An extended abstract [CCR26] of this article appears in the proceedings of USENIX Security Symposium 2026.
The full version of this article is available in the IACR eprint archive.

Anamorphic Messaging: Analyzing the Double Ratchet, Triple Ratchet, PQ3, and MLS

Hien Chu
TU Wien
hien.chu@tuwien.ac.at

Alessandro Corsi
FAU Erlangen-Nürnberg
alessandro.corsi@fau.de

Paul Rösler
FAU Erlangen-Nürnberg
paul.roesler@fau.de

Abstract

Anamorphic cryptography targets the scenario in which a dictator does not forbid the use of cryptography but requires all users to reveal their secret keys to them. Thus, the dictator can decrypt all honestly generated ciphertexts. The approach for bypassing this is to identify spots, such as random nonces, in existing cryptographic protocols in which secret messages can be hidden using an additional secret *double key*. So far, the literature mostly focused on identifying such spots in simple primitives like public-key encryption or signatures; only recently, an initial work identified limited spots in Signal's Double Ratchet Algorithm.

We are the first to leverage the statefulness of cryptographic communication protocols to employ *continuously updated double states* and, thereby, achieve *Forward Security*: Even if the adversary (i) observes all traffic, (ii) knows all users' regular secret key material at any stage of the protocol execution, and (iii) at some point learns the secret double state, the entire protocol execution looks benign although covert messages were previously hidden in the traffic anamorphically. We formalize this notion and also cover robustness and authenticity, which appear to be particularly relevant in the messaging context.

In this new model, we study four of the most relevant messaging protocols and identify hiding spots therein: Signal's Double Ratchet, Signal's Triple Ratchet, Apple's PQ3, and the two-party core of the Messaging Layer Security Standard. We focus on the cryptographic parts of these protocols and, despite their complexity, identify surprisingly few anamorphic hiding spots. We prove that all these protocols offer forward secure, authenticated anamorphic channels and we evaluate their bandwidths: While 16 bits can be embedded in every epoch of the Double Ratchet, Triple Ratchet and PQ3 provide 176 bits, respectively 256 bits, of bandwidth per post-quantum epoch, and MLS provides 688 bits per epoch.

1 Introduction

ANAMORPHIC CRYPTOGRAPHY. The concept of anamorphic cryptography was introduced in the seminal work by Persiano et al. [PPY22]: A dictator allows its citizens to use cryptography, but all their secret key material is required to be leaked to the dictator. Thereby, cryptography would protect the citizens' communication against external adversaries and other malicious users, but the dictator can eavesdrop all plaintexts. Multiple recent regulation attempts for requiring client-side scanning of end-to-end encrypted messaging (aka. "chat control") in the EU [Eur25] and other regions worldwide [UK 23, US 23] motivate that such strong(er) attacker models should be anticipated and studied.

To circumvent this strong form of surveillance, Persiano et al. [PPY22] as well as many follow-up works analyzed whether widely deployed cryptographic constructions allow for hiding messages from the dictator, even if the regular key material is leaked. Possible hiding spots include, for example, random nonces. For this, users generate an additional *double key* and encrypt *anamorphic messages* in the hiding spots. The dictator, then, cannot distinguish whether the hiding spots were computed honestly or whether anamorphic messages were embedded therein. To prevent this form of surveillance circumvention, the dictator would need to forbid every such suitable cryptographic construction.

ANAMORPHISM IN PRIMITIVES. Since the initial work [PPY22], many constructions of different primitives have been proven to allow for anamorphic channels, primarily including symmetric and asymmetric encryption [KPP+23b, BGH+24, PPY24, DPPY25, GC25, DL25a, BPRS25, CGM24a] and digital signatures [KPP+23a, JS24, DL25b, CDHT25]. Many of these works [WCHY23, JS24, BGH+24, CDHT25, DPPY25, BPRS25] also refined, improved, and strengthened the adversary model and extended the security requirements.

Another line of work investigated how far black-box methods, like rejection sampling, can and cannot be used to embed

anamorphic messages [CGM24b, CGM25]. Finally, a very recent list of papers developed constructions that are resistant against anamorphic embedding [DG25, CCGM25, ABG+25].

ANAMORPHISM IN PROTOCOLS. On the protocol level, so far only little work has be done to identify anamorphic channels (besides simply using the hiding spot of their primitive components): Wang et al. [WHL24] considered authenticated key exchange, Cinal et al. [CKKW25] analyzed anamorphism in Monero transactions, Giustolisi et al. [GRL25] studied anamorphic voting, and Luo and Jiang [LJ25] considered anamorphic private set intersection.

ANAMORPHISM IN MESSAGING. Although it is one of the primary means of digital communication today, messaging has only very recently been looked at from an anamorphic perspective by Rakeei et al. [RGR+25]. Concretely, this initial work identifies two hiding spots in Signal's Double Ratchet, proves their anamorphism, and evaluates their practicality by implementing them in Signal's code. One identified hiding spot is in the continuous Diffie-Hellman key updates, which we also use in our constructions. The other one uses transmitted Message Authentication Code (MAC) tags; this is, however, only useful if some of the regular Double Ratchet key material is never leaked to the attacker (which bypasses the original motivating attacker model). Another limitation is that, while the regular Double Ratchet uses continuously updated key material, the anamorphic extension proposed in [RGR+25] uses a static secret *double key*, hence it does not provide forward security for the anamorphic channel.

STATEFUL ANAMORPHISM AND FORWARD SECURITY. With our work, we overcome this limitation by allowing the anamorphic channel to update its additional secret double key material, which we call *anamorphic* (or *double*) *state*. Due to these updates, the anamorphic channel can provide forward security: If the anamorphic state is ever compromised, the attacker cannot learn whether the users ever embedded hidden messages in their communication.

We are only aware of two works that considered statefulness (yet without updating the static double key) in the context of anamorphic cryptography: Banfi et al. [BGH+24] allow their anamorphic encryption scheme to use a state for providing robustness, and Jaeger and Stracovsky [JS24] strengthen the notion of robustness for anamorphic signatures, introducing the notion of dictator unforgeability.

ROBUSTNESS AND AUTHENTICITY. Beyond forward secure anamorphism, we require that anamorphic extensions of messaging protocols are robust and preserve authenticity forward securely. This means that a receiver should never decrypt an anamorphic message from a regularly sent ciphertext; furthermore, the adversary must not be able to change ciphertexts such that anamorphic messages are manipulated in transmission.

Both properties are important per se and particularly relevant in the stateful anamorphic setting: If the receiver would interpret a regular ciphertext as an anamorphic one, they would perform an anamorphic receipt and state update that was not meant to be triggered by the sender; thus, the anamorphic states of sender and receiver would be diverged.

OUR SECURITY DEFINITION. In total, we require that: (1) Anamorphic extensions of regular messaging protocols cannot be detected as such, even if the attacker *continuously* learns the users' regular messaging states and *later* learns the users' additional anamorphic state; (2) Receiving a regular ciphertext does not yield an anamorphic message; and (3) Anamorphic messages cannot be manipulated in transmission.

DOUBLE RATCHET. Despite the complexity of the Double Ratchet [PM16, CCD+20, ACD19], we only identify a single hiding spot that allows for embedding anamorphic messages with respect to our security definition: the continuous Diffie-Hellman key updates. Concretely, Alice and Bob continuously generate and exchange fresh Diffie-Hellman shares. In our anamorphic extension, when Alice would generate such a fresh Diffie-Hellman share, she, instead, derives an exponent x' deterministically from her anamorphic state *ast*; next, she adds a deterministic encryption of her anamorphic message *am* to x', which yields the actual used exponent $x = x' + \mathrm{enc}(dk, am)$, where *dk* is a key from her anamorphic state *ast*; the resulting new Diffie-Hellman share sent to Bob is $X \leftarrow g^{x'+\mathrm{enc}(dk,am)}$. To decrypt this anamorphic message upon receipt of X, Bob will trial-generate exponent x the same way by deriving x' from his *ast* and, then, iterating over all possible anamorphic messages am^*. While this approach is efficient for the sender, the receiver costs are exponential in the number of transmitted bits. For this, we consider a bandwidth of 16 bits per Diffie-Hellman key update practical. For forward security, Alice and Bob update and replace their anamorphic state at every anamorphic send and receive operation using a pseudorandom generator. Rakeei et al. [RGR+25] use a similar embedding technique with a static anamorphic double key, and they improve the online receiver costs with a pre-computed lookup table. Due to the continuous anamorphic state updates in our construction, this speedup method is not applicable to our first anamorphic extension of the Double Ratchet.

Other than the Diffie-Hellman shares, all remaining components of regular Double Ratchet ciphertexts are derived deterministically from the Diffie-Hellman key exchanges. Thus, an attacker who permanently knows the users' regular messaging states can check whether these deterministic values are computed and transmitted honestly. With these results, we improve the idea of Rakeei et al. [RGR+25] by providing forward anamorphism.

PARTIAL SURVEILLANCE. The described extension of the Double Ratchet is aimed to provide a forward secure anamor-

phic channel for two users who are both surveilled by the dictator. If only *one* user is under control of the dictator and the other one is not (e.g., being located in a "free" country), it is reasonable to assume that the dictator does not learn the regular messaging state of the latter. We exploit this asymmetric corruption model and propose a second anamorphic extension for the Double Ratchet, in which Alice's regular messaging state stays secret. Based on that, we modify the generation and transmission of Diffie-Hellman shares further: Alice again encrypts her anamorphic message with key dk from her anamorphic state, and she encodes the resulting ciphertext to the group: $X \leftarrow \text{bin2gr}(\text{enc}(dk, am))$. Bob can decode and decrypt this message using the same key dk. To still provide correctness for the regular Double Ratchet, Alice needs to compute the Diffie-Hellman key exchange between her share X and Bob's share Y, without knowing the discrete logarithm of X. For this, both Alice and Bob deterministically derive Bob's exponent y from their shared anamorphic state, such that both can compute key X^y. With this improvement, the bandwidth of the anamorphic channel from Alice to Bob is increased to 176 bits, and Bob's anamorphic receive algorithm becomes fully efficient.

IMESSAGE PQ3. Apple recently designed and implemented a post-quantum secure variant [Ste24] of the Double Ratchet that, in addition to Diffie-Hellman key updates, performs key updates with ML-KEM [Nat24]. These post-quantum key updates are initiated at a lower frequency (every $\approx$ 50th message) than the Diffie-Hellman updates to reduce the resulting communication overhead. With each post-quantum update, one party generates a fresh ML-KEM key pair and shares the public key, and, in response, the other party encapsulates a symmetric key. The shared ML-KEM public key and the encapsulation ciphertext allow for embedding anamorphic messages. Furthermore, every PQ3 ciphertext is protected by a signature that allows for anamorphic embedding.

Concretely, the ML-KEM public key contains a randomly generated seed that, instead, can be the ciphertext of an anamorphic message; the ML-KEM ciphertext contains an encryption of a randomly sampled message that, instead, can be another ciphertext of an anamorphic message; the signature contains a nonce that, instead, can also be ciphertext of an anamorphic message; finally the Diffie-Hellman based key updates in PQ3 allow for the same embedding as the standard Double Ratchet.

The Double Ratchet and ML-KEM updates are conducted dynamically at different frequencies such that the bandwidth of PQ3 heavily varies: the Diffie-Hellman updates in every communication round-trip can contain 16 bits, the less frequent ML-KEM updates can contain 256 bits in the public key and the message, respectively, and the signature in every ciphertext can contain 256 bits of an anamorphic message. To ensure robustness and authenticity, the total bandwidth (up to $16+256+256 = 528$ bits) of each ciphertext is reduced by 80 bits due to adding a MAC tag to the anamorphic channel.

TRIPLE RATCHET. Similar to Apple's PQ3, also Signal improved the Double Ratchet for post-quantum security and named it *Triple Ratchet* [DJK+25]. The core idea of their improvement resembles the one of PQ3: In addition to Diffie-Hellman based key updates, ML-KEM based updates are continuously exchanged. However, instead of sending these updates at a lower frequency, the Triple Ratchet sends them in chunks: The shared ML-KEM public key and the encapsulation ciphertext are split into multiple smaller pieces; now, when sending a regular message in Signal, the resulting ciphertext is attached with one of these pieces. To send anamorphic messages, the embedding happens at the ML-KEM key generation and encapsulation, respectively, and the decryption can only happen when all chunks of each value are received. The overall anamorphic bandwidth, again, dynamically varies based on Diffie-Hellman round-trips and chunked ML-KEM based updates: up to $16+256-80 = 192$ bits.

MLS. The Messaging Layer Security (MLS) Standard [BBR+23, CGWZ25] was developed for efficient and secure communication in groups. In this work, we simplify its cryptographic core to the setting in which only two users communicate. This reduced variant of MLS covers continuous updates of the users' key material and the encryption of their regular messages. Concretely, Alice and Bob continuously generate fresh HPKE [BBLW22, ABH+21] key pairs, and they sign and, then, share the corresponding public keys; using the partner's public key, they encrypt a random seed for computing the group key; furthermore, each party can mix a pre-shared key and a public nonce into the derivation of the group key; finally, regular messages in MLS are protected with nonce-based encryption and another signature.

Each of these components of MLS yields an anamorphic hiding spot: the nonces in both signatures—one signing the shared public key and one signing regular message encryptions—can hide 256 bits each, the encrypted random seed for the group key computation can hide 256 bits, the public nonce mixed with pre-shared key into the group key computation can hide 256 bits, and the message encryption's nonce-reuse guard value can hide another 32 bits: overall up to $256+32-80 = 208$ bits per message and $256+256+256-80 = 688$ bits per epoch.

CONTRIBUTIONS. With our work, we demonstrate how to leverage the stateful potential of (messaging) protocols for forward securely hiding anamorphic messages in deployed real-world applications. For this, we specify definitions that cover a clean cryptographic core of Secure Messaging (Section 3.1), and we extend them for defining Forward Secure Anamorphic Messaging with Robustness and Authenticity (Section 3.3). To account for scenarios in which only one communication partner is under full control of the adversary, we also provide a meaningfully weakened notion of Partial Surveillance for

Anamorphic Messaging. We then build anamorphic extensions for the Double Ratchet (Section 4), Triple Ratchet (Section 5), PQ3 (Section 6), and the two-party essence of MLS (Section 7) in the full surveillance setting, and we provide a variant with higher bandwidth under partial surveillance for the Double Ratchet (Section 4.1). Finally, in Section 8 we evaluate the bandwidths of all anamorphic constructions and discuss the impact of our work.

FURTHER RELATED WORK. Our basic definitions of Secure Messaging are inspired by a long line of works, including [RMS18, CCD+17, CCD+20, BSJ+17, PR18b, PR18a, JS18, ACD19, BRV20, DHRR22, CR25]. Contrary to standard messaging definitions, however, in anamorphic cryptography corruptions are not temporary, and users cannot obtain confidentiality from the standard protocol.

A branch of the literature that tackles a similar problem from a different angle is Subversion Resilience: regular cryptographic algorithms are analyzed for whether modified variants thereof can be detected as such; in the context of messaging, this has been studied by Dodis et al. [DMST25]. Instead of embedding hidden messages in the technical building blocks of a protocol, Steganography [KJGR21, DCW+23, JBK24, BPL+25, WPC+25, LYSH25] can be used to hide messages in the payload itself. Finally, approaching the motivated problem from the opposite direction, Alwen et al. [ADGT25] propose a secure messaging protocol that integrates content moderation and policy compliance cryptographically.

Chakraborti et al. [CSS23] conduct the first analysis of covert channels in messaging protocols, concretely, by replacing the 16-byte initialization vector (IV) of Signal's Sealed Sender [Lun18]. Although they do not explicitly use the language of anamorphic cryptography, their threat model assumes an adversary with access to both users' private keys and system-level surveillance, against which they embed covert messages within a trusted execution environment (TEE). However, their embedding approach is incompatible with the Double Ratchet [PM16], which we consider in this work, as the latter derives IVs deterministically. How TEE-based approaches relate to the anamorphic constructions we develop here is an interesting question we leave for future work.

LIMITATIONS AND FUTURE WORK. The cryptographic core of secure messaging is only a small part of real-world messaging protocols; studying the full potential for embedding anamorphic channels in such protocols remains a greater task for future work that targets more than only the cryptographic components. In particular, our consideration of the MLS standard is limited to simple two-party communication; covering its entire protocol stack [ACDT20, ACJM20, BDR20, ACDT21, KPPW+21, PRSS21, AJM22, BDG+22, WPBB23, WPB25, CGWZ26] remains an open problem. Furthermore, transport protocols like TLS or Noise [DFGS21, DRS20, GHS+20, KNB19, CHH+17], interoperable extensions [RS23, LGGR26, LBP26], or anonymizing layers in messaging protocols [BRT23, TRB+25, MKA+21] could be analyzed. As we believe that post-compromise security for anamorphic communication can hardly be motivated, we refrain from capturing this notion in our work. Yet, it might be interesting to study more advanced or nuanced corruption models as well as forms of redundant security for anamorphic messaging in the future. For this, we consider our notion of Partial Surveillance a first step. Extending it to protocols other than the Double Ratchet, chosen here for its lowest full-surveillance bandwidth and thus largest relative gain, is left to future work. Further models could analyze the impact of side channels in the context of anamorphic encryption. Finally, it remains an open task to implement our anamorphic messaging extensions in their real codebase for evaluating their performance in practice and making them available for surveilled users.

2 Notation

By $x \leftarrow y$ we denote the assignment of the value of variable y to variable x and thus for a function I, $x \leftarrow I(y)$ means that x is assigned with the evaluation output of I on input y. For a probabilistic algorithm Y, by $x \leftarrow_{\$} Y(y)$ we denote the output x of the probabilistic evaluation of Y on input y and $x \leftarrow Y(y;r)$ denotes the output x of the deterministic evaluation of Y on y, where the evaluation's randomness is set to r. For a set X , $x \leftarrow_{\$} X$ is the uniform random sampling of value x from X.

We denote $\texttt{tru}, \texttt{fal}$ as Boolean values for true and false. Symbol 'ε' denotes an empty string and symbol '$\perp$' denotes an undefined element or an output that indicates rejections (thus it is not an element of explicitly defined sets).

If an element $x \in X$ is appended to list (or string) L then we denote this by $L \leftarrow L \| x$. Thus $\|$ also denotes the concatenation symbol of two strings. We define relations prefix-or-equal $\preceq$ and strictly-prefix $\prec$ over two strings. If x is a prefix of z (i.e., $z = x \| y$), then $z \xleftarrow{\lhd} x$ removes this prefix from z (i.e., $z \leftarrow y$).

In Appendix A, we provide standard security definitions for Strong Pseudorandom Permutations (PRP, resp. Symmetric Encryption, SE) and Authenticated Encryption with Associated Data (AEAD).

3 Definitions

3.1 Secure Messaging

We simplify the concept of Secure Messaging to its core such that we can cover the main cryptographic protocols of Signal's Double Ratchet [PM16, CCD+20, ACD19], Signal's Triple Ratchet [DJK+25], Apple's PQ3 [Ste24], and the Messaging Layer Security Standard [BBR+23, CGWZ25]. For clarity, we sometimes refer to these original versions as *Regular* or

just *Messaging Protocols* and to their modified variants as *Anamorphic Messaging Protocols*.

Syntax A Messaging Protocol is a tuple of algorithms $\mathrm{M} = (\mathrm{M.init}, \mathrm{M.snd}, \mathrm{M.rcv})$ with the following syntax:

- $\mathrm{M.init} : \emptyset \to_{\$} \mathcal{ST} \times \mathcal{ST}$ initializes the secret states of the two communication participants; this abstraction is standard and splits the initial key exchange from the actual messaging session.
- $\mathrm{M.snd} : \mathcal{ST} \times \mathcal{M} \to_{\$} \mathcal{ST} \times \mathcal{C}$ sends a message to the communication partner and updates the local secret state.
- $\mathrm{M.rcv} : \mathcal{ST} \times \mathcal{C} \to \mathcal{ST} \times \mathcal{M}$ receives a ciphertext from the communication partner and updates the state.

Communication Model We consider the communication participants to send and receive asynchronously and, hence, concurrently. However, for simplicity and clarity, we only require correctness under honest, reliable transmission of ciphertexts; this means that *immediate decryption* of re-ordered and lost ciphertexts is not required in our work.

Correctness Based on this, a Messaging Protocol is *correct* if every ciphertext, received by one party, yields the same message that was sent by the other party as long as all ciphertexts are forwarded honestly between the two communication partners. We capture this formally for Messaging Protocols and their anamorphic extensions in game CORR in Figure 1; the probability of winning this game (*i.e.*, of the game returning 1) must be 0 for correct schemes.

Security We refrain from defining security for Messaging Protocols as, in the anamorphic setting, all regular secret states are considered to be revealed to the adversary. Thus, the regular protocol cannot offer any meaningful security guarantees.

Nevertheless, the security of all protocols studied in this work has been analyzed in models [CCD+20, ACD19, DJK+25, Ste24, CGWZ25] that are in line with our syntax and correctness definitions.

3.2 Anamorphic Algorithms

We formally define how to extend Secure Messaging protocols for embedding anamorphic channels.

Syntax Extensions An Anamorphic Messaging Protocol is a tuple of algorithms $\mathrm{M} = (\mathrm{M.init}, \mathrm{M.snd}, \mathrm{M.rcv}, \mathrm{M.ainit}, \mathrm{M.asnd}, \mathrm{M.arcv})$ where M.ainit, M.asnd, and M.arcv have the following syntax:

- $\mathrm{M.ainit} : \mathcal{K} \times \mathcal{ST} \times \mathcal{AUX} \to \mathcal{AST}$ takes a symmetric key $k \in \mathcal{K}$ as well as the current state of the regular messaging protocol and some auxiliary information to derive a state for the anamorphic messaging protocol; each communication partner executes this algorithm individually with key k that is secretly shared out of band.
- $\mathrm{M.asnd} : \mathcal{ST} \times \mathcal{AST} \times \mathcal{M} \times \mathcal{M} \to_{\$} \mathcal{ST} \times \mathcal{AST} \times \mathbb{N} \times \mathcal{C}$ sends message $m \in \mathcal{M}$ via the regular messaging protocol as well as message $am \in \mathcal{M}$ via the anamorphic channel in ciphertext $c \in \mathcal{C}$; in addition to the updated states and the ciphertext, integer $l \in \mathbb{N}$ is output, which indicates how many bits of message am were transmitted in c; due to the behavior of the regular messaging protocol, the bandwidth of the anamorphic channel can vary dynamically.
- $\mathrm{M.arcv} : \mathcal{ST} \times \mathcal{AST} \times \mathcal{C} \to \mathcal{ST} \times \mathcal{AST} \times \mathcal{M} \times \mathcal{P}(\mathbb{N} \times \mathcal{M})$ receives ciphertext c and outputs the normal message and a set of anamorphic message chunks $am \in \mathcal{M}$, where counter $r \in \mathbb{N}$ indicates to which full anamorphic message each chunk belongs; the regular messaging protocol may dynamically chunk hiding spots and transmit them interleavedly.

