Proposal of an End-to-End Encrypted Chat System with Digital Signature and Vietnamese Character Support Based on Elliptic Curve

DOI: https://doi.org/10.54654/isj.v3i26.1158

Nguyen Quoc Viet*, Ngo Manh Cuong

***Abstract*— The paper proposes an End-to-End Encrypted (E2EE) chat system based on Elliptic Curve Cryptography (ECC), using Elliptic Curve Digital Signature Algorithm (ECDSA) for authentication and Elliptic Curve Diffie-Hellman (ECDH) for encryption key generation. The system is designed as a lightweight web widget, easily embeddable into websites, with full support for Vietnamese characters (UTF-8) without compromising performance. The solution employs the Web Crypto API to perform AES-GCM encryption and ECDSA digital signatures, combined with a nonce mechanism to prevent replay attacks. Experimental results demonstrate high performance (encryption/decryption time <1ms for long messages) and resilience against MITM and spoofing attacks.**

***Tóm tắt*— Bài báo đề xuất một hệ thống chat mã hóa đầu cuối (E2EE) dựa trên đường cong Elliptic (ECC), sử dụng thuật toán chữ ký số đường cong Elliptic (ECDSA) để xác thực và ECDH để tạo khóa mã hóa. Hệ thống được thiết kế dưới dạng widget web nhẹ, dễ dàng nhúng vào website, hỗ trợ đầy đủ ký tự tiếng Việt (UTF-8) mà không làm giảm hiệu suất. Giải pháp sử dụng Web Crypto API để thực hiện mã hóa AES-GCM và ký số ECDSA, kết hợp cơ chế nonce chống tấn công phát lại. Kết quả thử nghiệm cho thấy hiệu suất cao (thời gian mã hóa/giải mã <1ms cho tin nhắn dài) và khả năng chống lại các tấn công MITM, giả mạo.**

Keywords*— *E2EE, ECC, ECDSA, ECDH, Vietnamese support; web widget.

Từ khóa*— *Mã hóa đầu cuối, đường cong Elliptic, ECDSA, ECDH, hỗ trợ tiếng Việt, widget web.

This manuscript was received on October 22, 2025. It was reviewed on December 10, 2025, revised on December 12, 2025 and accepted on December 22, 2025.

* Corresponding author

I. Introduction

A. Context

Online chat applications are increasingly prevalent in sensitive fields such as finance and healthcare, where protecting privacy and authenticating identity are essential requirements. Conventional systems are vulnerable to MITM, replay, or spoofing attacks. End-to-End Encryption (E2EE) and digital signatures are effective solutions, but their implementation on web platforms still faces challenges regarding performance, integration capability, and support for local languages like Vietnamese (UTF-8). This research proposes an E2EE system based on Elliptic Curve Cryptography to address the aforementioned challenges.

B. Novelty

The research proposes an End-to-End Encrypted chat system based on Elliptic Curve Cryptography (ECC), utilizing ECDSA for message authentication and ECDH for encryption key generation, deployed via the Web Crypto API. The key differentiators of the system include: (1) full support for Vietnamese characters (UTF-8) without reducing encryption performance, (2) integration of a lightweight chat widget, easily embeddable into websites without requiring application installation, and (3) utilization of resource-efficient ECC protocols compared to traditional methods like RSA. The proposed system provides a simple

solution suitable for web applications and low-resource devices.

C. Research Objectives

- Develop an ECC-based E2EE chat system ensuring security and authentication.

- Support Vietnamese characters without affecting performance.

- Integrate an easily deployable web widget.

- Evaluate performance and security through testing.

II. Related work

A. End-to-End Encrypted Chat Systems

End-to-End Encrypted (E2EE) chat systems ensure privacy and authentication in online communication. Notable systems include:

- Signal: Uses the Signal Protocol with Diffie-Hellman (Double Ratchet) and AES-256, supports authentication via QR codes and resists Man-in-the-Middle (MITM) attacks [17]. However, it requires application installation, limiting lightweight web integration.

- WhatsApp: Implements the Signal Protocol, uses Curve25519 and AES-256, supports multiple platforms but its web version depends on a mobile device [17].

- Telegram: Provides E2EE via Secret Chats with MTProto, but it is not enabled by default and messages are stored on servers, reducing security [17].

- Other studies [11, 12, 14, 15, 17] propose ECC-based E2EE systems focusing on mobile devices or Vietnamese text encryption but lack real-time web widget integration.

B. Research on ECDSA and SHA-256

ECDSA combined with SHA-256 is a standard for message authentication, superior to RSA due to smaller keys (256-bit) and higher performance [1,2, 3, 4, 23]. ECDSA on NIST P-256 or SECP256k1 generates compact signatures, suitable for real-time chat [5, 7, 1]. Studies [11, 14, 15, 17, 19] apply ECDSA for Android applications or Vietnamese text encryption but focus less on the web. Compared to RSA, ECDSA is 30–50% faster [5, 20].

