Evaluation of Encryption Practices in Social Media and Messaging Platforms: A Comprehensive Analysis and Ranking

Vijay Bhuse
Grand Valley State University, Allendale, MI USA
Joon Son[*]
California State University San Bernadino, San Bernardino, CA USA
Venkata Varikoti
Grand Valley State University, Allendale, MI USA

ABSTRACT
This study evaluates encryption practices across major social media and messaging platforms, including Facebook, Instagram, WhatsApp, Twitter, LinkedIn, Slack, Discord, and Snapchat. We analyse the implementation of encryption protocols such as TLS, Signal, QUIC, and WebRTC to assess the security measures employed by these platforms. Our research methodology involves a comprehensive review of each platform's encryption technologies, known vulnerabilities, and proactive security measures. We introduce a novel ranking system that scores platforms based on encryption strength, security features, and identified vulnerabilities. Key findings reveal that WhatsApp, utilizing the Signal protocol, demonstrates the strongest security posture, while platforms like Facebook and Instagram face more significant security challenges despite employing advanced protocols. This comparative analysis provides valuable insights into the relative security strengths of popular social media platforms, highlighting the ongoing need for robust encryption practices in digital communications. Our research contributes to the field of cybersecurity by offering a systematic evaluation of encryption implementations across widely used social platforms, informing both users and developers about the current state of data protection in social media.

Keywords: social media platforms; encryption protocols; encryption practices; data protection; comprehensive analysis; ranking.
JEL: M15, O30

*corresponding author: json@csusb.edu

1. Introduction

In the digital age, social media has become a pivotal platform for personal branding and marketing, significantly amplifying the presence of influencers in the global market. The rise of social media influencers has been solidifying the presence of social media platforms in our daily lives. For instance, a detailed study of the 2013 German federal election highlighted the differences in topic prioritization between politicians and their constituents on platforms such as Facebook and Twitter, showcasing the distinct roles social media plays in political discourse (Stier et al., 2020). These advancements have led to increased scrutiny of how data is stored and utilized by these platforms.

Social media corporations are under heightened pressure from privacy advocates to adopt stronger data security measures. Initially, the trade-off between rapid growth and the stringent implementation of security protocols posed a significant challenge (Borisov et al., 2004). Early social media ventures often favoured expansion and market penetration over stringent operational and security protocols. Yet, as the industry matured, a shift towards prioritizing the security of their platforms and standardizing operations became evident. Although this realignment allowed for a temporary advantage in the market, it proved unsustainable in the long run.

In response, a paradigm shift towards a 'Secure by Design' philosophy has emerged, where security measures are integrated at the earliest stages of product development. This approach markedly diminishes the likelihood of vulnerabilities, leading to products that are 'Secure by Default.' Such products boast advanced security features, including multi-factor authentication (MFA), intrusion detection systems, and stringent access controls for sensitive data, often without imposing additional costs on the consumer. These features play a crucial role in ensuring the integrity and confidentiality of customer data, whether at rest, in transit, or in use (CISA, n.d.).

The evolution of social media applications has also embraced end-to-end encryption, a critical element for secure communication between devices. This encryption strategy relies extensively on advanced cryptographic methods. Companies such as Meta, Microsoft, Snapchat, Discord, and Twitter (now rebranded as X) have led the charge in this domain, leveraging their technical expertise to enhance data security.

2. Related Work

The rapid advancement of digital communications has underscored the need for robust security measures, particularly encryption protocols that safeguard user data across various social media platforms. This section provides a critical review of existing literature on encryption protocols employed by major social media platforms, underscoring the gaps that this study aims to fill and highlighting the unique contributions of our research.

End-to-end encryption (E2EE) has become a pivotal element in securing user communications on social media platforms. In their comprehensive study, researchers rigorously analysed the robust security features of the Signal protocol, widely adopted by platforms such as WhatsApp for its well-recognized robustness (Albrecht et al., 2016; Cohn-Gordon et al., 2016). These protocols ensure that only communicating users can read the messages, barring even the service providers from accessing the data.
However, while these studies provide a strong foundation for understanding individual encryption protocols, they often lack a comprehensive analysis across multiple platforms. The adoption of the QUIC protocol by platforms like Facebook and Snapchat has been

detailed by Iyengar and Thomson, highlighting its efficacy in reducing latency and improving connection resilience (J. Iyengar & Thomson, 2021).

Further, Kumar and Rai discuss the security issues associated with social networking sites, providing foundational insights into the security challenges these platforms face (Kumar et al., 2013). Kumari and Singh extend this discussion by critically analyzing privacy and security on social media, particularly focusing on how encryption techniques are implemented to safeguard user interactions (Kumari & Singh, 2015).

Bhuse (2023) provides a detailed review of end-to-end encryption for social media, discussing various cryptographic primitives used in social media apps like WhatsApp, Twitter, Facebook, Snapchat, and Instagram. This paper highlights the importance of robust encryption protocols like the Signal Protocol, which uses AES-256, HMAC-SHA256, and Curve25519 as its cryptographic primitives. Bhuse's work underscores the ongoing need for stronger encryption practices across social media platforms.

Existing literature predominantly focuses on individual platforms or specific protocols without offering a holistic view of how different social media sites compare in terms of encryption security. This paper addresses this critical gap by not only detailing and comparing the encryption protocols used by major social media platforms but also by developing a novel ranking system based on the strength and comprehensiveness of these protocols. This ranking system is designed to provide clear insights into which platforms prioritize user security.

3. End-to-End Encryption Protocols

End-to-end encryption protocols are crucial for application security, as the functionalities and features of an application are often aligned with these protocols.

3.1 Transport Layer Security Protocol Version 1.2

TLS 1.2 is a widely used and secure protocol that provides improvements in security, reliability, and performance for web application communication. It encrypts data using a combination of symmetric and asymmetric cryptography. Notably, TLS 1.2 enhances security during the handshake process by replacing the MD5/SHA-1 combination with a single hash function and offers greater flexibility in selecting hash and signature algorithms. It also supports enhanced authentication, encryption, TLS extensions, and AES cipher suites.

The TLS handshake in TLS 1.2 involves a series of messages, including the 'Client Hello,' 'Server Hello,' and certificate-related messages, which establish a secure communication channel. The certificates are cryptographically verified. The 'Client Key Exchange' message contains the client's public parameters for the ECDHE algorithm, and the master secret key is generated using these parameters, random values, and a Pseudo-Random Function (PRF). The 'Change Cipher Spec' message indicates the transition to encrypted communication. The 'Finished' message is designed to protect against man-in-the-middle attacks. For data encryption, TLS 1.2 employs the 'MAC-then-Encrypt' technique.

Despite its general robustness, TLS 1.2 is not without vulnerabilities. Issues such as BEAST, CRIME, POODLE, Lucky Thirteen, and SWEET32 have been identified, prompting the development of countermeasures and updates to mitigate these risks. In light of these vulnerabilities, the industry has gradually moved towards adopting TLS 1.3, a version that offers enhanced security features. To bolster security in TLS 1.2 deployments, it is recommended to adopt secure configuration practices. These include disabling weak cipher suites, enabling forward secrecy to protect past communications against future key compromises, and maintaining stringent key management practices. Such measures are essential not only to enhance the security of TLS 1.2 implementations but also to keep pace with the evolving landscape of cybersecurity threats (Ćurguz, 2016).

3.2 Transport Layer Security Protocol Version 1.3

TLS 1.3, developed in 2017, represents the next-generation protocol, featuring various updates from the handshake process, which establishes the connection between host and client, to the encryption of each transferred packet.

Building upon its predecessor, TLS 1.2, TLS 1.3 employs a combination of asymmetric and symmetric cryptography for encryption. Its advancements are particularly evident in the handshake process, which now reduces round trips to facilitate quicker connection establishment. TLS 1.3 mandates the use of robust hash functions like SHA-256 and SHA-384, addressing vulnerabilities such as collision, preimage, and birthday attacks found in older hashes like MD5 and SHA-1. The protocol simplifies the list of supported ciphers, focusing on strong, authenticated encryption modes and exclusively accepts data encrypted by AEAD (Authenticated Encryption with Associated Data) cipher suites, ensuring both encryption and integrity. Supported ciphers include AES-GCM, AES-CCM, and CHACHA20-POLY1305, notably eliminating CBC mode ciphers and their associated vulnerabilities, such as the BEAST and Lucky Thirteen attacks (Bhuse, 2023; Hassani Karbasi & Shahpasand, 2021).

While TLS 1.3 is a marked improvement over TLS 1.2, offering enhanced security and efficiency, ongoing vigilance for vulnerabilities remains essential. To date, no widespread vulnerabilities akin to BEAST, CRIME, or POODLE from previous versions have been identified in TLS 1.3. However, features like 0-RTT carry potential risks, particularly vulnerability to replay attacks if not properly implemented or misused (F5 Networks, n.d.)