The syntax of algorithms M.init, M.snd, and M.rcv is defined above. Construction-dependent algorithm $\mathcal{X}$ on input of the partner's regular state specifies the auxiliary information input to algorithm M.ainit.

Correctness An Anamorphic Messaging Protocol is *correct* if regular messages, sent via algorithms M.snd or M.asnd, are received by algorithms M.rcv or M.arcv, unless the corresponding ciphertexts are modified in transmission. We capture this formally for Regular and Anamorphic Messaging Protocols in game CORR in Figure 1; the probability of winning this game must be 0 for correct schemes.

For the consistency between sent and received *anamorphic messages*, we define *robustness* and *authenticity* at the end of this section. Since consistent detection and receipt of anamorphic ciphertexts relies on cryptographic hardness assumptions, we refrain from including such requirements in our correctness definition here.

3.3 Anamorphism

The core security property that we consider in this work is that the *Anamorphic Messaging Protocol* variant of an existing *Regular Messaging Protocol* cannot be distinguished from the unmodified version of the latter. For this, we define game $\mathrm{ANA}^{b,\delta}$ with challenge bit b and delay parameter δ in Figure 2 in which an adversary $\mathcal{A}$ can choose the randomness for the regular state initialization and, then, query the following oracles:

- $\mathrm{Snd}(P, m) \to (st_P, c)$ executes M.snd for sender P and returns the resulting regular state and ciphertext.

Game $\text{CORR}_{\text{M}}(\mathcal{A})$
00 $(st_A, st_B) \leftarrow_{\$} \text{M.init}$
01 $k \leftarrow_{\$} \mathcal{K}$
02 For $P \in \{A, B\}$:
03 $ast_P \leftarrow \text{M.ainit}(k, st_P, \mathcal{X}(st_{\bar{P}}))$
04 $s_P, r_P \leftarrow 0^2$
05 $oos_P \leftarrow \texttt{fal}$ \Out-of-sync variable
06 $MC_P[\cdot] \leftarrow \bot$
07 Invoke $\mathcal{A}$
08 Stop with 0

Oracle $\text{Snd}(P, m)$
09 $(st_P, c) \leftarrow_{\$} \text{M.snd}(st_P, m)$
10 If $oos_P = \texttt{fal}$:
11 $MC_P[s_P] \leftarrow (m, c)$
12 $s_P \leftarrow s_P + 1$
13 Return c

Oracle $\text{ASnd}(P, m, am)$
14 $(st_P, ast_P, l, c) \leftarrow_{\$}$ $\text{M.asnd}(st_P, ast_P, m, am)$
15 If $oos_P = \texttt{fal}$:
16 $MC_P[s_P] \leftarrow (m, c)$
17 $s_P \leftarrow s_P + 1$
18 Return (l, c)

Oracle $\text{Rcv}(P, c)$
19 $(st_P, m) \leftarrow \text{M.rcv}(st_P, c)$
20 $(m', c') \leftarrow MC_{\bar{P}}[r_P]$
21 If $oos_P = \texttt{fal} \wedge m \neq \bot \wedge c \neq c'$:
22 $oos_P \leftarrow \texttt{tru}$
23 If $oos_P = \texttt{fal} \wedge m \neq \bot$:
24 If $m \neq m'$: Stop with 1
25 $r_P \leftarrow r_P + 1$
26 Return m

Oracle $\text{ARcv}(P, c)$
27 (st_P, ast_P, m, am) $\leftarrow \text{M.arcv}(st_P, ast_P, c)$
28 $(m', c') \leftarrow MC_{\bar{P}}[r_P]$
29 If $oos_P = \texttt{fal} \wedge m \neq \bot \wedge c \neq c'$:
30 $oos_P \leftarrow \texttt{tru}$
31 If $oos_P = \texttt{fal} \wedge m \neq \bot$:
32 If $m \neq m'$: Stop with 1
33 $r_P \leftarrow r_P + 1$
34 Return (m, am)

Figure 1: Correctness for Secure and Anamorphic Messaging Protocols.

Game $\text{ANA}_{\text{M}}^{b,\delta}(\mathcal{A})$
00 $(\omega, \rho) \leftarrow_{\$} \mathcal{A}$ //No Oracles
01 $(st_A, st_B) \leftarrow \text{M.init}(\rho)$
02 $k \leftarrow_{\$} \mathcal{K}$
03 For $P \in \{A, B\}$:
04 $ast_P \leftarrow_{\$} \text{M.ainit}(k, st_P, \mathcal{X}(st_{\bar{P}}))$
05 $s_P, r_P \leftarrow 0^2$; $CH_P \leftarrow \emptyset$
06 $X_P \leftarrow \texttt{fal}$
07 $b' \leftarrow_{\$} \mathcal{A}(\omega)$
08 Stop with b'

Oracle $\text{Snd}(P, m)$
09 $(st_P, c) \leftarrow_{\$} \text{M.snd}(st_P, m)$
10 $s_P \leftarrow s_P + 1$
11 Return (st_P, c)

Oracle $\text{ASnd}(P, m, am)$
12 $(st_P, ast_P, l, c) \leftarrow_{\$}$ $\text{M.asnd}(st_P, ast_P, m, am)$
13 $s_P \leftarrow s_P + 1$
14 Return (st_P, l, c)

Oracle $\text{Expose}(P)$
15 Require $\nexists s' \in CH_{\bar{P}} : s' + \delta > r_P$
16 $X_P \leftarrow \texttt{tru}$; $X_{\bar{P}} \leftarrow \texttt{tru}$
17 Return ast_P

Oracle $\text{Chall}(P, m, am)$
18 Require $X_{\bar{P}} = \texttt{fal}$
19 If $b = 0$:
20 $(st_P, c) \leftarrow_{\$} \text{M.snd}(st_P, m)$
21 Else:
22 $(st_P, ast_P, l, c) \leftarrow_{\$}$ $\text{M.asnd}(st_P, ast_P, m, am)$
23 $CH_P \xleftarrow{\cup} \{s_P\}$
24 $s_P \leftarrow s_P + 1$
25 Return (st_P, c)

Oracle $\text{ARcv}(P, c)$
26 $(st_P, ast_P, m, am) \leftarrow$ $\text{M.arcv}(st_P, ast_P, k, c)$
27 For all $(r', am') \in am$:
28 If $r' \in CH_{\bar{P}}$:
29 $am \leftarrow am \setminus \{(r', am')\}$
30 If $m \neq \bot$: $r_P \leftarrow r_P + 1$
31 Return (st_P, m, am)

Figure 2: Anamorphism for Messaging with Forward Secure Continuous State Updates.

- $\text{ASnd}(P, m, am) \rightarrow (st_P, l, c)$ executes M.asnd for sender P and returns the resulting regular state, ciphertext, and length of the sent anamorphic message.
- $\text{Chall}(P, m, am) \rightarrow (st_P, c)$ depending on bit b either executes M.snd or M.asnd for sender P and returns the resulting regular state and the ciphertext, which is marked as a challenge. Challenges are only permitted if receiver $\bar{P}$ has not been exposed yet.
- $\text{ARcv}(P, c) \rightarrow (st_P, m, am)$ executes M.arcv and returns the resulting regular state, message, and anamorphic message chunks that were not marked as challenges.
- $\text{Expose}(P) \rightarrow (ast_P)$ reveals the anamorphic state of party P. Exposures are only permitted if all challenges sent to P were already received; parameter δ specifies the maximum delay that receiving anamorphic messages can take after sending.

At the end of the game, adversary $\mathcal{A}$ returns a guess b' such that the advantage in winning the game against Anamorphic Messaging Protocol M is defined as:

$$\text{Adv}_{\text{M}}^{\text{ana},\delta}(\mathcal{A}) := \left| \Pr[\text{ANA}_{\text{M}}^{1,\delta}(\mathcal{A}) = 1] - \Pr[\text{ANA}_{\text{M}}^{0,\delta}(\mathcal{A}) = 1] \right|.$$

Since we allow exposures of the anamorphic state as long as all prior challenges were received, our definition covers *forward security*. For all constructions except the Triple Ratchet, it holds that $\delta = 0$.

Symmetric Exposure In our definitions, we treat *both* parties' anamorphic key material as being revealed after *one* party is exposed. This allows for constructions that only rely on symmetric cryptography. Furthermore, as we demonstrate in the following sections, relying only on symmetric cryptography increases the bandwidth of the anamorphic channel.

Adapting our definitions to require that the exposure of party P remains harmless for the secrecy of anamorphic messages sent by that party P thereafter is straightforward. Clearly, this strengthens the security guarantees and captures a form of *sender post-compromise security*. However, as indicated above, this modification induces the need for public-key cryptography and, thereby, reduces the bandwidth of the anamorphic channel.

Post-Compromise Security We refrain from requiring that an exposed party recovers from an exposure (i.e., P generates and shares fresh public-key material with $\bar{P}$ such that anamorphic messages received in the future will be secure again). Such full recovery can make sense in Regular Messaging Protocols: After an active exposure attack ends, an implementation bug is fixed, etc., the parties may wish to regain secrecy in the future. In contrast, in the anamorphic setting, an adversary who learns about the existence of a hidden state through exposure will likely intensify their surveillance. Thus, we do not consider such forms of post-compromise security relevant in practice.

Partial Surveillance Anamorphic Messaging Protocols are aimed to hide communication between two users under strong

surveillance. It can also be useful for communication between one user who is heavily surveilled and another who is outside the control of the dictator. In this latter case, the adversary may only learn the regular messaging state of the surveilled party, whereas the regular messaging state of the other party remains uncorrupted.

We capture this notion of *partial surveillance* under *partially active adversaries* formally in the game $\text{PANA}^{b,\delta}$ in Figure 12, in Appendix B. The difference from our main game from Figure 2 is that, in Figure 12, the return values of oracles Snd, ASnd, Chall, and ARcv differ depending on whether they are queried for Alice or Bob: if Alice is queried to send or receive, only the resulting ciphertext, resp. message, is given to the adversary; otherwise, the oracle's output also includes Bob's current regular messaging state. Furthermore, ciphertexts received by Alice are required to be delivered honestly.

The advantage in winning game $\text{PANA}^{b,\delta}$ against Anamorphic Messaging Protocol M is defined as:

$$\text{Adv}^{\text{pana},\delta}_{\text{M}}(\mathcal{A}) := \left|\Pr[\text{PANA}^{1,\delta}_{\text{M}}(\mathcal{A})=1] - \Pr[\text{PANA}^{0,\delta}_{\text{M}}(\mathcal{A})=1]\right|.$$

Game $\text{AUTH}_{\text{M}}(\mathcal{A})$
00 $(\omega,\rho) \leftarrow_{\$} \mathcal{A}$ //No Oracles
01 $(st_A, st_B) \leftarrow \text{M.init}(\rho)$
02 $k \leftarrow_{\$} \mathcal{K}$
03 For $P \in \{A,B\}$:
04 $\quad ast_P \leftarrow_{\$} \text{M.ainit}(k, st_A, \mathcal{X}(st_{\bar{P}}))$
05 $\quad s_P, r_P, ra_P \leftarrow 0^3$; $AM_P[\cdot] \leftarrow \perp$
06 $\quad X_P \leftarrow \infty$
07 Invoke $\mathcal{A}(\rho)$
08 Stop with 0

Oracle $\text{Snd}(P,m)$
09 $(st_P, c) \leftarrow_{\$} \text{M.snd}(st_P, m)$
10 $AM_P[s_P] \leftarrow \varepsilon$
11 $s_P \leftarrow s_P + 1$
12 Return (st_P, c)

Oracle $\text{Expose}(P)$
13 $X_P \leftarrow s_P$; $X_{\bar{P}} \leftarrow ra_P$
14 Return ast_P

Oracle $\text{ASnd}(P,m,am)$
15 $(st_P, ast_P, l, c) \leftarrow_{\$}$
$\quad \text{M.asnd}(st_P, ast_P, m, am)$
16 $AM_P[s_P] \leftarrow am_{1,\ldots,l}$
17 $s_P \leftarrow s_P + 1$
18 Return (st_P, l, c)

Oracle $\text{ARcv}(P,c)$
19 (st_P, ast_P, m, am)
$\quad \leftarrow \text{M.arcv}(st_P, ast_P, c)$
20 For all $(r', am') \in am$:
21 $\quad$ If $am' \not\preceq AM_{\bar{P}}[r'] \wedge r' < X_{\bar{P}}$:
22 $\quad\quad$ Stop with 1
23 $\quad AM_{\bar{P}}[r'] \overset{\preceq}{\leftarrow} am'$
24 If $m \neq \perp$: $r_P \leftarrow r_P + 1$
25 $ra_P \leftarrow \min(r_P, r' \mid AM_{\bar{P}}[r'] \neq \varepsilon)$
26 Return (st_P, m, am)

Figure 3: Robustness and authenticity for Anamorphic Messaging Protocols. If x is a prefix of z (i.e., $z = x\|y$), then $z \overset{\preceq}{\leftarrow} x$ removes this prefix from z (i.e., $z \leftarrow y$).

Robustness and Authenticity An important property of anamorphic channels is *robustness*, which means that a regular ciphertext is not interpreted as an anamorphic one by the receiver; otherwise, the receiver may decrypt an anamorphic message that has never been sent as such by the sender. For our setting, this is particularly important because sending and receiving anamorphic messages updates the anamorphic states. Thus, incorrect anamorphic state updates would drift sender and receiver out of sync.

This property is closely related to *authenticity* of the anamorphic channel, which covers that an adversary must not be able to alter anamorphic messages between Alice and Bob. For this reason, we capture both properties together in game AUTH in Figure 3, in which adversary $\mathcal{A}$ can choose the randomness for the regular state initialization and then query oracles Snd and ASnd as in game ANA, described above. Oracle Expose marks the sender exposed. Oracle ARcv stops the game with return value 1 if the received anamorphic message chunk differs from the corresponding sent one, unless the sender has been marked exposed for this message. The advantage in winning game AUTH against Anamorphic Messaging Protocol M is defined as:

$$\text{Adv}^{\text{auth}}_{\text{M}}(\mathcal{A}) := \Pr[\text{AUTH}^{1}_{\text{M}}(\mathcal{A})=1].$$

4 Double Ratchet

The Double Ratchet (DR) protocol consists of a *symmetric* ratchet and an *asymmetric* ratchet. The asymmetric ratchet proceeds in epochs: Alice updates her Diffie-Hellman (DH) key g^s, then, when Bob receives this update, Bob updates his DH key g^r, and, when Alice receives this update, she begins the next epoch. These updates feed the symmetric key chains and, thereby, provide post-compromise security. A detailed specification of this regular Double Ratchet protocol is provided in Figure 16 in Appendix C, which is inspired by the notation by Alwen et al. [ACD19]. Our anamorphic extension, presented in Figure 4, makes use of the DH key updates to embed anamorphic messages. For this, the parties share two anamorphic double keys, dk_A and dk_B, which compose their anamorphic state *ast*: each party will use one of the keys to embed anamorphic messages, and the other key to decrypt them.

To embed an anamorphic message *am*, one solution could be to let the sender derive a nonce-key pair from the double key through a PRG $(n', mk, dk_{\mathtt{s}}) \leftarrow \text{G}(dk_{\mathtt{s}})$, then use the nonce-key pair to encrypt the anamorphic message into the DH exponent: $S \leftarrow g^{\text{enc}(mk,am)}$. The resulting anamorphic DH key S is then embedded in a normal DR ciphertext that, also, contains regular message m. However, this approach falls short when the dictator extracts S from the anamorphic ciphertext and simply sends m with a fresh DH key S^*, instead; the dictator can then embed the extracted anamorphic DH key S in a later ciphertext. For this, note that the adversary permanently knows all regular states and, therefore, can place the anamorphic DH key in arbitrary sent DR ciphertext.

For this reason, we use an alternative approach. As shown at line 10 we let the sender use the appropriate double key and a PRG G to generate a PRF key k and the new anamorphic state: $(k, dk_{\mathtt{s}}) \leftarrow \text{G}(dk_{\mathtt{s}})$. The PRF F_k is then used to generate a contextualized exponent s' and nonce-key pair (n', mk), taking as input the index n_S of the sent message: $(s', n', mk) \leftarrow \text{F}_k(n_S)$. Doing so, we implicitly bind the anamorphic message to

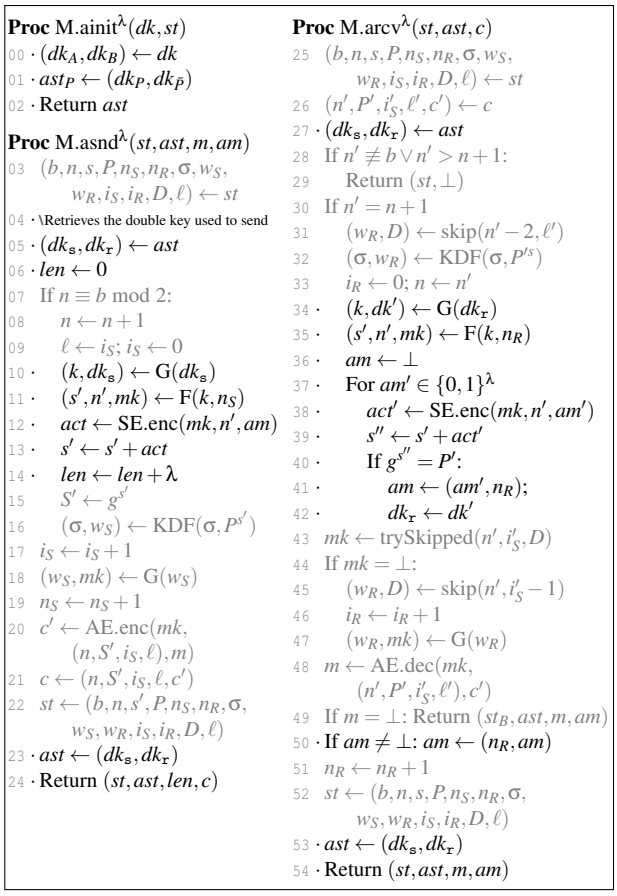

Figure 4: Anamorphic extension of the Signal scheme with Double Ratchet. Gray texts belong to the original protocol; black marks the anamorphic extension.

the n_S-th message sent and disable the mentioned reordering attack. Next, s' is added to the anamorphic ciphertext $act = \text{SE.enc}(mk, n', am)$ to generate the subsequent Diffie-Hellman secret key $s' + act$ (see lines 12,13).

To decrypt the anamorphic message, as shown at line 37, the receiver performs a trial decryption: they derive the same exponent and nonce-key pair s', n', mk of line 11, and check if an anamorphic message has been used to generate the received public key. For this, they brute-force over the anamorphic message space $\{0,1\}^{\lambda}$. If the decryption is successful, the receiving party will update their double key, synchronizing with the sender.

Bandwidth Although the anamorphic encryption of λ bits occurs in constant time, anamorphic decryption requires the receiver to perform $O(2^{\lambda})$ modular exponentiations. Therefore, this embedding spot can only be used to hide a limited amount of anamorphic bits; we assume a 16-bit bandwidth for this anamorphic channel, which the sending party can exploit once every epoch.

Analysis We prove correctness, anamorphism, robustness, and authenticity of our DR extension as defined in Figures 1 to 3.

Theorem 1 (Anamorphism)**.** *If* G *and* F *are random oracles and* SE *is* IND$-*secure, then the anamorphic extension from Figure 4 provides anamorphic security and, in particular, we have*

$$\begin{aligned}\text{Adv}^{\text{ana}}_{\text{M}}(\mathcal{A}) \leq\ & \frac{(q_{\text{G}_{\text{ext}}} + q_{\text{F}_{\text{ext}}})(q_{\text{ASnd}} + q_{\text{Chall}})}{|\mathcal{K}|} \\ & + \frac{(q_{\text{ASnd}} + q_{\text{Chall}})^2}{|\mathcal{K}|} + \frac{(q_{\text{ASnd}} + q_{\text{Chall}})^2}{p|\mathcal{K}||\mathcal{K}|} \\ & + \frac{q_{\text{ARcv}} 2^{\lambda}}{p},\end{aligned}$$

where q_{ASnd}, q_{Chall} and q_{ARcv} are the number of query calls to the ASnd, Chall *and* ARcv *oracles, $q_{\text{G}_{\text{ext}}}$ and $q_{\text{F}_{\text{ext}}}$ are the number of query calls to the* G *and* F*, respectively, and p is the order of g.*

Theorem 2 (Authenticity)**.** *If* G *and* F *are random oracles and* SE *is* IND$-*secure, then our anamorphic extension for the Double Ratchet algorithm provides authenticity, and, in particular*

$$\begin{aligned}\text{Adv}^{\text{auth}}_{\text{M}}(\mathcal{A}) \leq\ & \frac{(q_{\text{G}_{\text{ext}}} + q_{\text{F}_{\text{ext}}}) \cdot q_{\text{ASnd}}}{|\mathcal{K}|} \\ & + \frac{q^2_{\text{ASnd}}}{|\mathcal{K}|} + \frac{q^2_{\text{ASnd}}}{p|\mathcal{N}||\mathcal{K}|} \\ & + \frac{q_{\text{ARcv}} \cdot 2^{\lambda}}{p},\end{aligned}$$

where q_{ASnd} and q_{ARcv} are respectively the number of query calls to the ASnd *and* ARcv *oracles, $q_{\text{G}_{\text{ext}}}$ and $q_{\text{F}_{\text{ext}}}$ are calls to the external interfaces of the random oracles, and p is the order of g.*

The formal proofs are deferred to Appendix C.

4.1 Higher Bandwidth at Partial Surveillance

We proceed with considering a *partial surveillance* setting in which the dictator surveils only one of the two parties, say Bob, and does not expose Alice's regular state. Since the dictator can then no longer check if Alice's Diffie-Hellman share was generated honestly, Alice may place a full pseudorandom SE ciphertext in it instead of relying on a brute-forced 16-bit message as in the full-surveillance construction. This is shown in the construction of Figure 5, line 09, and is precisely what lets Alice use the entire DH public key for anamorphic bandwidth.

Similar to the first construction, the two parties share an anamorphic double key dk in their anamorphic state ast in order to derive a one-time SE key k per anamorphic message am.

To send am, Alice encrypts it using k as $ct \leftarrow \text{enc}(k, am)$, and maps the ciphertext to the DH public key $S' \leftarrow \text{bin2gr}(ct)$. To ensure the mapped ciphertext is pseudorandom over $\mathbb{G}$, we employ the solution from [BHKL13]. Bob can decrypt this message using the same double key dk. To still provide correctness for the regular Double Ratchet, Alice needs to compute the Diffie-Hellman key exchange between her share X and Bob's share Y. For this, both Alice and Bob deterministically derive Bob's exponent y from their shared anamorphic state, such that both can compute key X^y. Figure 5 shows this construction, highlighting only the anamorphic changes; its full description Figure 15 is deferred to Appendix D.