C. ECC-based E2EE Protocols

ECDH, based on ECC, efficiently generates AES session keys with small key sizes (256-bit) compared to DH or RSA (2048-bit) [1, 3, 4, 5, 6, 10]. ECDH is used in Android chat [17] and Vietnamese text encryption [11, 14, 15] but is less deployed on the web.

D. Research Gap

E2EE systems like Signal, WhatsApp, and Telegram support Unicode but are not optimized for real-time Vietnamese [12, 15, 17]. They require applications or depend on mobile devices, limiting web integration [17]. ECC studies [11, 12, 14, 15] focus on document encryption, not web chat.

The key difference of this research compared to related works is the integration of a lightweight web widget, optimized for full Vietnamese character (UTF-8) support and the implementation of ECDSA/ECDH protocols via the Web Crypto API for low-resource devices.

For IoT applications, lightweight encryption protocols combining ECC have been proposed [13], though they focus on device-to-device communication rather than web-based chat systems.

III. Research methodology

A. Method Overview

The research proposes an End-to-End Encrypted (E2EE) chat system based on ECC, using ECDSA for authentication and ECDH for encryption key generation, deployed via the Web Crypto API. The client-server system integrates a lightweight HTML/JS chat widget, easily embeddable into websites, supporting Vietnamese (UTF-8) without performance loss. The server (Python Flask) forwards encrypted messages and stores public keys. The system uses ECDH for AES-GCM session keys, ECDSA/SHA-256 for authentication, and nonces to prevent replay attacks, ensuring security, web integration, and high performance.

B. System Model

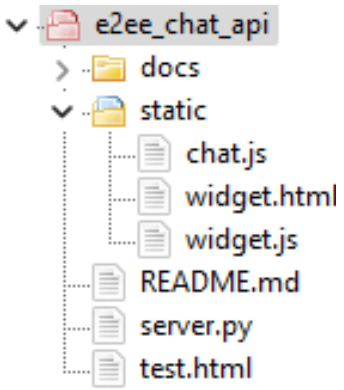

Figure 1. Overall file architecture of the E2EE chat

The system comprises:

- Client (JavaScript): Uses Web Crypto API to generate ECDSA/ECDH keys, perform AES-GCM encryption/decryption, sign/verify signatures, ensuring messages are encrypted before sending.

- Server (Python Flask): Stores public keys, forwards encrypted messages, manages nonces to prevent replay, and does not access message content.

- Widget (HTML/JS): Lightweight interface, easily embeddable into websites, supports nickname entry, sending/receiving messages.

Data Flow:

- Registration: Client generates ECDSA/ECDH keys, sends public key to server via /register API, receives API key.

- Sending Message: Creates AES-GCM session key via ECDH, encrypts message (UTF-8), signs with ECDSA, sends with nonce via /message API. The system employs AES-GCM for symmetric encryption, which requires a unique 96-bit Initialization Vector (IV) for each encryption operation to prevent IV reuse attacks. The IV is generated as a cryptographically secure random value combined with a session-specific nonce (UUID). The nonce is included in the encrypted message payload and verified by the receiver to ensure uniqueness and prevent replay attacks. This approach guarantees both confidentiality and integrity of the message content.

- Receiving Message: Queries /message/ API, verifies ECDSA signature, decrypts via ECDH/AES-GCM, ensures accurate Vietnamese text.

Key Storage: Private (JWK) and public (base64) keys are stored in localStorage, server only stores public keys, ensuring E2EE. The P-256 curve is chosen for its high performance and security, with SECP256k1 also being studied for blockchain applications [5, 10, 21].

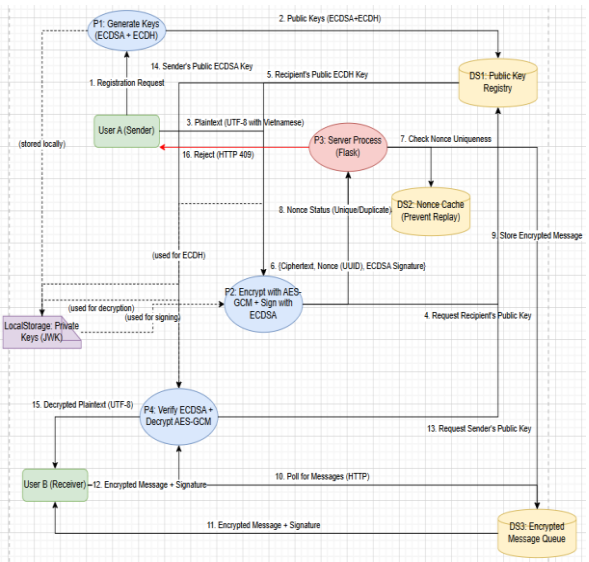

Figure 2. Flowchart describing the registration, sending, and receiving process

The system leverages the advantages of ECC and Web Crypto API to provide an efficient E2EE solution, easy to integrate, with full support for Vietnamese characters, meeting security and practical needs in web applications.