3.3 Signal Protocol

The Signal Protocol is distinguished by its advanced encryption architecture, integrating the Double Ratchet Algorithm, prekeys, and the Extended Triple Diffie–Hellman (X3DH) handshake to achieve secure communication. At the heart of the protocol, the Double Ratchet Algorithm plays a pivotal role in encrypting messages exchanged between two parties, utilizing a shared secret key. This encryption is fortified through the adoption of the Elliptic Curve Diffie-Hellman (ECDH) protocol, specifically employing the Curve25519 curve. The process is further enhanced by Key Derivation Function (KDF) chains, the Diffie-Hellman ratchet, and the symmetric-key ratchet, facilitating the periodic renewal of Diffie-Hellman public keys. This renewal process continuously updates the root, sending, and receiving chains, thereby maintaining a high level of security.

The establishment of a shared secret key through this method allows for the generation of session keys, which are then used to encrypt messages, secure voice and video calls, and verify the authenticity of messages. The Signal Protocol's use of top-tier encryption standards, including AES-256 for encryption and HMAC-SHA256 for message authentication, underscores its position as one of the leading encryption methods in securing private communications. Its commitment to providing exceptional security and privacy in messaging is supported by scholarly research, including works by Hassani Karbasi & Shahpasand (Hassani Karbasi & Shahpasand, 2021) and Bhuse (Bhuse, 2023).

3.4 QUIC Protocol

The QUIC protocol, a groundbreaking development by Google, is designed to significantly enhance the transport performance of HTTPS traffic, aiming to provide privacy assurances on par with Transport Layer Security (TLS). This innovative protocol incorporates elements from TCP, TLS, and HTTP/2, thereby streamlining the establishment of secure transport connections. It is distinguished by its unique handshake model, which facilitates two modes of operation: zero round-trip time (0-RTT) for instantaneous connections, allowing data to be sent before the connection is fully established, and one round-trip time (1-RTT) for conducting a more comprehensive handshake process, ensuring a secure connection from the outset.

The initiation of a connection under the QUIC protocol is marked by a Client Hello (CHLO) message, which includes a Connection Identifier (ConnID). This identifier, which is randomly generated, is crucial for both the client and the server to keep track of the ongoing connection. The ConnID plays a central role in the QUIC packet's public header, facilitating the seamless migration of connections across new IP addresses or ports, thus enhancing the protocol’s adaptability and resilience.

In instances where the server does not recognize a client's ConnID, it may respond by issuing a message that prompts the client to restart the connection process in a secure manner, rather than using a Reject (REJ) message with a Server Configuration Identifier (SCID). This approach reflects the continuous evolution and standardization of QUIC by the Internet Engineering Task Force (IETF), focusing on improving security and efficiency. The initial connection attempt includes mechanisms to prevent misuse and ensure the integrity of the connection, such as verifying the client's identity and establishing secure communication parameters.

Subsequent connections between the same client and server can leverage the 0-RTT feature, which enables faster setup by using cached information from previous connections, such as security settings and credentials. This mechanism significantly reduces latency, providing an immediate start to data transmission, which is especially beneficial for repeated connections to the same server (Sy et al., 2019).

Despite its numerous advantages, the QUIC protocol is not without potential security vulnerabilities. The 0-RTT feature, while efficient, introduces specific risks, including the possibility of replay attacks, which could potentially compromise the security of transmitted data. Furthermore, rigorous testing, including fuzzing tests that expose the protocol to a wide range of inputs, has revealed zero-day vulnerabilities in some implementations of QUIC, highlighting susceptibilities to Denial of Service (DoS) attacks. These tests are crucial for identifying and mitigating vulnerabilities, ensuring the protocol's robustness. Another area of concern involves the manipulation of data streams, which could allow attackers to disrupt the integrity of QUIC's data transmission, underscoring the importance of continuous security evaluations and updates to the protocol.

In conclusion, the QUIC protocol represents a significant advancement in secure and efficient web communication, promising enhanced performance for HTTPS traffic along with robust privacy protections. Its development reflects a commitment to evolving internet standards to meet the demands of modern web applications. However, like all technological innovations, it requires ongoing vigilance to address potential security vulnerabilities, ensuring that it remains a reliable and secure option for global internet communication.

3.5 WebRTC Protocol

Web Real-Time Communications (WebRTC) is an advanced open-source protocol that enables seamless real-time communication, including voice, text, and video streaming, directly between web browsers and devices without the need for intermediary plugins or applications. This protocol leverages a complex mechanism to establish connections between multiple devices, ensuring the integrity of these connections with the help of signaling servers, which play a crucial role in the initial stages of the WebRTC connection process.

The establishment of a WebRTC connection begins with an interaction involving signaling servers, through which peers exchange essential metadata. This exchange is facilitated by an intermediary server, setting the stage for direct communication between peers. The protocol adeptly navigates through various network barriers, such as Network Address Translators (NATs), by employing the Interactive Connectivity Establishment (ICE) framework. During this critical phase, peers—also known as candidates—engage in gathering potential network addresses and ports, a process that is fundamental to establishing a viable connection path. The candidates' ability to accumulate host addresses derives not only from the local system but also from responses facilitated by a Session Traversal Utilities for NAT (STUN) server. In scenarios where direct connectivity poses a challenge, the protocol may resort to utilizing a Traversal Using Relays around NAT (TURN) server. Through dedicated signaling channels, these candidates exchange gathered information, enabling a comprehensive mutual understanding of the available network paths. This meticulous process ensures the identification of the most optimal route for communication, which includes conducting tests on each address with specifically crafted packets to confirm the feasibility of a connection. Upon successful confirmation of a pair of candidates for connectivity, a direct peer-to-peer connection is established, utilizing the preferred addresses identified during this process. This establishment marks the beginning of a secure channel through which peers can transmit encrypted data, encompassing audio, video, or other forms of media, all of which are protected by Datagram Transport Layer Security (DTLS) (NTT Communication Project, n.d.).

Despite the robust design and capabilities of WebRTC, applications built on this protocol are not immune to security vulnerabilities. One notable risk is the susceptibility to cross-site scripting (XSS) attacks, where attackers can inject malicious JavaScript into the application, potentially compromising its integrity. This type of vulnerability poses a significant risk, especially in terms of exposing sensitive information such as IP addresses during the signaling process, thereby leading to potential privacy breaches. Furthermore, the file-sharing features inherent to WebRTC applications, if not meticulously verified for security before transmission, present an avenue that could be exploited to disseminate malware (Reiter & Marsalek, 2017). This aspect of WebRTC underscores the critical need for rigorous security measures and constant vigilance to safeguard against such vulnerabilities, ensuring the privacy and security of all parties involved in the communication process.

4. Brief Overview of Key Encryption Protocol Vulnerabilities

In this section, we offer a concise overview of significant vulnerabilities related to encryption protocols, including BEAST, CRIME, POODLE, Lucky Thirteen, SWEET32, 0-RTT, and XSS. This overview lays the groundwork for further discussions within this research paper. It is important to note that a detailed analysis of these vulnerabilities falls outside the scope of our study.

BEAST targets SSL 3.0/TLS 1.0, leading to the adoption of TLS 1.2 for its resistance to such attacks (Nidecki, 2020). CRIME exploits compression in SSL/TLS, which is mitigated by disabling compression. Lucky Thirteen is a theoretical attack on TLS's CBC mode, requiring careful TLS implementation and timing attack countermeasures (Holmes, n.d.). POODLE affects SSL 3.0 through padding oracle attacks, mitigated by disabling SSL 3.0 (Cobb, n.d.). SWEET32 targets 64-bit block ciphers, mitigated by using stronger ciphers and limiting key lifespan (Holmes, 2016). 0-RTT in TLS 1.3 introduces potential replay attacks, necessitating careful management. XSS impacts web application security, indirectly affecting encryption by compromising secure sessions.

These vulnerabilities highlight the ongoing battle between advancing cryptographic security and the evolving landscape of cyber threats. Each vulnerability has prompted specific mitigation strategies, often involving the upgrade to more secure protocols and the implementation of regular security updates. Understanding these vulnerabilities and their countermeasures is essential for maintaining the integrity and confidentiality of encrypted data. The subsequent diagram (Figure 1) illustrates the prevalence of these vulnerabilities as reflected by the number of reported CVEs, underscoring the importance of proactive security measures in the digital domain.

