, India.*

[1]*Corresponding Author : veenkat@gmail.com*

Received: 04 August 2025 Revised: 05 February 2026 Accepted: 12 February 2026 Published: 29 April 2026

Abstract - *In today's digital world, secure Credit Card Transactions (CCT) are crucial, especially over a wireless network. None of the existing works overcame the web skimming and shoulder surfing attacks during CCT. Hence, this paper proposes a novel Permutation Fisher-Yates Hexadecimal Card Verification Value (PFYHD-CVV)-based CCV generation to avoid web skimming and shoulder surfing attacks during CCT. Primarily, the customer registers on the E-commerce application. Here, by using the PFYHD-CVV technique, the 3-digit Card Verification Value (CVV) is dynamically generated. Next, the public and private keys are generated using the Kaplan-Yorke Map-Elliptic Curve Cryptographic (KYM-ECC) method. Then, utilizing the eXclusive OR Merchant Product-identification-Menezes-Qu-Vanstone (XORMP-MQV) method, the secret key is created. Then, the merchant-product legitimacy is authenticated using the Entropy Inverse Message Digest 5 Digital Signature Algorithm (EIMD5-DSA. Also, the payment processing webpage's Internet Protocol (IP) address is spoofed via Right Shift 2's Complement (RS2C). Lastly, credit card details are encrypted using the KYM-ECC. Therefore, the proposed work performs secured CCT with a security level of 98.8564%, showing better performance than the prevailing works.*

Keywords - *Secure Wireless Payment Protocol, IP Spoofing, Data Privacy Preservation, Digital Signature Algorithm, Authentication.*

1. Introduction

Credit Cards (CC) have become an essential tool in the modern digital economy for directing digital transactions [1]. CC makes transactions convenient and flexible for customers/merchants to purchase products online [2]. Yet, the development of the CCT has also made it a primary target for cybercriminals [3]. The attackers hack the E-commerce application or steal user data, causing financial loss. Protecting the privacy and security of this sensitive information, including CC details, has become a critical concern as financial data flows through various digital channels [4]. Although traditional security mechanisms were effective to a certain extent, they were often insufficient in preventing data breaches and fraud. This leads to growing interest in developing privacy-preserving methods that not only secure the transaction process but also ensure the confidentiality of the user's personal and financial data [5].

The prevailing privacy-preserving techniques, like Rivest-Shamir-Adleman (RSA) and Elliptic Curve Cryptography (ECC), aimed to protect individual identities and transaction details from unapproved access [6]. Likewise, the existing Message Authentication Code (MAC) and Password-Based Key Derivation Function (PBKDF) allowed for authentic transactions during the purchase of merchandise online [7]. Yet, these methods were either complex or had side-channel attacks [8]. Another significant area of innovation in traditional works is the integration of Machine Learning (ML) and Artificial Intelligence (AI) to identify fraudulent behavior during transactions via CC [9]. These systems flag anomalies and take preventive measures without compromising user privacy by analyzing transaction patterns and user behaviour [10]. Besides, such systems must be carefully balanced to ensure that they do not misuse private information during the transaction in the E-commerce application [11].

Despite these advancements in privacy preservation, most prevailing models involve a trade-off between security, efficiency, and user convenience [12]. Besides, the quickly evolving tactics of cybercriminals must be identified to secure the transactions [13]. Protection of the webpage link was mandatory to avoid spoofing [14]. Other breaches included the Man in the Middle attack, web skimming, shoulder surfing, and so on. Robust privacy-preserving security mechanisms were essential to select the product effectively and to make transactions securely as the trust in CCT continued to grow. So, a novel PFYHD-CVV-based dynamic CVV creation,

EIMD5-DSA-based merchant product authentication, and KYM-ECC-based credit card details protection are proposed to ensure the confidentiality, reliability, and availability of transactions for secured CCT.

1.1. Research Gap

The web skimming attack is also known as the CVV attack. None of the prevailing works avoided the web skimming and shoulder surfing attack during CCT, leading to continued security vulnerability.

1.2. Problem Statement

The existing works' limitations are given as follows,

- The prevailing [15] did not concentrate on the webpage spoofing process during the CCT process, thus leading to the Man in the Middle attack.
- The existing [16] did not have proper verification of merchant availability with corresponding concerns. This led the customer to order unavailable products.
- One of the most significant issues of CCT in the prevailing [17] was that it relied on static keys to secure the CC details, causing leakage in server-managed secrets.
- When the CC details were not secured, the attackers injected malicious Structured Query Language (SQL), leading to the compromise of E-commerce applications.

1.3. Novelty

The proposed system introduces a privacy-preserving framework that addresses the critical limitations identified in existing CCT models. Unlike the prevailing works that failed to avoid web skimming and shoulder surfing attacks, the proposed PFYHD-CVV generates one-time CVV values and avoids the vulnerabilities. Further, RS2C-based IP spoofing is carried out to ensure secure communication between customers and merchants, thus overcoming the restriction seen in [15]. Afterward, the EIMD5-DSA is used to ensure that the merchant's product availability is appropriately verified, which was a challenge in [16]. Meanwhile, the XORMP-MQV-based secret key generation prevents the exposure of the key during data encryption, which was not carried out by [17]. Finally, the SQL attacks that are not prevented by the prevailing works are defended using KYM-ECC. Thus, the proposed system represents a novel solution that enhances the security, privacy, and scalability in CCT systems.

1.4. Objective

The contributions of the proposed work are provided below,

- To prevent web skimming and shoulder surfing attacks during the CCT, the PFYHD-CVV technique is utilized for the generation of a dynamic CVV, which makes the transaction secure.
- The RS2C-based IP spoofing is carried out to secure communication between the user and the merchant during the purchase of the product via CC. This reduces the stealing of information from the E-commerce application.
- The EIMD5-DSA is implemented to ensure that the merchant's product availability is appropriately verified for further transactions. Thus, the correct product is selected, reducing the latency in product selection.
- The secret key is generated dynamically using the XORMP-MQV technique, thus eliminating the exposure of the key and enhancing overall security.
- To defend against the SQL injection attack during CCT, the CC details are secured using KYM-ECC.

The paper is arranged as follows: The existing works are described in Section 2, the proposed system's methodology is elucidated in Section 3, the proposed work's performance is detailed in Sections 4 and 5, and Section 6 concludes the paper with future scope.

2. Literature Survey

Credit Card Transaction (CCT) security was mostly analyzed via cryptographic, authentication, and privacy-preserving models. These models mitigated the growing threats in the digital payment system. Existing studies had explored the RSA, ECC, and blockchain-based models for enhancing secure transactions.

Moreover, the prevailing research focused on regulatory and multi-factor authentication for online transactions. However, these methods suffered from static key dependency, ineffective protection against web-based attacks, increased latency, and scalability issues. And, there was a need for an integrated security framework to counter the attacks in the modern CCT environment. The existing approaches are detailed below.

[15] recognized protection of personal data during E-commerce-based transactions. Here, the personal data were protected in line with the General Data Protection Regulation (GDPR). The data was collected and stored in the SQL server. Next, the data was transferred to customer service, accounts, or marketing with security using GDPR. Therefore, privacy protection was done openly and regulated effectively. Yet, the Man in the Middle attack could not be prevented, affecting the transactions.

[16] offered a secure E-commerce scheme for online transactions. Here, an E-commerce protocol known as a Secure E-commerce Scheme (SES) was utilized. To handle the sensitive data during CCT, the SES was used. The SES was simulated, and the data were secured. SES ensured that the user's details were not stored in the merchant's server, eliminating the breach of personal data. But the availability of the merchant's products was not verified, which decreased the purchase latency.

[17] established CCT using facial recognition technology. Here, for each CC customer, the facial image database was created and stored. The Haar Cascade algorithm was utilized in real-time for facial recognition. The transaction was continued for the matched image and declined if the face did not match. Therefore, the likelihood of fraudulent transactions was avoided. Nevertheless, the model relied on a static key for facial recognition, which resulted in the misuse of the database during verification.

[18] accomplished an E-commerce payment model using blockchain. Here, the direct peer-to-peer payments were done using a blockchain ledger. During the transaction, the MAC-based authentication was done. The CC data was also protected using public and private keys. Therefore, the authenticity during CCT was maintained effectively. Yet, the malicious SQL could be injected during the transaction, affecting the overall purchase of the product via an E-commerce application.

[19] examined CCT using the card-not-present method and secured transactions. To minimize the user data's dimensionality, the t-distributed Stochastic Neighbor Embedding (t-SNE) was used. Next, to extract the most important feature, the Principal Component Analysis (PCA) was used. Lastly, to prevent unauthorized transactions, Logistic Regression Learning (LRL) was used. Yet, the skimming attack could not be prevented by the model, leading to insecure transactions.

[20] assessed Federated Learning and blockchain for privacy-preserving CCT. The central cloud server was initialized during Federated Learning, and the parameters were sent to the local model via fog nodes. Then, each local model was trained to prevent data leakage. Then, for validating the CCT, the blockchain was used. Therefore, the sensitive data from the users was protected effectively. Nevertheless, the scalability issue occurred while handling large numbers of users.

[21] arranged contactless CC payment via ambient authentication. The keys were shared between the user and the merchant cryptographically. Next, based on these keys, mutual authentication was carried out. Afterward, by using sound signatures and Bluetooth footprints, ambient authentication was done to avoid the fake reader during CCT. After successful authentication, the product purchase was done via CC payment. Hence, the skimming attack was avoided. Yet, the latency in transaction processing time was increased due to back-to-back authentication.

[22] discovered Financial Technology (Fintech)-based money transactions via mobile for product purchases. Here, the Multi-Factor Authentication (MFA) was done using biometrics and a Personal Identification Number (PIN). Next, for transaction verification, Quick Response (QR) codes were used. After the verification, the CCT was continued. Therefore, the transaction security was increased effectively. However, the model failed to perform secure transactions in the real-time FinTech environment.