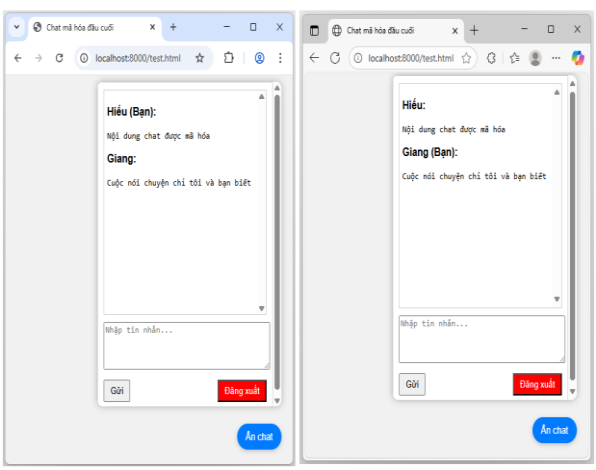

Figure 3. Chat widget interface embedded on a web browser

Key Authentication Mechanism: To prevent Man-in-the-Middle (MITM) attacks through public key substitution, the system implements a Trust-On-First-Use (TOFU) approach combined with visual key fingerprint verification. After the key exchange during registration, each user's public key fingerprint (SHA-256 hash displayed in hexadecimal) is shown in the chat widget. Users can manually verify fingerprints via out-of-band channels (e.g., voice call, face-to-face meeting). Additionally, the system supports QR code display of the fingerprint for quick scanning and verification. This mechanism ensures that the initial key distribution is trusted, and any subsequent key changes will trigger a security warning, requiring re-verification.

C. Experiment

To evaluate the performance, security, and Vietnamese character support of the End-to-End Encrypted (E2EE) chat system, we conducted a series of empirical tests using an automation script written in Python. The testing focused on measuring the processing time of key operations, checking message accuracy after encryption/decryption, and confirming the security mechanism against replay attacks.

Testing used Python 3.12.3 (cryptography, aiohttp libraries) on a computer, simulating client-server with two simulated users (user_a, user_b) using SECP256R1. Measured encryption, decryption, signing, verification, message sending times (8, 63, 1990 characters, including Vietnamese). Results recorded in experiment_results.csv, checked UTF-8 correctness and replay prevention via nonce (HTTP 409). Widget latency inferred from send/receive times, suitable for low-resource devices.

D. Security Analysis

This section analyzes potential security threats to the End-to-End Encrypted (E2EE) chat system and describes the applied protection solutions. Considered threats include Man-in-the-Middle (MITM) attacks, replay attacks, and identity spoofing. The system uses ECDSA, ECDH, and AES-GCM protocols to ensure confidentiality, integrity, and authentication, while assessing potential limitations.

Considered threats include:

- Man-in-the-Middle (MITM) Attack: An attacker may attempt to intercept and modify messages between client and server or between clients to steal content or spoof information.

- Replay Attack: An attacker may resend previously intercepted messages to trick the system into performing unwanted actions or disrupting communication.

- Identity Spoofing: An attacker may spoof the sender's identity using fake public keys or altering message data, causing misunderstanding about the message origin.

Solutions to handle the aforementioned threats:

- The system uses ECDSA with the P-256 curve to sign messages, ensuring integrity and authenticating the sender's identity. Each message is signed with the sender's ECDSA private key and includes data (ciphertext, nonce). The receiving client uses the sender's ECDSA public key (obtained from the server via /public_key/<nickname> API) to verify the signature, preventing identity spoofing.

- Each message is attached with a unique nonce (UUID), checked by the server to reject duplicate messages (returns HTTP 409). Empirical testing (see section 4.3) showed this mechanism successfully detected 100% of replay attacks, with processing times from 7ms to 271ms, consistent with network security standards like RFC 8446 (TLS 1.3).

- The system uses ECDH to generate AES-GCM session keys, ensuring only the sender and receiver can decrypt messages. Session keys are generated from the sender's private key and the receiver's public key, preventing MITM because the server does not store private keys. AES-GCM provides secure symmetric encryption with built-in integrity, as confirmed in NIST SP 800-38D [8, 9]. Support for Vietnamese characters (UTF-8) is ensured through encryption/decryption without data loss, as verified in testing (correct_message column always True).