Proc M.ainit(k, st_P)
$(dk_A, dk_B) \leftarrow k$
$ast_P \leftarrow (dk_P, dk_{\dot{P}})$
Return ast_P

Proc M.asnd(st_A, ast_A, m, am)
$(b, n, s, P, n_S, n_R, \sigma, w_S, w_R, i_S, i_R, D, \ell) \leftarrow st$
$(dk_s, dk_r, p, S) \leftarrow ast_A$
$len \leftarrow 0$
If $n \equiv b \bmod 2$
  $n \leftarrow n+1$
  $\ell \leftarrow i_S$; $i_S \leftarrow 0$
  $(k, dk_s) \leftarrow \text{G}(dk_s)$
  $(n', mk) \leftarrow F_k(n_S)$
  $ct \leftarrow_\$ \text{SE.enc}(mk, n', am)$
  $S' \leftarrow \text{bin2gr}(ct)$
  $len \leftarrow len + |am|$
  $(\sigma, w_S) \leftarrow \text{KDF}(\sigma, S'^p)$
  ...
$st_A \leftarrow (b, n, \bot, P, n_S, n_R, \sigma, w_S, w_R, i_S, i_R, D, \ell)$
$ast_A \leftarrow (dk_s, dk_r, p, S')$
Return (st_A, ast_A, len, c)

Proc M.asnd$^\lambda(st_B, ast_B, m, am)$
\allowed $am \in \mathcal{M} \cup \{\bot\}$
\adapted from M.asnd in Figure 4
  ...
$(dk_s, dk_r) \leftarrow ast_B$
$len \leftarrow 0$
If $n \equiv b \bmod 2$:
  $n \leftarrow n+1$
  $\ell \leftarrow i_S$; $i_S \leftarrow 0$
  $(k, dk_s) \leftarrow \text{G}(dk_s)$
  $(s', n', mk) \leftarrow F_k(n_S)$
  If $am \neq \bot$:
    $act \leftarrow$
      $\text{SE.enc}(mk, n', am)$
    $s' \leftarrow s' + act$
    $len \leftarrow len + \lambda$
  ...
$ast_B \leftarrow (dk_s, dk_r)$
Return (st_B, ast_B, len, c)

Proc M.arcv(st_B, ast_B, c)
  ...
$(dk_s, dk_r) \leftarrow ast_B$
If $n' \not\equiv b \vee n' > n+1$:
  Return $(st_B, \bot)$
If $n' = n+1$
  $(w_R, D) \leftarrow \text{skip}(n'-2, \ell')$
  $(\sigma, w_R) \leftarrow \text{KDF}(\sigma, P'^s)$
  $i_R \leftarrow 0$; $n \leftarrow n'$
  $(k, dk_r) \leftarrow \text{G}(dk_r)$
  $(n', mk) \leftarrow F_k(n_R)$
$ct \leftarrow \text{gr2bin}(P')$
$am \leftarrow \text{SE.dec}(mk, n', ct)$
  ...
$ast_B \leftarrow (dk_s, dk_r)$
Return (st_B, ast_B, m, am)

Proc M.arcv$^\lambda(st_A, ast_A, c)$
\adapted from M.arcv in Figure 4
  ...
$(dk_s, dk_r, p, S) \leftarrow ast_A$
If $n' \not\equiv b \vee n' > n+1$:
  Return $(st_A, \bot)$
If $n' = n+1$
  $(k, dk') \leftarrow \text{G}(dk_r)$
  $(s', n', mk) \leftarrow \text{F}_k(n_R)$
  $am \leftarrow \bot$; $p' \leftarrow s'$
  If $g^{s'} \neq P'$:
    For $am' \in \{0,1\}^\lambda$
      $act' \leftarrow \text{SE.enc}(mk, n', am')$
      $s'' \leftarrow s' + act'$
      If $g^{s''} = P'$:
        $am \leftarrow am'$; $dk_r \leftarrow dk'$
        $p' \leftarrow s''$
  $(w_R, D) \leftarrow \text{skip}(n'-2, \ell')$
  $(\sigma, w_R) \leftarrow \text{KDF}(\sigma, S'^{p'})$
  $i_R \leftarrow 0$; $n \leftarrow n'$
  ...
$ast_A \leftarrow (dk_s, dk_r, p', S)$
Return (st_A, ast_A, m, am)

Figure 5: Anamorphic extension of the Signal scheme using Double Ratchet, excluding Alice's regular state exposure. Gray texts belong to the original protocol; black marks the anamorphic extension.

Bob occasionally sends a 16-bit anamorphic message am to Alice, as illustrated in our first extension in Figure 4. He slightly alters his Diffie-Hellman (DH) exponent by adding the SE ciphertext of the anamorphic message. Alice can still verify that Bob follows the protocol and ensure the correctness of the regular DR. Indeed, Alice either confirms his use of the determined exponent, or successfully decrypts through brute-forcing all am.

Remark 1 (Security of the original DR protocol)**.** *In general, because anamorphic and regular execution are indistinguishable, the original messaging channel's security is not weakened by the anamorphic extension. The only caveat is that, under our partial surveillance scheme, exposing the anamorphic state undermines the post-compromise (PCS) property of the original DR. However, since regular and anamorphic states should be maintained separately (which is consistent with the anamorphic setting), our anamorphic extension does not weaken regular PCS as long as the anamorphic state remains uncompromised.*

Bandwidth We can leverage the entire 256-bit of the DH public key for a pseudorandom ciphertext of am. By using an SE primitive with an authentication tag of 80 bits, we achieve a bandwidth of 176 bits per epoch for the anamorphic channel from Alice to Bob. The channel from Bob to Alice still allows a bandwidth of 16 bits. This asymmetry is inherent to the model: as Bob remains surveilled, his Diffie-Hellman share must stay verifiable by the dictator, so the reverse channel from Bob to Alice retains the 16-bit brute-force bandwidth of the first construction.

Analysis We prove correctness, anamorphism under partial surveillance, robustness, and authenticity of our DR extension as defined in Figures 1, 3 and 12.

We defer the formal proofs to Appendix D.

Theorem 3 (Anamorphism)**.** *If* G *and* F *are random oracles and* SE *is an* IND$ *and* SUF-CMA*-secure encryption algorithm, then the anamorphic extension from Figure 5 provides anamorphic security, and, in particular, we have*

$$\begin{aligned}\text{Adv}^{\text{pana}}_{\text{M}}(\mathcal{A}) \leq\ & \frac{(q_{\text{G}_{\text{ext}}} + q_{\text{F}_{\text{ext}}})(q_{\text{ASnd}} + q_{\text{Chall}})}{|\mathcal{K}|} \\ & + \frac{(q_{\text{ASnd}} + q_{\text{Chall}})^2}{|\mathcal{K}|} + \frac{(q_{\text{ASnd}} + q_{\text{Chall}})^2}{p|\mathcal{N}||\mathcal{K}|} \\ & + \frac{q_{\text{ARcv}} 2^\lambda}{p} \\ & + (q_{\text{ASnd}} + q_{\text{Chall}})\text{Adv}^{\text{ind\$}}_{\text{SE}}(\mathcal{D}) + \text{Adv}^{\text{suf-cma}}_{\text{SE}}(\mathcal{D}),\end{aligned}$$

where q_{ASnd}, q_{Chall} and q_{ARcv} are the number of query calls to the ASnd, Chall *and* ARcv *oracles, $q_{\text{G}_{\text{ext}}}$ and $q_{\text{F}_{\text{ext}}}$ are the number of query calls to the* G *and* F*, respectively.*

Theorem 4 (Authenticity). *If* G *and* F *are random oracles and* SE *is an* IND$ *and* SUF-CMA*-secure encryption algorithm, then our anamorphic extension from Figure 5 provides authenticity, and, in particular*

$$\mathrm{Adv}^{\mathrm{auth}}_{\mathrm{M}}(\mathcal{A}) \leq \frac{(q_{\mathrm{G_{ext}}} + q_{\mathrm{F_{ext}}}) \cdot q_{\mathrm{ASnd}}}{|\mathcal{K}|} + \frac{q^2_{\mathrm{ASnd}}}{|\mathcal{K}|} + \frac{q^2_{\mathrm{ASnd}}}{p|\mathcal{N}||\mathcal{K}|} + \frac{q_{\mathrm{ARcv}} \cdot 2^{\lambda}}{p} + \mathrm{Adv}^{\text{suf-cma}}_{\mathrm{SE}}(\mathcal{D})$$

where q_{ASnd} *and* q_{ARcv} *are the number of query calls to the* ASnd *and* ARcv *oracles,* $q_{\mathrm{G_{ext}}}$ *and* $q_{\mathrm{F_{ext}}}$ *are the number of query calls to the* G *and* F*, respectively.*

5 Triple Ratchet

The Triple Ratchet [ADJ+25] is the newly deployed, quantum-safe key agreement protocol used in Signal. At its core, it employs another post-quantum key update mechanism, the *third ratchet*, on top of the classical double ratchet; in particular, a variant of ML-KEM, called *ML-KEM Braid* [Sch, Nat24], is used. The two parties take turns sending the encapsulation key and the KEM ciphertext, thereby defining *post-quantum epochs*.

During every post-quantum epoch, the party responsible for sending the encapsulation key first sends the seed ρ, which is used to generate parts of the encapsulation key, and then the rest of the key ek_v. Conversely, the party sending the ciphertext computes the first part of the ciphertext ct_1 and starts its transmission upon receiving ρ; they complete the computation of ct_2 and transmit it after receiving the second part of the encapsulation key ek_v. Subsequently, they will initiate the next post-quantum epoch, switching their role and refreshing their key material with the newly exchanged post-quantum key. The described optimization is called *opportunistic sending*, and constitutes one of the foundational components of the ML-KEM Braid protocol.

Additionally, to avoid spikes in bandwidth consumption, the Triple Ratchet employs a data chunking technique: both the encapsulation key and the ciphertext are sent split into chunks. For a complete description of the protocol state machine and the protocol itself, we refer the interested reader to [Sch, DJK+25, ADJ+25].

Due to the presence of the post-quantum ratchet, in addition to the embedding spots already provided by the Double Ratchet, we can exploit the anamorphic spots of the KEM. In particular, for each post-quantum epoch, the party that sends the encapsulation key can embed 32 bytes of an anamorphic message in the seed ρ, using an encryption algorithm SE with pseudorandom ciphertexts. Analogously, the party computing the ciphertext can embed 32 bytes of an anamorphic message in the random message m encrypted by the public-key

Proc KEM.agen(amk, n, am)
00 $seed \leftarrow_{\$} \{0,1\}^{256}$
01 $(\rho, \sigma) \leftarrow \mathrm{G}_1(seed)$
02 · $(\rho', \sigma) \leftarrow \mathrm{G}_1(seed)$
03 · $\rho \leftarrow$ SE.enc(amk, n, am)
04 $(encK, decK) \leftarrow$ KPKE.Gen(ρ, σ)
05 Return $(encK, decK)$

Proc KEM.adec($decK, ct, amk, n$)
06 $(decK', encK, h, z) \leftarrow decK$
07 $m' \leftarrow$ KPKE.Dec($decK', ct$)
08 $(K', r') \leftarrow \mathrm{H}(m' \,\|\, h)$
09 $ct' \leftarrow$ KPKE.Enc(m', r')
10 If $ct \neq ct'$:
11 $K' \leftarrow H'(z \,\|\, ct)$
12 · Else:
13 · $am \leftarrow$ SE.dec(amk, n, m')
14 Return (K', am)

Proc KEM.enc$_1$(ρ, h, amk, n, am)
15 $m \leftarrow_{\$} \{0,1\}^{256}$
16 · $m \leftarrow$ SE.enc(amk, n, am)
17 $(K, u) \leftarrow$ KPKE.Enc$_1$(m, ρ, h)
18 Return (K, u)

Proc KEM.enc$_2$($encK$)
19 \m is the same derived in Encaps$_1$
20 $(t, \rho) \leftarrow encK$
21 $v \leftarrow$ KPKE.Enc$_2$(t, ρ, m)
22 Return v

Figure 6: Main components of ML-KEM Braid encapsulation algorithm, with their anamorphic extension.

encryption algorithm in the Fujisaki Okamoto (FO) transform [FO13]. Since m is normally sampled at random, we can replace it using the IND$-secure encryption algorithm SE.enc, using the same technique employed to substitute ρ. Moreover, to exploit all the available anamorphic bandwidth, the two parties will use shared randomness—the double state—to generate the necessary nonces locally.

The described embeddings are implemented at the lines 13 and 21 of Figure 7: the sender can use them only when in the correct state, namely when sampling the encapsulation key header (in the state *sndHdr*), or when sampling the ciphertext (in the state $sndCt_1$). When the sender exploits those embeddings, they will update the return value *len* accordingly (see lines 14,22)

Similarly, the receiver can decrypt the anamorphic message contained in the post-quantum embedding spot only after receiving all the necessary chunks, specifically when receiving the second part of the ciphertext (in the state $rcvCt_2$, line 38), and when receiving the encapsulation key header (in the state *rcvHdr*, line 60): when the receiver can reconstruct the ciphertext or the header —*i.e.* they received enough chunks— they can perform the decryption using the decryption algorithm SE.dec. A description of the modified structure of ML-KEM to host anamorphic messages is provided in Figure 6, where KPKE is the public-key encryption used to obtain the KEM applying the FO transform. We omit a description of the second part of the encapsulation algorithm KEM.enc$_2$ since it is identical to the normal ML-KEM algorithm.

Furthermore, to provide robustness, the SE encryption algorithm satisfies the SUF-CMA property. Consequently, one part of the anamorphic bandwidth is used to send a tag that authenticates the received anamorphic share. To provide authentication to more than one share, it is possible to authenticate both the Diffie-Hellman and Post-Quantum shares with the SE tag.

It is worth noticing that an anamorphic message can be split between a DH share and a post-quantum share; however,

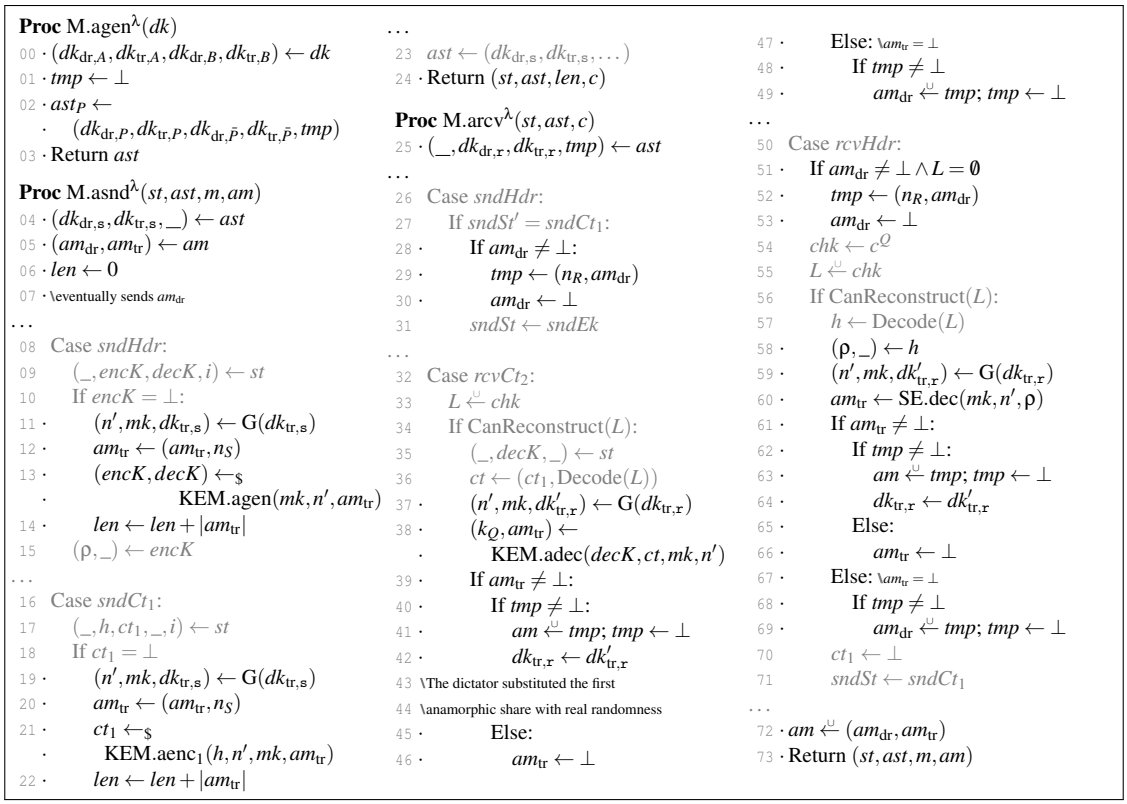

Figure 7: Anamorphic extension of the Triple Ratchet based on the anamorphic extension of ML-KEM Braid. Gray texts belong to the original protocol; black marks the anamorphic extension.

due to the chunking mechanism, the two parts of the anamorphic message will be received upon decryption of different ciphertexts. Hence, to verify the authenticity of the first share, the receiver can eventually keep in their *anamorphic state* a variable *tmp* to record such a share (lines 29,52). If, upon reception of the anamorphic message in the post-quantum share, *tmp* is not initialized, the receiver can detect a substitution attack by the dictator, hence rejecting am_{tr} (lines 45 and 45). To provide forward security, the content of *tmp* is erased once the whole anamorphic message is received (see lines 41 and 63).

A simplified overview of the additional operations carried out by the third ratchet is shown in Figure 17 in Appendix E, while an overview of the main changes implemented in the anamorphic extension for the Triple Ratchet protocol is shown in Figure 7.

Bandwidth During each post-quantum epoch, both the parties can send 256 bits of anamorphic messages, on top of the bandwidth provided from the Double Ratchet (see Section 4). Hence, we obtain a bandwidth of 16 bits for every epoch, plus 256 bits for every post-quantum epoch. However, 80 bits of the second share are used to send an authentication tag, reducing the post-quantum epoch bandwidth to 176 bits.

Security We prove correctness, anamorphic security and robustness of our TR construction as defined in Figures 1 to 3. The formal proofs are deferred to Appendix E.

Theorem 5 (Anamorphism)**.** *If* SE *is an* IND$ *and* SUF-CMA*-secure scheme,* F *and* G *are random oracles and* G_1 *is a secure PRG, then the anamorphic extension for the Triple Ratchet provides anamorphism, and in particular*

$$\begin{aligned}\mathrm{Adv}^{\mathrm{ANA},\delta}_{\mathrm{M}}(\mathcal{A}) \leq\ & \frac{2(q_{\mathrm{G_{ext}}} + q_{\mathrm{F_{ext}}})(q_{\mathrm{ASnd}} + q_{\mathrm{Chall}})}{|\mathcal{K}|} \\ &+ \frac{(q_{\mathrm{ASnd}} + q_{\mathrm{Chall}})^2}{p|\mathcal{N}||\mathcal{K}|} + \frac{(q_{\mathrm{ASnd}} + q_{\mathrm{Chall}})^2}{|\mathcal{K}||\mathcal{N}|} \\ &+ \frac{4(q_{\mathrm{ASnd}} + q_{\mathrm{Chall}})^2}{|\mathcal{K}|} + \mathrm{Adv}^{\text{suf-cma}}_{\mathrm{SE}}(\mathcal{B}) \\ &+ \frac{q_{\mathrm{ARcv}} 2^{\lambda}}{p} + q_{\mathrm{Chall}} \mathrm{Adv}^{\mathrm{prg}}_{\mathrm{G}_1}(\mathcal{C}) \\ &+ q_{\mathrm{Chall}} \mathrm{Adv}^{\text{ind\$}}_{\mathrm{SE}}(\mathcal{D})\end{aligned}$$

where q_{ASnd}, q_{Chall} *and* q_{ARcv} *are respectively the number of calls to the* ASnd, Chall *and* ARcv, $q_{\mathrm{G_{ext}}}$ *and* $q_{\mathrm{F_{ext}}}$ *are the number of query calls to the* G *and* F*, respectively, and* p *is the order of* g*. The parameter* δ *depends on the size of the chunks (for 32 bytes chunks, if the underlying KEM scheme is Kyber-1024,* $\delta = 49$*).*

Theorem 6 (Authenticity). *If* SE *is a* SUF-CMA*-secure scheme,* F *and* G *are random oracles, then the anamorphic extension for the Triple Ratchet provides authenticity, and in particular*

$$\begin{aligned}\mathrm{Adv}^{\mathrm{AUTH}}_{\mathrm{M}}(\mathcal{A}) \leq\ & \frac{(q_{\mathrm{G_{ext}}}+q_{\mathrm{F_{ext}}})2q_{\mathrm{ASnd}}}{|\mathcal{K}|} \\ & + \frac{4q^2_{\mathrm{ASnd}}}{|\mathcal{K}|} + \frac{q^2_{\mathrm{ASnd}}}{p|\mathcal{N}||\mathcal{K}|} + \frac{q^2_{\mathrm{ASnd}}}{|\mathcal{K}||\mathcal{N}|} \\ & + \frac{q_{\mathrm{ARcv}}2^{\lambda}}{p} + \mathrm{Adv}^{\text{suf-cma}}_{\mathrm{SE}}(\mathcal{B})\end{aligned}$$

where q_{ASnd} *and* q_{ARcv} *are respectively the number of calls to the* ASnd *and* ARcv*,* $q_{\mathrm{G_{ext}}}$ *and* $q_{\mathrm{F_{ext}}}$ *are the number of query calls to the* G *and* F*, respectively, and* p *is the order of* g*.*

6 iMessage PQ3

PQ3 is the ratcheting protocol employed in Apple's messenger *iMessage*. Analogously to the Triple Ratchet protocol, the PQ3 protocol combines an asymmetric Diffie-Hellman ratchet with a quantum-resistant KEM based ratchet; the KEM employed is the standardized version of ML-KEM [Nat24].

However, contrarily to the Triple Ratchet, PQ3 does not adopt a chunking strategy to limit the bandwidth usage of the post-quantum key updates, but rather limits their frequency: the two parties exchange new quantum-safe key material every 50 messages or every 7 days [LSB25]. The parties take turns in sending the encapsulation key and the ciphertext: if, at the epoch i, Alice sends the encapsulation key to Bob, then during the following epoch Bob responds with the corresponding KEM ciphertext. At this point both the parties update their key chains with the fresh post-quantum key material, starting a new post-quantum epoch. After 50 messages or one week, Bob samples a new KEM keypair and the ratcheting protocol is repeated with reversed roles.

