

Doctoral thesis

Doctoral theses at NTNU, 2023:201

Bor de Kock

From Lattice Crypto to *Lættis Krypto*: Various Approaches to Post-Quantum Key Exchange

NTNU
Norwegian University of Science and Technology
Thesis for the Degree of
Philosophiae Doctor
Faculty of Information Technology and Electrical
Engineering
Dept. of Information Security and
Communication Technology



Norwegian University of
Science and Technology

Bor de Kock

From Lattice Crypto to *Lættis Krypto*: Various Approaches to Post-Quantum Key Exchange

Thesis for the Degree of Philosophiae Doctor

Trondheim, June 2023

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Dept. of Information Security and Communication Technology



Norwegian University of
Science and Technology

NTNU

Norwegian University of Science and Technology

Thesis for the Degree of Philosophiae Doctor

Faculty of Information Technology and Electrical Engineering
Dept. of Information Security and Communication Technology

© Bor de Kock

ISBN 978-82-326-7104-5 (printed ver.)
ISBN 978-82-326-7103-8 (electronic ver.)
ISSN 1503-8181 (printed ver.)
ISSN 2703-8084 (online ver.)

Doctoral theses at NTNU, 2023:201

Printed by NTNU Grafisk senter

Lattice crypto

Cryptography based on *lattices*, mathematical groups constructed out of linear combinations of vectors.

Lættis krypto

Cryptography that is amusing, funny or even a bit ridiculous.

Acknowledgements

I owe an infinite amount of thanks to Colin Boyd, who has been my main supervisor, my main source of support, inspiration and whatever else I needed during the past four years. I consider myself fortunate to have had a supervisor that I could “connect” with, on more than just an academic level. It’s probably not an NTNU recommendation to spend a significant part of all supervisory meetings discussing the various Norwegian symphony orchestras and whatever concerts were happening in Trondheim every weekend, but I’m happy that we did. I am also very thankful to Kristian Gjøsteen, who ‘adopted’ me while Colin was on sabbatical. I’m honestly not sure how I had gotten through the first year of my PhD without you.

One of the cryptographers I look up to the most theorized that “a PhD student is a student, up until the last year of their PhD” [vV18]. It’s not a secret that I have enjoyed my time in Trondheim, and that I explored almost everything the (allegedly) best student city in Scandinavia has to offer. That was not the plan at all: I thought I was done with student life after finishing my time in Eindhoven, and moved to Trondheim to focus on more adult hobbies, such as my engagement for classical music. I applied to play the bass in the symphony orchestra at Studentersamfundet, and largely by accident that organization became my home away from home during the past four years. Samfundet is where I spent way too many hours organizing, discussing, making music, dancing, and drinking the occasional beer. A very tough period during the last couple of years was the corona pandemic, when The Netherlands was hit quite hard and I was not able to travel to see my friends and family — but I got to play the bass in *Spring Awakening*, and we sold out a week’s worth of shows. Not in the least Samfundet is where I learned the Norwegian language, got to know Norwegian culture like no one else, and met many, many amazing people.

Based on the stories I tell, it’s probably easy to forget that I actually *do* spend a lot of time in the big orange house at O.S. Bragstads plass 2. I think there are

only few academic environments that are so friendly as the Trondheim branch of the IIK department, where we eat lunch together every single day, get served pizza every week, and spend all year looking forward to the most absurd office Christmas party in this part of Europe. An honorable mention goes to the administration and the technical staff, who seem to have a solution to whatever problem we manage to put ourselves into. I am also very thankful to NaCl, our cryptography research group, where the travel budget is always negotiable but the cake-baking schedule is not.

I am truly grateful that many of my co-workers, cryptographers or not, have become my friends over the years, including former office mates, people with completely different research interests, support staff and those that have left the department since I arrived in Trondheim. When work is tough and none of my ideas seem to work out, there is always someone hanging around the coffee room to take my mind off whatever is bothering me. Going to work still feels like being part of some sort of big adventure, and going on conference trips still feels like going on a big cryptography summer camp with my friends.