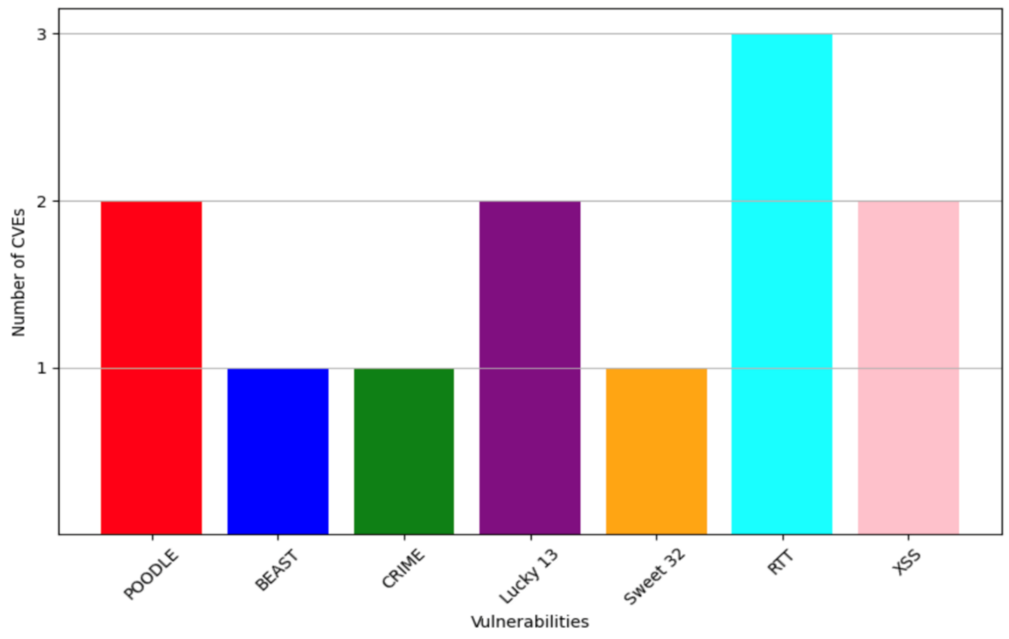

Figure 1 Impact vs Vulnerabilities.

5. Social Media Application

5.1 Meta Facebook

Under the umbrella of Meta Platforms Inc., formerly known as Facebook Inc., the company has been at the forefront of innovating and deploying robust and secure communication protocols tailored to its vast digital ecosystem. These endeavors aim at enhancing the safety and security of the platform while ensuring the privacy and protection of client data across its network.

Among the notable contributions from Facebook to the realm of internet protocols are mvfst, Fizz, and Proxygen. These tools, developed to address specific needs at different times, represent Facebook's commitment to optimizing internet communication through advanced networking and security libraries. Proxygen, introduced in 2014, serves as a comprehensive HTTP stack designed to seamlessly integrate with Facebook's sprawling infrastructure. It facilitates the implementation of emerging protocols such as SPDY and WebSocket, while also supporting monitoring tools like Thrift and ODS. Despite its scalability and integration benefits, Proxygen introduces complexities in maintenance and potential security vulnerabilities, largely due to its reliance on modern C++ practices and external libraries. Such dependencies might lead to unique challenges in performance under certain conditions (Sommerman & Frindell, 2014). The in-house development of Proxygen, while ensuring a tailored fit for Facebook's requirements, might limit the breadth of external scrutiny and review.

Fizz, launched in 2018, is a state-of-the-art TLS library that adopts TLS 1.3, with a particular focus on enhancing security and reducing latency, especially beneficial for mobile platforms. It introduces features like 0-RTT resumption, zero-copy encryption, and is built for asynchronous operations, catering to distributed server environments. However, the implementation of TLS 1.3 and 0-RTT data handling in Fizz poses potential vulnerabilities due to the complexity of the TLS protocols and the nuances of integrating with Facebook's systems. Challenges with middleboxes and navigating the TLS state machine further complicate the deployment of Fizz (Guzman et al., 2018; S. Iyengar, 2018).

Mvfst represents Facebook's venture into optimizing network protocols through the adoption of QUIC, aiming to enhance user experiences by addressing inherent limitations in TCP. By transitioning to QUIC and HTTP/3, Facebook sought to mitigate issues like request errors, tail latency, and the inefficiencies introduced by TCP's head-of-line blocking and network ossification. Despite the advancements, the integration of QUIC within Facebook's ecosystem presents its own set of challenges, including potential security vulnerabilities and performance dilemmas, necessitating a delicate balance in its deployment to achieve the intended efficiency improvements (Joras & Chi, 2020).

The integration of 0-RTT capabilities in QUIC and Facebook's Zero protocol significantly enhances network communication by enabling quicker data transmission without necessitating a full handshake, thereby reducing latency and ensuring swift server responses. This advancement, however, introduces certain vulnerabilities, notably replay attacks, which could result in issues such as duplicate transactions. In response to these vulnerabilities, Facebook has implemented measures including the use of time-bound 0-RTT data and a replay cache designed to identify and mitigate duplicates. Despite these precautions, the challenges of precise timekeeping and effective cache management remain, particularly in high-traffic environments. These challenges highlight the critical need to strike a balance between security and performance. Furthermore, in the realm of cryptographic design, the evolution of key derivation methods and the introduction of features such as the diversification nonce introduce additional complexity. These features are intended to enhance encryption robustness and ensure forward secrecy. Consequently, these advancements necessitate a meticulous selection of cryptographic algorithms, strict adherence to secure coding practices, and heightened vigilance against emerging threats. This vigilance underscores the perpetual challenge of advancing security measures while simultaneously maintaining optimal system performance.

Facebook's development and implementation of protocols such as Fizz, Proxygen, and mvfst have markedly improved network communication and security. Nonetheless, these protocols are not without their unique vulnerabilities. Specifically, Fizz encounters three primary vulnerabilities, with notable difficulties in handling 0-RTT data and the complexities associated with the implementation of TLS 1.3. Proxygen is identified with six vulnerabilities, grappling with issues related to scalability, its reliance on modern C++ and external libraries, and potential security weaknesses. Mvfst, on the other hand, is confronted with two main security challenges: the intricacies of integrating with existing systems and potential performance bottlenecks. These identified vulnerabilities highlight the indispensable need for ongoing security evaluations, timely updates, and a balanced methodology in the development of network protocols. This balanced approach is crucial for ensuring that the substantial benefits offered by these protocols are not undermined by security risks (Adami et al., 2022).

5.2 Instagram

Instagram, operating under the auspices of Meta Platforms Inc., is dedicated to creating a safe and secure platform that aligns with Meta's overarching commitment to user safety and privacy. This social media giant captivates its audience by allowing them to share photos and videos, connect with friends and family, follow public figures and brands, and explore new content through its engaging interface. Distinct from Facebook, Instagram places a strong emphasis on user safety and privacy, continuously evolving its interaction management tools and data visibility features to safeguard its community.

The platform is renowned for constantly rolling out innovative features such as Stories, Reels, and IGTV, aimed at fostering user engagement and creativity. Instagram's use of advanced algorithms for personalized content curation, along with its comprehensive advertising and monetization strategies for businesses and influencers, significantly enhances user experience. It also takes proactive measures to combat misinformation by leveraging partnerships and artificial intelligence. Through a combination of automated systems and human moderation, Instagram enforces stringent community guidelines to maintain a respectful and positive environment. Acknowledging the profound impact of social media on mental health, Instagram has introduced time management tools and mental health resources, underscoring its commitment to the well-being and safety of its users.

Meta, as the parent entity of both Facebook and Instagram, utilizes cutting-edge protocols such as mvfst, RSYS, and Fizz to secure data across its platforms, ensuring uniform security standards (Khot, 2020). A noteworthy advancement across Meta's platforms is the enhancement of messaging, along with audio and video calling functionalities, made possible by the RSYS video calling library. Developed using the latest WebRTC technologies, RSYS has become a cornerstone for Instagram, Messenger, Portal, and Workplace chat, especially vital for Instagram's visually-driven and interactive content. Compatible with Android and iOS, RSYS addresses the needs of Instagram's diverse user base with its reduced size, which enhances performance by approximately 20% compared to its predecessor, and its extensive test coverage ensures a seamless user experience. Designed to perform under various network conditions, RSYS is well-suited for Instagram's global audience, setting the stage for future innovations such as cross-application communication and enhanced remote presence functionalities.

While the integration of advanced network protocols like Fizz, RSYS, and mvfst into Instagram's infrastructure bolsters network communication and security, it also presents unique challenges. Fizz, in particular, exhibits vulnerabilities in handling 0-RTT data and the complexities of implementing TLS 1.3, necessitating continuous security monitoring and updates. RSYS, built on the WebRTC library, faces potential risks such as data leakage and unauthorized access to camera and microphone functionalities. These concerns are exacerbated by the need for cross-platform compatibility and its application across various Meta platforms, heightening the risk of security breaches. The development and maintenance of Fizz and RSYS illustrate the critical balance between adopting innovative technologies and managing associated risks, highlighting the importance of rigorous security assessments and updates to safeguard user data and maintain the integrity of the platform (Guzman et al., 2019).

5.3 WhatsApp

WhatsApp, a prominent messaging application under the ownership of Meta Platforms Inc., is designed to facilitate seamless and secure communication. It allows users to exchange text messages, share multimedia content including photos and videos, and make voice and video calls. Furthermore, it provides the functionality to create group chats, enhancing user engagement and connectivity.

The application employs the Signal protocol for end-to-end encryption (E2EE) across all forms of communication, including messages, voice calls, and video calls. This encryption ensures that only the communicating users can access the message content, safeguarding privacy from the moment the message is sent until it is received. The security of these communications is bolstered by the implementation of the Double Ratchet Algorithm, prekeys, and the Extended Triple Diffie–Hellman (X3DH) key agreement protocol. The Double Ratchet Algorithm uses the Diffie-Hellman key exchange mechanism to frequently update the shared secret between communicating parties. WhatsApp enhances this security by generating new Diffie-Hellman key pairs for each message, ensuring that each message has a unique encryption key derived from the public key sent with the message and the recipient's private key (WhatsApp, 2016).