[23] confirmed CC fraud detection for secured CCT. Here, the CCT details were collected from the database. Next, the important features were selected by using a Genetic Algorithm (GA). Also, for detecting the attack in CC details, the Decision Tree (DT), Random Forest (RF), Logistic Regression (LR), Artificial Neural Network (ANN), and Naive Bayes (NB) were utilized. However, the authentication during CCT was not considered, which reduced the customers' trust in CCT.

[24] assessed the security of transaction information in an E-commerce application. Here, using a combination of the Fernet and ElGamal algorithms, the transaction details were secured. Initially, the data was encrypted using the Fernet method, and the key was protected using the ElGamal technique. This provided additional security during CCT. Therefore, the model was efficient for real-time E-commerce use. Yet, the memory usage was higher and was resource-constrained for devices like smartphones.

[25] restored the consumer trust in e-commerce via blockchain technology. Here, the blockchain trust beliefs like privacy ownership, privacy monetization, metadata transparency, and illicit access were stored in the blockchain platform. Further, based on the willingness to pay additional fees and retail use intentions, the consumer's purchase in the online store increased, and the zero-knowledge proof was enhanced. However, the details during transactions were not secured, causing data leakage. [26] presented on-device privacy preservation for smart consumers using Federated Learning (FL). In this approach, client-side data was prepared. Next, the batching, shuffling, and formatting were carried out to analyse the pattern of the collected data. Finally, the data privacy through decentralization was attained using FL. Thus, the transaction success rate improved. On the contrary, this model did not support real-time processing.

[27] developed an explainable e-commerce transaction prediction using an Artificial Intelligence (AI) model. The data were collected from the e-commerce platforms. Further, the duplicate data were removed and passed into the Light Gradient Boosting Machine (LightGBM). The interpretability was also provided for a detailed explanation regarding data-driven decision-making. Yet, the critical security in CCT was not maintained, losing trust among consumers.

3. Proposed Secured Credit Card

3.1. Transaction Methodology

In the proposed work, the dynamic CVV is created, the selected product is verified, IP spoofing of the webpage is

done, and the CC details are secured for effective CCT. The important steps involved in the proposed framework are customer registration, key generation, secret key creation, merchant signature creation and verification, dynamic CVV generation, IP spoofing of E-commerce applications, and security of the CC information. In Figure 1, the proposed work's architecture is illustrated. The proposed architecture integrates user authentication, dynamic CVV generation, secure communication, and credential security. The synchronization between the dynamically generated CVV and cryptographic keys is achieved via the usage of PFYHD-CVV and XORMP-MQV.

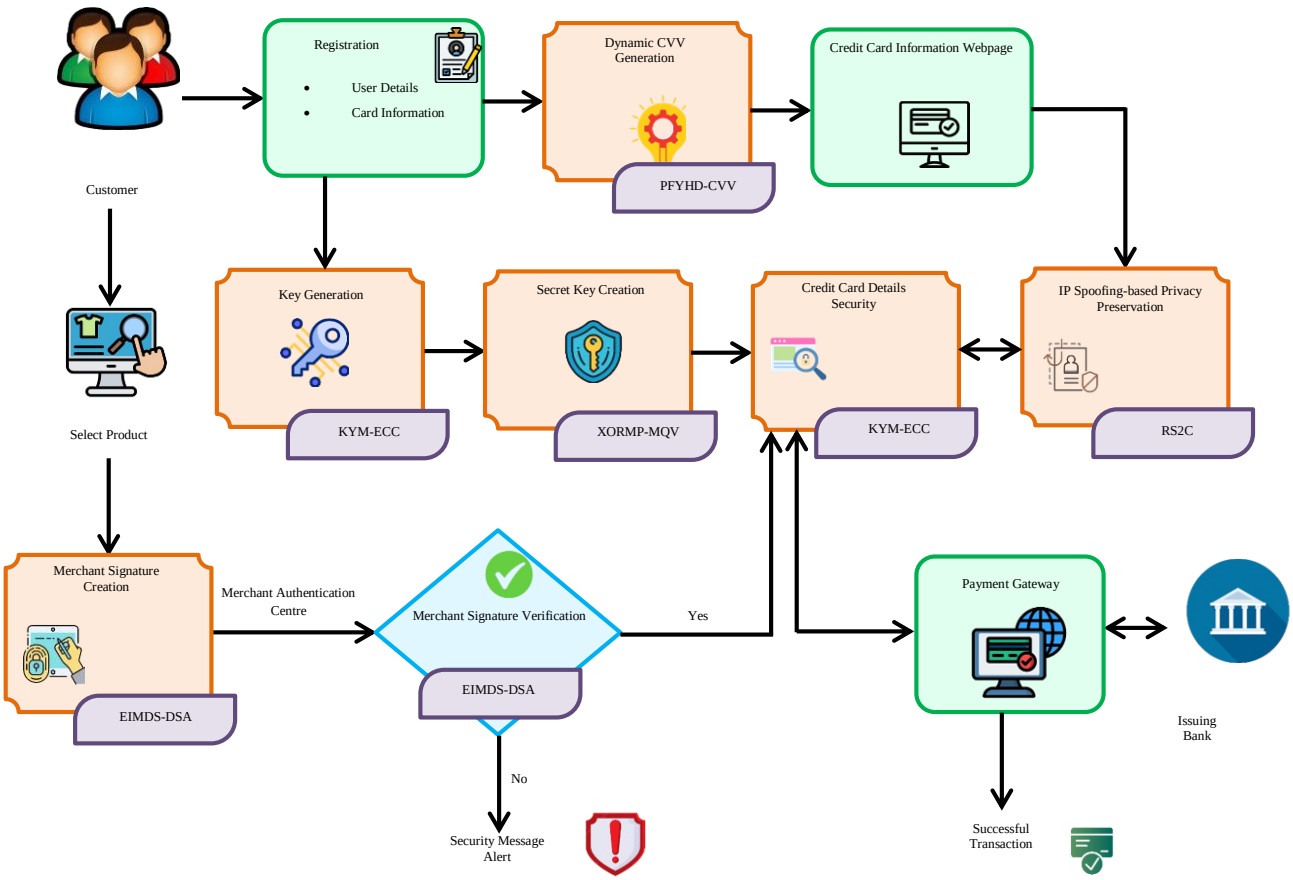

Fig. 1 Structure of the Proposed System

This ensures that each transaction is operated with unique and temporarily synchronized credentials. The protocol flow is maintained to provide consistency between client-side and merchant-side authentication processes regarding CCT.

As the user authentication and merchant verification are completed, replay attacks and key leakage attacks are prevented. And, the RS2C-based IP spoofing protocol secures the actual network addresses of the client and merchant during payment processing.

This significantly reduces the risk of a Man-in-the-Middle attack. Finally, the KYM-ECC-based encryption, which is based on the secured wireless payment protocol, secures the CC details and mitigates SQL injection and data leakage. Thus, the interaction of these protocol components creates a multi-layered security approach that maintains robustness and reliable execution of secure CCT.

3.2. Customer Registration

The proposed privacy preservation with CCT work starts with the registration of the customer into the E-commerce application for product purchase. During registration, the user details, such as user name(H^1), Identity (ID) number(H^2), age(H^3), sex(H^4), and address(H^5), and CC information like 16-digit CC number(H^6) and 3-digit CVV(H^7), are entered. The registration details(H)are expressed as,

$$H = (H^1, H^2, H^3, H^4, H^5, H^6, H^7) \quad (1)$$

After registration, the CVV is generated dynamically.

3.3. Dynamic CVV Generation

Here, the PFYHD-CVV is utilized for the dynamic generation of the customer's CVV(H^7). As the CVV number length is small, it is vulnerable to shoulder surfing and web skimming by hackers during CCT. To solve this issue, the

CVV number is dynamically generated and sent as the message for every transaction. Also, the Fisher-Yates (FY) shuffling algorithm, which generates one-time use CVV, is utilized, thus invalidating the CVV for future transactions.

The FY generates and shuffles the CVV to create various combinations of numbers. Yet, due to the pseudo-random number in FY, the data is shifted and creates repeated combinations of CVV. So, to avoid this problem, the Permutation process is included in FY, and the generated combination of CVV numbers is converted into Hexa Decimal (HD) number. The process of PFYHD-CVV is explained below,

Initially, the CVV$(H^7)$is converted into an array(α)using the Permutation process that rearranges the order of the digits(H^7). It is expressed as,

$$\alpha = \frac{b!}{(b-H^7)!} * H^7 \tag{2}$$

Here, (b)implies the total number of digits in(H^7). Now, the uniform random Permutation(γ)number is attained based on the swapping of $(\alpha)$the elements (b), which is given as,

$$\gamma = \alpha * f \forall [f \in (b-1, b)] \tag{3}$$

Here, (f)signifies the present and previous digit in(H^7). Thus, the shuffled digit is converted into a single-digit number(γ). Finally, the digital number$(\gamma)$is converted into an HD number, which masks the origin of the CVV and increases the complexity. Hence, the dynamically created CVV$(D)$is equated as,

$$D = \sum \gamma * 10^{(b-1)} * \beta(\gamma) \tag{4}$$

Here, $(\beta)$the HD string. This $(D)$is utilized for single use during the CCT. The user obtains a dynamic CVV(D) as a message, and after the successful verification of the merchant's product, the registered customer(V) enters the dynamic(D)CVV. It is expressed as,

$$V \rightarrow [D, H]\vartheta \tag{5}$$

Here, $(\vartheta)$the E-commerce application is implied, where the products are chosen, and the CCT is carried out. Since each transaction uses unique synchronization, the adversaries cannot reuse the information, thereby preventing the replay attacks. Next, the key is generated for the encryption of the CC details.

