International Journal of Science and Research Archive
eISSN: 2582-8185
Cross Ref DOI: 10.30574/ijsra
Journal homepage: https://ijsra.net/

(RESEARCH ARTICLE)

Check for updates

Balancing privacy protecting and auditability: A signal protocol based E2EE communication scheme for group chatting

Xingjian Tian *

Wenzhou Polytechnic, Wenzhou, Zhejiang, China.

International Journal of Science and Research Archive, 2025, 14(01), 093-099

Publication history: Received on 26 November 2024; revised on 02 January 2025; accepted on 04 January 2025

Article DOI: https://doi.org/10.30574/ijsra.2025.14.1.0018

Abstract

With the development of the Internet, instant messaging applications have ascended to a crucial position in our daily lives. As users increasingly attach importance to privacy issues, end-to-end encryption(E2EE) has become an indispensable requirement for messaging services. However, E2EE communication software face two major challenges. One is that completely being unregulated is rather perilous, as criminals may take advantage of it to cover up crimes and endanger society. Secondly, ordinary E2EE software only works between two parties, and there has been relatively little research on group chatting scenarios. Against these backdrops, this paper devises practical solutions to the problem of allowing audit access by authorized parties while safeguarding privacy in the context of group chatting. By leveraging the Signal protocol based on the open Noise protocol framework [1], we ingeniously transform the problem into multiple two-way chatting channels, thereby ensuring E2EE security and facilitating efficient group communication. Additionally, to address the auditing challenge, we delicately design the model with multiple cryptological techniques and safeguard the security of ordinary users while expediting the auditing process. This solution strikes a balance between privacy concerns and regulatory demands, and offers practical and innovative model design to secure group chatting, enhancing the usability and security of instant messaging applications.

Keywords: Noise protocol; Signal Protocol; End-to-End Security; Group Chatting; Privacy Protect

1. Introduction

Instant messaging applications has redefined the way we communicate and platforms such as WhatsApp, WeChat, and Telegram have amassed billions of users worldwide. As the volume and sensitivity of the information exchanged continue to grow, the demand for privacy-protected E2EE has surged, with users seeking to safeguard their conversations from prying eyes, whether they be hackers, advertisers, or even the service providers themselves.

However, this pursuit of privacy has led to two significant challenges. Firstly, the existing E2EE mechanisms, designed primarily for one-on-one conversations, struggle to provide the same level of security and privacy when scaled up to group settings. Secondly, the very nature of E2EE that provides anonymity and privacy to users has also created a conundrum for law enforcement. With the rise of "anonymous" and "self-destructing" messaging features, criminals have found new avenues to conduct illegal activities without leaving a trace. In this regard, the development of auditing mechanisms that can penetrate the veil of E2EE without compromising the privacy of law-abiding citizens has become a pressing issue.

This paper aims to address these two intertwined challenges by proposing a novel and balanced solution for group chatting based on the Signal protocol. Our approach involves transforming the group chatting scenario into multiple two-party chatting channels, and introduce a mechanism where messages are encrypted with the public key of the audit authority and stored in a way that only the authorized audit entity can access them. This strikes a balance between

* Corresponding author: Xingjian Tian

privacy protection and regulatory compliance, a feature that sets our solution apart from existing ones. Compared to previous attempts, our solution offers enhanced security, better privacy preservation, and a more practical approach to auditing, thus providing a more comprehensive and reliable solution for secure group chatting in the digital age.

2. Preliminaries and Related Work

2.1. Noise Protocol

The Noise itself is not a specific protocol, it is a protocol framework. It takes a basic set of encryption operations and combine them, so that a variety of security attributes and corresponding secure channels protocols can be provided, which makes It modular [2]. For example, an AEAD cipher, a hash function and a Diffie-Hellman scheme can form a specific Noise protocol and create secure channels for transmitting. In Noise protocol, each party has a long-term static key pair and/or an ephemeral key pair and the communication begins with a handshake that follows a particular pattern, which is a sequence of tokens that specifies the DH(Diffie-Hellman) operations performed when sending or receiving the message. The patterns are named after the state of these long-term static keys: NK, IK, XN, etc. The first letter indicates the status of the initiator's long-term static key, and the second letter indicates the status of the responder's, as illustrated in Table. 1.