On that note I owe many thanks to the cryptographic community as a whole, which I am proud to be a part of. A special thanks goes to all the co-authors of the works in this dissertation — but I also want to mention my almost-co-authors Tjerand Silde and Julia Hesse. I have really enjoyed doing research with you, although it did not lead to a publication so far.

The title of this dissertation has a technical meaning: during the last years I have moved my focus from cryptography based on lattices, to cryptography that is lættis: amusing, funny, or slightly odd. It also works on a meta level: I moved to Trondheim to do something clearly defined, structured and probably solvable somehow — but to the people outside of my bubble it looks like I have gotten myself into some intriguing Norwegian insider joke. I consider myself very fortunate that it ended up this way.

The biggest thanks of all goes to my family, especially my parents, Anne-Karin and Bertus, who have supported me through everything in life. Ik weet dat jullie het niet altijd leuk vinden dat ik zo ver weg ben, en dat de afgelopen jaren niet altijd zo makkelijk zijn geweest als we voor ogen hadden toen ik deze keuze maakte. Maar weet dat ik jullie ook elke dag mis, en dat ik het nóóit had volgehouden als ik niet wist hoe trots jullie zijn op mij en op alles wat ik de afgelopen jaren heb uitgespookt.

Bor de Kock
Trondheim, spring 2023.

Abstract

Key exchange is a cryptographic mechanism: it enables two or more parties to agree upon a shared key that is known only to them, even in the presence of an adversary that has access to all communication between the parties. In post-quantum key exchange we assume that this adversary additionally has access to a large-scale quantum computer that they can run computations on when trying to find the secret key. Several key exchange protocols that remedy this have been proposed in recent years, but a definitive solution is yet to be found.

This dissertation consists of four contributions that approach the issue of post-quantum key exchange from different angles. In the first contribution we create a new key exchange protocol using CSIDH, the Commutative variant of Supersingular Isogeny-based Diffie-Hellman. The protocol we introduce comes with an optimally tight security proof, due to CSIDHs similarity to classical (pre-quantum) Diffie-Hellman. The second contribution uses evolving symmetric keys to achieve the security properties typically found in public-key systems. In this work we provide five new protocols that all provide very small message sizes, and are proven to be secure in a new, strong, security model.

For the third contribution we use KEM, a primitive closely related to key exchange, as a modular component. We show that we can systematically build authenticated key exchange protocols, using KEM, digital signatures and Message Authentication Codes as modular building blocks. For the final contribution we build a non-interactive key exchange protocol based on lattice-cryptography. This is a construction that has been folklore for at least a decade, but has always been thought too impractical for real-world usage. We implement a passively secure variant of the scheme and show that it is significantly more practical than it was believed to be.

Sammendrag på norsk

Nøkkелutveksling er en kryptografisk mekanisme som gjør det mulig for to eller flere deltagere å bli enige om en delt nøkkel som bare de kjenner, selv om en motstander har tilgang til all kommunikasjon mellom deltagerne. I post-kvante nøkkelutveksling antar vi i tillegg at motstanderen har tilgang til en kvantedatamaskin av vesentlig størrelse, som hen kan benytte til å utføre beregninger for å finne den hemmelige nøkkelen. Flere nøkkelutvekslingsprotokoller er foreslått for å beskytte mot dette, men en endelig løsning har enn så lenge ikke blitt funnet.

Denne doktoravhandlingen består av fire bidrag som ser på spørsmålet rundt post-kvante nøkkelutveksling fra ulike vinkler. I det første bidraget skaper vi en ny nøkkelutvekslingsprotokoll ved bruk av CSIDH, den kommutative varianten av en Diffie-Hellman-protokoll basert på supersingulære isogenier mellom elliptiske kurver. Vår protokoll kommer med et optimalt bevis, der vi benytter likhetene mellom CSIDH og klassisk (pre-kvante) Diffie-Hellman. Det andre bidraget bruker symmetriske nøkler som med evolusjon oppnår sikkerhetsegenskapene vi vanligvis finner i systemer basert på offentlige nøkler. I dette arbeidet presenterer vi fem nye protokoller som alle gir meldinger av veldig liten størrelse, og som alle er bevist sikre i en ny, sterk sikkerhetsmodell.