Another difference between PQ3 and the Signal protocol regards the use of signatures: since Apple's protocol does not aim to offer *deniability*, it can rely on digital signatures to authenticate every message, with users signing their messages with a long-term identity key. An overview of the PQ3 protocol is provided in Figure 18, Appendix F, while a complete description and a security analysis of the protocol are given by Stebila [Ste24] and by Linker et al. [LSB25].

We identify three anamorphic embedding spots in PQ3. The first two are the same ones exploited in the Triple Ratchet construction: the Diffie-Hellman public key, as shown at line 13, and the ML-KEM spots—namely, the encapsulation key seed ρ and the random message m' in the FO transformed ciphertext, as shown at the lines 03 and 16 of Figure 6. The third hiding spot can be found in the digital signatures, using a well-known technique based on nonce recovery. Since PQ3 employs the ECDSA algorithm [Nat23, CMRR23], if both the parties know the signing key sk_σ, they can retrieve the nonce k used to obtain a signature (s,r) for a message m as

$$k \leftarrow s^{-1}(\mathrm{H}(m) + sk_\sigma r) \bmod q$$

where H is a hash function and q is the order of the elliptic curve. This is a known randomness-recovery technique that has been described by Kutylowski et al. in [KPP+23a].

We therefore embed an anamorphic message also in the nonce, using an IND$-secure encryption scheme SE and equipping the anamorphic state of the two users with the other party's long-term signing key, as shown at line 01 of Figure 8.

During a single execution of M.asnd, a user can embed anamorphic messages in up to three spots, depending on whether the message contains DH key material and ML-KEM key material. We note that all the anamorphic chunks are authenticated with a single authentication tag, hidden in the signature. Furthermore, to avoid reordering attacks, we generate keys, nonces and the contextualized exponent using a PRF F, inputted the index of the current message n_S (lines 07, 51).

An overview of the modifications to the protocol required by our anamorphic extension is presented in Figure 8.

Bandwidth We embed 16 bits of anamorphic message in the same spot used for the Double Ratchet; this spot can be exploited once every epoch. The usage of ML-KEM additionally provides 256 bits of bandwidth in each direction per post-quantum epoch, which occur approximately every 50 messages. Furthermore, embedding anamorphic messages in the signature yields additional 256 bits of bandwidth.

We assume that, for every call to M.asnd, 80 bits of the anamorphic bandwidth available in the signature are used to include an authentication tag. Our model captures this property authenticating also the other shares inputting them as authenticated data in the call to SE.enc at line 78.

Security We prove correctness, anamorphic security and robustness of our PQ3 construction as defined in Figures 1 to 3. The formal proofs are deferred to Appendix F.

Theorem 7 (Anamorphism). *If* G *and* F *are random oracles,* SE *is an* IND$ *and* SUF-CMA*-secure encryption scheme and* G_1 *is a secure PRG, the presented construction is anamorphic. In particular*

$$\begin{aligned}\mathrm{Adv}^{\mathrm{ANA}}_{\mathrm{M}}(\mathcal{A}) \leq\ & \frac{(q_{\mathrm{G_{ext}}}+q_{\mathrm{F_{ext}}})(q_{\mathrm{ASnd}}+q_{\mathrm{Chall}})}{|\mathcal{K}|} \\ & + \frac{2(q_{\mathrm{ASnd}}+q_{\mathrm{Chall}})^2}{|\mathcal{K}|} + \frac{(q_{\mathrm{ASnd}}+q_{\mathrm{Chall}})^2}{p|\mathcal{N}||\mathcal{K}|} \\ & + \frac{2(q_{\mathrm{ASnd}}+q_{\mathrm{Chall}})^2}{|\mathcal{N}||\mathcal{K}|} + \mathrm{Adv}^{\text{suf-cma}}_{\mathrm{SE}}(\mathcal{B}) \\ & + q_{\mathrm{Chall}}\mathrm{Adv}^{\mathrm{prg}}_{\mathrm{G_1}}(\mathcal{C}) + 2q_{\mathrm{Chall}}\mathrm{Adv}^{\mathrm{ind\$}}_{\mathrm{SE}}(\mathcal{D})\end{aligned}$$

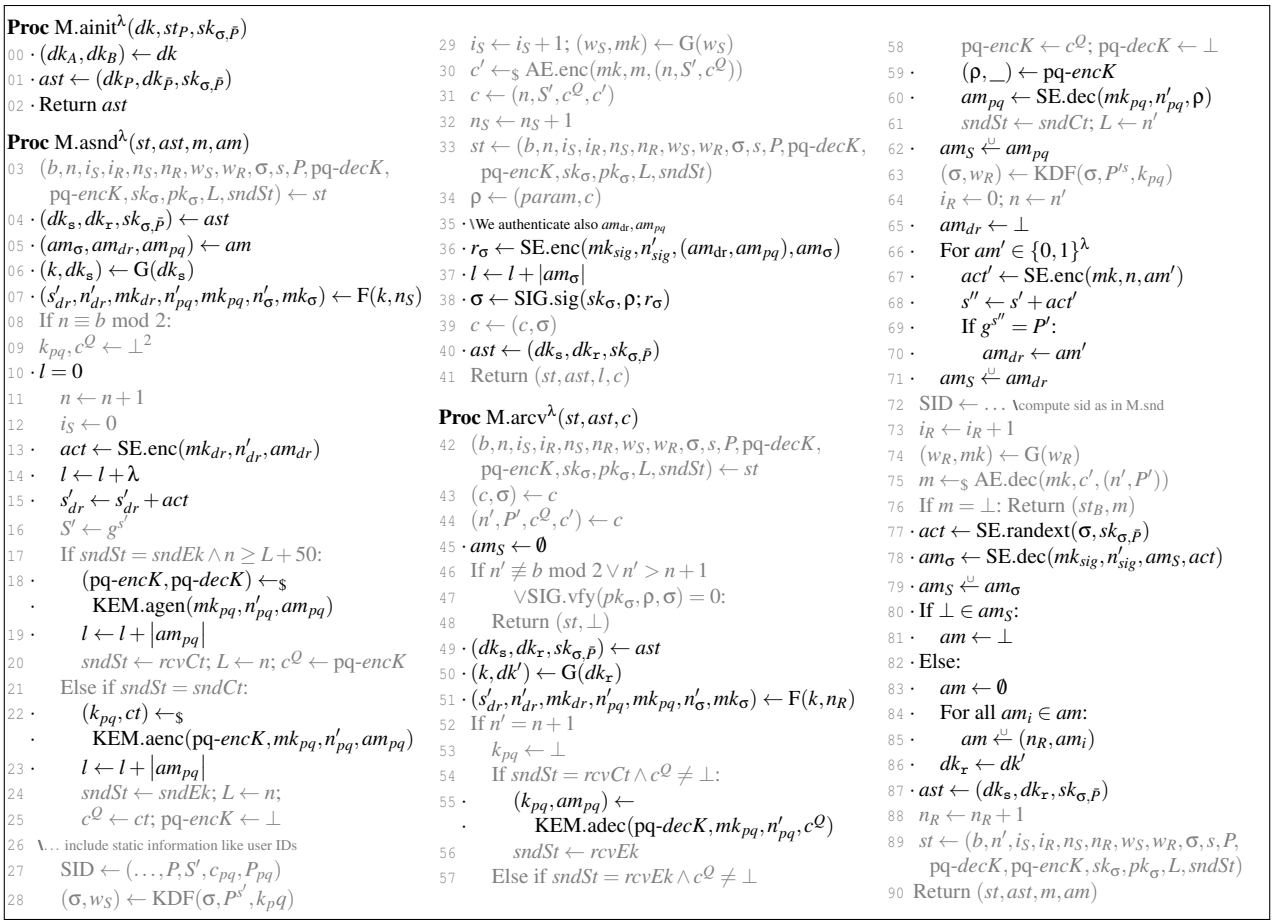

Figure 8: Anamorphic extension of the PQ3 algorithm. Gray texts belong to the original protocol; black marks the anamorphic extension.

where q_{ASnd} and q_{Chall} are respectively the number of calls to the ASnd *and* Chall *oracles, $q_{\mathrm{G}_{\mathrm{ext}}}$ and $q_{\mathrm{F}_{\mathrm{ext}}}$ are the number of query calls to the* G *and* F*, respectively, and p is the order of g.*

Theorem 8 (Authenticity)**.** *If* SE *is* SUF-CMA*-secure,* G *and* F *are random oracles and* G_1 *is a secure PRG, then the PQ3 anamorphic extension provides robustness. In particular, we have*

$$\mathrm{Adv}^{\mathrm{auth}}_{\mathrm{M}}(\mathcal{A}) \leq \frac{(q_{\mathrm{G}_{\mathrm{ext}}} + q_{\mathrm{F}_{\mathrm{ext}}}) q_{\mathrm{ASnd}}}{|\mathcal{K}|} + \frac{2q^2_{\mathrm{ASnd}}}{|\mathcal{K}|} + \frac{q^2_{\mathrm{ASnd}}}{p|\mathcal{K}||\mathcal{K}|} + \frac{2q^2_{\mathrm{ASnd}}}{|\mathcal{N}||\mathcal{K}|} + \mathrm{Adv}^{\text{suf-cma}}_{\mathrm{SE}}(\mathcal{B})$$

where q_{ASnd} is the number of calls to the ASnd *oracle, $q_{\mathrm{G}_{\mathrm{ext}}}$ and $q_{\mathrm{F}_{\mathrm{ext}}}$ are the number of query calls to the* G *and* F*, respectively, and p is the order of g.*

7 Message Layer Security

Message Layer Security (MLS) [BBR+23, CGWZ25] is a messaging protocol designed to scale for large groups. It consists of TreeKEM as a continuous group key exchange protocol to generate the shared group secret and a subsequent key schedule for message encryption. We present a simplified version of MLS for a static group of two users, in which the former subprotocol TreeKEM is built from an HPKE, a digital signature scheme, a KDF, and an AEAD scheme. Restricting the group to two members removes MLS's add and remove proposals and reduces TreeKEM's ratchet tree to a single root with two leaves, one per member. Each update then re-keys this shared root and sends the new secret to the partner via HPKE. Analyzing the full TreeKEM stack with dynamic membership remains future work. The key schedule receives a commit secret output from TreeKEM and a pre-shared key to update the epoch secret. This epoch secret is then used to derive the inputs for message encryption, including an encryption key and a nonce. The two-user version of MLS is detailed in Figure 19 in Appendix G.

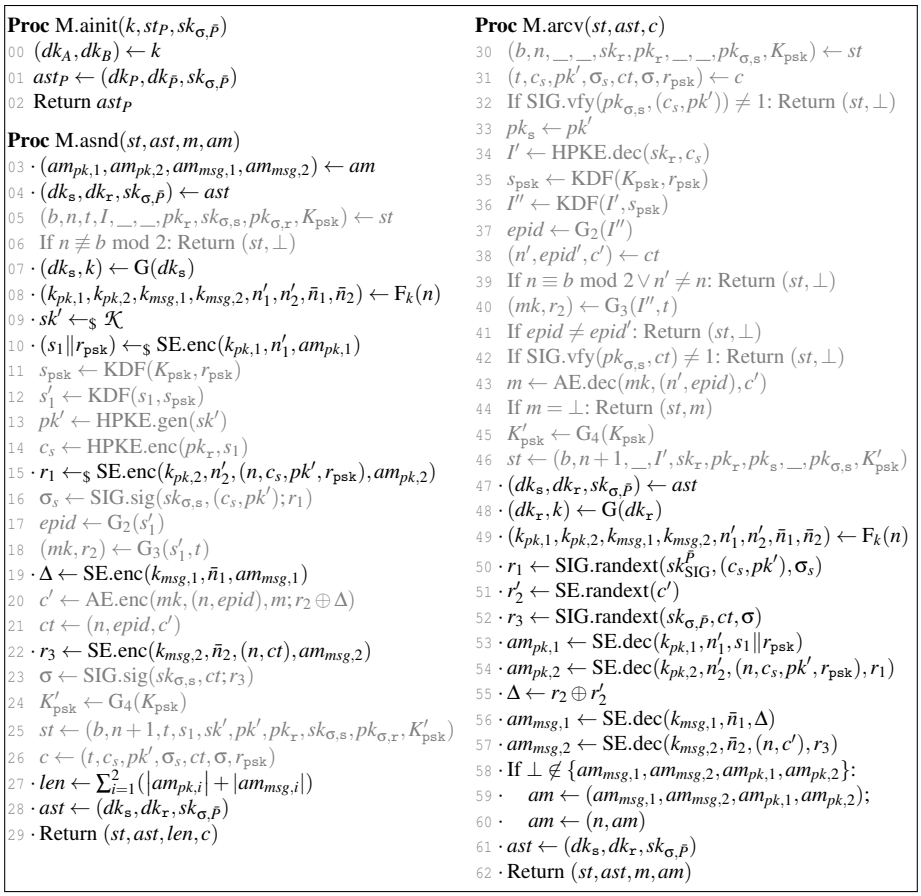

Figure 9: Anamorphic extension of two-party MLS. Gray texts belong to the original protocol; black marks the anamorphic extension.

Our anamorphic messaging construction for MLS leverages the following building blocks and specification guidelines:

1. The path secret s_1 is treated as a pseudorandom seed, which can be replaced by a random value (line 10, Figure 9), because MLS requires members to delete the seed used to derive the path secret immediately after use.

2. Every time a pre-shared key (PSK) is injected during the derivation of an epoch secret, the PSK nonce component in the PSK is a fresh 32-byte random value (for MLS-128 cipher suites). This is the second hiding spot as in line 10 of Figure 9.

3. The UpdatePath (containing the HPKE ciphertexts and the new public keys) is authenticated by a digital signature in line 16 of Figure 9. In MLS cipher suites using ECDSA signatures, the signature nonce can be a covert hiding spot as in line 15 of Figure 9.

4. To avoid nonce reuse, the actual nonce used in message encryption is xor-ed by a reuse guard, which contributes a 4-byte hiding spot as in line 19 of Figure 9.

5. The message encryption ciphertext is authenticated by the above digital signature, of which nonce is a hiding spot as in line 22 of Figure 9.

The two signature-nonce spots (items 3. and 5. listed above) are exploited via the same ECDSA nonce-recovery technique detailed for PQ3 in Section 6, equipping the anamorphic state with the partner's signing key. As in our other constructions, to avoid reordering attacks, the parties regenerate the relevant keys and nonces locally using a PRF on input message index n, as in line 08 and 49. Robustness is obtained as before: one SE share carries an 80-bit authentication tag (accounted for in the bandwidth below) that also authenticates the remaining shares, passed as associated data; thereby, any substitution by the dictator is detected and the anamorphic ciphertext rejected. Forward security, in turn, follows directly from MLS itself: members must delete the path secret immediately after use (the first spot), so a later exposure of the double state cannot reconstruct past epoch keys and previously sent anamorphic messages stay hidden. Finally, of the five spots only the reuse guard and the message signature nonce (items 4. and 5.) are available at every message, whereas the path secret, PSK nonce, and UpdatePath signature nonce (the first three) are

available once per epoch.

Bandwidth At each regular message transmission, both parties can send 256 bits and 32 bits of anamorphic messages through the nonce of ECDSA signature and the reuse guard value, respectively. At the beginning of each epoch, both parties obtain a bandwidth of 256 bits through the nonce of the ECDSA signature, a bandwidth of 256 bits through the path secret s_1 and a bandwidth of 256 bits through the PSK nonce. However, we estimate that 80 bits of the second share are used to send an authentication tag, reducing both the per-message bandwidth and the epoch bandwidth to 208 bits and 688 bits, respectively.

Analysis We prove correctness, anamorphism, robustness and authenticity of our MLS extension as defined in Figures 1 to 3.

We defer the formal proofs to Appendix G.

Theorem 9 (Anamorphism)**.** *If* G *and* F *are random oracles and* SE *is an* IND$ *secure encryption algorithm, then the anamorphic extension from Figure 9 provides anamorphic security, and, in particular, we have*

$$\begin{aligned}\mathrm{Adv}^{\mathrm{ana}}_{\mathrm{M}}(\mathcal{A}) \leq\ & \frac{(q_{\mathrm{G_{ext}}}+q_{\mathrm{F_{ext}}})(q_{\mathrm{ASnd}}+q_{\mathrm{Chall}})}{|\mathcal{K}|} \\ & + \frac{2(q_{\mathrm{ASnd}}+q_{\mathrm{Chall}})^2}{|\mathcal{K}|} + \frac{16(q_{\mathrm{ASnd}}+q_{\mathrm{Chall}})^2}{|\mathcal{N}||\mathcal{K}|} \\ & + 4(q_{\mathrm{ASnd}}+q_{\mathrm{Chall}})\mathrm{Adv}^{\mathrm{ind\$}}_{\mathrm{SE}}(\mathcal{C}) \\ & + 2\mathrm{Adv}^{\mathrm{suf\text{-}cma}}_{\mathrm{SE}}(\mathcal{E}),\end{aligned}$$

where q_{ASnd}, q_{Chall} *and* q_{ARcv} *are the number of query calls to the* ASnd, Chall *and* ARcv *oracles,* $q_{\mathrm{G_{ext}}}$ *and* $q_{\mathrm{F_{ext}}}$ *are the number of query calls to the external random oracles, respectively.*

Theorem 10 (Authenticity)**.** *If* G *and* F *are random oracles and* SE *is an* SUF-CMA *encryption algorithm, then our anamorphic extension from Figure 9 provides authenticity, and, in particular*

$$\begin{aligned}\mathrm{Adv}^{\mathrm{auth}}_{\mathrm{M}}(\mathcal{A}) \leq\ & \frac{(q_{\mathrm{G_{ext}}}+q_{\mathrm{F_{ext}}})q_{\mathrm{ASnd}}}{|\mathcal{K}|} + \frac{2q^2_{\mathrm{ASnd}}}{|\mathcal{K}|} + \frac{16q^2_{\mathrm{ASnd}}}{|\mathcal{N}||\mathcal{K}|} \\ & + 2\mathrm{Adv}^{\mathrm{suf\text{-}cma}}_{\mathrm{SE}}(\mathcal{E}),\end{aligned}$$

where q_{ASnd} *and* q_{ARcv} *are the number of query calls to the* ASnd *and* ARcv *oracles,* $q_{\mathrm{G_{ext}}}$ *and* $q_{\mathrm{F_{ext}}}$ *are the number of query calls to the external random oracles, respectively.*

8 Evaluation

Anamorphic extensions of messaging protocols are practical to deploy: one can modify existing messaging apps and distribute them via anti-surveillance channels such as Tor. Since their use cannot be distinguished from the honest app's use, this cannot be detected. We then discuss concrete instantiations of the SE building block and estimate the anamorphic bandwidth each protocol achieves, summarized in Table 1.

Selection of Encryption Schemes We employ two primitives of nonce-based symmetric encryption: one acts as a strong Pseudorandom Permutation (PRP) from plaintext to ciphertext used for bandwidth preserving, and the other is Authenticated Encryption with Associated Data (AEAD) for authentication purpose. The first primitive fits the SE building block used in the Double Ratchet extension with brute-force receiver in Figure 4. The second primitive fits the SE with associated data input in all remaining constructions. For the PRP, candidate schemes are FF1 [Dwo16] or a wide-block tweakable cipher like HCTR2 [Hel23]. For AEAD, instantiations, such as CBC composed with HMAC [Rog02], which is used in Signal, fit the requirement.

Bandwidth Estimate All the analyzed messaging protocols offer opportunities to embed anamorphic messages. We provide an overview of the respective bandwidths in Table 1. In the standard Double Ratchet, a party can embed 16 bits with practical efficiency for every epoch, and thus only at a limited frequency. In a scenario where only one of the parties is surveilled the other one can embed up to 176 bits per epoch.

The same hiding spot is present and used in the Triple Ratchet protocol and in PQ3; in addition, both schemes provide 256-bits hiding spots in the post-quantum ratchet. In Triple Ratchet, we reserve 80 of these 256 bits for an authentication tag, reducing the available bandwidth to 176 bits for every post-quantum epoch. Parties can then embed up to 192 bits in a single call to M.asnd. However, these bits are received at different times, since the receipt of the post-quantum material is delayed. Depending on the employed version of ML-KEM, up to 50 messages in each direction may be required between two post-quantum epochs.

In PQ3, anamorphic messages are additionally embedded in the signature scheme, which provides 256 extra bits per message. We, again, reserve 80 bits in every message to carry an authentication tag, reducing the available bandwidth to 176 bits. As in Triple Ratchet, parties in PQ3 refresh their post-quantum key material every 50 messages, or once every week, but in the best-case scenario, it is possible to embed 448 bits in total in a single ciphertext.

Bandwidth (bits)	Per message	Per classical epoch	Per post-quantum epoch	Best-case
Double Ratchet (Figure 4)	-	16	-	16
Double Ratchet (Figure 5)	-	176, 16 $^{(*)}$	-	176, 16 $^{(*)}$
Triple Ratchet (Figure 7)	-	16	176	192
PQ3 (Figure 8)	176	16	256	448
MLS (Figure 9)	208	688	-	896

Table 1: Bandwidths of anamorphic extensions for each studied messaging protocol (in bits). Values marked with $^{(*)}$ indicate directional bandwidth: Alice-to-Bob and Bob-to-Alice.

In MLS, we leverage the use of signatures and a nonce in every message to achieve a bandwidth of 288 bits. Furthermore, at the beginning of every epoch, parties can additionally exploit a bandwidth of 768 bits, obtained from a signature nonce and two 32 bytes protocol nonces. Since the two bandwidths are formally embedded in two channels, payload messages and control ciphertexts, we authenticate both of them with a separate authentication tag. Reserving 80 bits of each bandwidth results in 208 and 688 bits, respectively.

Acknowledgements

We are grateful to Davide Li Calsi and Dominique Schröder for insightful discussions during this project. Furthermore, we thank the reviewers for their valuable comments. Alessandro Corsi was supported by Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) as part of the Research and Training Group 2475 "Cybercrime and Forensic Computing" (grant number 393541319/GRK2475/2-2024).

References

[ABG+25] Gennaro Avitabile, Vincenzo Botta, Emanuele Giunta, Marcin Mielniczuk, and Francesco Migliaro. The malice of ELFs: Practical anamorphic-resistant encryption without random oracles. Cryptology ePrint Archive, Report 2025/305, 2025.

[ABH+21] Joël Alwen, Bruno Blanchet, Eduard Hauck, Eike Kiltz, Benjamin Lipp, and Doreen Riepel. Analysing the HPKE standard. In Anne Canteaut and François-Xavier Standaert, editors, *EUROCRYPT 2021, Part I*, volume 12696 of *LNCS*, pages 87–116. Springer, Cham, October 2021.

[ACD19] Joël Alwen, Sandro Coretti, and Yevgeniy Dodis. The double ratchet: Security notions, proofs, and modularization for the Signal protocol. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part I*, volume 11476 of *LNCS*, pages 129–158. Springer, Cham, May 2019.