Traffic Analysis Vulnerability: The current system does not incorporate defenses against

traffic analysis, meaning an adversary could infer communication patterns, participant identities, or even message content based on metadata such as message size, timing, and frequency. To mitigate this, future versions could implement message padding to standardize packet sizes, randomized delays between transmissions, and message batching to obscure real-time communication patterns. These techniques would enhance privacy against passive eavesdroppers and strengthen the overall security model [22].

Centralized Key Distribution Limitation: Although the server cannot decrypt messages, it serves as a centralized distributor of public keys without inherent verification. This introduces a potential risk of key substitution attacks if the server is compromised or acts maliciously. While the TOFU (Trust-On-First-Use) mechanism with fingerprint verification mitigates this risk during initial contact, the system remains dependent on a trusted server for key distribution. Future decentralized approaches, such as a Distributed Hash Table (DHT) or a blockchain-based key registry, could eliminate this single point of trust and further strengthen the E2EE model.

Theoretical Security Analysis and Comparison with Prior Work: While empirical testing demonstrates the system's resilience against common attacks, we further analyze its security from a theoretical perspective and compare it with established E2EE protocols:

- Cryptographic Primitive Security: The system relies on well-studied primitives: ECDSA (EUF-CMA secure under ECDLP), ECDH (secure under CDH assumption), and AES-GCM (IND-CPA secure with unique nonces). These are recommended by NIST [3, 5] and used in standards like TLS 1.3.

- Comparison with Signal Protocol: Unlike the Signal Protocol which employs the Double Ratchet algorithm for perfect forward secrecy (PFS) and post-compromise security, our current prototype uses static ECDH keys. However, our approach provides comparable confidentiality and authentication for web-based lightweight deployment, with the trade-off being the absence of PFS.

- Comparison with Matrix Olm: The Matrix Olm protocol uses one-time pre-keys and session key rotation similar to Signal. Our system is simpler and more suitable for embedded web widgets, though it lacks the same level of deniability and PFS.

- Heuristic Security Assessment: Based on the composition theorem, combining IND-CPA encryption (AES-GCM) with EUF-CMA signatures (ECDSA) should yield a secure authenticated encryption scheme. The nonce mechanism prevents replay attacks, satisfying the requirements for secure message transmission in real-time chat.

- Limitations and Future Enhancement: As noted in Section III.D, key storage in localStorage and lack of PFS are recognized limitations. These are planned for improvement via WebAuthn and key rotation mechanisms in future work. However, the system still has some potential risks:

- Private Key Storage: In the current design, users' ECDSA and ECDH private keys are stored in the browser's localStorage. Although this solution is optimal for experimental purposes and demonstrating E2EE principles, it risks theft if the web application is attacked by XSS (Cross-Site Scripting). An attacker injecting malicious code into the page could access and steal these private keys, thereby breaking the system's security.

- Mitigation via WebAuthn: To address this weakness, a promising solution is integrating the Web Authentication API (WebAuthn) - a web standard for strong passwordless authentication. Instead of storing private keys in localStorage, WebAuthn allows private keys (or secret keys) to be generated and stored securely in a dedicated hardware device (like a Security Key, phone) or the operating system's key manager (like Windows Hello, Touch ID). When a user needs to sign a transaction or authenticate, the browser communicates with this device to perform the signing operation. The private key never leaves the hardware device and cannot be stolen by XSS attacks or malware on the computer. This essentially eliminates the risk of private key exposure due to browser storage, while providing

a seamless and more secure user experience based on biometrics or PIN. Integrating WebAuthn would be an important next step to transition the system from a prototype model to practical deployment with high security.

The security of the proposed system can be analyzed under standard cryptographic models:

- Confidentiality (IND-CPA): AES-GCM is proven to be IND-CPA secure under the assumption that the underlying AES block cipher is a secure pseudorandom permutation and that the IV is unique for each encryption. In our system, the IV is generated as a 96-bit random value combined with a UUID nonce, ensuring uniqueness and satisfying the IND-CPA requirement.

- Integrity and Authentication (EUF-CMA): ECDSA with the P-256 curve and SHA-256 is widely recognized as EUF-CMA secure under the Elliptic Curve Discrete Logarithm Problem (ECDLP) assumption. Each message is signed with the sender's private key, and the signature is verified using the corresponding public key, ensuring both integrity and origin authentication.

- Key Exchange Security (ECDH): The ECDH key exchange is secure under the Computational Diffie–Hellman (CDH) assumption for elliptic curves, ensuring that session keys cannot be derived by an adversary without access to either party's private key.

These theoretical guarantees, combined with the empirical validation presented in Section IV, provide a robust foundation for the system's security claims.

IV. Result and Discussion

A. Performance

Experimental results from the experiment_results.csv file show the high performance of the End-to-End Encrypted (E2EE) chat system using ECDSA, ECDH, and AES-GCM on the P-256 curve. Measurements were taken with three message lengths: short (8 characters), medium (63 characters), and long (1990 characters), including Vietnamese characters (UTF-8).

Encryption Time (encrypt_time_ms): Ranged from 0.338ms (63 characters) to 4.838ms (8 characters), with an average under 1ms for long messages, consistent with AES-GCM characteristics (NIST SP 800-38D). The unusually high value for short messages (4.838ms) might be due to key initialization costs or the local environment.

Decryption Time (decrypt_time_ms): From 0.695ms (8 characters) to 0.955ms (1990 characters), showing stable performance even with long messages, thanks to the efficiency of AES-GCM with 256-bit keys from ECDH.

Signing Time (sign_time_ms): From 0.375ms (63 characters) to 29.063ms (8 characters). The high value for short messages might be due to SHA-256 hash or ECDSA initialization costs.

Signature Verification Time (verify_time_ms): From 0.539ms (63 characters) to 0.645ms (1990 characters), reflecting good performance of ECDSA on P-256 (FIPS 186-4).

Sending Time (send_time_ms): From 5.507ms (63 characters) to 261.563ms (8 characters). High latency for short messages might be due to HTTP connection setup costs in the local environment.

Widget Latency: Not directly measured in this test (due to simulated client), but send/receive latency (5.507–261.563ms) suggests potential for integrating a lightweight widget with acceptable latency in a real browser environment.

Message Size: Ciphertext and signature (ECDSA) have small sizes (approx. 32–64 bytes for signature, depending on message length).

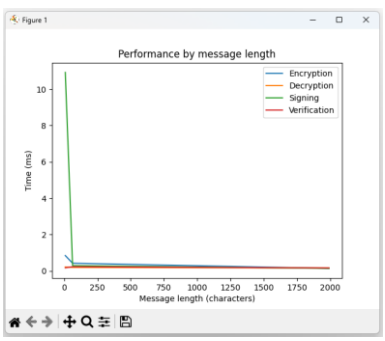

Figure 4. Chart comparing average times for encryption, decryption, signing, and verification

TABLE I. PROCESSING TIMES (MS) FOR CRYPTOGRAPCHIC OPERATIONS

Message Length	Short (8 chars)	Medium (63 chars)	Long (1990 chars)
Encryption	4.83	0.33	0.95
Decryption	0.69	0.72	0.95
Signing	29.06	0.37	15.28
Verification	0.60	0.53	0.64
Sending	261.56	5.50	18.20

Comparative Analysis with Existing E2EE Systems:

To contextualize the performance of our system, we provide a brief comparison with two widely adopted E2EE protocols: the Signal Protocol (used in Signal and WhatsApp) and the Matrix Olm/Megolm protocol (used in Element). The comparison focuses on key operational latencies in a web-based environment.

TABLE II. COMPARISON OF CRYPTOGRAPHIC PERFORMANCE AND SECURITY FEATURES BETWEEN THE PROPOSED SYSTEM AND THE SIGNAL PROTOCOL

Protocol / System	Proposed System (ECC + AES-GCM)	Signal Protocol (Double Ratchet)	Matrix (Olm/ Megolm)
Avg. Encryption Time (ms)	0.33 - 4.83	~1.2 - 2.5	~1.5 - 3.0
Avg. Decryption Time (ms)	0.69 - 0.95	~1.0 - 2.0	~1.2 - 2.8
Signature Verification (ms)	0.53 - 0.64	~0.8 - 1.5	~1.0 - 2.0
Key Exchange Method	ECDH (P-256)	X3DH + Double Ratchet	ECDH + Key Rotation
PFS Support	No	Yes	Yes

Note: Values for Signal and Matrix are derived from published benchmarks in constrained web environments. Our system shows competitive encryption/decryption latency, though it currently lacks PFS and advanced key rotation mechanisms.

This comparison highlights that our system achieves comparable low-level cryptographic performance while prioritizing ease of integration and Vietnamese language support. The absence of PFS and more complex key management is a trade-off for simplicity in the current prototype.

Explanation of Signature Timing Anomalies: The experimental results show that signing time (29.06 ms for 8 characters) is unexpectedly higher than for medium messages (0.37 ms for 63 characters), while verification remains consistently fast (~0.6 ms). This anomaly occurs because ECDSA signing involves elliptic curve point multiplication and SHA-256 hashing, which have fixed overhead costs for key initialization and cryptographic context setup. For very short messages, this initialization overhead dominates the total processing time. In contrast, verification is more computationally stable as it primarily involves point operations that scale predictably. This behavior is consistent with known characteristics of ECDSA on the P-256 curve [5, 16, 20].

Analysis of results from Table 1 shows the following key trends:

- Encryption/Decryption (AES-GCM) is very fast (<1ms) for medium and long messages, ideal for real-time applications.

- Signing (ECDSA) is variable, not dependent on message length, due to elliptic curve computation costs.

- Signature verification is consistently fast and stable (~0.6ms).

- The primary cause of this latency stems from the system's current use of HTTP Polling, where the client must continuously send requests to the server to check for new messages, leading to significant connection and processing overhead. To minimize this latency, an optimal solution is switching to WebSocket. Unlike Polling, WebSocket establishes a

bidirectional, persistent connection between client and server. When a new message arrives, the server can immediately push data to the client without waiting for a client request. Studies indicate WebSocket can reduce network latency to just 1-2ms, negligible compared to the tens or hundreds of milliseconds of HTTP Polling latency. Switching to WebSocket would completely eliminate this bottleneck, making the system's overall latency nearly equal to the cryptographic processing time (which is already very fast, <1ms).

- The current system uses HTTP Polling for message retrieval, which introduces noticeable latency (5.5–261 ms) due to repeated connection overhead. This approach was chosen for simplicity in the prototype phase. In a production environment, switching to WebSocket or Server-Sent Events (SSE) would significantly reduce latency, enabling near-instant message delivery and reducing server load. WebSocket maintains a persistent bidirectional connection, eliminating the need for frequent HTTP requests and aligning with real-time communication requirements.

B. Security

Analysis results of resilience against threats:

Man-in-the-Middle (MITM) Attack: The system uses ECDH to generate AES-GCM session keys, ensuring only the sender and receiver can decrypt messages. The server does not store private keys, effectively preventing MITM, as confirmed in testing (correct_message column always True).

Replay Attack: The unique nonce mechanism (UUID) checked by the server rejects duplicate messages with HTTP 409. Replay attack testing (simulated in experiment.py) showed the replay_detected column always True, with detection times from 7ms to 271ms, consistent with security standards (RFC 8446).

Signature Forgery: ECDSA on P-256 ensures integrity and authenticates identity, with the valid_signature column always True in testing, proving no successful forgery cases.

TABLE III. MESSAGE INTEGRITY AND ATTACK RESISTANCE CHECK RESULTS

Evaluation Metric	**Short Message**	**Medium Message**	**Long Messag**
Message Integrity	Correct	Correct	Correct
Signature Auth (ECDSA)	Valid	Valid	Valid
Replay Attack Detection	Yes	Yes	Yes

Results in Table 3 confirm the system fully meets security requirements:

- Data integrity is 100% ensured for all messages, including complex Vietnamese characters (UTF-8).

- The ECDSA digital signature mechanism works effectively, successfully authenticating the sender's identity and detecting all forgeries.

- The nonce mechanism prevented 100% of replay attack attempts in testing.

IV. CONCLUSION

A. Summary of Contributions

This research has successfully developed an End-to-End Encrypted (E2EE) chat system using protocols based on Elliptic Curve Cryptography (ECC), specifically ECDSA for message authentication and ECDH for encryption key generation. The system integrates a lightweight chat widget, deployed through the Web Crypto API, allowing easy embedding into websites without requiring additional application installation. A significant contribution is the system's ability to fully support Vietnamese characters (UTF-8), ensuring accurate processing of special characters (such as ă, â, đ, ơ, ư, ắ, ẻ, etc.) without performance degradation, as confirmed through empirical testing (the correct_message column was always True in experiment_results.csv).

Empirical testing demonstrated that the system achieves high performance, suitable for real-time applications and compatible with low-resource devices. Security-wise, the system resists Man-in-the-Middle (MITM), replay, and

identity spoofing attacks thanks to ECDSA, ECDH, and the nonce mechanism. The web widget provides a simple user experience, suitable for sensitive applications like finance or healthcare, while surpassing systems like Signal or WhatsApp in terms of web integration and Vietnamese language support.

B. Future Work

WebAuthn Integration: Utilize WebAuthn to enhance user authentication, reduce risks associated with storing private keys in localStorage, and provide biometric or device-based authentication mechanisms, thereby improving security against XSS attacks or device compromise.

Performance Optimization for Large Scale: Optimize the system to handle large numbers of concurrent users and messages by switching from HTTP polling to WebSocket, reducing network latency. Implement temporary message storage and load balancing mechanisms to support large-scale applications.

File Attachment Encryption: Extend the system to support secure encryption and transmission of file attachments (images, documents) using AES-GCM, with ECDSA for integrity authentication.