3.4. Key Generation

After the registration of the user into the product purchase application, the key is generated using KYM-ECC. The ECC performs faster encryption and decryption. It is also stronger than RSA; therefore, it is used for key generation and security of the CC details. However, the generation of private keys using random values can compromise the keys, leading to the loss of information. Thus, to overcome this issue, the private key is generated using the Kaplan-Yorke Map (KYM) function. The procedure for key generation is described as follows,

3.4.1. Curve Creation

Primarily, the elliptic curve(χ), which is the algebraic structure and supports group operations like addition, is created regarding the variables(p, q)over the finite field and constants(h_1, h_2), making them ideal for cryptographic use. The elliptic curve(χ) is denoted as,

$$\chi \rightarrow q^2 = p^3 + (h_1 * p) + h_2 \tag{6}$$

3.4.2. Public and Private Key Generation

Now, the public key$(I)$and private key(J) used for securing the CC details of the user are created. Here, to avoid the compensation of the private key(J), the KYM function that utilizes the chaotic control parameter(δ) to generate the private key with an unpredictable sequence is utilized. The key generation is evaluated as,

$$I = (h_1, h_2) * \chi \tag{7}$$

$$J = 1 - \delta(I) + \chi * mod(I) \tag{8}$$

Therefore, after the successful registration, the public key$(I)$and private key(J) of the customer are generated. These keys are further stored in the merchant Authentication Center(K), as shown below:

$$K = V(I, J) \tag{9}$$

These keys are utilized for the security of the CC information. Next, the secret key is generated.

3.5. Secret Key Creation

Concurrently, by using the XORMP-MQV algorithm, the secret key is created. Here, the Menezes-Qu-Vanstone (MQV), which is a cryptographic method that securely establishes the shared secret between the merchant and the customer, is utilized for the creation of a secret key. However, the created keys can be static.

So, to avoid this issue and improve the complexity of the MQV algorithm, the merchant ID, product ID, and customer ID are converted into binary format; then, the Exclusive OR (XOR) operation is performed between these binary numbers. Then, the XOR value is converted into a decimal number and utilized in the MQV technique. This secret key is generated at the time of product selection for purchase and sent to the customer. The process of the XORMP-MQV technique is detailed as follows,

Principally, the merchant ID(E), product ID(g), and customer ID(H^2)are converted into binary form for effective processing, which is expressed as,

$$E'' = \phi(E) \tag{10}$$

$$\tilde{g} = \phi(g) \tag{11}$$

$$\ddot{H}^2 = \phi(H^2) \tag{12}$$

Here, (ϕ)implies the binary string that is zero-padded, and $(E'', \tilde{g}, \ddot{H}^2)$depicts the binary form of the merchant ID, product ID, and the customer ID, respectively. Now, the bitwise XOR operation is performed $(E'', \tilde{g}, \ddot{H}^2)$ to improve the complexity in the creation of the secret key. The bitwise XOR value(M)is represented as,

$$M = E'' \oplus \tilde{g} \oplus \ddot{H}^2 \tag{13}$$

After the addition of the binary ID values of the merchant, product, and customer,(M)it is converted into a decimal number(ε), which is used as a dynamic parameter during the creation of the secret key. The decimal number(ε)is equated as,

$$\varepsilon = I + mod(M) * \eta \tag{14}$$

Here, (I)implies the public key, as derived in Section 3.3, and (η)is the decimal string. Finally, the secret key(Z)is computed based on(ε)the ephemeral key(u) and the long-term key(v)of the MQV as,

$$Z = \varepsilon \oplus u \oplus v \tag{15}$$

Thus, the secret key(Z)is automatically generated and sent as a message to the customer after the product selection and verification. Henceforth, the customer utilizes the public key(I), private key(J), and secret key(Z) for the privacy-preserved CCT. Next, the product selection and merchant signature creation and verification are done.

3.6. Select Product & Merchant Signature Creation

After successful registration and generation of keys and a dynamic CVV, the customer selects the product for purchase.

At that time, the system must check whether the selected product is linked to the registered merchant or not. For the purpose of merchant signature creation, the EIMD5-DSA is utilized.

The Digital Signature Algorithm (DSA), which generates the digital signature quickly and effectively, is utilized. However, the hashing in DSA will not be complex and will cause improper signature creation and verification.

Therefore, to avoid this problem, the Message Digest 5 (MD5), which provides a hash value faster with low memory usage, is used in DSA.

However, the MD5 can generate the same hash code. So, to create a unique hash code, the Entropy Inverse (EI) technique is used. In Figure 2, the authentication of the merchant's product by the customer is depicted.

Initially, by using the username and password, the customer logs in to the E-commerce application. Then, the registered customer(V)selects the product(A) for purchase from the E-commerce application(ϑ). It is represented as,

$$V \xrightarrow{selectproduct} A[\vartheta] \tag{16}$$

The merchant ID(E) and the product ID(g) are combined after the selection of the product by the customer, and it is represented as(G).

$$G = E + g \tag{17}$$

Now, the EI that makes the message(G)less predictable is utilized to maximize the uniqueness in hash value generation. The EI-based message$(\tilde{G})$is equated as,

$$\tilde{G} = \kappa \oplus d \times \left(\frac{1}{\kappa}\right) \tag{18}$$

$$\kappa = -\sum \lambda(G) * log[\lambda(G)] \tag{19}$$

Here, (κ)implies the entropy value of(G), (d)depicts the binary function, and (λ)implies the probability of(G). Now, the hash code(m)is created via MD5 $(\tilde{G})$. Primarily, the 32-bit buffer(U)is generated as,

$$U = \{U_1, U_2, U_3, U_4\} \tag{20}$$

Where (U_1, U_2, U_3, U_4) implies the buffers are utilized for creating the unique hash. Next, $(\tilde{G})$ it is split into sixteen 32-bit chunks($y \to 0to15$) for the creation of a hash code. It is expressed as,

$$y = [y^0, y^1, y^2, \ldots, y^{15}]\tilde{G} \tag{21}$$

Now, the modular addition and bitwise rotation are performed (y) to attain the unique hash code(m). It is evaluated as,

$$m = \{y \oplus U\} * \mu \tag{22}$$

Here, (μ) the rotation function is depicted. Now, with the help of the generated public key(I), as explained in Section 3.3, and the hash code(m), the merchant signature($\Im$)is generated, as shown below,

$$\Im = G(I \oplus m) \tag{23}$$

Therefore, the merchant signature is created after the customer selects the product from the E-commerce application. Next, $(\Im)$ it is verified as follows:

3.6.1. Merchant Signature Verification

Then, this signature$(\Im)$ is given to the merchant Authentication Center(K) for verification. During the verification, the signature$(B)$is created on the merchant's Authentication Center's side, and it is represented as:

$$B(K) = G(I'' \oplus m'') \tag{24}$$

Here, (I'', m'') depict the public key and the hash code stored in the merchant Authentication Center, respectively. At last, the signature verification$(S)$is done regarding $(\Im)$and (B)as,

$$S = \begin{cases} S^1 \forall (\Im = B) \\ S^2 \forall (\Im \neq B) \end{cases} \tag{25}$$

The condition states that if $(\Im)$and $(B)$are equal, then the center considers the merchant to be the registered merchant(S^1); otherwise, the merchant is said to be a non-registered/fake merchant(S^2), for(S^1) the CCT is continued. Moreover, if (S^2) it is attained, then the security message alert is given to the center and the customer. As the customer receives the alert message, they choose another product, and the process of merchant signature creation and verification continues. The pseudo-code for EIMD5-DSA regarding merchant signature creation and verification is given below,

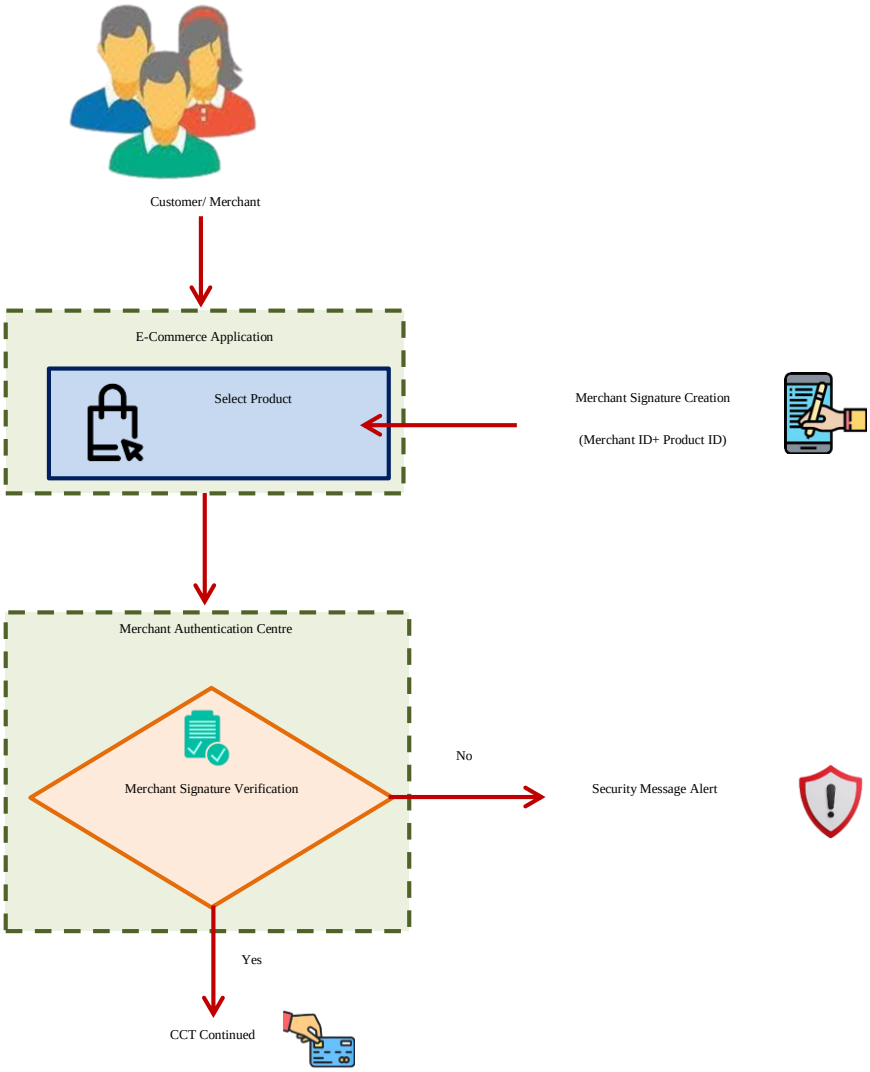

Fig. 2 Merchant Signature Creation and Verification Procedure

Pseudo-code for EIMD5-DSA

Input: Merchant ID(E), product ID(g)
Output: Signature verification(S)

Begin
Product selection
$V \xrightarrow{selectproduct} A[\vartheta]$
Combine merchant and product ID, $G = E + g$
For(G)
Evaluate entropy $\kappa = -\sum \lambda(G) * log[\lambda(G)]$
While(κ)
CalculateEI-based message
$$\tilde{G} = \kappa \oplus d \times \left(\frac{1}{\kappa}\right)$$
End while
For $U = \{U_1, U_2, U_3, U_4\}$
Split $(\tilde{G})$
$y = [y^0, y^1, y^2, \ldots, y^{15}]\tilde{G}$
Generate a unique hash code.
$m = \{y \oplus U\} * \mu$
Create merchant signature
$\Im = G(I \oplus m)$
End for
Estimate signature at (K)
$$B(K) = G(I'' \oplus m'')$$
//Signature verification
If $(\Im = B)$
Registered merchant(S^1)
Else if$(\Im \neq B)$
Fake merchant(S^2)
End for
Obtain Signature verification(S)

End

Thus, during successful merchant verification, the dynamic CVV and secret key introduce temporal variability in authentication details, preventing the replay attacks and reducing the effectiveness of credential theft. CC information is automatically entered into the webpage application after attaining(S^1) it.