[ACDT20] Joël Alwen, Sandro Coretti, Yevgeniy Dodis, and Yiannis Tselekounis. Security analysis and improvements for the IETF MLS standard for group messaging. In Daniele Micciancio and Thomas Ristenpart, editors, *CRYPTO 2020, Part I*, volume 12170 of *LNCS*, pages 248–277. Springer, Cham, August 2020.

[ACDT21] Joël Alwen, Sandro Coretti, Yevgeniy Dodis, and Yiannis Tselekounis. Modular design of secure group messaging protocols and the security of MLS. In Giovanni Vigna and Elaine Shi, editors, *ACM CCS 2021*, pages 1463–1483. ACM Press, November 2021.

[ACJM20] Joël Alwen, Sandro Coretti, Daniel Jost, and Marta Mularczyk. Continuous group key agreement with active security. In Rafael Pass and Krzysztof Pietrzak, editors, *TCC 2020, Part II*, volume 12551 of *LNCS*, pages 261–290. Springer, Cham, November 2020.

[ADGT25] Joël Alwen, Xiaohui Ding, Sanjam Garg, and Yiannis Tselekounis. Policy compliant secure messaging. In Goichiro Hanaoka and Bo-Yin Yang, editors, *ASIACRYPT 2025, Part II*, volume 16246 of *LNCS*, pages 34–67. Springer, Singapore, December 2025.

[ADJ+25] Benedikt Auerbach, Yevgeniy Dodis, Daniel Jost, Shuichi Katsumata, and Rolfe Schmidt. How to compare bandwidth constrained two-party secure messaging protocols: A quest for A more efficient and secure post-quantum protocol. In Lujo Bauer and Giancarlo Pellegrino, editors, *USENIX Security 2025*, pages 6717–6736. USENIX Association, August 2025.

[AJM22] Joël Alwen, Daniel Jost, and Marta Mularczyk. On the insider security of MLS. In Yevgeniy Dodis and Thomas Shrimpton, editors,

CRYPTO 2022, Part II, volume 13508 of *LNCS*, pages 34–68. Springer, Cham, August 2022.

[BBLW22] Richard Barnes, Karthikeyan Bhargavan, Benjamin Lipp, and Christopher A. Wood. Hybrid Public Key Encryption. RFC 9180, February 2022.

[BBR+23] Richard Barnes, Benjamin Beurdouche, Raphael Robert, Jon Millican, Emad Omara, and Katriel Cohn-Gordon. The Messaging Layer Security (MLS) Protocol. RFC 9420, July 2023.

[BDG+22] Alexander Bienstock, Yevgeniy Dodis, Sanjam Garg, Garrison Grogan, Mohammad Hajiabadi, and Paul Rösler. On the worst-case inefficiency of CGKA. In Eike Kiltz and Vinod Vaikuntanathan, editors, *TCC 2022, Part II*, volume 13748 of *LNCS*, pages 213–243. Springer, Cham, November 2022.

[BDR20] Alexander Bienstock, Yevgeniy Dodis, and Paul Rösler. On the price of concurrency in group ratcheting protocols. In Rafael Pass and Krzysztof Pietrzak, editors, *TCC 2020, Part II*, volume 12551 of *LNCS*, pages 198–228. Springer, Cham, November 2020.

[BGH+24] Fabio Banfi, Konstantin Gegier, Martin Hirt, Ueli Maurer, and Guilherme Rito. Anamorphic encryption, revisited. In Marc Joye and Gregor Leander, editors, *EUROCRYPT 2024, Part II*, volume 14652 of *LNCS*, pages 3–32. Springer, Cham, May 2024.

[BHKL13] Daniel J. Bernstein, Mike Hamburg, Anna Krasnova, and Tanja Lange. Elligator: elliptic-curve points indistinguishable from uniform random strings. In Ahmad-Reza Sadeghi, Virgil D. Gligor, and Moti Yung, editors, *ACM CCS 2013*, pages 967–980. ACM Press, November 2013.

[BM84] Manuel Blum and Silvio Micali. How to generate cryptographically strong sequences of pseudorandom bits. *SIAM Journal on Computing*, 13(4):850–864, 1984.

[BPL+25] Minhao Bai, Kaiyi Pang, Guorui Liao, Jinshuai Yang, and Yongfeng Huang. Shimmer: a provably secure steganography based on entropy collecting mechanism. In Lujo Bauer and Giancarlo Pellegrino, editors, *USENIX Security 2025*, pages 5949–5965. USENIX Association, August 2025.

[BPRS25] Shalini Banerjee, Tapas Pal, Andy Rupp, and Daniel Slamanig. Simple public key anamorphic encryption and signature using multi-message extensions. Cryptology ePrint Archive, Report 2025/370, 2025.

[BRT23] Alexander Bienstock, Paul Rösler, and Yi Tang. ASMesh: Anonymous and secure messaging in mesh networks using stronger, anonymous double ratchet. In Weizhi Meng, Christian Damsgaard Jensen, Cas Cremers, and Engin Kirda, editors, *ACM CCS 2023*, pages 1–15. ACM Press, November 2023.

[BRV20] Fatih Balli, Paul Rösler, and Serge Vaudenay. Determining the core primitive for optimally secure ratcheting. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part III*, volume 12493 of *LNCS*, pages 621–650. Springer, Cham, December 2020.

[BSJ+17] Mihir Bellare, Asha Camper Singh, Joseph Jaeger, Maya Nyayapati, and Igors Stepanovs. Ratcheted encryption and key exchange: The security of messaging. In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part III*, volume 10403 of *LNCS*, pages 619–650. Springer, Cham, August 2017.

[CCD+17] Katriel Cohn-Gordon, Cas Cremers, Benjamin Dowling, Luke Garratt, and Douglas Stebila. A formal security analysis of the Signal messaging protocol. In *2017 IEEE European Symposium on Security and Privacy*, pages 451–466. IEEE Computer Society Press, April 2017.

[CCD+20] Katriel Cohn-Gordon, Cas Cremers, Benjamin Dowling, Luke Garratt, and Douglas Stebila. A formal security analysis of the Signal messaging protocol. *Journal of Cryptology*, 33(4):1914–1983, October 2020.

[CCGM25] Davide Carnemolla, Dario Catalano, Emanuele Giunta, and Francesco Migliaro. Anamorphic resistant encryption: the good, the bad and the ugly. In Yael Tauman Kalai and Seny F. Kamara, editors, *CRYPTO 2025, Part III*, volume 16002 of *LNCS*, pages 472–503. Springer, Cham, August 2025.

[CCR26] Hien Chu, Alessandro Corsi, and Paul Rösler. Anamorphic messaging: Analyzing the Double Ratchet, Triple Ratchet, PQ3, and MLS. In *35th USENIX Security Symposium, USENIX Security 2026*. USENIX Association, 2026.

[CDHT25] Hien Chu, Khue Do, Lucjan Hanzlik, and Sri AravindaKrishnan Thyagarajan. When threshold meets anamorphic signatures: What is possible and what is not! Cryptology ePrint Archive, Report 2025/1044, 2025.

[CGM24a] Dario Catalano, Emanuele Giunta, and Francesco Migliaro. Anamorphic encryption: New constructions and homomorphic realizations. In Marc Joye and Gregor Leander, editors, *EUROCRYPT 2024, Part II*, volume 14652 of *LNCS*, pages 33–62. Springer, Cham, May 2024.

[CGM24b] Dario Catalano, Emanuele Giunta, and Francesco Migliaro. Limits of black-box anamorphic encryption. In Leonid Reyzin and Douglas Stebila, editors, *CRYPTO 2024, Part II*, volume 14921 of *LNCS*, pages 352–383. Springer, Cham, August 2024.

[CGM25] Dario Catalano, Emanuele Giunta, and Francesco Migliaro. Generic anamorphic encryption, revisited: New limitations and constructions. In Serge Fehr and Pierre-Alain Fouque, editors, *EUROCRYPT 2025, Part II*, volume 15602 of *LNCS*, pages 275–303. Springer, Cham, May 2025.

[CGWZ25] Cas Cremers, Esra Günsay, Vera Wesselkamp, and Mang Zhao. ETK: External-operations TreeKEM and the security of MLS in RFC 9420. Cryptology ePrint Archive, Report 2025/229, 2025.

[CGWZ26] Cas Cremers, Esra Günsay, Vera Wesselkamp, and Mang Zhao. ETK: External-operations TreeKEM and the security of MLS in RFC 9420. In Joan Daemen and Emmanuel Thomé, editors, *EUROCRYPT 2026, Part II*, volume 16542 of *LNCS*, pages 3–33. Springer, Cham, May 2026.

[CHH+17] Cas Cremers, Marko Horvat, Jonathan Hoyland, Sam Scott, and Thyla van der Merwe. A comprehensive symbolic analysis of TLS 1.3. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *ACM CCS 2017*, pages 1773–1788. ACM Press, October / November 2017.

[CKKW25] Adrian Cinal, Przemyslaw Kubiak, Miroslaw Kutylowski, and Gabriel Wechta. Anamorphic monero transactions: The threat of bypassing anti-money laundering laws. In Vincent Nicomette, Abdelmalek Benzekri, Nora Boulahia-Cuppens, and Jaideep Vaidya, editors, *ESORICS 2025, Part II*, volume 16054 of *LNCS*, pages 103–123. Springer, Cham, September 2025.

[CMRR23] Lily Chen, Dustin Moody, Andrew Regenscheid, and Karen Randall. Recommendations for Discrete Logarithm-Based Cryptography: Elliptic Curve Domain Parameters. NIST Special Publication 800-186, U.S. Department of Commerce, National Institute of Standards and Technology, February 2023. Companion document to FIPS 186-5; supersedes elliptic curve domain parameter specifications previously in FIPS 186-4 Appendix D.

[CR25] Daniel Collins and Paul Rösler. GURKE: Group unidirectional ratcheted key exchange. In Yael Tauman Kalai and Seny F. Kamara, editors, *CRYPTO 2025, Part VIII*, volume 16007 of *LNCS*, pages 75–108. Springer, Cham, August 2025.

[CSS23] Anrin Chakraborti, Darius Suciu, and Radu Sion. Wink: Deniable secure messaging. In Joseph A. Calandrino and Carmela Troncoso, editors, *USENIX Security 2023*, pages 1271–1288. USENIX Association, August 2023.

[DCW+23] Jinyang Ding, Kejiang Chen, Yaofei Wang, Na Zhao, Weiming Zhang, and Nenghai Yu. Discop: Provably secure steganography in practice based on "distribution copies". In *2023 IEEE Symposium on Security and Privacy*, pages 2238–2255. IEEE Computer Society Press, May 2023.

[DFGS21] Benjamin Dowling, Marc Fischlin, Felix Günther, and Douglas Stebila. A cryptographic analysis of the TLS 1.3 handshake protocol. *Journal of Cryptology*, 34(4):37, October 2021.

[DG25] Yevgeniy Dodis and Eli Goldin. Anamorphic-resistant encryption; or why the encryption debate is still alive. In Yael Tauman Kalai and Seny F. Kamara, editors, *CRYPTO 2025, Part III*, volume 16002 of *LNCS*, pages 440–471. Springer, Cham, August 2025.

[DHRR22] Benjamin Dowling, Eduard Hauck, Doreen Riepel, and Paul Rösler. Strongly anonymous ratcheted key exchange. In Shweta Agrawal and Dongdai Lin, editors, *ASIACRYPT 2022, Part III*, volume 13793 of *LNCS*, pages 119–150. Springer, Cham, December 2022.

[DJK+25] Yevgeniy Dodis, Daniel Jost, Shuichi Katsumata, Thomas Prest, and Rolfe Schmidt.

Triple ratchet: A bandwidth efficient hybrid-secure signal protocol. In Serge Fehr and Pierre-Alain Fouque, editors, *EUROCRYPT 2025, Part VIII*, volume 15608 of *LNCS*, pages 302–331. Springer, Cham, May 2025.

[DL25a] Amit Deo and Benoit Libert. Anamorphic signatures with dictator and recipient unforgeability for long messages. Cryptology ePrint Archive, Report 2025/2312, 2025.

[DL25b] Amit Deo and Benoît Libert. Anamorphic signatures with dictator and recipient unforgeability for long messages. In Goichiro Hanaoka and Bo-Yin Yang, editors, *ASIACRYPT 2025, Part VI*, volume 16250 of *LNCS*, pages 370–401. Springer, Singapore, December 2025.

[DMST25] Yevgeniy Dodis, Bernardo Magri, Noah Stephens-Davidowitz, and Yiannis Tselekounis. Guarding the signal: Secure messaging with reverse firewalls. In Yael Tauman Kalai and Seny F. Kamara, editors, *CRYPTO 2025, Part VIII*, volume 16007 of *LNCS*, pages 39–74. Springer, Cham, August 2025.

[DPPY25] Xuan Thanh Do, Giuseppe Persiano, Duong Hieu Phan, and Moti Yung. Anamorphism beyond one-to-one messaging: Public-key with anamorphic broadcast mode. In Serge Fehr and Pierre-Alain Fouque, editors, *EUROCRYPT 2025, Part III*, volume 15603 of *LNCS*, pages 429–455. Springer, Cham, May 2025.

[DRS20] Benjamin Dowling, Paul Rösler, and Jörg Schwenk. Flexible authenticated and confidential channel establishment (fACCE): Analyzing the noise protocol framework. In Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas, editors, *PKC 2020, Part I*, volume 12110 of *LNCS*, pages 341–373. Springer, Cham, May 2020.

[Dwo16] Morris J. Dworkin. Recommendation for block cipher modes of operation: Methods for format-preserving encryption. Technical Report NIST Special Publication (SP) 800-38G, National Institute of Standards and Technology (NIST), 2016.

[Eur25] European Commission. Proposal for a REGULATION OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL laying down rules to prevent and combat child sexual abuse (text with EEA relevance). Interinstitutional File 2022/0155(COD), Version for Trilogue on December 9, 2025, 2025. Original proposal: COM(2022) 209 final.

[FO13] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. *Journal of Cryptology*, 26(1):80–101, January 2013.

[GC25] SAYANTAN GANGULY and Shion Samadder Chaudhury. Computational quantum anamorphic encryption and anamorphic secret sharing. Cryptology ePrint Archive, Report 2025/399, 2025.

[GHS+20] Guillaume Girol, Lucca Hirschi, Ralf Sasse, Dennis Jackson, Cas Cremers, and David A. Basin. A spectral analysis of noise: A comprehensive, automated, formal analysis of Diffie-Hellman protocols. In Srdjan Capkun and Franziska Roesner, editors, *USENIX Security 2020*, pages 1857–1874. USENIX Association, August 2020.

[GRL25] Rosario Giustolisi, Mohammadamin Rakeei, and Gabriele Lenzini. Anamorphic voting: Ballot freedom against dishonest authorities. Cryptology ePrint Archive, Report 2025/647, 2025.

[Hel23] Nathan Helms. Length-preserving encryption with HCTR2. In *Third Workshop on Block Cipher Modes of Operation*. National Institute of Standards and Technology (NIST), 2023.

[JBK24] Tushar M. Jois, Gabrielle Beck, and Gabriel Kaptchuk. Pulsar: Secure steganography for diffusion models. In Bo Luo, Xiaojing Liao, Jun Xu, Engin Kirda, and David Lie, editors, *ACM CCS 2024*, pages 4703–4717. ACM Press, October 2024.

[JS18] Joseph Jaeger and Igors Stepanovs. Optimal channel security against fine-grained state compromise: The safety of messaging. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part I*, volume 10991 of *LNCS*, pages 33–62. Springer, Cham, August 2018.

[JS24] Joseph Jaeger and Roy Stracovsky. Dictators? Friends? Forgers. - breaking and fixing unforgeability definitions for anamorphic signature schemes. In Kai-Min Chung and Yu Sasaki, editors, *ASIACRYPT 2024, Part II*, volume 15485 of *LNCS*, pages 105–137. Springer, Singapore, December 2024.

[KJGR21] Gabriel Kaptchuk, Tushar M. Jois, Matthew Green, and Aviel D. Rubin. Meteor: Cryptographically secure steganography for realistic

distributions. In Giovanni Vigna and Elaine Shi, editors, *ACM CCS 2021*, pages 1529–1548. ACM Press, November 2021.

[KNB19] Nadim Kobeissi, Georgio Nicolas, and Karthikeyan Bhargavan. Noise explorer: Fully automated modeling and verification for arbitrary noise protocols. In *2019 IEEE European Symposium on Security and Privacy*, pages 356–370. IEEE Computer Society Press, June 2019.

[KPP+23a] Miroslaw Kutylowski, Giuseppe Persiano, Duong Hieu Phan, Moti Yung, and Marcin Zawada. Anamorphic signatures: Secrecy from a dictator who only permits authentication! In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part II*, volume 14082 of *LNCS*, pages 759–790. Springer, Cham, August 2023.

[KPP+23b] Miroslaw Kutylowski, Giuseppe Persiano, Duong Hieu Phan, Moti Yung, and Marcin Zawada. The self-anti-censorship nature of encryption: On the prevalence of anamorphic cryptography. *Proc. Priv. Enhancing Technol.*, 2023(4):170–183, 2023.

[KPPW+21] Karen Klein, Guillermo Pascual-Perez, Michael Walter, Chethan Kamath, Margarita Capretto, Miguel Cueto, Ilia Markov, Michelle Yeo, Joël Alwen, and Krzysztof Pietrzak. Keep the dirt: Tainted TreeKEM, adaptively and actively secure continuous group key agreement. In *2021 IEEE Symposium on Security and Privacy*, pages 268–284. IEEE Computer Society Press, May 2021.

[LBP26] Daji Landis, Elettra Bietti, and Sunoo Park. "security vs. interoperability" arguments: An analytical framework, 2026.

[LGGR26] Julia Len, Esha Ghosh, Paul Grubbs, and Paul Rösler. Interoperability in end-to-end encrypted messaging. In *Proceedings of the Symposium on Computer Science and Law*, pages 15–25, 2026.

[LJ25] Man Luo and Zhe Jiang. Anamorphic private set intersection. In *2025 2nd International Conference on Algorithms, Software Engineering and Network Security (ASENS)*, pages 314–319, 2025.

[LSB25] Felix Linker, Ralf Sasse, and David A. Basin. A formal analysis of apple's iMessage PQ3 protocol. In Lujo Bauer and Giancarlo Pellegrino, editors, *USENIX Security 2025*, pages 5015–5034. USENIX Association, August 2025.

[Lun18] Joshua Lund. Technology preview: Sealed sender for signal. https://signal.org/blog/sealed-sender/, 10 2018.

[LYSH25] Guorui Liao, Jinshuai Yang, Weizhi Shao, and Yongfeng Huang. A framework for designing provably secure steganography. In Lujo Bauer and Giancarlo Pellegrino, editors, *USENIX Security 2025*, pages 6837–6856. USENIX Association, August 2025.

[MKA+21] Ian Martiny, Gabriel Kaptchuk, Adam J. Aviv, Daniel S. Roche, and Eric Wustrow. Improving Signal's sealed sender. In *NDSS 2021*. The Internet Society, February 2021.

[Nat23] National Institute of Standards and Technology. Digital Signature Standard (DSS). Federal Information Processing Standards Publication FIPS PUB 186-5, U.S. Department of Commerce, National Institute of Standards and Technology, February 2023. Supersedes FIPS 186-4.

[Nat24] National Institute of Standards and Technology. Module-lattice-based key-encapsulation mechanism standard. Federal Information Processing Standards Publication NIST FIPS 203, National Institute of Standards and Technology, August 2024.

[PM16] Trevor Perrin and Moxie Marlinspike. The double ratchet algorithm. https://signal.org/docs/specifications/doubleratchet/doubleratchet.pdf, 11 2016.

[PPY22] Giuseppe Persiano, Duong Hieu Phan, and Moti Yung. Anamorphic encryption: Private communication against a dictator. In Orr Dunkelman and Stefan Dziembowski, editors, *EUROCRYPT 2022, Part II*, volume 13276 of *LNCS*, pages 34–63. Springer, Cham, May / June 2022.

[PPY24] Giuseppe Persiano, Duong Hieu Phan, and Moti Yung. Public-key anamorphism in (CCA-secure) public-key encryption and beyond. In Leonid Reyzin and Douglas Stebila, editors, *CRYPTO 2024, Part II*, volume 14921 of *LNCS*, pages 422–455. Springer, Cham, August 2024.

[PR18a] Bertram Poettering and Paul Rösler. Asynchronous ratcheted key exchange. Cryptology ePrint Archive, Report 2018/296, 2018.

[PR18b] Bertram Poettering and Paul Rösler. Towards bidirectional ratcheted key exchange. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part I*, volume 10991 of *LNCS*, pages 3–32. Springer, Cham, August 2018.

[PRSS21] Bertram Poettering, Paul Rösler, Jörg Schwenk, and Douglas Stebila. SoK: Game-based security models for group key exchange. In Kenneth G. Paterson, editor, *CT-RSA 2021*, volume 12704 of *LNCS*, pages 148–176. Springer, Cham, May 2021.

[RGR+25] Mohammadamin Rakeei, Rosario Giustolisi, Andy Rupp, Chuanwei Lin, and Gabriele Lenzini. Crypto wars in secure messaging: Covert channels in signal despite leaked keys. Cryptology ePrint Archive, Report 2025/2172, 2025.

[RMS18] Paul Rösler, Christian Mainka, and Jörg Schwenk. 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*, pages 415–429. IEEE Computer Society Press, April 2018.

[Rog02] Phillip Rogaway. Authenticated-encryption with associated-data. In Vijayalakshmi Atluri, editor, *ACM CCS 2002*, pages 98–107. ACM Press, November 2002.

[RS23] Paul Rösler and Jörg Schwenk. Interoperability between messaging services secure–implementation of encryption, 2023.

[Sch] Rolfe Schmidt. The ML-KEM Braid Protocol. https://signal.org/docs/specifications/mlkembraid/#representation-of-epochs. Accessed: 2025-11-20.

[Ste24] Douglas Stebila. Security analysis of the iMessage PQ3 protocol. Cryptology ePrint Archive, Report 2024/357, 2024.

[TRB+25] Lea Thiemt, Paul Rösler, Alexander Bienstock, Rolfe Schmidt, and Yevgeniy Dodis. Generic anonymity wrapper for messaging protocols. In Chun-Ying Huang, Jyh-Cheng Chen, Shiuh-Pyng Shieh, David Lie, and Véronique Cortier, editors, *ACM CCS 2025*, pages 350–364. ACM Press, October 2025.

[UK 23] UK Parliament. Online safety act 2023. 2023 c. 50, 2023.

[US 23] US Congress. Eliminating abusive and rampant neglect of interactive technologies (EARN IT) act of 2023. S. 3538, 118th Congress, 2023.