Further securing the communication, WhatsApp utilizes the Noise Protocol Framework with Noise Pipes, employing Curve25519, AES-GCM, and SHA256. This setup is designed for persistent interactive connections, offering rapid and efficient connection setups, metadata encryption for additional user anonymity, and secure client authentication by storing only the public keys on the servers. WhatsApp's approach to E2EE ensures that messages are encrypted from the sender's device to the recipient's device, without the possibility of third-party access, including WhatsApp itself.

For communications through WhatsApp Messenger or the Business App, messages are straightforwardly end-to-end encrypted. However, for Business API users, the encryption remains intact only if they manage their API endpoint servers. Should a business opt to delegate their WhatsApp Business API endpoint to a third-party service, including Facebook, the messages are no longer considered to be under end-to-end encryption. WhatsApp maintains its commitment to encryption across all chats, with the Signal protocol ensuring encryption regardless of the E2EE status. Users are kept informed about the encryption status, particularly when interacting with businesses that utilize third-party managed API endpoints. The protocol is designed to ensure that decryption keys are stored solely on the devices involved in the conversation, thereby preventing WhatsApp from accessing or decrypting messages (Dowling & Hale, 2021; WhatsApp Security, n.d.)

In 2022, WhatsApp encountered a significant security vulnerability where an integer underflow issue in versions prior to v2.22.16.2 on Android and v2.22.15.9 on iOS created a potential risk for remote code execution through specially crafted video files. Despite the robust encryption provided by the Signal protocol, it is not immune to security challenges. For instance, the absence of 'per epoch' authentication could allow attackers with compromised private keys to impersonate users, and it is susceptible to Display Overlay Attacks. These attacks involve deceiving users by overlaying a fraudulent QR code over the Signal Safety Number verification interface. These vulnerabilities underscore the ongoing security challenges faced by widely utilized communication platforms like WhatsApp, highlighting the importance of continuous vigilance and updates to safeguard user privacy and security (WhatsApp, n.d.).

5.4 Snapchat

Snapchat, created by Snap Inc., stands out in the realm of social media as a multimedia messaging app that primarily focuses on the sharing of pictures and videos, termed "Snaps." What sets Snaps apart is their ephemeral nature; they vanish after being viewed by the recipient, adding a layer of privacy and immediacy to digital communication.

The adoption of QUIC, a transport protocol developed by Google, has been pivotal in enhancing the user experience on Snapchat. As the foundation for HTTP/3, QUIC is built atop UDP and offers a modern alternative to the conventional TCP+TLS+HTTP/2 stack. It tackles both transport-layer and application-layer challenges, streamlining the process for application developers with minimal required adjustments. For Snapchat, the introduction of QUIC has brought numerous advantages, including expedited connection setups through zero round-trip time (0-RTT) handshakes and enhanced congestion management with algorithms like QUIC BBR v1 and v2. Its support for multiplexing eliminates head-of-line blocking, significantly cutting down latency—a vital benefit for users on mobile networks. Additionally, QUIC facilitates connection migration, allowing for seamless service continuity even when IP addresses change, and it enhances the detection of lost connections, thus reducing the repercussions of connectivity issues. The integration of QUIC has markedly boosted Snapchat's performance, as evidenced by decreased latency and fewer errors, especially in regions plagued by poor connectivity. Leveraging Cronet, an open-source library, Snapchat has incorporated QUIC into its client network stack, not only enhancing network efficiency but also augmenting observability through comprehensive metrics and logging. This improvement has led to a more fluid and responsive experience for Snapchat users (Snap Inc., 2021).

However, like any popular platform, Snapchat has faced its security hurdles, highlighted by various Common Vulnerabilities and Exposures (CVEs). A notable recent CVE concerned the TIOCLINUX IOCTL request vulnerability, which presents a security risk by allowing malicious Snaps to tamper with terminal inputs. This vulnerability could enable the execution of unauthorized commands outside the Snap's sandbox environment upon its termination, posing a potential threat to system security. It is crucial to mention that this specific vulnerability primarily affects snaps executed on virtual consoles and is less likely to impact users on graphical terminal emulators, such as xterm or gnome-terminal. These incidents underline the critical need for ongoing security attention and the implementation of stringent security protocols to safeguard users and maintain the integrity of platforms as widely utilized as Snapchat.

5.5 LinkedIn

LinkedIn stands as the premier professional networking platform on the internet, offering a rich array of services for job and internship discovery, professional networking, and skill development. Accessible through desktop, mobile app, or web browser, LinkedIn enables users to create detailed profiles showcasing their career experiences, skills, and educational background. This facilitates meaningful connections with potential opportunities and industry peers. Furthermore, LinkedIn enhances professional growth by providing features for organizing events, joining groups, publishing content, and sharing multimedia, thus establishing itself as a comprehensive tool for professional advancement. On the technical front, LinkedIn prioritizes user data security by implementing OAuth 2.0 for authorization and adopting the TLS 1.2 protocol for data transmission, ensuring both a safe and reliable user experience (Hardt, 2012).

Like many digital platforms, LinkedIn has encountered security challenges, particularly with vulnerabilities associated with the TLS 1.2 protocol. Despite TLS 1.2's general reliability, it has been vulnerable to several well-documented security issues over time, including BEAST, CRIME, POODLE, Lucky Thirteen, and SWEET32. These vulnerabilities have necessitated the implementation of various countermeasures and updates to maintain the integrity of communications secured by TLS 1.2 (Ristić, 2015). Consequently, the cybersecurity industry has advocated for a shift towards more secure protocols, such as TLS 1.3, and emphasized the importance of secure configuration practices to navigate the evolving cybersecurity threat landscape effectively (Oppliger, 2023). LinkedIn has addressed specific vulnerabilities, notably an issue in its implementation of the dustjs library up to version 2.x, highlighting the ongoing need for diligence and proactive security measures on platforms that manage significant volumes of sensitive user data.

Transport Layer Security (TLS) protocols, versions 1.2 and 1.3, are among the most widely used for encrypting data transmitted across the internet. While TLS 1.3 is regarded as the superior option for security, it is not universally applicable due to compatibility issues. Many devices, browsers, and servers have yet to support TLS 1.3, potentially leading to user attrition and decreased engagement. Furthermore, the enhanced security provided by TLS 1.3 does not invariably lead to improved performance; in some instances, it may introduce delays in the encryption process, adversely affecting user experience. Despite these considerations, TLS 1.2 is known for its security challenges. However, its widespread acceptance and compatibility track record make it a viable option for many organizations. The transition to TLS 1.3, while offering advanced security features, faces adoption barriers due to compatibility issues with legacy systems and the potential for significant implementation costs and time. TLS 1.2 remains a well-established, user-friendly standard, offering a balance of security and performance that is straightforward to implement, underscoring the need for a pragmatic approach to adopting newer security protocols.

5.6 Slack

Slack is a business communication application that facilitates the connection of individuals with the necessary information. The platform's servers are pivotal in supporting Slack by managing message and file storage, enabling real-time messaging and calls, ensuring data security, managing user access, and integrating with other software tools. These servers are essential for upholding the platform's performance, reliability, and security. Slack has instituted a thorough security program aimed at protecting its software development processes and customer data. This program integrates a Secure Development Lifecycle (SDL) that conducts risk assessments grounded in the OWASP Top 10 and benefits from the insights of Slack's dedicated Product Security team (OWASP Foundation, 2021).

The network infrastructure of Slack is carefully segmented into testing, development, and production environments, ensuring that customer data is exclusively housed within the secure confines of the production network. This adherence to the Principle of Isolation is critical for restricting access and minimizing potential security vulnerabilities. Internet access to Slack's production network is rigorously controlled, with only a limited number of servers being accessible online and strict limitations on permissible network protocols at the network's edge. Slack deploys advanced measures to counteract threats such as Distributed Denial of Service (DDoS) attacks, emphasizing the use of sophisticated encryption and authentication methods (Slack, n.d.-b). Data in transit is shielded by TLS 1.2 protocols (Dierks & Rescorla, 2008), AES256 encryption, and SHA2 signatures, whereas data at rest benefits from encryption practices that comply with the FIPS 140-2 standards (Evans, 2001). The management of encryption keys is conducted on separate servers with limited access, enhancing security. For authentication purposes, Slack leverages multi-factor authentication and private keys, particularly for sensitive data access. This is supplemented by intricate password policies and the encouragement of password manager use.

Furthermore, Slack's security arsenal includes exhaustive system monitoring, detailed logging, and stringent change control measures to defend against vulnerabilities and malware. Intrusion Detection and Prevention Systems (IDS/IPS) rely on host-based controls, given that network device management is the responsibility of their hosting provider. Slack actively monitors, logs, and audits activities within its production network, such as system calls, to develop alerts for potentially intrusive actions. The Principle of Least Privilege is meticulously applied across Slack's production network, ensuring that changes are strictly managed and only authorized personnel can make modifications, thus maintaining a high level of oversight (Slack, n.d.-a) .