I det tredje bidraget bruker vi KEM, et primitiv tett relatert til nøkkelutveksling, som modulær komponent. Vi viser at vi kan sette sammen autentiserte nøkkelutvekslingsprotokoller på systematisk vis ved bruk av KEM, digitale signaturer og meldingautentiseringskoder (MAC) som byggeklosser. I det siste bidraget skaper vi en nøkkelutvekslingsprotokoll uten interaksjon, der vi benytter kryptografi basert på gitte. Dette er en konstruksjon som har vært kjent i kryptografimiljøet minst et tiår, men som alltid har blitt ansett som for upraktisk for realistisk bruk. Vi implementerer en passivt sikker variant av protokollen, og viser at den er vesentlig mer praktisk gjennomførbar enn det som har vært antatt hittil.

Samenvatting in het Nederlands

Sleuteluitwisseling is een cryptografisch mechanisme: het stelt twee of meer partijen in staat een gedeelde sleutel overeen te komen die alleen bij hen bekend is, zelfs in de aanwezigheid van een tegenstander die toegang heeft tot alle communicatie tussen de partijen. In post-kwantumcryptografie doen we de aanname dat de af luisterende tegenstander daarnaast ook toegang heeft tot een kwantumcomputer van significant formaat, waarop diegene berekeningen kan uitvoeren om de geheime sleutel te proberen te vinden. Om dit tegen te gaan zijn er de afgelopen jaren verschillende sleuteluitwisselingsprotocollen ontwikkeld, maar een uiteindelijke oplossing is nog niet gevonden.

Dit proefschrift bestaat uit vier bijdragen, die het vraagstuk rond post-kwantum sleuteluitwisseling vanuit verschillende hoeken benaderen. In de eerste bijdrage creëren we een nieuw sleuteluitwisselingsprotocol dat gebruik maakt van CSIDH, de commutatieve variant van een op supersingulaire isogenieën tussen elliptische krommen gebaseerd Diffie-Hellmanprotocol. Ons protocol is voorzien van een optimaal strak bewijs, waarbij gebruik wordt gemaakt van de gelijknissen tussen CSIDH en klassiek (pre-kwantum) Diffie-Hellman. De tweede bijdrage maakt gebruik van evoluerende symmetrische sleutels om de beveiligingseigenschappen te bereiken die we normaal juist in systemen met publieke sleutels vinden. In dit werk presenteren we vijf nieuwe protocollen, die allemaal werken met zeer kleine berichten tussen de partijen, en die bovendien bewezen veilig zijn in een nieuw, sterk beveiligingsmodel.

In de derde bijdrage gebruiken we KEM, een primitief dat sterk aan sleuteluitwisseling gerelateerd is, als modulair component. We tonen aan dat we op systematische wijze geauthenticeerde sleuteluitwisselingsprotocollen kunnen samenstellen, waarbij we KEM, digitale handtekeningen en berichtauthenticatiecodes (MAC) als bouwstenen gebruiken. In de laatste bijdrage creëren we een sleuteluitwisselingsprotocol zonder interactie waarbij we gebruik maken van cryptografie gebaseerd op roosters in de euclidische ruimte. Dit is een construc-

tie die volgens de overgave al minstens een decennium bekend is, maar altijd beschouwd werd als te onpraktisch voor realistisch gebruik. We implementeren een passief-veilige variant van het protocol en tonen aan dat dit significant efficiënter is dan tot nu toe werd aangenomen.