[WCHY23] Yi Wang, Rongmao Chen, Xinyi Huang, and Moti Yung. Sender-anamorphic encryption reformulated: Achieving robust and generic constructions. In Jian Guo and Ron Steinfeld, editors, *ASIACRYPT 2023, Part VI*, volume 14443 of *LNCS*, pages 135–167. Springer, Singapore, December 2023.

[WHL24] Weihao Wang, Shuai Han, and Shengli Liu. Anamorphic authenticated key exchange: Double key distribution under surveillance. In Kai-Min Chung and Yu Sasaki, editors, *ASIACRYPT 2024, Part V*, volume 15488 of *LNCS*, pages 168–200. Springer, Singapore, December 2024.

[WPB25] Théophile Wallez, Jonathan Protzenko, and Karthikeyan Bhargavan. TreeKEM: A modular machine-checked symbolic security analysis of group key agreement in messaging layer security. In Marina Blanton, William Enck, and Cristina Nita-Rotaru, editors, *2025 IEEE Symposium on Security and Privacy*, pages 4375–4390. IEEE Computer Society Press, May 2025.

[WPBB23] Théophile Wallez, Jonathan Protzenko, Benjamin Beurdouche, and Karthikeyan Bhargavan. TreeSync: Authenticated group management for messaging layer security. In Joseph A. Calandrino and Carmela Troncoso, editors, *USENIX Security 2023*, pages 1217–1233. USENIX Association, August 2023.

[WPC+25] Yaofei Wang, Gang Pei, Kejiang Chen, Jinyang Ding, Chao Pan, Weilong Pang, Donghui Hu, and Weiming Zhang. SparSamp: Efficient provably secure steganography based on sparse sampling. In Lujo Bauer and Giancarlo Pellegrino, editors, *USENIX Security 2025*, pages 6817–6835. USENIX Association, August 2025.

[Yao82] Andrew Chi-Chih Yao. Theory and applications of trapdoor functions (extended abstract). In *23rd FOCS*, pages 80–91. IEEE Computer Society Press, November 1982.

A Further Preliminaries

A.1 Symmetric Encryption

A Symmetric Encryption scheme is a tuple of algorithms $\mathrm{SE} = (\mathrm{SE.gen}, \mathrm{SE.enc}, \mathrm{SE.dec})$ with the following syntax:

- $\mathrm{SE.gen} : \emptyset \to_{\$} \mathcal{K}$
- $\mathrm{SE.enc} : \mathcal{K} \times \mathcal{N} \times (\mathcal{AD} \cup \{\perp\}) \times \mathcal{M} \to \mathcal{C}$

- SE.dec : $\mathcal{K} \times \mathcal{N} \times (\mathcal{AD} \cup \{\bot\}) \times \mathcal{C} \to \mathcal{M}$

With this syntax, we optionally cover Authenticated Encryption with Associated Data (AEAD). Concretely, depending on which of the following two security definitions is fulfilled, we either (only) define a strong pseudorandom permutation or, if multi-instance strong unforgeability is provided, full AEAD. We define security for Symmetric Encryption via games IND$ and SUF-CMA in figures 10 and 11, respectively, where the advantage in winning the games is defined as:

$$\mathrm{Adv}^{\mathrm{ind\$}}_{\mathrm{SE}}(\mathcal{A}) := \left|\Pr[\mathrm{IND\$}^1_{\mathrm{SE}}(\mathcal{A}) = 1] - \Pr[\mathrm{IND\$}^0_{\mathrm{SE}}(\mathcal{A}) = 1]\right|$$

$$\text{and } \mathrm{Adv}^{\text{suf-cma}}_{\mathrm{SE}}(\mathcal{A}) := \Pr[\mathrm{SUF\text{-}CMA}_{\mathrm{SE}}(\mathcal{A}) = 1].$$

Game $\mathrm{IND\$}^b_{\mathrm{SE}}(\mathcal{A})$
00 $CH, C \leftarrow \emptyset^2$
01 $k \leftarrow_\$ \mathrm{SE.gen}$
02 $b' \leftarrow_\$ \mathcal{A}$
03 Stop with b'

Oracle $\mathrm{Enc}(n, ad, m)$
04 Require $\nexists (n, ad, m, \cdot) \in CH$
05 $c \leftarrow \mathrm{SE.enc}(k, n, ad, m)$
06 $C \xleftarrow{\cup} (n, ad, m, c)$
07 Return c

Oracle $\mathrm{Chall}(n, ad, m)$
08 Require $m \in \mathcal{M}$
09 Require $\nexists (n, ad, m, \cdot) \in CH \cup C$
10 $c_0 \leftarrow \mathrm{SE.enc}(k, n, ad, m)$
11 $c_1 \leftarrow_\$ \{0,1\}^{|c_0|}$
12 $CH \xleftarrow{\cup} (n, ad, m, c_b)$
13 Return c_b

Oracle $\mathrm{Dec}(n, ad, c)$
14 Require $\nexists (n, ad, \cdot, c) \in CH$
15 $m \leftarrow \mathrm{SE.dec}(k, n, ad, c)$
16 Return m

Figure 10: Indistinguishability of ciphertexts under chosen ciphertext attacks for symmetric encryption.

Game $\mathrm{SUF\text{-}CMA}_{\mathrm{SE}}(\mathcal{A})$
00 $s \leftarrow 0$
01 $MC \leftarrow \emptyset$
02 Invoke $\mathcal{A}$
03 Stop with 0

Oracle Gen
04 $k \leftarrow_\$ \mathrm{SE.gen}$
05 $s \leftarrow s + 1$

Oracle $\mathrm{Corrupt}(i)$
06 $CR \xleftarrow{\cup} \{i\}$
07 Return k_i

Oracle $\mathrm{Enc}(i, n, ad, m)$
08 $c \leftarrow \mathrm{SE.enc}(k_i, n, ad, m)$
09 $MC \xleftarrow{\cup} \{(i, n, ad, m, c)\}$
10 Return c

Oracle $\mathrm{Dec}(i, n, ad, c)$
11 $m \leftarrow \mathrm{SE.dec}(k_i, n, ad, c)$
12 If $(i, n, ad, m, c) \notin MC$
$\quad \wedge m \neq \bot \wedge i \notin CR$:
13 Stop with 1
14 Return m

Figure 11: Strong unforgeability for symmetric encryption.

A.2 Pseudorandom Generator

A secure pseudorandom generator [Yao82, BM84] is an algorithm $\mathrm{G} : \{0,1\}^n \to \{0,1\}^m$. We define the advantage of an adversary $\mathcal{A}$ in breaking the security of a pseudorandom generator G as

$$\mathrm{Adv}^{\mathrm{prg}}_{\mathrm{G}}(\mathcal{A}) := \Big|\Pr[\mathcal{A}(\mathrm{G}(s)) : s \leftarrow_\$ \{0,1\}^n] - \Pr[\mathcal{A}(r) : r \leftarrow_\$ \{0,1\}^m]\Big|.$$

B Further Definitions for Anamorphic Messaging

As motivated in Section 3.3, anamorphic messaging can (also) be useful for communication between one user who is heavily surveilled and another one who is outside the control of the dictator. In this latter case, the adversary may only learn the regular messaging state of the surveilled party, whereas the regular messaging state of the other party remains uncorrupted.

We capture this notion of *partial surveillance* under *partially active adversaries* formally in the game $\mathrm{PANA}^{b,\delta}$ in Figure 12. The difference to our main game from Figure 2 is that, in Figure 12, the return values of oracles Snd, ASnd, Chall, and ARcv differ depending on whether they are queried for Alice or Bob: if Alice is queried to send or receive, only the resulting ciphertext, resp. message, is given to the adversary; otherwise, the oracle's output also includes Bob's current regular messaging state. Furthermore, ciphertexts received by Alice are required to be delivered honestly.

Game $\mathrm{PANA}^{b,\delta}_{\mathrm{M}}(\mathcal{A})$
00 $(st_A, st_B) \leftarrow \mathrm{M.init}(\rho)$
01 $k \leftarrow_\$ \mathcal{K}$
02 For $P \in \{A, B\}$:
03 $\quad ast_P \leftarrow_\$ \mathrm{M.ainit}(k, st_P, \mathcal{X}(st_{\bar{P}}))$
04 $\quad s_P, r_P \leftarrow 0^2$; $CH_P \leftarrow \emptyset$
05 $\quad X_P \leftarrow \mathtt{fal}$; $C_P[\cdot] \leftarrow \bot$
06 $b' \leftarrow_\$ \mathcal{A}(st_B)$
07 Stop with b'

Oracle $\mathrm{Snd}(P, m)$
08 $(st_P, c) \leftarrow_\$ \mathrm{M.snd}(st_P, m)$
09 $C_P[s_P] \leftarrow c$
10 $s_P \leftarrow s_P + 1$
11 If $P = A$: Return c
12 Return (st_P, c)

Oracle $\mathrm{ASnd}(P, m, am)$
13 $(st_P, ast_P, l, c) \leftarrow_\$$
$\quad \mathrm{M.asnd}(st_P, ast_P, m, am)$
14 $C_P[s_P] \leftarrow c$
15 $s_P \leftarrow s_P + 1$
16 If $P = A$: Return (l, c)
17 Return (st_P, l, c)

Oracle Expose(P)
18 Require $\nexists s' \in CH_{\bar{P}} : s' + \delta > r_P$
19 $X_P \leftarrow \mathtt{tru}$; $X_{\bar{P}} \leftarrow \mathtt{tru}$
20 Return ast_P

Oracle $\mathrm{Chall}(P, m, am)$
21 Require $X_{\bar{P}} = \mathtt{fal}$
22 If $b = 0$:
23 $\quad (st_P, c) \leftarrow_\$ \mathrm{M.snd}(st_P, m)$
24 Else:
25 $\quad (st_P, ast_P, l, c) \leftarrow_\$$
$\quad\quad \mathrm{M.asnd}(st_P, ast_P, m, am)$
26 $C_P[s_P] \leftarrow c$
27 $CH_P \xleftarrow{\cup} \{s_P\}$
28 $s_P \leftarrow s_P + 1$
29 If $P = A$: Return c
30 Return (st_P, c)

Oracle $\mathrm{ARcv}(P, c)$
31 If $P = A$: Require $C_B[r_P] = c$
32 $(st_P, ast_P, m, am) \leftarrow$
$\quad\quad \mathrm{M.arcv}(st_P, ast_P, k, c)$
33 For all $(r', am') \in am$:
34 $\quad$ If $r' \in CH_{\bar{P}}$:
35 $\quad\quad am \leftarrow am \setminus \{(r', am')\}$
36 If $m \neq \bot$: $r_P \leftarrow r_P + 1$
37 If $P = A$: Return (m, am)
38 Return (st_P, m, am)

Figure 12: Anamorphism for Messaging with Forward Secure Continuous State Updates, Partial Regular State Exposure, and Partially Active Adversaries. Violet code differs from the game in Figure 2: Bob is considered to be under control and surveillance of the dictator, whereas Alice's regular messaging state is assumed to be hidden from the dictator; furthermore, ciphertexts received by Alice are assumed to be honestly transmitted.

C Security analysis of the first Double Ratchet extension

This section establishes the security of the scheme presented in Section 4, by showing that it satisfies the correctness, anamorphism, and authenticity. This result consists of the following Theorems 1, 2 and 11.

Theorem 11 (Correctness)**.** *The anamorphic extension of Figure 4 is correct.*

Correctness of our construction follows directly from the correctness of the underlying messaging scheme.

Proof of Theorem 1. We prove the anamorphism of the anamorphic extension using a sequence of game hops.

Game G_0 to **Game** G_4 are exactly the same used in the AUTH game of Theorem 2, with the difference that the random oracle bookkeeping happens also for calls to Chall.

The random oracles interfaces are the one presented in the AUTH game, in Figure 14.

In G_4, all the Diffie Hellman public keys are uniformly random, and the best strategy for the adversary is a random guess.

Therefore, the advantage bound from G_0 to G_4 is:

$$\begin{aligned}\mathrm{Adv}_{G_0}(\mathcal{A}) \leq \mathrm{Adv}_{G_5}(\mathcal{A}) &+ \frac{(q_{\mathrm{G_{ext}}}+q_{\mathrm{F_{ext}}})(q_{\mathrm{ASnd}}+q_{\mathrm{Chall}})}{|\mathcal{K}|} \\ &+ \frac{(q_{\mathrm{ASnd}}+q_{\mathrm{Chall}})^2}{|\mathcal{K}|} + \frac{(q_{\mathrm{ASnd}}+q_{\mathrm{Chall}})^2}{p|\mathcal{N}||\mathcal{K}|} \\ &+ \frac{q_{\mathrm{ARcv}}2^{\lambda}}{p}.\end{aligned}$$

In G_5 we don't have anymore authenticity violations, hence the adversary cannot infer from desynchronized states upon corruption whether the previous challenges were obtained calling M.asnd or M.snd.

To provide an intuition of why this matters, consider the case in which $b = 0$ (i.e., the challenges are obtained with a call to the non-anamorphic M.snd) and one of the challenges gets decrypted anamorphically in Rcv: in such a case, upon corruption of both the sender and the receiver, the adversary would obtain desynchronized states, inferring $b' = 0$.

Furthermore, since we use the random oracle to obtain s', in case of anamorphic encryption (in ASnd and Chall), the exponent of the public key is random, and in case of uncorrupted user the public key is indistinguishable from one that is obtained as g^s, where $s \leftarrow \mathbb{Z}_p$.

Since in G_5 there are no authenticity violations and normal ciphertexts are indistinguishable from anamorphic ciphertexts, the best strategy for the adversary is random guessing, and $\mathrm{Adv}_{G_5}(\mathcal{A}) = 0$. □

Proof of Theorem 2. We prove the result using a sequence of game hops presented in Figure 13.

Game G_0: Is the initial game

$$\mathrm{Adv}_{G_0}(\mathcal{A}) = \mathrm{Adv}^{\mathrm{AUTH}}_{\mathrm{M}}(\mathcal{A}).$$

The simulation performs bookkeeping, memorizing in two tables which random oracle calls have been made by the adversary and which by the simulation itself (i.e., as a result of a call to ASnd).

These two tables are maintained for both the random oracles G and F, with the two interfaces presented in Figure 14. Hence, it will keep $\mathrm{G_{ext}}$ and $\mathrm{G_{int}}$, $\mathrm{F_{ext}}$ and $\mathrm{F_{int}}$. When deriving a double key in ASnd, if the adversary didn't expose the key material, the simulation calls $\mathrm{G_{int}}$ with additional, bookkeeping information, i.e., the party P and the index of the call n_S. Since this two additional information, together with the double key dk, uniquely identifies an internal call to the ASnd oracle, it is sufficient to retroactively program the random oracles after an exposure.

After an exposure, the simulation retroactively programs the random oracle to coherently output the double keys dk: that is, the simulation can program the random oracle to output the values obtained with the internal queries on the external interface.

We use lazy sampling for G and F.

Game G_1: The first game is obtained by excluding the case that the adversary queries its interface $\mathrm{G_{ext}}$ or $\mathrm{F_{ext}}$ with one of the values queried by the simulation to $\mathrm{G_{int}}$ and $\mathrm{F_{int}}$. If this happens, the adversary could be able to break the coherence of the simulation.

We bound the advantage with

$$\mathrm{Adv}_{G_0}(\mathcal{A}) \leq \mathrm{Adv}_{G_1}(\mathcal{A}) + \frac{q_{\mathrm{G_{ext}}}q_{\mathrm{ASnd}}}{|\mathcal{K}|} + \frac{q_{\mathrm{F_{ext}}}q_{\mathrm{ASnd}}}{|\mathcal{K}|}.$$

Game G_2: Before an exposure, the simulation samples dk inputting additional information to the RO G, as specified before.

We exclude the possibility of collisions on the hash chain generated by $\mathrm{G_{int}}$: to obtain this bound we can use the birthday bound, accounting for q_{ASnd} queries to the ASnd oracle. Furthermore, we exclude the possibility of collisions on the F key, to avoid reordering attacks.

$$\mathrm{Adv}_{G_1}(\mathcal{A}) \leq \mathrm{Adv}_{G_2}(\mathcal{A}) + \frac{q^2_{\mathrm{ASnd}}}{|\mathcal{K}|}.$$

Game G_3: We abort in case of a collision on the exponent s', nonce n and encryption key mk outputted by F. This step is necessary to avoid simple reordering attacks.

$$\mathrm{Adv}_{G_2}(\mathcal{A}) = \mathrm{Adv}_{G_3}(\mathcal{A}) + \frac{q^2_{\mathrm{ASnd}}}{p|\mathcal{N}||\mathcal{K}|}.$$

Game G_4: We exclude the authenticity violation in the ARcv query via a statistical argument. To break authenticity,

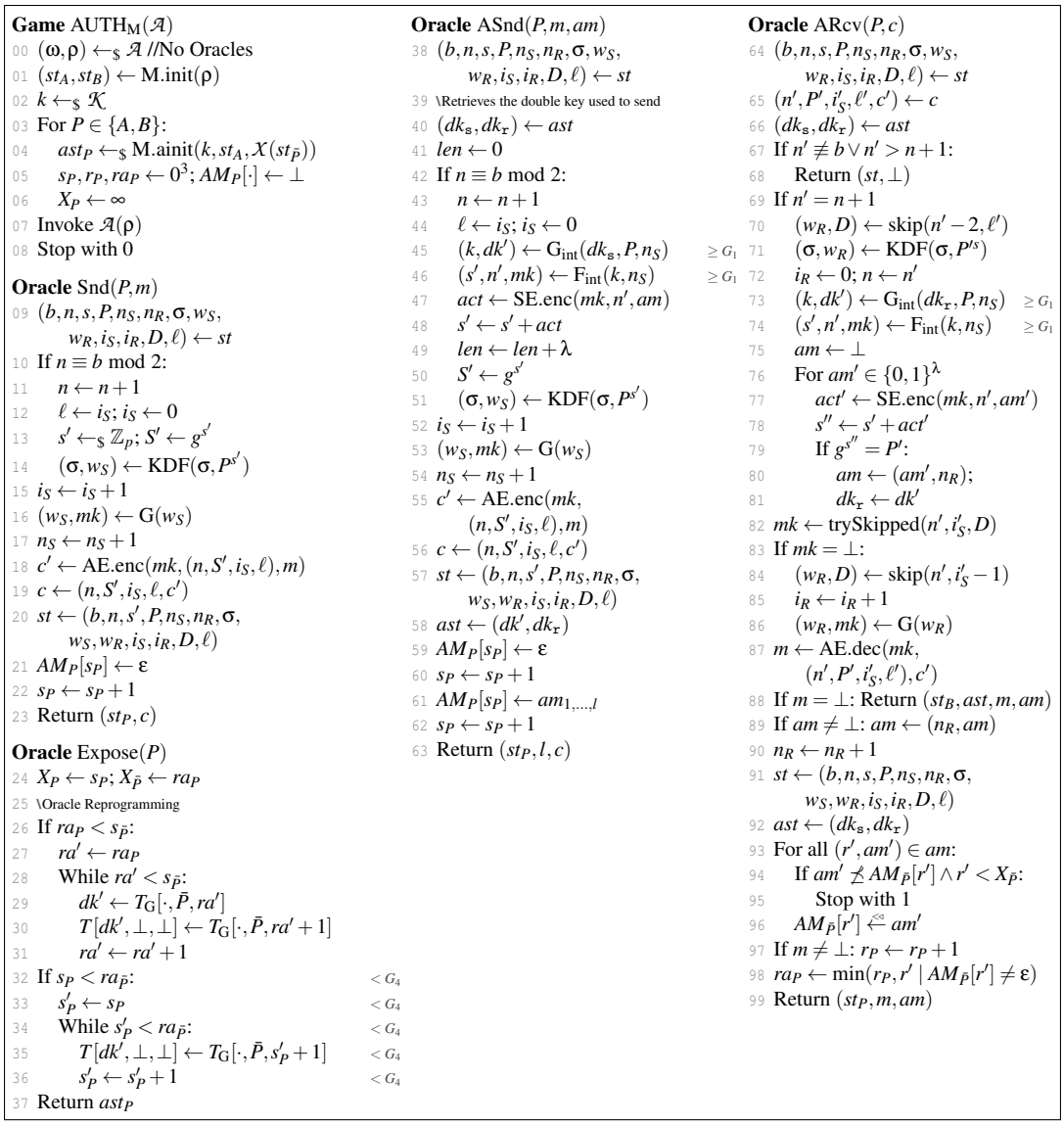

Figure 13: Hybrid arguments for robustness and authenticity of the Double Ratchet anamorphic extension. $\mathcal{A}$ has additional access to G_{ext} and F_{ext} in Figure 14.

the adversary must call ARcv on a message that hasn't been send on this round, that correctly decrypts to an anamorphic message. Hence, they must send a public key g^f such that $f = s' + \text{SE.enc}(mk, n, am')$ for $am' \in \{0,1\}^{\lambda}$. So, they must guess one of the 2^{λ} exponents that decrypt, out of p possible ones (where p is the order of g). Therefore, the bound is:

$$\text{Adv}_{G_4}(\mathcal{A}) \leq \text{Adv}_{G_5}(\mathcal{A}) + \frac{q_{\text{ARcv}} 2^{\lambda}}{p}.$$

To conclude the proof, we can note that in G_5, the adversary can't win the game anymore, so $\text{Adv}_{G_5}(\mathcal{A}) = 0$.

□

D Security analysis of the Double Ratchet anamorphic extension in partial surveillance

This section establishes security of the scheme presented in Section 4 by showing that it satisfies the correctness, anamorphism and authenticity. This result consists of the following Theorems 3, 4 and 12.

Theorem 12 (Correctness)**.** *The anamorphic extension of Figure 5 is correct.*

Proof. The correctness of the construction follows from the correctness of the underlying messaging scheme. □

Proof of Theorem 3. We prove the result using a sequence of game hops. The first five game hops are exactly the same with

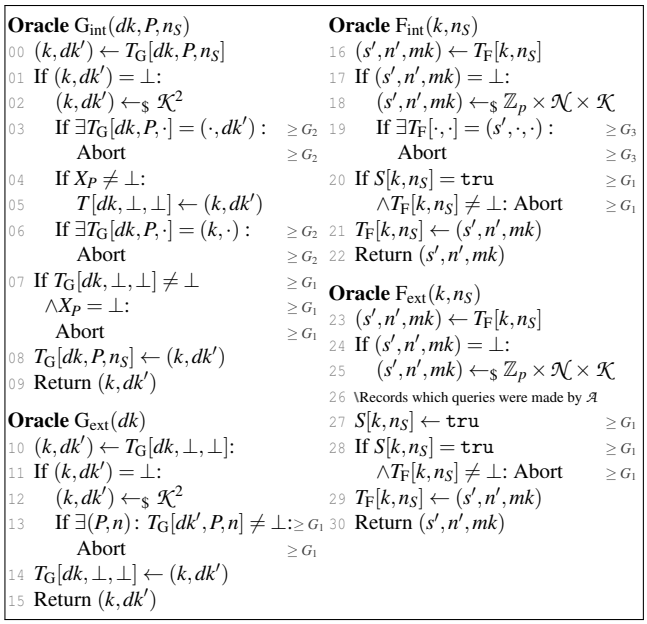

Figure 14: ROM interfaces for the AUTH game.

those in the proof of Theorem 2, with only two differences. First, the adversary is restricted from exposing Alice. The reduction built on a stronger adversary with Alice state exposure is trivially carried over. Second, all the ASnd, ARcv and Chall considered in the game hops are in the direction from Bob to Alice.

