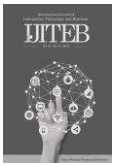
IJITEB
Vol. 6
No. 2
2024

International Journal of Information Technology and Business



Securing Text Messages Using E2EE *(End-To-End Encryption)* on Instant Messaging Applications

[a]Wasis Pancoro
[a]Satya Wacana Christian University,Indonesia

Keywords : End-to-End Encryption, Facebook Messenger, Salsa20, WhatsApp, X3DH

Abstract : with the development of technology in communication, many applications make it easy to send messages. security in sending messages is something that cannot be excluded. *E2EE (End-to-end encryption)* is one of the most encryption for securing messages. in the discussion of literature review produces information on how applications such as whatsApp, Skype, Facebook messenger, line and viber implement *E2EE* in encrypting text messages specifically.

1. Introduction

Progress in communication technology makes it easy to send messages quickly and safely. Many applications are developed by application developers to send instant and real time message delivery such as WhatsApp, Facebook Messenger, Line, Skype, e-Chat, Viber and others. Security communicating is important to avoid the potential of information tapping, so applications developed by application developers have their own way of securing messages sent from the sender to the recipient of the message. Securing messages on instant messaging applications using End-to-end encryption. Every instant messaging application, develops end-to-end encryption in different ways.

End-to-end encryption is a part of cryptography that is used for the security of messages sent from the sender to the recipient of both text, document, file, voice call, and video call messages. *End-to-end encryption (E2EE)* is one of the most widely used to send information to the internet safely. In principle, *E2EE* is a way to send information through a network in such a way that only recipients and senders can access it. *E2EE* contains components such as identity, protocol, algorithm, safe implementation, and safe operation. All of these components work together and run efficiently to provide the best security for end users. There are many different variations of each component that has been developed over the years for different applications [1].

Based on the importance of security in sending messages to instant messaging applications, this writing is a literature study that discusses how to use E2EE (End-to-end Encryption) in instant messaging applications in text message security. E2EE implementation that will be discussed at this writing is only on instant messaging applications such as WhatsApp, Facebook Messenger, Line, Skype and Viber.

Previous Research

In a previous study *"Penggunaan Enkripsi End-to-End dalam Pengamanan Pesan dan Video Call pada Whatsapp",* discussed how the WhatsApp Application applies End-to-end Encryption. Whatsapp uses end-to-end encryption that is private. End-to-end encryption is one of the cryptographic parts that provides high-level security features on WhatsApp both in message security and video calls. Messages sent via WhatsApp are text and attachments (documents, images, sounds, locations, contacts, and videos). The end-to-end encryption method provides protection for whatsapp users against hacking, brute force, etc. This not only applies to sending messages but also to facetime services or video calls. so that the information sent only has one path for the recipient. Users who have never had a call history on the recipient form an encrypted session by generating a random 32-bit SRTP master secret so that the encrypted message is sent in the form of an incoming call so that the SRTP automatically encrypts the call when the call is received [2].

research entitled "WhatsApp End-to-End Encryption :Are Our Messages Private? "gives a detailed description of the end to end algorithms used by signal protocols. Research is doing experimentation and designing that aims to compare traffic from both

applications, and to analyze the distinguishing features of WhatsApp. Results from experiments that implement WhatsApp the signal protocol is different from the Signal application implementation [3].

2. Literature Review

WhatsApp Inc. founded by Jan Koum and Brian Acton in Santa Clara, California in 2009. Starting as an iPhone application, WhatsApp later became popular and also available for Android, Windows Phone, BlackBerry and Nokia. In February 2014, Facebook Inc. bought WhatsApp for 19 billion. WhatsApp Messenger is a cross platform client message exchange application for smartphones. This application uses the internet to send text, document, image, video, user location and audio message media messages to other users who use standard cellular phone numbers [4].

Skype is a messenger application that is more focused on voice messenger services than text so that Skype is better known as a *VoIP (Voice over Internet Protocol)* application that is voice communication via internet protocol. Skype was first developed by a trio of Estonian programmers named Ahti Heinla, Priit Kasesalu, and Jaan Tallinn. Actually Skype is the result of the development of the Peer-to-Peer application program called Kazaa. Chat capabilities owned by Skype are almost the same as other messaging applications. We can share files, send messages, voice calls, video calls and play games [5].

Facebook Messenger is Facebook's instant messaging service, used to send messages in the form of text, document files, voice calls and video calls. Facebook launched Messenger in August 2011 after the acquisition of a group messaging application named Beluga. Although owned and operated by Facebook, the application and website are separate from Facebook. to use Messenger does not have to be on the Facebook website or even have a Facebook account, to use Messenger. Although both are partially connected when having a Facebook account.

LINE is a free instant messaging sender application that can be used on various platforms using the internet network so that *LINE* users can carry out activities such as sending text messages, sending pictures, videos, voice messages, and others. *LINE* was developed by a Japanese company called NHN Corporation. *LINE* was first released in June 2011 and initially it can only be used on iOS and Android systems. After success in both systems *LINE* entered the BlackBerry operating system. Then in 2012, *LINE* officially launched an application that can be used on Mac and Windows devices.

Viber is a messenger with the most features in it compared to other similar applications. *Viber* has high quality video calls with high bandwidth networks. Viber provides a voice call service that can make calls to any cellphone number, even though the number dialed does not use *Viber*. *Viber* offers group chat rooms like other messenger services. *Viber* also offers what is called public chat that allows users to communicate openly, according to the type of hobby group, fan club communication, or even large communities. *Viber* also offers two sound quality modes, namely normal and HD modes with little noise. the most unique feature is Viber Out, which allows Viber users to connect with people who are not in service. *Viber* is protected by *E2EE*, so it can ensure that communication is secure [6].

WhatsApp

When registering for the first time, the user transmits the public key, public Signed Pre Key (with its signature) and One-Time Pre Keys to the server. whatsapp server stores a public key that is associated with user identification and whatsapp server cannot access the user's private key. To be able to exchange messages via WhatsApp, someone who took the initiative to send a message first prepared a session. Some terms in preparing sessions, are as follows: public key types, session key types, client regristration.

Public Key Types

Curve 25519 is an advanced generation of elliptic curves adopted by the IETF (Internet Engineering Task Force). In the basic concept Curve25519 each user has 32 secret key bits and 32 public key bits. each pair of users has a 32 Bit shared key that is used to authenticate and encrypt messages between two users [7].

a. *Identity Key Pair* - A long-term Curve25519 key pair, generated at install time.
b. *Signed Pre Key* - A medium-term Curve25519 key pair, generated at install time, signed by the Identity Key, and rotated on a periodic timed basis.
c. *One-Time Pre Keys* - A queue of Curve25519 key pairs for one time use, generated at install time, and replenished as needed .

Session Key Types

- *Root Key* – A 32-byte value that is used to create Chain Keys .
- *Chain Key* – A 32-byte value that is used to create Message Keys .
- *Message Key* – An 80-byte value that is used to encrypt message contents. 32 bytes are used for an AES *(Advanced Encryption Standard)*-256 key, 32 bytes for a HMAC-SHA256 key, and 16 bytes for an IV.

Initiating Session Setup

To communicate with another WhatsApp user, a WhatsApp client first needs to establish an encrypted session. Once the session is established, clients do not need to rebuild a new session with each other until the existing session state is lost through an external event such as an app reinstall or device change [8]. To establish a session:

a. The initiating client ("initiator") requests *the public Identity Key*, public *Signed Pre Key*, and a single public *One-Time Pre Key* for the recipient.
b. The server returns the requested *public key* values. A *One-Time Pre Key* is only used once, so it is removed from server storage after being requested.
c. The initiator saves the recipient's *Identity Key* as Irecipient, the *Signed Pre Key* as $S_{recipient}$, and the *One-Time Pre Key* as $O_{recipient}$.
d. The initiator generates an Curve25519 key pair, $E_{initiator}$.
e. The initiator loads its *own Identity Key* as $I_{initiator}$.
f. The initiator calculates a master secret as *master_secret* = ECDH(Iinitiator, Srecipient) ‖ ECDH($E_{initiator}$, $I_{recipient}$) ‖ ECDH($E_{initiator}$, $S_{recipient}$) ‖ ECDH($E_{initiator}$, $O_{recipient}$) . If there is no *One Time Pre Key*, the final ECDH is omitted.
g. The initiator uses HKDF to create a *Root Key* and *Chain Keys* from the *master_secret* .

Message exchange on WhatsApp between users is protected by *Message Key* by using *AES256 encryption* in *CBC (Chiper Block Chaining)* mode and using *HMAC-SHA256* as data integrity. The Message Key is obtained from the *Chain key* which is directed to be forwarded to each message sent [8]. calculation of the *Message Key* and the *Chain Key*, it is calculated as :

$$M \quad g \quad K \quad y = \text{HMAC} - \text{SHA256}(\text{Chain Key},$$

0x0.1)

The Chain Key is then updated as :

$$h \quad i \quad K \quad y = \text{HMAC} - \text{SHA256}(\text{Chain Key},$$

0x0.2)

The *message key* for each message is always different because it is ephemeral and cannot be used again. *Public key* Curve25519 is only used for sending messages at that time. *Chain Key* is only used to send messages from one user so the message key cannot be used again. then after the response is received, a new *Chain Key* and *Root Key* can be calculated with the following steps:

$$\text{ephemeral\_secret} = \text{ECDH}(\text{Ephemeral}_{\quad n} \quad , \text{Ephemeral}_{recipient})$$

$$\text{Chain Key}, \text{Root Key} = \text{HKDF}(\quad K \quad y, \quad h \quad \_ \quad)$$

Skype

Skype Private Conversation is an optional mode of messaging when all audio calls, texts, or media files like images, audio, or video are end-to-end encrypted. Skype Private Conversation end-to-end encryption uses Signal Protocol by Signal. Signal Protocol Library is the underlying encryption provider used by devices. The Signal Protocol Library uses the X3DH algorithm [9].

X3DH *(Extended Triple Di ffi e-Hellman)* is a cryptographic algorithm used for data encryption. X3DH establishes a shared secret key between two parties who mutually authenticate each other based on *public keys*. X3DH provides forward secrecy and cryptographic deniability. X3DH is designed for asynchronous settings where one user is offline but has published some information to a server. Another receiver wants to use that information to send encrypted data to sender, and also establish a shared *secret key* for future communication [10].

A device is identified by a globally unique *deviceId*. All the following key-pairs are generated using Curve25519. The *private key* component never leaves the device and is protected in device-local storage. In this document, public keys are uppercase, while private keys are lowercase [9].

Identity Key (IK, ik) is a long-term key-pair associated with the device. It is generated after a user logs in. The Identity Key is unique per device. After the user has logged in, an Identity Key and Signed Pre-Key are generated.

Signed Pre-Key (SPK, spk) is a medium-term key-pair that is rotated periodically then periodically renewed every two weeks. *Pre-Key* (PK, pk) is a one-time key-pair used to set up conversations between devices. It is generated on a per-needed basis.

A Private Conversation is established between a pair of devices designated by users participating in the conversation. Conversation setup for Private Conversation has three steps :

a. Invite, User A sends a message from one device to invite user B to a Private Conversation. The invite provides the following information to B: IK(A), SPK(A), PK(A), deviceId(A). B receives the invite on all their devices.
b. Accept, User B accepts the invite and their device sends an accept message to A containing the following information: IK(B), SPK(B), PK(B), deviceId(B).
c. Confirm, This is an automatic response from A notifying all devices of both A and B of negotiated endpoints. The Signal Protocol Library to establish the end-to-end encrypted session between two Skype users. After the session is established, users A and B can view the fingerprint of the session and confirm the session is secure.

Before sending a message, both sender and receiver must complete a handshake and setup Protocol session signal. step of the process in encrypting the message :

a. Sender generates a per-message random symmetric encryption key Ke, initialization vector IV, and authentication key Ka.
b. Sender encrypts plaintext of the message to be sent using AES-256 in CBC mode with key Ke and IV to obtain ciphertext CT.

c. Sender computes authentication tag T using HMACSHA256 over IV and CT with key Ka.
d. Sender computes SHA-256 hash over IV, CT and T to obtain hash value H.
e. Sender then encrypts tuple Ke, Ka, H, IV, T to the intended recipient device using the Signal Protocol session established with the device to obtain a dictionary D = {<device>: <encrypted data>}.
f. Sender then sends CT and dictionary D to the intended recipient.
g. The server sends the messages to all recipient devices. Devices not listed in the dictionary will drop the messages and not decrypt the contents. A dictionary is used to allow the extension to multiple devices in the future.