3.7. Credit Card Information Webpage

Here, as explained in Section 3.2, the CC details, such as the customer's user name(H^1), Identity (ID) number(H^2), age(H^3), sex(H^4), and address(H^5), and CC information, like the 16-digit CC number(H^6) and the dynamic CVV(D), are passed into the E-commerce website for CCT. All the CC information is represented as(Q).

$$Q = (H^1, H^2, H^3, H^4, H^5, H^6, D) \tag{26}$$

Also, to avoid the Man in the Middle attack, the E-commerce webpage IP spoofing is done.

3.7.1. IP Spoofing-based Privacy Preservation

After merchant verification, the card information collection webpage(R) for the CCT is primarily verified. It checks whether the webpage is a secure webpage or not.

$$T(R) = \begin{cases} (i) if URL starts with "https://" \\ (j) if URL starts with "http://" \end{cases} \tag{27}$$

Here, (T)implies the webpage security verification output, $(i)$is the secured condition when the Uniform Resource Locator (URL) starts with "https://", and $(j)$is the non-secured webpage condition when the URL starts with "http://". If (j) it is attained, then the webpage for CCT is not secured, and the alert message(ρ) is sent to the customer(V). It is expressed as,

$$\rho \xrightarrow{notified(j)} V \tag{28}$$

$$i \xrightarrow{continue} CCT \tag{29}$$

For the purchase of the selected product from the E-commerce application, the CCT is continued(i). According to the notification attained(V), the customer decision handling(X) is done to decide whether to continue the transaction or not. It is denoted as,

$$X = \begin{cases} X'' \\ \ddot{X} \end{cases} \tag{30}$$

Here, (X'')implies the condition applied by the customer to continue the transaction via the non-secured webpage(j), and$(\ddot{X})$ is the condition to cancel the CCT by the customer. By using RS2C, which transforms the original IP address into a completely different binary pattern, the sender and receiver's IP addresses are converted into a spoofed format when(j)it is attained, and the customer(V) chooses(X''). It prevents the attackers from easily identifying the sender's or receiver's location. The process of RS2C is given below,

The sender and receiver's IP addresses are denoted as(k, l), respectively. These IP addresses are converted into binary format and shifted to the right by 1 bit as follows,

$$\dddot{k} = (k * \tau) >> 1 \tag{31}$$

$$\ddot{l} = (l * \tau) >> 1 \tag{32}$$

Here, $(\ddot{k}, \ddot{l})$imply the right-shifted sender and receiver's IP addresses, respectively, and$(\tau)$is the binary conversion factor. Further, each bit of $(\ddot{k}, \ddot{l})$is inverted (1's complement) using the bitwise NOT operator(r)as,

$$\tilde{k} = r(\ddot{k} >> 1) \tag{33}$$

$$\tilde{l} = r\left(\dot{l} >> 1\right) \quad (34)$$

Here, $(\tilde{k}, \tilde{l})$depict the inverted sender and receiver's IP addresses, respectively. Now, the spoofing of the IP addresses is done regarding 2's complement as shown below,

$$k^* = r\left(\tilde{k} >> 1\right) + 1 \quad (35)$$

$$l^* = r\left(\tilde{l} >> 1\right) + 1 \quad (36)$$

Where, (k^*, l^*)imply the spoofed IP address of the sender and receiver, respectively. By this spoofing, the CCT is done with privacy preservation even when the webpage is not secure. Thus, the RS2C-based IP spoofing model avoids Man-in-the-Middle and traffic analysis attacks during payment processing. Finally, the CC information(Q) is secured for effective CCT.

3.8. Credit Card Details Security

After successful merchant verification, the dynamic CVV number(D) and secret key$(Z)$are generated (as mentioned in Sections 3.2 and 3.4) and sent to the customer(V). Finally, the CC details$(Q)$are secured using KYM-ECC to achieve a secure wireless payment protocol. It is expressed as follows,

$$N = Q \oplus I \oplus J \oplus Z \quad (37)$$

Here, (N)depicts the encrypted CC information, and $(I, J)$are the public key and private key generated using KYM-ECC, respectively, as explained in Section 3.3. Thus, the CC details are preserved against the SQL injection attack. The pseudo-code for KYM-ECC is detailed below,

Pseudo-code for KYM-ECC

Input: CC details(Q)
Output: Encrypted CC information(N)

Begin
  Create Elliptic curve
    $\chi \rightarrow q^2 = p^3 + (h_1 * p) + h_2$
  For(χ)
  While$(q, p \in \chi)$
  Calculate Public Key
    $I = (h_1, h_2) * \chi$
  Determine Private Key using KYM.
    $J = 1 - \delta(I) + \chi * mod(I)$
  Store (I, J) into merchant Authentication Center(K)
    $K = V(I, J)$
  For(I, J)
  Encrypt CC details for secured CCT
  $N = Q \oplus I \oplus J \oplus Z$
  End for
  End while
  Obtain Encrypted CC information(N)
End

After securing the CC details, the CCT is continued. Thus, by KYM-ECC, the brute-force, key leakage, and SQL injection attacks are prevented.

3.8.1. Payment Gateway & Issuing Bank

The CC details are protected and sent via the secured payment gateway for CCT. Lastly, the encrypted CC information is decrypted at the issuing bank as follows,

$$N^* = Q - I - J - Z \quad (38)$$

Here, (N^*)implies the decrypted CC information. The issuing bank checks the credit card information(N^*) and initiates further processing. After checking the details, the CCT is made successful, and the customer purchases the product from the E-commerce application. Thus, the proposed work privacy-preserved CCT by avoiding web skimming, shoulder surfing attacks, SQL injection attacks, and Man in the Middle attacks. The performance of the proposed framework is given in Section 4.

4. Results

Here, the proposed methods' performance is evaluated and compared with the existing techniques to show the improvement of the model. The implementation of the proposed approach is done using JAVA, which is a versatile, object-oriented programming language designed to be platform-independent through the "Write Once, Run Anywhere" (WORA) principle.

4.1. Dataset Description

The "Credit Card Transactions Fraud Detection (CCTFD) Dataset" [28] and the "Credit Card Fraud Detection (CCFD)" dataset [29] are used in the proposed system for secured CCT. The CCTFD dataset consists of details, such as transaction date, transaction time, CC number, merchant name, amount, gender, street, city, state, zip code, latitude, longitude, and so on. The CCFD dataset consists of information like transaction ID, transaction date, amount, merchant ID, transaction type, location, and so on. These details are used for encryption and decryption processes. The dataset link is provided in the reference section.

4.2. Performance Validation Regarding CCTFD Dataset

Here, the performance of the proposed techniques, such as PFYHD-CVV, KYM-ECC, EIMD5-DSA, and XORMP-MQV, is assessed and compared with the prevailing models.

Table 1 compares the performance of the proposed PFYHD-CVV with the existing models regarding the creation of dynamic CVV. The proposed technique utilized the Permutation function and the HD numbering to generate unique CVV and avoided the web skimming and shoulder surfing attack. Henceforth, the PFYHD-CVV model attained a CVV number generation time of 750ms. On the other hand, the existing Fisher-Yates-shuffle-CVV (FY-CVV), Riffle-

shuffle-CVV (R-CVV), Mongean-shuffle-CVV (M-CVV), and Pile-shuffle-CVV (P-CVV) attained a CVV number generation time of 1295ms, 1983ms, 2568ms, and 4019ms, respectively. This proved that the proposed model performed better than prevailing models regarding dynamic CVV number creation for secured CCT.