Table 1 Abbreviations

Letter	Meaning
N	No long-term static key is present
K	The long-term static key is Known to the other party before the handshake
X	The long-term static key is transmitted (Xmitted) to the other party
I	The long-term static key (for the initiator) is Immediately transmitted to the responder, despite absent/reduced identity hiding

All the standard handshake patterns require an exchange of ephemeral keys: this is done to provide forward secrecy, so that a later compromise of long-term static keys would not reveal the plaintext contents of previous communications [3]. Noise has this property in common with TLS 1.3, which also requires the exchange of ephemeral keys, an upgrade from previous versions of TLS where it was optional. Some Noise protocols also offer identity hiding properties, depending on when the static keys are transmitted [4].

There is an example of XK pattern with three passes, as is shown in Figure 1. $g^A$and g^B denote the long-term public DH shares of parties A and B, g^a and g^b denote their ephemeral shares, and $enc_k(ad, m)$ is an AEAD encryption. Once a Noise handshake is completed, the result is an AEAD-protected transport channel that provides various forms of confidentiality, integrity, and authenticity guarantees. that provides various forms of confidentiality, integrity, and authenticity guarantees.

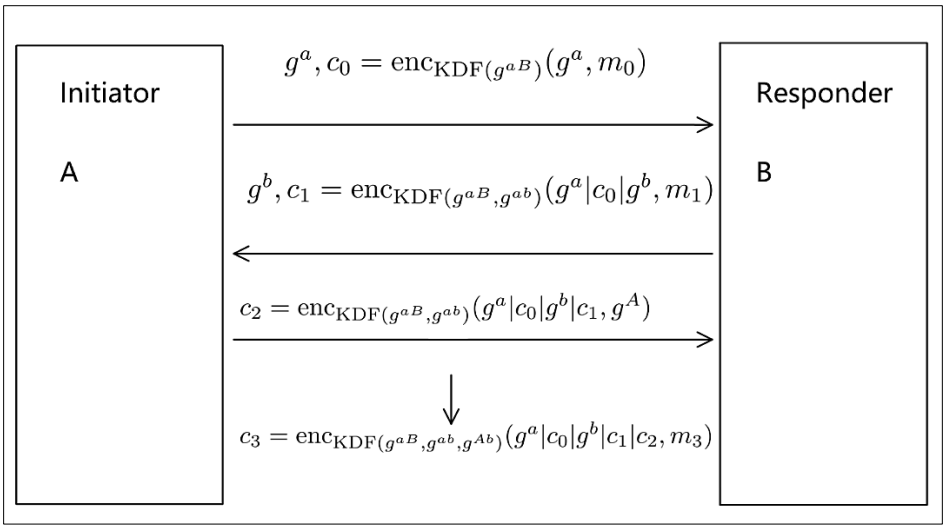

Figure 1 Noise's XK Pattern

2.2. Double-Ratchet Algorithm

The Double-Ratchet algorithm is the core of Signal protocol, which is used by two parties to exchange encrypted messages based on a shared secret key. The parties derive new keys for every message so that earlier keys cannot be calculated from later ones. The parties also send DH public values attached to their messages. The results of DH calculations are mixed into the derived keys so that later keys cannot be calculated from earlier ones. These properties give protection to earlier or later encrypted messages in case of a compromise of a party's keys. Consequently, a complete Double-Ratchet algorithm consists of a DH ratchet and a symmetric-key ratchet algorithm [5]. When a message is sent or received, a symmetric-key ratchet step is applied to the sending or receiving chain to derive the message key. A typical Double-Ratchet process is as Figure 2 shows.