Facebook Messenger

Secret Conversations is a specialised conversation mode in Messenger. Messages in Secret Conversations are E2EE between the sender and the recipient using the Signal Protocol and open source implementations.

Keys

Each device manages various cryptographic keys. All keys are generated or derived on-device. Private keys are never sent to Facebook. All public key operations use Curve25519. Each device uses the following public-secret key pairs [11]:

a. The *Identity Key keypair* (IK_{pk},IK_{sk}). This is a long-term keypair which is generated the first time Messenger runs.
d. The *Signed Pre-Key keypair* (SPK_{pk},SPK_{sk}). This is a medium-term keypair which is rotated periodically. It is signed by IK_{sk}.
e. The *One-Time Pre-Key* keypairs ($OTPK_{pk}$,$OTPK_{sk}$). These keypairs are generated in batches by clients. They facilitate asynchronous conversation initiation.

Pairwise Message Exchange

Each pairwise message is encrypted with AES-CBC and authenticated using HMAC-SHA256. The unique MK is derived from the current CK and RK. The first value as follow:

CK = RK
MK = HKDF(CK)

the sender generates a fresh ephemeral key pair (K^{n} , K^{n}). The recipient calculates the

Pairwise Channel Initiation Pairwise Channel Initiation

Each pairwise channel consists of two devices: one Initiator device and one Responder device (I and R respectively). Let HKDF be a secure hash-based key derivation function, and ECDH indicate the elliptic curve DiffieHellman function applied to a secret and public key. To create a new pairwise channel:

a. The Initiator obtains from Facebook I_{I}, I' $I_{I}$and I I' for an one-time pre-key keypair generated by the Responder device.
b. Facebook deletes $I_{I}$and The Initiator generates a fresh ephemeral keypair (I_{I} I I', I').
c. Using the RK the Initiator can calculate the first CK and MK (as described next) and use those to start sending messages. The Initiator now computes the first root key RK as follows:

IR IR
= (K , K_p), = (K , K_p)
= (K , K_p , = (K , K_p
RK = HKDF(a‖b‖c‖d) I R) IR)

p
current MK value using EKsender pk and can decrypt
the message, then generates a fresh ephemeral keypair
i , K i) and derives new keys RK',
(K_p
CK' and MK' for use with the next response by updating the previous symmetric key values as follows:

f. The *Ephemeral Key* keypairs (EK_{pk},EK_{sk}). A new ephemeral keypair is generated for each round of communication within a secret conversation and is subsequently discarded.

cryptographic channel the participating devices derive symmetric session keys [11]. These are:

a. The *Root Key* (RK) is a 256-bit key which is used to derive *Chain Keys* in the Signal Protocol ratche ts.
b. *Chain Keys* (CK) are each 256-bit values which are used to derive Message Keys.
c. *Message Keys* (MK) are each 640-bit values which consist of 256 bits for an AES-256 key, 256 bits for an HMAC·SHA256 key, and 128 bits for an Initialization Vector (IV) for AES-CBC encryption.

Messaging traffic between LINE clients and servers is protected with forward-secure

Key Generation and Registration

In order to be able to send encrypted messages, each *LINE* client application generates a Letter Sealing *ECDH* key pair. After generating the device *key pair*, each *LINE* client registers its *public key* with LINE's messaging server. The server associates the key with the currently authenticated user and sends back a unique key ID to the client. Each key ID is bound to a specific user and represents the current version of that user's *public key*.

Client-to-Client Key Exchange

In order to be able to exchange encrypted messages, the client must retrieve the recipient's current public key. Next, the client passes its own privacy key and the recipient's public key to the ECDH algorithm to generate shared secrets. The recipient generates the same shared secret using their own private key and the sender's public key, as shown below[12].