Table 1. Comparative Analysis of PFYHD-CVV

Methods	CVV Number Generation Time (ms)
Proposed PFYHD-CVV	750
FY-CVV	1295
R-CVV	1983
M-CVV	2568
P-CVV	4019

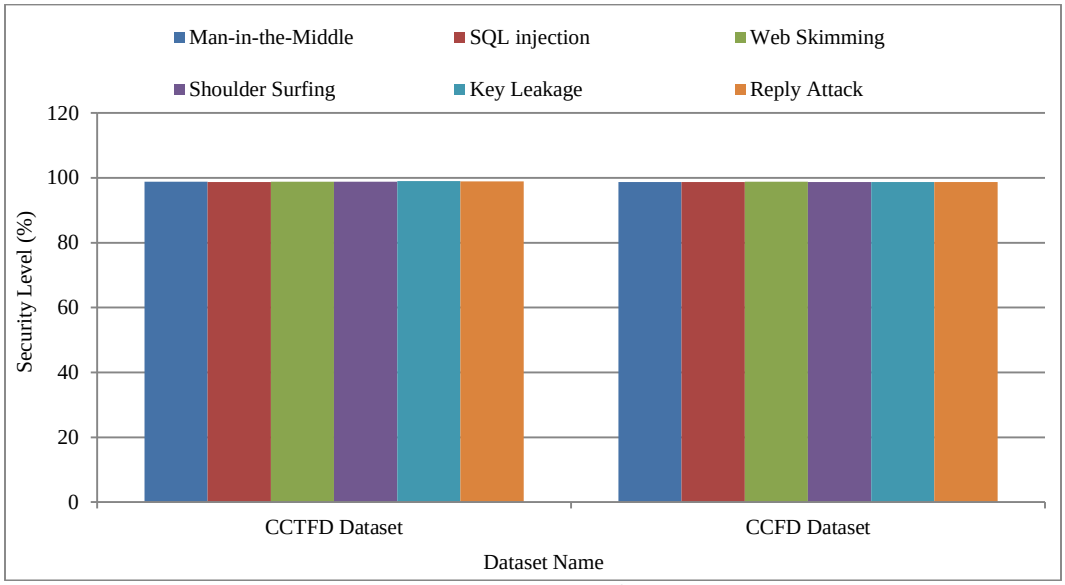

Fig. 3 Graphical Comparison of KYM-ECC

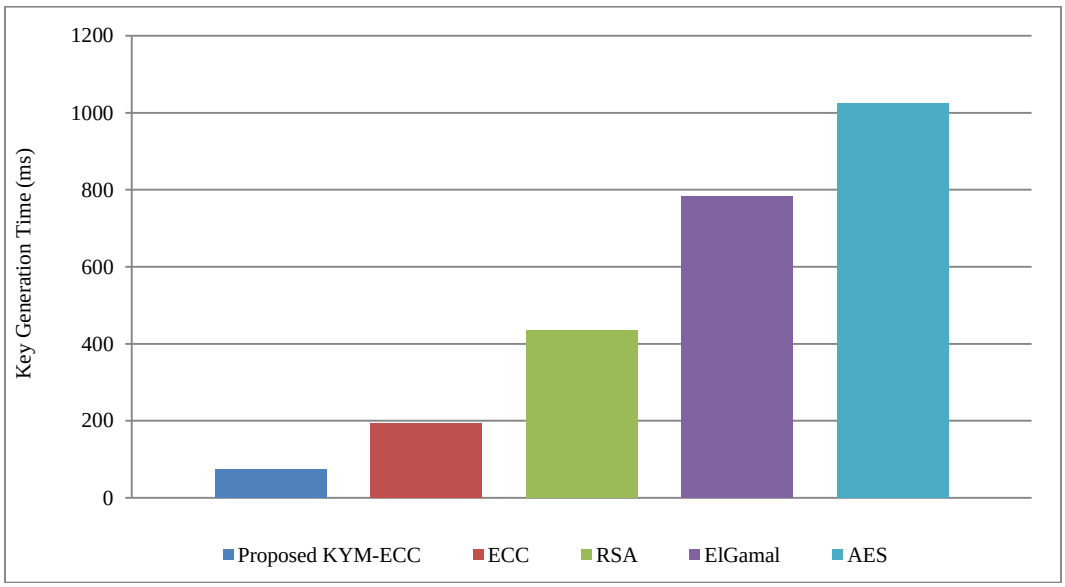

Fig. 4 Comparative Analysis Regarding Key Generation Time

In Figure 3 and 4, the proposed cryptographic method's performance is evaluated with the prevailing techniques regarding security level and Key Generation Time (KGT). As the private key was generated using the KYM function, the proposed model secured the CC information with a security level of 98.8564% and KGT of 75ms. Yet, the existing ECC,

RSA, ElGamal, and Advanced Encryption Standard (AES) attained a security level of 95.8471%, 92.6598%, 89.7451%, and 87.6532%, and KGT of 194ms, 436ms, 783ms, and 1025ms, respectively.

Therefore, the proposed KYM-ECC outperformed the traditional models regarding the security of the CC information of the customer and also avoided attacks like brute-force, key leakage, and SQL injection.

Table 2. Comparative Analysis of KYM-ECC

Techniques	Computational Complexity (ms)	Latency (ms)	Throughput (kb/s)	Energy Consumption (joule)
Proposed KYM-ECC	498	1045	4754	874
ECC	1024	1547	4458	1248
RSA	1754	1986	4051	1754
ElGamal	2241	2451	3658	2369
AES	2712	2865	3365	2784

The proposed KYM-ECC's performance is compared with the prevailing ECC, RSA, ElGamal, and AES regarding metrics, such as computational complexity, latency, throughput, and energy consumption in Table 2. The proposed technique effectively encrypted the CC details using the generated public key, private key, and secret key. Therefore, the proposed KYM-ECC obtained a computational complexity of 498ms, a latency of 1045ms, a throughput of 4754kb/s, and energy consumption of 874 joules.

Yet, the prevailing techniques like ECC, RSA, ElGamal, and AES achieved a computational complexity of 1024ms, 1754ms, 2241ms, and 2712ms, latency of 1547ms, 1986ms, 2451ms, and 2865ms, throughput of 4458 kb/s, 4051 kb/s, 3658 kb/s, and 3365 kb/s, and energy consumption of 1248 joules, 1754 joules, 2369 joules, and 2784 joules, respectively. Henceforth, the proposed KYM-ECC secured the CC details of the customer more securely than the prevailing models.

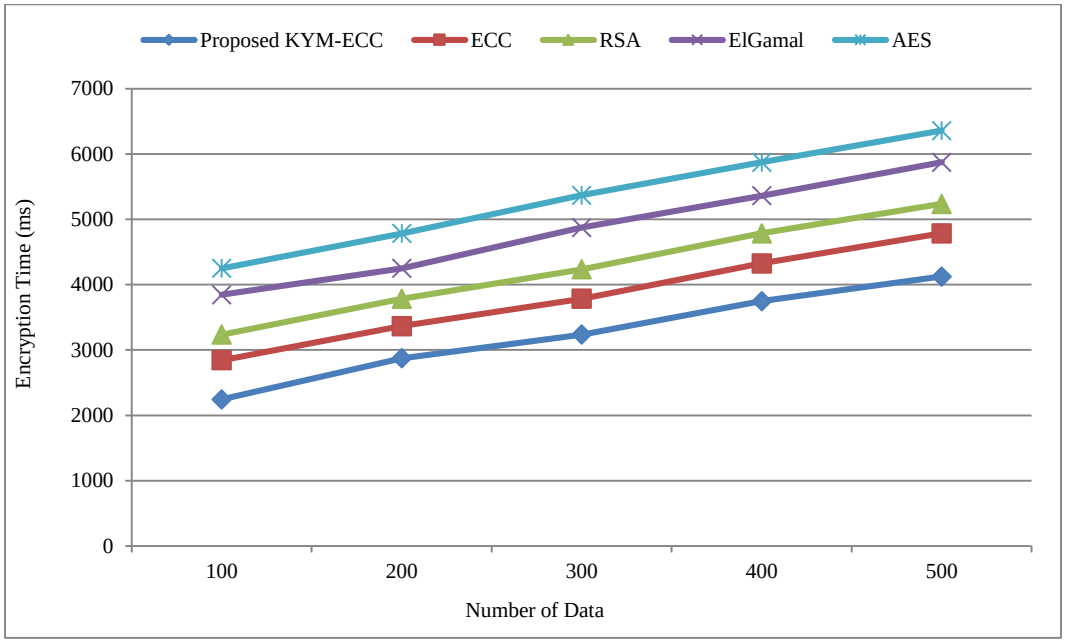

Fig. 5 Comparison regarding Encryption Time

The proposed KYM-ECC is graphically compared with prevailing techniques regarding the encryption and decryption of CC details in Figure 5 and 6.

The public and private keys were generated effectively by KYM-ECC, and the dynamic secret key was also utilized for the security of the CC information. Therefore, the proposed KYM-ECC achieved an encryption time of 2245ms, 2874ms, 3236ms, 3748ms, and 4125ms, and a decryption time of 2356ms, 2874ms, 3369ms, 3784ms, and 4458ms for 100, 200, 300, 400, and 500 numbers of data, respectively. Nevertheless, the prevailing techniques achieved an average encryption and decryption time of 3544ms and 3560ms for 100 data, 4046ms and 4035ms for 200 data, 4566ms and 4571ms for 300 data, 5086ms and 5074ms for 400 data, and 5563ms and 5585ms for 500 data, correspondingly. Therefore, the proposed model's encryption and decryption were better when weighed against prevailing techniques.

