Poster: Committee Moderation on Encrypted Messaging Platforms

Alistair Pattison
University of Minnesota and Carleton College
pattisona@carleton.edu

Nicholas Hopper
University of Minnesota
hoppernj@umn.edu

I. Introduction

Over the past decade, the increased prevalence of mobile computing and a growing desire for privacy have lead to a surge in the use of encrypted messaging services like WhatsApp, Facebook Messenger, and Signal. The deniability, anonymity, and security provided by these services are crucial to their widespread adoption, but by construction, these properties make it impossible to hold users accountable for the messages they send. With no accountability, these platforms are ripe for abuse, the spread of misinformation [1], and election tampering [2]. WhatsApp group chats have been used to spread misinformation that has influenced the Brazilian presidential election [3] and even incited the murder of a woman in India [4]. With no way of identifying or verifying the senders of these messages, little can be done. Is there a middle ground that balances accountability with privacy?

Previous works have addressed this concern by allowing a moderator to verify the original sender of a message if a message is reported; if not reported, messages maintain all security guarantees. However, these works all concentrate the power of oversight and moderation to a single individual who is charged with determining if a message requires moderation. This is undesirable. Using primitives from threshold cryptography, this work extends the message-reporting protocol Hecate [5] to a setting in which consensus among a group of moderators is required to reveal and verify the identity of a message's sender.

II. Previous Works

An obvious road to accountability is to require users to sign their messages, but this completely destroys deniability, which is an important feature in many use cases. Tyagi, Grubbs, Len, *et al.* [6] addressed this problem by using a cryptographic primitive that allows signatures to be verified only by one person who is chosen at the time of signing. By making the designated verifier a trusted third party (for example, law enforcement or a school principal) and attaching to each message a zero-knowledge proof that the signature is valid, users can be confident that abusive messengers can be held accountable for the messages they send while still preserving deniability against everyone else. However, this protocol uses heavy crypto machinery and is quite expensive.

This work was supported by NSF grant 1814753.

More recent work by Issa, Alhaddad, and Varia [5] introduced a protocol called Hecate that provides the same security guarantees as Tyagi, Grubbs, Len, *et al.* [6] but is significantly cheaper in terms of the number of invocations of cryptographic primitives. It works as follows: In advance of sending any messages, users request "tokens" from a moderator containing (among other things) an encryption of the message-sender's identity,

$$x_1 \coloneqq \textsc{Enc}_{mod}(id_{src}), \tag{1}$$

and a single-use ephemeral key pair (pk_{eph}, sk_{eph}). The token also includes a signature σ_{mod} made with the moderators private key that binds the key pair to x_1.

When a user sends a message, he consumes a token and attaches the signature

$$\sigma_{src} = \textsc{Sign}_{sk_{eph}}(x_2) \quad \text{where} \quad x_2 \coloneqq x_1 \oplus H(m) \tag{2}$$

to the message along with the original moderator token. The signature binds x_1 to the sent message, and this metadata is carried with the message throughout its entire forwarding chain. If the message is ever reported, the moderator decrypts x_1 with her private key to obtain the original sender's identity. To everyone else, the token provides no information.

We direct readers to Issa, Alhaddad, and Varia [5] for a richer description of the protocol and its security.

III. Our Protocol

Our modified protocol retains the same general flow of token-issuing and message reporting from Hecate [5], but modifies the process by which x_1 is created and decrypted. We call our new protocol Cerberus (the name of the multi-headed dog that guards the river Styx) as a nod to the multiple moderators in the protocol and the greek name of the original Hecate protocol.

In Cerberus, there are n moderators, and k of them must cooperate to recover id_{src} from x_1. (These k and n values are tunable protocol parameters.) The token-creation process is described below using Elgamal threshold encryption and the FROST [7] signature algorithm, although different threshold schemes could be substituted. In the following pseudocode G is a (secure) group of order q with generator g. The moderators' public encryption key is pk_{mod}, and the private key is divided into shares $s_1, \ldots, s_n$ using a Shamir secret-sharing scheme [8]. Each moderator gets one share.

function CREATETOKEN(id_{src})

Generate $r \leftarrow_{\$} \mathbb{Z}_q$ and an an ephemeral keypair (pk_{eph}, sk_{eph}).

Compute $x_1 \coloneqq \langle g^r,\ id_{src} \oplus H(pk_{mod}^r) \rangle$.

Package x_1 into a token and send a signature request to each moderator along with the randomness r.

Each moderator verifies that x_1 is a valid encryption of id_{src} with the provided randomness and returns their signature share on the token.

Once sufficient responses are received, combine the signature shares into a valid Schnorr signature σ_{mod} and distribute the token containing x_1 and the ephemeral keypair.

This function is executed by one of the moderators who is designated as the coordinator who has no more power than the other moderators, except for the ability to block token creation. The token generated by this process is identical to one in the Hecate protocol [5], and the message is processed as is described in that paper.