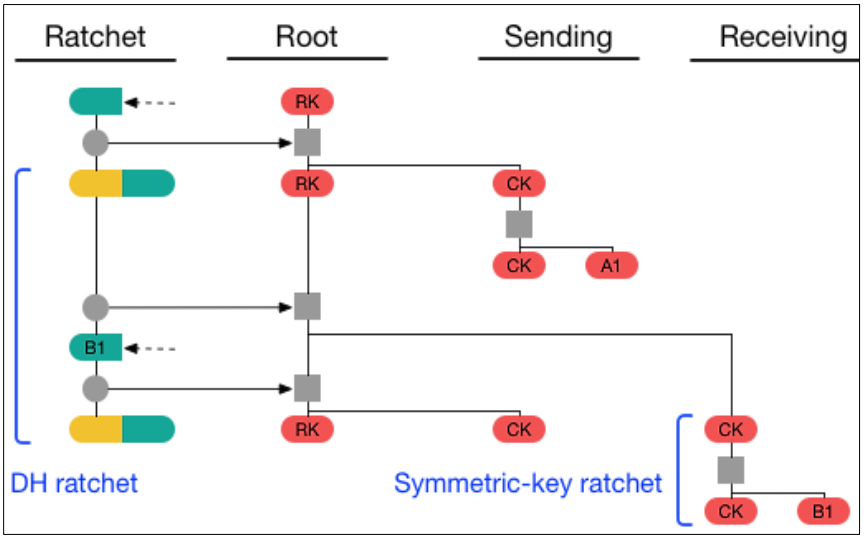

Figure 2 Double-Ratchet Process

2.3. Signal Protocol

Signal is one of the most popular messaging services that achieves E2EE security level for instant messaging, born of the idea of Double-Ratched algorithm integrated with X3DH algorithm [6]. The paper adopts a role-agnostic naming scheme, describing stages as "$-ir$" if they are used for the initiator to send to the responder, and as "$-ri$" if they are used for the responder to send to the initiator. This maintains the invariant that stages with the same name generate the same key(s). At a given party, the paper counts the number of symmetric updates in the x −th symmetric chain in a variable y; thus, we can refer to the y-th update in the x-th symmetric chain as stage $[sym - ri{:}\, x, y]$ or $[sym - ir{:}\, x, y]$. The Signal protocol can be separated into following 4 phases:

- Registration: For each user, some DH keys including a long-term identity key ik, a medium-term signed pre-key $prek$ and multiple one-time pre-keys $eprek$ will be generated. And the corresponding public keys will be stored in a key distribution server, together with a signature on $prek$ using ik.
- Session set up: In this phase, public keys are exchanged and used to initialize secrets in the session memory. And after this set up phase is done, root keys for the communicating clients' all KDF chains are generated as well.
- Symmetric-Ratchet communication: Following the second phase, this phase shows the sending and receiving chains of both clients. From my understanding, it is like the SYN and ACK number in the TCP, but with the form as encryption/decryption keys. Every time Alice sends or receive a new message (but in the same asymmetric ratchet phase), the sending or receiving key chains update using a KDF.
- DH-Ratchet updating: The last phase is the dh-ratchet update. In this phase, when a client tries to send a new message after receiving messages from the other end, it also starts the DH-ratchet updating, sending new DH public keys and derive new shared secrets.

After the four phases, the Double-Ratchet stage trees can be shown as Figure 3.

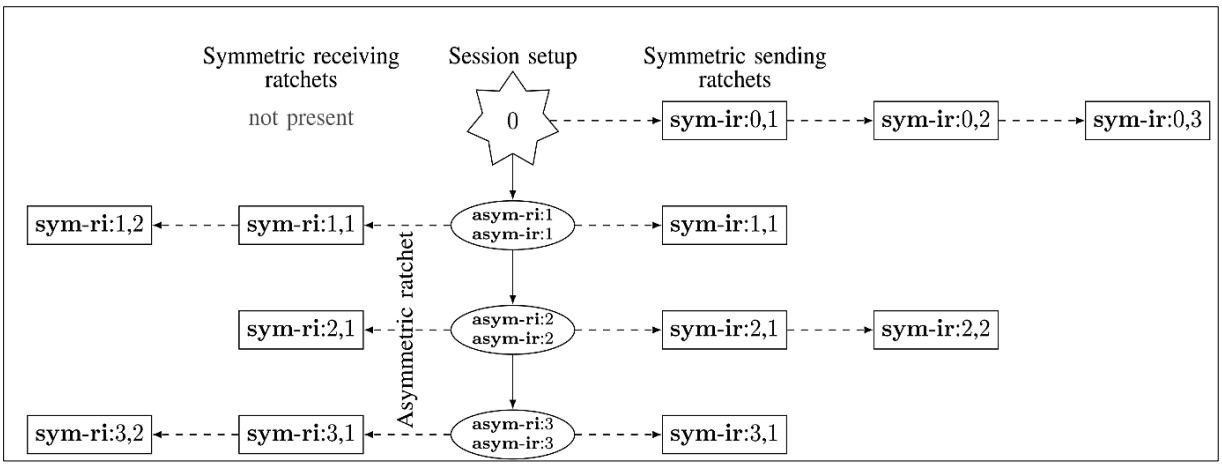

Figure 3 Double-Ratchet Communication Tree

By using such delicate and beautiful algorithm, Signal protocol can successfully create E2EE security channels for transmitting [7]. However, it is limited in sessions involved with two parties. This has left a gap in the security infrastructure of modern messaging platforms, as group chats often involve sensitive discussions that require equal, if not greater, protection [8].