When evaluating Slack's network security, the employment of the Principles of Least Privilege and Isolation is vital for reducing the risk of security breaches. However, the use of TLS 1.2, despite bolstering data security, presents vulnerabilities, including those known as BEAST, CRIME, POODLE, Lucky Thirteen, and SWEET32 (Ristić, 2015). Addressing these issues requires Slack to adopt a proactive stance, implementing countermeasures, updating systems, and securing configurations through methods such as disabling weak ciphers, employing forward secrecy, and ensuring robust key management practices. This highlights the necessity for continual vigilance and adaptation in network security to combat emerging threats and safeguard the integrity and security of customer data within Slack's network infrastructure.

5.7 Twitter[X]

Twitter, now rebranded as X, is a prominent social media platform celebrated for its microblogging service that enables users to post concise messages, known as "tweets." This platform is widely recognized for its role in the instantaneous dissemination of news, fostering networking opportunities, and hosting public discourse. Serving as a critical hub for marketing, activism, and entertainment, X (formerly Twitter) bridges communication gaps between individuals, celebrities, businesses, and media entities. Its capability to spotlight trending topics and facilitate swift exchanges of information positions it as a significant force in molding public opinion and steering social trends.

X maintains a comprehensive security framework to protect user data, incorporating a mixture of sophisticated encryption technologies and protocols. For data in transit, including tweets and direct messages from a user's device to X's servers, the platform employs TLS (Transport Layer Security) 1.2 encryption. This protocol encrypts data during its journey across the internet, safeguarding it against unauthorized access or interception. Moreover, for connections between users' devices and its servers, X utilizes the advanced and more secure TLS 1.3 protocol, which provides superior security features and quicker connection speeds, thereby enhancing the overall security and efficiency of the user experience (Rescorla, 2018). In the realm of authentication, X integrates OAuth 2.0 and is considering the implementation of OAuth 3.0. These standards offer secure, token-based authentication mechanisms, enabling users to permit websites or applications to access their information on other web services securely, without exposing their passwords (Hardt, 2012). Additionally, X ensures secure communication between users' web browsers and its platform by implementing HTTPS, an encrypted version of HTTP. This multifaceted approach to security, encompassing diverse encryption and authentication methods, underscores X's commitment to safeguarding user data and ensuring a secure online presence (Twitter, n.d.).

Despite the robust security measures in place, the protocols employed by X, including TLS 1.2 and 1.3, OAuth 2.0, and HTTPS, are not devoid of security vulnerabilities. TLS protocols, which secure data in transit, can be compromised through weak cipher suites, implementation errors, downgrade attacks, and session hijacking risks (Ristić, 2015). OAuth protocols, essential for authentication, are susceptible to challenges such as manipulation of redirect URLs, exposure of access tokens, and client impersonation (OWASP Foundation, 2021). Even though HTTPS encrypts communication, it remains vulnerable to man-in-the-middle attacks and issues related to certificates. A specific vulnerability identified as CVE-2023-29218 (National Vulnerability Database NIST, n.d.-b) highlights a unique security concern on X. This vulnerability allows the manipulation of X's Recommendation Algorithm to conduct a denial-of-service attack, adversely impacting a target account's reputation score through coordinated activities like unfollowing and blocking. While X views the influence of such actions on its algorithm as a design choice, this incident illustrates the potential for exploitation of platform features, emphasizing the need for ongoing vigilance and adaptive security measures to contend with evolving threats and safeguard the integrity and security of user data within its network infrastructure (National Vulnerability Database NIST, n.d.-b).

5.8 Discord

Discord has evolved from a platform initially designed for gamers to a comprehensive communication tool utilized by a wide range of communities. It offers an array of interactive features, including voice and video calls, text messaging, and the ability to share media and files. Users have the option to engage in private conversations or become part of larger communities, known as "servers." These servers function as virtual spaces where individuals can gather to discuss, share, and collaborate, making Discord a versatile choice for various purposes beyond gaming, such as socializing, education, business meetings, and collaborative projects (NTT Communication Project, n.d.).

At the heart of Discord's real-time communication capabilities is the WebRTC (Web Real-Time Communication) standard. This technology, which integrates networking, audio, and video functionalities, is supported by contemporary web browsers and can be incorporated into native applications via libraries. Discord's application architecture is designed to operate flawlessly across web browsers (including Chrome, Firefox, and Edge), desktop applications (for Windows, macOS, and Linux), and mobile apps (for iOS and Android). This cross-platform compatibility is achieved through strategic code reuse and the adept utilization of WebRTC. Specifically, Discord's browser application leverages the WebRTC functionalities built into browsers, while its desktop and mobile applications employ a custom C++ media engine derived from the WebRTC native library to enhance user experience features such as volume control and bandwidth optimization (Vass, n.d.).

On the backend, Discord employs key services developed in Elixir, including the Discord Gateway, Discord Guilds, and Discord Voice. The Gateway is responsible for managing WebSocket connections, facilitating real-time updates, whereas the Discord Voice server processes audio data in voice channels, optimizing resource allocation based on server demand and geographic location. A distinctive aspect of Discord's backend infrastructure is its custom Selective Forwarding Unit (SFU), which is tailored to efficiently route audio and video traffic, thereby ensuring high-quality communication across various devices and platforms (Vass, n.d.).

Despite the advantages offered by WebRTC, Discord faces security challenges common to applications that utilize this technology. These challenges include vulnerabilities to cross-site scripting (XSS) attacks, which could allow attackers to inject harmful JavaScript, potentially compromising user privacy by exposing sensitive information such as IP addresses. Moreover, the platform's file-sharing feature could pose security risks if stringent checks on transmitted files are not in place, potentially facilitating the dissemination of malware. Discord has addressed specific vulnerabilities, including CVE-2021-29466 (National Vulnerability Database NIST, n.d.-a) and CVE-2021-29465 (National Vulnerability Database NIST, n.d., p. 29465), which impacted the Discord-Recon bot and could have allowed remote attackers to access or modify local files on the server, posing a risk of remote code execution. Discord's proactive measures to remedy these vulnerabilities in subsequent software updates highlight the platform's commitment to maintaining a secure and trustworthy environment for its users.

6. Comparative Analysis of Encryption Protocol Deployment Across Leading Social Media Platforms

In the previous section, we delved into the encryption protocols adopted by various digital platforms, offering an overview of how these technologies contribute to securing data and communication. The following summarizes how each social network platform deploys one or more encryption protocols to ensure data privacy and secure communication.

Platform-Specific Encryption Protocols:

- **Facebook and Instagram**: Both platforms implement a combination of protocols, including QUIC/mvfst, TLS 1.3/Fizz, and TLS 1.2/Proxygen. This multifaceted approach ensures robust security and high performance.
- **WhatsApp**: Exclusively uses the Signal/Noise Pipes protocol for its end-to-end encryption needs, providing a focused approach to secure communication.
- **Snapchat**: Relies on QUIC/mvfst for its encryption processes, reflecting its reliance on this single protocol for security.
- **LinkedIn**: Utilizes a dual-protocol strategy, incorporating TLS 1.2/Proxygen and OAuth. This combination balances security and user authentication.
- **Slack**: Uses TLS 1.2/Proxygen as its sole encryption mechanism, indicating a straightforward approach to securing its communications.
- **Twitter (X)**: Integrates both TLS 1.3/Fizz and OAuth into its security architecture, reflecting a robust approach to encryption and authentication.
- **Discord**: Adopts WebRTC/RSYS for its encryption needs, emphasizing real-time communication security through this protocol.

The above can be summarized in the table below. It provides an overview of the encryption protocols deployed by these industry leaders to protect user data across different states—at rest, in transit, and in use.

Table 1 Encryption protocols used by the leading platforms or applications.

Platform/Application Name	**Company**	**End-to-End Encryption Protocols**
Facebook	Meta	TLS 1.3 v(Fizz), QUIC (mvfst) and HTTP(Proxygen)
Instagram	Meta	TLS 1.3 v(Fizz), QUIC (mvfst), and WebRTC(RSYS)
WhatsApp	Meta	Signal
Slack	Slack	TLS 1.2 V
Snapchat	Snapchat	QUIC
Twitter	Twitter(X)	TLS V 1.2, TLS V 1.3, OAuth 2.0, and HTTPS
LinkedIn	Microsoft	TLS 1.2 V, HTTPS, OAuth 2.0, and OAuth 3.0
Discord	Discord	WebRTC

The Figure 2 below presents a comparative analysis through a visual representation. In the figure, the vertical axis labeled “Presence of Combined Protocol” indicates the implementation status of specific protocols within the security framework of each application, with a value of 1 signifying implementation and 0 indicating absence. This graphical representation allows for an at-a-glance comparison of the encryption protocols utilized by each platform, highlighting their approach to ensuring data privacy and secure communication. This comparative diagram serves as a visual guide to understanding the diversity of encryption protocols across different platforms, reflecting each application's strategic choice in balancing security, performance, and user experience.

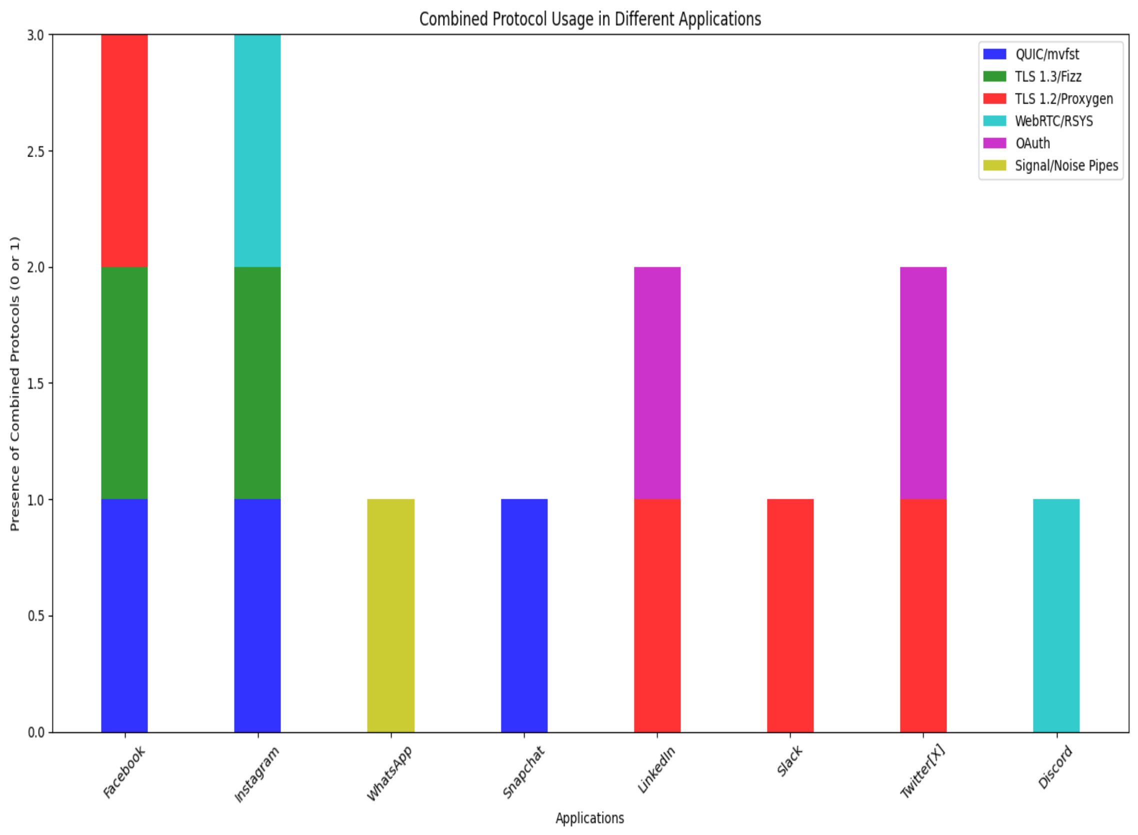

Figure 2 Encryption protocols used in the platforms/applications.

7. Distribution and Adoption of Encryption Protocol Across Major Social Platforms

In this section, we analyze the distribution of encryption protocols across major social media platforms. The focus here is to identify which encryption protocols are most widely adopted by these platforms, reflecting industry trends and preferences in ensuring data security and privacy.

The table below provides an overview of the encryption protocols used by leading social media platforms. It highlights the specific protocols each platform has

Protocol	Number of platforms using the protocol
TLS 1.2	3 (Slack, Twitter, Linked In)
TLS 1.3	3 (Facebook, Twitter, Instagram)
QUIC	3 (Facebook, Instagram, Snapchat)
WebRTC	2 (Discord, Instagram)
Signal	1 (WhatsApp)

implemented, showcasing the widespread adoption of certain encryption standards.

Error! Reference source not found. visually represents the distribution of encryption protocols across the major platforms. This figure illustrates the adoption rate of each protocol, providing a clear understanding of which encryption methods are preferred by industry leaders. The table and pie chart show the proportion of platforms using each protocol and highlight the following key points:

Table 2 Distribution of Encryption Protocols Across Platforms

1. **TLS 1.2**: Widely adopted by platforms like Slack, Twitter, and LinkedIn, indicating its continued relevance despite the emergence of newer protocols.
2. **TLS 1.3**: Shows significant adoption, particularly by Facebook, Twitter, and Instagram, reflecting its enhanced security features and performance improvements over TLS 1.2.
3. **QUIC**: Embraced by Facebook, Instagram, and Snapchat, demonstrating its growing popularity for improving connection speed and security.
4. **WebRTC**: Used by Discord and Instagram, emphasizing its importance for real-time communication applications.
5. **Signal**: Exclusively adopted by WhatsApp, showcasing its strong focus on end-to-end encryption for secure messaging.

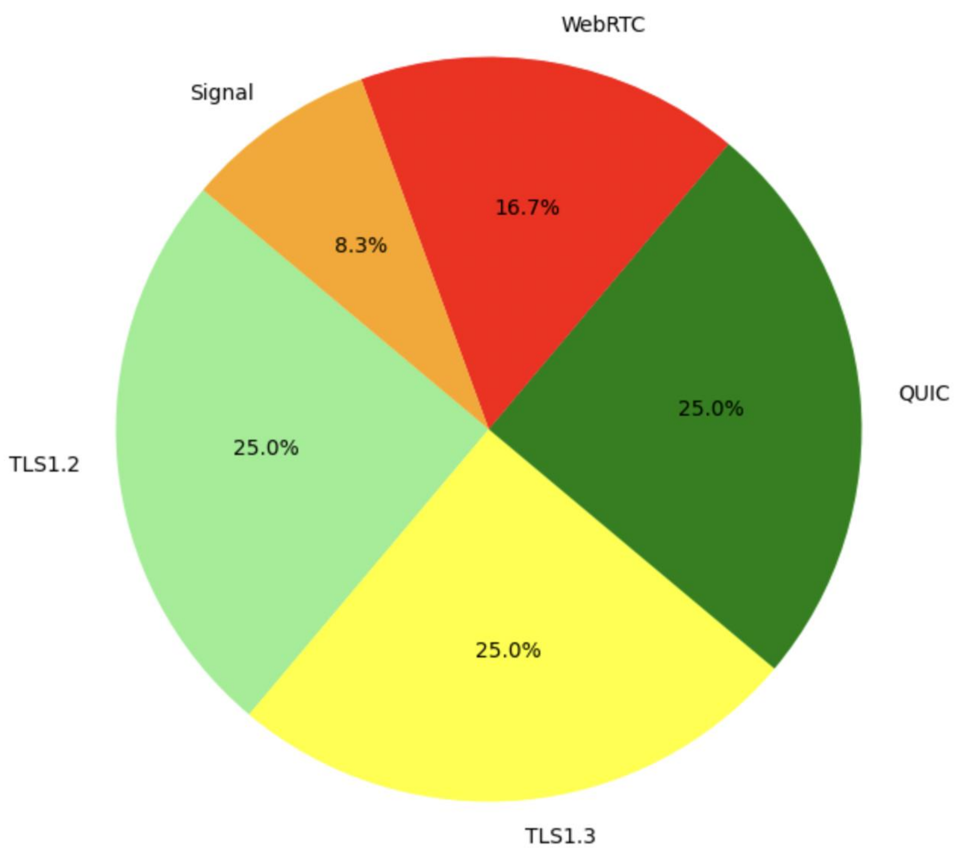

Figure 3 Encryption Protocol Distribution

8. Comprehensive Ranking of Social Media Applications Based on Encryption Protocol Security

In our detailed analysis, we delved into the complex realm of encryption protocols and their adoption by prominent social media and messaging platforms. Key protocols, including Transport Layer Security (TLS) versions 1.2 and 1.3, Signal, QUIC, and WebRTC, are pivotal in safeguarding data privacy and securing communications. Our research uncovered that each protocol has its unique strengths and weaknesses, which significantly impact the security posture of the applications they safeguard.