Contents

1 Introduction	15
1.1 Motivation	16
1.2 Open problems	17
1.3 Overview of published works	18
1.4 Outline	19
2 Background	21
2.1 Key Exchange	21
2.2 Post-quantum cryptography	23
2.2.1 Lattice-based cryptography	24
2.2.2 Supersingular Isogeny-based cryptography	25
2.3 AKE with Symmetric Keys	26
2.4 Concluding remarks	27
2.4.1 Comparison to open problems in the field	28
2.4.2 Future work	29
2.4.3 Regarding real-world security	30
A Practical Isogeny-Based Key-Exchange with Optimal Tightness	37
B Symmetric Key Exchange with Full Forward Security and Robust Synchronization	73
C Modular Design of KEM-Based Authenticated Key Exchange	143
D SWOOSH: Practical Lattice-Based Non-Interactive Key Exchange	189

Chapter 1

Introduction

The field of cryptography deals with topics related to secure communication, including encryption and authentication. Within this field, key exchange is the primitive that deals with the very first step of a secure conversation: how do two parties agree on a shared secret if their conversation is public? Although cryptography with pre-shared keys dates back thousands of years, key exchange mostly became a relevant problem around the advent of large-scale computing and the internet, in the late 20th century. Diffie and Hellman essentially solved the problem in 1976 with the introduction of what became known as Diffie-Hellman protocol [DH76]. We briefly explain the textbook version: within a known group G with generator g , Alice and Bob choose secret keys a and b respectively, and exchange their ephemeral public keys $A = g^a$ and $B = g^b$. Now since $A^b = B^a$, Alice and Bob can both compute a shared secret. Since we know computing discrete logarithms is hard (i.e. when $u = v^w$, it is hard to find w given only u and v), we know that it is not feasible for an attacker to find the shared secret, or the value of a and b given only the public messages A and B .

Although the protocol has undergone some changes over the years, for instance replacing numbers in discrete groups with fast elliptic curve computations such as Curve25519 [Ber06], Diffie-Hellman is still secure enough to be in use on the internet today. The reason we consider Diffie-Hellman and its alternatives secure, is what we call its computational hardness: DH is not unbreakable, but breaking it is so hard that we cannot feasibly do it, even given all the computer power in the world.

The introduction of quantum computing will fundamentally change this: in the

1980s a quantum computer was proposed, and subsequently, Shor and Grover showed that this computer, if built on a large enough scale, would be able to break most cryptographic systems in use today [Sho94, Gro96].

The field we denote as post-quantum cryptography works on exactly this transition: how can we change the cryptographic systems and algorithms we have been using for half a century, so that they achieve the same security goals using a completely different approach?

The research project that has led to this publication was originally called *Post-Quantum Key Exchange*. The resulting dissertation does not provide one definitive answer on how we should do key exchange in a post-quantum world, nor does it claim that one cryptographic problem is better than the others: instead we explore various ways post-quantum key exchange can take place, depending on the restrictions posed by the situation that we're in, the hardware that we use or the security goals that we try to achieve. The full title reflects that both our field in itself, as well as my understanding of it, has broadened through the years — and that in cryptography, the best solutions often are a little bit amusing.

1.1 Motivation

The field of post-quantum cryptography is relatively new, but the time pressure is enormous. Physicists estimate that a large enough quantum computer will be available within decades, which means that new systems will need to be available as soon as possible. Designing, implementing and standardizing cryptographic systems takes time, and there also needs to be time to roll out the system to users [BL17].

Various initiatives have been started to move towards a 'quantum safe' internet: perhaps most prominently the United States National Institute of Standards and Technology (NIST) is currently aiming to standardize post-quantum methods for key establishment, as well as a digital signature scheme. This focus on public-key schemes is due to the fact that the impact of the quantum computer is significantly larger on public-key cryptography than it is on symmetric schemes, where AES with larger key-sizes are expected to remain secure. The first round of this standardization process ("The NIST competition") started with the submission of proposals roughly a year before this project, and has in summer 2022 led to one proposed key encapsulation mechanism (KEM) for standardization [SAB⁺20], with four additional KEMs labeled for possible future standardization [ABB⁺22, ABC⁺22, AAB⁺22, JAC⁺22]. The NIST competition has not only led to the proposal of interesting KEM-candidates, but also to a large influx of

papers related to these KEMs in the form of attacks, analyses, implementations and improvements. In that sense, this is a truly exciting time to work on KEM research.