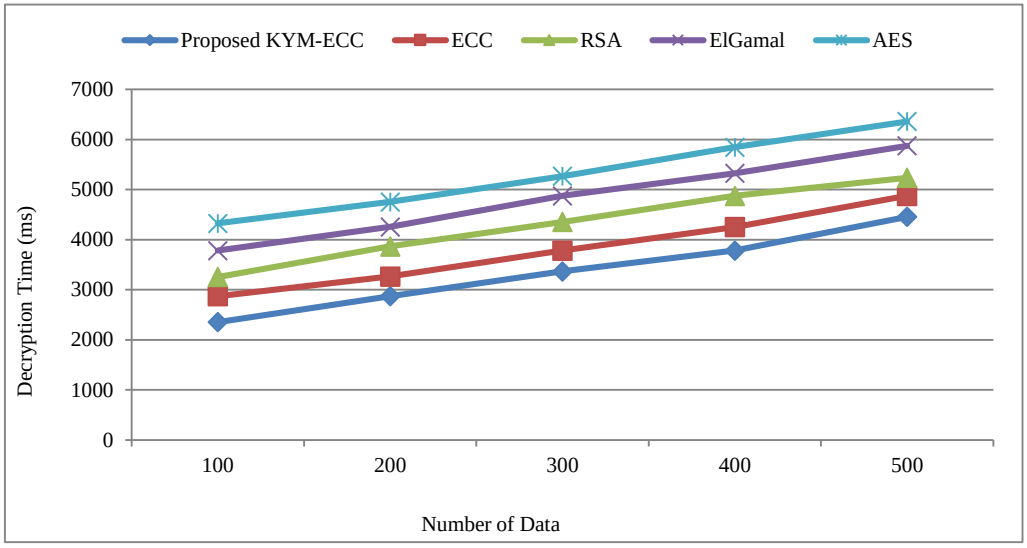

Fig. 6 Comparison regarding Decryption Time

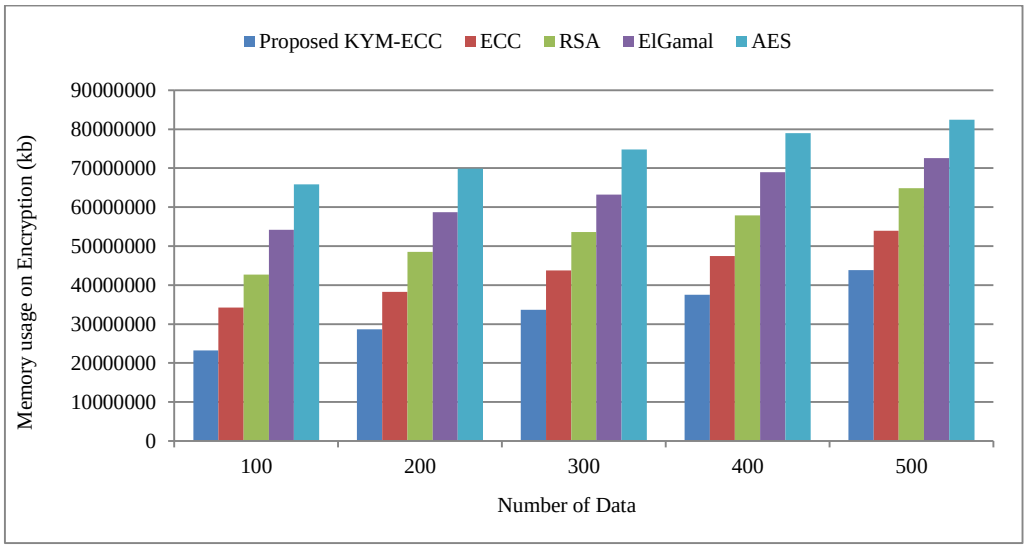

Fig. 7 Comparison regarding Memory usage on Encryption

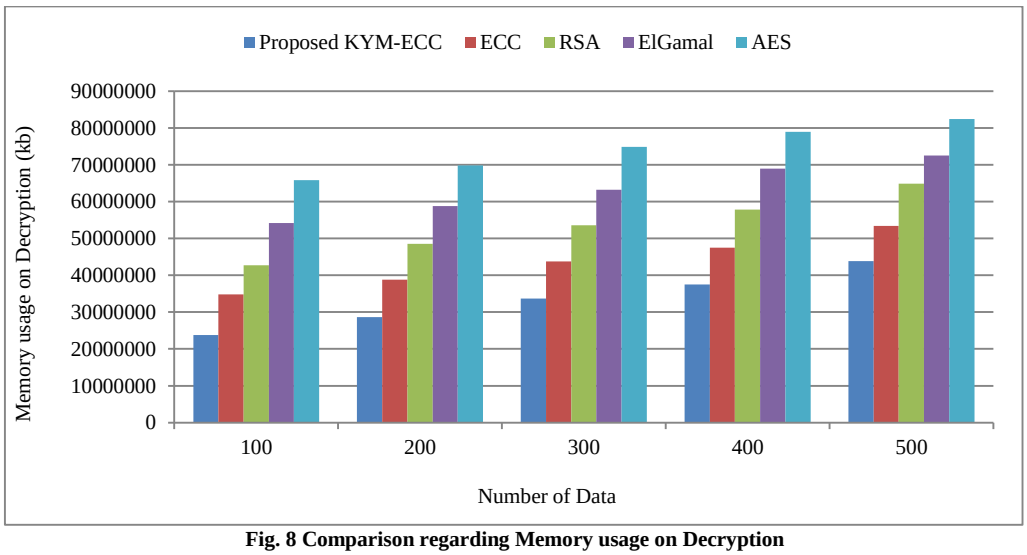

Fig. 8 Comparison regarding Memory usage on Decryption

As the private and public keys were created effectively based on the Elliptic curve and the KYM function, the proposed model used 23218756kb and 23748565kb for 100 data and 43875465kb and 43874512kb for 500 data during encryption and decryption, respectively. This KYM-ECC also avoided the SQL injection attack during CCT. On the contrary, the memory used by the existing ECC for encryption and decryption was 34254784kb and 34785421kb for 100 data and 53958645kb and 53412543kb for 500 data, respectively. Likewise, as shown in Figure 7 and 8, other existing techniques, such as RSA, ElGamal, and AES, also use higher memory for the encryption and decryption processes than the proposed KYM-ECC. Therefore, the proposed technique performed better than the prevailing models regarding scalability.

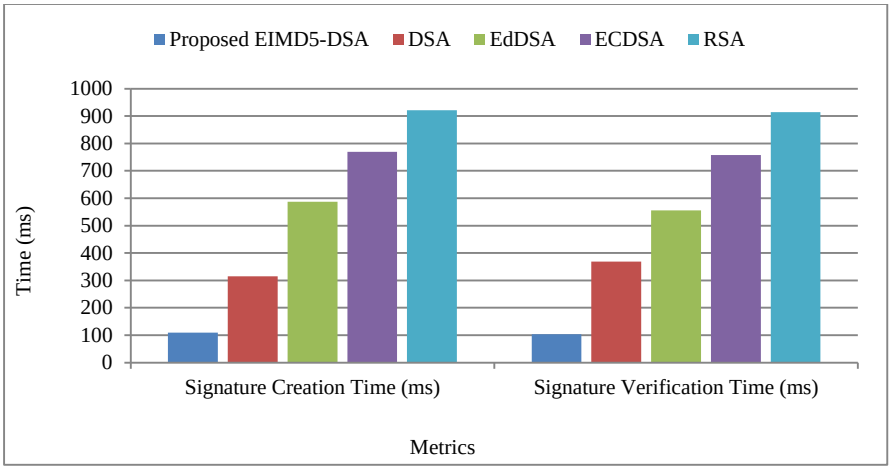

Fig. 9 Comparative Analysis of EIMD5-DSA

The proposed EIMD5-DSA is compared with the existing DSA, Edwards-curve Digital Signature Algorithm (EdDSA), Elliptic Curve Digital Signature Algorithm (ECDSA), and RSA regarding the merchant signature creation and verification in the merchant Authentication Center. As given in Figure 9, the proposed EIMD5-DSA and the prevailing DSA, EdDSA, ECDSA, and RSA achieved a Signature Creation Time (SCT) of 110ms, 315ms, 587ms, 769ms, and 921ms, and Signature Verification Time (SVT) of 104ms, 369ms, 556ms, 758ms, and 914ms, respectively. Thus, the usage of EI along with MD5 improved the proposed model's performance regarding the merchant product's creation and verification.

Table 3. Comparative Analysis of XORMP-MQV

Models	Secret Key Generation Time (ms)	Key Size (kb)
Proposed XORMP-MQV	512	19.8475
MQV	596	14.2658
SSEN	998	9.6578
WCDH	1458	4.5145
DH	1968	2.4578

Table 4. Existing Works Comparison

Study	Methods	Security Level (%)	Advantages	Drawbacks
Proposed Work	KYM-ECC	98.8564	Privacy preservation-based secured CCT.	During CCT, fraud detection was not considered.
[30]	TDCB-D3P	95	Security and scalability were attained.	Introduced complexity and training overhead.
[31]	EMV	95.34	Skimming attacks were avoided through dynamic authentication.	The card-not-present transaction was less effective.
[32]	FFX	90	Key management was effective.	Authentication was not done.
[33]	PSD2	-	Financial security regulations were maintained.	Failed to protect the customer's details.
[34]	MPS	-	The user's confidentiality was maintained.	This model was not generalizable globally.

In Table 3, a comparative evaluation of the proposed model and existing techniques regarding secret key generation is given. The proposed XORMP-MQV converted the ID of the merchant, product, and customer to a binary form in which the XOR operation was done, followed by transforming it to a decimal number. The generation of a secret key further helped in securing the CC details.

Therefore, the proposed XORMP-MQV generated the secret key within 512ms with a key size of 19.8475kb. However, the prevailing MQV, Secure Shell Extension Negotiation (SSEN), Weierstrass Curves Diffie-Hellman (WCDH), and Diffie-Hellman (DH) methods generated the secret key within 596ms, 998ms, 1458ms, and 1968ms, with key sizes of 14.2658kb, 9.6578kb, 4.5145kb, and 2.4578kb, respectively. This proved the efficiency of the proposed model over the existing methods regarding secret key creation.

The proposed work's performance is evaluated with existing works regarding the protection of the CC details for privacy-preserving and secured CCT in Table 4. In the existing works, the state-of-the-art methods, such as Trust-centric Delegated Consensus Blockchain Dueling Double Deep-Q-network with Prioritized experience replay (TDCB-D3P), Europay Mastercard and Visa (EMV) technology, Format-preserving Feistel-based Variable encryption (FFX), Payment Services Directive 2 (PSD2), and Mobile Payment Service (MPS) technology, were utilized. Unlike the existing models that focused on blockchain and static encryption, the proposed system integrated CVV generation dynamically and secured the communication via a unique protocol. The proposed model generated the public, private, and secret keys effectively and secured the CC details for CCT.

Therefore, a security level of 98.8564% was attained. Yet, the existing [30-32], models were either complex or not optimal for online transactions, or the authentication was not proper, attaining a security level of 95%, 95.34%, and 90%, respectively. Similarly, prevailing [33, 34] had drawbacks like failure in the protection of customers' details and were not generalizable.

This proved that the proposed work achieved better performance regarding CC information security and privacy-preserved CCT, and also achieved better robustness and practical applicability in privacy-preserving CCT.