Game G_6: is the game obtained by substituting all the outputs of SE.enc in ASnd and Chall queries in the direction from Alice to Bob with randomly sampled elements of ciphertext space $\mathcal{C}$. Similar to game G_6 of Theorem 9, this hybrid adds to the bound a term of $\mathrm{Adv}^{\mathrm{ind\$}}_{\mathrm{SE}}(\mathcal{D})$, multiplying by the number of ASnd and Chall queries in the direction from Alice to Bob.

Game G_7: is the game obtained by forbidding the anamorphic decryption of messages that were not generated with M.asnd. Similar to game G_6 of Theorem 4, this hybrid adds to the bound a single term of $\mathrm{Adv}^{\text{suf-cma}}_{\mathrm{SE}}(\mathcal{D})$: a single SUF-CMA reduction forwards every SE.enc query and every anamorphic-decryption forgery attempt from both directions to one SUF-CMA challenger, so the term carries a tightness factor of 1 and is not multiplied by a query count.

In total, the IND$ terms from both directions add up to $(q_{\mathrm{ASnd}} + q_{\mathrm{Chall}})\mathrm{Adv}^{\mathrm{ind\$}}_{\mathrm{SE}}(\mathcal{D})$, as the total queries are still ASnd + Chall in both directions, while the single SUF-CMA reduction already covers both directions at once, giving the final bound. □

Proof of Theorem 4. We prove the result using a sequence of game hops. The first five game hops from G_1 to G_5 are exactly the same as those in the proof of Theorem 2. Similar to the proof of anamorphism above, the repetitive games arrive with only two differences. First, the adversary is restricted from exposing Alice. The reduction built on a stronger adversary with Alice state exposure is trivially carried over. Second, all the ASnd, ARcv and considered in the game hops are in the direction from Bob to Alice.

Game G_6: This is the game obtained by forbidding the anamorphic decryption of Alice-to-Bob messages that were not generated with M.asnd. Note that forged ciphertexts from Bob to Alice are invalidated in the prior game hops. On a PPT distinguisher between this game and the previous game, we build a reduction to the SUF-CMA experiment.

The reduction works similarly to the one introduced in Theorem 8: it uses its own Enc oracle to answer ASnd queries from Alice. When the adversary queries Expose, the reduction queries its own Corrupt oracle to obtain the corrupted keys. In addition to the corrupted keys, the reduction needs to adaptively corrupt all the keys that can be deterministically derived from the attacker using the ROs interfaces and which the reduction cannot simulate (*e.g.*, since upon corruption of P the adversary learns also the chains used to encrypt by $\bar{P}$, the reduction needs to corrupt the SUF-CMA instances related to calls to $\mathrm{ASnd}(\bar{P})$ that have not yet been received). To decrypt adversarially forged ciphertexts with ARcv queries on Bob's side, the reduction uses its Dec oracle. An adversary that wins the AUTH game with a ARcv forgery on Bob's side can be used directly to win the SUF-CMA game.

In the end, this hybrid adds to the bound a single term of $\mathrm{Adv}^{\text{suf-cma}}_{\mathrm{SE}}(\mathcal{D})$ and concludes the bound. □

E Security analysis for the Triple Ratchet anamorphic extension

Theorem 13 (Correctness)**.** *The anamorphic extension for the Triple Ratchet protocol is correct.*

The proof follows directly from the correctness of the original Triple Ratchet protocol [DJK+25, ADJ+25].

Proof of Theorem 5. We proceed with a sequence of game hops; the first 5 hops are equivalent to the ones used in the AUTH proof (see Theorem 6), so we are gonna describe from Game G_6 onwards.

Game G_0: Is the original ANA game. **Game** G_1 to **Game** G_5 are derived as in the proof of Theorem 6.

Game G_6: We substitute the output of G_1 in KEM.agen with a random 32-bytes string. Indistinguishability between G_5 and G_6 follows from the security property of G_1. We substitute the output in every call to Chall.

$$\mathrm{Adv}_{G_5}(\mathcal{A}) \leq \mathrm{Adv}_{G_6}(\mathcal{A}) + q_{\mathrm{Chall}}\mathrm{Adv}^{\mathrm{prg}}_{\mathrm{G}_1}(\mathcal{C}).$$

Game G_7: We substitute the output of the SE.enc with a random element of $\mathcal{C}$ in the post-quantum embedding spot in Chall. We reach G_7 it with a series of q_{Chall} hybrids. In the i-th hybrid, we substitute the first i outputs of SE.enc in

Chall with random elements of $\mathcal{C}$. The distance between two subsequent hybrids $G_{6,i}$ and $G_{6,i+1}$ is

$$\mathrm{Adv}_{G_{6,i}}(\mathcal{A}) \leq \mathrm{Adv}_{G_{6,i+1}}(\mathcal{A}) + \mathrm{Adv}^{\mathrm{ind\$}}_{\mathrm{SE}}(\mathcal{D}_i)$$

and, since $G_{6,0} = G_6$ and $G_{6,q_{\mathrm{Chall}}} = G_7$

$$\mathrm{Adv}_{G_6}(\mathcal{A}) \leq \mathrm{Adv}_{G_7}(\mathcal{A}) + q_{\mathrm{Chall}}\mathrm{Adv}^{\mathrm{ind\$}}_{\mathrm{SE}}(\mathcal{D}).$$

To conclude the proof, we notice that in G_7 all the ciphertexts generated by SE in Chall are replaced by random ciphertexts, and the adversary cannot forge ciphertexts to bring the states of the two parties out of sync. Thus, the best strategy is a random guess, and $\mathrm{Adv}_{G_7} = 0$.

□

Proof of Theorem 6. **Game** G_0: The original AUTH game.

$$\mathrm{Adv}_{G_0}(\mathcal{A}) = \mathrm{Adv}^{\mathrm{AUTH}}_{\mathrm{M}}(\mathcal{A})$$

Similarly to the proof of Theorem 2, the simulation does some bookkeeping, keeping track of which random oracle calls are made to the external interface, and which to the internal interface.

Game G_1: It's the game obtained aborting if the adversary queries the external oracles on an input used during the simulation, before a corruption. We can bound the probability of this event using a statistical argument, similar to the one used in other games. Since in each call to q_{ASnd} up to two queries to G are made, there is an additional factor of 2:

$$\mathrm{Adv}_{G_0}(\mathcal{A}) \leq \mathrm{Adv}_{G_1}(\mathcal{A}) + \frac{(q_{\mathrm{G_{ext}}} + q_{\mathrm{F_{ext}}})2q_{\mathrm{ASnd}}}{|\mathcal{K}|}$$

where $q_{\mathrm{G_{ext}}}$ and $q_{\mathrm{F_{ext}}}$ are the number of calls to the external interfaces of G and F.

Game G_2: We introduce lazy sampling for G and F, aborting if a collision on the hash-chain or a collision on key for F occur. Furthermore, since the RO G is used to generate both the double ratchet and the triple ratchet key chains dk_{dr} and dk_{tr}, in this game we rule out a collision on both the chains. The advantage bound is obtained via the birthday bound, considering that there are two hash chains:

$$\mathrm{Adv}_{G_1}(\mathcal{A}) \leq \mathrm{Adv}_{G_2}(\mathcal{A}) + \frac{4q^2_{\mathrm{ASnd}}}{|\mathcal{K}|}.$$

Game G_3: We generalize the Game G_3 of 2 to the Triple Ratchet case. We abort in case of a collision on the exponent s', nonce and key of the double ratchet embedding spot, or nonce and key in the triple ratchet embedding spot occurs. We can bound the advantage using the birthday bound:

$$\mathrm{Adv}_{G_2}(\mathcal{A}) \leq \mathrm{Adv}_{G_3}(\mathcal{A}) + \frac{q^2_{\mathrm{ASnd}}}{p|\mathcal{N}||\mathcal{K}|} + \frac{q^2_{\mathrm{ASnd}}}{|\mathcal{K}||\mathcal{N}|}.$$

Game G_4: We forbid the possibility of forgeries in the double ratchet embedding spot, using the same statistical argument used in Theorem 2. The advantage bound is

$$\mathrm{Adv}_{G_3}(\mathcal{A}) \leq \mathrm{Adv}_{G_4}(\mathcal{A}) + \frac{q_{\mathrm{ARcv}}2^{\lambda}}{p}.$$

Game G_5: We construct a reduction to the SUF-CMA security of SE. The reduction works similarly to the one introduced in Theorem 8: it uses its own Enc oracle to answer queries to ASnd that contain a post-quantum embedding spot (*i.e.,* the calls that execute lines 21 and 13). When the adversary queries Expose, the reduction queries its own Corrupt oracle to obtain the corrupted keys. In addition to the corrupted keys, the reduction needs to adaptively corrupt all the keys that can be deterministically derived from the attacker using the ROs interfaces and which the reduction cannot simulate (*e.g.*, since upon corruption of P the adversary learns also the chains used to encrypt by $\bar{P}$, the reduction needs to corrupt the SUF-CMA instances related to calls to $\mathrm{ASnd}(\bar{P})$ that have not yet been received). To decrypt adversarially forged ciphertexts, the reduction uses its Dec oracle, hence an adversary that wins the AUTH game with a forgery on the post-quantum embedding spot can be used directly to win the SUF-CMA game. The advantage bound between G_4 and G_5 is:

$$\mathrm{Adv}_{G_4}(\mathcal{A}) \leq \mathrm{Adv}_{G_5}(\mathcal{A}) + \mathrm{Adv}^{\text{suf-cma}}_{\mathrm{SE}}(\mathcal{B}).$$

In G_5, since we disallowed forgeries both in the Diffie-Hellman ratchet and in the ML-KEM ratchet, we have $\mathrm{Adv}_{G_5}(\mathcal{A}) = 0$.

□

F Security analysis of the PQ3 anamorphic extension

Theorem 14 (Correctness)**.** *The anamorphic extension for the PQ3 protocol is correct.*

The proof follows directly from the correctness of the normal PQ3 protocol.

Proof of Theorem 7. We proceed with a sequence of game hops.

Game G_0: is the base ANA game.

Game G_1 to **Game** G_4 are the same games used in the AUTH game. Thus, in G_4 no forgeries are allowed in the ARcv oracle.

Game G_5: We substitute the outputs of the PRG G_1 in Chall with a random element of $\{0,1\}^{256}$. We construct q_{Chall} hybrid, and in the i-th hybrid we substitute the first i outputs of G_1 with a random output. The distance between two subsequent hybrids is bounded by the advantage of an adversary in the G game:

$$\mathrm{Adv}_{G_{4,i}}(\mathcal{A}) \leq \mathrm{Adv}_{G_{4,i+1}}(\mathcal{A}) + \mathrm{Adv}^{\mathrm{prg}}_{\mathrm{G}_1}(\mathcal{C}_i).$$

Consequently, since $G_{4,0} = G_4$ and $G_{4.q_{\mathrm{Chall}}} = G_5$, the final bound is

$$\mathrm{Adv}_{G_4}(\mathcal{A}) \leq \mathrm{Adv}_{G_5}(\mathcal{A}) + q_{\mathrm{Chall}}\mathrm{Adv}^{\mathrm{prg}}_{\mathrm{G}_1}(\mathcal{C}).$$

Game G_6: We substitute the output of SE.enc in Chall with random elements of $\mathcal{C}$. We define a sequence of q_{Chall} hybrids: in the i-th hybrid, we substitute the output of SE.enc in the first i calls to Chall. Since in each query to Chall there are two encryptions, we split each hybrid in two sub-hybrids: in the first one we substitute only the output of SE.enc in the signature embedding spot, and in the second we substitute also the output in the post-quantum ratchet embedding spot. Then, the difference between two subsequent hybrids is:

$$\mathrm{Adv}_{G_{5,i}}(\mathcal{A}) \leq \mathrm{Adv}_{G_{5,i+1}}(\mathcal{A}) + 2\mathrm{Adv}^{\mathrm{ind\$}}_{\mathrm{SE}}(\mathcal{D}_i)$$

and the distance between the two games

$$\mathrm{Adv}_{G_5}(\mathcal{A}) \leq \mathrm{Adv}_{G_6}(\mathcal{A}) + 2q_{\mathrm{Chall}}\mathrm{Adv}^{\mathrm{ind\$}}_{\mathrm{SE}}(\mathcal{D}).$$

Since in G_6 all the visible outputs of SE.enc in Chall have been substituted with random values, the best strategy for an adversary is random guess, and their advantage is $\mathrm{Adv}_{G_6}(\mathcal{A}) = 0$.

□

Proof of Theorem 8. We proceed with a sequence of game hops.

Game G_0 is the base game. Similarly to the previous proofs, the simulation does bookkeeping to record which calls were made to the ROs through internal or external oracle interfaces. In particular, the simulation holds two tables for each random oracle. When internally deriving a new key chain with a call to $\mathrm{G}_{\mathrm{int}}$, the simulation inputs additional information (*i.e.*, P and s_P or P and ra_P).

Hence, the ROM interface behaves through the game similarly to the ones presented in Figure 14, with the difference that, due to the reduction to the SUF-CMA game, we need to reprogram also the random oracle F. Therefore, each query to the F obtained with the internal interface must also include additional, bookkeeping, information, to efficiently program the RO in case of exposure.

Game G_1: We introduce lazy sampling for the Random Oracles. In G_1, it aborts if an external oracle call to G or F is made on an internal input. Thus, we can bound this probability with a statistical argument:

$$\mathrm{Adv}_{G_0}(\mathcal{A}) \leq \mathrm{Adv}_{G_1}(\mathcal{A}) + \frac{(q_{\mathrm{G}} + q_{\mathrm{F}})q_{\mathrm{ASnd}}}{|\mathcal{K}|}$$

Game G_2: The simulation aborts in case of a collision on the hash chain and on the F key k. Hence,

$$\mathrm{Adv}_{G_1}(\mathcal{A}) \leq \mathrm{Adv}_{G_2}(\mathcal{A}) + \frac{2q^2_{\mathrm{ASnd}}}{|\mathcal{K}|}.$$

Game G_3: We abort in case of a collision on the tuples $(s', n_{\mathrm{dr}}, mk_{\mathrm{dr}})$, (n_{pq}, mk_{pq}) and (n_σ, mk_σ). We bound the probability of this event using the birthday bound.

$$\mathrm{Adv}_{G_2}(\mathcal{A}) \leq \mathrm{Adv}_{G_3}(\mathcal{A}) + \frac{q^2_{\mathrm{ASnd}}}{p|\mathcal{N}||\mathcal{K}|} + \frac{2q^2_{\mathrm{ASnd}}}{|\mathcal{N}||\mathcal{K}|}.$$

Game G_4: We disallow forgeries, constructing a reduction to the SUF-CMA game of SE. Since the authentication tag is verified on the decryption of the signature embedding, the reduction answers as follows:

- On query to ASnd, it simulates the eventual anamorphic encryptions in the Diffie-Hellman and in the ML-KEM embedding spots. To encrypt the anamorphic message in the signature spot, it initializes a new instances of the SUF-CMA game with a query to the oracle Gen, and forwards the encryption to the Enc oracle. The simulation returns to the adversary the ciphertext obtained with the simulated anamorphic ciphertexts and the result of Enc.
- On query to ARcv, the simulator can either simulate a decryption, if the queried ciphertext corresponds to the one outputted with the corresponding call to ASnd, or it can forward it to the Dec oracle of the SUF-CMA game.
- On query to Expose(P), the simulator corrupts the SUF-CMA instance corresponding to the previous call to ASnd($P, \cdot$). After receiving the SE key k_i, the simulation can program the random oracle F to output k_i on input the exposed key.

  Furthermore, since an exposure exposes also the receiving key (ratcheted in ARcv(P)), we need to adaptively corrupt the instances of the SUF-CMA game that were generated with queries to ASnd($\bar{P}$) subsequent to the last query to ARcv(P), and program the RO F to respond coherently.

We can notice that an adversary that creates a forgery in G_3 can be reduced to an adversary that breaks SUF-CMA of SE, hence we obtain the bound

$$\mathrm{Adv}_{G_3}(\mathcal{A}) \leq \mathrm{Adv}_{G_4}(\mathcal{A}) + \mathrm{Adv}^{\text{suf-cma}}_{\mathrm{SE}}(\mathcal{B}).$$

After G_4, the adversary cannot win the AUTH game, and $\mathrm{Adv}_{G_4}(\mathcal{A}) = 0$.

□

G Analysis of our MLS anamorphic extension

We prove correctness, anamorphism, robustness and authenticity of our MLS extension.

Theorem 15 (Correctness)**.** *The MLS anamorphic extension of Figure 9 is correct.*

Proof of Theorem 15. The proof follows directly from the correctness of the original MLS protocol. □

Proof of Theorem 9. Let $q \coloneqq q_{\text{ASnd}} + q_{\text{Chall}}$, we proceed the games G_0-G_3 as in Proof of Theorem 10 to simulate random oracles G and F. Then we proceed the games G_4-G_5 as in Proof of Theorem 10 to exclude any forged anamorphic ciphertexts.

Game G_6. This hybrid is identical to game G_5 except the following change. In each query to oracle ASnd and Chall when $b = 1$, the values s_1 and $r_{\texttt{psk}}$ at line 10, Figure 9 is instead generated uniformly as random.

We show the indistinguishability between G_5 and G_6 following the IND$ property of SE. Specifically, we use a sequence of $q_{\text{ASnd}} + q_{\text{Chall}}$ hybrids $G_{5,i}$ for $0 \le i \le q_{\text{ASnd}} + q_{\text{Chall}}$. At the ith hybrid, the outputs of SE.enc obtained at the first i calls to ASnd or Chall are substituted with random elements in $\mathcal{C}$. In particular, at the ith hybrid, the reduction $\mathcal{F}_i$ playing against the IND$ challenger samples a random bit b' and proceeds the first $i-1$ queries of ASnd or Chall with a random value s_1. At the ith query, $\mathcal{F}_i$ calls their own challenge oracle on am, receives c^*, and then computes the ciphertext for $\mathcal{A}$ using $s_1 \coloneqq c^*$ if the query was to ASnd or to Chall with $b' = 1$, or will compute it by sampling s_1 at random. If $b = 0$ in the IND$ game, then $\mathcal{F}_i$ is simulating $G_{5,i-1}$, otherwise it's simulating $G_{5,i}$. We can then bound the distinguishing advantage between two consecutive hybrids with the advantage of $\mathcal{F}_i$ winning the IND$ game.

$$\text{Adv}_{G_{5,i-1}}(\mathcal{A}) \le \text{Adv}_{G_{5,i}}(\mathcal{A}) + \text{Adv}^{\text{ind\$}}_{\text{SE}}(\mathcal{F}_i)$$

and, since there are $q_{\text{ASnd}} + q_{\text{Chall}}$ hybrids, the final bound is

$$|\Pr[G_5 = 1] - \Pr[G_6 = 1]| \le q \cdot \text{Adv}^{\text{ind\$}}_{\text{SE}}(\mathcal{C}).$$

Game G_7. This hybrid is identical to game G_6 except the following change. In each query to oracle ASnd and Chall when $b = 1$, the value r_1 at line 15, Figure 9 is instead generated uniformly as random. Similar to the change from G_5 to G_6 above, we have

$$|\Pr[G_6 = 1] - \Pr[G_7 = 1]| \le q \cdot \text{Adv}^{\text{ind\$}}_{\text{SE}}(\mathcal{C}).$$

Game G_8. This hybrid is identical to game G_7 except the following change. In each query to oracle ASnd and Chall when $b = 1$, the value Δ at line 19 is instead generated uniformly as random. Similar to the game change above, we have

$$|\Pr[G_7 = 1] - \Pr[G_8 = 1]| \le q \cdot \text{Adv}^{\text{ind\$}}_{\text{SE}}(\mathcal{C}).$$

Game G_9. This hybrid is identical to game G_8 except the following change. In each query to oracle ASnd and Chall when $b = 1$, the value r_3 at line 22 is instead generated uniformly as random. Similar to the game change above, we have

$$|\Pr[G_8 = 1] - \Pr[G_9 = 1]| \le q \cdot \text{Adv}^{\text{ind\$}}_{\text{SE}}(\mathcal{C}).$$

Observing that $\Pr[G_9 = 1] = 0$ concludes the proof. □

Proof of Theorem 10. Let $q \coloneqq q_{\text{ASnd}}$, we proceed with the following sequence of game hops.

We proceed with a sequence of game hops.

Game G_0 is the base game. Similarly to the previous proofs, the simulation does bookkeeping to record which calls were made to the ROs through internal or external oracle interfaces. In particular, the simulation holds two tables for each random oracle. When internally deriving a new key chain with a call to G_{int}, the simulation inputs additional information (*i.e.*, P and s_P or P and ra_P).

Hence, the ROM interface behaves through the game similarly to the ones presented in Figure 14, with the difference that, due to the reduction to the SUF-CMA game, we need to reprogram also the random oracle F. Hence, each query to the F obtained with the internal interface must also include additional, bookkeeping, information, to efficiently program the RO in case of exposure.

We introduce lazy sampling for the Random Oracles.

Game G_1: In G_1, it aborts if an external oracle call to G or F is made on an internal input. Thus, we can bound this probability with a statistical argument:

$$\text{Adv}_{G_0}(\mathcal{A}) \le \text{Adv}_{G_1}(\mathcal{A}) + \frac{(q_{\text{G}_{\text{ext}}} + q_{\text{F}_{\text{ext}}})q_{\text{ASnd}}}{|\mathcal{K}|}$$

Game G_2: The simulation aborts in case of a collision on the hash chain and on the F key k. Hence,

$$\text{Adv}_{G_1}(\mathcal{A}) \le \text{Adv}_{G_2}(\mathcal{A}) + \frac{2q^2_{\text{ASnd}}}{|\mathcal{K}|}.$$

Game G_3: We abort in case of a collision on a nonce-key tuple $(k_{pk,1}, n'_1), (k_{pk,2}, n'_2), (k_{msg,1}, \bar{n}_1), (k_{msg,2}, \bar{n}_2)$. We bound the probability of this event using the birthday bound.

$$\text{Adv}_{G_2}(\mathcal{A}) \le \text{Adv}_{G_3}(\mathcal{A}) + \frac{16q^2_{\text{ASnd}}}{|\mathcal{N}||\mathcal{K}|}.$$

Game G_4. This hybrid is identical to game G_3 except the following change. When G_3 stops with 1 during a query to oracle ARcv, we let the simulation abort if there is a successful forgery r_1 to the underlying SE scheme at line 15.