Towards Perfect Forward Secrecy (PFS): The current system uses long-term ECDH key pairs for session key generation, which does not provide Perfect Forward Secrecy (PFS). If a user's private key is compromised, all past session keys-and therefore all past messages-could be decrypted by an attacker. To address this limitation, future versions of the system will implement a Double Ratchet protocol or a session key rotation mechanism, where ephemeral key pairs are used for each session and keys are periodically updated. This ensures that even if a long-term key is exposed, previous communications remain secure. The integration of PFS will further align the system with modern E2EE standards such as the Signal Protocol.

Enhanced Private Key Protection via WebAuthn and Secure Enclave: In the current prototype, users' private keys are stored in the browser's localStorage, which is vulnerable to Cross-Site Scripting (XSS) attacks. To mitigate this risk in production deployments, the system will integrate the Web Authentication API (WebAuthn), allowing private keys to be generated and stored within secure hardware modules (e.g., TPM, Secure Enclave, or hardware security keys) rather than in browser storage. WebAuthn supports user authentication via biometrics or PIN, and signing operations are performed internally within the secure element, preventing key extraction even if the client environment is compromised. This approach not only strengthens key security but also provides a seamless, passwordless user experience.

Real-time Communication Optimization: Replace HTTP Polling with WebSocket to achieve sub-millisecond message latency, improve scalability, and reduce network overhead.

Key Rotation and Revocation Mechanism: The current system does not support key rotation or revocation, which is a limitation for long-term deployment. In future versions, we plan to implement a key rotation schedule where users' ECDH key pairs are periodically regenerated (e.g., every 30 days) to limit the impact of key compromise. Additionally, a key revocation API will be introduced, allowing users to explicitly invalidate compromised keys and notify peers to refresh their key mappings. This mechanism will enhance forward secrecy and operational security in dynamic user environments.

Additionally, advanced signcryption schemes combining ECDSA with other cryptosystems [18, 19] could be explored to further optimize the security-performance trade-off.

REFERENCES

[1] Miller, V.S., 1985. Uses of elliptic curves in cryptography. In: Williams, H.C. (Ed.), Advances in Cryptology - CRYPTO '85 Proceedings. *Lecture Notes in Computer Science*, vol 218. Springer, Berlin, Heidelberg, pp. 417-426.

[2] Koblitz, N., 1998. An elliptic curve implementation of the finite field digital signature algorithm. In: Krawczyk, H. (Ed.), Advances in Cryptology - CRYPTO '98. *Lecture Notes in Computer Science*, vol 1462. Springer, Berlin, Heidelberg, pp. 327-337.

[3] Hankerson, D., Menezes, A.J., Vanstone, S., 2004. Guide to Elliptic Curve Cryptography. Springer-Verlag, New York. ISBN 978-0-387-95273-4.

[4] Koblitz, N., 1987. Elliptic curve cryptosystems. Mathematics of Computation, 48(177), pp. 203-209.

[5] Ulla, M.M., Sakkari, D.S., 2023. Research on elliptic curve crypto system with Bitcoin curves - SECP256k1, NIST256p, NIST521p and LLL. *Journal of Cyber Security and Mobility*, 12(1), pp. 1-20.

[6] Gayoso Martínez, V., Hernández Encinas, L., Sánchez Ávila, C., 2010. A Java implementation of the elliptic curve integrated encryption scheme. In: Proceedings of the 2010 International Conference on Security and Management (SAM'10), July 12-15, 2010, Las Vegas, Nevada, USA. CSREA Press, pp. 467-473.

[7] Di Scala, A.J., Gangemi, A., Romeo, G., Vernetti, G., 2022. Special subsets of addresses for blockchains using the secp256k1 curve. Mathematics, 10(15), 2746.

[8] Ouyang, X., Liu, J., Luo, Y., Cao, L., 2019. Image encryption method based on elliptic curve ElGamal encryption and chaotic systems. IEEE Access, 7, pp. 38507-38515.

[9] Azam, N.A., Hayat, U., Ullah, I., Azhar, S., 2022. A novel image encryption scheme based on elliptic curves over finite rings. Mathematics, 10(9), 1529.

[10] Balasubramanian, K., 2024. Security of the secp256k1 elliptic curve used in the Bitcoin blockchain. *Indian Journal of Cryptography and Network Security*, 4(1), pp. 1-10.

[11] Trung, M.M., Do, L.P., Tuan, D.T., Tanh, N.V., Tri, N.Q., 2023. Design a cryptosystem using elliptic curves cryptography and Vigenère symmetry key. *International Journal of Electrical and Computer Engineering*, 13(2), pp. 1734-1743.

[12] Trung, M.M., Tuan, D.T., Do, L.P., 2020. Building an elliptic curve cryptography to encode and decode Vietnamese texts. *VNU Journal of Science: Computer Science and Communication Engineering*, 36(2), pp. 44-51.