Pessimistically, we can note that we in a sense already are too late. Large-scale actors such as nation-state adversaries are currently storing large quantities of data with the goal of decrypting it as soon as possible [ON21]. For some data currently considered secret, this is not an issue, while it can be problematic for military information and other state secrets.

1.2 Open problems

We can classify the open problems in our research field as follows.

1. **Primitives.** Gaining better insights in the mathematical background of post-quantum cryptographic primitives, through reductions and other forms of provable security. Examples of primitives that are currently being used and that can be researched are
 - (a) supersingular isogenies on elliptic curves [JDF11], which uses paths walked in the graph of isogenies between elliptic curves;
 - (b) code-based cryptography, based on known-hard problems in coding theory. An example is the McEliece scheme [EOS06], which is based on decoding a general linear code;
 - (c) multivariate cryptography [DS04], which is based on the hardness of solving systems of multivariate equations. An example of such a system is the Unbalanced Oil and Vinegar scheme [KPG99];
 - (d) lattice-based cryptography like the New Hope scheme [ADPS16a], which is based on the hardness of the Ring-Learning With Errors problem;
2. **Cryptanalysis.** Research related to the feasible attacks on post-quantum cryptographic primitives and how these can be optimized. There are different types of attacks for each class of cryptographic primitive mentioned above, and we can differentiate between attacks on a scheme's mathematical security, versus attacks on an implementation. In lattice-based protocols we can for instance break the underlying mathematical problems by apply lattice-reduction and enumeration attacks [LP11]. By improving these kinds of attacks we get a better insight into how secure our schemes actually are and what our parameter choices need to be for a scheme to be secure in the real world. When it comes to attacking implemented

schemes, an important type of cryptanalysis is formed by side-channel attacks. Here we try to obtain information about a cryptographic key by measuring variations in e.g. power consumption or running time [Koc96].

3. **Instantiations.** How we can improve the primitives or instantiations. Examples of this include:
 - (a) finding a suitable post-quantum alternative of the Diffie-Hellman key exchange [DH76], which plays a central role in the cryptography from before the post-quantum era;
 - (b) creating key-encapsulation mechanisms based on those key exchange protocols, which is the type of transformation that a large number of NIST-proposals [Kim16] rely on for their protocol;
 - (c) investigating the possibility of transforming existing two-party protocols such that they are usable for other purposes, for instance group key exchanges and password-based key exchanges;
 - (d) investigating other security properties of the proposed key exchanges, for example forward secrecy, forward anonymity, plausible deniability and so on.
 - (e) investigating how the results obtained on cryptographic attacks relate to the design and parameter choices of the protocols, and seeing how these attacks can be mitigated through improving these choices.
4. **Tight security.** Insights into how closely specific instances of protocols relate to the theory, and applying the theory to see how we can improve the security and efficiency of these instances by making different parameter choices.

These open problems have served as the starting point for the work in this dissertation — but the scope of the field is so large that we have not been able to address (let alone solve) all of them. In Section 2.4 we look at how the results obtained in the various papers align with these open problems.

1.3 Overview of published works

The main contribution covered in this thesis consists of the following four papers, in chronological order:

- A. *Practical Isogeny-Based Key-Exchange with Optimal Tightness* [dKGV20]
Bor de Kock, Kristian Gjøsteen and Mattia Veroni

Published at the International Conference on Selected Areas in Cryptography, SAC 2020: Selected Areas in Cryptography pp 451–479 (LNSC, volume 12804).

- B. *Symmetric Key Exchange with Full Forward Security and Robust Synchronization* [BDdK⁺21]

Colin Boyd, Gareth T. Davies, Bor de Kock, Kai Gellert, Tibor Jager and Lise Millerjord

Published at the International Conference on the Theory and Application of Cryptology and Information Security ASIACRYPT 2021: Advances in Cryptology – ASIACRYPT 2021 pp 681–710 (LNSC, volume 13093)

- C. *Modular Design of KEM-Based Authenticated Key Exchange* [BdKM23]

Colin Boyd, Bor de Kock and Lise Millerjord

Accepted for publication at ACISP 2023 (the 28th Australasian Conference on Information Security and Privacy).