We can notice that an adversary that creates a forgery in this game can be reduced to an adversary that breaks SUF-CMA of SE, similar to game G_4 in Proof of Theorem 8. Hence, we obtain the bound

$$\text{Adv}_{G_3}(\mathcal{A}) \le \text{Adv}_{G_4}(\mathcal{A}) + \text{Adv}^{\text{suf-cma}}_{\text{SE}}(\mathcal{B}).$$

Game G_5. This hybrid is identical to game G_4 except the following change. When G_4 stops with 1 during a query to oracle ARcv, we let the simulation abort if there is a successful forgery r_3 to the underlying SE scheme at line 22. Similar to the game change above, we have

$$|\Pr[G_5 = 1] - \Pr[G_4 = 1]| \leq \mathrm{Adv}^{\text{suf-cma}}_{\mathrm{SE}}(\mathcal{C}).$$

Note that game G_5 never stops with 1. Indeed, in any attempt to ARcv oracle, the decrypted anamorphic message (r, am), in which am is rewritten as four parts, is never fresh, as each part never fresh in game G_{i+1}. Therefore, $\Pr[G_5 = 1] = 0$ and we conclude the bound. □

H Normal Protocols

For completeness, we include a description of the protocols analyzed, without their anamorphic extension: Signal's messaging scheme based only on the Double Ratchet in Figure 16, Signal's messaging scheme based on the Triple Ratchet in Figure 17, Apple's PQ3 messaging scheme based on the Double Ratchet and ML-KEM in Figure 18, and the messaging scheme of MLS in Figure 19.

Proc M.ainit(k, st_P)
00 $(dk_A, dk_B) \leftarrow k$
01 $ast_P \leftarrow (dk_P, dk_{\bar{P}})$
02 Return ast_P

Proc M.asnd(st_A, ast_A, m, am)
03 $(b, n, s, P, n_S, n_R, \sigma, w_S, w_R, i_S, i_R, D, \ell) \leftarrow st$
04 · $(dk_s, dk_r, p, S) \leftarrow ast_A$
05 · $len \leftarrow 0$
06 If $n \equiv b \bmod 2$
07 $n \leftarrow n+1$
08 $\ell \leftarrow i_S$; $i_S \leftarrow 0$
09 · $(k, dk_s) \leftarrow \mathrm{G}(dk_s)$
10 · $(n', mk) \leftarrow F_k(n_S)$
11 · $ct \leftarrow_{\$} \mathrm{SE.enc}(mk, n', am)$
12 · $S' \leftarrow \mathrm{bin2gr}(ct)$
13 · $len \leftarrow len + |am|$
14 · $(\sigma, w_S) \leftarrow \mathrm{KDF}(\sigma, S'^{p})$
15 $i_S \leftarrow i_S + 1$;
16 $(w_S, mk) \leftarrow \mathrm{G}(w_S)$
17 $n_S \leftarrow n_S + 1$
18 $c' \leftarrow \mathrm{AE.enc}(mk, (n, S', i_S, \ell), m)$
19 $c \leftarrow (n, S', i_S, \ell, c')$
20 · $st_A \leftarrow (b, n, \perp, P, n_S, n_R, \sigma, w_S, w_R, i_S, i_R, D, \ell)$
21 · $ast_A \leftarrow (dk_s, dk_r, p, S')$
22 · Return (st_A, ast_A, len, c)

Proc M.asnd$^{\lambda}(st_B, ast_B, m, am)$
23 $(b, n, s, P, n_S, n_R, \sigma, w_S, w_R, i_S, i_R, D, \ell) \leftarrow st$
24 · \Retrieves the double key used to send
25 · $(dk_s, dk_r) \leftarrow ast_B$
26 · $len \leftarrow 0$
27 If $n \equiv b \bmod 2$:
28 $n \leftarrow n+1$
29 $\ell \leftarrow i_S$; $i_S \leftarrow 0$
30 · $(k, dk_s) \leftarrow \mathrm{G}(dk_s)$
31 · $(s', n', mk) \leftarrow F_k(n_S)$
32 · If $am \neq \perp$:
33 · $act \leftarrow$
34 · $\mathrm{SE.enc}(mk, n', am)$
35 · $s' \leftarrow s' + act$
36 · $len \leftarrow len + \lambda$
37 $S' \leftarrow g^{s'}$
38 $(\sigma, w_S) \leftarrow \mathrm{KDF}(\sigma, P^{s'})$
39 $i_S \leftarrow i_S + 1$
40 $(w_S, mk) \leftarrow \mathrm{G}(w_S)$
41 $n_S \leftarrow n_S + 1$
42 $c' \leftarrow \mathrm{AE.enc}(mk, (n, S', i_S, \ell), m)$
43 $c \leftarrow (n, S', i_S, \ell, c')$
44 $st \leftarrow (b, n, s', P, n_S, n_R, \sigma, w_S, w_R, i_S, i_R, D, \ell)$
45 · $ast_B \leftarrow (dk_s, dk_r)$
46 · Return (st_B, ast_B, len, c)

Proc M.arcv(st_B, ast_B, c)
47 $(b, n, s, P, n_S, n_R, \sigma, w_S, w_R, i_S, i_R, D, \ell) \leftarrow st$
48 $(n', P', i'_S, \ell', c') \leftarrow c$
49 · $(dk_s, dk_r) \leftarrow ast_B$
50 If $n' \not\equiv b \vee n' > n+1$:
51 Return $(st_B, \perp)$
52 If $n' = n+1$
53 $(w_R, D) \leftarrow \mathrm{skip}(n'-2, \ell')$
54 $(\sigma, w_R) \leftarrow \mathrm{KDF}(\sigma, P'^{s})$
55 $i_R \leftarrow 0$; $n \leftarrow n'$
56 · $(k, dk_r) \leftarrow \mathrm{G}(dk_r)$
57 · $(n', mk) \leftarrow F_k(n_R)$
58 · $ct \leftarrow \mathrm{gr2bin}(P')$
59 · $am \leftarrow \mathrm{SE.dec}(mk, n', ct)$
60 $mk \leftarrow \mathrm{trySkipped}(n', i'_S, D)$
61 If $mk = \perp$:
62 $(w_R, D) \leftarrow \mathrm{skip}(n', i'_S - 1)$
63 $i_R \leftarrow i_R + 1$
64 $(w_R, mk) \leftarrow \mathrm{G}(w_R)$
65 $m \leftarrow \mathrm{AE.dec}(mk, (n', P', i'_S, \ell'), c')$
66 If $m = \perp$: Return (st_B, m)
67 $n_R \leftarrow n_R + 1$
68 $st_B \leftarrow (b, n, s, P, \sigma, w_S, w_R, i_S, i_R, D, \ell)$
69 · $ast_B \leftarrow (dk_s, dk_r)$
70 · Return (st_B, ast_B, m, am)

Proc M.arcv$^{\lambda}(st_A, ast_A, c)$
71 $(b, n, \perp, P, n_S, n_R, \sigma, w_S, w_R, i_S, i_R, D, \ell) \leftarrow st_A$
72 $(n', P', i'_S, \ell', c') \leftarrow c$
73 · $(dk_s, dk_r, p, S) \leftarrow ast_A$
74 If $n' \not\equiv b \vee n' > n+1$:
75 Return $(st_A, \perp)$
76 If $n' = n+1$
77 · $(k, dk') \leftarrow \mathrm{G}(dk_r)$
78 · $(s', n', mk) \leftarrow \mathrm{F}_k(n_R)$
79 · $am \leftarrow \perp$; $p' \leftarrow s'$
80 · If $g^{s'} \neq P'$:
81 · For $am' \in \{0,1\}^{\lambda}$
82 · $act' \leftarrow \mathrm{SE.enc}(mk, n', am')$
83 · $s'' \leftarrow s' + act'$
84 · If $g^{s''} = P'$:
85 · $am \leftarrow am'$; $dk_r \leftarrow dk'$
86 · $p' \leftarrow s''$
87 $(w_R, D) \leftarrow \mathrm{skip}(n'-2, \ell')$
88 $(\sigma, w_R) \leftarrow \mathrm{KDF}(\sigma, S^{p'})$
89 $i_R \leftarrow 0$; $n \leftarrow n'$
90 $mk \leftarrow \mathrm{trySkipped}(n', i'_S, D)$
91 If $mk = \perp$:
92 $(w_R, D) \leftarrow \mathrm{skip}(n', i'_S - 1)$
93 $i_R \leftarrow i_R + 1$
94 $(w_R, mk) \leftarrow \mathrm{G}(w_R)$
95 $m \leftarrow \mathrm{AE.dec}(mk, (n', P', i'_S, \ell'), c')$
96 If $m = \perp$: Return (st_A, ast, m, am)
97 · If $am \neq \perp$:
98 · $am \leftarrow (n_R, am)$; $dk_r \leftarrow dk'$
99 $n_R \leftarrow n_R + 1$
100 $st_A \leftarrow (b, n, \perp, P, n_S, n_R, \sigma, w_S, w_R, i_S, i_R, D, \ell)$
101 · $ast_A \leftarrow (dk_s, dk_r, p', S)$
102 · Return (st_A, ast_A, m, am)

Figure 15: Full description for anamorphic extension of the Signal scheme using Double Ratchet excluding Alice's regular state exposure.

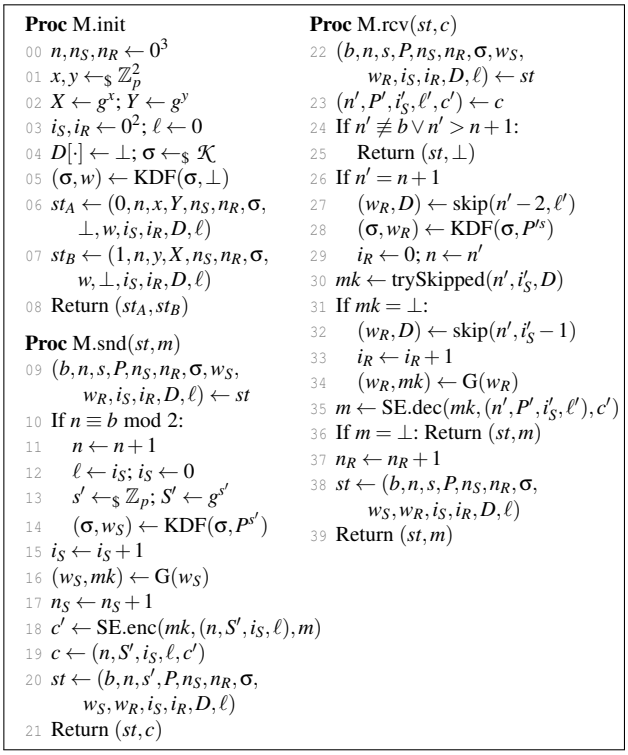

Figure 16: Overview of the Signal's messaging scheme based on the Double Ratchet.

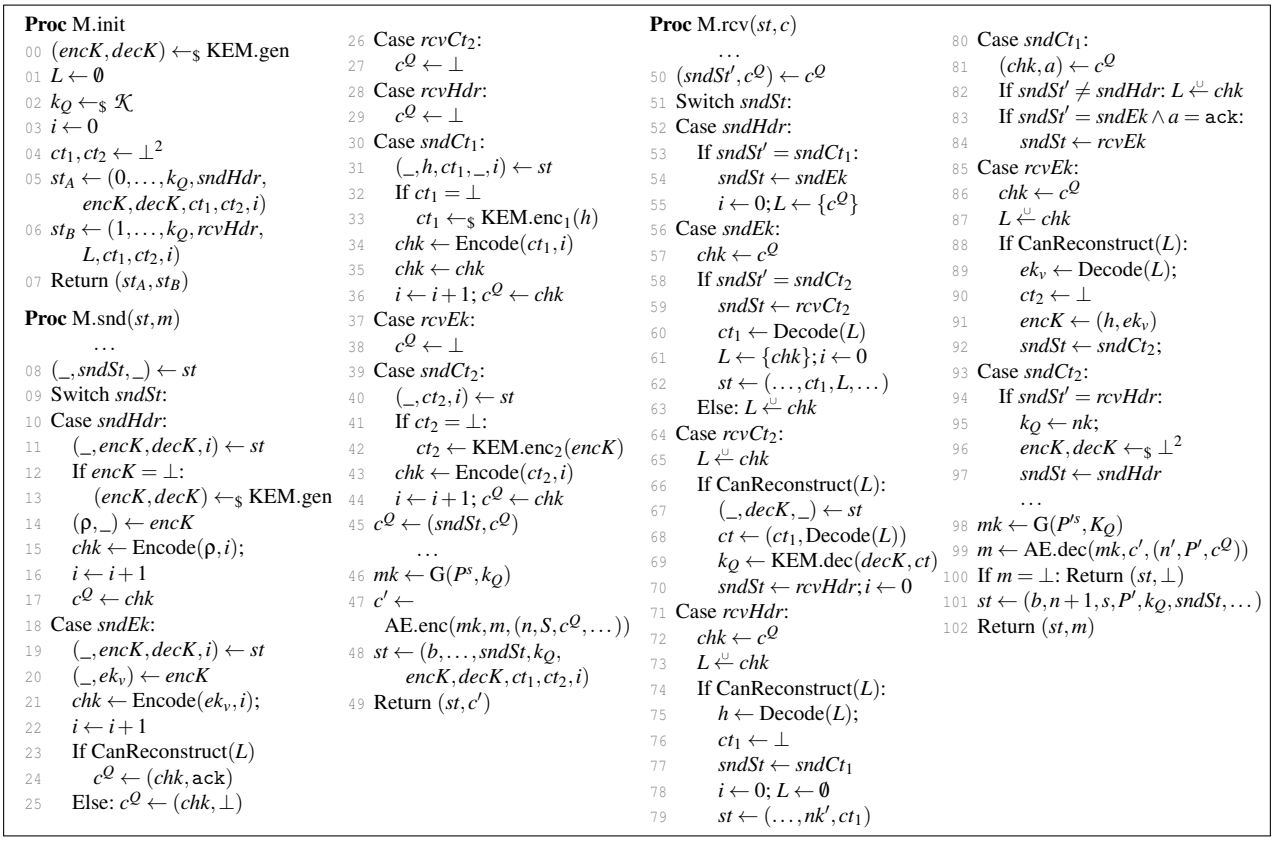

Figure 17: Overview of simplified Triple Ratchet with Opportunistic Sending.

Proc M.init

$n, i_S, i_R, L \leftarrow 0^4$
$n_S, n_R \leftarrow 0^2$
$x, y \leftarrow_\$ \mathbb{Z}_p^2$
$X \leftarrow g^x$; $Y \leftarrow g^y$
$(\text{pq-}encK_A, \text{pq-}decK_A) \leftarrow_\$ \text{KEM.gen}()$
$(\text{pq-}encK_B, \text{pq-}decK_B) \leftarrow_\$ \text{KEM.gen}()$
$\sigma \leftarrow_\$ \mathcal{K}$; $(\sigma, w) \leftarrow \text{KDF}(\sigma, \bot)$
$(sk_{\sigma,A}, pk_{\sigma,A}) \leftarrow \text{SIG.gen}()$ \Generate long-term
$(sk_{\sigma,B}, pk_{\sigma,B}) \leftarrow \text{SIG.gen}()$ \signature keypairs
$st_A \leftarrow (0, n, i_S, i_R, n_S, n_R, \bot, w, \sigma, x, Y, \text{pq-}decK_A, \text{pq-}encK_B, sk_{\sigma,A}, pk_{\sigma,B}, L, sndEk)$
$st_B \leftarrow (1, n, i_S, i_R, n_S, n_R, w, \bot, \sigma, y, X, \text{pq-}decK_B, \text{pq-}encK_A, sk_{\sigma,B}, pk_{\sigma,A}, L, rcvEk)$
Return (st_A, st_B)

Proc M.rcv(st, c, σ)

$(b, n, i_S, i_R, n_S, n_R, w_S, w_R, \sigma, s, P, \text{pq-}decK, \text{pq-}encK, sk_\sigma, pk_\sigma, L, sndSt) \leftarrow st$
$(n', P', c^Q, c') \leftarrow c$
If $n' \not\equiv b \bmod 2 \vee n' > n+1 \vee \text{SIG.vfy}(pk_\sigma, \rho, \sigma) = 0$:
  Return $(st, \bot)$
If $n' = n+1$
  $k_{pq} \leftarrow \bot$
  If $sndSt = rcvCt \wedge c^Q \neq \bot$:
    $k_{pq} \leftarrow \text{KEM.dec}(\text{pq-}decK, c^Q)$
    $sndSt \leftarrow rcvEk$
  Else if $sndSt = rcvEk \wedge c^Q \neq \bot$
    pq-$encK \leftarrow c^Q$; pq-$decK \leftarrow \bot$
    $sndSt \leftarrow sndCt$; $L \leftarrow n'$
  $(\sigma, w_R) \leftarrow \text{KDF}(\sigma, P'^s, k_{pq})$
  $i_R \leftarrow 0$; $n \leftarrow n'$
SID $\leftarrow \ldots$ \compute sid as in M.snd
$i_R \leftarrow i_R + 1$
$(w_R, mk) \leftarrow \text{G}(w_R)$
$m \leftarrow_\$ \text{AE.dec}(mk, (n', P'), c')$
If $m = \bot$: Return (st_B, m)
$n_R \leftarrow n_R + 1$
$st \leftarrow (b, n', i_S, i_R, n_S, n_R, w_S, w_R, \sigma, s, P, \text{pq-}decK, \text{pq-}encK, sk_\sigma, pk_\sigma, L, sndSt)$
Return (st, m)

Proc M.snd(st, m)

$(b, n, i_S, i_R, n_S, n_R, w_S, w_R, \sigma, s, P, \text{pq-}decK, \text{pq-}encK, sk_\sigma, pk_\sigma, L, sndSt) \leftarrow st$
If $n \equiv b \bmod 2$:
  $k_{pq}, c^Q \leftarrow \bot^2$
  $n \leftarrow n+1$
  $i_S \leftarrow 0$
  $s' \leftarrow_\$ \mathbb{Z}_p$ \Classical ratchet
  $S' \leftarrow g^{s'}$
  If $sndSt = sndEk \wedge n \geq L + 50$:
    $(\text{pq-}encK, \text{pq-}decK) \leftarrow_\$ \text{KEM.gen}$
    $sndSt \leftarrow rcvCt$; $L \leftarrow n$; $c^Q \leftarrow \text{pq-}encK$
  Else if $sndSt = sndCt$:
    $(k_{pq}, ct) \leftarrow_\$ \text{KEM.enc}(\text{pq-}encK)$
    $sndSt \leftarrow sndEk$; $L \leftarrow n$;
    $c^Q \leftarrow ct$; pq-$encK \leftarrow \bot$
  SID $\leftarrow (\ldots, P, S', c_{pq}, P_{pq})$ \... include static information like user IDs
  $(\sigma, w_S) \leftarrow \text{KDF}(\sigma, P^{s'}, k_pq)$
$i_S \leftarrow i_S + 1$; $(w_S, mk) \leftarrow \text{G}(w_S)$
$c' \leftarrow_\$ \text{AE.enc}(mk, (n, S', c^Q), m)$
$c \leftarrow (n, S', c^Q, c')$
$n_S \leftarrow n_S + 1$
$st \leftarrow (b, n, i_S, i_R, n_S, n_R, w_S, w_R, \sigma, s, P, \text{pq-}decK, \text{pq-}encK, sk_\sigma, pk_\sigma, L, sndSt)$
$\rho \leftarrow (param, c)$
$\sigma \leftarrow \text{SIG.sig}(sk_\sigma, \rho)$
Return (st, c, σ)

Figure 18: Core of the PQ3 Algorithm using DDH-based continuous key agreement and post-quantum ML-KEM. The relevant subroutines are provided in Figure 6.

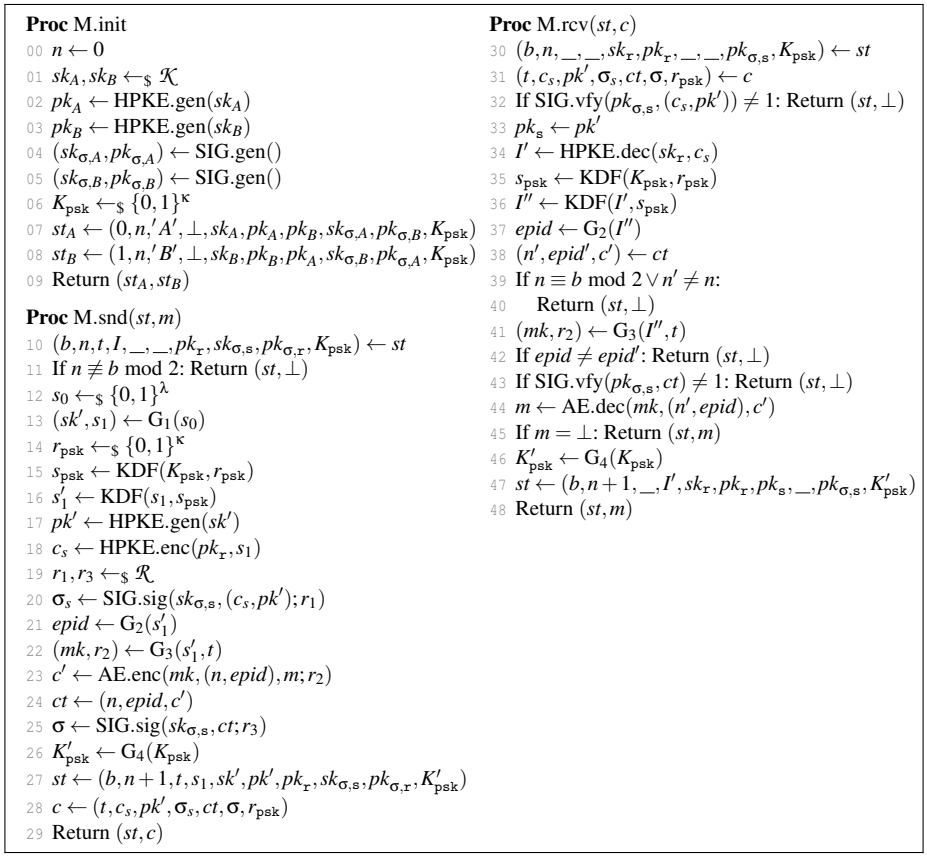

Figure 19: Overview of the core MLS Algorithm for a group of two.