3. Proposal

3.1. E2EE Group Chatting

The first challenge is that Signal protocol only applies for two parties. To address this issue, in our proposed group chatting model, we introduce an encrypted message queue [9], where all the messages come in and out following the order of timestamp. A client can deliver his/her messages to the server, then the server will check if the queue is empty. If not, deliver the top message of the queue to all clients except the one that originally sent the message [10]. In this way, multiple clients can transmit through different security channels with each other and all these channels are independently secure by applying Double-Ratchet algorithm, making the group chatting becoming multiple end-to-end chatting.

3.2. Authorized Audit

The second challenge of E2EE group chatting is to provide authorized auditing functionality for law enforcement, or else it would be harmful when crimes and unlawful acts going on in E2EE environment [11],[12]. The security expectation is that only when audit authority requires will the messages be encrypted and audited. Otherwise, the group chatting stays E2EE security level. To reach this goal, server has to a back-up or store the encrypted messages, while having no access to decrypt the messages [13]. We implement it by following steps:

- Firstly, the audit authority creates list of public-private key pairs, using ECC or other algorithms. Then store the list of public keys to the server. The server randomly chooses a public key $PubK_i$ for group i's chatting room, and give it out for all the clients (the server will also send the public key to the ones that join later). Meanwhile, a public key infrastructure (PKI) system is brought in for the server to verify the request and the signature from any other parties trying to have audit access.
- Then for each client, when he/she sends messages, not only the Double-Ratchet encrypted messages but also a copy of the plain messages encrypted by the public-key owned by audit authority, will be sent through the secure channal. Some other basic information, user identity-group id pairs for example, can be encrypted and stored in the server too, while the server has no access to the private key of it.
- Finally, if the audit authority decides to audit group i or user j, it will send a request with signature to the server. Then server verifies the request and signature and sent corresponding data to the audit (messages of group i or basic information for finding user j). And then the audit authority can use the corresponding private key SKi to decrypt and review the data and perform operations.

3.3. Model Design

After analyzing the challenges, this paper gives the model of auditable E2EE group chatting, the overall design is depicted in Figure 4. The model mainly consists of three components: Clients, Server, and Audit. Now let us elaborate in following sections.