4.3. Performance Validation Regarding CCFD Dataset

The comparative analysis of the proposed model and the existing state-of-the-art models regarding key generation and CC details security is provided in Table 5. This analysis is done by processing the data from the CCFD dataset.

Table 5. Comparative Analysis of KYM-ECC

Techniques	Security Level (%)	Key Generation Time (ms)	Computational Complexity (ms)	Latency (ms)	Throughput (kb/s)	Energy Consumption (joule)
Proposed KYM-ECC	98.7352	81	532	1172	4574	902
ECC	95.7126	281	1742	1942	4281	1389
RSA	92.3621	562	2041	2385	3902	1948
ElGamal	89.2481	894	2642	2894	3521	2475
AES	87.3728	1147	3194	3291	3205	2904

The proposed KYM-ECC secured the CC details with a security level of 98.7352%, KGT of 81ms, computational complexity of 532ms, latency of 1172ms, throughput of 4574 kb /s, and energy consumption of 902 joules.

However, the existing EXX, RSA, ElGamal, and AES attained a security level of 95.7126%, 92.3621%, 89.2481%, and 87.3728%, KGT of 281ms, 562ms, 894ms, and 1147ms, computational complexity of 1742ms, 2041ms, 2642ms, and 3194ms, latency of 1942ms, 2385ms, 2894ms, and 3291ms, throughput of 4281 kb/s, 3902 kb/s, 3521 kb/s, and 3205 kb/s, and energy consumption of 1389 joule, 1948 joule, 2475 joule, and 2904 joule, respectively.

Thus, the usage of IP-spoofing-based privacy preservation and KYM function enhanced the performance of the proposed KYM-ECC.

4.4. Threat Analysis

The threat analysis of the proposed system regarding the security level is mapped in Figure 10. The proposed KYM-ECC secured the CC details with a security level of 98.8586%, 98.7578%, 98.8089%, 98.8375%, 98.9887%, and 98.8875% against attacks like Man-in-the-Middle, SQL injection, Web Skimming, Shoulder Surfing, Key Leakage, and Reply attacks, respectively, regarding the data from CCTFD Dataset.

Further, for the CCFD Dataset, the proposed model secured the credentials with a security level of 98.7387%, 98.6898%, 98.8078%, 98.7371%, 98.7092%, and 98.7291% against the Man-in-the-Middle, SQL injection, Web Skimming, Shoulder Surfing, Key Leakage, and Reply attacks, respectively. This proved that the proposed system effectively secured the CC details of the consumers from the threat.

Fig. 10 Threat Analysis regarding CCTFD Dataset and CCFD Dataset

4.5. Statistical Significance

The statistical analysis of the proposed framework with respect to the p-test is depicted in Figure 11 and 12. The proposed framework attained a p-value of 0.035 and 0.043 regarding CCTFD and CCFD datasets, respectively.

As the attained p-values were lower than the standard threshold value of 0.05, the values obtained after the analysis were statistically significant. Thus, the experimental results were reliable and robust regarding privacy preservation in CCT.

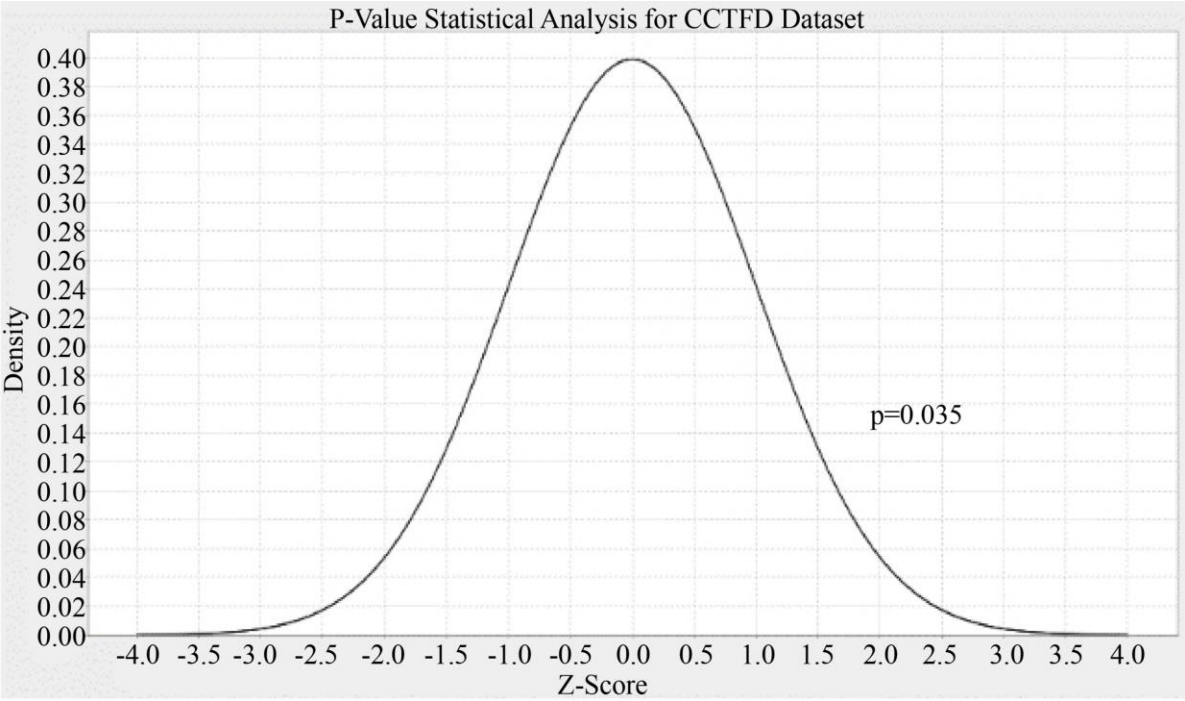

Fig. 11 Statistical Test regarding CCTFD Dataset

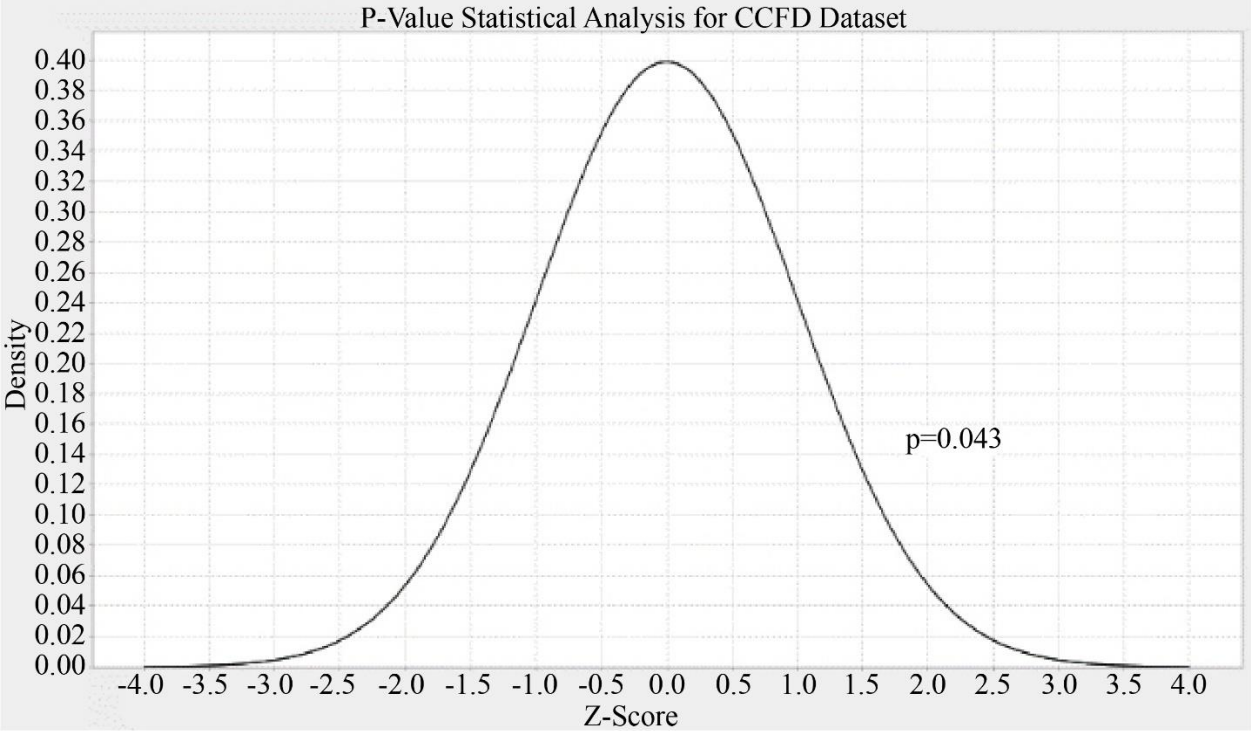

Fig. 12 Statistical Test regarding CCFD Dataset

5. Discussion

The superior performance of the proposed models and state-of-the-art techniques has been analyzed. The dynamic PFYHD-CVV approach reduced the risk of web skimming and shoulder surfing by eliminating static verification codes. And, the KYM-ECC-based key generation introduced unpredictability that reduced key compromise probability. Further, the EIMD5-DSA-based merchant verification and the XORMP-MQV-based dynamic secret key generation improved reliability and reduced latency. Experimental results on public datasets, such as CCTFD and CCFD, demonstrated improvement regarding security level, throughput, latency, and memory efficiency.

This proved the robustness of the proposed approach under realistic transaction scenarios. From threat analysis, it was proven that the proposed work effectively mitigated major attacks, including Man-in-the-Middle, SQL injection, web skimming, shoulder surfing, and key leakage attacks. This was done by a payment protocol that achieves security via dynamic CVV generation, RS2C-based IP spoofing, and KYM-ECC encryption. Moreover, the statistical significance analysis using the p-test showed that the performance of the proposed models was statistically meaningful, validating the reliability of the proposed models over the existing techniques.

