Threshold Moderation for End-to-End Encrypted Messaging

Max Christman, Saba Eskandarian

Department of Computer Science, University of North Carolina at Chapel Hill

Objectives

Our goal was to create a moderation system with the following properties:

- Messages are confidential unless they are reported
- Messages are deniable unless they are reported
- Only valid reports will be accepted
- **Many users can act as moderators, allowing diversity of opinion**

Introduction

End-to-end encrypted (E2EE) messaging is a vital tool for private communication, used by billions daily. In E2EE schemes, message content is visible only to the sender and their intended recipient, unless either party shares their view of the conversation. Most E2EE schemes have deniability, which means that the scheme does not cryptographically prove that a user has sent a message. Thus, if questioned, a user could plausibly deny having sent a message.

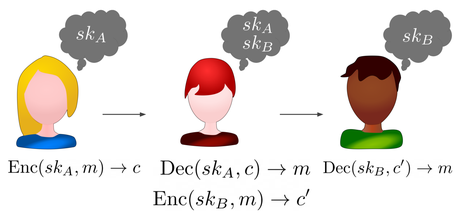

Figure 1: Encryption in transit

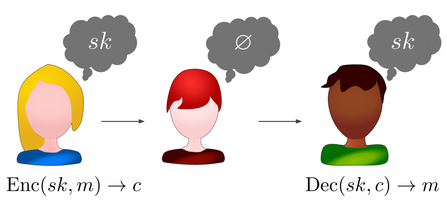

Figure 2: End-to-end encryption

Message Franking

While deniable E2EE messaging can provide excellent privacy, it can also allow users to send harmful messages with impunity. To that end, in 2016, Meta created a protocol called message franking [1], which aims for the following security properties:

- **Confidentiality:** No one can learn anything about the message, except the intended recipient
- **Authenticity:** No one should be able to send an unreportable message
- **Third-party deniability:** No one except the platform should be convinced that a given user sent a given message

Problems with Message Franking

Message franking places both the **burden** and the **authority** of moderation entirely on the platform. What if we modified message franking so that breaking message deniability would require a majority of moderators to consent? We can do this with threshold secret sharing.

Secret Sharing

To distribute trust across multiple moderators, we use a protocol called secret sharing. In additive secret sharing, the procedure is as follows:

1. The dealer chooses a secret which they represent with a number.
2. For n parties, the dealer generates $n-1$ random numbers and chooses the last number so that the sum is the secret number.
3. The dealer gives one number to each party.
4. If all parties talk to each other, they can add the numbers together to recover the secret number.

However, we use a modified scheme called threshold secret sharing [2] which lets a subset of the parties reconstruct the number. In our case, the secret number will be a secret key that the moderators use to generate commitments to messages.

Commitment Schemes

Message franking is able to break deniability while maintaining confidentiality through a tool called a commitment scheme. A commitment scheme takes a message m and a randomness r, and returns a commitment C:

$$\text{Commit}(m, r) \to C$$

A secure commitment will be **hiding**, meaning that C reveals nothing about m, as well as **binding**, meaning that it should be difficult to find distinct commitment/randomness pairs such that:

$$\text{Commit}(m, r) = \text{Commit}(m', r')$$

Threshold Moderation

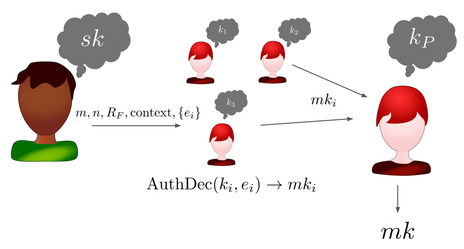

Figure 3: Reporting a message

Standard message franking involves the platform generating a commitment to the users' encrypted messages. Instead, we have multiple moderators who generate threshold secret-shared MAC keys, which they then use to generate commitments on encrypted messages.

Security

We argue that threshold moderation has the following security properties:

- **Confidentiality:** As we use a secure cipher and hiding commitment scheme
- **Authenticity:** As we use a binding commitment scheme
- **Threshold deniability:** By the security of threshold secret sharing

Future Work

In the future, we want to address the following situations:

- What if moderators are actively malicious?
- Can message metadata be kept private?

References

[1] Messenger secret conversations technical whitepaper. Technical report, Facebook, 2016. Accessed: 2023-03-22.

[2] Adi Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, November 1979.

Acknowledgements

I want to thank Saba Eskandarian for his many hours of guidance and encouragement. I also want to thank Roni Sengupta for helping me explain this work to a general computer science audience.

Contact Information

- **Web:** maxchristman.com/senior-thesis.pdf
- **Email:** max@cs.unc.edu

THE UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL