



UNIVERSITÉ DU
LUXEMBOURG

PhD-FSTM-2026-041

The Faculty of Science, Technology and Medicine

DISSERTATION

Defense held on 05/03/2026 in Luxembourg

to obtain the degree of

DOCTEUR DE L'UNIVERSITÉ DU LUXEMBOURG

EN INFORMATIQUE

by

Mohammadamin RAKEEI

Born on 26/11/1991 in Shiraz (Islamic Republic of Iran)

**Securing Communication Under Coercion or Surveillance:
Coercion-Resistance and Covert Channels**

Dissertation defence committee

Dr. Gabriele LENZINI, Dissertation supervisor

Associate Professor, Université du Luxembourg

Dr. Véronique CORTIER

Research Director CNRS, LORIA

Dr. Peter Y. A. RYAN, Chairman

Professor, Université du Luxembourg

Dr. Rosario GIUSTOLISI

Associate Professor, IT University of Copenhagen

Dr. Andy RUPP

Head of Cryptographic Protocols Université du Luxembourg & KASTEL SRL

Affidavit

I hereby declare that the work presented in this thesis is my own and was carried out under the scientific supervision of Gabriele Lenzini, in accordance with the principles of honesty, integrity, and responsibility that govern academic research. The thesis was written independently and without the use of sources other than those explicitly cited. This work has not been previously submitted, either in this country or elsewhere, in the same or in a substantially similar form, to any other examination body.

Mohammadamin Rakeei

Acknowledgements

First and foremost, I would like to express my sincere gratitude to my supervisor, Gabriele Lenzini, for welcoming me into the IRiSC group, giving me the opportunity to pursue my PhD, and granting me the freedom to carry out my research independently. Throughout these years, he was always available to advise me, gently guide me back when I occasionally deviated, and support me with trust and openness. He has been a leader, a mentor, and also a good friend. I truly enjoyed our discussions—both scientific and non-scientific—and I always felt that speaking with him brought value, and new perspectives.

Second, I would like to warmly thank my colleagues. In particular, Rosario Giustolisi, who during my four years of PhD was not only an unofficial advisor, but also a close friend. I always felt comfortable calling him to discuss ideas, doubts, or new directions. I genuinely enjoyed the time we spent working together and the research we conducted, and I hope that time will allow us to collaborate again in the future. I would also like to thank all the members of the IRiSC group, who were consistently supportive, helpful, and kind. They created a friendly and motivating atmosphere that made the research journey much more enjoyable.

I would also like to thank Prof. Peter Ryan for the valuable scientific discussions we had together. It was an honor to have him on my CET committee and to discuss technical topics with him from the first year of my PhD until the final stages of writing this thesis. I am also deeply grateful to all my co-authors across my publications—both those included in this thesis and those from other projects. Working with all of you has been a truly valuable and enriching experience.

Last, but certainly not least, I would like to thank my family. My parents and my brother have always encouraged me at every stage of my life, and their support has meant more to me than words can express. I am deeply grateful to my wife, Sara, who brought calm and peace throughout my PhD years, offering constant support and believing in me even during difficult times. Her encouragement was a continuous source of motivation. Finally, to my lovely daughter, Nahal, who has always been a source of energy and joy. I had to spend less time with her during periods of heavy workload—sorry, Nahal—but your presence always reminded me of what truly matters.

Abstract

Electronic examinations (e-exams) promise scale and flexibility, but they change the trust assumptions of assessment. Beyond authentication, integrity, privacy, and verifiability, this thesis focuses on a human-centered threat: coercion. In high-stakes settings, a coercer may supervise a session, demand keys or transcripts, and later verify compliance. Under such pressure, anonymity tools and audit logs can turn into evidence rather than protection. The central question is how to address coercion while maintaining established security goals for e-exams.

I first study coercion in e-exams and define two properties, *Anonymous Submission* and *Single-Blindness*, that capture privacy under pressure. I show that a state-of-the-art protocol fails to meet these notions, and I design CREX, the first e-exam protocol to satisfy them alongside core privacy and authentication goals. The scheme employs a new shuffling mechanism to mitigate collusion and coercion. The analysis also exposes and fixes a linkability issue in the exponentiation mixnet used by prior work. All claims are modeled and checked in ProVerif.

I then strengthen the threat model to allow leakage of all participant secrets. Under this stronger model, I identify new attacks on CREX and introduce SCREX, which employs a new pre-assignment protocol as its main coercion-resistance strategy. SCREX includes a concrete mixnet model and a full formal analysis in ProVerif that proves coercion-resistance as well as individual and universal verifiability.

However, this new design relies on an untappable channel, which is a strong assumption and difficult to realize at scale on today's internet. This leads to a broader question: can secure and undetectable communication be achieved over ordinary networks, without special links, using the building blocks of deployed systems? From this perspective, surveillance is the adjacent threat to study. It operates at scale and is especially relevant in e-voting and secure messaging. Therefore, I examine two mature settings—e-voting

and the Signal Double Ratchet—to design covert channels that remain compatible with deployed protocols and maintain indistinguishability under surveillance.

In e-voting, I introduce *anamorphic voting* and the notion of *ballot freedom*: a voter can covertly convey the true choice to an auditor while casting a normal-looking ballot, even against dishonest authorities. I realize this by composing primitives already present in deployed systems (IVXV, CHVote, Helios, and Belenios). This yields the first ElGamal-based and Schnorr-style anamorphic constructions designed for deployment.

In secure messaging, I integrate anamorphism into the Signal Double Ratchet to realize a covert channel that lets users communicate undetectably even when authorities can decrypt messages. I show that, even when end-to-end encryption is effectively compromised by lawful access or similar controls, parties can still communicate securely and indistinguishably through this channel. I propose three practical constructions: two embed hidden data in ephemeral Diffie–Hellman public keys and one in MAC tags. The resulting messages are indistinguishable from normal Signal traffic. I implement the designs in the official Signal library and measure their performance, demonstrating practical feasibility and efficiency across a range of anamorphic bit-rate settings.

Overall, the thesis advances the security of communication under coercion and surveillance at two levels: (i) new models and formally verified e-exam protocols that remain secure under weak and strong coercion, and (ii) deployable anamorphic constructions in voting and messaging that create hidden, undetectable channels without special infrastructure. Although I do not present an anamorphic e-exam construction in this thesis, I lay the groundwork and outline concrete next steps for future work in e-exams and beyond.

Index

Abstract	iv
1 Introduction	1
1.1 Background	1
1.2 Motivation	7
1.2.1 Seeking Practical Defenses	8
1.2.2 From Coercion in E-Exams to Broader Applications	10
1.3 Research Objectives/Questions	11
1.4 Contributions	12
1.5 Thesis Outline	13
2 CRex: Secure Internet Exams Despite Coercion	15
Abstract	16
2.1 Introduction	16
2.2 Related Work	18
2.3 Background	19
2.3.1 Security Properties of an e-Exam	19
2.3.2 Coercion resistant e-Exams	19
2.4 Use Case and a Relevant Attack on Mixnet	21
2.4.1 Remark! in a nutshell	21
2.4.2 Intermezzo: An attack and a fix on Remark!’s mixnet	22
2.4.3 A Mixnet resilient to Injection Attacks	23
2.5 Security Formal Analysis	23
2.5.1 Threat Model and Formalization of the New Properties	25
2.6 A Secure Coercion Resistant Exam Protocol	27
2.7 Conclusion	31

3	SCREX: Secure and Verifiable Coercion-Resistant Electronic Exam	33
	Abstract	34
3.1	Introduction	34
3.2	Previous works, Background, and Tools	37
3.2.1	Applied Π -Calculus	39
3.2.2	ProVerif	39
3.3	Formalizing e-Exams and Security Properties	40
3.3.1	E-Exams Modelling	40
3.3.2	Authentication Properties	41
3.3.3	Privacy Properties	42
3.3.4	Verifiability	44
3.3.5	Coercion Properties	46
3.4	A New Verifiable Coercion-Resistant Protocol	47
3.4.1	Registration	48
3.4.2	Assignment Protocol	48
3.4.3	Testing	50
3.4.4	Marking	51
3.4.5	Notification	53
3.5	Formal Security Analysis	54
3.5.1	Authentication and Secrecy in SCREX	56
3.5.2	Privacy in SCREX	56
3.5.3	Coercion in SCREX	57
3.5.4	Verifiability in SCREX	57
3.6	Conclusion	59
4	Anamorphic Voting: Ballot Freedom Against Dishonest Authorities	61
	Abstract	62
4.1	Introduction	62
4.2	Anamorphic Voting	63
4.2.1	From Anamorphic Encryption to Anamorphic Voting	63
4.2.2	A Running Example: CHVote	65
4.2.3	Ballot Freedom in CHVote	66
4.3	More Anamorphic Constructions	67
4.4	The Estonian Internet Voting System	70

4.4.1	Protocol Description.....	70
4.4.2	Ballot Freedom in IVXV.....	72
4.5	Helios	73
4.5.1	Ballot Freedom in Helios.....	74
4.6	Belenios	75
4.6.1	Protocol Description.....	75
4.6.2	Ballot Freedom in Belenios	76
4.7	Related Work	76
4.8	Discussion and Future Work	77
5	Crypto Wars in Secure Messaging: Covert Channels in Signal Despite Leaked Keys	79
	Abstract	80
5.1	Introduction	81
5.1.1	Motivation	81
5.1.2	Contributions	82
5.2	Background	83
5.2.1	Cryptographic functions	83
5.2.2	The Double Ratchet protocol	85
5.2.3	Anamorphic encryption.....	88
5.3	System Design	89
5.4	Covert Channel in DH Ratchet	90
5.4.1	Base Construction	90
5.4.2	Stronger Security Guarantee – IND-CCA.....	94
5.4.3	Instantiation of Double Key dk	94
5.4.4	Optimization.....	95
5.5	Covert Channel in Symmetric Ratchet	97
5.5.1	Base Construction	97
5.5.2	Stronger Security Guarantee – IND-CCA.....	102
5.6	Performance	103
5.7	Related Work	106
5.8	Conclusions	108
	Appendix	109

6	Discussion	120
6.1	On CREX.....	121
6.1.1	Limitations, possible improvements, and future work.....	121
6.2	On SCREX	124
6.2.1	Limitations, possible improvements, and future work.....	125
6.3	On AnaVoting	127
6.3.1	Limitations, possible improvements, and future work.....	127
6.4	On AnaSignal	129
6.4.1	Limitations, possible improvements, and future work.....	129
6.5	Final remarks	131

List of Figures

2.1	Pseudonym generation using the exponentiation mixnet	22
2.2	The message sequence chart of Coercion-Resistant Electronic Exam (CREX) ..	30
3.1	Pseudonym generation using the secure exponentiation mixnet	48
3.2	The message sequence chart of Assignment Protocol.....	51
3.3	The message sequence chart of SCREX protocol	52
5.1	The symmetric ratchet.....	86
5.2	The DH ratchet	86
5.3	An instance of the Double Ratchet protocol for Alice.....	87
5.4	Average decryption latency in Signal Android	103
5.5	Average round-trip latency in Signal Android	104
5.6	IND-CCA Security experiment for Covert Channel in DH Ratchet.....	110
5.7	IND-CCA Security experiment for Covert Channel in Symmetric Ratchet	118

List of Tables

2.1	Synthesis of the findings of the formal analysis under different threats.	29
3.1	Individual and universal verifiability.	45
3.2	Equational Theory.	55
3.3	Summary of the analysis of SCREX for I.V. requirements	58
3.4	Summary of the analysis of SCREX for U.V. requirements	58
3.5	Analysis results of our SCREX protocol	59
5.1	Anamorphic constructions under various threat models	88
5.2	Notation used in Algorithms 8 and 9	95
5.3	Performance on Android real device and Android emulator	103

Chapter 1

Introduction

1.1	Background	1
1.2	Motivation	7
1.2.1	Seeking Practical Defenses	8
1.2.2	From Coercion in E-Exams to Broader Applications	10
1.3	Research Objectives/Questions	11
1.4	Contributions	12
1.5	Thesis Outline	13

1.1 Background

Contextual Overview When assessing a person’s knowledge, an exam is often the first method that comes to mind. At first glance, the process appears simple: select the right questions, gather responses, and evaluate them. Yet a closer examination reveals that assessment is not straightforward and involves multiple layers of complexity.

Questions arise, such as:

- What is the very purpose of conducting an examination, and what kinds of knowledge or competence is it meant to capture?
- Who is entrusted with defining the standards of knowledge, and on what basis are these standards justified?

- Do we possess the expertise to formulate valid and meaningful questions? If not, this responsibility must be delegated to a qualified party.
- Do we have the means to reliably evaluate responses? If not, grading must be entrusted to a competent authority.
- How can we ensure that answers genuinely reflect the candidate's own knowledge, without external assistance or unfair advantage?
- Are we always assessing one individual at a time, or might we be evaluating many simultaneously? If so, do the same challenges apply at scale?
- What logistical difficulties arise in managing examinations when candidates and examiners are geographically dispersed?
- How can we ensure fairness and consistency when examination conditions vary due to location, timing, or available resources?
- And in today's increasingly digital world, how must examinations adapt to operate effectively in online or technology-mediated environments?

These last questions naturally extend beyond the traditional classroom setting. Examinations today are not confined to university courses or standardized tests; they also appear in corporate training, professional certifications, industry licensing, and other high-stakes contexts where outcomes can shape careers or professional standing.

Taken together, they point to the central focus of this thesis: *electronic examinations* (e-exams), where the entire process—from question delivery to answer submission and evaluation—is conducted through digital channels.

Digital Transformation in Assessments Over the last two decades, digital technologies have reshaped not only the delivery of education but also the way knowledge and competence are assessed across multiple domains. From university courses and standardized testing to corporate training, professional certifications, and industry licensing, assessment mechanisms are increasingly migrating from physical settings to the digital domain. One of the most transformative elements of this shift has been the adoption of *electronic examinations* (e-exams).

The appeal of e-exams is clear. They offer *scalability*, enabling institutions and organizations to conduct assessments with hundreds or thousands of candidates simultaneously.

They enable *geographical flexibility*, allowing participation without the constraints of physical examination halls. They can also streamline *administration and grading*, particularly for objective question formats such as multiple-choice. For both educational institutions and professional organizations with geographically dispersed or international participants, these capabilities are not merely convenient—they are essential to operational viability.

However, replacing traditional in-person examinations with their digital counterparts changes not only the medium but also the trust assumptions upon which the assessment process depends. Where physical exams rely on proctors, controlled environments, and physical document security, e-exams must operate in the less predictable context of open networks, diverse end-user devices, and potentially untrusted intermediaries. This introduces new security challenges alongside the traditional ones, demanding solutions that address both technical vulnerabilities and human or organizational risks. In high-stakes scenarios—whether admission to a degree program, obtaining a professional license, or certification for employment—a breach of security or privacy can invalidate results, undermine institutional trust, and damage the future prospects of candidates.

The Threat Landscape E-exams must operate securely in environments that may not be fully trusted. Examinees may be located remotely, using personal devices and connecting via networks susceptible to interception or disruption. The process also involves two other central roles: the *examiner*, responsible for evaluating responses and assigning grades, and the *Exam Authority (EA)*, the organization overseeing the process, distributing questions, and ensuring policy compliance. Both the EA and examiners may themselves be targets of attack, as compromising them can yield widespread control over the process.

Potential adversaries include:

- **Dishonest examinees** seeking to gain an unfair advantage.
- **External attackers** attempting to disrupt the exam or leak its contents.
- **Dishonest or colluding officials** administrators, invigilators, or examiners with privileged access that can subvert policy or outcomes.
- **Coercive actors** individuals or groups who use social, economic, or institutional means to influence actions or results.

A secure e-exam protocol must therefore consider both technical attacks, which attempt to subvert the system directly, and social or organizational threats, which exploit human

factors. We refer to this last category broadly as *coercers*, a term that will be used throughout this thesis.

At an operational level, institutions combine supervised spaces, device lockdown, and remote monitoring to stabilize logistics and deter opportunistic misconduct. Such measures help with heterogeneity and scale, but they neither offer end-to-end guarantees nor mitigate abuse by powerful officials. In this work, they are treated as complements to protocol mechanisms, not substitutes.

The threats listed above motivate the security goals enumerated next and the need for designs that remain robust even when some roles or infrastructure cannot be fully trusted.

Security Requirements in E-Exams Understanding the interplay between examinees, examiners, and the exam authority is essential for designing protocols that meet security goals under realistic operational constraints. Over the past two decades, researchers have developed secure e-exam protocols aimed at addressing threats such as impersonation, answer tampering, and information leakage. Commonly accepted security properties in the literature include:

- **Confidentiality:** Questions remain secret until the start of the exam; answers are kept private during and after submission.
- **Integrity:** Data cannot be altered without detection.
- **Authenticity and authorization:** Only registered examinees may take part; only authorized examiners may grade.
- **Verifiability:** The correctness of the process and results can be verified without compromising privacy.
- **Fairness:** No participant gains an undue advantage through timing, resource manipulation, or privileged access.
- **Accountability:** Misbehavior by any actor can be detected and attributed with publicly verifiable evidence, supporting dispute resolution, and appeals.

Among the above items, *fairness* has received considerable attention, with a special focus on preventing *cheating*. These efforts are not strictly about *protocol* design; rather, they involve technical and human-centered measures beyond the protocol layer to promote

fairness. For example, there are works on purely technical approaches to prevent or detect or mitigate cheating (e.g., remote proctoring, lockdown browsers, and camera monitoring [1, 2, 3]) and on socio-technical approaches (e.g., [4]).

As this thesis focuses on the protocol layer, we keep discussion of non-protocol approaches brief and concentrate on security protocols for the remainder.

Beyond the high-level aims listed above, the literature refines security into a compact taxonomy of authentication, privacy, and verifiability [5, 6], and studies concrete protocols in formal models (e.g., the applied π -calculus with automated verification [7, 8]). This combination clarifies what “secure” means in practice, underpins defenses against attacks such as impersonation and tampering, and exposes failure modes under realistic assumptions. A concise literature review, organized by goal and by threat model, will follow in Chapter 2 and Chapter 3.

Technical Background: Privacy-Oriented Designs for E-Exams Privacy in e-exams is not only about hiding content such as marks or answers; it is also about hiding *links*. Typical goals include unlinkability between candidates and the scripts they submit, and unlinkability between examiners and the marks they issue. Recent proposals pursue these requirements in both computer-assisted [9, 6] exams and fully electronic exam settings [10] using cryptographic tools and systems.

A common ingredient is the use of *pseudonyms* so that identities are decoupled from exam records and actions. Candidates obtain one-time or exam-specific pseudonyms during registration and perform all subsequent actions during the examination using only these identifiers; examiners grade pseudonymous scripts; and a controlled, sometimes auditable step later rebinds pseudonyms to real identities to release results [11, 12]. When examiner privacy is needed, a symmetric approach assigns pseudonyms to examiners during marking.

In the literature, unlinkability is typically achieved via anonymous channels. Designs include classic mixes and verifiable shuffles [13, 14, 15], low-latency onion routing [16], scalable/anytrust mix networks (mixnets) [17, 18, 19], and anonymous write/return-channel systems [20, 21, 22]; all aim to hide sender–recipient links, with some offering publicly verifiable shuffle proofs.

Similar systems have also been used in e-exams commonly via mixnet [10, 23]. In this setting, mixnets anonymize question distribution, answer submission, and the assignment

of submissions to examiners, supporting the unlinkability goals above [11, 12]. Verifiable shuffles make anonymity auditable without revealing content.

When verifiability is required, a common practice is to publish protocol events on a *bulletin board*—a public log that stores registrations, commitments, ciphertexts, assignments, grades, and proofs. This enables *individual verifiability* (each participant can check their own records) and *universal verifiability* (anyone can recompute or validate the result from the public evidence) without learning private inputs and private links.

Two patterns follow from this line of work. First, privacy is achieved by removing explicit links in the workflow (identity $\leftrightarrow$ script; examiner $\leftrightarrow$ mark) while keeping outcome integrity publicly checkable. Second, anonymity mechanisms must be coupled with verifiability and accountability: signatures, proofs, and logging ensure that misbehavior can be detected and attributed *without* reintroducing the links that privacy seeks to hide [12, 24, 10].

Parallels with E-Voting E-exams share important structural and security similarities with electronic voting. In both cases, confidential inputs (answers or votes) must remain private while still allowing public verifiability of the outcome. The process must ensure fairness, preventing early disclosure from benefiting later participants, and the correctness of the final result must be demonstrable without revealing private inputs. The e-voting literature also studies *receipt-freeness*—voters should not be able to prove how they voted—and *coercion-resistance*—voters should not be able to prove if they followed a coercer’s instructions. These properties have direct analogues in e-exams—for example, an examiner should not be forced to alter an outcome to favor a particular candidate—yet experience from e-voting shows that secrecy, privacy, and verifiability alone do not neutralize human pressure when a coercer can demand observable compliance. E-exams share this structure, but distribute leverage across candidates, examiners, and the exam authority. We therefore turn to coercion as a central threat and examine how it arises in practice.

Coercion: A Broader Human-Centered Threat Because of the substantial personal, professional, or financial consequences of exam outcomes, adversaries may exploit social or institutional pressure, namely coercion, to influence participants.

In the e-exam setting, coercion can target examiners, candidates, or the exam authority. Examiner coercion occurs when a candidate or a connected party pressures or incentivizes the examiner to alter grades, overlook irregularities, or disclose confidential grading infor-

mation. Because examiners control evaluation, coercing an examiner can directly affect the outcome for one or many candidates. Candidate coercion arises when an external party—such as an employer, instructor, or other authority—forces a candidate to take the exam under supervision, disclose answers, or share authentication credentials. It exploits the candidate’s inability to refuse without personal or professional consequences and can succeed even in protocols that are cryptographically sound. EA coercion involves pressuring the exam authority to modify results, leak exam content, or assign favorable examiners to certain candidates. Given the EA’s central role, coercion at this level can have systemic consequences across the entire examination process.

Taken together, these observations point beyond routine privacy toward privacy under pressure: a coercer must be unable to verify compliance even with supervision or compelled disclosure. Standard mitigations—pseudonymizing candidates during the exam, separating powers across administrators/invigilators/examiners, and publishing publicly verifiable evidence—reduce targeting and support accountability, but they do not stop coerced revelation. In general, anonymity mechanisms such as onion routing (Tor) [16], low-latency mixnets (Loopix) [17], verifiable shuffles with zero-knowledge proofs, anonymous broadcast/write systems (Riposte, Scalix) [20, 25], hide who talks to whom and unlink identities from actions under normal use. Under coercion, however, privacy can fail without breaking any cryptography: a victim can be forced to reveal the identity–pseudonym binding, hand over keys or transcripts, turning audit evidence into proof of compliance. What is required instead is a deniable hidden channel embedded in ordinary protocol messages, so that even after key disclosure any “proof of compliance” is indistinguishable from regular behavior—this motivates the defenses developed next.

1.2 Motivation

In high-consequence assessment contexts, a determined coercer wants not only to learn information but to *verify* that their instructions were followed; this necessitates guarantees that make enforcement ineffective even under supervision or compelled disclosure. In practice, three requirements follow. First, even if a candidate appears to cooperate—by revealing keys, regenerating a claimed submission, or taking the exam under observation—the coercer should still be unable to link any submitted answer to that candidate. Second, even if an examiner appears to cooperate—by disclosing keys, showing grading

records, or acknowledging assigned scripts—the coercer should still be unable to link any issued mark to that examiner. Third, even when pressure is applied at the organizational level—for example, by demanding special treatment or visibility from the EA—the coercer must not be able to convert that pressure into verifiable influence over assignments, content release, or outcomes.

Accordingly, we contribute to formalizing some of these coercion-focused requirements for e-exams and to showing, at a conceptual level, how protocol design can make demanded “proofs of compliance” uninformative while preserving authentication, privacy, and verifiability. The aim is plausible compliance without a way to enforce it: participants may appear to obey, yet the coercer cannot test whether obedience actually occurred.

1.2.1 Seeking Practical Defenses

Several paths for tackling coercion can be considered. One direction, inspired by receipt-freeness in e-voting, models a *weak* or *passive* coercer who does not supervise every step. Such a coercer issues demands and seeks to verify compliance after-the-fact, but lacks the means to force continuous oversight or disclosure. This is a realistic case in e-exams—for example, a candidate who pays an examiner to falsify a mark may want post-hoc confirmation yet cannot monitor grading in real time.

Some approaches in coercion-resistant designs—especially in e-voting—rely on mechanisms that let participants send hidden, authentic information to a trusted authority without the coercer’s knowledge. A canonical example is the *untappable channel*, which ensures that communication cannot be intercepted, altered, or monitored by an adversary.

Building on this, a second direction addresses settings with a *strong* coercer: one able to supervise sessions and force disclosures, tracking the state and timing of events as they unfold. The corresponding response is to place the most sensitive exchanges behind untappable channels and tighten the interfaces between roles, so that even forced disclosure yields no verifiable leverage. Under these assumptions, strong guarantees are achievable.

While effective in theory, such channels are difficult to realize over the internet [26, 27, 28] and cannot guarantee protection against eavesdropping by a powerful adversary with who permanently monitors the channel.

This motivates the search for methods that could achieve a similar effect—allowing a participant to transmit a concealed message that is indistinguishable from ordinary communication to an outside observer. In essence, this means embedding a “hidden channel”

within standard exchanges, such that even if a powerful adversary demands keys or intercepts all traffic, they cannot detect that any extra information was sent.

While the limitations of untappable channels provide one clear motivation for seeking alternatives, the broader aim is to find solutions that are general, adaptable, and applicable across domains facing coercion-like threats. Rather than relying on heavy infrastructure, this research seeks realistic and deployable alternatives that address coercion without sacrificing robustness. This means looking beyond e-exams to techniques that work under practical constraints while still meeting strong security requirements.

Deniable Encryption: A Plausible Yet Insufficient Candidate The notion of deniable encryption was first introduced by Canetti *et al.* [29] to address scenarios where an adversary may compel parties to reveal their private randomness or keys after a ciphertext has been transmitted. In such schemes, the sender (or receiver) can provide “fake” randomness consistent with any chosen plaintext, making the ciphertext appear as an encryption of a different message. Subsequent work sought more practical constructions, most notably the Sahai–Waters scheme [30], which leveraged indistinguishability obfuscation to resolve a long-standing open problem and gave the first efficient public-key deniable encryption without pre-planning. More recently, Canetti *et al.* extended the model to *fully deniable* interactive encryption, offering protection even when both parties are simultaneously coerced, and introducing the stronger notion of off-the-record deniability [31]. The most recent advance by Coladangelo *et al.* considers deniability in the quantum world, showing that quantum computation can unlock new forms of protection such as perfect unexplainability, which even offers limited defense against coercion *before-the-fact* [32].

Despite this impressive line of work, deniable encryption is fundamentally designed for scenarios where coercion occurs only after the communication has taken place. The coercer’s role in the threat model is retrospective: once ciphertexts are transmitted, the adversary demands explanations. This mismatch makes deniable encryption a partial fit for our setting. It may be regarded as a natural candidate for the first line of defense against coercion, yet it cannot account for adversaries who intervene during the protocol execution itself. For that alternative direction—where coercion is not restricted to the after-the-fact case—we require a different solution.

Anamorphic Encryption as a Natural Fit Anamorphic encryption, introduced by Persiano *et al.* in 2022 [33], is a cryptographic technique that can be a good candidate to realize

this kind of hidden channel. It allows a single ciphertext to carry two distinct interpretations: one that appears ordinary and the other that conveys concealed information to a party who knows the right secret. Even if an adversary demands decryption keys, the sender can reveal a key that produces only the ordinary interpretation, and the adversary cannot prove that anything else exists. The coercer must accept the ciphertext as genuine and complete. Crucially, unlike deniable encryption—whose intrinsic model protects only after-the-fact—anamorphic encryption explicitly covers coercion both *before* and *after* communication [33]. In particular, a coercer can approach the sender *prior* to transmission and demand a specific plaintext; the sender can still embed the concealed message while producing public evidence consistent with the mandated plaintext, and any later verification attempt remains inconclusive.

This property aligns naturally with our strong coercion model for e-exams, where an adversary may intervene communications, force key release, and supervise the victim, yet is forced to accept public evidence that remains indistinguishable from compliant behavior.

Despite its apparent suitability, anamorphic encryption is still a young area: most existing work develops foundations and primitives [34, 35, 36] rather than deployable constructions. This thesis therefore focuses on designing and evaluating working constructions in adjacent, well-studied domains, with the aim of identifying properties and patterns that may be applicable to e-exam settings.

1.2.2 From Coercion in E-Exams to Broader Applications

To explore and refine this idea, the investigation extends beyond e-exams into domains where coercion-like or dictatorship threats are well recognized or naturally arise in the security model.

E-Voting In e-voting, coercion is extensively studied through receipt-freeness and coercion-resistance. A related but less explored threat appears when dishonest authorities can decrypt ballots at will. This is not the usual setting of an external coercer targeting a single voter. It is a mass-surveillance scenario in which authorities verify whether voters follow a preferred choice and punish non-compliance. Such control prevents voters from freely expressing their intentions. The design goal that follows is to restore the voter’s ability to convey the true choice to an honest party without altering the public shape of the ballot, which in this work is called *ballot freedom*. The question is whether a covert channel

can be built so that voters can escape this control in a secure and unnoticeable way, and whether anamorphism can provide the mechanism for such protection.

Secure Messaging If such techniques can be realized, it is natural to ask whether they apply to a domain where surveillance is even more relevant, namely secure messaging. In secure messaging, and in particular end-to-end encrypted (E2EE) communication, the possibility of a state-level or institutional actor monitoring, intercepting, or mandating disclosure of private communications is both realistic and concerning. Ongoing discussions and proposals seek to weaken E2EE to allow inspection of user messages. The stated aim is surveillance of communications, often presented as crime prevention. If these measures are adopted, confidentiality would no longer be guaranteed and private communication could be widely observed, allowing governments or service operators to inspect content or metadata at scale and turn private exchanges into reviewable records. The problem can be stated simply: surveillance of E2EE communication by breaking its confidentiality. The corresponding question is equally direct: can a messaging system still support secure and undetectable communication when its intrinsic E2EE is weakened or broken. To study this concretely, the Signal protocol is taken as the target due to its widespread use in systems such as WhatsApp, Signal, Facebook Messenger, and Google Messages. Because these threats involve powerful adversaries capable of compelling disclosure of cryptographic secrets, they provide a natural setting to investigate whether anamorphic encryption can offer a hidden channel of resistance. The societal and political impact of such threats makes secure messaging a highly relevant domain for examining whether anamorphism can mitigate surveillance.

1.3 Research Objectives/Questions

RQ How can e-exams address coercion, and can solutions for dictatorship-style threats from other domains be adapted?

RQ1 How should coercion be modeled formally in e-exams, and which properties and protocol designs satisfy this model?

RQ2 How can a voter signal their true intention to an honest party in an election without violating protocol rules? Which definitions and primitives are needed

to support this?

RQ3 In the Double Ratchet (as implemented by Signal), if E2EE is compromised, can covert communication persist without detectable changes?

1.4 Contributions

This thesis makes four main contributions, each corresponding to a published, accepted, or submitted paper:

1. **Secure Internet Exams Despite Coercion:** We present the first e-exam protocol that explicitly models and mitigates both candidate and examiner coercion. The design employs mixnet-based pseudonyms, partitioned question assignment, and bulletin-board transparency to preserve privacy and authenticity even in the presence of collusion. We formalize and verify properties including *Anonymous Submission* and *Single-Blindness*, together with their coercion-resistant variants, under a range of dishonest participant configurations.
2. **Secure and Verifiable Coercion-Resistant Electronic Exam:** We extend the coercion-resistance model for e-exams to capture adversaries capable of learning all participant secrets. We show that a recent coercion-resistant protocol fails under this stronger threat model, and we introduce a new design that employs *untappable channels*—a concept absent from the original protocol—to retain coercion resistance while also providing verifiability. In addition, the new protocol benefits from a more precise *mixnet* model recently introduced in the literature as a fix for the anonymity attack we identified in our previous work. Unlike the original design, which relied on an abstract mixnet model, we incorporate this improved model directly into our formal analysis. The protocol is formally analyzed in ProVerif and shown to satisfy both traditional privacy guarantees and the newly defined notions from prior work.
3. **Anamorphic Voting: Ballot Freedom Against Dishonest Authorities:** We introduce the notion of *ballot freedom*, enabling a voter to covertly communicate their genuine choice to a trusted auditor even when authorities control all communication channels. We realize ballot freedom by composing primitives already present in deployed voting systems, and present the first anamorphic constructions built from such combi-

nations. The approach preserves compatibility with existing platforms such as IVXV, CHVote, Helios, and Belenios.

4. **Crypto Wars in Secure Messaging: Covert Channels in Signal Despite Leaked Keys:** We design two practical anamorphic channels for the Signal Double Ratchet, embedding payloads in Diffie–Hellman keys or in MAC tags, while keeping messages indistinguishable from normal Signal traffic. We require no server changes, implement in the official Signal libraries, and observe low overhead in tests. Our optimized DH variant supports 32 covert bits per ratchet step, and the MAC-based channel supports up to 64 bits per message. We provide formal security in the Double Ratchet setting and prove indistinguishability under lawful-access and key-disclosure models.

1.5 Thesis Outline

Chapter 2 presents my first work on coercion-resistance e-exams. This chapter is a reprint of the following published paper [37].

Mohammadamin Rakeei, Rosario Giustolisi, and Gabriele Lenzini. **Secure internet exams despite coercion.** In: *International Workshop on Data Privacy Management*. Springer. 2022, pp. 85–100.

Chapter 3 presents my second work on coercion-resistance e-exams. This chapter is a reprint of the following accepted paper [38].

Mohammadamin RAKEEI et al. **Secure and Verifiable Coercion-Resistant Electronic Exam.** English. In: *Journal of Cybersecurity* (In press). HDL: <https://orbilu.uni.lu/10993/67137>.

Chapter 4 presents my first work on anamorphic encryption, more specifically, anamorphic voting. This chapter is a reprint of the following accepted paper [39].

Rosario Giustolisi, Mohammadamin Rakeei, and Gabriele Lenzini. **Anamorphic Voting: Ballot Freedom Against Dishonest Authorities.** In: *Financial Cryptography and Data Security. FC 2025 International Workshops*. Cham: Springer Nature Switzerland, 2026, pp. 266–280.

Chapter 5 presents my second work on anamorphic encryption, more specifically, anamorphism in Signal Double Ratchet. This chapter is a reprint of the following submitted paper [40].

Mohammadamin Rakeei et al. **Crypto Wars in Secure Messaging: Covert Channels in Signal Despite Leaked Keys**. Cryptology ePrint Archive, Paper 2025/2172. 2025. URL: <https://eprint.iacr.org/2025/2172>.

Chapter 6 I present the key findings, reflect on the state of the research after my contributions, and discuss the challenges and limitations of each topic. I also outline possible improvements and future directions, and I close the thesis with final remarks.

Chapter 2

CRex: Secure Internet Exams Despite Coercion

Abstract	16
2.1 Introduction	16
2.2 Related Work	18
2.3 Background	19
2.3.1 Security Properties of an e-Exam	19
2.3.2 Coercion resistant e-Exams.....	19
2.4 Use Case and a Relevant Attack on Mixnet	21
2.4.1 Remark! in a nutshell.....	21
2.4.2 Intermezzo: An attack and a fix on Remark!'s mixnet	22
2.4.3 A Mixnet resilient to Injection Attacks	23
2.5 Security Formal Analysis	23
2.5.1 Threat Model and Formalization of the New Properties.....	25
2.6 A Secure Coercion Resistant Exam Protocol	27
2.7 Conclusion	31

Abstract

We study coercion-resistance for online exams. We propose two new properties, Anonymous Submission and Single-Blindness which preserve the anonymity of the links between tests, test takers, and examiners even when the parties coerce one another into revealing secrets. The properties are relevant: not even Remark!, a secure exam protocol that satisfies anonymous marking and anonymous examiners, results to be coercion resistant. Then, we propose a coercion-resistance protocol which satisfies, in addition to known anonymity properties, the two novel properties we have introduced. We prove our claims formally in ProVerif. The paper has also another contribution: it describes an attack (and a fix) to an exponentiation mixnet that Remark! uses to ensure unlinkability. We use the secure version of the mixnet in our new protocol.

2.1 Introduction

One of the most tangible consequences of the Corona virus pandemic in education has been that academic institutions moved exams to the Internet. The migration to an online format did aggravate the problem of fraud, which is now a concern for all institutions worldwide [41].

Cheating at exams, *i.e.*, taking advantage of the process for one's own benefit, is a practice as old as the establishment of exams, and it is unsurprising that candidates try to cheat; However, the use of Information and Communication Technologies (ICT) in exams makes it easier and more appealing even for the authorities to temper with the exam processes. Authorities have been found to increase the grades in order to boost their university national ranking ¹; More recently, a famous legal firm has been fined 100M \$ because hundreds of auditors at the firm cheated at ethics tests required to keep their professional license ².

With electronic exams (or e-exams), the implementation of traditional anti-cheating mitigation actions is harder and less scalable than in traditional pencil-and-paper exams. Frauds perpetrated by attacking the underline communication infrastructure are subtle and

¹Valerie Strauss, "Remember the Atlanta schools' cheating scandal? It is not over", 1 February 2022, [Washington Post](#), accessed on 2022/06

²Tory Newmyer, "Ernst & Young hit with a 100 million fine over cheating on ethics tests", 28 June 2022, [Washington Post](#), accessed 2022/07

hard-to-detect. The use of ICT in exams, and the migration of them online, may compromise the quality of education assessment unless we can provide exam protocols that are *secure by-design*.

Security in e-exam is not a new topic. Previous research, shows that is possible to formally express and study properties such as anonymous marking, question secrecy, and mark integrity [42]; that e-exam protocols can preserve privacy despite curious authorities [11]; and that e-exams can be designed to be *verifiable* [12], and *privacy-preserving verifiable* [43].

Contribution This work contributes to the state of the art in e-exam security, in several ways. First, it studies two new properties meant to capture *coercion resistance*, a property never studied in e-exams. The new properties are called *Anonymous Submission* and *Single-Blindness*. The former, expresses that examiners cannot get to know the link between candidates (*i.e.*, test takers) and the tests that they had submitted, even if examiners force the candidates to reveal their secrets (*i.e.*, private keys); the latter, says that candidates cannot learn the link between examiners and the tests that the examiners had marked, even if they compel examiners to reveal their secrets. *Anonymous Submission* and *Single-Blindness* are novel, although they may resemble to *Anonymous Marking* (marks remain secret until notification) and *Anonymous Examiner* (test takers ignore the identities of examiners). To answer the question, we select a state-of-the-art secure exam protocol, Remark! [10], which uses the exponentiation mixnet proposed by Haenni-Spycher [44] to preserve anonymity and unlinkability, and that satisfies *Anonymous Marking* and *Anonymous Examiner* even if candidates and examiners can collude. Under collusion, Remark! does not satisfy *Anonymous Submission* and *Single-Blindness*, supporting that *Anonymous Submission* and *Single-Blindness* are unprecedented.

Besides, we propose a new *coercion-resistant exam protocol*, which we call CREX. It satisfies *Anonymous Submission* and *Single-Blindness* under coercion, and we prove this statement formally using Proverif [8], a model checker. We also verify that CREX satisfies the other security properties already met by Remark!.

This work has another orthogonal, but not less important, contribution. We discover that by injecting specific messages into the *mixnet* used by Remark!—the Haenni-Spycher’s exponentiation mixnet— an attacker can break unlinkability. An immediate consequence is that, in Remark!, various anonymity properties no longer hold, including *Anonymous*

Marking and *Anonymous Examiner*. We discuss how to secure the exponentiation mixnet so that to restore *Anonymous Marking* and *Anonymous Examiner* in Remark!. We believe that our fix also improves the security of Haenni-Spycher’s mixnet in other domains than the online assessment, a statement that will be investigated in future work.

2.2 Related Work

Although a discussion about security in e-exams has recently gained attention, the topic is not new and has been researched before. For a fairly extensive list of requirements for e-exams, we refer the reader to [42]; for a brief account, we recall that security requirements for e-exams have been informally expressed in [45, 46], while [12] defines a formal framework in the applied π -calculus to specify and analyze authentication and privacy in e-exams; in addition, the authors define and show how to prove a set of verifiability properties for exams [47]. In a similar domain, that of computer-supported collaborative work, [48] formalizes confidentiality, proposing exams as a case study.

In the state-of-the-art, we find not only works that describe properties for secure and private assessment, but also proposals for new security protocols for computer-assisted exams. [49] designs a protocol that meets authentication and privacy properties in the presence of a fully trusted exam manager. [6] proposes an e-exam, which considers a corrupted examiner, but assumes an honest-but-curious anonymiser. This assumption was later removed in [11].

[23] tackles on-line exams and discusses an Internet-based exam protocol with few trust requirements on principals. [10] describes *Remark!*, another Internet-based exam protocol that ensures authentication and conditional anonymity requirements with minimal trust assumptions; this is the protocol we refer to here as use case.

The interest in e-exams is not limited to researchers. Many institutions for language proficiency tests, for example the Educational Testing Service (ETS) ³, or organizations for personnel selection, for instance Pearson⁴ or the EU EPSO ⁵, offer computer-assisted and online testing.

Some related protocols have been proposed in the area of conference management systems. Our proposed properties can be seen as double-blind review properties, typically

³<https://www.ets.org/>

⁴<https://www.pearsonassessments.com/>

⁵<https://epso.europa.eu/>

requested in the submission of conference papers. [50] introduced *CoCon*, a formally verified implementation of a conference management system that guarantees confidentiality. Arapinis *et al.* introduced and formally analysed *ConfiChair*, a cryptographic protocol that addresses secrecy and privacy risks coming from a *malicious-but-cautious* cloud. Their work has been extended to support any cloud-based system that assumes honest managers, such as the public tender management and recruitment process [24]. All such works do not mention or consider coercion resistance, a class of properties that, instead, we study here for e-exams.

2.3 Background

Exam procedures differ one another in the details, but they all share the same organization and information flow at a certain level of abstraction. They are generally organized in four distinct phases: *Registration*, *Examination*, *Marking*, and *Notification*. Several roles are involved: *candidates*, the test takers; *question committee*, which prepares the exam questions; *examiners*, those who mark the tests; and *exam authorities*, a set of potentially distinct agents that help *e.g.*, distribute the test to students, assign examiners, and notify marks.

2.3.1 Security Properties of an e-Exam

An e-exam protocol is expected to guarantee specific security properties. The ones this work refers to are: (i) *Test Answer Authentication* (only tests submitted by eligible candidates are collected); (ii) *Examiner Authentication* (only authenticated examiners are allowed to marking tests); (iii) *Mark Privacy* (the marks given to a candidate's test remain unknown to the other candidates); (iv) *Anonymous Marking* (examiners do not learn the owner of a test they are marking, until after the markings are notified); (v) *Anonymous Examiner* (candidates do not learn the identity of the examiners who marked their tests). To this list, we study *Anonymous Submission* and *Single-Blindness* (see Sec. 2.3).

2.3.2 Coercion resistant e-Exams

Since no one studied coercion-resistance *in e-exams* (the property has been studied extensively but in other domain, for instance, e-voting), we propose our interpretation. In layman terms, it means that an exam protocol should preserve privacy between exami-

nees and examiners even when they force one another into revealing secrets the lead to de-anonymization, such as, private keys.

The threat is not negligible considering that e-exams are used not only for university grading but also, for instance, in national and international educational assessments like for instance, the PISA⁶; the PIAAC⁷; the various Tests for English as a Foreign Language required to get US VISA and studentships. Here, the benefits at stake are higher and so are the incentives to stress the procedure for one's own profit.

To address coercion resistance, we propose two additional security requirements for e-exams, *Anonymous Submission* and *Single-Blindness*. *Anonymous Submission* states that an examiner cannot learn which answer is submitted by which candidate. At first glance, it sounds similar to *Anonymous Marking*, as both aim to achieve unlinkability between the identity of a candidate and the answer that she submitted. The difference is that the definition of *Anonymous Submission* is based on the indistinguishability of the answers submitted by two known candidates, while the definition of *Anonymous Marking* is based on the indistinguishability of the candidates' identities who submitted two known answers. As shall we see in Section 2.5.1, this is a key difference that allows us to model coercion in an e-exam by starting from the definition of *Anonymous Submission*. The property is relevant when an examiner tries to coerce candidates to reveal their private keys, or the answers they submitted.

Single-Blindness ensures that no candidate can learn which mark has been assigned by an examiner. This property recalls *Anonymous Examiner*, as they both aim to achieve unlinkability between the identity of an examiner and the mark he provided. However, while the definition of *Anonymous Examiner* is based on the indistinguishability of the examiners' identities who provided a mark for two known tests, *Single-Blindness* is based on the indistinguishability of the marks provided by two known examiners. Here, the main difference is that *Single-Blindness* does not consider the tests at all. The property is relevant when a candidate attempts to compel an examiner to reveal a private key to get access to the marks he has reported to the exam authority.

⁶The Programme for International Student Assessment

⁷Programme for the International Assessment of Adult Competencies

2.4 Use Case and a Relevant Attack on Mixnet

We are interested to verify whether our new properties are novel and relevant, and not other properties renamed. Thus, we pick a state-of-the-art exam protocol Remark! [10]—an internet-based exam that preserves all the properties listed in the previous section, including *Anonymous Marking* and *Anonymous Examiner* and study our new properties against it. In Remark!’s threat model, all parties are malicious and can collude. It seems a good choice to start with to explore resilience against coercion.

2.4.1 Remark! in a nutshell

It is organized into four phases: at *Registration* private/public keys and pseudo-identities are generated and distributed to candidates and examiners using an exponentiation mixnet; during *Examination*, questions are anonymously distributed (relying on a bulletin board) to candidates; the exam authority receives the answers submitted by the candidates; at *Marking* the exam authority anonymously distributes the tests to examiners; at *Notification*, the exam authority records the marks and notifies the candidates their results. Dreier *et al.* in [12] formally proved that Remark! satisfied *Test Answer Authentication*, *Mark Privacy*, *Anonymous Marking* and *Anonymous Examiner*.

Remark! relies on the Haenni-Spycher’s exponentiation mixnet [44] to generate pseudo-identities for examiners and candidates.

Haenni-Spycher’s Exponentiation Mixnet

[44] proposed a structure called *verifiable shuffling mixnet* or *exponentiation mixnet*, and designed an e-voting protocol based on it. The aim of the mixnet is to generate a list of pseudonyms from a list of public keys in a way that only the owner of a public key can find the corresponding pseudonym in the output list. This mixnet construction is used in Remark! to output a list of pseudo-public keys (pseudonyms) from an input list of public keys, while preserving unlinkability between input and output lists. To do so, the exam authority sends the list of eligible candidates’ and examiners’ public keys to the mixnet. Upon receiving the list of public keys the mixnet generates the output list as depicted in Figure 2.1. In this structure, $PK_i = g^{SK_i}$, $\overline{PK}_i = h_C^{SK_i}$, $\bar{r}_k = \prod_{i=1}^k r_i$ and $\bar{\pi}_k = \pi_k \circ \dots \circ \pi_1$, where SK_i , r_k and π_k are, respectively, the private key of party i_{th} , the secret element of the mixnet k_{th} , and the secret permutation of the mixnet k_{th} . For a more detailed description,

	mix_1	mix_2	mix_m
C_1	PK_1	$PK_{\pi_1(1)}^{\bar{r}_1}$	$PK_{\pi_m(1)}^{\bar{r}_m} = \overline{PK}_1$
C_2	PK_1	$PK_{\pi_1(2)}^{\bar{r}_1}$	$PK_{\pi_m(2)}^{\bar{r}_m} = \overline{PK}_2$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
C_n	PK_n	$PK_{\pi_1(n)}^{\bar{r}_1}$	$PK_{\pi_m(n)}^{\bar{r}_m} = \overline{PK}_n$
	g	$g^{\bar{r}_1}$	$g^{\bar{r}_m} = h_C$

Figure 2.1: Pseudonym generation using the exponentiation mixnet. All the terms within the box are published on the bulletin board.

see [44].

2.4.2 Intermezzo: An attack and a fix on Remark!’s mixnet

Before proceeding further, we have to comment on a finding that we discovered while reflecting on the role of the mixnet in preserving anonymity and unlinkability. It is a new attack against the implementation of Haenni-Spycher’s mixnet used in Remark! that compromises anonymity and unlinkability, and which we have to fix before proceeding further.

The attack allows an attacker to link any public key to its corresponding pseudonym. Let $L = \{g^{t_1}, \dots, g^{t_n}\}$ be a list of n values where g is the generator of a multiplicative subgroup G_q of prime order q and $t_i \in \mathbb{Z}_q$, $1 \leq i \leq n$.

Let us assume that E is a party that receives L and is requested to send it to the mixnet. We show that a malicious E is capable of deanonymizing an arbitrary element of L by adding an additional input to the list. E first chooses a random number $s \in \mathbb{Z}_q$ and computes g^s . Then, she selects the pseudonym she wants to deanonymize, say $g^{t_i} \in L$, it inserts $g^{t_i} \cdot g^s$ into L , and sends L to the mixnet. The mixnet returns $L' = \{L'_1, \dots, L'_{n+1}\} = \{g^{t_1 \cdot r}, \dots, g^{t_n \cdot r}, (g^{t_i} \cdot g^s)^r\}$ as output and publicly publishes g^r . Now, E computes $(g^r)^s$ and searches in L' for the element $g^{rs} \cdot L_j$. That element is the pseudonym that links to t_i .

The attack exploits the fact that $g^{rs} \cdot g^{t_i \cdot r} = (g^{t_i} \cdot g^s)^r$. It lets E learn $L'_j = g^{t_i \cdot r}$ which is the corresponding output for the chosen input g^{t_i} without E knowing the secret element t_i . If E runs it for each element, in $O(n^2)$ time, he can learn all the n links between L and L' .

It is worth to stress that if the mixnet has a procedure to check the eligibility of the input public keys run by E , the attack is possible if E is untrustworthy, which is usually a health assumption (*i.e.*, better not to have trusted parties): E can violate this check and inputs

a manipulated public key. If we assume that the mixnet has no eligibility check feature for inputs, then the attack can even be done by any external attacker who only knows a member from L .

The attack mentioned above is independent of the system in which the mixnet is used. Any protocol that relies on the Haenni-Spycher’s exponentiation mixnet is vulnerable to this injection attack. In Remark!, exam authority can launch this attack to learn the link between public keys and their associated pseudonyms that violates *Anonymous Marking* and *Anonymous Examiner*.

2.4.3 A Mixnet resilient to Injection Attacks

To secure the mixnet against our linkability attack, one possible solution is to prevent the injection of biased public keys into the mixnet. We propose an *Injection-Resistant Exponentiation Mixnet* setup based on Haenni-Spycher’s mixnet, where the input public keys should be accompanied by their Zero Knowledge Proofs of Knowledge (ZKPKs) for their corresponding private keys. The rest of the structure remains unchanged.

With this simple fix, the mixnet only accepts a public key that is associated with a verified ZKPK and aborts otherwise.

2.5 Security Formal Analysis

Having fixed the mixnet construction, a question remains open: does Remark! with the stronger mixnet (for reference, we call it Injection-Resistant Exponentiation Mixnet (IRemix)) satisfy *Anonymous Submission* and *Single-Blindness* even if we assume that the parties can coerce one another? If that were true, *Anonymous Submission* and *Single-Blindness* could be simply achieved by ensuring unlinkability. We could even suspect that *Anonymous Submission* and *Single-Blindness* are implied by *Anonymous Marking* and *Anonymous Examiner*. Fortunately, this is not the case, as we argue later in this section, after we set up the formal framework where we perform our security analysis.

Formalizing e-Exams. Since we aim at a formal verification, we need to model an e-exam protocol in a formal language. We choose the applied π -calculus [51] for the task, and we refer to the strategy advanced by Dreier *et al.* in [12], which we remind here.

Definition 1. (E-exam protocol) [12]. An e-exam protocol is a tuple $(C, E, Q, A_1, \dots, A_l, \tilde{n}_p)$, where C is the process executed by the candidates, E is the process executed by the examiners, Q is the process executed by the question committee, A_i 's are the processes executed by the authorities, and $\tilde{n}_p$ is the set of private channel names.

Note that all candidates and all examiners execute the same process, but with different variable values, e.g., keys, identities, and answers.

Definition 2. (E-exam instance) [12]. An e-exam instance is a closed process $EP = v\tilde{n}.(C\sigma_{id_1}\sigma_{a_1} \mid \dots \mid C\sigma_{id_j}\sigma_{a_j} \mid E\sigma_{id'_1}\sigma_{m_1} \mid \dots \mid E\sigma_{id'_k}\sigma_{m_k} \mid Q\sigma_q \mid A_1\sigma_{dist} \mid \dots \mid A_l)$, where $\tilde{n}$ is the set of all restricted names, which includes the set of the protocol's private channels; $C\sigma_{id_i}\sigma_{a_i}$'s are the processes run by the candidates, the substitutions σ_{id_i} and σ_{a_i} specify the identity and the answers of the i^{th} candidate respectively; $E\sigma_{id'_i}\sigma_{m_i}$'s are the processes run by the examiners, the substitution $\sigma_{id'_i}$ specifies the i th examiner's identity, and σ_{m_i} specifies for each possible question and answer pair the corresponding mark; Q is the process run by the question committee, the substitution σ_q specifies the exam questions; the A_i 's are the processes run by the exam authorities, the substitution σ_{dist} determines which answers will be submitted to which examiners for grading. Without loss of generality, we assume that A_1 is in charge of distributing the copies to the examiners.

Definition 2 does not specify whether the examiners are machines or humans. For the purpose of our model this distinction is not necessary; it is sufficient that an examiner attributes a mark to a given answer.

Note that Q and A_1 could coincide if for instance there is only one authority A , in that case we can write simply $A\sigma_q\sigma_{dist}$ instead of $Q\sigma_q \mid A_1\sigma_{dist}$.

Model Checking and equational theories. We use ProVerif [8] for the analysis of e-exam protocols. ProVerif allows one to analyze reachability and equivalence-based properties in the symbolic attacker model. We chose ProVerif mainly because it has been extensively used to analyze exam protocols [42], hence we could easily check formerly defined security properties for exams. ProVerif is also one of the few tools that allows for automated analysis of privacy properties using observational equivalence; therefore, we can check *Anonymous Submission* and *Single-Blindness* in our protocol automatically. The input language of ProVerif is the applied π -calculus [51], which the tool automatically translates to Horn clauses. Cryptographic primitives can be modeled by means of equational theories. An equational theory E describes the equations that hold on terms built from the signature.

Terms are related by an equivalence relation $\approx$ induced by E . For instance, the equation $dec(enc(m, pk(k)), k) = m$ models an asymmetric encryption scheme. The term m is the message, the term k is the secret key, the function $pk(k)$ models the public key, the term enc models the encryption function, and the term dec models the decryption function. The list of all equational theories used to model the Remark! protocol can be found in [12].

2.5.1 Threat Model and Formalization of the New Properties

We let the attacker read all public data on the bulletin board and impersonate misbehaving parties, including an unbounded number of dishonest examiners when checking *Anonymous Submission*, and an unbounded number of dishonest candidates when checking *Single-Blindness*. In addition, we check both properties under a coercion scenario, meaning that the coerced candidate (resp. examiner) reveals their secrets before notification.

Privacy properties can be modeled as observational equivalence properties. To model *Anonymous Submission*, we consider two honest candidates and an unbounded number of dishonest examiners; to model *Single-Blindness*, we consider two honest examiners and an unbounded number of dishonest candidates. We check both *Anonymous Submission* and *Single-Blindness* considering an honest exam authority. We also check *Anonymous Marking* and *Anonymous Examiner* considering a dishonest exam authority.

First, we check whether *Anonymous Submission* holds in our protocol: specifically, that even if two honest candidates swap their answers in two different runs of the protocol then the attacker cannot distinguish the two resulting systems.

Definition 3 (*Anonymous Submission*). *An exam protocol ensures Anonymous Submission if any exam process EP , any two candidates id_1 and id_2 , and any two answers a_1 and a_2 .*

$$EP_{\{id_1, id_2\}}[C\sigma_{id_1}\sigma_{a_1} | C\sigma_{id_2}\sigma_{a_2}]|_{\text{marked}} \approx_l EP_{\{id_1, id_2\}}[C\sigma_{id_1}\sigma_{a_2} | C\sigma_{id_2}\sigma_{a_1}]|_{\text{marked}}$$

The difference between *Anonymous Submission* and *Anonymous Marking* is that the latter considers two honest candidates who swap their secret keys in two different runs. While both properties aim at hiding the link between candidate's key and answer, the definition of *Anonymous Submission* crucially enables a definition of coercion-resistance. To model this, we additionally let the candidates publish their answers and their secret keys on the public channel, and verify that *Anonymous Submission* still holds:

$$EP_{\{id_1, id_2\}}[C\sigma_{id_1}\sigma_{a_1} | C\sigma_{id_2}\sigma_{a_2}]|_{\text{marked}} \approx_l EP_{\{id_1, id_2\}}[C' | C\sigma_{id_2}\sigma_{a_1}]|_{\text{marked}}$$

where C' is a process such that $C'^{\text{out}(chc, \cdot)} \approx_l C\sigma_{id_1}\sigma_{a_2}$, i.e. C' is the process that acts like one submitting answer a_2 , but pretends to cooperate with the attacker by revealing their secrets through channel chc .

Then, we check whether *Single-Blindness* holds in our protocol. We check that if two honest examiners swap their marks in two different runs of the protocol, then the attacker cannot distinguish the two resulting systems.

Definition 4 (*Single-Blindness*). *An exam protocol ensures Single-Blindness if for any exam process EP , any two examiners id_1 and id_2 , any two marks m_1 and m_2*

$$EP_{\{id_1, id_2\}} [E\sigma_{id_1}\sigma_{m_1} | E\sigma_{id_2}\sigma_{m_2}] \approx_l EP_{\{id_1, id_2\}} [E\sigma_{id_1}\sigma_{m_2} | E\sigma_{id_2}\sigma_{m_1}]$$

As for *Anonymous Submission*, we also check that *Single-Blindness* holds under examiner coercion. We let the examiners publish their marks and secret keys on the public channel:

$$EP_{\{id_1, id_2\}} [E\sigma_{id_1}\sigma_{m_1} | E\sigma_{id_2}\sigma_{m_2}] \approx_l EP_{\{id_1, id_2\}} [E' | E\sigma_{id_2}\sigma_{m_1}]$$

where E' is a process such that $E'^{\text{out}(chc, \cdot)} \approx_l E\sigma_{id_1}\sigma_{m_2}$.

The difference between *Single-Blindness* and *Anonymous Examiner* is that the latter considers two honest examiners who swap their keys in two different runs. While both properties aim at hiding the link between key and mark, the definition of *Single-Blindness* enables the definition of coercion-resistance.

Security Findings. Under the only threat of collusion, ProVerif proves that both properties hold in Remark!. However, under coercion threats, it finds attacks.

Anonymous Submission does not hold under candidate coercion, because the exam authority publishes on the bulletin board the answer of a candidate along with their pseudo public keys (see step 6 in [10]). ProVerif shows an attack trace in which a coercer, who knows the secret key of a coerced candidate, finds out the candidate's answer by retrieving their pseudo public key from the output of the exp. mixnet (see step 1 in [10]).

Single-Blindness does not hold under examiner coercion either, because, at notification, the mark and the answer are signed with the examiner's pseudo signing key (see step 8 in [10]). ProVerif shows an attack trace in which a coercer, finds out which answers have been marked by a coerced examiner by retrieving their pseudo signing key from the output of the exp. mixnet (see step 2 in [10]). Table 2.1 summarizes the findings.

2.6 A Secure Coercion Resistant Exam Protocol

We now present our new protocol, CREX, whose goal is to guarantee all the properties outlined above. CREX has four main phases *Registration*, *Examination*, *Marking*, and *Notification* and one additional phase which is run before the exam begins. We assume that each test consists of at least two questions. We also assume that there is an append-only bulletin board (BB), which records data concerning examiners, candidates and exam authority. The bulletin board is used to publish public parameters and is also needed for verifiability guarantees. Moreover, we assume that each principal has a pair of Elgamal public/private key which used in an Elgamal cryptosystem [52]. The five phases of CREX are depicted in Figure 2.2 and are as follows:

Pre-Assignment: Let us assume n candidates and d examiners. Before starting the exam, the exam authority forms $A = \{1, \dots, n\}$ and then partitions A into d subsets as $A_p = \{A_{p_1}, \dots, A_{p_d}\}$, and labels them as $P = \{P_1, \dots, P_d\}$. The exam authority sends A_p and P to each examiner through a secure channel. The examiners sign the partitions and send them back to exam authority, which distributes the signatures among all examiners so that each examiner can check if they have received the same message as the others.

Registration: This phase is as the registration phase in Remark!, but with the IRemix explained in Section 2.4. In this phase, we create pseudo public keys (pseudonyms) as outputs from a list of public keys as inputs while preserving the unlinkability between input and output lists. exam authority sends the list of public keys of eligible candidates and examiners and the list of corresponding *Zero-Knowledge Proofs of Knowledge* (ZKPKs) of those keys to the mixnet. Each candidate and examiner, during the authorization and eligibility checks, provides the ZKPK of their keys. When receiving the list of public keys *accompanied by their ZKPKs*, the mixnet checks the proofs and generates the output list in the same fashion as in Figure 2.1.

Testing: The exam authority signs and encrypts the questions with the candidate's pseudonym and posts them on the bulletin board. The candidate decrypts the test, checks the signature, and answers the questions. Then she sends $[Sign_{SK_C, h_C}(quest, ans, \overline{PK}_C)]_{PK_A}$ where SK_A , PK_A , $quest$, ans and $\overline{PK}_C$ are the private key of exam authority, the public key of exam authority, the answers, and the candidate pseudonym, respectively. Furthermore, exam authority sends the candidate $[Sign_{SK_A}H(quest, ans, \overline{PK}_C, \alpha)]_{PK_A}$ where H and α are a secure hash function and a random value generated by exam authority.

Marking: In this phase, we introduce a new technique called *Shuffled Answers* to assign

the collected tests to the examiners in a way that guarantees *Anonymous Submission* and *Single-Blindness* also in case of coercion. The idea behind this technique is that when the *Marking* phase starts, each examiner receives a test where each $(question, answer)$ pair belongs to a random candidate. After collecting the tests from the candidates, exam authority forms a matrix named T , which consists of all candidates' $(question, answer)$ pairs. Then it chooses a secure permutation matrix named Π and applies it to T to form a new matrix T^π , that is $\Pi(T) = T^\pi$. The following transform mathematically expresses the shuffling procedure. Here, if we assume k questions and n candidates, $(q, a)_{(i,j)}$ is the answer of candidate j to question i .

$$T = \begin{bmatrix} (q, a)_{1,1} & \cdots & (q, a)_{1,n} \\ \vdots & & \vdots \\ (q, a)_{k,1} & \cdots & (q, a)_{k,n} \end{bmatrix} \xrightarrow{\Pi} \begin{bmatrix} (q, a)_{(1,\pi_1^1)} & \cdots & (q, a)_{(1,\pi_n^1)} \\ \vdots & & \vdots \\ (q, a)_{(k,\pi_1^k)} & \cdots & (q, a)_{(k,\pi_n^k)} \end{bmatrix} = T^\pi$$

Let us define T and T^π as sets of n vectors $T = [V_1, \dots, V_n]$ and $T^\pi = [V_1^\pi, \dots, V_n^\pi]$, respectively. V_j , $1 \leq j \leq n$ shows the test of candidate j , while V_j^π , $1 \leq j \leq n$ represents a new test whose each question belongs to a random author. exam authority signs T^π with its private key and publishes it on the bulletin board in front of its public key. Then exam authority randomly assigns each examiner a member of P and posts this assignment on the bulletin board. Therefore, each examiner sees a label in front of their pseudonym on the bulletin board that means which subset they should mark. Let us name P_E the label assigned to the examiner E . E grades the corresponding assigned vectors and, for each, sends the message $[Sign_{SK_E, h_E}(M_j^\pi, A_{P_E})]_{PK_A}$ to exam authority where M_j^π is the vector of marks associated with V_j^π and $j \in A_{P_E}$.

Notification: Let $S_j^\pi = Sign_{SK_E, h_E}(M_j^\pi, A_{P_E})$. When the exam authority receives all messages from all examiners, first it checks if the examiners marked the correct assigned subsets, and then it constructs two $S^\pi = [S_1^\pi, \dots, S_n^\pi]$ and $M^\pi = [M_1^\pi, \dots, M_n^\pi]$ matrices. Then, it applies the inverse permutation Π^{-1} to these matrices and generates $S = [S_1, \dots, S_n]$ and $M = [M_1, \dots, M_n]$, respectively. M is the matrix of final marks and each column of it, which means M_j , is a vector showing the marks for the j^{th} candidate's test. S represents a notification matrix which includes the marks signed by the eligible examiners. The exam authority signs and encrypts each column of S with the pseudonym of the corresponding candidate and then posts the output on the bulletin board. Finally, the mixnet servers

reveal their secret exponents that are used to anonymize the candidates. The anonymity of the candidate is revoked, and the exam authority can register the marks and reveal the secret element α at the end of the exam.

Let us now assume that the coercer has asked a candidate to reveal her private key and her submitted test. The coerced candidate first looks at T^π and, from each row, picks an arbitrary pair of $(question, answer)$. Then, she reveals her real private key, as well as the set of $(question, answer)$ pairs chosen. We claim that the coercer cannot distinguish whether the candidate has demonstrated her real test or a fake one. Instead, let us assume an examiner E_j , $1 \leq j \leq d$ who is coerced by a candidate and is supposed to mark the tests labeled $P_r \in P$. In addition to P_r , E_j marks $P_{r'}$ where $P_{r'} \in P$ and $r \neq r'$. Now, if the coercer asks E_j to reveal her secret keys, E_j pretends that she has marked only the tests labeled $P_{r'}$. Our assertion is that the coercer cannot distinguish whether E_j lies about her assigned partition.

Security Formal Analysis ProVerif proves that our protocol meets both *Anonymous Submission* and *Single-Blindness*, including the coercion alternatives. Our protocol resists the attack on candidate coercion seen in Remark! because the exam authority re-randomizes the pseudo public key of the candidate. Single-Blindness is proved also under examiner coercion because, differently from Remark!, a coerced examiner can lie to a coercer by claiming that the examiner is marking a different partition of tests.

In addition to *Anonymous Submission* and *Single-Blindness*, we prove in ProVerif that our protocol meets all the original properties meet by Remark!, namely, anonymous marking, anonymous examiner, test answer authentication and examiner authentication. Table 2.1 summarizes the findings.

Properties	Remark! (with IRemix)	CREX	Threat Model
Answer Authentication	✓	✓	Dishonest E
Examiner Authentication	✓	✓	Dishonest C
Mark Privacy	✓	✓	Dishonest (C^*, E^*)
Anonymous Marking	✓	✓	Collusion(EA, E)
Anonymous Examiner	✓	✓	Collusion(EA, C)
Anonymous Submission	✓	✓	Dishonest E
Single-Blindness	✓	✓	Dishonest C
Anonymous Submission	✗	✓	Coercion(E)
Single-Blindness	✗	✓	Coercion(C)

Table 2.1: Synthesis of the findings of the formal analysis under different threats.

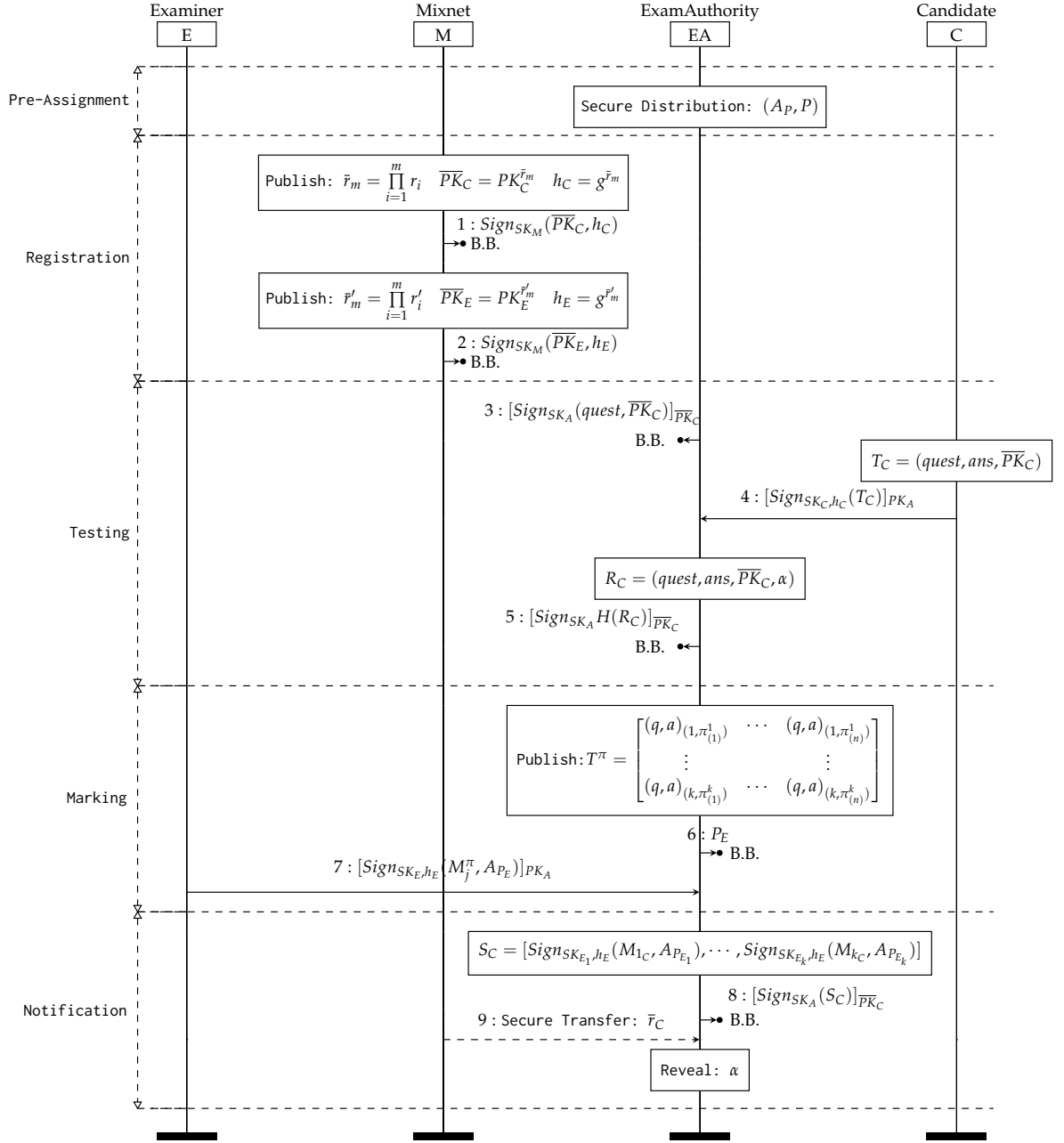


Figure 2.2: The message sequence chart of CReX

Table 2.1 highlights the dishonest parties in our analysis. For *Mark Privacy*, C^* and E^* are, respectively, all candidates and examiners except the candidate and examiner con-

cerned. Collusion is possible when at least two parties misbehave. Therefore, *Anonymous Marking* and *Anonymous Examiner* are discussed in the collusion threat model with respectively *EA & E*, and *EA & C* being dishonest. *Anonymous Submission* and *Single-Blindness* are studied under both coercion and non-coercion scenarios. The ProVerif codes are accessible online ⁸.

2.7 Conclusion

The use of online exams, which peaked during the pandemic, raises issues of security and privacy, since it is easier to cheat when exams are held in remote.

In certain sectors, like in e-voting, security and privacy are already well-established subjects of research. Various academic and industrial collaboration activities have been established, which now support the sectors with ideas, prototypes, forums for discussions, and projects. Compared to e-voting, electronic exams seem underrepresented. One could discuss whether what is at stake in e-voting, aka one important cog in the democracy process, is more important than the quality of assessment of skills and knowledge of people, but recent events show that the government attention to a fair and honest assessment is not second to anything. At the time of writing, giant companies like Ernst & Young, admitted their employees had cheated on ethics exams, an act that cost them a fine of a hundred million USD ⁹. This episode is not isolated. In March 2022, PwC has been sanctioned about one million USD for “having faulty quality control standards that allowed more than 1,200 professionals to cheat on internal training courses” ¹⁰. In a different domain, many universities are struggling to achieve robust online assessment systems, where at stake is the trust that we have in the general reward strategy of our educational systems. It is clear that to adhere to a code of conduct is not a sufficient guarantee for an honest outcome and that we need better, more secure, and private by-design exam protocols.

This work studies a class of exam security requirements missing in the state-of-the-art: coercion resistance. Following this main goal, we obtained several important achievements. First, we formally defined two new properties, *Anonymous Submission* and *Single-Blindness*, which allow one to reason about the phenomenon of coercion in online exams. Thanks to

⁸C-Rex Code Repository

⁹*id. at 2*

¹⁰Soyoung Ho, “Canada for Widespread Employee Cheating on Internal Tests”, [Thomson Reuters](#), 2 March 2022, last access 2022/07

this effort, we find that a state-of-the-art protocol is not coercion resistant in the sense we describe with our properties. Coercion resistance requires a different approach, and we propose a new cryptographic protocol CREX. To our knowledge, it is the first coercion-resistant e-exam protocol: unlike Remark!, it guarantees *Anonymous Submission* and *Single-Blindness* even if the parties can coerce one another into revealing secrets.

CREX implements a new secure exponentiation mixnet which is also an original, although orthogonal, contribution of this work. In fact, while investigating the security of a state-of-the-art e-exam protocol, Remark!, we found an linkability attack on the Haenni-Spycher's mixnet, the main building block of Remark!. The attack completely breaks the claimed unlinkability property of this mixnet. The new injection-resistant exponentiation mixnet which we propose, called IRemix, is fully based on the structure of Haenni-Spycher but add an important steps requiring ZKPKs which, we prove informally, works as a countermeasure against the attack. Formally verifying the security of IRemix remains an important further work. Furthermore, in the *Pre-Assignment* phase, we proposed a simple protocol which assures that all parties have received the same set of partitions. It is desirable for future research to formally verify this protocol.

Future studies could also investigate the security of the systems that used Haenni-Spycher's mixnet as an identity mixer, against the linkability attack that we have found. The first one could be Haenni-Spycher's e-voting protocol [44], which seems, but we need to corroborate this statement, vulnerable to this attack if the election authority is dishonest. Other schemes that might be prone to the attack are [53], [54], [55], and [56]. Since our formal verification of the C-Rex protocol in this work are about authentication and privacy properties, another direction for future research is investigating the verifiability guarantees of C-Rex.

Chapter 3

SCREX: Secure and Verifiable Coercion-Resistant Electronic Exam

Abstract	34
3.1 Introduction	34
3.2 Previous works, Background, and Tools	37
3.2.1 Applied Π -Calculus	39
3.2.2 ProVerif	39
3.3 Formalizing e-Exams and Security Properties	40
3.3.1 E-Exams Modelling	40
3.3.2 Authentication Properties	41
3.3.3 Privacy Properties	42
3.3.4 Verifiability	44
3.3.5 Coercion Properties	46
3.4 A New Verifiable Coercion-Resistant Protocol	47
3.4.1 Registration	48
3.4.2 Assignment Protocol	48
3.4.3 Testing	50
3.4.4 Marking	51
3.4.5 Notification.....	53
3.5 Formal Security Analysis	54

3.5.1	Authentication and Secrecy in SCREX	56
3.5.2	Privacy in SCREX	56
3.5.3	Coercion in SCREX	57
3.5.4	Verifiability in SCREX	57
3.6	Conclusion	59

Abstract

Since they enable efficient assessment of large cohorts of students and test-takers, electronic exams (e-exams) have become popular. However, the transition from pencil-and-paper tests to e-exams comes with challenges: researchers needed to ensure a comparable level of security and privacy as that enjoyed before the transition; at the same time, they have to address threats due to the use of information and communication technology. Research has shown that, for the reason of assessment fairness, e-exams should satisfy a list of peculiar security properties, for instance, about authentication, secrecy, integrity, anonymity, and correctness, including their universal and individual verifiability. Recently, e-exams have been scrutinized for their resistance to collusion and coercion. Subsets of participants have an interest in teaming up, or forcing one another, to gain an unfair advantage over the honest others. In this work, we study coercion-resistance for e-exams. We propose a novel strong definition of coercion where all secrets are leaked to the attacker. Under this threat, we prove that a recent coercion-resistant exam protocol is subject to attacks. We improve the protocol by ensuring that all its properties are maintained and that it is coercion-resistant under the new threat model. Our new protocol is also verifiable, which is a must-have property whenever there is the need to prove that fairness is preserved despite anyone attempting to subvert it. All our claims are formally verified using ProVerif. Notably, our formal verification includes proving the security of a recent exponentiation mixnet framework proposed in the literature.

3.1 Introduction

People's skills and knowledge are generally assessed with tests or exams: universities run tests to select students and follow their progression and proficiency with quizzes;

professional organizations release certifications after applicants have passed exams; and companies resort to IQ and attitudinal tests to decide on recruitment and promotion.

Since the 1990s, exams have relied more and more on information and communication technology. Depending on the degree of dependence, exams can be computer-aided or entirely computer-based, but in general, we talk in both cases of *electronic exams* (in short, e-exams). Electronic exams promise to be better than paper-and-pencil-based testing: they ease exam administration, enable faster display of results, and make identification and authentication of candidates stronger.

However, the hasty adoption of information and communication technology and the development of testing applications have been motivated by efficiency reasons and by educational best practices rather than security. In the professional context of computer-based assessment and testing, the few discussions about security have remained informal, addressing standard systems security issues, such as access control, password strength, and network security.

Only recently, the security of exams have become the subject of research and thus of formal analysis. Dreier *et al.* [12] were the first to define a framework in the applied π -calculus [7] to analyze authentication and privacy, taking two existing electronic exam protocols as use cases. The formal approach to security led to the discovery of numerous attacks, calling for new designs and leading to the definition of peculiar security properties. This also includes individual and universal verifiability, whose formal analysis in traditional and electronic exams has been pioneered in [47]. With time, the set of security requirements got richer: for instance, accountability, anonymous marking, and dispute resolutions were formalized and realized in a prototype by Bella *et al.* [57].

Recently, electronic exam protocols have been studied for their resilience to coercion and collusion [37]. Differently from other domains such as e-voting, where coercion has the goal of directing votes towards specific candidates, in the world of e-exams, coercion happens because of an interest to corrupt certain parties (*e.g.*, examiners), to subvert what is probably the most important high-level and transversal principle guiding any exam: a fair assessment for anyone. Coercion threatens that process, especially when one party tries to change the outcome of the assessment. For instance, a candidate could blackmail an examiner to get higher marks; in anonymous marking, an examiner can force a test taker to reveal his/her identity to bias the assessment in his/her disfavour. Collusion also threatens fairness when a subset of players tries to take advantage of other honest parties, including

the exam administration, to tamper with the markings in diplomas and certifications, assets of extreme importance since, with them, one can get access to jobs or apply for VISAs.

In pioneering the study of coercion in electronic exams, Rekeei *et al.* [37], proposed a protocol called CREX, which they proved coercion-resistance under a threat model where some, but not all, private information are leaked to the attacker/coercer. Getting technical, the security of CREX also relies on a specific assignment sub-protocol, only informally argued to be secure while formally assumed to be so in the formal analysis. These are known limitations, and as we will see later, by refining the model using a concrete instantiation of the assignment protocol, the protocol suffers from attacks.

CREX is surely an interesting and rich protocol, the first one that addresses coercion in exams, but differently from the other protocols proposed previously, it does not ensure verifiability, a class of properties that is necessary accountability [58] and in support of that overarching guarantee of fairness that must hold for the whole assessment process. We also provide a novel design that instead is fully verifiable.

Contributions Also this work is about coercion resistance. It builds from the state of the art in this topic, which is the mentioned protocol CREX. In particular, it studies the limitations of the protocol, finds and describes a few vulnerabilities that in the original protocol remained hidden under optimistic assumptions, proposes a stronger definition of coercion resistance, and extends CREX with more security against coercion under the stronger threat model. Since we stressed the importance of ensuring verifiability, this work proposes a coercion-resistant protocol that is also fully verifiable. All the security properties, the new and the old, are validated in Proverif with a substantial work of modelling and with several rounds of formal analysis (all the code is available). Listed in items, the contributions of this work are:

- a stronger definition of coercion, where all pieces of information are leaked to the intruder. Under this new definition, we revisit the definition of coercion, and in particular the security of CREX. This new definition allows us to discover a weakness in the Anonymous Submission property. We also refine the assignment sub-protocol from CREX, which allows us to find a second attack on privacy.
- a new protocol that corrects the issues of the CREX protocol and offers better security. We call this protocol *Secure CREX*, in short, SCREX. In SCREX, we conserve the spirit

of CREX, but modify some message exchanges to achieve strong coercion resistance and verifiability (missing in the CREX design).

- a concrete instantiation of the assignment protocol using a recent MixNet model proposed in [59] We are also the first to give a full formal model in Proverif and a complete security analysis of this particular Mixnet, whose security has been, so far, only informally argued.
- a full model of SCREX and all its cryptographic components and a rigorous formal analysis in Proverif of all the security claims we have on the protocol. This includes a formal analysis of the verifiability of SCREX. All our code and proofs are available as an artefact in a public repository.

Outline Section 3.2 discusses the related work on e-exams and recalls the basics of the applied π -calculus and ProVerif, the automatic verification tool that we use to prove the security of our protocol. In Section 3.3, we present how to model electronic exams in ProVerif and review relevant security properties: authentication, privacy and verifiability. In Section 3.4, we describe our novel SCREX protocol. Section 3.5 discusses the model of the protocols that we used for the formal analysis of SCREX, including that of the Mixnet, and the result of the analysis. All our ProVerif files are available online ¹.

3.2 Previous works, Background, and Tools

The discussion about security in e-learning and online exams was first introduced by Foley *et al.* in [48], where they formalized confidentiality in electronic exams. The notions of security and privacy were later informally expanded in [46], [60], and [45]. Giustolisi *et al.* in [5] introduced additional security properties and proposed a standard e-exam system by defining essential security properties for the various players and phases of an e-exam protocol. Dreier *et al.* [12] proposed the first formal framework in applied π -calculus for defining and analyzing authentication and security properties of both traditional and electronic exam protocols. Dreier *et al.* in [47] defined another framework to formally express and prove verifiability properties in e-exams. Rakeei *et al.* in [37] introduced the notion of coercion in e-exams and formalized it by defining two new properties.

¹ProVerif Files

In the literature, several e-exam protocols have been proposed with different threat models. Castella-Roca *et al.* [49] designed an e-exam management system with student, teacher, and manager entities, whose security is based on the existence of a trusted-third-party (TTP) in the manager role. Huszti and Petho in [23] proposed a new e-exam protocol that aimed to offer security and privacy guarantees without relying on a trusted party. However, Dreier *et al.* [12] demonstrated an attack on the secrecy and anonymity of the Reusable-anonymous-return-channel [21], the main building block of [23], showing that their scheme could not meet the main security and privacy requirements. Giustolisi *et al.* in [10] designed the *Remark!* e-exam protocol with minimal trust assumptions to ensure authentication, privacy, and verifiability properties. The guarantee for the anonymity of candidates and examiners is mainly provided by exponentiation mixnet servers, assuming that only one of them is trusted. Recently, Rakeei *et al.* [37], through pen-and-paper cryptanalysis, and Dreier *et al.* [59], through a model checker security proof, demonstrated a linkability attack on Haenni-Spycher's exponentiation mixnet [44] used in *Remark!*, breaking all anonymity features of the *Remark!* protocol. Bella *et al.* [6] developed WATA IV, a computer-assisted exam protocol in the presence of corrupted examiners and an honest-but-curious TTP as the anonymizer. In WATA IV, visual cryptography, a kind of secret sharing scheme, plays the main role in anonymizing candidate identities. Although informally discussed, the authors equipped their scheme with a set of verifiability tests and a dispute resolution mechanism. They extended their research on the WATA IV protocol in [11] by adding more security requirements and formally proving its security using the ProVerif model checker [61]. Rakeei *et al.* [37] proposed the CREX protocol as the first e-exam scheme designed to address the coercion threat model. They introduced a fix to prevent the linkability attack on Haenni-Spycher's mixnet and used this secure version of the mixnet to ensure anonymity of identities in their protocol. The most recent protocol in the literature, advanced by Vecsi and Petho [62], focuses on accountability. They adopted a similar shuffled answer technique and pre-assignment protocol used in [37] to mitigate collusion during the marking phase. To establish anonymous communication, the authors used a modified version of the Loopix anonymizer [17], called the Scalix network [25], which features enhanced logging for auditing authorities.

3.2.1 Applied Π -Calculus

In this work, we adopted the symbolic model to formally represent our protocol using the applied π -calculus [7] and employed the ProVerif verification tool [63] for analysis. The symbolic model allows cryptographic protocols to be analyzed under the Dolev-Yao attacker model [64], where the adversary has complete control over the network—able to read, modify, delete, or inject messages. However, the attacker can only execute cryptographic operations if they possess the correct keys, an assumption known as perfect cryptography. The applied π -calculus, an extension of the π -calculus, serves as a language for modeling security protocols and is designed to describe and analyze the properties of concurrent computations. In the following, we provide an overview of the ProVerif tool used to analyze our protocol and present the formal modeling of e-exams in the applied π -calculus.

3.2.2 ProVerif

A very large number of tools exists for verifying security protocols in the symbolic model. The protocol verifier ProVerif [61] analyzes protocols using an abstract internal representation via a set of Horn clauses. ProVerif is fully automatic: the user only gives the specification of the protocol and the property to verify. For example, to test the secrecy of a ground term M , i.e., whether the attacker can learn M , the following query is included in the input file before the main process: **query attacker** (M).

The tool can handle many different cryptographic primitives (encryption, signature, hash function, Diffie-Hellman Key agreement) specified as rewrite rules or equations. ProVerif can also handle an unbounded number of sessions and an unbounded message space.

ProVerif is able to prove the following properties:

- Secrecy (as reachability properties): the adversary cannot learn the secret.
- Authentication (as correspondence properties): if an event has been executed, then other events have been executed as well.
- Equivalences between processes (as observational equivalence): the adversary does not see the difference between two processes.

3.3 Formalizing e-Exams and Security Properties

This section reminds a formal toolkit to model and analyze e-exams in applied π -calculus and ProVerif.

We refer to the e-exam model in applied π -calculus introduced in [12], while for all the properties here in considered, we refer to the formalization proposed in different works [12, 47, 37].

The reason for referring to these works, in addition to relying on models and definitions that work perfectly fine, is that we intend to analyze the security of our novel protocol against all those previous properties before showing that our protocol is also correction-resistant and verifiable. In other words, our protocol must also preserve all the good properties of previous state-of-the-art e-exam design(s). Using the same formal framework of modelling and analysis of previous work is, therefore, a sound experimental choice to ensure consistency with state-of-the-art research and replicability for future research.

3.3.1 E-Exams Modelling

Electronic exams have been formally defined by Dreier *et al.* in [12]. In a nutshell, an *exam protocol* is a tuple $(C, E, Q, A_1, \dots, A_l, \tilde{n}_p)$, where C is the process executed by the candidates, E is the process executed by the examiners, Q is the process executed by the question committee, A_i 's are the processes executed by the authorities, and $\tilde{n}_p$ is the set of private channel names. Given the tuple $(C, E, A_1, \dots, A_l, \tilde{n})$ an *exam instance* is a closed process $EP = \nu \tilde{n}. (C\sigma_{id_1}\sigma_{a_1} \mid \dots \mid C\sigma_{id_j}\sigma_{a_j} \mid E\sigma_{id'_1}\sigma_{m_1} \mid \dots \mid E\sigma_{id'_k}\sigma_{m_k} \mid Q\sigma_q \mid A_1\sigma_{dist} \mid \dots \mid A_l)$, where $\tilde{n}$ is the set of all restricted names, which includes the private channels; $C\sigma_{id_i}\sigma_{a_i}$'s are the processes run by the candidates, the substitutions σ_{id_i} and σ_{a_i} specify the identity and the answers of the i^{th} candidate respectively; $E\sigma_{id'_i}\sigma_{m_i}$'s are the processes run by the examiners, the substitution $\sigma_{id'_i}$ specifies the i^{th} examiner's identity, and σ_{m_i} specifies for each question/answer pair the corresponding mark; Q is the process run by the question committee, the substitution σ_q specifies the exam questions; the A_i 's are the processes run by the exam authorities, the substitution σ_{dist} determines which answers will be submitted to which examiners for grading.

Given the process EP , $EP_I[\cdot]$ is the corresponding context, without the identities in the set I and $EP|_e$ stands for process EP without the code that follows the event e .

3.3.2 Authentication Properties

In ProVerif, authentication properties are proven using correspondence assertions based on defined events. In the applied π -calculus, these events are internal messages that, along with arbitrary arguments, act as markers to capture the state of a process at specific points within a protocol trace. They do not affect the protocol's behavior and are inaccessible to attackers. Instead, they help the verifier reason about the reachability of certain states and thereby model and prove authentication properties. To prove authentication properties in our SCREX protocol, we first define all relevant events and the situations in which they are emitted as follows:

- **registered**(id_c): triggered by the exam authority at the point where the registration of candidate (id_c) is completed.
- **submitted**($id_c, ques, ans$): raised by the candidate id_c when she submits the test ($ques, ans$) to the exam authority.
- **collected**($id_c, ques, ans$): triggered by the exam authority at the location where it accepts the test ($ques, ans$) from the candidate id_c .
- **marked**($id_e, ques, ans, mark$): triggered by examiner id_e at the point where he assigns the mark $mark$ to the test ($ques, ans$).
- **recorded**($id_c, id_e, ques, ans, mark$): triggered by the exam authority at the point where it officially records the mark $mark$, assigned by examiner id_e to the test ($ques, ans$) of candidate id_c .
- **notified**($id_c, id_e, ques, ans, mark$): triggered by candidate id_c when she accepts the mark $mark$, assigned by examiner id_e to her submitted test ($ques, ans$).

Based on the events defined earlier, we now define four authentication properties within the SCREX protocol. The first property, called *Test Origin Authentication*, ensures that the exam authority accepts tests only from registered candidates.

Definition 3.3.1 (Test Origin Authentication [42]). An exam protocol guarantees Test Origin Authentication if for every exam process EP the following assertion holds across all execution traces

$$inj\text{-}collected(id_c, ques, ans) \rightsquigarrow inj\text{-}registered(id_c)$$

Note that the term *inj* refers to an injective correspondence assertion, signifying a one-to-one relationship between events. This means that each occurrence of the event $\text{collected}(id_c, ques, ans)$ is preceded by a unique occurrence of the event $\text{registered}(id_c)$.

The next property is *Answer Authenticity*, which ensures that the content of the candidate's submitted test is not modified when the ea collects it.

Definition 3.3.2 (Answer Authenticity [42]). An exam protocol ensures Answer Authenticity if for every exam process EP the following assertion holds across all execution traces

$$\text{inj-collected}(id_c, ques, ans) \rightsquigarrow \text{inj-submitted}(id_c, ques, ans)$$

The *Mark Authenticity* property guarantees that the mark assigned to a test by the examiner is recorded without any modification.

Definition 3.3.3 (Mark Authenticity [42]). An exam protocol ensures Mark Authenticity if for every exam process EP the following assertion holds across all execution traces

$$\text{inj-recorded}(id_c, id_e, ques, ans, mark) \rightsquigarrow \text{inj-marked}(id_e, ques, ans, mark)$$

The final property, *Mark Authentication*, certifies that the mark the candidate receives for her test at the end of the examination is recorded without any changes.

Definition 3.3.4 (Mark Authentication [42]). An exam protocol guarantees Mark Authentication if for every exam process EP the following assertion holds across all execution traces

$$\text{inj-notified}(id_c, id_e, ques, ans, mark) \rightsquigarrow \text{inj-recorded}(id_c, id_e, ques, ans, mark)$$

3.3.3 Privacy Properties

We give formal definitions of privacy properties of exam protocols as defined in [12]. These properties are modeled as observational equivalences, a standard choice for such kind of properties [7]. We use *labeled bisimilarity* ($\approx_l$) to express the equivalence between two processes. In a nutshell, two processes are considered equivalent if an observer cannot differentiate between them.

The first property called *Question Indistinguishability*, requires that two processes with different questions remain observationally equivalent until the end of the registration

phase. This prevents the attacker from acquiring information about the exam questions before the examination phase begins. For this property to hold, the question committee must be honest; otherwise, it can be easily compromised if the committee reveals the questions to the attacker.

Definition 3.3.5 (Question Indistinguishability [12]). *Two processes with different questions have to be observationally equivalent until the registration phase. For every e-exam process EP that ends with the registration phase any questions q_1 and q_2*

$$EP_{\{id_Q\}}[Q\sigma_{q_1}]|_{reg} \approx_l EP_{\{id_Q\}}[Q\sigma_{q_2}]|_{reg}$$

The second property ensures that the evaluation procedure is carried out anonymously, i.e., that two instances in which candidates exchange their responses cannot be distinguished until the evaluation phase has concluded. This anonymity is desirable to guarantee impartiality in the grading process and may be a mandatory requirement in certain exam settings, such as those at specific universities or for competitive examinations.

Definition 3.3.6 (Anonymous Marking [12]). *A protocol ensures Anonymous Marking if the process where id_1 answers a_1 and id_2 answers a_2 is equivalent to the process where id_1 answers a_2 and id_2 answers a_1 . Formally, for every e-exam process EP that ends with the marking phase, any two candidates id_1 and id_2 and any two answers a_1 and a_2 :*

$$EP_{\{id_1, id_2\}}[C\sigma_{id_1}\sigma_{a_1}|C\sigma_{id_2}\sigma_{a_2}]|_{mark} \approx_l EP_{\{id_1, id_2\}}[C\sigma_{id_1}\sigma_{a_2}|C\sigma_{id_2}\sigma_{a_1}]|_{mark}$$

To prevent bribing or coercion of the examiners, it may be advantageous to guarantee their anonymity, thereby ensuring that no candidate is aware of the examiner who assigned their copy. The third property is meant to guarantee examiners' anonymity.

Definition 3.3.7 (Anonymous Examiner [12]). *A protocol ensures Anonymous Examiner if process in which examiner id'_1 grades the exam form of candidate id_1 and examiner id'_2 grades that of candidate id_2 cannot be distinguished from a process in which id'_1 grades the exam form of candidate id_2 and examiner id'_2 grades that of candidate id_1 . Formally, for every e-exam process EP, any two candidates id_1 and id_2 , any two examiners id'_1 and id'_2 and any two marks m_1 and m_2 :*

$$EP_{\{id_1, id_2, id'_1, id'_2, id_{A_1}\}}[C\sigma_{id_1}\sigma_{a_1}|C\sigma_{id_2}\sigma_{a_2}|E\sigma_{id'_1}\sigma_{m_1}|E\sigma_{id'_2}\sigma_{m_2}|A_1\sigma_{dist_1}] \approx_l \\ EP_{\{id_1, id_2, id'_1, id'_2, id_{A_1}\}}[C\sigma_{id_1}\sigma_{a_1}|C\sigma_{id_2}\sigma_{a_2}|E\sigma_{id'_1}\sigma_{m_2}|E\sigma_{id'_2}\sigma_{m_1}|A_1\sigma_{dist_2}]$$

The next property ensures that the mark itself has to remain private. Depending on the exam policy, this can be an optional property since some exam systems may publicly disclose the marks of the candidates.

Definition 3.3.8 (Mark Privacy [12]). *A protocol ensures Mark Privacy if two processes where the examiner id'_1 assigns for the same answer two different marks m_1 and m_2 cannot be distinguished from each other. Formally, for any e-exam process EP any marks m_1 and m_2*

$$EP_{\{id'\}}[E\sigma_{id'}\sigma_{m_1}] \approx_l EP_{\{id'\}}[E\sigma_{id'}\sigma_{m_2}]$$

3.3.4 Verifiability

Dreier *et al.* [47] propose a formalization to reason about individual and universal verifiability. We summarize the core ideas of the framework. The reader can refer to the original article for the full details of the definitions.

Definition 3.3.9. (Exam Abstract Model)[47] *An exam E is a tuple (I, Q, A, M, α) where I of type $\mathcal{I}$ is a set of candidate identities, Q of type $\mathcal{Q}$ is a set of questions, A of type $\mathcal{A}$ is a set of answers, M of type $\mathcal{M}$ is a set of marks, and α is the set of the following relations:*

- $\text{Accepted} \subseteq I \times (Q \times A)$: the candidates' tests accepted by the collector authority;
- $\text{Marked} \subseteq I \times (Q \times A) \times M$: the marks given to the candidates' tests;
- $\text{Assigned} \subseteq I \times M$: the marks assigned to the candidates;
- $\text{Correct} : (Q \times A) \rightarrow \mathcal{M}$: the function used to mark a test.

Definition 3.3.9 can be extended with two specific subsets:

- $I_r \subseteq I$ as the set of identities of candidates who registered for the exam;
- $Q_g \subseteq Q$ as the set of questions generated by the question committee.

The structure of a verifiability requirement consists of a verifiability test function and a predicate modeling a specific property. The evidence needed for the verifiability test comes from information about the exam execution and from private knowledge of involved roles.

According to [47], an exam is *testable* if it provides an algorithm (i.e., the verifiability test) that checks a specific property on the exam execution. The verifiability test must be

Definition	Individual Verifiability	Universal Verifiability
Registration		$R_{UV}(e) \Leftrightarrow I_r \supseteq \{i : (i, x) \in \text{Accepted}\}$
Question Validity	$QV_I(i, q, a, m, p) \Leftrightarrow (q \in Q_g)$	
Marking Correctness	$MC_{IV}(i, q, a, m, p) \Leftrightarrow (\text{Correct}(q, a) = m)$	$MC_{UV}(e) \Leftrightarrow (\forall (i, x, m) \in \text{Marked}, \text{Correct}(x) = m)$
Test Integrity	$TI_{IV}(i, q, a, m, p) \Leftrightarrow ((i, (q, a)) \in \text{Accepted} \wedge \exists m' : (i, (q, a), m') \in \text{Marked})$	$TI_{UV}(e) \Leftrightarrow \text{Accepted} = \{(i, x) : (i, x, m) \in \text{Marked}\}$
Mark Integrity	$MI_{IV}(i, q, a, m, p) \Leftrightarrow \exists m' : ((i, (q, a), m') \in \text{Marked} \wedge (i, m') \in \text{Assigned})$	$MI_{UV}(e) \Leftrightarrow \text{Assigned} = \{(i, m) : (i, x, m) \in \text{Marked}\}$
Mark Notification Integrity	$MNI_{IV}(i, q, a, m, p) \Leftrightarrow (i, m) \in \text{Assigned}$	

Table 3.1: Individual and universal verifiability.

both sound and complete for the given property. In Table 3.1, this is represented by the symbol $\Leftrightarrow$.

Individual verifiability considers the perspective of exam candidates who have access to private knowledge acquired during the exam. Universal verifiability, on the other hand, involves any auditor who does not necessarily have prior knowledge of the exam. The auditor relies on public information to conduct the verifiability tests. The requirements concern the verifiability of candidate registration, the validity of questions, and the integrity of tests, marks, and notification. The requirements are concisely listed in Table 3.1.

Regarding individual verifiability, the candidate uses public information and private knowledge to verify some aspects of the exam. The candidate knows her identity i , the test she submitted, which consists of question q and answer a , and the notified mark m . The candidate also knows the perspective p from the exam process, which includes the messages sent and received by the candidate during the exam. Note that the candidate's perspective p is not necessary to specify the predicate that models the properties to verify. In fact, the perspective never appears in the predicate c . However, the perspective may be necessary for implementing the verifiability-test. The five individual verifiability requirements are Question Validity, Marking Correctness, Test Integrity, Mark Integrity, and Mark Notification Integrity.

The definitions of universal verifiability are not pivoted around any exam role but consider the viewpoint of an external auditor. The auditor runs the verifiability tests on the public information available after an exam protocol run. Hence, the knowledge of the auditor consists of a general variable e that contains evidence. Moreover, the auditor does not

necessarily know the questions and the answers given during the exam. This is captured in the definitions with a generic term x .

The four universal verifiability requirements concern the registration of the candidates and the integrity of the batch of tests from the submission until after the marking. Therefore, the requirements are similar to the ones already seen for individual verifiability, plus Registration. The requirements of Question Validity and Mark Notification Integrity, are only relevant for individual verifiability because the external auditor has no knowledge of the questions nor the marks received by the candidates, but only of public data.

3.3.5 Coercion Properties

Coercion resistance may be regarded as an extension of the basic property of privacy. It is a strong form of privacy that assumes the adversary can interact with examiners and candidates. Specifically, the adversary may instruct targeted examiners to reveal their private keys after registration or direct them to correct exams in a particular manner. If the adversary can determine whether the examiners followed these instructions, it gains the ability to influence the correction process. In [37], Rakeei *et al.* were the first to define and formalize coercion properties in e-exams. Coercion resistance means that an exam protocol should preserve privacy between candidates and examiners, even if one party attempts to force the other to reveal secrets that could lead to deanonymization.

The first coercion resistance property, called *Anonymous Submission*, is an enhanced version of the privacy property *Anonymous Marking* defined in Section 3.3.6. This property ensures that examiners cannot discover the link between candidates and their submitted tests, even if they coerce candidates into revealing their secrets. It relies on the distinguishability of the answers submitted by two known candidates.

Definition 3.3.10 (Anonymous Submission [37]). *An e-exam protocol guarantees Anonymous Submission if for every e-exam process that ends with the marking phase where a candidate id_1 answers c according to the intruder's wishes is indistinguishable from the process where id_1 answers something else (a_1), i.e. if there exists a closed process C' such that for all answers c and a_1 :*

- $C'^{\backslash out(chc, \cdot)} \approx_l C\sigma_{id_1}\sigma_{a_1}$
- $EP_{\{id_1, id_2\}}[(C\sigma_{id_1}\sigma_c)^{chc} | C\sigma_{id_2}\sigma_{a_1}]|_{mark} \approx_l EP_{\{id_1, id_2\}}[C' | C\sigma_{id_2}\sigma_c]|_{mark}$

C' represents a process in which candidate id_1 answers a_1 while pretending to cooperate with the coercer. The coercer cannot distinguish between the scenario where id_1 genuinely cooperates and submits c and the scenario where id_1 feigns cooperation but actually submits a_1 .

The second coercion resistance property named *Single Blindness* is a strong version of the privacy property *Anonymous Examiner* defined in Section 3.3.6. It aims to guarantee that candidates cannot learn the link between examiners and tests they marked even if candidates force examiners to reveal their secrets. It is based on the distinguishability of the marks provided by two examiners.

Definition 3.3.11 (Single Blindness [37]). *An e-exam protocol guarantees Single Blindness if for every e-exam process that ends with the marking phase where an examiner id'_1 marks an e-exam according to the intruder's wishes e is indistinguishable from the process where id'_1 marks something else m_1 , i.e., if there exists a closed process E' such that for all marks e and m_1 :*

- $E'^{\text{out}(che, \cdot)} \approx_l E\sigma_{id'_1}\sigma_{m_1}$
- $EP_{\{id'_1, id'_2\}}[(E\sigma_{id'_1}\sigma_e)^{che} | E\sigma_{id'_2}\sigma_{m_1}]|_{\text{mark}} \approx_l EP_{\{id'_1, id'_2\}}[E' | E\sigma_{id'_2}\sigma_e]|_{\text{mark}}$

E' describes a scenario where examiner id'_1 gives the mark m_1 while feigning cooperation with the coercer. The coercer cannot distinguish between genuine cooperation, where id'_1 assigns the mark e , and a situation where id'_1 is only pretending to cooperate but actually assigns m_1 .

3.4 A New Verifiable Coercion-Resistant Protocol

The protocol, which we call Secure CREX (SCREX), includes four roles: Examiner(E), Mixnet(M), Exam Authority(EA), and Candidate(C). It is organized in five phases: *Registration*, *Assignment*, *Testing*, *Marking*, and *Notification*. We assume the existence of a Public Append-only Bulletin Board (B.B.) used for publishing public protocol parameters. Anyone has access to this B.B., including potential attackers; hence, any message posted on this board must be signed by its respective author. Additionally, we assume the presence of an Untappable channel between EA and C that prevents the attacker from discovering the contents of any communicated message and from being aware of the communication occurrence itself.

3.4.1 Registration

This phase aims to generate two sets of pseudonyms for examiners and candidates in a way that ensures only the owner of a public key can learn the corresponding pseudonym. To achieve this, the mixnet M takes as inputs the list of public keys for the examiners and candidates. Utilizing the secure exponentiation mixnet introduced in [37], it produces two lists of pseudo-public keys: one for the candidates and another for the examiners. The mixnet then signs these lists and publishes them on the bulletin board (B.B.). Later in Section 3.5, we demonstrate that this pseudonymization is necessary to achieve *Anonymous Marking*, *Anonymous Examiner*, *Anonymous Submission*, and *Single-Blindness*. The detailed generation of pseudonyms for candidates is illustrated in Figure 3.1. The same procedure is applied to the examiners. Note that in this setup, along with a public key, the mixnet receives a strong zero-knowledge proof (SZKP) as described in [59], which proves the knowledge of the corresponding private key.

	inputs	mix_1	mix_2	mix_m	outputs
C_1	$(PK_1, SZKP_1)$	$PK_{\pi_1(1)}^{r_1}$	$PK_{\pi_2(1)}^{r_2} \dots$	$PK_{\pi_m(1)}^{r_m} =$	$\overline{PK}_1$
C_2	$(PK_2, SZKP_2)$	$PK_{\pi_1(2)}^{r_1}$	$PK_{\pi_2(2)}^{r_2} \dots$	$PK_{\pi_m(2)}^{r_m} =$	$\overline{PK}_2$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	
C_n	$(PK_n, SZKP_n)$	$PK_{\pi_1(n)}^{r_1}$	$PK_{\pi_2(n)}^{r_2} \dots$	$PK_{\pi_m(n)}^{r_m} =$	$\overline{PK}_n$
	g	g^{r_1}	$g^{r_2} \dots$	$g^{r_m} =$	h_C

Figure 3.1: Pseudonym generation using the secure exponentiation mixnet. All the terms within the box are published on the bulletin board.

3.4.2 Assignment Protocol

One of the responsibilities of the exam authority in an e-exam protocol is to distribute tests to eligible examiners. One possible solution to this problem is the *Assignment Protocol* introduced in [37]. However, this scheme has two main drawbacks: (i) The protocol is only discussed informally, with no formal proof provided to ensure its security. (ii) The authors implicitly assume that in case of examiner coercion, the coercer is not aware of the communications that occur during this protocol. In other words, the coercion threat is only considered during the Marking phase. To address these issues, we propose a secure assignment protocol that allows the exam authority to anonymously allocate a set of num-

bers, representing test identifiers, to a group of examiners. This ensures each examiner will only learn their assigned numbers and will have no knowledge of the numbers assigned to others. Additionally, the link between the numbers and the examiners remains secret from the exam authority. Later in Section 3.5, we demonstrate how our SCREX protocol, which utilizes this assignment protocol, maintains security across defined threat models, including examiner coercion. Now, we describe our *Assignment* protocol in two steps:

Step 1 Let's consider an exam with n candidates, d examiners. First, the exam authority EA randomly generates d sets denoted as $\{P_1, \dots, P_d\}$ where $P_i \subset \{1, \dots, n\}$, $\forall i \in \{1, \dots, d\}$. Then, the exam authority publishes message M_1 on the BB.

$$M_1 = \{Enc_1, \dots, Enc_d\} = \{Enc(P_1)_{PK_{EA}}, \dots, Enc(P_d)_{PK_{EA}}\}$$

In the above message, Enc is a probabilistic encryption function and PK_{EA} is the public key of the exam authority. The idea of encrypting messages with PK_{EA} is hiding the content of the subsets $\{P_1, \dots, P_d\}$ and preparing a proof by the exam authority for designated examiners in the next step.

Step 2 Following publishing M_1 , the exam authority securely transmits the following message

$$\begin{aligned} M_2 &= \{Enc'_1, \dots, Enc'_j, \dots, Enc'_d\} \\ &= \{Enc(P_1)_{\overline{PK}_{E_1}}, \dots, Enc(P_j)_{\overline{PK}_{E_j}}, \dots, Enc(P_d)_{\overline{PK}_{E_d}}\} \end{aligned}$$

through an untappable channel to the examiners where $\overline{PK}_{E_j}$ represents the pseudo-public key of examiner j . Additionally, the exam authority generates d proofs as:

$$\begin{aligned} (x_1, \omega_1) \in S_1 &\Leftrightarrow \left(SK_{EA} = f(PK_{EA}) \wedge F(Enc_1, Enc'_1) = true \right) \\ &\vee \left(\overline{SK}_{E_1} = f(\overline{PK}_{E_1}) \right) \end{aligned}$$

$\vdots$

$$\begin{aligned}
(x_j, \omega_j) \in S_j &\Leftrightarrow \left(SK_{EA} = f(PK_{EA}) \wedge F(Enc_j, Enc'_j) = true \right) \\
&\quad \vee \left(\overline{SK}_{E_j} = f(\overline{PK}_{E_j}) \right) \\
&\quad \vdots \\
(x_d, \omega_d) \in S_d &\Leftrightarrow \left(SK_{EA} = f(PK_{EA}) \wedge F(Enc_d, Enc'_d) = true \right) \\
&\quad \vee \left(\overline{SK}_{E_d} = f(\overline{PK}_{E_d}) \right)
\end{aligned}$$

where $x_j = (PK_{EA}, Enc_j, Enc'_j)$, $Enc_j = Enc(P_j)_{PK_{EA}}$, $Enc'_j = Enc(P_j)_{\overline{PK}_{E_j}}$, and $\omega_j = (SK_{EA}, P_j) \vee (SK_{E_j})$. The exam authority sends $\pi = \{\pi_1, \dots, \pi_d\}$ to each examiner where π_i , for $i \in \{1, \dots, d\}$, is the ZK proof for the statement S_i . In this following statement, $F(Enc(m_1)_{\overline{PK}_1}, Enc(m_2)_{\overline{PK}_2})$ is a deterministic polynomial-time function that returns true if $m_1 = m_2$. Upon receiving $\{M_2, \pi_i\}$, E_i first extracts his corresponding P_i from M_2 message. Then he verifies $\{\pi_1, \dots, \pi_d\}$ and checks if $P_i \subseteq \{1, \dots, n\}$. As the above proofs imply, at the end of the *Assignment* protocol, each examiner not only learns the assigned subset to him but also can verify that message M_2 is encrypting the same subsets, previously committed by the exam authority through message M_1 .

3.4.3 Testing

This phase pertains to the process of assigning questions to candidates and gathering their corresponding answers. In the SCREX protocol, we assume that exam authority is responsible for creating the exam questions. For each candidate C , the exam authority signs and encrypts the test with the candidate's pseudonym and then posts it on the B.B. Candidate C decrypts the test, verifies the signature, and proceeds to answer the questions. Then, she sends $[Sign_{SK_C, h_C}(quest, ans, \overline{PK}_C)]_{PK_A}$ to the exam authority over an untappable channel where SK_A , PK_A , $quest$, ans and $\overline{PK}_C$ represent the private key of the exam authority, the public key of the exam authority, the questions, the answers, and the candidate pseudonym, respectively. Additionally, the exam authority generates a receipt for the candidate as $[Sign_{SK_A} H(quest, ans, \overline{PK}_C, \alpha)]_{PK_A}$ where H and α denote a secure hash function and a random value generated by the exam authority, respectively, and publishes it on the BB.

It is worth mentioning that in the *Testing* phase, we implement two countermeasures against candidate coercion threats. First, the use of an untappable channel ensures that a coercer cannot determine whether communication occurs between C and EA . Second, the

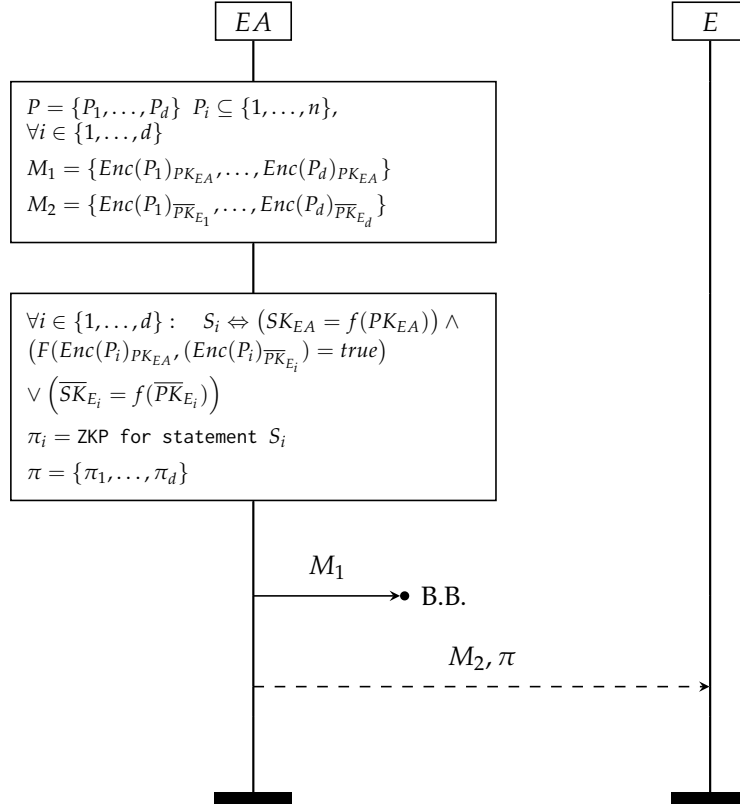


Figure 3.2: The message sequence chart of Assignment Protocol
 Public transfer: $\longrightarrow$ Untappable channel: $-\longrightarrow$

receipt generated by exam authority is masked by a secure α value that will be revealed at the end of the exam. Consequently, the participation of a specific candidate cannot be detected by a coercer until the notification phase. The security analysis in Section 3.5 confirms this claim.

3.4.4 Marking

In this phase, the tests that have been collected in the previous phase, are assigned to the examiners by the exam authority. Upon gathering the tests from the candidates, the EA constructs a matrix named T comprising all candidates' (*question, answer*) pairs. Afterward,

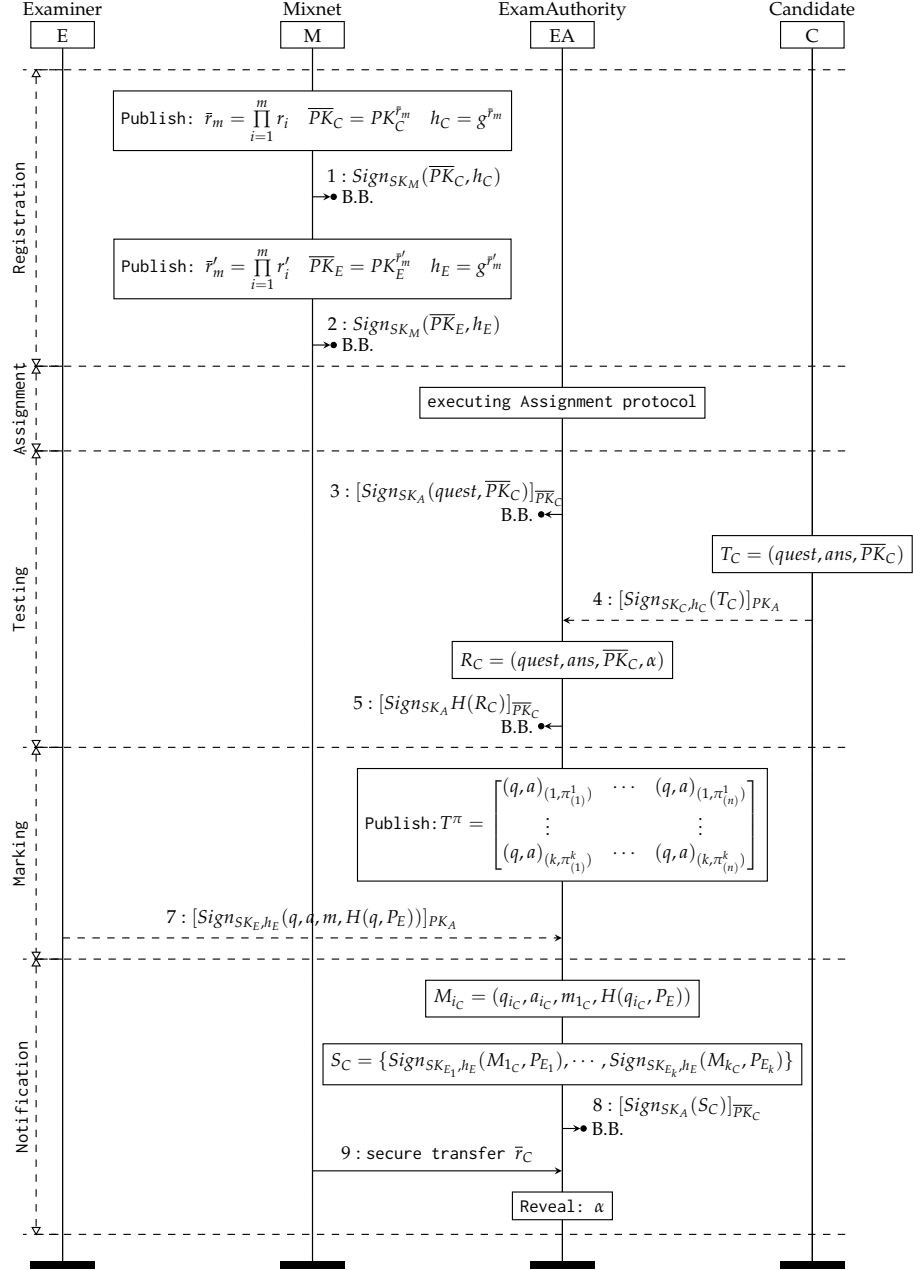


Figure 3.3: The message sequence chart of SCREX protocol
Public channel: $\longrightarrow$ Untappable channel: $\dashrightarrow$

it selects a secure permutation matrix named Π and applies it to T to generate a new matrix T^π , denoted as $\Pi(T) = T^\pi$. The following mathematical transformation illustrates the shuffling procedure.

$$T = \begin{bmatrix} (q,a)_{1,1} & \cdots & (q,a)_{1,n} \\ \vdots & & \vdots \\ (q,a)_{k,1} & \cdots & (q,a)_{k,n} \end{bmatrix} \xrightarrow{\Pi} \begin{bmatrix} (q,a)_{(1,\pi_1^1)} & \cdots & (q,a)_{(1,\pi_n^1)} \\ \vdots & & \vdots \\ (q,a)_{(k,\pi_1^k)} & \cdots & (q,a)_{(k,\pi_n^k)} \end{bmatrix} = T^\pi$$

Here, assuming k questions and n candidates, $(q,a)_{(x,y)}$ represents the answer of candidate y to question x . Let T and T^π be sets of n vectors $T = [V_1, \dots, V_n]$ and $T^\pi = [V_1^\pi, \dots, V_n^\pi]$, respectively. V_y , $1 \leq y \leq n$, denotes the test of candidate y , while V_y^π , $1 \leq y \leq n$, signifies a new test where each question belongs to a random author. The exam authority signs each element of T^π with its private key and publishes the signed matrix on the BB. Now E_j marks $(q,a)_{(x,y)}$, $\forall y \in P_j$ and sends $\text{Sign}_{SK_E, h_E}(q, a, m, H(q, P_E))_{PK_A}$ to the exam authority over an untappable channel where m is the mark for the (q,a) test and H is a secure hash function.

The shuffling mechanism applied in this phase serves two main purposes. First, by breaking down a complete test and distributing (q,a) pairs to multiple examiners, a candidate's test is marked by several examiners rather than just one. This approach mitigates the threat of examiner-candidate collusion. Second, compared to a non-shuffled solution, a test graded by multiple examiners enhances the candidate's perception of *fairness* in marking.

3.4.5 Notification

The main objective of this phase is to notify the candidate of her mark. Let T_C be the C^{th} column of T , representing the test of candidate C . Furthermore, let T_{iC} denote the i^{th} element of vector T_C . When the exam authority receives the message $S_{q,a} = \text{Sign}_{SK_E, h_E}(q, a, m, H(q, P_E))_{PK_A}$ from examiner E , it first decrypts the message with its private key and verifies whether (q,a) exists in T . Then, it regenerates $H(q, P_E)$ to ensure it matches the hash received from E . After collecting all messages from the examiners, the exam authority constructs $S_C = \{S_{C_1}, \dots, S_{C_k}\}$ where $S_{C_i} = S_{q,a}$ s.t. $(q,a) = T_{iC}$. Finally, it signs S_C , encrypts it with $\overline{PK}_C$ and publishes the result as the notification message for candidate C on the BB. The term S_C contains the mark of candidate C signed by corresponding examiners, which allows C to verify that the mark she received is graded

by legitimate examiners and is not modified.

In general, the $P_1, \dots, P_d$ subsets are not necessarily disjoint, meaning that two or more examiners might be assigned the same (q, a) pair and consequently provide different marks. In such cases, the exam authority selects one of the marks and notifies the candidate through the S_C term. As we will prove in Section 3.5.4, the *Marking Correctness* verifiability test ensures that the mark assigned to the candidate's test is correct, regardless of which examiner generated it. Therefore, the notification of marks remains robust even if multiple examiners grade the same (q, a) .

3.5 Formal Security Analysis

In applied π -calculus and ProVerif, term manipulation by any participant, including adversaries, is handled through *destructor* functions, which are part of what's called equational theory. The specific equational theory we used is detailed in Table 3.2. The first four equations, covering *Exponentiation*, *ElGamal Decryption*, *Digital Signatures* and *Non-Interactive Zero-Knowledge Proofs (NIZKP)* were initially proposed by Dreier *et al.* [12, 59]. The final equation in the table, which we are presenting for the first time, describes the designated verifier ZKP used in the *Assignment* protocol. We will now go through each part of this equational theory in more detail.

- **Exponentiation:** The symbol *exp* here represents the exponentiation operation used by the participants in the protocol. The first part of the equation ensures that the protocol behaves as expected, specifically that $(g^x)^y = (g^y)^x$. The second part addresses an attack highlighted in [37], which targets exponentiation mixnets used in the protocol to conceal the link between public keys and pseudonyms. It is important to note that g in these equations is a constant; otherwise, ProVerif's pre-treatment of the equations will not terminate.
- **ElGamal Decryption:** The function symbol *enc* takes as argument a plaintext m , a basis h , a public key $exp(h, sk)$ and a random coin r . To decrypt the encrypted message m , one needs to specify the ciphertext, the basis used for the encryption h and the corresponding secret key sk .
- **Digital Signature:** The function symbol *sign* is used to model a digital signature. It takes as argument a message m to sign, a basis h and a secret key sk . Thus, to verify

the signature, we apply *checksign* with the corresponding public key $\text{exp}(h, sk)$.

- NIZKP: Since the participants must provide zero-knowledge proofs to the mixnet, we added an equation to model the non-interactive zero-knowledge proof of the discrete log.
- Designated Verifier ZKP: In this equational theory, the *checkdzkp* function returns *true* under two conditions: (i) if a valid proof is correctly generated by the exam authority, or (ii) if any fake proof is generated by an examiner. The function *priv_zk* is the proof generator, which takes as inputs an exponentiation base g , secret key sk , subset p and randomness r . The first equation returns *true* if the two encryption functions have the same arguments and the exam authority generates the proof. The second condition is satisfied if the proof is generated by an examiner holding the private key skE' . It is important to note that the randomness used in the *priv_zk* proof generator is crucial, as it prevents a coercer who forces an examiner to reveal all secret keys and proofs from reproducing the proof. Otherwise, the coercer could deduce which subset was sent to the examiner.

Primitive	Equation
Exponentiation	$\text{exp}(\text{exp}(g, x), y) = \text{exp}(\text{exp}(g, y), x)$ $\text{exp}(\text{exp}(\text{exp}(g, x), y), z) = \text{exp}(\text{exp}(\text{exp}(g, x), z), y)$
ElGamal Decryption	$\text{dec}(\text{enc}(m, h, \text{exp}(h, sk), r), h, sk) = m$
Digital Signature	$\text{checksign}(\text{sign}(m, h, sk), h, \text{exp}(h, sk)) = m$ $\text{getmessage}(\text{sign}(m, h, sk)) = m$
NIZKP	$\text{zkpcheck}(\text{zkp}(A, g, x), g, \text{exp}(g, x)) = \text{true}$
Designated Verifier ZKP	$\text{checkdzkp}(\text{exp}(g, skEA), \text{exp}(\text{exp}(g, rce), skE),$ $\text{enc}(x, g, \text{exp}(g, skEA), r1),$ $\text{enc}(x, \text{exp}(g, rce), \text{exp}(\text{exp}(g, rce), skE), r2),$ $\text{priv_zk}(g, skEA, x, r3)) = \text{true} \text{ otherwise}$ $\text{checkdzkp}(\text{exp}(g, skE'), \text{exp}(\text{exp}(g, rce), skE), c, d,$ $\text{priv_zk}(g, skE', p, r3)) = \text{true}.$

Table 3.2: Equational Theory.

The results of our formal analysis using ProVerif are summarized in Table 3.5. The timings for the latter were done on macOS with an M1 processor and 16 GB of RAM.

In this table, the *Corrupted Parties* column indicates the maximum number of parties that can be compromised while still ensuring the property holds. In other words, the property remains satisfied even if all the specified roles are compromised, but it will break if one additional role is corrupted beyond those listed. Based on the security properties defined earlier in Section 3.3, we will now discuss the results of our analysis in more detail.

3.5.1 Authentication and Secrecy in SCREX

Given the model we introduced for authentication properties in Section 3.3.2, ProVerif demonstrates that all the correspondence assertions hold. Additionally, using reachability queries in ProVerif, we investigate two secrecy properties: *Mark Secrecy* and *Subset Secrecy*. The former ensures that marks generated by examiners remain secret from the attacker, while the latter guarantees the secrecy of the subsets assigned to examiners. ProVerif successfully shows that both these secret values are unreachable to the attacker. As shown in Table 3.5, for authentication and secrecy properties, even if the mixnet is corrupted, the secrecy and authentication requirements in SCREX are still satisfied. This result supports the fact that the role of the mixnet is merely to pseudonymize identities to achieve privacy and coercion properties, and its trustworthiness is not necessary for maintaining authentication and secrecy properties.

3.5.2 Privacy in SCREX

Table 3.5 demonstrates that SCREX satisfies all defined privacy requirements. In the SCREX protocol, since the exam authority generates the questions, it must remain honest to ensure *Question Indistinguishability* is satisfied. For *Anonymous Marking*, at least two honest candidates are necessary; otherwise, an attacker could collude with all corrupted candidates to easily compromise the mark anonymity of the sole honest candidate. Similarly, *Anonymous Examiner* requires at least two honest examiners to uphold the anonymity of the examiners. For *Mark Privacy*, which concerns the privacy of a mark given to a candidate, anyone except the concerned candidate, the examiner and the exam authority can be dishonest. As the result shows, the mixnet should be trusted for maintaining *Anonymous Examiner* and *Anonymous Marking*.

3.5.3 Coercion in SCREX

In SCREX, the coercer is assumed to have access to all of the victim's secret information, including the randomness used in probabilistic encryption, whereas, in the CREX protocol, it was only assumed that the coercer would obtain the private key of the coerced party. However, the assumption regarding the temporariness of *Examiner Coercion* in CREX remains unchanged in SCREX, meaning that coercion is only considered relevant until the end of the marking phase.

If we apply the new coercion threat model to the CREX protocol, ProVerif demonstrates that both *Anonymous Submission* and *Single-Blindness* are compromised. The primary issue lies in the public communication between the exam authority and examiners, as well as between the exam authority and candidates. In the SCREX protocol, as shown in Figure 3.3, these two channels are replaced with *untappable* channels, and ProVerif successfully verifies the aforementioned coercion-resistance properties.

The security analysis indicates that the exam authority must remain honest to achieve *Anonymous Submission* and *Single-Blindness*. If the exam authority is dishonest, it could expose the data transmitted over untappable channels, allowing the coercer to detect non-cooperation by the coerced party. Additionally, the mixnet must also be trusted to ensure anonymity, thereby satisfying *Anonymous Submission* and *Single-Blindness*.

3.5.4 Verifiability in SCREX

The ProVerif model used for authentication and privacy is also used to analyze SCREX for verifiability.

All relations are built from the posts that appear on the bulletin board. The tuples $(i, (q, a), m)$ are generated from the marked test signed by the examiner, that is, $Sign_{SK_E, H_E}(M_C)$. The tuples (i, m) instead are built from the encryption of the marked test generated by the exam authority, that is, $\{Sign_{SK_E, H_E}(M_C)\}_{PK_C}$. The function *Correct*, which is the algorithm used to mark the tests, is modeled as a ProVerif.

Individual verifiability definitions require the existence of verifiability tests that candidates run to check the properties of the protocol. We show that SCREX has the necessary verifiability tests. In ProVerif, the event OK signals that the verifiability test succeeds, while the event KO that the verifiability test fails.

We check the event OK is always preceded by the event emitted in the part of the code where the predicate becomes satisfied to prove soundness. This can be modeled as a

Requirement	Soundness	Completeness
Question Validity	✓ (EA)	✓ (all)
Test Integrity	✓	✓ (all)
Marking Correctness	✓ (EA)	✓ (all)
Mark Integrity	✓	✓ (all)
Mark Notification Integrity	✓	✓ (all)

Table 3.3: Summary of the analysis of SCREX for I.V. requirements

Requirement	Soundness	Completeness
<i>Registration</i>	✓	✓ (all)
Marking Correctness	✓ (EA)	✓ (all)
Test Integrity	✓	✓ (all)
Mark Integrity	✓	✓ (all)

Table 3.4: Summary of the analysis of SCREX for U.V. requirements

reachability property in ProVerif. We assume an honest candidate who plays the role of the verifier. The other principals are usually corrupted, if not stated otherwise. The verifiability test receives the data from the candidate via a private channel, and the remaining data posted on the bulletin board via public channels. This allows an attacker to manipulate the input data. Corrupted principals may collude with the attacker.

We check that the event $K0$ cannot be emitted to prove completeness. We consider only honest principals as the test succeeds if its input data is correct.

Table 3.3 summarises the results of the individual verifiability analysis of SCREX and reports the roles required to be honest.

To prove the soundness of the universal verifiability definitions, we check that the event $K0$ cannot be emitted. Every time the verifiability test succeeds, we check that the decryption of the concerned ciphertext gives the expected plaintext. If not, the event $K0$ is emitted. This is necessary as the candidate can be corrupted, hence it can be hard to find a ProVerif process that can be annotated with events to check soundness via correspondence assertions.

Table 3.4 summarises the results of the universal verifiability analysis of SCREX and reports the roles required to be honest. Note that it is not possible to automatically prove the universal verifiability requirements in ProVerif because ProVerif cannot iterate over all candidates (*i.e.*, ProVerif does not support loops). In [47], manual induction proofs

that generalize the ProVerif result are provided to fully prove the universal verifiability requirements. The same strategy can be used in our case.

	Property	Corrupted Parties	Result	Time
Secrecy	Mark Secrecy	N	✓	2 s
	Subset Secrecy	C, N	✓	2 m 52 s
Authentication	Test Origin Authentication	N	✓	4 s
	Answer Authenticity	N	✓	3 s
	Mark Authenticity	N	✓	5 s
	Mark Authentication	N	✓	4 m 1 s
Privacy	Question Indistinguishability	C, E, N	✓	5 s
	Anonymous Marking	C^*, E, EA	✓	7 s
	Anonymous Examiner	C, E^*, EA	✓	14 s
	Mark Privacy	N	✓	19 m 32 s
Coercion	Single Blindness	C	✓	12 m 40 s
	Anonymous Submission	C^*, E	✓	23 m 5 s
Individual Verifiability	Question Validity	C, E, N	✓	11 m 13 s
	Marking Correctness	C, E, N	✓	2 m
	Test Integrity	C, E, EA, N	✓	2 m 16 s
	Mark Integrity	C, E, EA, N	✓	3 m 4 s
	Mark Notification Integrity	C, E, EA, N	✓	1 m 15 s
Universal Verifiability	Registration	C, E, EA	✓	50 s
	Marking Correctness	C, E, N	✓	24 s
	Test Integrity	C, E, EA, N	✓	23 s
	Mark Integrity	C, E, EA, N	✓	25 s

Table 3.5: Analysis results of our SCREX protocol. The * symbol indicates a scenario where all parties except two are corrupted.

The results of the analysis are depicted in Table 3.5 and show that all our properties are verified.

3.6 Conclusion

Exams are often seen as a way to evaluate students' knowledge in schools, but the concept of exams, or assessments in general, covers a much broader range of activities. In fact, assessments can be more significant than typical school or university exams. For exam-

ple, they might include professional skills evaluations in companies, language proficiency tests like TOEFL or IELTS, or even peer reviews for academic papers. Security is a major concern in all these types of exams, as the results are crucial for measuring a candidate's competence and can have far-reaching consequences, such as influencing hiring decisions or visa approvals.

When exams are conducted electronically, e-exams not only need to meet the functionality of traditional exams but also address new security challenges that arise due to their digital nature. Recently, research revealed that security in e-exams is defined through a set of authentication, integrity, and confidentiality properties that are peculiar and differ in their specification from those in similar domains, like e-voting. They bring new insights in their design of secure systems. One of the latest is about coercion, a threat that has indeed found relevance in the news since a party can be interested in altering the result of an assessment (think of where e-exams are a precondition to applying for a VISA or a promotion). In coercion scenarios, a candidate, examiner, or even an external coercer may force a party to reveal their identity or act in a way that manipulates the outcome of the exam.

The CREX protocol introduced in [37] was the first to address the coercion threat in e-exams, but it had some security flaws and incomplete aspects. In this work, we first analyzed the security of the CREX protocol. To do so, we refined the assignment process, which was not fully specified in the original paper, and considered a stronger coercion model. Through this more precise modeling, we uncovered attacks on privacy properties in CREX.

As a result, we developed SCREX, a new e-exam protocol designed to resist stronger coercion threats. Additionally, verifiability, which ensures fairness in exams, was missing in the original CREX scheme. We designed SCREX to include verifiability as well, using a state-of-the-art framework to prove this property for e-exams. Our protocol was shown to be verifiable, and we created formal models in ProVerif to automatically verify the security properties of SCREX. Our analysis highlights the importance of having a clear, detailed specification for all aspects of a protocol and taking a realistic approach to coercion threats. Moving forward, we intend to investigate stronger threat models in which the exam authority may act dishonestly under coercion. Furthermore, we plan to extend our methodology to a broader class of security protocols that require both privacy and verifiability.

Chapter 4

Anamorphic Voting: Ballot Freedom Against Dishonest Authorities

Abstract	62
4.1 Introduction	62
4.2 Anamorphic Voting	63
4.2.1 From Anamorphic Encryption to Anamorphic Voting.....	63
4.2.2 A Running Example: CHVote.....	65
4.2.3 Ballot Freedom in CHVote	66
4.3 More Anamorphic Constructions	67
4.4 The Estonian Internet Voting System	70
4.4.1 Protocol Description	70
4.4.2 Ballot Freedom in IVXV	72
4.5 Helios	73
4.5.1 Ballot Freedom in Helios	74
4.6 Belenios	75
4.6.1 Protocol Description	75
4.6.2 Ballot Freedom in Belenios	76
4.7 Related Work	76
4.8 Discussion and Future Work	77

Abstract

Electronic voting schemes typically ensure ballot privacy by assuming that the decryption key is distributed among tallying authorities, preventing any single authority from decrypting a voter’s ballot. However, this assumption may fail in a fully dishonest environment where all tallying authorities collude to break ballot privacy.

In this work, we introduce the notion of *anamorphic voting*, which enables voters to convey their true voting intention to an auditor while casting an (apparently) regular ballot. We present new cryptographic techniques demonstrating that several existing voting schemes can support anamorphic voting.

4.1 Introduction

According to the Democracy Index published by the Economist Group [65] in 2024, 59 countries are classified as authoritarian regimes. To remain in power, such regimes need to demonstrate overwhelming support for the ruling party, often designing voting systems to ensure that the dictator decisively wins the election. Therefore, elections in these countries often lack transparency, raising concerns about the secrecy and integrity of the ballots.

An authoritarian regime that (falsely) seeks to address such concerns could still implement a voting scheme that is proven to be secure, while violating its trust assumptions. For example, ballot privacy is clearly compromised in most voting systems if an attacker controls all the election decryption keys and has access to the datasets containing the ballots cast at the voting phase. The absence of vote privacy intimidates voters, preventing them from freely expressing their true preferences. However, is it still possible for voters to communicate their real voting intentions without being detected by the authoritarian regime?

In this work, we introduce *anamorphic voting*, exploring how participants in an election might communicate covertly even when an authoritarian regime controls *all* communication channels and implements an *existing* voting scheme without satisfying the necessary trust assumptions. Anamorphic voting is inspired by the recent concept of anamorphic encryption [33] in the context of electronic voting. While anamorphic encryption studies whether *a single* well-established public-key cryptosystem supports covert communication under stringent dictator conditions, anamorphic voting examines whether *a combination* of

cryptographic primitives within a voting scheme can enable covert communication even in the presence of dishonest authorities.

We introduce the term *ballot freedom* to capture the most useful application of anamorphic voting. Ballot freedom is the ability of a voter to freely communicate their true voting intention to an auditor without the knowledge of the authoritarian regime.

At first glance, ballot freedom seems similar to receipt-freeness or coercion-resistance (by replacing the terms "auditor" with "tallier" and "coercer" with "authoritarian regime"). However, there are at least two key differences. First, receipt-freeness and coercion-resistance are properties that can be enforced by designing a new voting scheme, while ballot freedom is a property achieved through the combination of cryptographic primitives in an existing voting scheme. Second, receipt-free and coercion-resistant voting schemes need to assume the existence of anonymous and/or private channels at some point [66]. This very assumption is ruled out in the context of anamorphic voting, as authoritarian regimes often control all communication channels.

In this paper, we introduce the first constructions for anamorphic voting by investigating combinations of the most common cryptographic primitives in voting schemes. We show that anamorphic voting is possible in the Estonian Internet Voting System (IVXV) [67], CHVote [68], Helios [69] and Belenios [70] voting schemes. Additionally, we demonstrate that different levels of ballot freedom can be achieved depending on the available datasets. To the best of our knowledge, our constructions are the first to demonstrate anamorphism within the context of a combination of cryptographic primitives. Thus, these constructions may also prove useful in domain beyond voting.

4.2 Anamorphic Voting

We provide a primer of anamorphic encryption, from which we informally introduce anamorphic voting and ballot freedom, leaving a formal treatment for future work. We then discuss how to make CHVote [68] anamorphic, achieving ballot freedom using different cryptographic primitives.

4.2.1 From Anamorphic Encryption to Anamorphic Voting

Anamorphic encryption [33] enables entities to encrypt hidden messages evading the censorship imposed by a dictator within a well-established public-key cryptosystem. In its

stronger formulation, the dictator owns the receiver's secret key (no receiver privacy) and can force the sender to send ciphertexts over a channel controlled by the dictator (no sender freedom), although sender and receiver might have previously exchanged keys over private channels.

In strongly secure anamorphic encryption, the dictator forces the sender to encrypt a message fm using a forced public key fpk for which the dictator knows the corresponding private key fsk . The sender can evade censorship by generating randomness for the encryption using a coin-toss faking algorithm $fRandom$, which takes as input fm , fpk , a double public key dpk (with the corresponding double private key dsk known only by the receiver) and a hidden message dm , and generates an anamorphic ciphertext that gives fm when decrypted with fsk , and dm when decrypted with dsk by the receiver.

In anamorphic voting, we extend the concept of anamorphic encryption to support the multiple cryptographic primitives that constitute a voting scheme, that is, we split anamorphic encryptions across multiple components, such as multiple ciphertexts, ciphertext and signature, or ciphertext and proof of knowledge. This approach allows entities to encrypt hidden messages, evading the censorship imposed by dishonest authorities within a well-established voting scheme.

Anamorphic voting is agnostic about the participants. It can apply to scenarios where, for instance, a voter wants to communicate with another voter or an honest authority with another honest authority. To address the specific case of voters conveying their true vote intentions to an auditor despite the presence of dishonest authority, we introduce the concept of ballot freedom.

Ballot freedom assumes the following strong threat model:

- The auditor double public key dpk is known to everyone, including voters and authority.
- The authority executes the voting scheme according to its prescribed specifications, including setup parameters.
- The authority knows at least the election decrypting key and may hold additional keys based on the roles in the voting scheme.
- The authority has access to all data generated within the voting scheme.
- The voter can only communicate through channels controlled by the authority (i.e. no anonymous, private, or untappable channels exist).

A notable distinction from anamorphic encryption is that ballot freedom does not assume the existence of private channels.

4.2.2 A Running Example: CHVote

CHVote [68] has been one of the two major e-voting system proposals in Switzerland. The system is universally verifiable, which means that independent verifiers can confirm the correctness of the election results. Verification ensures that only votes cast by eligible voters are included, that no voter casts more than one vote, and that every valid vote is tallied as recorded.

The election process in CHVote involves several key parties: Election Administrator, Election Authority, Printing Authority, Voter, and Verifier. CHVote is designed to support various election formats, including multiple elections and multiple answers. For simplicity, we focus on the case where voters cast a vote for a single candidate among n possible candidates. It relies on several cryptographic primitives, including ElGamal encryption [71], Schnorr signatures, and Schnorr identification proofs [72].

CHVote can be divided in three phases as follows:

Preparation Phase. The election administrator defines the election parameters, such as the number of candidates n , and works with the election authorities to generate a public encryption key pk . Each authority prepares partial verification codes for each voting option, which are hashed and combined into a single verification code v_c . The printing authority collectively generates and prints eligibility, confirmation, and validity data on the voter's election card.

Vote Casting Phase. To cast a vote, the voter provides their choice m to the voting client, which generates a ballot $\alpha = (\hat{x}, b, pk)$. Here, $\hat{x}$ is the voter's public credential derived from their private credential x , and $b = (b_1, b_2) = \text{Enc}(pk, m; r) = (g^r, m \cdot pk^r)$ is an Oblivious Transfer (OT) query constructed using ElGamal encryption. The ballot also includes a non-interactive zero knowledge proof (NIZKP). The authorities, acting as OT senders, respond to this query, enabling the voting client to derive verification codes for the chosen candidate. Additionally, the ballot contains a Schnorr-like identification proof linking the public credential $\hat{x}$ and the encrypted vote. Let H be a secure hash function, and let $\hat{g}$ be the generator of $\mathbb{Z}_{\hat{p}}$, where $\hat{q} \mid \hat{p} - 1$, the NIZKP, consisting of three combined proofs, is

generated as follows:

- pick $w_1 \xleftarrow{\$} \mathbb{Z}_{\hat{q}}$, $w_2 \xleftarrow{\$} \mathbb{Z}_p$, $w_3 \xleftarrow{\$} \mathbb{Z}_q$
- compute $t_1 = \hat{g}^{w_1} \bmod \hat{p}$, $t_2 = w_2 \cdot pk^{w_3} \bmod p$, $t_3 = g^{w_3} \bmod p$
- compute $s = H(\alpha, t_1, t_2, t_3)$
- compute $r_{v_1} = w_1 - s \cdot x \bmod \hat{q}$, $r_{v_2} = w_2 \cdot m^{-s}$, $r_{v_3} = w_3 - s \cdot r \bmod q$
- return $\pi = (r_v, s) = ((r_{v_1}, r_{v_2}, r_{v_3}), s)$

To verify the proofs, one computes $t'_1 = \hat{x}^s \cdot \hat{g}^{r_{v_1}}$, $t'_2 = b_2^s \cdot r_{v_2} \cdot pk^{r_{v_3}}$, and $t'_3 = b_1^s \cdot g^{r_{v_3}}$, and then checks if $s = H(\alpha, t'_1, t'_2, t'_3)$.

Tally Phase. After the election, authorities anonymize the ballots via a verifiable mixnet and collectively decrypt them using private key shares. The administrator finalizes the decryption and publishes the results, which independent verifiers validate for correctness and integrity. Several data sets, including encrypted votes and their proofs, are published for auditing purposes, as detailed in Table 7.2 of [68].

4.2.3 Ballot Freedom in CHVote

CHVote achieves ballot freedom thanks to an anamorphic construction that combines ElGamal and the Schnorr-like identification proof as follows.

Construction 1 (combining ElGamal and Schnorr). The voter chooses a random $w_1 \xleftarrow{\$} \mathbb{Z}_{\hat{q}}$ and then computes the ElGamal randomness as $r := \hat{H}(dpk^{w_1} \cdot g^{dm})$, where $\hat{H}$ is a hash function that maps group elements into the randomness space, $dpk = \hat{g}^{dsk}$ is the auditor double public key, and dm is the double (short) message or private vote chosen by the voter. Then, the voter computes the NIZKP and the ElGamal ciphertext $ct := \text{Enc}(pk, m; r)$ accordingly to the CHVote protocol specifications outlined above.

Given the public available proof $\pi = ((r_{v_1}, r_{v_2}, r_{v_3}), s)$ and ballot $b = (b_1, b_2) = (g^r, m \cdot pk^r)$, the auditor can retrieve the double message dm using the procedure $\text{dDecS}(dsk, (b, (r_{v_1}, s), \hat{x}))$ as detailed in Algorithm 1. This works because $b_1 = g^t = g^{\hat{H}(t_1^{dsk} \cdot g^{dm})} = g^{\hat{H}(\hat{g}^{w_1 \cdot dsk} \cdot g^{dm})} = g^{\hat{H}(dpk^{w_1} \cdot g^{dm})} = g^r$. Note that the authority

knows all keys, including dpk , except dsk . The construction establishes a level of robustness, ensuring that the auditor, and only the auditor, can retrieve and understand if the encrypted message contains or not a double message.

This particular construction requires that the voter private credential x is not known to the authority. If so, given the ballot signature (r_v, s) the authority can compute $w_1 := s \cdot x + r_{v_1}$ and check whether $b_1 = g^{\hat{H}(dpk^{w_1} \cdot g^{dm})}$. To avoid this, the voter can instead link the randomness in the ElGamal encryption to the proof of knowledge of that very randomness, namely, r_{v_3} . The voter chooses a random $w_3 \xleftarrow{\$} \mathbb{Z}_q$ and computes $r = \hat{H}(dpk^{w_3} \cdot g^{dm})$. The proof $\pi = ((r_{v_1}, r_{v_2}, r_{v_3}), s)$ is generated as before, and the auditor retrieves the double message dm using $dDecS(dsk, (b, (r_{v_3}, s), \mathbf{b}_1))$.

Algorithm 1 $dDecS(dsk, (b, (r_{v_1}, s), \hat{x}))$ // (Schnorr)

```

1:  $b := \{b_1, b_2\}$ 
2:  $t_1 := \hat{x}^s \cdot \hat{g}^{r_{v_1}}$ 
3: for  $i \in \mathbb{M}$  do
4:    $t := \hat{H}(t_1^{dsk} \cdot g^i)$ 
5:   if  $b_1 = g^t$  then
6:     return  $i$ 
7:   end if
8: end for
9: return  $\perp$ 
```

4.3 More Anamorphic Constructions

We provide a few more generic anamorphic constructions that can be used to achieve ballot freedom in other voting schemes.

Common to all the constructions are the forced message $fm \in \mathbb{M}$, the attacker key pair $(fpk, fsk) \leftarrow \text{EKeyGen}()$, the receiver double key pair $(dpk, dsk) \leftarrow \text{EKeyGen}()$, the double message $dm \in \mathbb{M}$, and a hash function $\hat{H}$ that maps group elements into the randomness space. $\text{EKeyGen}()$ is the ElGamal key generation function that outputs a pair of keys $(sk, pk = g^{sk})$ where $sk \xleftarrow{\$} \mathbb{Z}_p$.

Also here the attacker knows all keys except dsk and all constructions establish robustness, ensuring that decrypting anamorphic ciphertexts with the correct double key results in a valid message, otherwise an explicit abort signal is returned.

Algorithm 2 $\text{dDec}(dsk, (ct_1, ct_2))$ (ElGamal)

```

1:  $ct_1 := (c_{11}, c_{12})$ 
2:  $ct_2 := (c_{21}, c_{22})$ 
3:  $t := c_{11}^{dsk}$ 
4: for  $i \in \mathbb{M}$  do
5:    $k := \hat{H}(t \cdot g^i)$ 
6:   if  $c_{21} = g^k$  then
7:     return  $i$ 
8:   end if
9: end for
10: return  $\perp$ 

```

Algorithm 3 $\text{fRandom}(dpk, dm)$ // (ℓ -bit message)

```

1:  $dm := b_1 \| b_2 \| \dots \| b_\ell \in \{0, 1\}^\ell$ 
2: for  $i \in [\ell + 1]$  do
3:   if  $i = 1$  then
4:      $r_i \xleftarrow{\$} \mathbb{Z}_q$ 
5:   else
6:      $r_i := \hat{H}(dpk^{r_{i-1}} \cdot g^{b_{i-1}})$ 
7:   end if
8: end for
9:  $R \leftarrow \{r_i\}_{i \in [\ell+1]}$ 
10: return  $R$ 

```

Construction 2 (combining two ElGamal ciphertexts). The sender chooses a random $r_1 \xleftarrow{\$} \mathbb{Z}_q$ and computes $ct_1 = \text{Enc}(fpk, fm; r_1)$. Given another forced message $fm' \in \mathbb{M}$, the sender computes $r_2 \leftarrow \hat{H}(g^{dm} \cdot dpk^{r_1})$ and ciphertext $ct_2 = \text{Enc}(fpk, fm'; r_2)$, and sends (ct_1, ct_2) . The receiver retrieves dm using the decryption function outlined in Algorithm 2. This works because $c_{21} = g^k = g^{\hat{H}(g^{dm} \cdot c_{11}^{dsk})} = g^{\hat{H}(g^{dm} \cdot dpk^{r_1})} = g^{r_2}$. Note that if $dm \in \mathbb{M} \setminus \{0\}$, r_2 can be computed as $r_2 \leftarrow \hat{H}(dm \cdot dpk^{r_1})$ and the receiver can avoid the exponentiations in step 5 in Algorithm 2.

Since the receiver must compute modular exponentiations to retrieve the message, the double message space is expected to be small, which is generally acceptable in voting, as the message space for the candidates is typically small.

Construction 3 (combining $\ell+1$ ElGamal ciphertexts). Alternatively, the sender can anamorphically send an ℓ -bit long double message dm by casting $\ell + 1$ ciphertexts. The sender selects the set of randomness $R := \{r_i\}_{i \in [1, \ell+1]}$ using the fake randomness generator as described in Algorithm 3, which is a simplified version of the sender-anamorphic

Algorithm 4 dDecB(dsk, CT) // ℓ -bit message

```

1:  $CT := \{(c_{i1}, c_{i2})\}_{i \in [1, \ell+1]}$ 
2: for  $i \in [2, \ell + 1]$  do
3:    $b_i := \{c_{i1}, c_{i2}\}$ 
4:    $t_i^0 := \hat{H}(c_{1i-1}^{dsk})$ 
5:    $t_i^1 := \hat{H}(c_{1i-1}^{dsk} \cdot g)$ 
6:   if  $c_{1i} = g^{t_i^0}$  then
7:      $b'_{i-1} := 0$ 
8:   else if  $c_{1i} = g^{t_i^1}$  then
9:      $b'_{i-1} := 1$ 
10:  else
11:    return  $\perp$ 
12:  end if
13: end for
14: return  $b'_1 \| b'_2 \| \dots \| b'_\ell$ 
    
```

extension for hybrid PKE with special KEM from [73]. Differently from [73], our fake randomness generator does not require key encapsulation. Algorithm 3 takes in dpk and dm , and outputs R to be used in the ElGamal encryptions. In fact, it creates a link between the randomness used in the encryptions. This is only noticeable by the receiver owning the double secret key dsk . The receiver can decrypt dm using Algorithm 4.

Construction 4 (combining ElGamal and DSA). The last construction consider the DSA signature scheme, which we briefly discuss as follows.

DSA Signature. The DSA signature consists of three algorithms as:

- **DKeyGen**(1^λ): on input of a security parameter 1^λ , outputs a pair of keys $(vsk, vpk = g^{vsk})$ where $vsk \xleftarrow{\$} \mathbb{Z}_q$, with (p, q, g) satisfying $q \mid (p - 1)$ and g being a generator of the subgroup of order q in $\mathbb{Z}_p^*$.
- **DSign**($vsk, m; w$): which, given a signing key vsk , a message $m \in \mathbb{M}$, and randomness $w \xleftarrow{\$} \mathbb{Z}_q$, and a hash function H , outputs a signature (r_v, s) where $r_v = (g^w \bmod p) \bmod q$, and $s = w^{-1}(H(m) + vsk \cdot r_v) \bmod q$.
- **DVerify**($vpk, m, \sigma = (r_v, s)$): given a verification key vpk , a message m , and a signature σ , outputs $\top$ (valid) if $r, s \in \mathbb{Z}_q \setminus \{0\}$, $v = H(m)s^{-1} \bmod q$, $u_1 = g^v \bmod p$, $u_2 = vpk^{(r_v \bmod q)} \bmod p$, and $(u_1 \cdot u_2 \bmod p) \bmod q = r_v$. Otherwise, it outputs $\perp$.

Algorithm 5 dDecDSA($dsk, ct, (r_v, s)$) (DSA)

```

1:  $ct := \{c_1, c_2\}$ 
2: for  $i \in \mathbb{M}$  do
3:    $t := \hat{H}(c_1^{dsk} \cdot g^i)$ 
4:   if  $r_v = g^t$  then
5:     return  $i$ 
6:   end if
7: end for
8: return  $\perp$ 

```

The sender chooses a random $r \xleftarrow{\$} \mathbb{Z}_q$ and computes $ct := \text{Enc}(fpk, fm; r)$. Given a (forced) message sm , the sender computes $w := \hat{H}(g^{dm} \cdot dpk^r)$, generates a DSA signature $(r_v, s) := \text{DSign}(vsk, sm; w)$ and sends $(ct, (r_v, s))$. The receiver retrieves dm using the decryption function outlined in Algorithm 5. This works because $r_v = g^t = g^{\hat{H}(g^{dm} \cdot c_1^{dsk})} = g^{\hat{H}(g^{dm} \cdot dpk^r)} = g^w$.

All the constructions above are very efficient. For Construction 1, 2, and 4, the sender (voting device) just needs an extra exponentiation to send an anamorphic ciphertext. All the burden is on the receiver (auditor), who needs to perform a linear number of exponentiations in the size of the double message. Therefore, the double message space for this constructions is expected to be small, like a vote preference. For Construction 3, the sender needs to perform $2 \times \ell$ extra exponentiations per bit-message. For example, to send a 32-bit long double message, the sender needs to perform only 64 extra exponentiations. However, it is different if we consider the burden on the voter rather than the voting device. In this case, it mainly depends on the specificities of the voting system, its implementation, and its user interface.

4.4 The Estonian Internet Voting System

The Estonian Internet Voting System, IVXV [67], allows voters to cast their vote online. To sign and encrypt a ballot, it uses a digital signature scheme and a homomorphic public-key encryption scheme, which is instantiated with ElGamal. Proof of shuffle and proof of correct decryption are also employed.

4.4.1 Protocol Description

IVXV is based essentially on the following participants: the election organizer, the registration service, the voter, the vote collector, the ballot box processor (IBBP), the mixing service,

and the data auditor. The protocol can be divided into five distinct stages: initialization, voting, post-voting, tallying, and auditing.

Initialization. Before voting begins, the election organizer sets up and configures the IVXV system. In particular, this process includes generating cryptographic keys that will later be used to encrypt and decrypt votes as follows. On input of the security parameter 1^λ , electoral roll $\mathbb{V}$, and candidate list $\mathbb{C}$, the election organizer computes $(pkE_T, skE_T) \xleftarrow{\$} \text{EKeyGen}(1^\lambda)$. Moreover, each voter v has access to their signing key pair (spk_v, ssk_v) , which were generated by governmental service by computing $(spk_v, ssk_v) \xleftarrow{\$} \text{DKeyGen}(1^\lambda)$. In the IVXV implementation, the signature scheme is RSA-SHA256 (for authentication with ID card or digital ID) or ECDSA-SHA256 (for authentication with mobile ID). Here, we consider only the latter.

Voting. An eligible voter $v \in \mathbb{V}$ encrypts their choice $c_v \in \mathbb{C}$ using the election's public encryption key pkE_T and a randomly generated value r_v resulting in ballot $b = \text{Enc}(pkE_T, c_v, r_v)$. The voter then signs the ballot b using their private signing key ssk_v and a random value rs_v . This creates $vote_v = \text{Sign}(ssk_v, b, rs_v)$. The voter submits their identifier v , a certificate proving their voting eligibility, and $vote_v$ to the vote collector (VC).

Upon receipt, VC processes the submission and responds with a unique identifier vid , used for (individual) verification of the vote. Note that a voter can cast multiple votes. The set D_{VC} contains all cast votes during this phase, stored without removal.

Post-voting. After the voting phase concludes, IBBP checks the eligibility of each vote and cross-references the registration confirmations, which are stored in D_{RS} , with the data from D_{VC} to ensure consistency. Once this verification is complete, the IBBP compiles a new list of ballots, retaining only the most recent valid vote for each voter and removing those who also cast paper ballots. Then, the IBBP anonymizes the ballots by stripping any voter-identifying information (e.g. signatures) producing B_1 . Optionally, the set B_1 can be mixed, producing an output set B_2 along with a proof of correct operation P_{mix} ensuring secure shuffling and re-encryption.

Tallying. The election organizer decrypts each choice c' and computes the voting result. Additionally, it provides a proof of correct decryption P_{dec} along with the plaintext.

Auditing. Different levels of auditing can be performed. Universal verifiability can be achieved by a *complete* audit, in which the auditor has access to all data sets D_{VC} , D_{RS} , B_1 , B_2 , P_{mix} , P_{dec} . As outlined in [67], a complete audit might leak a lot of information, such as whether a voter re-voted. This increases the risk of coercion, though the dictator already breaks ballot privacy, possibly allowing a complete audit. As an alternative, the auditor can perform a *partial* audit, in which only the data sets B_1 , B_2 , P_{mix} , P_{dec} are available.

4.4.2 Ballot Freedom in IVXV

We now explore how ballot freedom can be achieved in the IVXV system across three different scenarios as follows.

Case 1: complete audit with revoting. In this scenario, the auditor has access to all data sets, including D_{VC} containing encrypted votes and their signatures. Revoting allows the voter to cast multiple votes, which the auditor can detect. Ballot freedom can be achieved using the constructions that combine ElGamal ciphertexts outlined in Section 4.3. According to Construction 2, the voter casts two ballots to communicate a message within a small message space. According to Construction 3, the voter casts $\ell + 1$ ballots to communicate an ℓ -bit-long double message. The auditor can identify all ballots cast by the same voter by examining the signatures.

Case 2: complete audit without revoting. In this scenario, the voter casts only one vote. If the auditor has access to D_{VC} , the voter can link the randomness in the encryption and the randomness in the DSA signature using Construction 4. Interestingly, the IVXV architecture document suggests that the P-384 curve can be used for ElGamal encryption, which is the same curve commonly used in ECDSA. Therefore, they likely share the same parameters, including order and generator. Even if the parameters differ, the hash function in Construction 4 can map elements between the groups. A practical challenge arises if the signature service is provided by an app that might not allow the voter to choose the randomness. Therefore, the voter would need to a modified version of the app that supports customized randomness input.

Case 3: partial audit. In this scenario, the auditor has access to anonymized ballots in B_1 , therefore the previous approaches cannot straightforwardly signal the voter's real inten-

tion. However, if voters reveal their randomness to other voters, they could still signal their true vote intentions by using Construction 2 or Construction 3. The latter construction is more challenging because voters must reveal their randomness in sequence to form a link between them. In both cases, the decryption algorithms dDec and dDecB require the auditor to check all other ballots in B_1 for randomness links. Despite these challenges, ballot freedom remains feasible even in the partial audit scenario.

4.5 Helios

Helios [69] builds on Benaloh’s Simple Verifiable Voting [74], separating ballot preparation and casting. Ballots can be prepared and viewed without authentication, while authentication is required only for casting, ensuring auditability by allowing anyone to verify the system’s integrity. In this paper, we focus on Helios 2.0 [75], which, compared to the first protocol, replaces mixnet-based tallying with homomorphic tallying. In addition, voters now provide a NIZKP to demonstrate the validity of their encrypted votes without revealing their content.

The election process in Helios 2.0 is as follows, where the cryptographic descriptions are taken from [76]:

Ballot Casting. The voter selects their choice $m \in \{min, \dots, max\}$ and forms the vote as $v = g^m$. Then, the voter encrypts v under the trustees’ public key pk using ElGamal encryption yielding $ct = (c_1, c_2) = \text{Enc}(pk, v; r) = (g^r, g^m \cdot pk^r)$, and generates a NIZKP showing that ct is a valid encryption for a message m . The proof is generated as follows.

- For all invalid plaintexts $i \in \{m_{min}, \dots, m_{m-1}, m_{m+1}, \dots, m_{max}\}$: randomly generate a challenge $s_i \in \mathbb{Z}_q^*$ and a response $r_i \in \mathbb{Z}_q^*$. Compute the witnesses $c_{1i} = g^{r_i} / c_1^{s_i}$ and $c_{2i} = pk^{r_i} (c_2 / g^i)^{s_i}$.
- For the valid plaintext m : select a random nonce $w \in \mathbb{Z}_q^*$. Compute the witnesses $c_{1m} = g^w$ and $c_{2m} = pk^w$. Derive the challenge s_m using the hash function H :

$$s_m = H(c_{1m_{min}}, c_{2m_{min}}, \dots, c_{1m_{max}}, c_{2m_{max}}) - \sum_{i \neq m} c_i \mod q.$$

Compute the response $r_m = w + r \cdot s_m$. The proof for the vote is $\pi = (r_m, s_m)$

Before casting, the voter has the option to audit the ballot to confirm it accurately represents their intended vote. During this process, the randomness used in creating the ballot is revealed, allowing independent verification of its correctness.

Once satisfied, the voter submits their ballot to the election authority. The election authority authenticates the voter, verifies their eligibility, and checks the validity of the NIZKP. A valid ballot is published on the bulletin board along with the voter's identity.

After submission, the voter can verify that their ballot appears on the bulletin board. Observers can confirm that the proofs attached to each ballot are valid, ensuring all published ballots represent legitimate votes.

Tallying. After the voting period ends, the election authority homomorphically combines all the encrypted ballots to produce an encrypted tally. The encrypted tally is published on the bulletin board.

Each trustee publishes a partial decryption of the encrypted tally. Alongside the partial decryption, trustees include a signature of knowledge to prove its correctness. These proofs are publicly available for verification. The election authority decrypts the final tally and publishes the election result.

4.5.1 Ballot Freedom in Helios

Case 1: audit with revoting. In Helios, the voter can cast multiple ballots, all of which are published on a public bulletin board. Once they authenticate a ballot, that ballot is displayed on the bulletin board as their final choice, while all previously cast ballots are archived. Since the bulletin board data is auditable by anyone at any time, the voter can achieve ballot freedom by using any constructions that combine ElGamal ciphertexts. In Construction 2, the voter casts two ballots, whereas in Construction 3, the voter casts $\ell + 1$ ballots. Note that the voter must be able to freely select the randomness r in the encryption $\text{Enc}(pk, v; r)$.

Case 2: audit without revoting. The voter can also use Construction 1 by using the ElGamal ciphertext and its corresponding proof. The voter selects $w \in \mathbb{Z}_q^*$ and computes the ElGamal randomness as $r = \hat{H}(dpk^r \cdot g^{dm})$. The voter generates the proof $\pi = (r_m, s_m)$, and when the auditor retrieves $ct = (c_1, c_2)$ and its corresponding proof π from the bulletin board, they can decrypt the double message dm using $\text{dDecS}(dsk, (ct, (r_m, s_m)), c_1)$.

4.6 Belenios

Belenios [70] is an electronic voting system built upon Helios [69] that offers vote privacy and verifiability. Similar to Helios, it uses ElGamal and there is a public bulletin board that contains the accepted ballots and the result of the voting. Schnorr signatures are used to sign an encrypted ballot to avoid ballot stuffing.

4.6.1 Protocol Description

Due to space limitations, we refer the reader to the original paper [70] for a detailed description of the protocol. Here we provide a brief overview of the protocol that is instrumental to assess ballot freedom. We briefly recall the Schnorr signature scheme as follows.

Schnorr Signature. The Schnorr signature consists of three algorithms as:

- $\text{SKeyGen}(1^\lambda)$: on input of a security parameter 1^λ , outputs a pair of keys $(vsk, vpk = g^{vsk})$ where $vsk \xleftarrow{\$} \mathbb{Z}_q$, with (p, q, g) satisfying $q \mid (p - 1)$ and g being a generator of the subgroup of order q in $\mathbb{Z}_p^*$.
- $\text{SSign}(vsk, m; w)$: given a signing key vsk , a message $m \in \mathbb{M}$, and randomness $w \xleftarrow{\$} \mathbb{Z}_q$, and a hash function H , outputs a signature (r_v, s) where $s = H(m|g^w) \bmod q$ and $r_v = w - vsk \cdot s \bmod q$.
- $\text{SVerify}(vpk, m, \sigma = (r_v, s))$: given a verification key vpk , a message m , and a signature σ , outputs $\top$ if $s = H(m|A)$ where $A = vpk^s g^{r_v}$. Otherwise, it outputs $\perp$.

Note that a variant of the Schnorr signature can be used to prove the knowledge of a discrete logarithm. For example, to prove knowledge of the randomness r used to generate an ElGamal ciphertext $ct = (c_1, c_2) = (g^r, m \cdot pk^r)$, set $vsk = r$ and $vpk = c_1$.

For each voter id , a unique signing key $sk_{id} \in \mathbb{Z}_q$ is generated. Then, the voter V encrypts the vote preference v as $ct = \text{Enc}(pk, v; r)$ and generate a signature $s = \text{SSign}(vsk, ct; w)$, where vsk is their signing key. The voter computes a hash $h = \text{hash}(b)$ to serve as a tracking number. The server appends b to the election data D . At any point, V can verify that h is included in the list of *pretty ballots* (P_B) , which consists of the hashes of the final ballots submitted by voters. If the election includes non-homomorphic ciphertexts, the trustees shuffle the ballots. The server homomorphically

combines the ciphertexts and sends the result to the trustees to compute the final election outcome.

An auditor can perform verifications during the voting phase and after the tally. During the voting phase, an auditor can retrieve the public board to verify its consistency. They check that for each ballot b , the proofs and signature of b are valid. After the tally, the auditor retrieves the public data D , including the list of ballots (B), shuffles (Σ), partial decryptions (Δ), and the result (res). They verify the consistency of B , ensure it matches previously monitored data, compute $\hat{B} = \text{last}(B)$ (the list of only the latest ballots for each voter), and confirm that the proofs in Σ , Δ , and the result res are valid with respect to $\hat{B}$.

4.6.2 Ballot Freedom in Belenios

In Belenios, the revoting policy allows voters to cast multiple ballots, with only the last ballot being considered for the tally. If all ballots are publicly available for auditing during the voting phase, a voter can use Construction 2 or Construction 3 as introduced in Section 4.3 to achieve ballot freedom.

If only the last ballots cast by each voter are available to the auditor, the voter can use Construction 1 to achieve ballot freedom. In doing so, the voter links the randomness in the ElGamal encryption to the Schnorr signature of the ballot. Similar to the case of ballot freedom in CHVote, this construction requires that the voter signing key sk_{id} is unknown to a dishonest authority. If the authority knows the key, given the ballot signature (r_v, s) , it can compute $w := s \cdot sk_{id} + r_v$ and check whether the first element of the ElGamal encryption of the vote $ct := (c_1, c_2)$ is $c_1 = g^{\hat{H}(dpk^w \cdot g^{dm})}$. In this case, the voter can instead link the randomness in the ElGamal encryption to the proof of knowledge of that very randomness. The proof is generated as per the Schnorr signature by substituting sk_{id} with the ElGamal randomness r and vpk with c_1 .

4.7 Related Work

Dishonest authorities can compromise the security of voting systems in various ways. Bernhard et al. [77] showed that weak Fiat-Shamir heuristics lead to security breaches in Helios when malicious trustees are present. Similarly, Cortier et al. [78] demonstrated that an attacker who corrupts both the voting server and trustees can break verifiability in Belenios if a variant of Fiat-Shamir heuristic is used. Haines et al. [79] showed that having honest

trustees is sometimes insufficient to guarantee ballot privacy. Moreover, ensuring trustworthy trustees is not a trivial task. Benaloh et al. [80] examined the practical challenges that trustees face in preserving the privacy of votes.

There is a long stream of research on covert and subliminal channels in voting [81, 82, 83] but they all focus on the maliciousness of the channel rather than the ability of the voter to communicate their true intention.

Coercion, in general, requires that trust assumptions be carefully vetted. Finogina et al. [84] demonstrated that in coercion-resistant voting, a coercer can utilize new cryptographic tools to prevent voters from evading coercion. Tally-hiding [85, 86] and cleansing-hiding [87] voting schemes typically rely on MPC to mitigate coercion even in the presence of dishonest trustees. All the schemes outlined above assume at least one honest trustee and/or a private or anonymous channel. Budurushi et al. [88] proposed a voting scheme in which the attacker controls all channels but not all tally servers. Similarly, Caveat coercitor [89] does not require a private channel but assumes the existence of at least one honest trustee. Solutions involving DRE machines without tallying servers have been proposed [90, 91]. These schemes assume a private voting booth and that the machines do not reveal the voter’s choice. The novel aspect of our work is to demonstrate that a voter can still achieve privacy in communicating with an auditor using an existing voting scheme, even when the attacker controls all communication channels and election keys.

4.8 Discussion and Future Work

In this work, we introduced the concept of anamorphic voting, which studies whether a combination of cryptographic primitives in a voting scheme can be used to send covert messages in the presence of dishonest authorities. We demonstrated that IVXV, CHVote, Belenios, and Helios each enable voters to freely communicate their true voting intention to an auditor without the knowledge of dishonest authorities, a property we refer to as ballot freedom.

Anamorphic voting is in its infancy, and several future research directions can be explored. One is to investigate ballot freedom in voting schemes recently implemented in real-world elections, such as ElectionGuard [92], or in schemes that rely on cryptographic primitives not covered in this work, such as commitment schemes and lattice-based cryptography. Another direction involves formalizing the concept of ballot freedom to enable

proofs of whether a given voting scheme achieves or fails to achieve ballot freedom. Formalization can also help clarify the relationship between ballot freedom and other security properties, such as receipt-freeness. Intuitively, ballot freedom appears to be orthogonal to receipt-freeness: if a voting scheme supports anamorphic voting, a coerced voter could reveal their ballot to the coercer (auditor) as proof! However, our constructions might provide a form of coercion evidence such as in [89] but in a cleaner way.

Our constructions can be viewed as a form of strong anamorphic encryption instantiated across multiple cryptosystems. Exploring this independently of voting is worthwhile, as it may also prove useful for other secure applications.

Chapter 5

Crypto Wars in Secure Messaging: Covert Channels in Signal Despite Leaked Keys

Abstract	80
5.1 Introduction	81
5.1.1 Motivation.....	81
5.1.2 Contributions	82
5.2 Background	83
5.2.1 Cryptographic functions	83
5.2.2 The Double Ratchet protocol.....	85
5.2.3 Anamorphic encryption	88
5.3 System Design	89
5.4 Covert Channel in DH Ratchet	90
5.4.1 Base Construction.....	90
5.4.2 Stronger Security Guarantee – IND-CCA	94
5.4.3 Instantiation of Double Key dk	94
5.4.4 Optimization	95
5.5 Covert Channel in Symmetric Ratchet	97
5.5.1 Base Construction.....	97

5.5.2 Stronger Security Guarantee – IND-CCA	102
5.6 Performance	103
5.7 Related Work	106
5.8 Conclusions	108
Appendix	109

Abstract

End-to-end encryption (E2EE) is the foundation of modern secure messaging, with the Signal protocol as the de facto standard in applications such as Signal, WhatsApp, Facebook Messenger and Google Messages. At the same time, the deployment of E2EE has led to growing pressure from authorities to decrypt user traffic under lawful enforcement. This raises a critical question: if an adversary can routinely decrypt Signal messages (for example via a mandated access or a leaked key), can users still communicate securely and covertly?

We address this question through the lens of anamorphic encryption, which enables hidden communication within seemingly legitimate ciphertexts, even against an adversary who can decrypt them. We design two constructions that embed covert channels into the existing Signal Double Ratchet protocol. Concretely, we show how to embed covert messages (i) into Diffie-Hellman keys used in the asymmetric ratchet, or (ii) into authentication tags produced in the symmetric ratchet. Our techniques are compatible with existing Signal-style deployments and require no changes by the service provider.

We formalize security in threat models that capture adversaries with decryption capabilities granted through lawful-access mechanisms, and prove that the resulting protocol transcripts are indistinguishable from those of standard Signal. We implement our constructions in the official Signal library and Android client, and show that they incur low overhead and are practical in real-world settings. Our results show that covert communication channels can persist even when conventional E2EE guarantees are compromised.

5.1 Introduction

Secure messaging has become a central part of modern digital communication, especially as users grow more aware of privacy threats and surveillance. Protocols such as Signal aim to protect users by offering strong end-to-end encryption (E2EE), ensuring that only the sender and receiver can access the contents of a message. Even if the messaging server is compromised, or if an adversary tries to intercept communication, these protocols aim to keep messages private and secure. The Signal protocol stands out as the most widely adopted secure messaging protocol today. It is used not only in the Signal app itself but also as the foundation for messaging in WhatsApp, Facebook Messenger, Google Message and other major platforms. Signal’s design combines two main components: the X3DH key agreement protocol [93] and the Double Ratchet algorithm [94]. Together, they allow for asynchronous messaging with forward secrecy, post-compromise security, and cryptographic deniability. At the heart of Signal’s security is the Double Ratchet, which updates keys after each message to protect future and past communication, even if a device is temporarily compromised. These properties have been the subject of extensive security analysis and are widely considered a strong defense against powerful attackers. [95, 96, 97, 98, 99, 100, 101]

5.1.1 Motivation

E2EE messaging is increasingly targeted by political and institutional efforts to introduce surveillance, typically by weakening or bypassing E2EE, often justified as lawful access or national security. Among the various use cases, the most notable are the efforts by the EU [102], UK [103], Sweden [104], and more recently the Australian government [105], asking Signal to undermine its E2EE to allow the government to decrypt all encrypted messages.

As a concrete illustration of this pressure, the EU’s so-called *chat control* policy, based on client-side scanning, moves inspection onto users’ devices [106]. This approach has been challenged by cryptography and security researchers who argue that on-device scanning weakens E2EE, erodes privacy, enlarges the attack surface, and normalizes surveillance with a high risk of function creep ¹.

We consider a general model that, when activated, enables the authorities to decrypt

¹Security Scientists Open Letter, accessed Nov 2025

messages. How this capability is achieved is left open. It may arise through client-side scanning hooks, through a platform-run key escrow where user keys are stored encrypted under a law enforcement public key with access granted only on the basis of a warrant, or through other policy-driven mechanisms under active debate worldwide. Our approach is general and does not depend on how surveillance is realized. We ask whether covert, indistinguishable communication can persist within the Double Ratchet under this key disclosure model.

This paper addresses a fundamental question:

If Signal's E2EE is compromised under authoritarian control, can users still communicate securely and undetectably?

Our goal is to design mechanisms that enable hidden communication between sender and receiver, while preserving the normal behavior of the protocol, even when Signal is no longer trustworthy in the traditional sense. We propose two constructions that preserve secure and undetectable communication under key disclosure. In both cases, we assume the message decryption key, possibly along with other user secrets, is revealed, so the confidentiality of ordinary Signal messages is broken. We achieve this goal with *anamorphic encryption* [33], a recent approach that allows the sender and receiver to have covert communication by hiding sensitive messages inside other normal-looking ciphertexts, even when the adversary can decrypt messages. Our proposed constructions produce cryptographic elements that are indistinguishable from normal Signal outputs, making hidden communication possible without changing the structure or behavior of the protocol.

5.1.2 Contributions

We design and integrate two novel anamorphic channels into the Signal Double Ratchet protocol. The first construction embeds hidden messages into the Diffie-Hellman (DH) key pairs generated in the asymmetric ratchet; the second embeds hidden data into MAC authentication tags in the symmetric ratchet. These constructions are compatible with existing Signal implementations, require no structural changes, and rely solely on standard cryptographic assumptions. To the best of our knowledge, no previously proposed anamorphic construction can be applied to the Signal Double Ratchet protocol or to an existing E2EE messaging application.

Each construction targets a key-exposure scenario. For the DH-based channel, we support either a symmetric double key shared by the endpoints or an asymmetric double

public key published by the endpoints, covering cases where the server-side leakage includes Double Ratchet state but excludes the hidden double secret. For the MAC-based channel, we address settings in which either the root key or the private DH ratchet key is leaked, but not both, so that the derived MAC keys remain secret.

We formalize security via indistinguishability of anamorphic executions from standard Signal Double Ratchet transcripts, which will be referred as anamorphic security. And we extend the base constructions with transcript-binding upgrade such that the construction resist attacks in an IND-CCA2-style sense.

We provide concrete instantiations and optimizations. The DH-based channel is instantiated over X25519 using Edwards–Montgomery mappings, together with a lookup-table optimisation that replaces an exhaustive search over messages with one group operation and a table lookup, making a 32-bit covert payload per DH step practical. The MAC-based channel supports payloads up to the MAC size; in our implementation this is 64 bits.

We implement both channels in a Signal-compatible Double Ratchet library and Android client, evaluate their performance, and observe low overhead. The designs require no server-side changes, and remain compatible with existing Signal-style deployments.

These results indicate that covert messaging remains feasible even when traditional E2EE guarantees are compromised, and they extend the scope of anamorphic encryption to real-world secure messaging systems.

5.2 Background

This work primarily focuses on the two main components of the Double Ratchet protocol, namely, DH and symmetric ratchets. Therefore, we focus on the cryptographic primitives used in these two components, and refer the reader to [94] for additional details.

5.2.1 Cryptographic functions

The Double Ratchet protocol is primarily built upon ECDH key exchange and the HKDF key derivation function [107]. In the following, we assume familiarity with standard cryptographic notions and outline the relevant cryptographic functions, specifying for each its interface, implementation, and key features.

- $\text{Keygen}(\lambda) \rightarrow (\text{sk}, \text{pk})$: Given a security parameter λ , the key generation algorithm outputs a key pair (sk, pk) . In the DH ratchet we recall later, it generates an ECDH

key pair using either the X25519 or X448 function, denoted as $(sk, pk := sk \cdot G)$, where G is the base point of the elliptic curve, $sk \in \mathbb{Z}_p$ is a randomly chosen private key, and p is the prime order of the subgroup, denoted as G with a generator G . The default curve in the Signal protocol is Curve25519. We write $a \cdot G$ as a shorthand notation for G^a , so that $(a \pm b) \cdot G$ represents $G^{a \pm b}$.

- $DH(sk_a, pk_b) \rightarrow dh_{out}$: Given A's secret key sk_a and B's public key pk_b , the DH function computes DH shared secret $dh_{out} := sk_a \cdot pk_b$. Note that $DH(sk_a, pk_b) = DH(sk_b, pk_a)$.
- $PRF(prf_{key}, x) \rightarrow y$: Given a key prf_{key} and an input x , the pseudorandom function (PRF) outputs an element y . It is required that the output be computationally indistinguishable from a value from a truly random function mapping from the same domain to the same codomain.
- $KDF(kdf_{key}, x) \rightarrow y$: Given a key kdf_{key} and an input x , the key derivation function outputs an element y . It is required that the output be computationally indistinguishable from a value sampled uniformly at random from the same range. Moreover, the KDF output can be split into several parts while the knowledge of the partial output should not leak information on the other generated part.

The KDFs for the double ratchet protocol are HMAC-based KDF, consisting of two components: The first extract function takes as input the salt and the input key material, producing a PRF key. Then, the expand function takes the PRF key (along with the implicit auxiliary information and the desired output length) as input, and outputs the derived key. We specify each KDF involved in the ratchet as follows.

- $KDF_{rk}(rk, dh_{out}) \rightarrow (rk_{new}, ck)$: Given an old root key rk as salt and the shared secret dh_{out} as input key material, the HMAC-based key derivation function (HKDF) generates a new root key rk_{new} and a new chain key ck .
- $KDF_{ck}(ck) \rightarrow (ck_{new}, mk)$: Given the old chain key ck , the HKDF generates a new chain key ck_{new} and a message key mk . In this function, a constant value is used as key salt.
- $Enc(mk, pt, ad) \rightarrow (aes_{key}, mac_{key}, IV, ct, mac)$: Given a message key mk , a plaintext pt and the associated data ad , the encryption algorithm, implemented as Authenticated Encryption with Associated Data (AEAD) in the protocol, outputs an AES key aes_{key} ,

a MAC key mac_{key} , the initialization vector IV , a ciphertext ct and a MAC mac . It proceeds in the following three steps, combining CBC and HMAC to ensure misuse resistance in cases where a key is inadvertently reused:

1. $\text{KDF}_{\text{mk}}(\text{mk}) \rightarrow (\text{aes}_{\text{key}}, \text{mac}_{\text{key}}, \text{IV})$: Given mk as input key material, the HKDF outputs aes_{key} , mac_{key} and IV . The HKDF salt is a zero-filled sequence that matches the output length of the hash function;
2. $\text{AES}_{\text{CBC}}(\text{aes}_{\text{key}}, \text{IV}, \text{pt}) \rightarrow \text{ct}$: Given aes_{key} , IV and pt , the encryption function, AES-256 in CBC mode with PKCS#7 padding, outputs a ciphertext ct ;
3. $\text{MAC}(\text{mac}_{\text{key}}, \text{ct}, \text{ad}) \rightarrow \text{mac}$: Given mac_{key} , ct and ad , the MAC function returns a mac .

5.2.2 The Double Ratchet protocol

Many messaging applications use the Signal’s Double Ratchet protocol to generate message keys in support of E2EE. This protocol ensures secure messaging by constantly updating encryption keys. It provides both forward secrecy and post-compromise secrecy. In a nutshell, every time a message is sent, a new key is generated, limiting the exposure of past or future messages if a key is compromised. The protocol bootstraps from a shared secret that sender and receiver have previously agreed based on a secure key agreement protocol. In Signal, this protocol is X3DH [93]. The Double Ratchet protocol consists of the symmetric ratchet and the DH ratchet.

Symmetric ratchet. The goal of the symmetric ratchet is to generate a new message key for each message sent or received. It involves the use of the functions KDF_{ck} and Enc . The symmetric ratchet initiates with an old chain key, and derives a new chain key and a message key as shown in Figure 5.1.

DH ratchet. The goal of the DH ratchet is to provide forward secrecy and post-compromise security against an attacker who gains access to a chain key. In the DH ratchet, each party generates a DH key pair, which serves as their current ratchet key pair (sk, pk) . Every message is sent with a header that includes the sender’s current public key pk . When one party receives a new public key from the other, they execute DH, replacing their own key pair with a fresh one. This update triggers KDF_{ck} to generate a new chain key, ensuring continuous key update for enhanced security. The DH ratchet is illustrated in Figure 5.2.

In short, the general rules to run either ratchet can be summarised as follows:

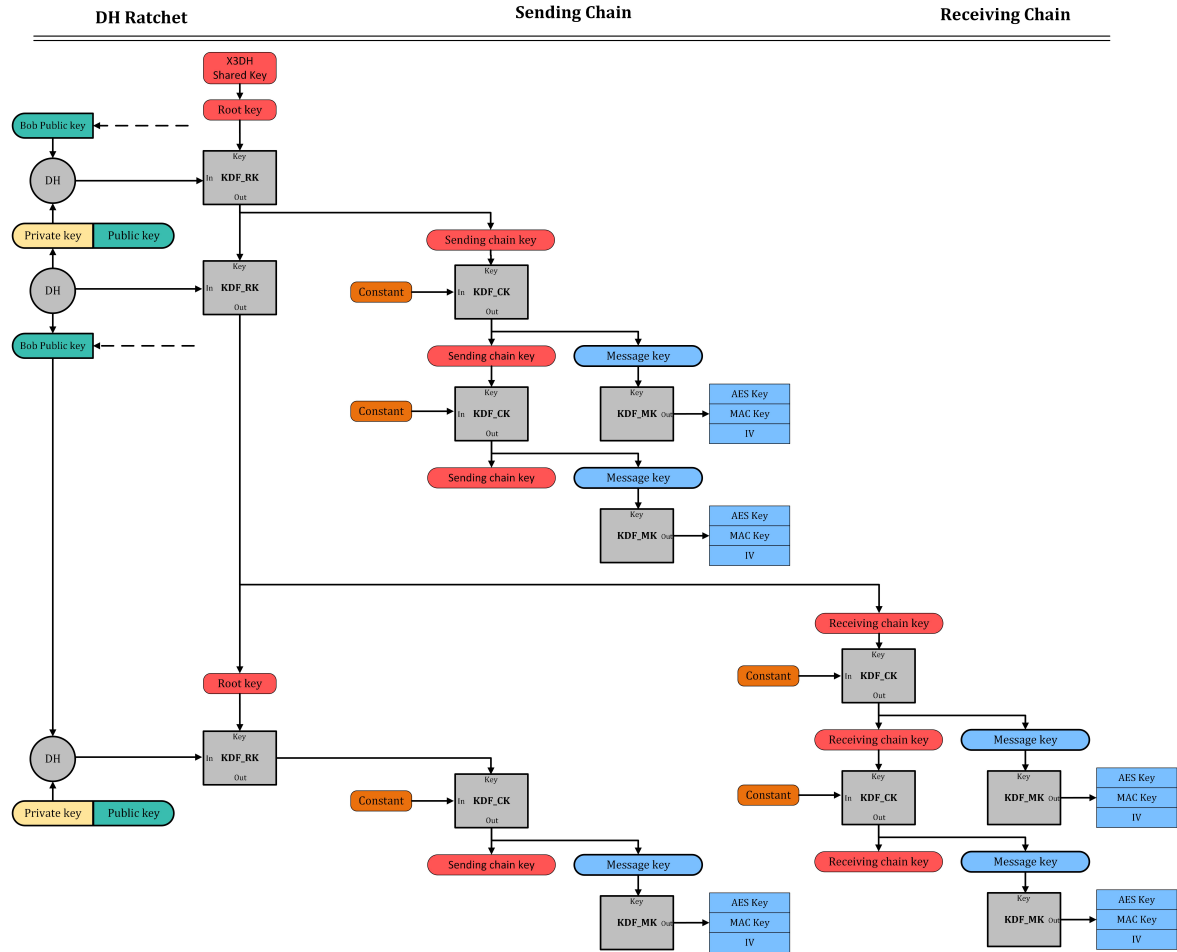


Figure 5.3: An instance of the Double Ratchet protocol for Alice

then performs a DH ratchet step to obtain her first sending chain key. She then sends two messages within this chain. Subsequently, Bob replies with a new message, including a new ratchet public key. Alice generates a new receiving chain key and derives the corresponding message keys for the messages received from Bob. The final part of this communication involves Alice sending a message along with a new ratchet public key. Note that Bob performs the same operations on his side, generating the corresponding message keys. Since only Alice and Bob share the initial root key generated via X3DH, no one else can compute the sending and receiving keys, and consequently, the message keys. This shared secret also ensures security against man-in-the-middle attacks by the server, which only relays the communication.

Figure 5.2 also illustrates that compromising a sending or receiving chain key does not affect the security of future keys, as KDF_{ck} takes in the output of dh_{out} , which is known only to Alice and Bob.

Construction	Leaked key					No PSK required
	Root	DH private	Chain/Message	MAC	AES	
Sym-DH	✓	✓	✓	✓	✓	×
Asym-DH	✓	✓	✓	✓	✓	✓
Sym-MAC	◇	◇	×	×	✓	✓

Table 5.1: Anamorphic constructions under various threat models. ◇ allows either the Root key or the DH private key to be leaked.

5.2.3 Anamorphic encryption

Anamorphic encryption [33] protects communication even when a dictator restricts the permitted cryptographic primitives and holds the communication secret key. It augments the standard encryption mode with an anamorphic mode: a sender can use a *double key* to embed a hidden message that only the intended receiver can recover with the same key pair. To the dictator, the security of an anamorphic encryption scheme guarantees that the keys and the ciphertexts in the standard and anamorphic modes are indistinguishable.

An anamorphic encryption scheme linked to a standard encryption scheme of $(\text{Gen}, \text{Enc}, \text{Dec})$ consists of three algorithms $(\text{aGen}, \text{aEnc}, \text{aDec})$. In an anamorphic mode, the aGen algorithm returns an anamorphic public key apk , an anamorphic secret key ask and a double key dk . The aEnc algorithm encrypts a normal message msg together with an anamorphic message amsg with the double key dk into a ciphertext, which can only be decrypted by the aDec algorithm with the double key.

For the Double Ratchet, we design the three anamorphic components in a similar manner. Specifically, there is a double key generation algorithm aGen that samples a double key (pair). Using the double key (pair), the anamorphic message can be encrypted and decrypted. We omit the explicit invocation of double key generation algorithm and assume that, when necessary, the double key pair is generated separately.

5.3 System Design

The essential properties of E2EE are confidentiality, integrity, authentication, and forward secrecy [108]. Optional features include availability, loss resilience, deniability, and post-compromise security. As the main goal of surveillance is to monitor users' messages, in all our threat models we focus on the loss of confidentiality of Signal messages. In the Signal Double Ratchet this is captured by leakage of the per-message encryption key (AES key) to the server.

Scope. We consider key-disclosure mechanisms that enable decryption without changing the protocol's wire format or state machine. For example, apart from the surveillance mechanisms such as key escrow mentioned earlier, one may also consider introducing a backdoor that reveals only the decryption keys. Such designs might modify other parts of the Signal Double Ratchet and are out of scope, as their broader security impact requires separate analysis.

Adversary and setting. We model the Signal server as corrupted, giving the adversary full monitoring capability over all users' communications. We discuss our threat models considering both passive adversaries who only monitor the protocol and active ones who can also interact with the protocol.

Key-leakage scenarios. We study disclosures of the Signal Double Ratchet state to the server. Revealing some keys may enable derivation of others; for instance, exposing both the root key and the private DH ratchet key lets the adversary reconstruct chain and message keys. Throughout, we assume aes_{key} is leaked so that the server can always decrypt plaintexts. Beyond this baseline, we consider scenarios in which the DH ratchet state, the symmetric ratchet state, or both are exposed, as summarized in Table 5.1. Note that in this table, the DH private key refers to the key securely generated during the DH ratcheting process. We omit the double key generation process and assume that, depending on the construction, the double keys may or may not need to be pre-shared through a secure channel.

System overview. We present two classes of constructions that capture three scenarios, remain compatible with existing Signal deployments and require no server changes:

1. a DH-based channel with a symmetric double key shared by the endpoints;
2. a DH-based channel with an asymmetric double public key published by the receiver;

3. a symmetric-ratchet channel that embeds payloads into MAC tags.

Security goals. For our constructions, we define and prove their *anamorphic security* ensuring that an adversary cannot distinguish the anamorphic mode from the real mode, i.e., it cannot detect whether the observed data (e.g., the public and secret keys or the MACs) contain an anamorphic message or not. Anamorphic security implies IND-CPA security, which one could also define for these settings, since if it is not possible to detect the existence of a chosen anamorphic message (anamorphic security), it is also not possible to decide which of two chosen messages is embedded (IND-CPA). Moreover, we also present a transcript-binding upgrade that attains a stronger, CCA-style guarantee.

5.4 Covert Channel in DH Ratchet

To embed covert messages into the DH public keys used in the DH ratchet, we construct an anamorphic channel with a hidden double key. For now, we assume that all keys other than this double key may be leaked. Later in this section, we explain how this assumption is instantiated in different scenarios.

5.4.1 Base Construction

Recall that in the DH ratchet, fresh DH key pairs are generated regularly. We show how to create *fake* DH key pairs that embed covert messages while remaining indistinguishable from honest ones, and we present two algorithms. The first generates the fake DH key pairs and the second extracts the hidden messages.

Here, we provide an abstract description over a group G . A standard DH key generation algorithm samples $sk \leftarrow \mathbb{Z}_p$ and outputs $pk = sk \cdot G \in G$, where G is the base point of the elliptic curve. We write $a \cdot G$ as a shorthand notation for G^a , so that $(a \pm b) \cdot G$ represents $G^{a \pm b}$.

Algorithm 6 generates $\ell + 1$ key pairs that embed an anamorphic message vector $\overrightarrow{\text{ams}} \in \mathcal{M}^\ell$ using the double key $dk \in \{0, 1\}^\lambda$. Here, the pseudorandom function PRF with key space $\{0, 1\}^\lambda$ maps elements from G into $\mathbb{Z}_p$.

Algorithm 7 recovers the anamorphic message vector $\overrightarrow{\text{ams}} \in \mathcal{M}^\ell$ using the double key dk from PK consisting of $\ell + 1$ public keys pk_i .

Since it requires performing modular exponentiations to recover anamorphic messages, the anamorphic message space $\mathcal{M}$ should be small. As we show later, it is feasible to

Algorithm 6 fakeKeygen($dk, \overrightarrow{ams\bar{g}}$)

```

1: parse  $\overrightarrow{ams\bar{g}} = \{ams\bar{g}_i\}_{i=1}^\ell$ 
2:  $(sk_1, pk_1) \leftarrow \text{Keygen}(\lambda)$ 
3: for  $i \in [2, \ell + 1]$  do
4:    $sk_i = \text{PRF}(dk, pk_{i-1}) + ams\bar{g}_{i-1} \bmod p$ 
5:    $pk_i = sk_i \cdot G$ 
6: end for
7: return  $\{(sk_i, pk_i)\}_{i=1}^{\ell+1}$ 

```

Algorithm 7 aDec(dk, PK)

```

1: parse  $PK = (pk_i)_{i \in [1, \ell+1]}$ 
2: for  $i \in [2, \ell + 1]$  do
3:    $ams\bar{g}_{i-1} = \text{dlog}_G(pk_i - \text{PRF}(dk, pk_{i-1}) \cdot G)$ 
4: end for
5: return  $\{ams\bar{g}_i\}_{i=1}^\ell$ 

```

choose it up to 16 bits. Note that this and all other constructions ensure robustness: decrypting an anamorphic ciphertext under the correct double key yields a valid anamorphic message, and otherwise returns an explicit abort signal, i.e., when the discrete logarithm computation fails to produce a valid anamorphic message.

To integrate this into the full Double Ratchet protocol, Alice and Bob perform the following steps to transmit ℓ anamorphic messages within the DH ratchet using $\ell + 1$ DH keys. Without loss of generality, we present our construction in the setting where Alice sends covert messages to Bob, and assume for now that Alice and Bob share the same private double key dk .

1. Alice runs a fake key generation algorithm fakeKeygen($dk, \overrightarrow{ams\bar{g}}$) as described in Algorithm 6 with her private double key dk .
2. Alice sends $\ell + 1$ Signal messages to Bob. Before sending each message, she waits for a successful delivery notification ².
3. Bob collects the last $\ell + 1$ ratchet public keys $PK = \{pk_1, \dots, pk_{\ell+1}\}$ sent by Alice and runs aDec(dk, PK) to retrieve $\overrightarrow{ams\bar{g}}$ described in Algorithm 7.

We now argue correctness and security of the construction. Intuitively, correctness requires that, given the correct double key dk , the anamorphic messages should be recovered

²This is necessary to avoid raising suspicion from the server. If Alice sends all $\ell + 1$ messages at once, it may appear suspicious that she is not using the same sending chain key for some of them. When Bob sends a Signal message, including a successful delivery notification, he also includes a new DH ratchet public key

from the anamorphic ciphertexts, that is, from the fake DH public keys produced by our algorithms. Anamorphic security requires that it is infeasible to detect the presence of the covert channels: an adversary should not be able to distinguish honestly generated fresh key pairs and the key pairs that embed anamorphic messages.

Correctness. Each anamorphic message amsg_i is correctly recovered since for all $i \in [2, \ell + 1]$, it holds that

$$\begin{aligned}
 \text{amsg}_{i-1} &= \text{dlog}_G (\text{pk}_i - \text{PRF}(\text{dk}, \text{pk}_{i-1}) \cdot G) \\
 &= \text{dlog}_G (\text{sk}_i \cdot G - \text{PRF}(\text{dk}, \text{pk}_{i-1}) \cdot G) \\
 &= \text{dlog}_G ((\text{PRF}(\text{dk}, \text{pk}_{i-1}) + \text{amsg}_{i-1}) \cdot G \\
 &\quad - \text{PRF}(\text{dk}, \text{pk}_{i-1}) \cdot G) \\
 &= \text{dlog}_G (\text{PRF}(\text{dk}, \text{pk}_{i-1}) \cdot G + \text{amsg}_{i-1} \cdot G \\
 &\quad - \text{PRF}(\text{dk}, \text{pk}_{i-1}) \cdot G) \\
 &= \text{dlog}_G (\text{amsg}_{i-1} \cdot G).
 \end{aligned}$$

Anamorphic Security. The construction achieves anamorphic security if for every probabilistic polynomial-time (PPT) adversary $\mathcal{D}$ it holds that

$$|\Pr[\text{GReal}_{\text{Kgen}}^{\mathcal{D}}(\lambda) = 1] - \Pr[\text{GANam}_{\text{fkgen}}^{\mathcal{D}}(\lambda) = 1]| \leq \text{negl}(\lambda)$$

where the security games are defined as follows:

$\text{GReal}_{\text{Kgen}}^{\mathcal{D}}(\lambda) :$
1. $\text{dk} \leftarrow \text{aGen}(\lambda)$
2. return $\mathcal{D}^{\mathcal{O}_{\text{Kgen}}(\cdot)}(\lambda)$ where $\mathcal{O}_{\text{Kgen}}(\overrightarrow{\text{amsg}}) = \{\text{Keygen}(\lambda)\}_{i=1}^{\ell+1}$
$\text{GANam}_{\text{fkgen}}^{\mathcal{D}}(\lambda) :$
1. $\text{dk} \leftarrow \text{aGen}(\lambda)$
2. return $\mathcal{D}^{\mathcal{O}_{\text{fkgen}}(\cdot)}(\lambda)$ where $\mathcal{O}_{\text{fkgen}}(\overrightarrow{\text{amsg}}) = \text{fakeKeygen}(\text{dk}, \overrightarrow{\text{amsg}}, \ell)$

In the games defined for this construction, we explicitly include the generation of the double key dk , which was omitted in the original construction. This explicit step ensures that in both games, the double key dk in both games is of the same distribution. The value dk is consumed only by the anamorphic oracle and is not revealed to the distinguisher $\mathcal{D}$; the real oracle ignores it. Moreover, in the game here, the $\mathcal{D}$ does not require access to other keys, as they do not influence the generation of DH ratchet key pairs.

Theorem 1. *The construction present in Section 5.4.1 satisfies anamorphic security if PRF used in Algorithm 6 is a secure PRF, as introduced in Section 5.2.1.*

Proof. In the following sequence of hybrid games, let $\mathcal{D}$ be a PPT adversary and Q its total number of oracle queries.

Game 0: This is the anamorphic game $\text{GAnam}_{\text{Kgen}}^{\mathcal{D}}(\lambda)$ in the security definition, where the oracle generates (pk_i, sk_i) as described in Algorithm 6, $sk_i = \text{PRF}(dk, pk_{i-1}) + \text{amsg}_{i-1}$, $pk_i = sk_i \cdot G$.

Game 1: In the second game, we replace all applications of $\text{PRF}(dk, \cdot)$ by the application of a uniformly chosen function $\mathcal{F} : \mathbb{G} \rightarrow \mathbb{Z}_p$.

Game 2: In the third game, we define $\mathcal{F}$ iteratively by lazy sampling: each time $\mathcal{F}$ should be applied, we check if we already established a mapping for the given input a , i.e., $a \mapsto z$ is contained in some set $\mathcal{M}_{\mathcal{F}}$, and if this is the case z is the output. Otherwise, some fresh $z \leftarrow \mathbb{Z}_p$ is sampled uniformly and the mapping $a \mapsto z$ is added to $\mathcal{M}_{\mathcal{F}}$.

Game $(3, j)$ for $j \in [0, \ell Q]$: In this series of games, we slightly modify Game 2 until we end up with Game 4, by modifying the way each of the ℓQ applications of $\mathcal{F}$ is computed. We define Game $(3, 0)$ to be identical to Game 2. In Game $(3, j)$, the challenger handles the first j applications of $\mathcal{F}$ as follows: Instead of checking if the a mapping for the corresponding input is already defined in $\mathcal{M}_{\mathcal{F}}$, it just samples a uniform output and adds the corresponding mapping to $\mathcal{M}_{\mathcal{F}}$. Note that in Game $(3, \ell Q)$ all applications of $\mathcal{F}$ are replaced by uniformly sampling elements from $\mathbb{Z}_p$.

Game 4: This is the real game $\text{GReal}_{\text{Kgen}}^{\mathcal{D}}(\lambda)$ in the anamorphic security definition, where the oracle generates (pk_i, sk_i) by choosing $sk_i \leftarrow \mathbb{Z}_p$ uniformly at random and setting $pk_i = sk_i \cdot G$.

Game 0 and Game 1 are computationally indistinguishable, as a distinguisher for them could be turned in to one for the pseudo-randomness property of PRF. The games Game 1 and Game 2 are perfectly indistinguishable as Game 2 just uses an equivalent way to define $\mathcal{F}$. Game $(3,0)$ coincides with Game 2 by definition. The difference between games Game $(3, j-1)$ and Game $(3, j)$ consists in how the output of the j -th application of $\mathcal{F}$ is computed. In Game $(3, j-1)$ we check whether the input pk_{j-1} occurred before in $\mathcal{M}_{\mathcal{F}}$, while we do not perform this check in Game $(3, j)$. Observing that all pk_i for $i \leq j-1$ have been chosen uniformly at random from $\mathbb{Z}_p$, the probability that $pk_i \mapsto z$, for some z , occurs in $\mathcal{M}_{\mathcal{F}}$ is upper bounded by $\frac{i-1}{p}$. Thus, in total the probability that games Game $(3,0)$ and Game $(3, \ell Q)$ differ is upper bounded by $\frac{(\ell Q)^2}{p}$, which is negligible. Finally, note that Game $(3, \ell Q)$ and Game 4 are identical by observing that $sk_i = z_i + \text{amsg}_{i-1}$ for $z_i \leftarrow \mathbb{Z}_p$ yields a uniformly random sk_i in Game $(3, \ell Q)$. $\square$

5.4.2 Stronger Security Guarantee – IND-CCA

We now strength our security guarantee. Analogous to IND-CCA2 security for an encryption scheme, we require that security holds even against an adversary that obtains ciphertexts generated in anamorphic mode and actively interacts with the protocol. In particular, such an adversary must neither be able to distinguish anamorphic executions from honest ones nor learn any information about the embedded anamorphic messages.

Intuitively, the updated fake key generation algorithm binds the covert encoding to the current transcript and adds an extra key pair used as an integrity check. The decryption algorithm is modified accordingly to verify this binding before extracting the anamorphic message. This yields active security at the cost of only one additional key pair. We defer the construction and the security proof to Appendix 5.8

5.4.3 Instantiation of Double Key dk

So far, our covert channel in the DH keys of the DH ratchet assumes that Alice and Bob share a double key dk . We now show how to instantiate this assumption under different threat models.

Case 1: Symmetric using DH Ratchet (Sym-DH). Both parties share a pre-shared secret dk established out-of-band. Alice uses dk to generate fake DH keys encoding covert messages, and Bob uses dk to recover them. The adversary, who observes the full Double Ratchet

state but not dk , cannot detect or decode the covert channel.

Case 2: Asymmetric using DH ratchet (Asym-DH). The receiver Bob holds a double key pair (dsk_B, dpk_B) and publishes the public component dpk_B . In the same sense, Alice also publishes her public key dpk_A . Alice embeds covert messages using dpk_B , and Bob recovers them using dsk_B , as they both can compute a common “double key” used in the algorithms via $dk = DH(dsk_B, dpk_A) = DH(dsk_A, dpk_B)$. This removes the need for prior coordination and suits settings where publishing a long-term public key is feasible (e.g., a journalist or organization posting a double public key on their website).

5.4.4 Optimization

As discussed before, since recovering anamorphic messages requires modular exponentiations, the anamorphic message space $\mathcal{M}$ must be small. We now present optimized variants of Algorithms 6 and 7 that use a precomputed lookup table, which is also used in our implementation. Intuitively, the optimized algorithms optimize the anamorphic decryption by replacing an exhaustive search with a single group operation and a table lookup. We successfully tested the feasibility of this optimisation for values up to 32 bits. However, it can be extended further, since the main computational effort lies in precomputing the lookup table only once. The same lookup-table technique extends to the construction with a stronger security guarantee in a similar way. The following is a concrete instantiation over Curve25519.

Symbol	Description
G_E	Base point of the Edwards curve E .
P_E and Q_E	Points on the Edwards curve E .
$f_{E \rightarrow M}$	Map from Edwards curve to Montgomery representation. This function also provides the extra bit of information sign needed for inversion, since the Montgomery model does not retain sign information.
$f_{M \rightarrow E}$	Map from Montgomery to Edwards representation, using the supplied choice of sign for the Edwards point.
$lsb(pk_i)$	Least significant bit of pk_i .
$lookup(Q_E)$	Lookup table mapping an Edwards curve point to a message.

Table 5.2: Notation used in Algorithms 8 and 9

In Algorithm 8, each fake secret key is chosen as a PRF-derived scalar (keyed by dk and the previous public key) plus the anamorphic message. The corresponding public key is

then selected so that its least significant bit determines the correct sign to invert back to the unique Edwards point. This ensures that the public keys are distributed as in the honest scheme, while encoding a small message-dependent offset in a reversible way.

Algorithm 8 $\text{fakeKeygen}(\text{dk}, \overrightarrow{\text{amsg}}, \ell)$

```

1:  $(\text{sk}_1, \text{pk}_1) = \text{Keygen}(\lambda)$ 
2: for  $i \in [2, \ell + 1]$  do
3:    $\text{ctr} = 0$ 
4:   repeat
5:      $\text{sk}_i = \text{PRF}(\text{dk}, \text{pk}_{i-1}, \text{ctr}) + \text{amsg}_{i-1} \bmod p$ 
6:      $P_E = \text{sk}_i \cdot G_E$ 
7:      $(\text{pk}_i, \text{sign}) = f_{E \rightarrow M}(P_E)$ 
8:      $\text{ctr} = \text{ctr} + 1$ 
9:   until  $\text{lsb}(\text{pk}_i) = \text{sign}$ 
10: end for
11: return  $\{(\text{sk}_i, \text{pk}_i)\}_{i=1}^{\ell+1}$ 

```

Algorithm 9 uses dk to recompute and subtract the PRF-derived scalar from the recovered Edwards point, yielding a point of the form $\text{amsg} \cdot G_E$. Since we precomputed a lookup table mapping all candidate points $\text{amsg} \cdot G_E$ to their corresponding messages, the decryption of each anamorphic block requires only one scalar multiplication and a table lookup, without iterating over all possible messages. This makes, for example, a 32-bit anamorphic message space efficiently practical in our implementation. The correctness and security guaranties of the original construction remain unchanged.

Algorithm 9 $\text{aDec}(\text{dk}, \text{PK})$

```

1:  $\text{PK} = (\text{pk}_i)_{i \in [1, \ell+1]}$ 
2: for  $i \in [2, \ell + 1]$  do
3:    $\text{ctr} = 0$ 
4:   repeat
5:      $x = \text{PRF}(\text{dk}, \text{pk}_{i-1}, \text{ctr})$ 
6:      $P_E = f_{M \rightarrow E}(\text{pk}_i, \text{lsb}(\text{pk}_i))$ 
7:      $Q_E = P_E - x \cdot G_E$ 
8:      $\text{amsg}_{i-1} = \text{lookup}(Q_E)$ 
9:      $\text{ctr} = \text{ctr} + 1$ 
10:  until  $\text{amsg}_{i-1} \neq \perp$ 
11: end for
12: return  $\{\text{amsg}_i\}_{i=1}^{\ell}$ 

```

5.5 Covert Channel in Symmetric Ratchet

To embed hidden messages into the symmetric ratchet, we create an anamorphic channel based on the MAC key that remains unknown to the server. Only the intended receiver, who holds this key, can recover the hidden messages as part of the integrity verification. This construction, namely *Sym-MAC*, targets scenarios where *either* the root key *or* the private DH ratchet key is leaked to the server, but not both, so that the resulting chain, message, and MAC keys are never fully exposed. From the adversary's view, the MACs carrying hidden messages are indistinguishable from ordinary MACs. No pre-shared key or other public double key is required, as the anamorphic messages are embedded using MAC keys derived within the Double Ratchet.

5.5.1 Base Construction

We first describe two algorithms that specify how to generate *fake* MACs and how to extract hidden messages from them.

Algorithm 10 embeds an anamorphic message $\text{amsg} \in \mathcal{M}$ into a MAC tag under key mac_{key} by XORing amsg with an honestly computed MAC. Algorithm 11 recovers the anamorphic message $\text{amsg} \in \mathcal{M}$ by recomputing the honest MAC tag under mac_{key} and extracting amsg via XOR.

Algorithm 10 $\text{fakeMAC}(\text{mac}_{\text{key}}, \text{ct}, \text{ad}, \text{amsg})$

```
1:  $t = \text{MAC}(\text{mac}_{\text{key}}, \text{ct}, \text{ad})$ 
2:  $\text{mac} = t \oplus \text{amsg}$ 
3: return  $\text{mac}$ 
```

Algorithm 11 $\text{aDec}(\text{mac}_{\text{key}}, \text{ct}, \text{ad}, \text{mac})$

```
1:  $t = \text{MAC}(\text{mac}_{\text{key}}, \text{ct}, \text{ad})$ 
2: if  $\text{mac} = t$  then
3:   skip ▷ Not anamorphic
4: else
5:    $\text{amsg} = t \oplus \text{mac}$ 
6:   return  $\text{amsg}$ 
7: end if
```

In this construction, we eliminate the small message-space constraint required by the covert channel construction in DH keys. Here, the anamorphic message space is $\mathcal{M} =$

$\{0,1\}^\lambda$, where λ is the MAC length. In our implementation, we embed 64-bit hidden messages, matching the MAC size specified in the Signal Double Ratchet documentation.

To integrate this into the full Double Ratchet protocol, Alice and Bob perform the following steps to transmit one anamorphic message amsg along with one normal Signal message m .

1. Alice encrypts a normal Signal message m into $\text{ct} = \text{AES}_{\text{CBC}}(\text{aes}_{\text{key}}, \text{IV}, m)$ and computes a tag as described in Algorithm 10 instead of the original mac .
2. Bob detects and recovers the anamorphic message amsg by running aDec as shown in Algorithm 11.

We now argue for the correctness and security of the construction. Correctness requires that, given the MAC key mac_{key} , the hidden message can always be recovered; this follows directly from the invertibility of the XOR operation.

Anamorphic security requires that MACs carrying hidden messages be indistinguishable from honestly generated MACs. This holds as long as mac_{key} remains secret and the underlying MAC produces pseudorandom tags, since XOR with a fixed message preserves their indistinguishability from random. As illustrated in Figure 5.1, secrecy of mac_{key} implies secrecy of the corresponding message key and, in turn, of the chain key. Because deriving the chain key requires both the DH output and the root key, the construction remains secure even if either the private ratchet keys or the root key is compromised, as long as they are not both leaked simultaneously.

Anamorphic Security. The construction achieves anamorphic security if for every PPT adversary $\mathcal{D}$ it holds that

$$|\Pr[\text{GReal}_{\text{MAC}}^{\mathcal{D}}(\lambda) = 1] - \Pr[\text{GAnam}_{\text{aMAC}}^{\mathcal{D}}(\lambda) = 1]| \leq \text{negl}(\lambda)$$

where the security games are defined as follows.

In the security games, the distinguisher is given either rk or dh_{out} according to the security model. We explicitly show the intermediate steps to generate aes_{key} , mac_{key} and IV as they depend on the previous steps and ensure that in both games they are from the same distribution. We show that without the full knowledge of rk and dh_{out} , the keys generated along the KDF key are computationally indistinguishable from the random values sampled from the same distribution. The hybrid arguments then show that the anamorphic game is

indistinguishable from one where all keys are uniformly sampled and MACs are generated with random MAC keys, which is finally shown to be indistinguishable from the real game.

GReal^D_{MAC}(λ) :

1. $rk \leftarrow \{0,1\}^\lambda$
2. $(sk_A, pk_A) \leftarrow \text{Keygen}(\lambda)$
3. $(sk_B, pk_B) \leftarrow \text{Keygen}(\lambda)$
4. $dh_{\text{out}} \leftarrow \text{DH}(sk_A, pk_B)$
5. **return** $\mathcal{D}^{\mathcal{O}_{\text{MAC}}(\cdot, \cdot, \cdot)}(\lambda, \text{info})$ where

$\mathcal{O}_{\text{MAC}}(\text{amsg}, \text{ad}, m)$:

$(rk_{\text{new}}, ck) \leftarrow \text{KDF}_{rk}(rk, dh_{\text{out}})$
 $(ck_{\text{new}}, mk) \leftarrow \text{KDF}_{ck}(ck)$
 $(aes_{\text{key}}, mac_{\text{key}}, IV) \leftarrow \text{KDF}_{mk}(mk)$
 $ct \leftarrow \text{AES}_{\text{CBC}}(aes_{\text{key}}, IV, m)$
return $aes_{\text{key}}, IV,$
 $\text{MAC}(mac_{\text{key}}, ct, ad)$

GAnam^D_{aMAC}(λ) :

1. $rk \leftarrow \{0,1\}^\lambda$
2. $(sk_A, pk_A) \leftarrow \text{Keygen}(\lambda)$
3. $(sk_B, pk_B) \leftarrow \text{Keygen}(\lambda)$
4. $dh_{\text{out}} \leftarrow \text{DH}(sk_A, pk_B)$
5. **return** $\mathcal{D}^{\mathcal{O}_{\text{aMAC}}(\cdot, \cdot, \cdot)}(\lambda, \text{info})$ where

$\mathcal{O}_{\text{aMAC}}(\text{amsg}, \text{ad}, m)$:

$(rk_{\text{new}}, ck) \leftarrow \text{KDF}_{rk}(rk, dh_{\text{out}})$
 $(ck_{\text{new}}, mk) \leftarrow \text{KDF}_{ck}(ck)$
 $(aes_{\text{key}}, mac_{\text{key}}, IV) \leftarrow \text{KDF}_{mk}(mk)$
 $ct \leftarrow \text{AES}_{\text{CBC}}(aes_{\text{key}}, IV, m)$
return $aes_{\text{key}}, IV,$
 $\text{fakeMAC}(mac_{\text{key}}, ct, ad, \text{amsg})$

In both games, info = dh_{out} or rk

Theorem 2. *The construction Sym-MAC defined in Section 5.5.1 satisfies anamorphic security if the Decisional Diffie–Hellman problem is hard in $\mathbb{G}$, MAC used in Algorithm 10 is modelled as a secure PRF and all KDFs involved are secure KDFs, as introduced in Section 5.2.1.*

Proof. In the following sequence of hybrid games let $\mathcal{D}$ be a PPT adversary and Q its total number of oracle queries.

Game 0: This is the anamorphic game $\text{GAnam}_{\text{aMAC}}^{\mathcal{D}}(\lambda)$ in the security definition, where the oracle generates anamorphic MACs as described in Algorithm 10, returning $t = \text{MAC}(\text{mac}_{\text{key}}, \text{ct}, \text{ad}) \oplus \text{amsg}$.

Game 1: In this game, we replace the use of KDF_{rk} , KDF_{ck} and KDF_{mk} to generate ck , mk , aes_{key} , mac_{key} and IV with elements chosen uniformly at random.

Game 2: In this game, we replace the application of MAC in Algorithm 10 by the application of a uniformly random function $\mathcal{F} : \{0,1\}^* \rightarrow \{0,1\}^\lambda$.

Game 3: In this game, we define $\mathcal{F}$ iteratively by lazy sampling: each time $\mathcal{F}$ should be applied, we check if we already established a mapping for the given input a , i.e., $a \mapsto z$ is contained in some set $\mathcal{M}_{\mathcal{F}}$, and if this is the case z is the output. Otherwise, some fresh $z \leftarrow \{0,1\}^\lambda$ is sampled uniformly and the mapping $a \mapsto z$ is added to $\mathcal{M}_{\mathcal{F}}$.

Game $(4, j)$ for $j \in [0, Q]$: In this series of games we slightly modify Game 3 until we end up with Game 4, by modifying the way each of the Q applications of $\mathcal{F}$ are computed. We define Game $(4, 0)$ to be identical to Game 3. In Game $(4, j)$, the challenger handles the first j applications of $\mathcal{F}$ as follows: Instead of checking if the a mapping for the corresponding input is already defined in $\mathcal{M}_{\mathcal{F}}$, it just samples a uniform output and adds the corresponding mapping to $\mathcal{M}_{\mathcal{F}}$. Note that in Game $(4, Q)$ all applications of $\mathcal{F}$ are replaced by uniformly sampling elements from $\{0,1\}^\lambda$.

Game 5: In this game, we replace the XOR computation with a uniformly random element independent of amsg .

Game 6: In this game, the oracle applies a freshly sampled MAC key to ct and ad .

Game 7: In this game, we replace the uniformly random chosen aes_{key} , mac_{key} and IV by generating them using the KDFs again. This reverses the changes made in Game 1.

Game 8: This is the real game $\text{GReal}_{\text{MAC}}^{\mathcal{D}}(\lambda)$ in the security definition, where the relevant keys are generated by KDFs and the oracle applies a freshly sampled MAC key to ct and ad.

Game 0 and Game 1 are computationally indistinguishable, by Lemma 1. Game 1 and Game 2 are computationally indistinguishable, as a distinguisher for them could be turned in to one for pseudorandom property of PRF. The games Game 2 and Game 3 are perfectly indistinguishable as Game 3 just uses an equivalent way to define $\mathcal{F}$. Game (4,0) coincides with Game 3 by definition. The difference between games Game (4, $j-1$) and Game (4, j) consists in how the output of the j -th application of $\mathcal{F}$ is computed. In Game (4, $j-1$) we check whether the input $\text{mac}_{\text{key}_{j-1}}$ occurred before in $\mathcal{M}_{\mathcal{F}}$, while we do not perform this check in Game (4, j). Observing that all $\text{mac}_{\text{key}_i}$ for $i \leq j-1$ have been chosen uniformly at random from $\mathcal{K}$, the probability that $\text{mac}_{\text{key}_i} \mapsto z$, for some z , occurs in $\mathcal{M}_{\mathcal{F}}$ is upper bounded by $\frac{i-1}{2^\lambda}$. Thus, in total the probability that games Game (4,0) and Game (4, Q) differ is upper bounded by $\frac{Q^2}{2^\lambda}$, which is negligible. Since XOR with any value is a bijection on $\{0,1\}^\lambda$, the joint distribution of all oracle outputs is identical in Game (4, Q) and Game 5, even if some inputs repeat, because lazy sampling preserves consistency. Thus Game 4, Q and Game 5 are perfectly indistinguishable. Finally, note that Game (4, Q) and Game 6 are identical. The computational indistinguishability from Game 6 to Game 7 is again from Lemma 1, and Game 7 and Game 8 are identical. $\square$

Lemma 1. *Without the full knowledge of the root key and the DH output, a PPT adversary $\mathcal{D}$ could not distinguish whether aes_{key} , mac_{key} and IV are generated through a KDF chain or sampled uniformly at random.*

We defer the proof for this lemma to the Appendix 5.8.

5.5.2 Stronger Security Guarantee – IND-CCA

Similar to our previous construction, here, we also require that security holds even against an adversary that obtains ciphertexts generated in anamorphic mode and actively interacts with the protocol.

Intuitively, Alice embeds ℓ anamorphic messages, each of length λ , within $\ell + 1$ normal Signal messages. The $\ell + 1$ MAC tags are generated in a way that the first ℓ tags used in the symmetric ratchet to carry the corresponding covert payloads, and the final tag authen-

		Signal-Android				libSignal		
Construction	Anamorphic length	Decryption		Round-trip		Encryption	Decryption	
		Avg	Worst	Avg	Worst	Avg	Avg	Worst
Android real device								
Original	None	1.56	—	20.5	—	1.53	1.42	—
Base (A)Sym-DH	16-bit	822.8	1622.2	841.48	1673.28	1.71	811.34	1617.41
Sym-MAC	64-bit	1.57	—	20.6	—	1.53	1.40	—
Android emulator								
Original	None	1.50	—	120.10	—	1.11	1.11	—
Base (A)Sym-DH	16-bit	438.2	888.2	567.3	1010.2	1.81	440.3	883.2
Optimized	16-bit	7.01	14.12	149.96	150.14	1.13	5.77	13.20
(A)Sym-DH	32-bit	7.22	65.10	150.32	200.18	1.32	5.43	65.88
Sym-MAC	64-bit	1.51	—	120.15	—	1.11	1.12	—

Table 5.3: Performance (in milliseconds) on Android real device (top) and Android emulator (bottom).

ticates the preceding transcript to provide an integrity check. We defer the construction and security proof to Appendix 5.8.

5.6 Performance

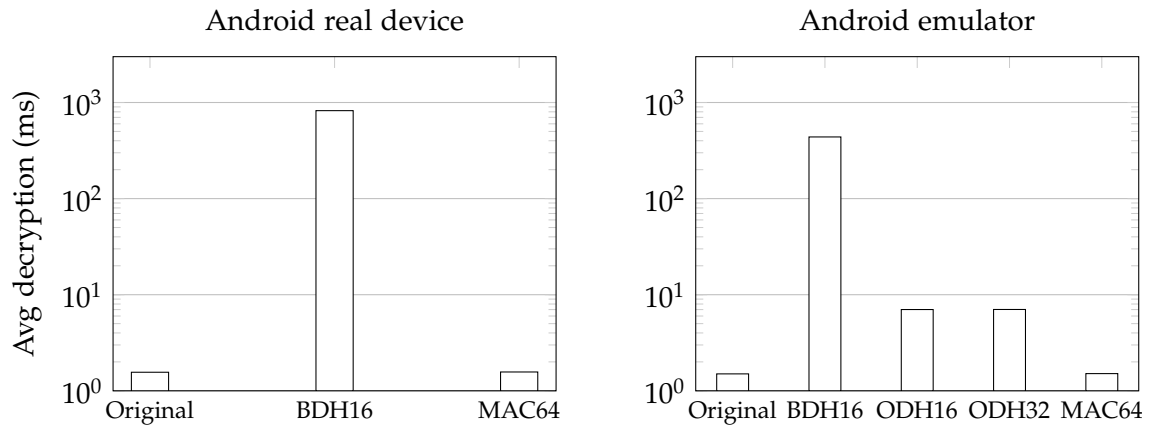


Figure 5.4: Average decryption latency in the Android emulator and Android real device measured in Signal Android library, grouped by construction and anamorphic bit-length (log scale). BDH16, MAC64, ODH16, and ODH32 stand for Base (A)Sym-DH 16-bit, Sym-MAC 64-bit, Optimized (A)Sym-DH 16-bit, and Optimized (A)Sym-DH 32-bit, respectively.

We implement prototypes of anamorphic messaging constructions [109] in the official Signal application [110] using Rust, Java, and Kotlin. We empirically evaluate the impact of these implementations across different covert bit-rate settings on the overall performance

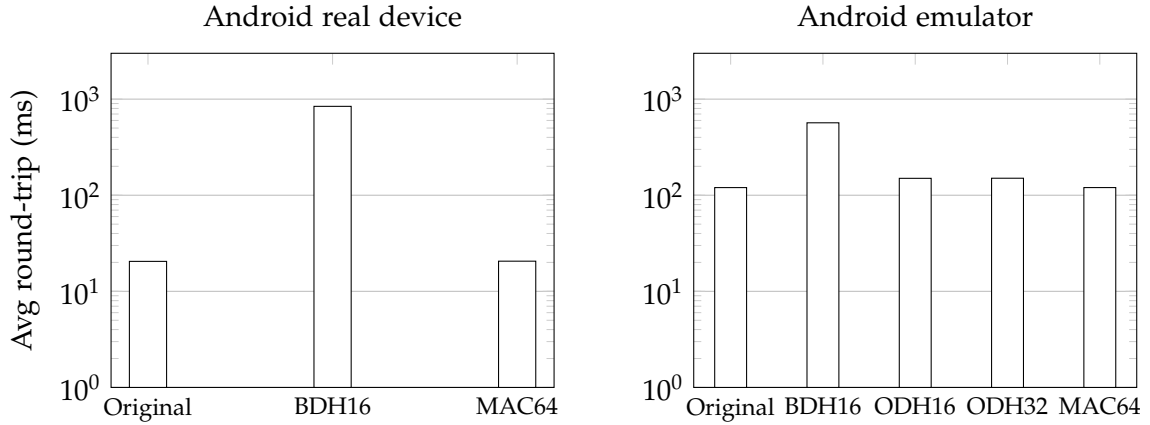


Figure 5.5: Average round-trip latency in the Android emulator and Android real device measured in Signal Android library, grouped by construction and anamorphic bit-length (log scale).

of the Signal application, both in the official Signal-Android client [111] and libSignal [112], the platform-agnostic library used by the official Signal clients which is mainly written in Rust.

We run our experiments on an Android device running Android 14 with a Qualcomm Snapdragon 888 processor and 8 GB of RAM, and on Android Studio emulators hosted in a 2023 MacBook Pro laptop on an M2 Pro processor with 16GB of RAM.

Tables 5.3 presents performance results for experiments conducted on a real Android device and an Android emulator. We report the average encryption and decryption times for random messages using each anamorphic messaging construction, alongside the original Signal protocol.

We also evaluate the average worst-case decryption time, which occurs when a message does *not* contain an anamorphic payload. In addition, we measure the round-trip time on the recipient side, defined as the duration between receiving the message from the server and sending the corresponding delivery receipt back. This metric is important for assessing potential side-channel attack vectors, as the time taken to generate and send the receipt could reveal to the server whether the message was anamorphic.³

As decryption is the most computationally intensive part of our constructions and a key performance indicator, we evaluate the decryption time in both Signal-Android and

³One could avoid sending the "delivery" message at all. In such a case, the DH ratchet would not be updated (since the delivery message carries a fresh public key) and the sender would need to wait for a message reply from the receiver before sending a new anamorphic message in the Sym-DH and Asym-DH constructions. In addition, not sending a delivery message might appear suspicious to the server, as sending a delivery message is standard in Signal.

libsignal to gain a comprehensive view of performance. To compute the function `aDec` in our constructions (Sym-DH, Asym-DH, and Sym-MAC), we implemented the base variants. Moreover, we implemented the optimized variants of the construction (A)Sym-DH on the Android emulator to show their huge performance advantage. Note that the PRFs in Sym-DH and Asym-DH are instantiated using HMAC-SHA256.

In general, message encryption times are not affected by the generation of anamorphic messages. Looking at libsignal on the real device, the original Signal protocol takes approximately 1.53 ms to encrypt a message, while all anamorphic constructions require nearly the same time to encrypt an anamorphic message, regardless of its size. Similar results can be observed on the Android emulator, as shown in Table 5.3.

The decryption time is affected by the type of construction. This is expected, as the decryption process might involve iterative operations, particularly the discrete logarithm computation required to recover the anamorphic message in Sym-DH and Asym-DH constructions. However, there is no significant difference, on average, within the optimized (A)Sym-DH construction, nor between the decryption in the core libSignal library and the Signal-Android client.

The round-trip time, as measured in the Signal-Android code, varies considerably only in the base (A)Sym-DH construction, while it does not deviate from the original Signal implementation in either the Sym-MAC or the optimized (A)Sym-DH constructions. Therefore, the Sym-MAC construction is the most efficient according to all indicators. However, it supports the weakest threat model. The optimized (A)Sym-DH construction supports the strongest threat model and performs within the same order of magnitude as the original Signal construction. The base (A)Sym-DH construction takes significantly longer than the original Signal protocol, as the receiver must decrypt up to 16 bits of anamorphic information, involving multiple scalar multiplications, as shown in Figures 5.4 and 5.5. This construction is therefore suitable only when a lookup table cannot be maintained. In this case, to mitigate side-channel risks, the receiver can send the delivery receipt after a fixed delay matching the average round-trip time in the original implementation, rather than waiting for successful decryption. However, the sender should transmit each message using an interval longer than the worst-case round-trip time—approximately 1600 ms on the real device and 1000 ms on the emulator.

5.7 Related Work

The foundations of modern secure messaging go back to Off-the-Record Messaging (OTR) [113], which introduced forward secrecy and deniability. Later, TextSecure [114] extended these ideas with a structured cryptographic framework that evolved into the Signal protocol [94], now considered the standard for private messaging. At the heart of Signal is the Double Ratchet algorithm, which has been formally studied in multiple works. Alwen et al. [99] presented a modular analysis of its components; Poettering and Rösler [115] investigated its compromise-resilience; and Cohn-Gordon et al. [98] provided symbolic and computational proofs of its E2EE guarantees. Bienstock et al. [97] proposed a more complete Universally Composable framework, introducing a minimal extension called the Triple Ratchet, which captures stronger adversarial models with no added cost.

More recently, Teng and Rasmussen [116] showed that key verification in Signal can be automated within a modified protocol, removing the need for out-of-band user checks such as safety-number comparison or QR scanning and strengthening resistance to man-in-the-middle attacks. Martiny et al. [117] analyzed the Sealed Sender mechanism in Signal and demonstrated how statistical disclosure attacks could compromise its anonymity guarantees.

In parallel, the Signal protocol has been extended to resist emerging threats such as quantum adversaries. PQXDH [118] replaces the X3DH handshake with a post-quantum alternative, while Dodis et al. [119] introduced a Triple Ratchet protocol that incorporates a more efficient PQ-ratchet and erasure coding, addressing bandwidth limitations in hybrid-secure settings. In a related development within the quantum-secure Signal, the ML-KEM Braid protocol [120] was proposed, based on the ideas of [121]. It is a Sparse Continuous Key Agreement (SCKA) built on ML-KEM that lets two parties derive a sequence of post-quantum shared secrets with forward secrecy and post-compromise security. The Classical Continuous Key Agreement in the Double Ratchet emits a fresh secret each round trip, whereas SCKA handles bandwidth constraints by sending large KEM data in pieces so new secrets may arrive over multiple round trips, and the resulting keys can feed a Double Ratchet for secure messaging. Additionally, efforts have been made to enhance secure messaging protocols like Signal's X3DH and Double Ratchet with subversion resilience, employing techniques such as *Reverse Firewalls*, which protect against attacks that modify cryptographic algorithms, ensuring that users can detect tampering even with compromised implementations. [122].

The robustness of secure messaging systems, particularly Signal, lies in its ability to guarantee security and privacy even in adversarial environments. However, as the landscape of cryptographic security evolves, new threats such as deployment-level compromises emerge, challenging the reliability of E2EE. In light of these challenges, anamorphic encryption offers a promising approach to preserve security in the presence of surveillance that compromises E2EE.

Anamorphic encryption was first introduced by Persiano et al. [33] as a method for enabling covert communication when a surveillance authority can decrypt messages, but a co-operating sender and receiver can still exchange hidden content embedded inside normal-looking ciphertexts. Since its introduction, several works have proposed stronger and more robust definitions of anamorphic encryption, adapting it to current cryptographic primitives [35, 36, 123, 34, 124].

One of the early developments was by Wang et al. [124], who introduced the notion of ℓ -sender-anamorphic encryption, formalizing secure constructions for pseudo-random public key encryption (PKE) schemes. Following this work, Banerjee et al. [125] introduced a framework for public-key anamorphic μ -message extensions, embedding covert messages within multiple ciphertexts of CPA-secure PKE schemes, without relying on randomness recovery.

Another research domain that has a goal on the opposite side of anamorphic encryption is Anamorphic-Resistant Encryption (ARE). Dodis and Goldin [126] and Carnemolla et al. [127] propose encryption schemes that specifically block covert communication, opposing the core goal of anamorphic encryption. ARE schemes aim to make covert channels either slow or entirely impractical, thereby undermining anamorphic encryption's ability to enable hidden communication in the presence of surveillance that breaks E2EE. Avitabile et al. [128] present a practical Anamorphic-Resistant Encryption (ARE) scheme that addresses key limitations of previous constructions [126, 127], notably by eliminating the reliance on random oracles and indistinguishability obfuscation. Their scheme, based on Extremely Lossy Functions (ELFs) and Unique Non-Interactive Zero-Knowledge proofs (UNIZKs), ensures that covert communication is limited to a logarithmic number of bits, effectively hindering the viability of anamorphic encryption for transmitting long anamorphic messages.

In recent years, research on anamorphism in application-level cryptographic protocols has gained considerable momentum. Giustolisi et al. [129] adapted the ElGamal-based

construction from Wang et al. [124] to electronic voting systems, enabling voters to embed hidden preferences within ballots that are indistinguishable from legitimate ones, even against malicious authorities.

More recently, Gligoroski et al. [130] explored the application of anamorphic encryption in conversational settings, embedding covert messages within text exchanged over public chat channels. Their framework introduced an LLM-based embedding function, enabling both symmetric and public key encryption schemes (e.g., password-authenticated encryption and elliptic curve DH key exchange with ephemeral keys) to operate over natural language outputs. This work opened new directions for covert communication through LLM-generated text.

Despite its theoretical foundations, anamorphic encryption has seen limited exploration in practical applications. Secure messaging systems like Signal are often targeted by adversaries or regulatory efforts seeking to undermine E2EE. Legislative proposals for lawful access to encrypted data [102] highlight the need for secure messaging protocols that remain functional under E2EE compromise. Our work builds on the principles of anamorphic encryption, applying it to the Signal Double Ratchet protocol to enable covert communication even if the confidentiality of E2EE is intentionally broken.

5.8 Conclusions

In this work, we presented two constructions on anamorphic channels for the Signal Double Ratchet protocol: a DH-based channel that embeds payloads in DH key pairs and a MAC-based channel that embeds payloads in authentication tags of the symmetric ratchet. Our designs remain compatible with existing Signal-style deployments, require no server changes, and operate even when one or more Double Ratchet keys are disclosed (for example as a result of a surveillance mechanism). Our analysis and implementations show that the DH-based and MAC-based channels can efficiently carry at least 32 and 64 hidden bits per Signal message, respectively. To the best of our knowledge, this is the first practical instantiation of anamorphic encryption on a deployed system, the Signal Double Ratchet, that has impact on billions of people.

Several policymakers and governments have pushed Signal and other E2EE messaging platforms to enable surveillance by breaking E2EE confidentiality to snoop on messages. Our results indicate that even if decryption keys are exposed, covert channels that preserve

normal protocol behavior can still exist.

A key direction for future work concerns the potential deployment of Anamorphic-Resistant Encryption schemes. If ARE systems become efficient and widely used, it will be important to understand whether some form of anamorphism can still be maintained, and under what leakage or functionality constraints. In particular, studying ARE-style defenses in the context of the Signal protocol would clarify the minimal conditions under which covert channels remain possible.

Finally, our approach is worth investigating in protocols that reuse Double Ratchet-style building blocks, including recent post-quantum proposals such as Triple Ratchet [119] and Apple’s PQ3 [131]. Beyond messaging, similar embeddings may apply to other E2EE services, including file sharing and real-time audio and video, offering a clearer path to supporting E2EE even under key-disclosure pressure.

Appendix

IND-CCA Security for Covert Messaging in DH Ratchet

We update the fake key generation algorithm (Algorithm 6) and the anamorphic message decryption algorithm (Algorithm 7) to Algorithm 12 and Algorithm 13, respectively. We add an auxiliary PRF* (same key space) that maps the transcript to the secret key, binding keys to the transcript.

Algorithm 12 $\text{fakeKeygen}(\text{dk}, \overrightarrow{\text{amsg}})$

```

1: parse  $\overrightarrow{\text{amsg}} = \{\text{amsg}_i\}_{i=1}^\ell$ 
2:  $(\text{sk}_1, \text{pk}_1) \leftarrow \text{Keygen}(\lambda)$ 
3: for  $i \in [2, \ell + 1]$  do
4:    $\text{sk}_i = \text{PRF}(\text{dk}, \text{pk}_{i-1}) + \text{amsg}_{i-1} \bmod p$ 
5:    $\text{pk}_i = \text{sk}_i \cdot G$ 
6: end for
7:  $T = \text{pk}_1 \parallel \text{pk}_2 \parallel \dots \parallel \text{pk}_{\ell+1}$ 
8:  $\text{sk}_{\ell+2} = \text{PRF}^*(\text{dk}, T)$ 
9:  $\text{pk}_{\ell+2} = \text{sk}_{\ell+2} \cdot G$ 
10: return  $\{(\text{sk}_i, \text{pk}_i)\}_{i=1}^{\ell+2}$ 

```

Definition 5.1 (IND-CCA for Covert Channel in DH Ratchet). *Let $\mathcal{A}$ be a PPT algorithm. The security experiment for indistinguishability under adaptive chosen ciphertext attack is $\text{Exp}_{\text{DH}}^{\text{ind-cca}}$*

Algorithm 13 aDec(dk, PK)

```

1: parse PK = (pki)i∈[1,ℓ+2]
2: T = pk1 || pk2 || ... || pkℓ+1
3: if pkℓ+2 ≠ PRF*(dk, T) · G then
4:   return ⊥
5: end if
6: for i ∈ [2, ℓ + 1] do
7:   amsgi-1 = dlogG(pki - PRF(dk, pki-1) · G)
8: end for
9: return  $\overrightarrow{\text{amsg}}$ 

```

shown in Figure 5.6. The advantage of $\mathcal{A}$ in the experiment is

$$\text{Adv}_{\text{DH}}^{\text{ind-cca}}(\mathcal{A}) = \left| \Pr \left[\text{Exp}_{\text{DH}}^{\text{ind-cca}}(\mathcal{A}) \rightarrow 1 \right] - \frac{1}{2} \right|$$

$\text{Exp}_{\text{DH}}^{\text{ind-cca}}(\mathcal{A})$
<pre> 1: dk ←\$ {0, 1}^λ 2: S = ∅ 3: b ←\$ {0, 1} 4: b' ← $\mathcal{A}^{\mathcal{O}_{\text{fKGen}_b}(\cdot, \cdot), \mathcal{O}_{\text{aDec-DH}}(\cdot)}(1^\lambda, 1^\ell)$ 5: if b' = b then 6: return 1 7: else 8: return 0 </pre>
$\mathcal{O}_{\text{fKGen}_b}(\overrightarrow{\text{amsg}}_0, \overrightarrow{\text{amsg}}_1)$
<pre> 1: C ← fakeKeygen(dk, $\overrightarrow{\text{amsg}}_b$) 2: S = S ∪ {C} 3: return C </pre>
$\mathcal{O}_{\text{aDec-DH}}(C^*)$
<pre> 1: if C* ∈ S 2: return ⊥ 3: else 4: parse C = {(pk_i, sk_i)}_{i=1}^{ℓ+2} 5: return aDec(dk, {pk_i}_{i=1}^{ℓ+2}) </pre>

Figure 5.6: Security experiment for IND-CCA for Covert Channel in DH Ratchet

Theorem 3. *If PRFs used in Appendix 5.8 are secure PRFs, for every PPT adversary $\mathcal{A}$, it holds that $\text{Adv}_{\text{DH}}^{\text{ind-cca}}(\mathcal{A}) \leq \text{negl}(\lambda)$.*

Proof. Let $\Pr[\text{Game } i]$ be the adversary's success probability in game Game i , Q be the total number of oracle queries.

Game 0: This game is exactly the IND-CCA experiment and oracle. Hence,

$$\text{Adv}_{\text{DH}}^{\text{ind-cca}}(\mathcal{A}) = \left| \Pr[\text{Game 0}] - \frac{1}{2} \right|$$

Game 1: In this game, we replace PRF^* in the decryption algorithm aDec with a random function $\mathcal{F} : \{0, 1\}^* \mapsto \mathbb{Z}_p$.

Let $\mathcal{B}_1$ be a PPT adversary against PRF security. By the security of PRF, we have

$$|\Pr[\text{Game 0}] - \Pr[\text{Game 1}]| \leq \text{Adv}^{\text{prf}}(\mathcal{B}_1)$$

Game 2: In this game, we define $\mathcal{F}$ iteratively by lazy sampling: we maintain a table for $\mathcal{F}$, and on input T : if T is already in the table, return the stored $\text{pk}_{\ell+2}$, otherwise sample $\text{pk}_{\ell+2} \leftarrow \$ \mathbb{Z}_p$ uniformly, store and return it. Game 1 and Game 2 use equivalent methods to define $\mathcal{F}$, and thus

$$\Pr[\text{Game 1}] = \Pr[\text{Game 2}]$$

Game 3: We first define a Forge event: the aDec algorithm accepts a *fresh* ciphertext. In Game 2, for any fresh T^* , $\text{pk}_{\ell+2} = \text{PRF}^*(\text{dk}, T^*)$ is uniform and hidden. Acceptance requires guessing $\text{pk}_{\ell+2} = \text{sk}_{\ell+2} \cdot G$. In a group of prime order p , this succeeds with probability $\frac{1}{p}$ per query, and thus

$$\Pr[\text{Forge}] \leq \frac{Q}{p}.$$

Now in this game, it is identical to Game 2 except that on the first accepting fresh decryption query, the decryption oracle directly aborts. Then,

$$|\Pr[\text{Game 2}] - \Pr[\text{Game 3}]| \leq \Pr[\text{Forge}]$$

Game 4: In this game, we replace $\{(\text{pk}_i)\}_{i=1}^{\ell+1}$ with fresh DH pairs.

Let $\mathcal{B}_2$ be a PPT adversary against anamorphic security we defined before, then we have

$$|\Pr[\text{Game 3}] - \Pr[\text{Game 4}]| \leq \text{Adv}_{\text{DH}}^{\text{ana}}(\mathcal{B}_2).$$

Game 5: In the final game, all the $\{(\text{pk}_i)\}_{i=1}^{\ell+2}$ has been replaced by random values, and the decryption oracle gives no information. This is identical to the previous Game 4 and the adversary's entire view does not depend on the hidden bit b . Therefore, we have

$$\Pr[\text{Game 5}] = \Pr[\text{Game 4}] = \frac{1}{2}.$$

Combining the hops, we derive

$$\text{Adv}_{\text{DH}}^{\text{ind-cca}}(\mathcal{A}) \leq \text{Adv}^{\text{prf}}(\mathcal{B}_1) + \text{Adv}_{\text{DH}}^{\text{ana}}(\mathcal{B}_2) + \frac{Q}{p}$$

which is negligible. □

Proof for Lemma 1

Proof. To prove this, we define the following two games and show that for every PPT adversary $\mathcal{D}$ it holds that

$$|\Pr[\text{GReal}_{\text{key}}^{\mathcal{D}}(\lambda) = 1] - \Pr[\text{GRand}_{\text{rkey}}^{\mathcal{D}}(\lambda) = 1]| \leq \text{negl}(\lambda).$$

For simplicity, in the experiments, we show the KDF chain of only one KDF_{rk} block, one KDF_{ck} block and one KDF_{mk} block. When the KDF_{rk} block is fixed, the longer KDF chain leads to the more iterations of the games involving KDF_{ck} and KDF_{mk} where uniformly random ck and mk are used as inputs respectively.

```

GRealkey $\mathcal{D}$ ( $\lambda$ ) :
1.  $rk \leftarrow \{0,1\}^\lambda$ 
2.  $(sk_A, pk_A) \leftarrow \text{Keygen}(\lambda)$ 
3.  $(sk_B, pk_B) \leftarrow \text{Keygen}(\lambda)$ 
4.  $dh_{\text{out}} \leftarrow \text{DH}(sk_A, pk_B)$ 
5. return  $\mathcal{D}^{\mathcal{O}_{\text{key}}(\cdot)}(\lambda, \text{info})$  where
    $\mathcal{O}_{\text{key}}(\hat{\text{info}})$ :
        $\text{prf}_{\text{key}} \leftarrow \text{HMAC}(\text{info}, \hat{\text{info}})$ 
        $(rk_{\text{new}}, ck) \leftarrow \text{HKDF}_{\text{expand}}(\text{prf}_{\text{key}})$ 
        $(ck_{\text{new}}, mk) \leftarrow \text{KDF}_{ck}(ck)$ 
       return  $\text{KDF}_{mk}(mk)$ 

GAndrkey $\mathcal{D}$ ( $\lambda$ ) :
1.  $rk \leftarrow \{0,1\}^\lambda$ 
2.  $(sk_A, pk_A) \leftarrow \text{Keygen}(\lambda)$ 
3.  $(sk_B, pk_B) \leftarrow \text{Keygen}(\lambda)$ 
4.  $dh_{\text{out}} \leftarrow \text{DH}(sk_A, pk_B)$ 
5. return  $\mathcal{D}^{\mathcal{O}_{\text{key}}(\cdot)}(\lambda, \text{info})$  where
    $\mathcal{O}_{rkey}(\hat{\text{info}}) = (\text{aes}_{\text{key}}, \text{mac}_{\text{key}}, IV) \leftarrow \{0,1\}^{3\lambda}$ 

```

In the games, info is either rk or dh_{out} and $\hat{\text{info}}$ is the other. We explicitly detail the extract-then-expand technique used by the HKDF KDF_{rk} in the games, which consists of two steps:

1. The extract function takes as input the salt rk and the input key material dh_{out} , producing a PRF key. This extract function is instantiated using HMAC with SHA-256.
2. The expand function takes the PRF key, auxiliary information, and the desired output length as input, and outputs the new root key and chain key. We omit the auxiliary information and length here, which will be instantiated by PRF.

In the following sequence of hybrid games let $\mathcal{D}$ be a PPT adversary and Q its maximum number of queries to each oracle.

Game 0: This is the real game $\text{GReal}_{\text{key}}^{\mathcal{D}}(\lambda)$ in the definition, where the oracle generates all the keys as in the symmetric ratchet.

Game 1 if $\hat{\text{info}} = \text{dh}_{\text{out}}$: In this game, we replace $\hat{\text{info}}$ with a uniformly random element u from $\mathbb{G}$.

Game 2: In this game, we replace HMAC by the application of a uniformly random function $\mathcal{F}_0 : \mathbb{G} \mapsto \{0, 1\}^\lambda$ if $\hat{\text{info}} := \text{dh}_{\text{out}}$, otherwise $\mathcal{F}_0 : \{0, 1\}^\lambda \mapsto \{0, 1\}^\lambda$.

Game 3: In this game, we define $\mathcal{F}_0$ by lazy sampling: each time $\mathcal{F}_0$ should be applied, we check if we already established a mapping for the given input a , i.e., $a \mapsto z$ is contained in some set $\mathcal{M}_{\mathcal{F}_0}$, and if this is the case z is the output. Otherwise, some fresh $z \leftarrow \{0, 1\}^\lambda$ is sampled uniformly and the mapping $a \mapsto z$ is added to $\mathcal{M}_{\mathcal{F}_0}$.

Game $(4, j)$ for $j \in [0, Q]$: In this series of games we slightly modify Game 3 until we end up with Game 5, by modifying the way each of the Q applications of $\mathcal{F}_0$ are computed. We define Game $(4, 0)$ to be identical to Game 3. In Game $(4, j)$, the challenger handles the first j applications of $\mathcal{F}$ as follows: Instead of checking if the mapping for the corresponding input is already defined in $\mathcal{M}_{\mathcal{F}}$, it just samples a uniform output and adds the corresponding mapping to $\mathcal{M}_{\mathcal{F}_0}$. Note that in Game $(4, Q)$ all applications of $\mathcal{F}_0$ are replaced by uniformly sampling elements from $\{0, 1\}^\lambda$.

Game 5: In this game, prf_{key} is sampled uniformly at random.

Game 6: In this game, we replace the expand algorithm by the application of a uniformly random function $\mathcal{F}_1 : \{0, 1\}^\lambda \mapsto \{0, 1\}^{2\lambda}$.

Game 7: In this game, we define $\mathcal{F}_1$ by lazy sampling, in the same manner as Game 3.

Game $(8, j)$ for $j \in [0, Q]$: In this series of games we slightly modify Game 6 until we end up with Game 9, by modifying the way each of the Q applications of $\mathcal{F}_0$ is computed. These steps are in the same manner as in Game 8 and in Game $(8, Q)$, all applications of $\mathcal{F}_1$ are replaced by uniformly random elements from $\{0, 1\}^{2\lambda}$.

Game 9: In this game, $(\text{rk}_{\text{new}}, \text{ck})$ is sampled uniformly at random.

Game 10: In this game, we replace KDF_{ck} by the application of a uniformly random function $\mathcal{F}_2 : \{0,1\}^\lambda \mapsto \{0,1\}^{2\lambda}$.

Game 11: In this game, we define $\mathcal{F}_2$ by lazy sampling, in the same manner as Game 3.

Game (12, j) : **for** $j \in [0, Q]$: In this series of games we slightly modify Game 11 until we end up with Game , by modifying the way each of the Q applications of $\mathcal{F}_2$ is computed. These steps are in the same manner as in Game 4 and in Game (12, Q), all applications of $\mathcal{F}_2$ are replaced by uniformly random elements from $\{0,1\}^{2\lambda}$.

Game 13: In this game, $(\text{ck}_{\text{new}}, \text{mk})$ is sampled uniformly at random.

Game 14: In this game, we replace KDF_{mk} by the application of a uniformly random function $\mathcal{F}_3 : \{0,1\}^\lambda \mapsto \{0,1\}^{3\lambda}$.

Game 15: In this game, we define $\mathcal{F}_3$ by lazy sampling, in the same manner as Game 3.

Game (16, j) for $j \in [0, Q]$: In this series of games we slightly modify Game 15 until we end up with Game , by modifying the way each of the Q applications of $\mathcal{F}_3$ is computed. These steps are in the same manner as in Game 4 and in Game (16, Q), all applications of $\mathcal{F}_3$ are replaced by uniformly random elements from $\{0,1\}^{3\lambda}$.

Game 17: In this game, $(\text{aes}_{\text{key}}, \text{mac}_{\text{key}}, \text{IV})$ is sampled uniformly at random, which is the ideal game $\text{GRand}_{\text{rkey}}^{\mathcal{D}}$.

Game 0 and Game 1 (if there is) are computationally indistinguishable as a distinguisher for them could be turned into one for solving DDH problem in $\mathbb{G}$. Game 0 and Game 2 are computationally indistinguishable, as a distinguisher for them could be turned into one for the PRF security of HMAC. Game 2 and Game 3 are perfectly indistinguishable as Game 3 just uses an equivalent way to define $\mathcal{F}_0$. Game (4, 0) coincides with Game 3 by definition. The difference between games Game (4, $j - 1$) and Game (4, j) consists in how the output of the j -th application of $\mathcal{F}_0$ is computed. In Game (4, $j - 1$) we check whether the input rk_{j-1} occurred before in $\mathcal{M}_{\mathcal{F}_0}$, while we do not perform this check in Game (4, j). Observing that all $\text{prf}_{\text{key}_i}$ for $i \leq j - 1$ have been chosen uniformly at random from $\{0,1\}^\lambda$, the probability that $\text{mac}_{\text{key}_i} \mapsto z$, for some z , occurs in $\mathcal{M}_{\mathcal{F}}$ is upper bounded by $\frac{j-1}{2^\lambda}$. Thus, in total the probability that games Game (4, 0) and Game (4, Q) differ is upper bounded by $\frac{Q^2}{2^\lambda}$, which is negligible. Finally, note that Game (4, Q) and Game 5 are identical.

Game 5 and Game 6 are computationally indistinguishable, as a distinguisher for them could be turned into one for the pseudorandomness of PRF. Then the distinguishability from Game 6 to Game 9 follows in the same way as that from Game 2 to Game 5, except that the total probability that games Game (8,0) and Game (8,Q) differ is upper bounded by $\frac{Q^2}{2^{2\lambda}}$, which is negligible as well.

Game 9 and Game 10 are computationally indistinguishable, as a distinguisher for them could be turned into one for the security of KDF. Then the distinguishability from Game 10 to Game 13 follows in the same way as that from Game 6 to Game 9, where the total probability that games Game (12,0) and Game (12,Q) differ is upper bounded by $\frac{Q^2}{2^{2\lambda}}$, which is negligible.

Game 13 and Game 14 are computationally indistinguishable, as a distinguisher for them could be turned into one for the security of KDF. Then the distinguishability from Game 14 to Game 17 follows in the same way as that from Game 10 to Game 13, except that the total probability that games Game (16,0) and Game (16,Q) differ is upper bounded by $\frac{Q^2}{2^{3\lambda}}$, which is negligible as well.

Finally, note that Game (16,Q) and Game 17 are identical. $\square$

Stronger Security Guarantee for Covert Messaging in Symmetric Ratchet

We update the fake MAC algorithm (Algorithm 10) and the anamorphic message decryption algorithm (Algorithm 11) to Algorithm 14 and Algorithm 15, respectively. We add an auxiliary PRF (same key space as MAC key) that maps the transcript to the tag.

Alice and Bob perform the following steps to transmit ℓ anamorphic messages $\overrightarrow{\text{amsg}}$ within $\ell + 1$ normal Signal messages $\overrightarrow{m}$.

Algorithm 14 $\text{fakeMAC}(\overrightarrow{\text{mac}}_{\text{key}}, \overrightarrow{\text{ct}}, \overrightarrow{\text{ad}}, \overrightarrow{\text{amsg}})$

```

1: for  $i \in [1, \ell]$  do
2:    $\text{mac}_i = \text{MAC}(\text{mac}_{\text{key}_i}, \text{ct}_i, \text{ad}_i) \oplus \text{amsg}_i$ 
3: end for
4:  $T = \text{mac}_1 \parallel \dots \parallel \text{mac}_\ell$ 
5:  $\text{mac}_{\ell+1} = \text{PRF}(\text{mac}_{\text{key}_{\ell+1}}, T)$ 
6: return  $\overrightarrow{\text{mac}}$ 

```

1. Alice encrypts $\ell + 1$ normal Signal messages $\overrightarrow{m}$ into $\overrightarrow{\text{ct}} = \text{AES}_{\text{CBC}}(\overrightarrow{\text{aes}}_{\text{key}}, \overrightarrow{\text{IV}}, \overrightarrow{m})$ where $\overrightarrow{\text{aes}}_{\text{key}}$ and $\overrightarrow{\text{IV}}$ are generated honestly along the KDF chain.

Algorithm 15 $\text{aDec}(\overrightarrow{\text{mac}}_{\text{key}}, \overrightarrow{\text{ct}}, \overrightarrow{\text{ad}}, \overrightarrow{\text{mac}})$

```

1:  $\overrightarrow{t} = \text{MAC}(\overrightarrow{\text{mac}}_{\text{key}}, \overrightarrow{\text{ct}}, \overrightarrow{\text{ad}})$ 
2: if  $\text{mac}_i = t_i$  for  $i \in [1, \ell + 1]$  then
3:   skip ▷ Not anamorphic
4: else
5:   for  $i \in [1, \ell]$  do
6:      $\text{amsg}_i = \text{mac}_i \oplus t_i$ 
7:   end for
8:    $T = t_1 \parallel \dots \parallel t_\ell$ 
9:    $\sigma = \text{PRF}(\text{mac}_{\text{key}_{\ell+1}}, T)$ 
10:  if  $\sigma = \text{mac}_{\ell+1}$  then
11:    return  $\overrightarrow{\text{amsg}}$ 
12:  else
13:    return  $\perp$ 
14:  end if
15: end if

```

2. Alice then computes the tags as described in Algorithm 14 and uses them as MACs corresponding to the messages instead of the original $\overrightarrow{\text{mac}}$.

3. Bob detects and recovers the anamorphic message amsg by running aDec as shown in Algorithm 15.

Definition 5.2 (IND-CCA for Covert Channel in Symmetric Ratchet). *Let $\mathcal{A}$ be a PPT algorithm. The security experiment for indistinguishability under adaptive chosen ciphertext attack is $\text{Exp}_{\text{MAC}}^{\text{ind-cca}}$ shown in Figure 5.7. The advantage of $\mathcal{A}$ in the experiment is*

$$\text{Adv}_{\text{MAC}}^{\text{ind-cca}}(\mathcal{A}) = \left| \Pr \left[\text{Exp}_{\text{MAC}}^{\text{ind-cca}}(\mathcal{A}) \rightarrow 1 \right] - \frac{1}{2} \right|$$

Theorem 4. *If the Decisional Diffie–Hellman problem is hard in $\mathbb{G}$, MAC used in Algorithm 14 is modelled as a secure PRF and all KDFs involved are secure KDFs, for every PPT adversary $\mathcal{A}$, it holds that $\text{Adv}_{\text{MAC}}^{\text{ind-cca}}(\mathcal{A}) \leq \text{negl}(\lambda)$.*

Proof. Let $\Pr[\text{Game } i]$ be the adversary’s success probability in game Game i , Q be the total number of oracle queries.

Game 0: This game is exactly the IND-CCA experiment and oracle. Hence,

$$\text{Adv}_{\text{MAC}}^{\text{ind-cca}}(\mathcal{A}) = \left| \Pr[\text{Game } 0] - \frac{1}{2} \right|$$

$\text{Exp}_{\text{MAC}}^{\text{ind-cca}}(\mathcal{A})$
1 : $\text{rk}_0 \leftarrow \{0,1\}^\lambda$ 2 : $(\text{sk}_A, \text{pk}_A) \leftarrow \text{Keygen}(\lambda)$ 3 : $(\text{sk}_B, \text{pk}_B) \leftarrow \text{Keygen}(\lambda)$ 4 : $\text{dh}_{\text{out}} \leftarrow \text{DH}(\text{sk}_A, \text{pk}_B)$ 5 : $\text{info} = \text{dh}_{\text{out}} \text{ or } \text{rk}$ 6 : $S = \emptyset$ 7 : $b \leftarrow \{0,1\}$ 8 : $b' \leftarrow \mathcal{A}^{\mathcal{O}_{\text{fMAC}_b}(\cdot, \cdot, \cdot), \mathcal{O}_{\text{aDec-MAC}}(\cdot)}(1^\lambda, 1^\ell, \text{info})$ 9 : if $b' = b$ then 10 : return 1 11 : else 12 : return 0 $\mathcal{O}_{\text{fMAC}_b}(\overrightarrow{\text{ams}}_0, \overrightarrow{\text{ams}}_1, \overrightarrow{m})$
1 : for $i \in [0, \ell - 1]$ do 2 : $(\text{rk}_{i+1}, \text{ck}_i) \leftarrow \text{KDF}_{\text{rk}}(\text{rk}_i, \text{dh}_{\text{out}})$ 3 : $(\text{ck}_{i+1}, \text{mk}_i) \leftarrow \text{KDF}_{\text{ck}}(\text{ck}_i)$ 4 : $(\text{aes}_{\text{key}_{i+1}}, \text{mac}_{\text{key}_{i+1}}, \text{IV}_{i+1}) \leftarrow \text{KDF}_{\text{mk}}(\text{mk}_i)$ 5 : $\text{ct}_{i+1} \leftarrow \text{AES}_{\text{CBC}}(\text{aes}_{\text{key}_{i+1}}, \text{IV}_{i+1}, m_{i+1})$ 6 : $C \leftarrow \text{fakeMAC}(\overrightarrow{\text{mac}}_{\text{key}}, \overrightarrow{\text{ct}}, \overrightarrow{\text{ad}}, \overrightarrow{\text{ams}}_b)$ 7 : $S = S \cup \{C\}$ 8 : return C $\mathcal{O}_{\text{aDec-MAC}}(C^*)$
1 : if $C^* \in S$ 2 : return $\perp$ 3 : else 4 : return $\text{aDec}(\overrightarrow{\text{mac}}_{\text{key}}, \overrightarrow{\text{ct}}, \overrightarrow{\text{ad}}, C^*)$

Figure 5.7: Security experiment for IND-CCA for Covert Channel in Symmetric Ratchet

Game 1: In this game, we replace PRF in the decryption algorithm aDec with a random function $\mathcal{F} : \{0,1\}^* \mapsto \{0,1\}^\lambda$.

Let $\mathcal{B}_1$ be a PPT adversary against PRF security. By the security of PRF, we have

$$|\Pr[\text{Game 0}] - \Pr[\text{Game 1}]| \leq \text{Adv}^{\text{prf}}(\mathcal{B}_1)$$

Game 2: In this game, we define $\mathcal{F}$ iteratively by lazy sampling: we maintain a table for

$\mathcal{F}$, and on input T : if T is already in the table, return the stored σ , otherwise sample $\sigma \leftarrow_{\$} \{0,1\}^\lambda$ uniformly, store and return it. Game 1 and Game 2 use equivalent methods to define $\mathcal{F}$, and thus

$$\Pr[\text{Game 1}] = \Pr[\text{Game 2}]$$

Game 3: We first define a Forge event: the aDec algorithm accepts a *fresh* ciphertext. In Game 2, for any fresh T^* , $\sigma = \text{PRF}(\text{mac}_{\text{key}}, T^*)$ is uniform and hidden. Acceptance requires guessing $\sigma = \text{mac}_{\ell+1}$. As MAC space is of λ bits, this succeeds with probability $\frac{1}{2^\lambda}$ per query, and thus

$$\Pr[\text{Forge}] \leq \frac{Q}{2^\lambda}.$$

Now in this game, it is identical to Game 2 except that on the first accepting fresh decryption query, the decryption oracle directly aborts. Then,

$$|\Pr[\text{Game 2}] - \Pr[\text{Game 3}]| \leq \Pr[\text{Forge}]$$

Game 4: In this game, we replace mac_i for $i \in [1, \ell]$ with fresh random elements.

Let $\mathcal{B}_2$ be a PPT adversary against anamorphic security we defined before, then we have

$$|\Pr[\text{Game 3}] - \Pr[\text{Game 4}]| \leq \text{Adv}_{\text{MAC}}^{\text{ana}}(\mathcal{B}_2).$$

Game 5: In the final game, all mac_i have been replaced by random values, and the decryption oracle gives no information. This is identical to the previous Game 4 and the adversary's entire view does not depend on the hidden bit b . Therefore, we have

$$\Pr[\text{Game 5}] = \Pr[\text{Game 4}] = \frac{1}{2}.$$

Combining the hops, we derive

$$\text{Adv}_{\text{MAC}}^{\text{ind-cca}}(\mathcal{A}) \leq \text{Adv}^{\text{prf}}(\mathcal{B}_1) + \text{Adv}_{\text{MAC}}^{\text{ana}}(\mathcal{B}_2) + \frac{Q}{2^\lambda}$$

which is negligible. □

Chapter 6

Discussion

6.1	On CREX	121
6.1.1	Limitations, possible improvements, and future work	121
6.2	On SCREX	124
6.2.1	Limitations, possible improvements, and future work	125
6.3	On AnaVoting	127
6.3.1	Limitations, possible improvements, and future work	127
6.4	On AnaSignal	129
6.4.1	Limitations, possible improvements, and future work	129
6.5	Final remarks	131

This chapter reflects on my four main contributions, discussing each work separately in terms of what it achieves, where it falls short, and how it can be improved. For each design, I identify concrete limitations, outline feasible refinements, and point to directions that I consider both meaningful and technically within reach. My aim is not to restate results, but to extract the design choices that matter, the assumptions that carry most weight, and the trade-offs that shaped the final form. I also highlight open questions and alternative paths that emerged during this thesis. Since each work has its own discussion section, here I focus on aspects that were not covered there or that emerged after those works were prepared, and I avoid repeating points already discussed in their respective locations. The goal is to offer a clear roadmap, both lessons learned and next steps, for readers who may wish to continue this line of research or adapt these ideas to adjacent settings. For ease of

reference, I refer to the works in Chapters 2, 3, 4, and 5 as CREX, SCREX, AnaVoting, and AnaSignal, respectively.

6.1 On CREX

The first effort to address coercion appears in Chapter 2, where I argue for the need to reason about coercion formally and introduce two new notions, *Anonymous Submission* and *Single-Blindness*. Building on these definitions, I propose a new e-exam protocol that aims to satisfy established e-exam security properties together with the two newly defined ones.

A central novelty is a shuffling-based marking workflow: instead of assigning whole submissions, individual question–answer pairs are distributed to (pseudo-)examiners via the *assignment* protocol. This yields two benefits. First, it mitigates collusion: a colluding examiner cannot control an entire script, so a corrupted candidate’s test cannot be wholly graded by a single ally. Second, it strengthens the perception of fairness: since all submissions are decomposed and distributed, candidates are more likely to be marked by a comparable mix of examiners, reducing disparities when some examiners are known to be consistently rigorous or permissive. This shuffling is the backbone of the design’s coercion-resistance during the marking phase.

Another contribution is a pen-and-paper attack on the exponentiation mixnet used in prior schemes (in particular, the Remark! e-exam protocol), exposing a critical flaw in what was then a state-of-the-art scheme. Subsequently, Dreier *et al.* corroborated this line of attack using the ProVerif model checker and examined its broader impact [59].

6.1.1 Limitations, possible improvements, and future work

While the paper offers clear contributions to secure e-exams, it also carries limitations worth noting.

Informal and incomplete *assignment* protocol

In CREX, the need for the *assignment* protocol is motivated and its functionality explained, but the construction remains informal and no concrete, fully specified design is given. The core idea is that the exam authority assigns a set of numbers representing the column indices of the shuffled matrix. Later, during the marking phase, there is no direct commu-

nication between the exam authority and the examiners; instead, once the exam authority publishes the full matrix of questions/answers, the examiners start the grading procedure.

How to fix. A concrete fix for this limitation is presented in Section 3.4.2 of my subsequent work. In brief, the exam authority issues designated ZK proofs to the examiners at assignment time, proving the correctness of the assignment. This binds the exam authority to that assignment and prevents it from later cheating by altering assignments during the marking phase.

Temporary coercion-resistance.

The threat model in CREX (Section 2.5.1) only considers *temporal* coercion up to the end of the marking phase. Even if the design were otherwise complete, it would not resist examiner coercion when the coercer can act *after* marking. The reason is that, in the notification phase, the candidate receives an exam authority-signed result S_C that reveals the mapping between answers and the (pseudo-)examiners who graded them.

$$S_C = \{ \text{Sign}_{SK_{E_1}, H_E}(M_{1_C}, P_{E_1}), \dots, \text{Sign}_{SK_{E_k}, H_E}(M_{k_C}, P_{E_k}) \}$$

$$\text{notification} = \text{Sign}_{SK_A}(S_C)_{\overline{PK}_C}$$

Concretely, the notification includes per-question outcomes that make the assignment *linkable*. A coercing candidate (or third party) can then verify whether a targeted examiner followed instructions by checking the notification against the expected grading order or examiner tag.

How to fix. The root of the issue is the examiners' explicit signatures on marks, which allow the candidate to know who graded which of their tests by verifying the signatures upon notification. Without delving into details, I outline several possible directions that require deeper investigation. One approach is for examiners to use group signatures [132, 133, 134, 135] or randomizable [136, 137] signature schemes instead of a standard signature scheme, hiding the explicit link between the signature and the issuer's identity. With group signatures, the candidate can verify that a member of the examiner group signed, but cannot learn who did so. However, deciding on the group manager and the conditions under which the master key may be revealed is challenging and requires careful consideration, as

the protocol must also handle the coercion threat model.

In the case of randomizable signatures, one initial design is as follows: for each assignment an examiner derives a fresh pseudonymous public key and signs under it; the exam authority attaches a zero-knowledge membership proof that this pseudonymous key belongs to the authorized examiner set, without revealing which one. Candidates can verify that it was “signed by an authorized examiner,” but cannot link the signature to a specific examiner or across assignments. Of course, this approach still requires careful consideration of other security factors, including the coercion threat model.

Weak coercion model

In both the examiner and the candidate coercion scenarios described in Section 2.6, the assumption is that the only secret leaked under coercion is the victim’s secret key. In other words, the design assumes that the randomness used for probabilistic encryption is not revealed to the coercer. This assumption is plausible when candidates or examiners can credibly claim that they have no access to the randomness source, for example, when a secure device ingests the secret key and performs encryption by sampling randomness that is inaccessible to the user. However, this restricts the applicability of the protocol: in many settings, the coercer can and will demand the randomness along with the victim’s other secret values.

How to fix. This limitation is addressed in my later work [38] by employing untappable channels during the assignment and submission steps. In short, these channels allow examiners or candidates to conceal their transmitted messages and plausibly deny specific communications, so that even under full disclosure of secret parameters to the coercer, the required unlinkability guarantees still hold. The details of this design are described in Section 3.4.

No formal model for the mixnet

As mentioned earlier, one of the achievements of CREX is the discovery of a critical flaw in the *exponentiation* mixnet that had previously been used by several security protocols, including the Remark! e-exam. Although CREX proposes an informal and seemingly adequate fix, it is not formalized. Instead, the protocol models the mixnet as an abstract secure component.

How to fix. A complete solution that provides a formal model of the mixnet is presented in [59]. In addition, this secure mixnet is later used as the core pseudonymizer in my work [38].

Absence of verifiability

As pointed out earlier in this thesis, verifiability is a key requirement for e-exam protocols, and CREX is no exception. The use of a public append-only bulletin board suggests that verifiability was intended in the design. However, the work does not specify verifiability goals or auditing procedures, nor does it provide a formal definition or analysis of verifiability.

How to fix. A standard approach to verifiability is to formally define the required properties and then analyze whether the protocol satisfies them. This direction is pursued in my later work [38].

6.2 On SCREX

SCREX is designed to overcome the shortcomings of its predecessor CREX and to offer a higher level of security and privacy. The most visible improvement is the concrete design of the assignment protocol, a building block that was largely absent in CREX. SCREX employs a novel distribution of indices to examiners together with designated zero-knowledge proofs, ensuring that only the intended examiner can verify the assignment.

The scheme also adopts the revised exponentiation mixnet [59] to achieve secure public-key pseudonymization, in contrast to CREX, where the mixnet was treated abstractly. Regarding the threat model, SCREX models coercion more strongly than CREX, as the coercer is now assumed to have access to the encryption randomness as well. This shift mandates a mechanism for plausible deniability of certain communications. Accordingly, in SCREX the sensitive exchanges are carried over an untappable channel, which provides the deniable communication required under the coercion threat model. Finally, SCREX defines and proves key verifiability properties, enabling both individual and universal verification that the procedures are performed as specified.

6.2.1 Limitations, possible improvements, and future work

SCREX is a significant step forward in the domain of secure e-exams. However, it is natural that it also carries security and functionality limitations, which I outline here.

Temporary coercion-resistance. SCREX inherits the same limitation as CREX: coercion-resistance holds only up to the end of the marking phase. The reason is that the underlying cause, namely the information revealed in the notification phase, is unchanged in SCREX. Consequently, as discussed earlier for CREX, a coercer can detect non-obedience upon observing the notification on the public bulletin board.

How to fix. To address this issue, the same remedy applies and is detailed in Section 6.1.1. Thus, the use of group signatures or randomizable signatures would be a plausible solution.

Reliance on untappable channel. SCREX assumes an untappable channel for the most sensitive exchanges during assignment and testing to preserve deniable communication under coercion. This is effective in the model but is a practical limitation: such channels are hard to realize at scale on ordinary networks, and the assumption places deployment risk on infrastructure that approximates untapability.

How to fix. A better direction is to replace the special channel with one that provides protocol-level deniability over standard networks. A plausible approach is to construct an *anamorphic channel*, leveraging anamorphic encryption to embed covert messages inside ordinary ciphertexts, thereby removing the need for an untappable link while retaining the required unlinkability and deniability guarantees. The aim of my later works [39, 40] is to construct such channels using specific cryptographic primitives.

There are three places in the SCREX protocol where an untappable channel is used:

1. When the exam authority sends M_2 message to examiners:

$$M_2 = \{Enc(P_1)_{\overline{PK}_{E_1}}, \dots, Enc(P_d)_{\overline{PK}_{E_d}}\}$$

2. When a candidate submits their test to the exam authority:

$$[Sign_{SK_C, h_C}(T_C)]_{PK_A}$$

3. When an examiner sends a grade to the exam authority:

$$[\text{Sign}_{SK_E, H_E}(q, a, m, H(q, P_E))]_{PK_A}$$

In all three cases, a message is encrypted using a probabilistic scheme (e.g., ElGamal), and the ciphertext is sent over the untappable channel. A natural alternative is to construct an anamorphic channel over ElGamal ciphertexts to carry hidden information on a public channel.

Scope note. The feasibility and exact form of an anamorphic channel in SCREX depend on the chosen signature and encryption schemes and on reasonable assumptions about secrets that need to be pre-shared. Only the first flow naturally fits constructions for limited message domains. For the second and third cases, either the protocol must be redesigned so that limited-domain constructions such as 4.3 apply, or a pre-shared double key is needed to enable a symmetric anamorphic channel. The protocol rules also influence the choice. If re-submission or re-grading is allowed, one can leverage multi-ciphertext constructions, such as two- or $(\ell + 1)$ - ElGamal ciphertexts. If re-submission and re-grading are disallowed, Schnorr-like signatures combined with ElGamal, as in Chapter 4, become attractive. The constructions presented in Chapter 4 and Chapter 5 are techniques that must be customized and adapted to the specific requirements of the protocol under consideration.

Coercion of the exam authority. The current threat model of the SCREX assumes that the exam authority does not cooperate with a coercer. Under this assumption, pseudonymization via the mixnet prevents linking pseudo-public keys to known rigorous or permissive examiners, which mitigates targeted manipulation. If the exam authority itself can be coerced into acting dishonestly, this protection weakens and assignment may be compromised. A natural next step is to consider a stronger threat model in which all parties except the mixnet (candidates, examiners, and the exam authority) may be coerced simultaneously. A full treatment of this setting is left to future work, where developing mechanisms that preserve assignment integrity and anonymity under these conditions appears both feasible and worthwhile.

6.3 On AnaVoting

AnaVoting introduces *ballot freedom*, defined as the ability of a voter to communicate their true voting intention to an auditor without the knowledge of dishonest authorities, even when those authorities control all channels and hold the election keys. Two complementary design families are presented: the first uses multiple ciphertexts, is inspired by Wang *et al.* [124], and fits revoting regimes; the second is single-shot and ties encryption randomness to attached proofs or signatures, using ElGamal and Schnorr-like components. The latter, to the best of my knowledge, is the first anamorphic construction that combines two different cryptographic primitives—in this case, ElGamal and Schnorr. These techniques integrate with Helios, Belenios, CHVote, and IVXV without altering protocol behavior and without weakening individual or universal verifiability. The evidence an auditor must see is identified so that deployment can follow a clear checklist. Overall, the results indicate that ballot freedom is achievable in deployed-style settings under standard cryptographic assumptions and public bulletin-board transparency.

6.3.1 Limitations, possible improvements, and future work

AnaVoting is, to the best of my knowledge, the first work to bridge anamorphic encryption and e-voting by examining whether anamorphic encryption can provide a hidden channel within existing voting systems. This is new ground and it raises challenges that require deeper study before the approach can meet its intended goals. In what follows, I outline the main challenges and limitations, and I point to possible improvements and directions for future research.

Formal definition of ballot freedom

The notion of ballot freedom is introduced but not formally defined. A formal definition is crucial since it enables rigorous analysis of this property and of the security of the overall voting system to assess whether all required properties are satisfied. A major concern is compatibility; If e-voting specific properties such as receipt-freeness and coercion-resistance are claimed, they must continue to hold when a voter uses a technique to signal their true voting intention to an auditor.

Formal proof of anamorphic constructions

The main cryptographic driver for ballot freedom in AnaVoting is the set of anamorphic constructions presented in Section 4.3. Although the signaling technique is novel, and two of the designs based on ElGamal encryption and Schnorr-like signatures/proofs are the first of their kind, they are not accompanied by formal security proofs.

How to fix. A concrete path forward is to formally define an *anamorphic security* notion via indistinguishability games that compare an ordinary world with an anamorphic world, and then prove that no efficient adversary can tell them apart [33, 124].

Message-space limits and efficiency

AnaVoting establishes the feasibility of anamorphic voting, or more precisely, ballot freedom, but it does not quantify performance and efficiency in detail, especially for the decryption algorithms. The effective message-space limit is also not characterized: since the number of candidates determines the message-space size, it remains to establish for how many candidates these constructions remain efficient in practice..

How to proceed. In terms of timing, it is sufficient to show that the constructions, once applied, have no effect on ordinary verification and decryption times, since they only replace a uniform random value with a biased one that carries an anamorphic message. For the honest auditor, however, performance is relevant: the auditor does not know whether a ballot contains an anamorphic vote, so every ballot must be checked, and the anamorphic decryption algorithms may require a search over the entire (albeit limited) message space. The effort should therefore focus on measuring the time the auditor needs to perform decryption for each construction, across different numbers of candidates, and for each relevant voting system and election use case, so that the cost of recovering an anamorphic vote is clearly characterized.

Investigating other voting systems

One concrete direction for extending AnaVoting is to investigate whether ballot freedom can be achieved in other voting systems, including those not yet deployed. This may require

designing new constructions tailored to the cryptographic primitives used by the systems in question.

6.4 On AnaSignal

This work asks a blunt question: if policymakers or platforms weaken E2EE through surveillance mechanisms, can ordinary users still communicate covertly? We show that the answer is yes by embedding anamorphic communication inside Signal’s Double Ratchet without changing its observable behavior. We design practical constructions that carry hidden data in artifacts already present in the protocol (DH ratchet elements and MAC tags) so that messages remain indistinguishable from ordinary traffic to the service and to any governmental authority aiming to surveil them. We prototype the designs in the official application and find that they operate within normal performance envelopes while conveying a non-trivial covert payload per message. Our aim is to demonstrate that weakening E2EE is a misguided strategy: covert channels can be realized efficiently within the same cryptographic building blocks these proposals target, making efforts to break or weaken E2EE ineffective in practice.

6.4.1 Limitations, possible improvements, and future work

AnaSignal is a significant step toward showing that deliberately compromising the E2EE of messaging applications, specifically the Signal protocol, is futile, by proposing practical anamorphic constructions. At the same time, this remains an active area with substantial scope for further work. The topic is of broad interest to the public, policymakers, and industry, and it directly affects the privacy of millions of users. In what follows, I outline lines of research and improvement suggestions that merit consideration.

More efficient algorithms

The current MAC-based construction in 5.5.1 is efficient, as it performs only one MAC and one XOR operation. For the DH-based constructions, however, the base designs are not efficient per se, and an optimized algorithm using a lookup table is proposed in 5.4.4 to support higher anamorphic bit rates, up to 32 bits or more. Even with this optimization, it is worth exploring more efficient algorithms and implementation techniques that could yield better performance. For example, if the message content is human-readable text,

using dictionary coding can yield a large bit rate advantage. In doing so, Alice and Bob can agree on a fixed indexed dictionary; to send the anamorphic message, they transmit the index rather than the whole word, which requires fewer bits to encode. One can observe that a 10000-word dictionary can be indexed with a 14-bit number, so regardless of word length, a word can be encoded as an anamorphic message using a 14-bit index. By contrast, unencoded printable ASCII uses 8 bits per character, so any word longer than two letters requires more than 16 bits. For English text messaging, a word is about five characters on average [138], which is roughly 40 bits, while the indexed variant is fixed at 14 bits. This is close to a threefold improvement in bit rate.

Moreover, depending on the construction, other standard source-coding techniques such as entropy coding can be applied to improve the effective bit rate of the hidden payload.

Investigating post-quantum secure messaging systems

As discussed in the paper, exploring anamorphism in post-quantum (PQ) designs is a natural direction, but it is not a straightforward adaptation of this work. The first challenge is the threat model: the adversary is equipped with a PQ computer and can break standard assumptions such as the discrete logarithm problem, so DH based designs, in their current form, no longer apply.

In the specific case of Triple Ratchet [119], the authors prevent *bad randomness* by requiring random elements to be generated by an honest source that is not controllable by the attacker. In our setting, the *attacker* corresponds to the sender (Alice or Bob), who must bias the randomness to carry an anamorphic message. Triple Ratchet therefore appears to deny access to the randomness source by design, which is a significant obstacle for constructing an anamorphic channel. A potential research direction is to identify alternative carrier signals, if any exist in the protocol state or metadata, that allow embedding an anamorphic message without relying on sender-controlled randomness.

Side-channel considerations

Side-channel attacks are a powerful class of attacks against both software and hardware. It is important to reduce their risk in our design. In our setting, however, the attacker is typically the server and does not have physical access to user devices running Signal. The most relevant channel is therefore remote timing. The protocol should ensure that

operations related to anamorphic communication do not create timing patterns that the attacker can measure to guess whether an anamorphic message is being sent. In practice, this means keeping timings, especially for decryption and MAC verification, within the normal range of the baseline system, that could produce a distinctive timing footprint.

6.5 Final remarks

In this dissertation, my aim was to focus on coercion in electronic assessment, to explain why it is relevant, to analyze its impact and the incentives of different parties, to define and model it, to design solutions that mitigate or prevent it, and to outline open questions that can strengthen future defenses against this real and powerful threat. I later broadened the direction of research to a closely related threat: surveillance, by looking beyond e-exams to a problem common to several security systems, namely whether one can design a secure, undetectable channel within an existing system. I studied this question in secure messaging, with a focus on the Signal protocol, and in e-voting for deployed and in-use schemes. In all three settings (e-exams, e-voting, and Signal) such a channel, if built properly, can add meaningful security. In e-exams it can serve as a backbone for coercion-resistance. In e-voting it can enable a voter to signal their true choice to an auditor. In Signal it can provide a reliable communication path even if E2EE is intentionally compromised by an actor wishing to surveil users' messages. I hope the techniques and models here help bridge theory and practice in domains where coercion is real.

References

- [1] Matthias Baume. “Online proctored exams: Where and how are they used? Basics, practical scenarios and technical solutions for online proctoring at european universities and educational institutions”. In: *INTED2019 Proceedings*. IATED. 2019, pp. 5216–5225.
- [2] Tobias Halbherr et al. “Making examinations more valid, meaningful and motivating: The online exams service at ETH Zurich”. In: *EUNIS Journal of Higher Education* (2014).
- [3] Sanaa Kaddoura and Abdu Gumaei. “Towards effective and efficient online exam systems using deep learning-based cheating detection approach”. In: *Intelligent Systems with Applications* 16 (2022), p. 200153.
- [4] Suvadeep Mukherjee et al. “The effects of privacy-non-invasive interventions on cheating prevention and user experience in unproctored online assessments: An empirical study”. In: *Computers & Education* 207 (2023), p. 104925.
- [5] Rosario Giustolisi, Gabriele Lenzini, and Giampaolo Bella. “What security for electronic exams?” In: *2013 International Conference on Risks and Security of Internet and Systems (CRiSIS)*. IEEE. 2013, pp. 1–5.
- [6] Giampaolo Bella, Rosario Giustolisi, and Gabriele Lenzini. “Secure exams despite malicious management”. In: *2014 Twelfth Annual International Conference on Privacy, Security and Trust*. IEEE. 2014, pp. 274–281. DOI: [10.1109/PST.2014.6890949](https://doi.org/10.1109/PST.2014.6890949).
- [7] Martin Abadi, Bruno Blanchet, and Cedric Fournet. “The applied pi calculus: Mobile values, new names, and secure communication”. In: *Journal of the ACM (JACM)* 65.1 (2017), pp. 1–41. DOI: [10.1145/3127586](https://doi.org/10.1145/3127586).
- [8] Bruno Blanchet et al. “An efficient cryptographic protocol verifier based on prolog rules.” In: *csfw*. Vol. 1. 2001, pp. 82–96.

- [9] Giampaolo Bella et al. "Remote management of face-to-face written authenticated though anonymous exams." In: *CSEDU* (2). 2011, pp. 431–437.
- [10] Rosario Giustolisi, Gabriele Lenzini, and Peter YA Ryan. "Remark!: A secure protocol for remote exams". In: *Security Protocols XXII: 22nd International Workshop, Cambridge, UK, March 19-21, 2014, Revised Selected Papers* 22. Springer. 2014, pp. 38–48. DOI: [10.1007/978-3-319-12400-1_5](https://doi.org/10.1007/978-3-319-12400-1_5).
- [11] Giampaolo Bella et al. "Trustworthy exams without trusted parties". In: *Computers & Security* 67 (2017), pp. 291–307. DOI: [10.1016/j.cose.2016.12.005](https://doi.org/10.1016/j.cose.2016.12.005).
- [12] Jannik Dreier et al. "Formal analysis of electronic exams". In: *2014 11th International Conference on Security and Cryptography (SECRYPT)*. IEEE. 2014, pp. 1–12. DOI: [10.5220/0005050901010112](https://doi.org/10.5220/0005050901010112).
- [13] David L Chaum. "Untraceable electronic mail, return addresses, and digital pseudonyms". In: *Communications of the ACM* 24.2 (1981), pp. 84–90.
- [14] Dogan Kesdogan, Jan Egner, and Roland Büschkes. "Stop-and-go-mixes providing probabilistic anonymity in an open system". In: *International Workshop on Information Hiding*. Springer. 1998, pp. 83–98.
- [15] C Andrew Neff. "A verifiable secret shuffle and its application to e-voting". In: *Proceedings of the 8th ACM conference on Computer and Communications Security*. 2001, pp. 116–125.
- [16] Roger Dingledine, Nick Mathewson, and Paul Syverson. "Tor: The second-generation onion router". In: (2004).
- [17] Ania M Piotrowska et al. "The loopix anonymity system". In: *26th unix security symposium (unix security 17)*. 2017, pp. 1199–1216.
- [18] Albert Hyukjae Kwon et al. "Riffle: An efficient communication system with strong anonymity". In: (2015).
- [19] Albert Kwon et al. "Atom: Scalable anonymity resistant to traffic analysis". In: *CoRR abs/1612.07841* (2016).
- [20] Henry Corrigan-Gibbs, Dan Boneh, and David Mazières. "Riposte: An anonymous messaging system handling millions of users". In: *2015 IEEE Symposium on Security and Privacy*. IEEE. 2015, pp. 321–338.

- [21] Philippe Golle and Markus Jakobsson. “Reusable anonymous return channels”. In: *Proceedings of the 2003 ACM Workshop on Privacy in the Electronic Society*. 2003, pp. 94–100.
- [22] Nethanel Gelernter, Amir Herzberg, and Hemi Leibowitz. “Two cents for strong anonymity: The anonymous post-office protocol”. In: *International Conference on Cryptology and Network Security*. Springer. 2017, pp. 390–412.
- [23] Andrea Huszti and Attila Petho. “A secure electronic exam system”. In: *Publicationes Mathematicae Debrecen* 77.3-4 (2010), pp. 299–312.
- [24] Myrto Arapinis, Sergiu Bursuc, and Mark Ryan. “Privacy-supporting cloud computing by in-browser key translation”. In: *Journal of Computer Security* 21.6 (2013), pp. 847–880.
- [25] Adam Vecsi and Attila Petho. “Scalix mix network”. In: *Acta Cybernetica, to Appear* (2022). DOI: [10.14232/actacyb.307671](https://doi.org/10.14232/actacyb.307671).
- [26] Roland Wen and Richard Buckland. “Masked ballot voting for receipt-free online elections”. In: *International Conference on E-Voting and Identity*. Springer. 2009, pp. 18–36.
- [27] David Chaum et al. “VoteXX: A solution to improper influence in voter-verifiable elections”. In: (2022).
- [28] Riccardo Barelli, Mario D’Onghia, and Stefano Longari. “Towards Secure Electronic Voting: a Survey on E-Voting Systems and Attacks”. In: *IEEE Access* (2025).
- [29] Rein Canetti et al. “Deniable encryption”. In: *Annual International Cryptology Conference*. Springer. 1997, pp. 90–104.
- [30] Amit Sahai and Brent Waters. “How to use indistinguishability obfuscation: deniable encryption, and more”. In: *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*. 2014, pp. 475–484.
- [31] Ran Canetti, Sunoo Park, and Oxana Poburinnaya. “Fully deniable interactive encryption”. In: *Annual International Cryptology Conference*. Springer. 2020, pp. 807–835.
- [32] Andrea Coladangelo, Shafi Goldwasser, and Umesh Vazirani. “Deniable encryption in a quantum world”. In: *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*. 2022, pp. 1378–1391.

- [33] Giuseppe Persiano, Duong Hieu Phan, and Moti Yung. “Anamorphic encryption: private communication against a dictator”. In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2022, pp. 34–63.
- [34] Fabio Banfi et al. “Anamorphic encryption, revisited”. In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2024, pp. 3–32.
- [35] Dario Catalano, Emanuele Giunta, and Francesco Migliaro. “Anamorphic encryption: New constructions and homomorphic realizations”. In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2024, pp. 33–62.
- [36] Dario Catalano, Emanuele Giunta, and Francesco Migliaro. “Generic anamorphic encryption, revisited: New limitations and constructions”. In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2025, pp. 275–303.
- [37] Mohammadamin Rakeei, Rosario Giustolisi, and Gabriele Lenzini. “Secure internet exams despite coercion”. In: *International Workshop on Data Privacy Management*. Springer. 2022, pp. 85–100.
- [38] Mohammadamin RAKEEI et al. “Secure and Verifiable Coercion-Resistant Electronic Exam”. English. In: *Journal of Cybersecurity* (In press). HDL: <https://orbi.lu.uni.lu/10993/67137>.
- [39] Rosario Giustolisi, Mohammadamin Rakeei, and Gabriele Lenzini. “Anamorphic Voting: Ballot Freedom Against Dishonest Authorities”. In: *Financial Cryptography and Data Security. FC 2025 International Workshops*. Cham: Springer Nature Switzerland, 2026, pp. 266–280.
- [40] Mohammadamin Rakeei et al. *Crypto Wars in Secure Messaging: Covert Channels in Signal Despite Leaked Keys*. Cryptology ePrint Archive, Paper 2025/2172. 2025. URL: <https://eprint.iacr.org/2025/2172>.
- [41] George R. Watson and James Sottile. “Cheating in the Digital Age: Do Students Cheat More in Online Courses?” In: *Online Journal of Distance Learning Administration* (2010).

- [42] Rosario Giustolisi. *Modelling and Verification of Secure Exams*. Information Security and Cryptography. Springer, 2018. ISBN: 978-3-319-67106-2. DOI: [10.1007/978-3-319-67107-9](https://doi.org/10.1007/978-3-319-67107-9).
- [43] Rosario Giustolisi, Vincenzo Iovino, and Gabriele Lenzini. “Privacy-Preserving Verifiability: A Case for an Electronic Exam Protocol”. In: *14th International Conference on Security and Cryptography (SECRYPT)*. SCITEPRESS Digital Library. 2017, pp. 139–150.
- [44] Rolf Haenni and Oliver Spycher. “Secure Internet Voting on Limited Devices with Anonymized {DSA} Public Keys”. In: *EVT/WOTE*. 2011.
- [45] Edgar R Weippl. *Security in e-learning*. Vol. 16. Springer Science & Business Media, 2005. DOI: [10.1145/1070939.1070943](https://doi.org/10.1145/1070939.1070943).
- [46] SM Furnell et al. “A security framework for online distance learning and training”. In: *Internet Research* 8.3 (1998), pp. 236–242. DOI: [10.1108/10662249810217821](https://doi.org/10.1108/10662249810217821).
- [47] Jannik Dreier et al. “A framework for analyzing verifiability in traditional and electronic exams”. In: *International Conference on Information Security Practice and Experience*. Springer. 2015, pp. 514–529. DOI: [10.1007/978-3-319-17533-1_35](https://doi.org/10.1007/978-3-319-17533-1_35).
- [48] Simon N Foley and Jeremy L Jacob. “Specifying security for computer supported collaborative working”. In: *Journal of Computer Security* 3.4 (1995), pp. 233–253.
- [49] Jordi Castella-Roca, Jordi Herrera-Joancomarti, and Aleix Dorca-Josa. “A secure e-exam management system”. In: *First International Conference on Availability, Reliability and Security (ARES’06)*. IEEE. 2006, 8–pp.
- [50] Sudeep Kanav, Peter Lammich, and Andrei Popescu. “A conference management system with verified document confidentiality”. In: *International Conference on Computer Aided Verification*. Springer. 2014, pp. 167–183.
- [51] Martín Abadi and Cédric Fournet. “Mobile values, new names, and secure communication”. In: *ACM Sigplan Notices* 36.3 (2001), pp. 104–115.
- [52] Taher ElGamal. “A public key cryptosystem and a signature scheme based on discrete logarithms”. In: *IEEE Trans. on information theory* (1985).
- [53] Eric Dubuis et al. “Verifizierbare internet-wahlen an schweizer hochschulen mit univote”. In: *INFORMATIK 2013* (2013).

- [54] Philipp Locher and Rolf Haenni. “A lightweight implementation of a shuffle proof for electronic voting systems”. In: *Informatik 2014* (2014).
- [55] Peter YA Ryan. “Crypto Santa”. In: *The New Codebreakers*. Springer, 2016, pp. 543–549.
- [56] Rolf Haenni and Reto E Koenig. “A generic approach to prevent board flooding attacks in coercion-resistant electronic voting schemes”. In: *Computers & Security* 33 (2013), pp. 59–69.
- [57] Giampaolo Bella et al. “A secure exam protocol without trusted parties”. In: *IFIP International Information Security and Privacy Conference*. Springer. 2015, pp. 495–509.
- [58] Alessandro Bruni, Rosario Giustolisi, and Carsten Schuermann. “Automated analysis of accountability”. In: *Information Security: 20th International Conference, ISC 2017, Ho Chi Minh City, Vietnam, November 22-24, 2017, Proceedings* 20. Springer. 2017, pp. 417–434. DOI: [10.1007/978-3-319-69659-1_23](https://doi.org/10.1007/978-3-319-69659-1_23).
- [59] Jannik Dreier, Pascal Lafourcade, and Dhekra Mahmoud. “Shaken, not Stirred - Automated Discovery of Subtle Attacks on Protocols using Mix-Nets”. In: *33rd USENIX Security Symposium (USENIX Security 24)*. Philadelphia, PA: USENIX Association, 2024, pp. 3135–3150. ISBN: 978-1-939133-44-1.
- [60] Khalil El-Khatib et al. “Privacy and security in e-learning”. In: *International Journal of Distance Education Technologies (IJDET)* 1.4 (2003), pp. 1–19. DOI: [10.4018/jdet.2003100101](https://doi.org/10.4018/jdet.2003100101).
- [61] Bruno Blanchet. “An Efficient Cryptographic Protocol Verifier Based on Prolog Rules”. In: *14th IEEE Computer Security Foundations Workshop (CSFW-14 2001), 11-13 June 2001, Cape Breton, Nova Scotia, Canada*. IEEE Computer Society, 2001, pp. 82–96. DOI: [10.1109/CSFW.2001.930138](https://doi.org/10.1109/CSFW.2001.930138).
- [62] Adam Vecsi and Attila Petho. “Auditable Anonymous Electronic Examination”. In: *Cryptography* 8.2 (2024), p. 19. DOI: [10.3390/cryptography8020019](https://doi.org/10.3390/cryptography8020019).
- [63] Bruno Blanchet. “The security protocol verifier ProVerif and its horn clause resolution algorithm”. In: *arXiv preprint arXiv:2211.12227* (2022). DOI: [10.48550/arXiv.2211.12227](https://doi.org/10.48550/arXiv.2211.12227).
- [64] D. Dolev and A. Yao. “On the security of public key protocols”. In: *IEEE Transactions on Information Theory* 29.2 (1983), pp. 198–208. DOI: [10.1109/TIT.1983.1056650](https://doi.org/10.1109/TIT.1983.1056650).

- [65] The Economist Group. *Democracy Index 2023*. <https://www.eiu.com/n/campaigns/democracy-index-2023>.
- [66] Benoît Chevallier-Mames et al. “On Some Incompatible Properties of Voting Schemes”. In: *Towards Trustworthy Elect.* 2010.
- [67] Sven Heiberg et al. “Improving the Verifiability of the Estonian Internet Voting Scheme”. In: *E-Vote-ID*. 2016.
- [68] Rolf Haenni et al. “CHVote protocol specification”. In: *Cryptology ePrint Archive* (2017).
- [69] Ben Adida. “Helios: Web-based Open-Audit Voting.” In: *USENIX security symposium*. Vol. 17. 2008, pp. 335–348.
- [70] Véronique Cortier et al. “Election Verifiability for Helios under Weaker Trust Assumptions”. In: *ESORICS*. Springer, 2014, pp. 327–344.
- [71] T. Elgamal. “A public key cryptosystem and a signature scheme based on discrete logarithms”. In: *Information Theory, IEEE Transactions on* 31.4 (1985), pp. 469–472. ISSN: 0018-9448. DOI: [10.1109/TIT.1985.1057074](https://doi.org/10.1109/TIT.1985.1057074).
- [72] Claus-Peter Schnorr. “Efficient Identification and Signatures for Smart Cards”. In: *CRYPTO*. 1989, pp. 239–252.
- [73] Yi Wang et al. “Sender-Anamorphic Encryption Reformulated: Achieving Robust and Generic Constructions”. In: *ASIACRYPT*. 2023.
- [74] Josh Benaloh. “Simple verifiable elections.” In: *EVT* 6 (2006), pp. 5–5.
- [75] Ben Adida et al. “Electing a university president using open-audit voting: Analysis of real-world use of Helios”. In: *EVT/WOTE* 9.10 (2009).
- [76] Véronique Cortier and Ben Smyth. “Attacking and fixing Helios: An analysis of ballot secrecy”. In: *Journal of Computer Security* 21.1 (2013), pp. 89–148.
- [77] David Bernhard, Olivier Pereira, and Bogdan Warinschi. “How not to prove yourself: Pitfalls of the fiat-shamir heuristic and applications to helios”. In: *ASIACRYPT*. Springer. 2012.
- [78] Véronique Cortier, Pierrick Gaudry, and Quentin Yang. “How to fake zero-knowledge proofs, again”. In: *E-Vote-Id*. Tal Tech Press. 2020.
- [79] Thomas Haines, Johannes Mueller, and Inigo Querejeta-Azurmendi. “Scalable Coercion-Resistant E-Voting under Weaker Trust Assumptions”. In: *SAC*. 2023.

- [80] Josh Benaloh, Michael Naehrig, and Olivier Pereira. “REACTIVE: Rethinking Effective Approaches Concerning Trustees in Verifiable Elections”. In: *Cryptology ePrint Archive* (2024).
- [81] Ariel J. Feldman and Josh Benaloh. “On subliminal channels in encrypt-on-cast voting systems”. In: EVT/WOTE. USENIX Association, 2009.
- [82] Chris Karlof, Naveen Sastry, and David Wagner. “Cryptographic voting protocols: A systems perspective”. In: *USENIX Security Symposium* (2005).
- [83] Ben Adida and C. Neff. “Efficient Receipt-Free Ballot Casting Resistant to Covert Channels.” In: *USENIX EVT/WOTE* (2008).
- [84] Tamara Finogina, Javier Herranz, and Peter B Roenne. “Expanding the toolbox: coercion and vote-selling at vote-casting revisited”. In: *E-Vote-ID* (2024).
- [85] J Liedtke et al. “Ordinos: a verifiable tally-hiding electronic voting protocol”. In: *IEEE EuroS&P*. 2020.
- [86] Véronique Cortier, Pierrick Gaudry, and Quentin Yang. “A Toolbox for Verifiable Tally-Hiding E-Voting Systems”. In: *ESORICS*. ESORICS, 2022, pp. 631–652.
- [87] Véronique Cortier, Pierrick Gaudry, and Quentin Yang. “Is the JCJ voting system really coercion-resistant?” In: *CSF*. 2024, pp. 186–200.
- [88] Jurlind Budurushi et al. “Pretty Understandable Democracy - A Secure and Understandable Internet Voting Scheme”. In: *2013 International Conference on Availability, Reliability and Security*. 2013.
- [89] Gurchetan S Grewal et al. “Caveat coercitor: Coercion-evidence in electronic voting”. In: *2013 IEEE Symposium on Security and Privacy*. IEEE. 2013, pp. 367–381.
- [90] Siamak Shahandashti and Feng Hao. “DRE-ip: A Verifiable E-Voting Scheme Without Tallying Authorities”. In: *ESORICS*. 2016, pp. 223–240.
- [91] Luke Harrison, Samiran Bag, and Feng Hao. “Camel: E2E Verifiable Instant Runoff Voting without Tallying Authorities”. In: *Asia CCS*. 2024, pp. 742–752.
- [92] Josh Benaloh et al. “ElectionGuard: a Cryptographic Toolkit to Enable Verifiable Elections”. In: *USENIX Security*). 2024.
- [93] Trevor Perrin and Moxie Marlinspike. “The X3DH Key Agreement Protocol”. In: <https://signal.org/docs/specifications/x3dh/> (2016).

- [94] Trevor Perrin and Moxie Marlinspike. “The double ratchet algorithm”. In: <https://signal.org/docs/specifications/doubleratchet/> 112.4 (2016).
- [95] Nadim Kobeissi, Karthikeyan Bhargavan, and Bruno Blanchet. “Automated verification for secure messaging protocols and their implementations: A symbolic and computational approach”. In: *2017 IEEE European symposium on security and privacy (EuroS&P)*. IEEE. 2017, pp. 435–450.
- [96] Ran Canetti et al. “Universally composable end-to-end secure messaging”. In: *Annual International Cryptology Conference*. Springer. 2022, pp. 3–33.
- [97] Alexander Bienstock et al. “A more complete analysis of the signal double ratchet algorithm”. In: *Annual International Cryptology Conference*. Springer. 2022, pp. 784–813.
- [98] Katriel Cohn-Gordon et al. “A formal security analysis of the signal messaging protocol”. In: *Journal of Cryptology* 33.4 (2020), pp. 1914–1983.
- [99] Joël Alwen, Sandro Coretti, and Yevgeniy Dodis. “The double ratchet: security notions, proofs, and modularization for the signal protocol”. In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2019, pp. 129–158.
- [100] Nihal Vatandas et al. “On the cryptographic deniability of the signal protocol”. In: *International Conference on Applied Cryptography and Network Security*. Springer. 2020, pp. 188–209.
- [101] Daniel Collins, Doreen Riepel, and Si An Oliver Tran. “On the tight security of the double ratchet”. In: *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*. 2024, pp. 4747–4761.
- [102] European Commission. *On ProtectEU: A European Internal Security Strategy*. Accessed: 07/2025. 2025. URL: <https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=celex:52025DC0148>.
- [103] BBC news. *Signal would walk from UK if Online Safety Bill undermined encryption*. Accessed: 08/2025. 2023. URL: <https://www.bbc.com/news/technology-64584001>.
- [104] digwatch. *Sweden considers law requiring encrypted messaging backdoors, Signal threatens to exit*. Accessed: 08/2025. 2025. URL: <https://dig.watch/updates/sweden-consider-s-law-requiring-encrypted-messaging-backdoors-signal-threatens-to-exit>.

- [105] cyberdaily.au. *Signal threatens to withdraw from Australia as government pushes for backdoor data access*. Accessed: 08/2025. 2025. URL: <https://www.cyberdaily.au/security/12437-signal-threatens-to-withdraw-from-australia-as-government-pushes-for-backdoor-data-access>.
- [106] EU Presidency. *Proposal for a Regulation of the European Parliament and of the Council laying down rules to prevent and combat child sexual abuse*. Accessed: 10/2025. 2025. URL: https://cdn.netzpolitik.org/wp-upload/2025/07/2025-07-24_Council_Presidency_LEWP_CSA-R_Compromise-texts_11596.pdf.
- [107] Hugo Krawczyk. “Cryptographic Extraction and Key Derivation: The HKDF Scheme”. In: *IACR Cryptol. ePrint Arch.* (2010), p. 264. URL: <http://eprint.iacr.org/2010/264>.
- [108] Mallory Knodel et al. *Definition of End-to-end Encryption*. Cryptology ePrint Archive, Paper 2024/2085. 2024. URL: <https://eprint.iacr.org/2024/2085>.
- [109] Anonymised. *AnamorphicSignal implementations*. <https://anonymous.4open.science/r/AnamorphicSignal-MAC64>. 2025.
- [110] Signal. *signalapp*. <https://github.com/signalapp>. Github repository, Accessed: 05/2025. 2025.
- [111] Signal. *Signal-Android*. <https://github.com/signalapp/Signal-Android>. Github repository, Accessed: 05/2025. 2025.
- [112] Signal. *libsignal*. <https://github.com/signalapp/libsignal>. Github repository, Accessed: 05/2025. 2025.
- [113] Nikita Borisov, Ian Goldberg, and Eric Brewer. “Off-the-record communication, or, why not to use PGP”. In: *Proceedings of the 2004 ACM Workshop on Privacy in the Electronic Society*. 2004, pp. 77–84.
- [114] Moxie Marlinspike. “Advanced cryptographic ratcheting”. In: <https://signal.org/blog/advanced-ratcheting/> (2013).
- [115] Bertram Poettering and Paul Rösler. “Towards bidirectional ratcheted key exchange”. In: *Annual International Cryptology Conference*. Springer. 2018, pp. 3–32.
- [116] Wil Liam Teng and Kasper Rasmussen. “Defending Against Attack on the Cloned: In-Band Active Man-in-the-Middle Detection for the Signal Protocol”. In: *arXiv preprint arXiv:2410.16098* (2024).

- [117] Ian Martiny et al. “Improving Signal’s Sealed Sender.” In: *NDSS*. 2021.
- [118] Ehren Kret and Rolfe Schmidt. “The PQXDH key agreement protocol”. In: <https://signal.org/docs/specifications/pqxdh/pqxdh.pdf> (2023).
- [119] Yevgeniy Dodis et al. “Triple Ratchet: A Bandwidth Efficient Hybrid-Secure Signal Protocol”. In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2025, pp. 302–331.
- [120] Rolfe Schmidt. “The ML-KEM Braid Protocol”. In: <https://signal.org/docs/specifications/mlkembraid/> (2025).
- [121] Benedikt Auerbach et al. “How to Compare Bandwidth Constrained Two-Party Secure Messaging Protocols: A Quest for A More Efficient and Secure Post-Quantum Protocol”. In: *Proceedings of the 34th USENIX Security Symposium*. 2025.
- [122] Yevgeniy Dodis et al. “Guarding the Signal: Secure Messaging with Reverse Firewalls”. In: *Cryptology ePrint Archive* (2025).
- [123] Mirosław Kutyłowski et al. “The self-anti-censorship nature of encryption: On the prevalence of anamorphic cryptography”. In: *Proceedings on Privacy Enhancing Technologies* 2023.4 (2023), pp. 170–183.
- [124] Yi Wang et al. “Sender-anamorphic encryption reformulated: Achieving robust and generic constructions”. In: *International Conference on the Theory and Application of Cryptology and Information Security*. Springer. 2023, pp. 135–167.
- [125] Shalini Banerjee et al. “Simple Public Key Anamorphic Encryption and Signature using Multi-Message Extensions”. In: *Cryptology ePrint Archive* (2025).
- [126] Yevgeniy Dodis and Eli Goldin. “Anamorphic-resistant encryption; or why the encryption debate is still alive”. In: *Cryptology ePrint Archive* (2025).
- [127] Davide Carnemolla et al. “Anamorphic resistant encryption: the good, the bad and the ugly”. In: *Cryptology ePrint Archive* (2025).
- [128] Gennaro Avitabile et al. “The Malice of ELFs: Practical Anamorphic-Resistant Encryption without Random Oracles”. In: *Cryptology ePrint Archive* (2025).
- [129] Rosario Giustolisi, Mohammadamin Rakeei, and Gabriele Lenzini. “Anamorphic Voting: Ballot Freedom Against Dishonest Authorities”. In: *Cryptology ePrint Archive* (2025).

- [130] Danilo Gligoroski, Mayank Raikwar, and Sonu Kumar Jha. “An LLM Framework For Cryptography Over Chat Channels”. In: *arXiv preprint arXiv:2504.08871* (2025).
- [131] Apple Security Engineering and Architecture (SEAR). “iMessage with PQ3: The new state of the art in quantum-secure messaging at scale”. In: <https://security.apple.com/blog/imessage-pq3/> (2024). Accessed: 07/2025.
- [132] David Chaum and Eugène Van Heyst. “Group signatures”. In: *Workshop on the Theory and Application of Cryptographic Techniques*. Springer. 1991, pp. 257–265.
- [133] Jan Camenisch and Markus Stadler. “Efficient group signature schemes for large groups”. In: *Annual international cryptology conference*. Springer. 1997, pp. 410–424.
- [134] S Dov Gordon, Jonathan Katz, and Vinod Vaikuntanathan. “A group signature scheme from lattice assumptions”. In: *International conference on the theory and application of cryptology and information security*. Springer. 2010, pp. 395–412.
- [135] Alexandra Boldyreva. “Threshold signatures, multisignatures and blind signatures based on the gap-Diffie-Hellman-group signature scheme”. In: *International Workshop on Public Key Cryptography*. Springer. 2002, pp. 31–46.
- [136] Jan Camenisch and Anna Lysyanskaya. “Signature schemes and anonymous credentials from bilinear maps”. In: *Annual international cryptology conference*. Springer. 2004, pp. 56–72.
- [137] David Pointcheval and Olivier Sanders. “Short randomizable signatures”. In: *Cryptographers’ Track at the RSA Conference*. Springer. 2016, pp. 111–126.
- [138] Carmen Pérez-Sabater. “Discovering language variation in WhatsApp text interactions”. In: *Onomázein* 31 (2015), pp. 113–126.