A manuscript is made public on the Cryptology ePrint Archive under number 2023/167.

- D. SWOOSH: *Practical Lattice-Based Non-Interactive Key Exchange* [GdKQ⁺23]

Phillip Gajland, Bor de Kock, Miguel Quaresma, Giulio Malavolta and Peter Schwabe

The work is in submission. A manuscript is made public on the Cryptology ePrint Archive under number 2023/271.

In all works except Paper [D], we follow the mathematical tradition of listing authors in alphabetical order of last name. Papers [A.] and [B.] have been published at peer-reviewed venues, and the full version is included as part of this thesis. Papers [C.] and [D.] are at the time of writing in submission: the version included in the thesis is a manuscript or pre-print, and these are not expected to be the final versions of these articles.

All works that are included in this thesis are available publicly through the cryptology ePrint archive, and are released under a Creative Commons-license.

1.4 Outline

This dissertation has the form of a paper collection. In this introductory chapter we have given the motivation and a brief introduction to the field, after which we have mentioned the original goals of the projects and introduced the papers in this dissertation.

¹See the 2004 AMS Statement on The Culture of Research and Scholarship in Mathematics: <https://www.ams.org/profession/leaders/culture/CultureStatement04.pdf>

In the next chapter, we give the necessary technical context to understand the field we work in, and explain how the technical contributions in the dissertation relate to this context, and the cryptographic research field in its entirety. The second part contains of the four papers that were written during the course of this project.

Chapter 2

Background

We now introduce the context for the various papers that this dissertation consists of, and explain how these papers fit together.

2.1 Key Exchange

Introduced in the previous chapter, *key exchange* is the technology used to agree upon a shared secret, between two parties, on a publicly available net. Besides the obvious *correctness* aspect (do both parties compute the same key?), there are various other properties we are interested in.

Security definitions and experiments. As the various works in this thesis consider different models and settings, we defer to the specific papers for the formal definitions of security in each case, but as an example we can consider the perhaps most basic key exchange experiment: We execute a key exchange protocol, and make a transcript of all messages. Now we give this transcript to our adversary \mathcal{A} , along with a key \hat{k} . Depending on a coin flip, \hat{k} is either the real key from the execution, or a randomly generated one. Can \mathcal{A} tell whether \hat{k} is real [KL14]?

Authentication. We use the term *authenticated key exchange (AKE)*, to denote a key exchange protocol that authenticates the users with respect to each other: party A is convinced that party B is indeed the person who they claim to be (and vice-versa).

Interactivity. *Non-interactive key exchange (NIKE)* is a class of key exchange algorithms that does not require *interaction*, i.e. the parties communicate asynchronously. When instantiating key exchange with a new partner, this means a party can simply download that partner’s public key, derive a shared secret and send an encrypted message before going offline again. When the other partner comes online, they are then able to derive the shared secret and decrypt the message. NIKE is very easy to achieve in the pre-quantum world — Diffie-Hellman is for instance a NIKE almost by default.

Symmetric cryptography. Symmetric cryptography is in some sense the “opposite” of the public-key cryptography most of this work is about: it is the cryptographic field that uses a previously agreed upon key to encrypt a stream of data, and decrypts it with the same key. Because of this difference, we often denote public-key cryptography as asymmetric cryptography.

In reality symmetric and asymmetric cryptography are often used hand-in-hand: we use key exchange to agree on a shared secret which we then use to derive a symmetric key, or as input to a next step. This way we achieve the efficiency of symmetric cryptography, combined with the interesting practical aspects and security properties of public-key cryptography. We therefore often use a primitive called a *key encapsulation mechanism (KEM)*, which informally has the goal to output a shared secret to both parties such that they can engage in further communications using a different scheme.