To report a message, a user sends out requests to every moderator, each of whom then decides individually whether or not to respond with a decryption share d_i serving as a vote that the message should be acted upon. If more than k decryption shares are received, i.e., if more than k moderators believe that the message requires moderation, then one can recover the identity of the sender, id_{src}. If there are insufficient responses, id_{src} remains hidden. This process of "voting" adresses the question posed in [5] of how to handle reported messages that are not necessarily abusive or misinfirmative. A formal description is as follows:

function HANDLEREPORT(m, $T = \langle x_1, \ldots \rangle$)

A user sends a request to all n moderators containing the reported message m and its attatched token T containing the encrypted id, $x_1 = \langle c_1, c_2 \rangle$.

Each moderator verifies the token and—if she believes that the message requires intervention—responds with the decryption share $d_i \coloneqq c_1^{s_i}$.

If k decryption shares are received, they can be combined with appropriate Lagrange coefficients $\lambda_i \coloneqq \prod_{j \neq i} \frac{j}{j-i}$ to obtain

$$id_{src} = H\left(\prod_i d_i^{\lambda_i}\right) \oplus c_2. \tag{3}$$

IV. BENCHMARKS

Issa, Alhaddad, and Varia [5] includes an implementation and benchmark of the whole message cycle, so we focus on the modified portions of the protocol: token creation and message reporting. We implement these steps in Rust and run each party in a separate Linux container communicating over HTTP. More details of the benchmarks as well as the source code is available on github [9]. Cerberus is much slower to create tokens than Hecate (on the order of 30-50x), but this isn't a huge surprise: the benchmarks were run on slower hardware, and the distributed nature of this protocol involves many verification, serialization, and communication steps that are not required in Hecate. With that said, the operational costs of implementing a protocol such as this are still *well* within the budgetary constraints of a large company like Facebook.

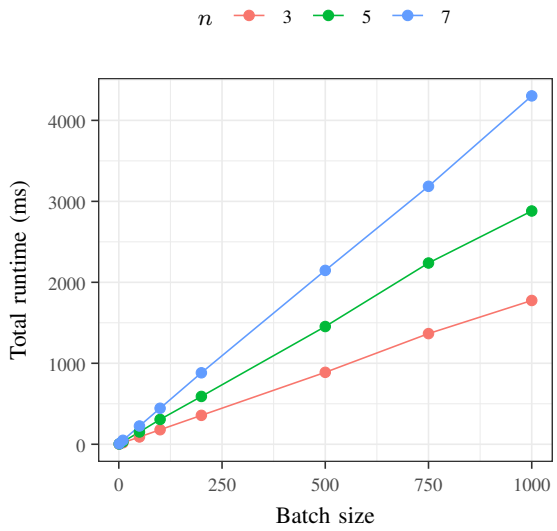

Fig. 1. Token creation times as a function of batch size and number of moderators.

REFERENCES

[1] E. Dwoskin and A. Gowen, "On whatsapp, fake news is fast and can be fatal," *Washington Post*, Jul. 23, 2018. [Online]. Available: https://www.washingtonpost.com/business/economy/on-whatsapp-fake-news-is-fast-and-can-be-fatal/2018/07/23/a2dd7112-8ebf-11e8-bcd5-9d911c784c38_story.html (visited on 04/04/2023).

[2] S. Woolley and P. Howard, "Computational propaganda worldwide: Executive summary," Oxford University, Working Paper 2017.11, 2017. [Online]. Available: https://demtech.oii.ox.ac.uk/wp-content/uploads/sites/12/2017/06/Casestudies-ExecutiveSummary.pdf.

[3] L. Belli, "Opinion: Whatsapp skewed brazilian election, showing social medias danger to democracy," *PBS*, Dec. 5, 2018. [Online]. Available: https://www.pbs.org/newshour/science/whatsapp-skewed-brazilian-election-showing-social-medias-danger-to-democracy (visited on 04/04/2023).

[4] V. Goel, S. Raj, and P. Ravichandran, "How whatsapp leads mobs to murder in india," *The New York Times*, Jul. 18, 2018. [Online]. Available: https://nyti.ms/383uZ59 (visited on 04/04/2023).

[5] R. Issa, N. Alhaddad, and M. Varia, "Hecate: Abuse reporting in secure messengers with sealed sender," in *31st USENIX Security Symposium (USENIX Security '22)*, USENIX Association, 2022. [Online]. Available: https://www.usenix.org/conference/usenixsecurity22/presentation/issa.

[6] N. Tyagi, P. Grubbs, J. Len, I. Miers, and T. Ristenpart, *Asymmetric message franking: Content moderation for metadata-private end-to-end encryption*, Cryptology ePrint Archive, Paper 2019/565, 2019. [Online]. Available: https://eprint.iacr.org/2019/565.

[7] C. Komlo and I. Goldberg, *Frost: Flexible round-optimized schnorr threshold signatures*, Cryptology ePrint Archive, Paper 2020/852, 2020. [Online]. Available: https://eprint.iacr.org/2020/852.

[8] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, pp. 612–613, 1979, ISSN: 0001-0782. DOI: 10.1145/359168.359176. [Online]. Available: https://doi.org/10.1145/359168.359176.

[9] A. Pattison, *Cerberus*, github, 2023. [Online]. Available: http://github.com/alipatti/cerberus.