Application Security Rankings Based on Protocol Vulnerabilities

1. **WhatsApp (Meta)**: WhatsApp, employing the Signal protocol, ranks as the most secure in our analysis. Its use of Curve25519, AES-GCM, and SHA256 within the Noise Protocol Framework adds significant security layers, albeit facing challenges like integer overflow vulnerabilities.
2. **LinkedIn (Microsoft):** LinkedIn maintains a robust security posture using TLS 1.2 and OAuth protocols. Its proactive approach in addressing vulnerabilities such as BEAST and POODLE, coupled with its professional network context, secures it a high position in terms of security.
3. **Slack:** Slack's use of TLS 1.2 and comprehensive internal security measures, including the Principle of Least Privilege and Principle of Isolation, positions it as a secure option for business communications despite known TLS vulnerabilities.
4. **Twitter (X):** Twitter's implementation of TLS 1.2 and 1.3 alongside OAuth protocols provides robust security for data in transit. Nevertheless, vulnerabilities in OAuth and the recent CVE-2023-29218 highlight potential security risks.
5. **Snapchat (Snap Inc.):** Snapchat's adoption of QUIC offers improved performance, yet it also brings security concerns, particularly regarding 0-RTT vulnerabilities and potential data stream manipulation.
6. **Discord:** Discord, utilizing WebRTC, ensures efficient and secure real-time communication. Nonetheless, it contends with typical WebRTC-based application challenges such as XSS attacks and file-sharing vulnerabilities.
7. **Instagram (Meta):** Similar to Facebook in its use of Fizz, Proxygen, and mvfst, Instagram confronts unique challenges in RSYS video calling and potential vulnerabilities in its TLS implementation, resulting in a lower security ranking compared to WhatsApp.
8. **Facebook (Meta):** Facebook’s advanced protocols such as Fizz, Proxygen, and mvfst enhance network communication but introduce specific vulnerabilities, particularly in 0-RTT data handling and TLS 1.3 implementation complexity, leading to a lower security ranking.

New Ranking Methodology

To accurately rank various applications, we established a scoring system that evaluates them based on multiple criteria pertaining to security and encryption standards. Below is an in-depth explanation of the ranking process:

Criteria Definition:

- **End-to-End Encryption**: Applications employing end-to-end encryption received a bonus of +10 points, highlighting its importance in privacy and security by ensuring that only the communicating users can access the messages.
- **Modern Encryption Protocols**: The adoption of contemporary protocols such as TLS 1.3 or QUIC was rewarded with +8 points, reflecting the latest advancements in secure communication.
- **Older Encryption Protocols**: Applications utilizing older but secure protocols like TLS 1.2 or WebRTC were given +6 points.
- **Proactive Security Measures**: Applications that demonstrated a proactive approach to security, including regular security audits and timely updates, were granted an additional +2 points.
- **Vulnerabilities**: The presence of each known vulnerability led to a deduction of 1 point from the application's total score.

Score Assignment:

- Applications were assessed based on the aforementioned criteria, with points allocated for encryption usage and proactive security efforts, while deductions were made for each identified vulnerability.

Adjustment for Vulnerabilities:

- The final score of each application was adjusted by subtracting points for any known vulnerabilities, offering a net perspective on the application's security posture.

Sorting and Ranking:

- Applications were then ordered in descending sequence according to their adjusted scores, providing a relative comparison of their security features and vulnerabilities.

Ranked Applications:

Application	Encryption Strength	Proactive Security Measure	Vulnerability	Total Points
WhatsApp	10 (end-to-end)	Noise Protocol Framework (+2)	CVE-2021-24042, CVE-2022-27492, CVE-2022-36934 (-3)	9 points
LinkedIn	6 (TLS 1.2)	Principle of Least Privilege (+2)	None	8 points
Slack	6 (TLS 1.2)	Principle of Least Privilege and	None	8 points

		Principle of Isolation (+2)		
Twitter	8 (TLS 1.3, TLS 1.2)		CVE-2023-29218 (-1)	7 points
Snapchat	8 (QUIC)		CVE-2023-1523 (-1)	7 points
Discord	6 (WebRTC)		CVE-2021-29466 (-1)	5 points
Instagram	8 (TLS 1.3, QUIC, WebRTC)		CVE-2023-44487, CVE-2023-29747, CVE-2023-29748, CVE-2023-23759, CVE-2020-1895, CVE-2014-6690 (-6)	2 points
Facebook	8 (TLS 1.3, QUIC, Proxygen)		CVE-2023-44487, CVE-2023-23759, CVE-2021-24218, CVE-2021-24217, CVE-2014-6392, CVE-2008-0660 (-6)	2 points

This scoring and ranking approach offers a systematic method to assess and contrast the security attributes of different applications, acknowledging both their strengths, such as the implementation of sophisticated encryption techniques, and their weaknesses, such as known security vulnerabilities.

Our ranking is also depicted in the pie chart provided below for easy comparison:

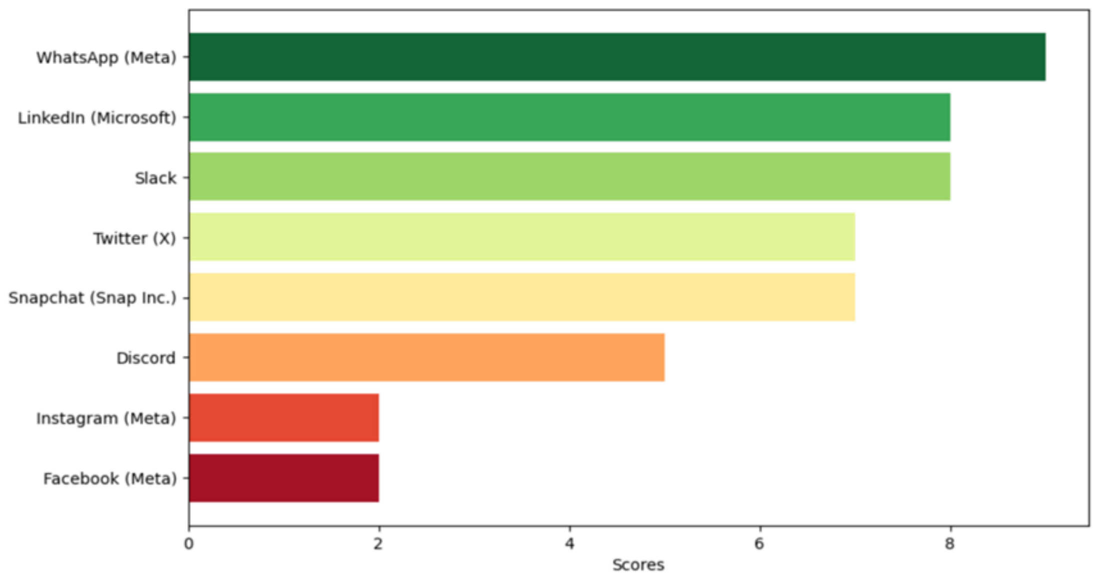

Figure 4 Ranking of Application Platforms Based on the Encryption Protocols

TLS 1.2, widely adopted, has shown susceptibility to various known vulnerabilities, leading to a gradual shift towards the more secure and efficient TLS 1.3. The Signal protocol, especially within WhatsApp, is notable for its robust end-to-end encryption, ensuring message confidentiality from sender to receiver. QUIC, used by platforms like Snapchat, improves performance but faces challenges with 0-RTT vulnerabilities and data stream manipulation. WebRTC, vital for real-time communications in apps like Discord, struggles with XSS attacks and the risk of malware spread through file sharing.

The ranking method developed here allows us to systematically assess the security posture of various social media applications based on the encryption protocols they use. By focusing on encryption strength, proactive security measures, and vulnerabilities, we provide a comprehensive view of each platform's security. This method underscores the critical importance of adopting advanced encryption standards and proactive security measures to ensure robust data protection in the evolving digital landscape. It's important to recognize that application security is multifaceted, encompassing aspects like authentication mechanisms, user access controls, and software development practices, which are all critical in determining an application's overall security efficacy. Therefore, while this analysis offers significant insights into protocol-based security, it represents only one aspect of a broader security picture. Future research could explore other vital areas such as advanced authentication methods, providing a more comprehensive view of application security.

9. Conclusion

The adoption of end-to-end encryption across social media and messaging platforms marks a significant evolution in the landscape of digital communication. This study has meticulously examined the deployment of encryption protocols, emphasizing their crucial role in enhancing user data security and privacy. Notably, platforms such as Facebook, Instagram, Twitter, LinkedIn, WhatsApp, Discord, Snapchat, and Slack have implemented diverse encryption strategies, from WhatsApp's robust use of the Signal protocol to the advanced TLS 1.3 and QUIC implementations by Facebook and Instagram.

Our study contributes to the field of digital communication security in several significant ways. First, we provide a systematic evaluation of encryption protocols across multiple social media platforms, offering a comprehensive overview that is absent in existing literature. Second, we introduce a unique ranking system that assesses and compares the security measures of major platforms, providing valuable insights into their relative strengths and weaknesses. Third, our study offers a current analysis of vulnerabilities across various platforms and protocols, contributing to the ongoing discourse on social media security. Fourth, by providing a clear comparison of platform security, our research aims to inform user choices and promote awareness of encryption practices in social media. Lastly, our technical analysis provides a foundation for understanding the current state of encryption across major platforms, which can inform policy discussions and decisions.

Looking ahead, the future of end-to-end encryption hinges on the continual advancement of sophisticated encryption techniques and the establishment of global standards that ensure consistent security across all platforms. The insights provided by this study serve as a foundational step for further research and development in cryptographic security, paving the way towards a more secure digital communication environment.

References

Adami, N., Franco, M., Penna, F., & Palazzi, C. E. (2022). QUIC Employment: Comparing the Response Time of Facebook and Twitter. *2022 IEEE 42nd International Conference on Distributed Computing Systems Workshops (ICDCSW)*, 231–236.

Albrecht, M. R., Paterson, K. G., & Millican, J. (2016). The Signal protocol: Side channels, covert channels, and more. *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, 1465–1477.

Bhuse, V. (2023). Review of End-to-End Encryption for Social Media. *International Conference on Cyber Warfare and Security*, *18*(1), 35–37. https://doi.org/10.34190/iccws.18.1.1017

Borisov, N., Goldberg, I., & Brewer, E. (2004). Off-the-record communication, or, why not to use PGP. *Proceedings of the 2004 ACM Workshop on Privacy in the Electronic Society*, 77–84. https://doi.org/10.1145/1029179.1029200

CISA. (n.d.). *Secure by Design | CISA*. Retrieved January 16, 2024, from https://www.cisa.gov/securebydesign

Cobb, M. (n.d.). *The POODLE vulnerability and its effect on SSL/TLS security | TechTarget*. Security. Retrieved January 25, 2024, from https://www.techtarget.com/searchsecurity/tip/The-POODLE-vulnerability-and-its-effect-on-SSL-TLS-security

Cohn-Gordon, K., Cremers, C., Dowling, B., Garratt, L., & Stebila, D. (2016). *A Formal Security Analysis of the Signal Messaging Protocol*. https://eprint.iacr.org/2016/1013

Ćurguz, J. (2016). Vulnerabilities of the SSL/TLS Protocol. *Computer Science & Information Technology*, *6*, 245–256.

Dierks, T., & Rescorla, E. (2008). *RFC 5246: The Transport Layer Security (TLS) Protocol Version 1.2*. RFC Editor.

Dowling, B., & Hale, B. (2021). Secure Messaging Authentication against Active Man-in-the-Middle Attacks. *2021 IEEE European Symposium on Security and Privacy (EuroS&P)*, 54–70. https://doi.org/10.1109/EuroSP51992.2021.00015

Evans, D. (2001). *FIPS PUB 140-2 Security Requirements For Cryptographic Modules*.

F5 Networks. (n.d.). *CVE - CVE-2022-34651*. Retrieved January 16, 2024, from https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2022-34651

Guzman, A., Nekritz, K., & Iyengar, S. (2018, August 6). Deploying TLS 1.3 at scale with Fizz, a performant open source TLS library. *Engineering at Meta*. https://engineering.fb.com/2018/08/06/security/fizz/

Guzman, A., Nekritz, K., & Iyengar, S. (2019, November 1). Delegated credentials: Improving the security of TLS certificates. *Engineering at Meta*. https://engineering.fb.com/2019/11/01/security/delegated-credentials/

Hardt, D. (2012). *The OAuth 2.0 authorization framework*.

Hassani Karbasi, A., & Shahpasand, S. (2021). SINGLETON: A lightweight and secure end-to-end encryption protocol for the sensor networks in the Internet of Things based on cryptographic ratchets. *The Journal of Supercomputing*, *77*(4), 3516–3554. https://doi.org/10.1007/s11227-020-03411-x

Holmes, D. (n.d.). *Exposure to Lucky Thirteen; SSL Vulnerability* |. DevCentral - an F5 Community. Retrieved January 25, 2024, from https://community.f5.com/kb/technicalarticles/exposure-to-ldquolucky-thirteenrdquo-ssl-vulnerability/274387

Holmes, D. (2016, September 28). *I Got 99 Problems, But SWEET32 Isn't One*. SecurityWeek. https://www.securityweek.com/i-got-99-problems-sweet32-isnt-one/

Iyengar, J., & Thomson, M. (2021). *QUIC: A UDP-Based Multiplexed and Secure Transport* (Request for Comments RFC 9000). Internet Engineering Task Force. https://doi.org/10.17487/RFC9000

Iyengar, S. (2018, December 4). Moving fast at scale: Experience deploying IETF QUIC at Facebook. *Proceedings of the Workshop on the Evolution, Performance, and Interoperability of QUIC*. CoNEXT '18: The 14th International Conference on emerging Networking EXperiments and Technologies, Heraklion Greece. https://doi.org/10.1145/3284850.3322434

Joras, M., & Chi, Y. (2020, October 21). *How Facebook is bringing QUIC to billions* [Meta Engineering]. https://engineering.fb.com/2020/10/21/networking-traffic/how-facebook-is-bringing-quic-to-billions/

Khot, I. (2020, December 21). A smaller, faster video calling library for our apps. *Engineering at Meta*. https://engineering.fb.com/2020/12/21/video-engineering/rsys/

Kumar, A., Gupta, S. K., Rai, A. K., & Sinha, S. (2013). Social networking sites and their security issues. *International Journal of Scientific and Research Publications*, *3*(4), 1–5.

Kumari, S., & Singh, S. (2015). A Critical Analysis of Privacy and Security on Social Media. *2015 Fifth International Conference on Communication Systems and Network Technologies*, 602–608. https://doi.org/10.1109/CSNT.2015.21

National Vulnerability Database NIST. (n.d.-a). *NVD - CVE-2021-29466*. Retrieved January 25, 2024, from https://nvd.nist.gov/vuln/detail/CVE-2021-29466

National Vulnerability Database NIST. (n.d.-b). *NVD - CVE-2023-29218*. Retrieved January 25, 2024, from https://nvd.nist.gov/vuln/detail/CVE-2023-29218

National Vulnerability Database NIST. (n.d.). *NVD - CVE-2021-29465*. Retrieved January 25, 2024, from https://nvd.nist.gov/vuln/detail/CVE-2021-29465

Nidecki, T. A. (2020, May 21). *What Is the BEAST Attack*. Acunetix. https://www.acunetix.com/blog/web-security-zone/what-is-beast-attack/

NTT Communication Project. (n.d.). *A Study of WebRTC Security · A Study of WebRTC Security*. Retrieved January 16, 2024, from https://webrtc-security.github.io/

Oppliger, R. (2023). *SSL and TLS: Theory and Practice*. Artech House.

OWASP Foundation. (2021). *OWASP Top Ten*. https://owasp.org/www-project-top-ten/

Reiter, A., & Marsalek, A. (2017). WebRTC: Your privacy is at risk. *Proceedings of the Symposium on Applied Computing*, 664–669. https://doi.org/10.1145/3019612.3019844

Rescorla, E. (2018). *The Transport Layer Security (TLS) Protocol Version 1.3* (Request for Comments RFC 8446). Internet Engineering Task Force. https://doi.org/10.17487/RFC8446

Ristić, I. (2015). *Bulletproof SSL and TLS*. Feisty Duck.

Slack. (n.d.-a). *Security Practices | Legal*. Slack. Retrieved January 16, 2024, from https://slack.com/security-practices

Slack. (n.d.-b). *Slack's approach to security*. https://a.slack-edge.com/4c1ae/img/security_ent/Security_White_Paper.pdf

Snap Inc. (2021, June 24). *QUIC at Snapchat—Snap Engineering*. https://eng.snap.com/quic-at-snap

Sommerman, D., & Frindell, A. (2014, November 5). Introducing Proxygen, Facebook's C++ HTTP framework. *Engineering at Meta*. https://engineering.fb.com/2014/11/05/production-engineering/introducing-proxygen-facebook-s-c-http-framework/

Stier, S., Bleier, A., Lietz, H., & Strohmaier, M. (2020). Election campaigning on social media: Politicians, audiences, and the mediation of political communication on Facebook and Twitter. In *Studying Politics Across Media* (pp. 50–74). Routledge.

Sy, E., Burkert, C., Federrath, H., & Fischer, M. (2019). A QUIC Look at Web Tracking. *Proc. Priv. Enhancing Technol.*, *2019*(3), 255–266.

Twitter. (n.d.). *TLS*. Connecting to Twitter API Using TLS. Retrieved January 16, 2024, from https://developer.twitter.com/en/docs/authentication/guides/tls

Vass, J. (n.d.). *How Discord Handles Two and Half Million Concurrent Voice Users using WebRTC*. Retrieved January 16, 2024, from https://discord.com/blog/how-discord-handles-two-and-half-million-concurrent-voice-users-using-webrtc

WhatsApp. (n.d.). *WhatsApp Security Advisories*. WhatsApp.Com. Retrieved January 25, 2024, from https://www.whatsapp.com/security/advisories

WhatsApp. (2016). *WhatsApp Encryption Overview* [Technical White Paper]. https://scontent-ord5-2.xx.fbcdn.net/v/t39.8562-6/384251896_820338303082371_8514785982310046047_n.pdf?_nc_cat=100&ccb=1-7&_nc_sid=e280be&_nc_ohc=scT90_tZtCkAX9v-jxq&_nc_oc=AQldtRWxlFPVsfxSCKr2TNVnoVt6rHWg0UDkswrGyPgPVRAB8TBF7UtAHPKmUWQZm9A&_nc_ht=scontent-ord5-2.xx&oh=00_AfB-MbWderDWDRxmu4VOjEjPDmwZysDFTM9u8Pqld4WJrg&oe=65C25211

WhatsApp Security. (n.d.). *About end-to-end encryption*. Retrieved January 16, 2024, from https://faq.whatsapp.com/820124435853543/?helpref=hc_fnav