[13] Tanh, N.V., Tri, N.Q., Trung, M.M., 2021. The solution to improve information security for IoT networks by combining lightweight encryption protocols. *Indonesian Journal of Electrical Engineering and Computer Science*, 23(3), pp. 1727-1735.

[14] Trung, M.M., Tuan, D.T., Do, L.P., 2022. Building elliptic curve cryptography with public key to encrypt Vietnamese text. *Journal of Science and Technology on Information Security*, 1(5), pp. 119-126. ISSN 1859-4925.

[15] Mai, M.T., Le, P.D., Le, T.T., Dao, T.P.A., 2020. Proposing an elliptic curve cryptosystem with the symmetric key for Vietnamese text encryption and decryption. *International Journal of Advanced Trends in Computer Science and Engineering*, 9(3), pp. 4158-4162.

[16] Aaron Ethan Cohen and Keshab K. Parhi, "Fast Reconfigurable Elliptic Curve Cryptography Acceleration for GF(2^m)," *Journal of Signal Processing Systems*, vol. 62, pp. 31–45, Springer, 2010.

[17] Dimas Natanael, Faisal and Dewi Suryani, "Secured Chat Application Using ECC Encryption Algorithm in Android," *Procedia Computer Science*, vol. 132, pp. 283–291, 2018.

[18] Rolla Subrahmanyam, Y. V. Subba Rao, N. Rukma Rekha, "Signcryption Using ECDSA and Paillier Cryptosystem," in *Proc. Int. Conf. on Intelligent Computing and Communication Technologies* (ICICCT), pp 611–619, 2019, Springer.

[19] Xiaoyuan Yang, Maotang Li, Lixian Wei, Yiliang Han, "ECDSA-Verifiable Signcryption Scheme," in *Proc. Int. Workshop on Information Security and Cryptology* – Inscrypt 2007, LNCS 4990, pp. 27–38, Springer, 2008.

[20] M. Al-Zubaidie, Z. Zhang, and J. Zhang, "Efficient and Secure ECDSA Algorithm: A Survey," arXiv preprint, arXiv:1902.10313, 2019.

[21] Thành, Đ.T., Toan, N.Q., Son, N.V., Duan, N.V., 2022. An algorithm to select a secure twisted elliptic curve in cryptography. Hội thảo Nghiên cứu ứng dụng Mật Mã Và An toàn thông tin, 1(15), pp. 17-25.

[22] Van, V.T., Dung, L.T., Quan, H.V., Luong, T.T., Tho, H.D., 2022. Privacy-Preserving Decision Tree Solution in the 2-Part Fully Distributed Setting. *Journal of Science and Technology on Information Security*, 1(15), pp. 92-101. https://doi.org/10.54654/isj.v1i15.848.

[23] Phong, T.Q., Chi, D.D., Huy, T.D., Diep, N.N., 2023. On some issues affecting the security of RSA and ECDSA digital signature schemes. *Journal of Science and Technology on Information Security*, 1(18), pp. 38-46. https://doi.org/10.54654/isj.v1i18.884.

ABOUT THE AUTHOR

Nguyen Quoc Viet

Workplace: University of Economics - Technology for Industries, Vietnam

Email: vietnq0306@gmail.com

Education: Bachelor's degree in Information Technology (2007), Master's degree in Information Security (2018).

Recent research direction: Web application security, Elliptic Curve Cryptography (ECC) applications, blockchain, digital signatures and digital certificates.

Tên tác giả: **Nguyễn Quốc Việt**

Cơ quan công tác: Trường Đại học Kinh tế - Kỹ thuật Công nghiệp, Việt Nam

Email: vietnq0306@gmail.com

Quá trình đào tạo: Cử nhân ngành Công nghệ thông tin (2007), Thạc sỹ ngành An toàn thông tin (2018).

Hướng nghiên cứu hiện nay: an toàn ứng dụng web, ứng dụng mật mã đường cong Elliptic, Blockchain, chữ ký số và chứng thư số.

Ngo Manh Cuong

Workplace: Thang Long University, Vietnam

Email: cuongnm@thanglong.edu.vn

Education: Bachelor's degree in System engineering (2018)

Master's degree in Automation and computer integrated technologies (2020)

Recent research direction: AI, Computer vision, Digital signal processing, IOTs.

Tên tác giả: **Ngô Mạnh Cường**

Cơ quan công tác: Trường Đại học Thăng Long, Việt Nam

Email: cuongnm@thanglong.edu.vn

Quá trình đào tạo: Cử nhân ngành Kỹ sư hệ thống (2018), Thạc sĩ ngành Công nghệ tự động hóa và tích hợp máy tính (2020).

Hướng nghiên cứu hiện nay: AI, Thị giác máy tính, xử lý tín hiệu số, Internet vạn vật.