Shared Secret

$$= 25519(K\ y_{U\ 1P\ i}, K\ y_{U\ 2\ P\ i})$$

$$= 25519(K\ y_{U\ 2P\ i}, K\ y_{U\ 1}$$

$$M_{n} = (K\ y_{n\ yp}, M_{p\ in}[0:15] \oplus M_{p\ n}[16:31]$$

Finally, the following data is included in the message sent to the recipient [12]:

version	Content type	salt	C	MAC	Sender key ID	Recipient key ID

The version and content type fields serve to identity the Letter Sealing version used to create the message. Recipients use the sender key ID to retrieve the public key used to encrypt the message. The recipient key ID value helps verify that the message can be decrypted using the current local private key. Once the recipient determines that they

Pairwise Session keys When starting a pairwise

LINE

$$_{i}, K_{i}))$$

$$MK' = K (K')$$

encryption,
and both text messages and media streams in VoIP calls are end-to-end encrypted. LINE messages are locally encrypted on each client device before being sent to LINE's messaging server, and can only be decrypted by their intended recipient. Letter Sealing is applied only to message payloads, and message metadata (sender ID, recipient ID, and so on) is not encrypted.

Cryptographic algorithms applied to LINE such as
the Key exchange algorithm using ECDH over Curve25519, Message encryption algorithm using AES-256 in CBC mode and Message hash function using SHA-256 [12].

$$_{P\ i})$$

Message Encryption

LINE encrypts each message with a unique encryption key and IV. The encryption key and IV are derived from the shared secret calculated in Client-to-Client Key Exchange and a randomly generated 8-byte salt as follows [12]:

$$K\ y_{n\ p} = 256(h\ ||\ ||"K\ y")$$

$$V_p = 256(h\ ||\ ||"\ V")$$

$$V_{n\ p} = V_p\ [0:15] \oplus V_p\ [16:31]$$

The generated key and IV are used to encrypt the message payload M using 256-bit AES in CBC block mode.

$$= (K\ y_{n\ yp}, V_{n\ yp}, M)$$

Next, LINE calculates a message authentication code (MAC) of the ciphertext C, as follows [12]:

$$M_{p\ in} = 256( )$$

can decrypt a message, they derive the shared secret, symmetric encryption key, and IV. Next, LINE calculates the MAC of the received ciphertext, and compares it with the MAC value included in the message. If they match, the contents of the message is decrypted and displayed. Otherwise, the message is discarded.

VIBER

To identify a viber account using a long-term Key- a ID 256-bit Curve25519 key pair used. Pre Keys a set of medium-term Curve25519 key pairs used to establish one-on-one secure sessions between devices

Secure Session Setup

Exchange messages safely need to create a conversation session, the session only needs to be made once and can be used to send an unlimited number of messages in any direction.

To create a session with a different account, User A wants to create a session with User B, then User A requests Query telephone number B to the Viber server. then the server responds with the Public Key and Pre Key from User B. User B then generates two 256-bit devices, Curve-25519 key-pairs as its own handshake and ratchet keys, and derives a Root Key [13]. as follows:

*DH indicates the use of Elliptic-Curve DiffieHellman key-exchange algorithm.
*HS indicates Handshake Key

RootKey = SHA256 (DH(IDUserB,HSUserA) || DH(HSUserB, IDUserA) || DH(HSUserB, HSUserA))

The RootKey is then used to derive a session key using:

TempKey = HMAC_SHA256(RootKey, DH (RatchetUserB, RatchetUserA))

New RootKey = HMAC_SHA256(TempKey, "root")
SessionKey = HMAC_SHA256(TempKey, "mesg")

Exchange Messages

Encryption on Viber uses *E2EE*. to exchange messages the sender must encrypt the message for each session with each receiving device. then the 128-bit ephemeral one-time symmetric key is the body using *Salsa20* encryption algorithm. then this ephemeral message key is encrypted using the recipient's session key. the sending device sends a message that contains a ciphertext that has been encrypted and a set of ephemeral keys. then the server divides the message and sends the message part that is relevant to the recipient [13].

Salsa20 is a stream cipher based cryptographic algorithm. This cryptographic algorithm was developed by Daniel J. Bernstein in 2005. Salsa20 cryptographic algorithms generate flows key from input keys, then perform XOR operations between streams key with text or plain text ciphers. The Salsa20 algorithm receives input in the form of 32 bytes keys, 8 bytes nonce (maximum), 8 byte block counters (maximum). After the key flow development process is carried out then an XOR operation is carried out between the key flow and plain text. The resulting key stream is 64 bytes [14].