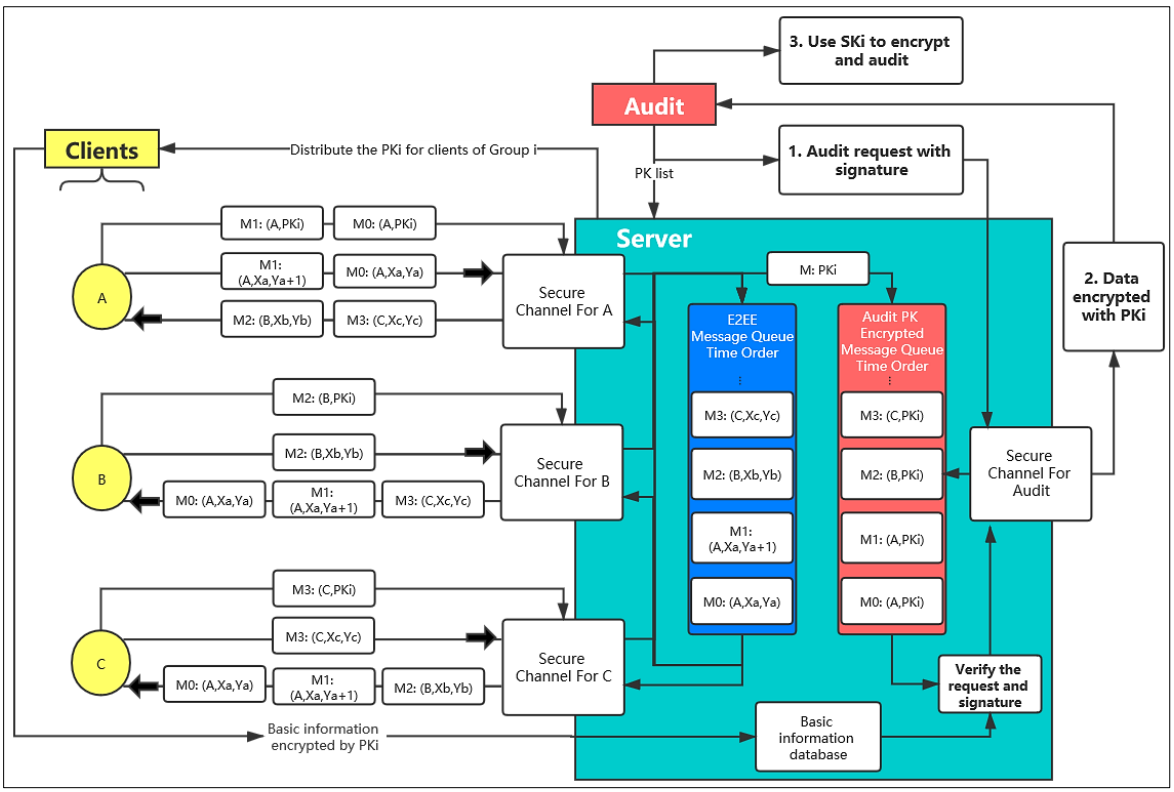

Figure 4 Proposed Auditable E2EE Group Chatting Model

- Clients: Assume that there are 3 clients: A, B, C, they send some messages as Mn: $(Client, X', Y')$, n means nothing but the order of timestamp. X', Y' denote the stage number and message number, just like in the last section of introducing Signal protocol. And for each client, there are E2EE security level channels for receiving and sending messages with the server, while only authentic parties can have the audit access, bypassing the PKI system brought within the server.
- Server: The server is the core of the model, responsible for handling client messages and audit requests, consisting of three major modules:
    - Message Queues: The server contains three message queues corresponding to clients A, B, and C respectively. These queues are arranged according to the E2E message queue time order. When a client sends a message, it enters the corresponding queue. For instance, messages sent by client A enter the queue for A;

  - PKI system: The server maintains a PKI system for verifying the identities of clients and the audit party. The server distributes the public key list to clients and uses public keys to encrypt and verify messages;
  - Basic Information Database: The server also has a basic information database. The basic information of clients is stored here and encrypted by PKI to ensure security.
- Audit: The audit component is responsible for monitoring and auditing group chat messages to ensure compliance, containing above two parts:
  - Audit Request Process: The audit party first sends an audit request with a signature to the server. Then the server verifies the request and signature to ensure the legality. The auditor uses its private key SKi to decrypt the data obtained from the server.
  - Message Encryption and Storage: There is a special encrypted message queue for audit in the server. After a client sends a message, besides the message sent through the E2EE secure channel, a copy of the message encrypted with the audit party's public key is also sent to the audit message queue in the server.

The overall process of our proposed E2EE group chatting model can be described as below:

- Clients send messages to the server, and the messages within a timestamp identifier enter into the corresponding message queues of the server.
- The server distributes messages to other clients according to requests, ensuring secure and timestamp ordered message transmission.
- When the audit party needs to audit, it sends an audit request with a signature to the server.
- After verifying the request, the server sends the encrypted message data to the audit party.
- The audit party decrypts the data using its private key and conducts audit operations.

4. Result and Discussion

The group chatting model we proposed transforms the problem into multiple two-party chats. By doing so, the model can leverage the well-established security mechanisms of two-party chats and simplify the complexity of securing group conversations by breaking them down into more manageable, pairwise interactions. It remains E2EE security level naturally and evidently the server knows nothing about the message content. When client A receives message from B, it will use the keys of $A - B$ pair to decrypt the messages and update Double-Ratchet trees, meaning that the server has no access to the actual message content. This is a fundamental security feature because it prevents the server from being a point of vulnerability for message interception or leakage. And a client can enter a specific chatting room, after requiring $prek$s of the members of the chatting room, and start new sessions with each of them. This allows for seamless and secure integration into the group chatting environment.

To enable authorized auditing, the model requires that messages of group chatting be copied, encrypted, and stored on the server. However, these stored messages can only be decrypted by the audit authority. This ensures that the privacy of regular group chat participants is maintained while still allowing for regulatory compliance through auditing. Consequently, the server plays a crucial role in the auditing process. It must verify the request and signature of the audit authority before releasing any data. This verification is based on a PKI system. The PKI system provides a reliable mechanism for authenticating the audit authority's requests, preventing unauthorized access to the stored messages.