6. Conclusion

Here, the paper presented an effective framework for secured CCT. Here, for the purchase of the product, the customer registers on the E-commerce application. After registration, the dynamic CVV was generated using PFYHD-CVV with a CVV generation time of 750ms. Now, by using KYM-ECC, the public and private keys were generated with a KGT of 75ms. Then, the secret key was generated using XORMP-MQV, which attained a key size of 19.8475kb. Also, the customer selected the product, and the merchant-product signature creation and verification were done using EIMD5-DSA, which achieved an SCT of 110ms and SVT of 104ms, respectively. Next, the CC details were automatically given to the payment webpage. Also, the unsecured webpage IP was spoofed. Lastly, the CC details were privacy-preserved for effective CCT using KYM-ECC, which obtained a security level of 98.8564%, a throughput of 4754kb/s, and a computational complexity of 498ms.

Lastly, by using KYM-ECC, the CC details were decrypted by the issuing bank with a decryption time of 4458ms for 500 user data. Therefore, the proposed system efficiently carried out the CCT for the purchase of the product via an E-commerce application. Although the CCT was done securely by the proposed system, fraud detection and prevention were not concentrated. Thus, during the CCT, fraud detection and the respective preventive measures will be considered in the future to improve the efficiency of the proposed system.

Practical Implication

In the real-world deployment, the proposed system effectively integrates the e-commerce and payment infrastructure. The design of the proposed PFYHD-CVV,

KYM-ECC, EIMD5-DSA, and XORMP-MQV enables interoperability with banking servers, payment gateways, and authentication models. Further, from the usability perspective, the dynamic CVV generation and automated key management ensure enhanced security and do not burden the end users.

The proposed approach is compatible with devices like smartphones, laptops, tablets, and Internet of Things (IoT)-based payments. Further, by providing trust and reducing the fraud risk, the user confidence and the adoption of secure digital payment systems have been improved.

Thus, the proposed model effectively supports the large-scale deployment in the modern financial environment more than existing approaches.

Regulatory and Ethical Compliance

The proposed system is designed with regulatory and security standards like Payment Services Directive 2 (PSD2), General Data Protection Regulation (GDPR), and Payment Card Industry Data Security Standard (PCI-DSS). Here, the secure authentication mechanisms support PSD2 requirements. Further, the encryption and minimization of data exposure during transactions contribute to GDPR principles. And, the processing of consumer data is done with the help of PCI-DSS. From the ethical point of view, data handling and fairness are ensured by preventing unauthorized access. No data are collected unethically, and the dataset used in this work is publicly available. No personal information is accessed during experimentation. All the experiments are done in a controlled environment.

References

[1] Saqib Saeed, "A Customer-Centric view of E-Commerce Security and Privacy," *Applied Sciences*, vol. 13, no. 2, pp. 1-22, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[2] Qianqian Li et al., "A Symmetric Projection Space and Adversarial Training Framework for Privacy-Preserving Machine Learning with Improved Computational Efficiency," *Applied Sciences*, vol. 15, no. 6, pp. 1-26, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[3] Mengna Yang, Yejun He, and Jian Qiao, "Federated Learning-based Privacy-Preserving and Security: Survey," *2021 Computing, Communications and IoT Applications*, Shenzhen, China, pp. 312-317, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[4] Mohammed Naif Alatawi, "Detection of Fraud in IoT based Credit Card Collected Dataset using Machine Learning," *Machine Learning with Applications*, vol. 19, pp. 1-16, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[5] A.F.M. Shahen Shah et al., "On the Vital Aspects and Characteristics of Cryptocurrency-A Survey," *IEEE Access*, vol. 11, pp. 9451-9468, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[6] Vishnu Laxman et al., "Emerging Threats in Digital Payment and Financial Crime: A Bibliometric Review," *Journal of Digital Economy*, vol. 3, pp. 205-222, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[7] Ibrahim Y. Hafez et al., "A Systematic Review of AI-Enhanced Techniques in Credit Card Fraud Detection," *Journal of Big Data*, vol. 12, no. 1, pp. 1-35, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[8] S. Siva Alagesh et al., "Privacy Preservation using Block chain for Credit Card Data," *2022 2nd International Conference on Innovative Practices in Technology and Management (ICIPTM)*, Gautam Buddha Nagar, India, pp. 725-730, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[9] Latifa Albshaier, Seetah Almarri, and M.M. Hafizur Rahman, "A Review of Blockchain's Role in E-Commerce Transactions: Open Challenges, and Future Research Directions," *Computers*, vol. 13, no. 1, pp. 1-42, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[10] Qiwei Han et al., "Towards Privacy-Preserving Digital Marketing: An Integrated Framework for User Modeling using Deep Learning on a Data Monetization Platform," *Electronic Commerce Research*, vol. 23, no. 3, pp. 1701-1730, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[11] S. Surya et al., "Protecting Online Transactions: A Cybersecurity Solution Model," *2023 3rd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*, Greater Noida, India, pp. 2630-2634, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[12] Rejwan Bin Sulaiman, Vitaly Schetinin, and Paul Sant, "Review of Machine Learning Approach on Credit Card Fraud Detection," *Human-Centric Intelligent Systems*, vol. 2, no. 1-2, pp. 55-68, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[13] Xiaopeng LUO et al., "The Utility Impact of Differential Privacy on Credit Card Data in Machine Learning Algorithms," *Procedia Computer Science*, vol. 221, pp. 664-672, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[14] Oluwabusayo Adijat Bello et al., "Machine Learning Approaches for Enhancing Fraud Prevention in Financial Transactions," *International Journal of Management Technology*, vol. 10, no. 1, pp, 85-108, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[15] Zlatan Morić et al., "Protection of Personal Data in the Context of E-Commerce," *Journal of Cybersecurity and Privacy*, vol. 4, no. 3, pp. 731-761, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[16] Sena Efsun Cebeci, Kubra Nari, and Enver Ozdemir, "Secure E-Commerce Scheme," *IEEE Access*, vol. 10, pp. 10359-10370, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[17] S. Gandhimathi, and J. Soundarya, "Credit Card Transaction Security using Facial Recognition Technology," *International Journal of Scientific Research in Computer Science Engineering and Information Technology*, vol. 11, no. 2, pp. 2358-2365, 2025. [CrossRef] [Publisher Link]

[18] Shee-Ihn Kim, and Seung-Hee Kim, "E-Commerce Payment Model using Blockchain," *Journal of Ambient Intelligence and Humanized Computing*, vol. 13, no. 3, pp. 1673-1685, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[19] Abdul Razaque et al., "Credit Card-Not-Present Fraud Detection and Prevention using Big Data Analytics Algorithms," *Applied Sciences*, vol. 13, no. 1, pp. 1-27, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[20] Tahani Baabdullah et al., "Efficiency of Federated Learning and Blockchain in Preserving Privacy and Enhancing the Performance of Credit card Fraud Detection (CCFD) Systems," *Future Internet*, vol. 16, no. 6, pp. 1-22, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[21] Ming-Hour Yang et al., "Contactless Credit Cards Payment Fraud Protection by Ambient Authentication," *Sensors*, vol. 22, no. 5, pp. 1-22 , 2022. [CrossRef] [Google Scholar] [Publisher Link]

[22] Habib Ullah Khan et al., "Role of Authentication Factors in Fin-Tech Mobile Transaction Security," *Journal of Big Data*, vol. 10, no. 1, pp. 1-37, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[23] Emmanuel Ileberi, Yanxia Sun, and Zenghui Wang, "A Machine Learning based Credit Card Fraud Detection using the GA Algorithm for Feature Selection," *Journal of Big Data*, vol. 9, no. 1, pp. 1-17, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[24] Mishall Al-Zubaidie, and Ghanima Sabr Shyaa, "Applying Detection Leakage on Hybrid Cryptography to Secure Transaction Information in E-Commerce Apps," *Future Internet*, vol. 15, no. 8, pp. 1-20, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[25] Louisa Uchikoshi, and Björn Frank, "Restoring Consumer Trust in E-Commerce: The Role of Blockchain Technology and its Behavioral Implications," *Technological Forecasting and Social Change*, vol. 224, pp. 1-14, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[26] Alexandros I. Bermperis et al., "On-Device Privacy-Preserving Fraud Detection for Smart Consumer Environments using Federated Learning," *Applied Sciences*, vol. 16, no. 2, pp. 1-16, 2026. [CrossRef] [Google Scholar] [Publisher Link]

[27] Wang Junhai, Wang Yunfeng, and Othman Ibrahim, "Explainable E-Commerce Transaction Prediction using LightGBM and Permutation Importance," *IEEE Access*, vol. 14, pp. 10153-10169, 2026. [CrossRef] [Google Scholar] [Publisher Link]

[28] Kartik Shenoy, Credit Card Transactions Fraud Detection Dataset, Kaggle, 2020. [Online]. Available: https://www.kaggle.com/datasets/kartik2112/fraud-detection

[29] Bhadra Mohit, Credit Card Fraud Detection Trends and Tactics in Modern Credit Card Fraud, Kaggle, 2024. [Online]. Available: https://www.kaggle.com/datasets/bhadramohit/credit-card-fraud-detection

[30] Yunyeong Goh et al., "Secure Trust-based Delegated Consensus for Blockchain Frameworks using Deep Reinforcement Learning," *IEEE Access*, vol. 10, pp. 118498-118511, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[31] Pavan Kumar Joshi, and Rajesh Kotha, "An In-depth Knowledge on EMV Tags and their Adoption in FinTech and Traditional Banking," *Journal of Scientific and Engineering Research*, vol. 10, no. 9, pp. 77-86, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[32] Jamal Habibi Markani et al., "Security Establishment in ADS-B by Format-Preserving Encryption and Blockchain Schemes," *Applied Sciences*, vol. 13, no. 5, pp. 1-16, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[33] Larisa Găbudeanu et al., "Privacy Intrusiveness in Financial-Banking Fraud Detection," *Risks*, vol. 9, no. 6, pp. 1-22, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[34] Yoonyoung Hwang, Sangwook Park, and Nina Shin, "Sustainable Development of a Mobile Payment Security Environment using Fintech Solutions," *Sustainability*, vol. 13, no. 15, pp. 1-15, 2021. [CrossRef] [Google Scholar] [Publisher Link]