Ratcheting is the process by which both devices take turns entering the session key and will produce a new Ratchet key pair. Calculate of Ratcheting process as follow :

TempKey = HMAC_SHA256(RootKey, DH (RatchetUserB, RatchetUserA))

New RootKey = HMAC_SHA256(TempKey, "root")
SessionKey = HMAC_SHA256(TempKey, "mesg")

3. Result

literature review results in Securing Text Messages Using E2EE (End-To-End Encryption) On Instant Messaging Applications can be seen in the comparison table below:

Table 1. Comparison in the use of E2EE at Instant Messaging Applications

	Category	Instant Messaging Applications				
		WhatsApp	Skype	FB Messe	LINE	Viber
	Key Exchange Encryption	Curve 25519	Curve 25519	Curve 25519	Curve 25519	Curve 25519
2	Message encryption	AES *(Advanced Encryption Standard)*-256 key	AES *(Advanced Encryption Standard)*-256 key	AES *(Advanced Encryption Standard)*-256 key	AES *(Advanced Encryption Standard)*-256 key	Salsa20
3	Mode CBC (Chiper Block Chaining)	*HMAC-SHA256*	HMAC-SHA256	HMAC-SHA256	SHA-256	HMAC-SHA256
4	cryptographic algorithm	AES 256	X3DH *(Extended Triple Diffie-Hellman)*	AES 256	ECDH Algorithm	*Salsa20* Encryption Algorithm
5	Encryption Protocol	Signal protocol	Signal protocol	Signal protocol		
6	the place of the encryption process	Device	Device	Device	Device	Device
7	cryptographic type	symmetric	symmetric	symmetric	symmetric	symmetric

With Ratchetthis_device being the private part of the newly derived key-pair. Alongside each message, the public part of the Ratchetthis_device is also sent. The recipient runs DH with its last private ratchet together with the sender's public ratchet.

4. Conclusion

The conclusion of the research review literature is that the messaging application encrypts using E2EE (*End-to-end encryption*) with different calculation methods and encryption algorithms. from instant messaging applications that are reviewed all encrypted messages are encrypted on the device sender, while the server only accepts encrypted messages. so that only the sender and receiver know the contents of the message.before sending a message, what is done first is to make a secure session first.

Reference

E. Moran, "END TO END ENCRYPTION AN ANSWER TO SECURITY CONCERNS IN PRIVATE SECTOR," 2017.

nU. Santria and N. Arsoetar, "Penggunaan Enkripsi End-to-End dalam Pengamanan Pesan dan Video Call pada Whatsapp," pp. 137–142, 2017.

S. R. De Vries and S. Aissaoui, "WhatsApp End-to-End Encryption : Are Our Messages Private ? Research project by students of the SnE masters programme," pp. 1–19, 2019.

P. T.-G. M. Seufert, T. Hoβfeld, A. Schwind, V. Burger, "Group-based communication in WhatsApp," IFIP Netw. Conf. (IFIP Networking) Work., pp. 536–541, 2016.

D. M. Manalu, "Komunikasi Antarpribadi Melalui Media Sosial (Skype) Pada Mahasiswa Universitas Riau," Fisip, vol. 1, no. 2, pp. 1–13, 2014.

T. Sutikno, L. Handayani, D. Stiawan, M. A. Riyadi, and I. M. I. Subroto, "WhatsApp, viber and telegram: Which is the best for instant messaging?," Int. J. Electr. Comput. Eng., vol. 6, no. 3, pp. 909–914, 2016.

D. J. Bernstein, "Curve25519: New Diffie-Hellman speed records," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 3958 LNCS, pp. 207–228, 2006. WhatsApp, "WhatsApp Encryption Overview," pp. 1–12, 2017. Skype, "Skype Private Conversation Technical white paper," pp. 0–6, 2018.

M. Marlinspike and T. Perrin, "The X3DH Key Agreement Protocol," Signal, p. 11, 2016.

T. Whitepaper, "Messenger Secret Conversations," 2016.

T. Whitepaper, "LINE Encryption Overview," 2016.

S. S. Setup, E. Messages, E. Calls, F. Sharing, S. Groups, and S. D. Registration, "Viber Encryption Overview."

P. L. T. Irawan, "Implementasi Teknik Kriptografi Stream Cipher Salsa20 Untuk Pengamanan Basis Data," Smatika J., vol. 5, no. 02, pp. 88–92, 2015.