Moreover, even if an attacker were to breach the server, they would still need the private key SKi of the audit authority to decrypt the messages. Given the computational complexity of decrypting without the proper key, this provides a significant security barrier against unauthorized access to the chat messages. And if the audit authority does not request a review, the group chatting remains at the E2EE security level. This means that the auditing functionality does not interfere with the normal, secure operation when not in use, ensuring that the privacy and security of the group chatting participants are always maintained.

5. Conclusion

In this digital era, the demand for privacy in messaging has surged, yet challenges remain. This study addressed two key issues in group chatting: the lack of effective group E2EE and the need for authentic auditing. By leveraging the Signal protocol, we transformed group chats into multiple secure two-party channels, ensuring E2EE. Additionally, we designed an auditing mechanism with encrypted message copies for the audit authority. Our solution balances privacy and regulatory needs, outperforming existing approaches. It offers enhanced security and a practical auditing method, making group chats more secure and accountable.

Future work would focus on system implementation, optimizing performance and exploring broader applications. Overall, our research contributes to the advancement of secure group communication in the digital age

Compliance with ethical standards

Acknowledgments

This work was supported by the 2024 Wenzhou Philosophy and Social Sciences Planning Project [24WSK213YBM].

References

[1] Perrin, T. (2018). The noise protocol framework. noiseprotocol, Protocol Revision, 34.

[2] Ho, S., Protzenko, J., Bichhawat, A., & Bhargavan, K. (2022, May). Noise: A Library of Verified High-Performance Secure Channel Protocol Implementations. In 2022 IEEE Symposium on Security and Privacy (SP) (pp. 107-124). IEEE.

[3] Dowling, B., Rösler, P., & Schwenk, J. (2020). Flexible authenticated and confidential channel establishment (fACCE): Analyzing the noise protocol framework. In Public-Key Cryptography–PKC 2020: 23rd IACR International Conference on Practice and Theory of Public-Key Cryptography, Edinburgh, UK, May 4–7, 2020, Proceedings, Part I 23 (pp. 341-373). Springer International Publishing.

[4] Marlinspike, M., & Perrin, T. (2016). The x3dh key agreement protocol. Open Whisper Systems, 283(10).

[5] Bienstock, A., Fairoze, J., Garg, S., Mukherjee, P., & Raghuraman, S. (2022, August). A more complete analysis of the signal double ratchet algorithm. In Annual International Cryptology Conference (pp. 784-813). Cham: Springer Nature Switzerland.

[6] Cohn-Gordon, K., Cremers, C., Dowling, B., Garratt, L., & Stebila, D. (2020). A formal security analysis of the signal messaging protocol. Journal of Cryptology, 33, 1914-1983.

[7] Schliep, M., & Hopper, N. (2019, November). End-to-end secure mobile group messaging with conversation integrity and deniability. In Proceedings of the 18th ACM Workshop on Privacy in the Electronic Society (pp. 55-73).

[8] Cohn-Gordon, K., Cremers, C., Garratt, L., Millican, J., & Milner, K. (2018, October). On ends-to-ends encryption: Asynchronous group messaging with strong security guarantees. In Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security (pp. 1802-1819).

[9] Borisov, N., Goldberg, I., & Brewer, E. (2004, October). Off-the-record communication, or, why not to use PGP. In Proceedings of the 2004 ACM workshop on Privacy in the electronic society (pp. 77-84).

[10] Green, M. D., & Miers, I. (2015, May). Forward secure asynchronous messaging from puncturable encryption. In 2015 IEEE Symposium on Security and Privacy (pp. 305-320). IEEE.

[11] Rösler, P., Mainka, C., & Schwenk, J. (2018, April). More is less: On the end-to-end security of group chats in signal, whatsapp, and threema. In 2018 IEEE European Symposium on Security and Privacy (EuroS&P) (pp. 415-429). IEEE.

[12] Yang, Z., Liu, C., Liu, W., Zhang, D., & Luo, S. (2018). A new strong security model for stateful authenticated group key exchange. International Journal of Information Security, 17, 423-440.

[13] Bresson, E., Chevassut, O., & Pointcheval, D. (2001, November). Provably authenticated group Diffie-Hellman key exchange—the dynamic case. In International Conference on the Theory and Application of Cryptology and Information Security (pp. 290-309). Berlin, Heidelberg: Springer Berlin Heidelberg.