Paper C. Modular Design of KEM-Based Authenticated Key Exchange

Authenticated key exchange (AKE) can be built up using various strategies: the most straightforward approach is taking a key exchange mechanism and applying an authenticator to it. Perhaps the most common way of doing this traditionally is using signatures, but using a KEM, a MAC or another authentication method is also possible: in a post-quantum context this is extremely relevant, since KEMs can be more efficient than signatures. In addition, the post-quantum field is moving rapidly and our understanding of the various algorithms changes constantly: the NIST competition works as a catalyst given its focus on KEMs and signatures. It is therefore also important to be less dependent on one specific authentication algorithm.

In this work we abstract the various KEMs and authenticators: using this model and the various modular proofs we provide it is then possible to derive various protocols for authentication and key exchange. Some of these systematically built-up protocols already exist in the literature, others are new.

Although this paper does not introduce any new post-quantum primitives, it creates a model in which such primitives can be rolled out in an effective manner; it thus lays the ground work for new, post-quantum, systems yet to be developed.

2.2 Post-quantum cryptography

As introduced in Chapter 1, post-quantum cryptography or PQ-crypto is the field of research that focuses on introducing new cryptographic algorithms that are able to resist the kinds of attacks that become possible when our adversaries get access to a large-scale quantum computer. Post-quantum cryptography is a field that emerged during the previous decades. Although no quantum computer large enough to break practical cryptography has been built to date, its capabilities have been known since the 1990s. In 1994 Shor introduced an algorithm for the efficient prime factorization of large numbers, and a variant for the computation of the discrete logarithm. As an example, the factorization of a large number $N = pq$ with p and q prime can be done in $O(n^3 \log n)$ operations, using Shor's algorithm on $2n + 3$ physical *quantum bit operators* (qbits). Grover's algorithm, introduced in 1996, can attack even more cryptographic systems, but it will only speed up computations from N to \sqrt{N} [Sho94, Gro96, BL17].

Within the scope of public-key cryptography, Shor's algorithm is the largest issue. While we can secure a symmetric protocol like AES simply by picking larger key size, protocols like RSA and Diffie-Hellman are considered completely broken and need to be replaced.

Research in post-quantum key exchange can roughly be divided into four "families" of mathematical problems: (1) supersingular isogeny-based cryptography, (2) code-based cryptography, (3) multivariate cryptography and (4) cryptography based on lattices.

During summer 2022, NIST presented its the result of the third round of its competition. In 'our' category *Public-key Encryption and Key-establishment Algorithms* the candidate for standardization is CRYSTALS-KYBER, a lattice-based scheme [SAB⁺20]. Candidates for a fourth round of further research are three code-based schemes (BIKE [ABB⁺22], Classic McEliece [ABC⁺22] and HQC [AAB⁺22]) and the (since then retracted) isogeny-based scheme SIKE [JAC⁺22].

In this work we focus on schemes based on supersingular isogenies, and lattices.

2.2.1 Lattice-based cryptography

A lattice is a mathematical structure consisting of points in a multidimensional plane, formed by composing *base vectors* from a given set. Interestingly enough lattices pose various hard computational challenges, for instance finding the nearest lattice point to a given vector in our plane (the *closest vector problem*), or finding the shortest vector that exists in a given lattice (the *shortest vector problem*). If the lattice is of a high enough dimension, these problems are believed to be computationally infeasible to solve even by using a quantum computer.

Following the analogy above, where we built the Diffie-Hellman exchange (a scheme) upon the discrete logarithm problem (a primitive) to create a KEM, our goal with lattice cryptography is to build new schemes using these lattice problems as the underlying hard problems.

Lattice-based cryptography is often based on a problem called *Learning With Errors (LWE)*, and its variants *Ring-LWE* and *Module-LWE*. R-LWE rose to early prominence as a possible post-quantum scheme after an early scheme called New Hope [ADPS16b] was implemented into Google Chrome during a 2016 test run. [Lan16b] [Lan16a]

Although LWE-based schemes are versatile and can be used for many applications, they generally require some sort of (interactive) reconciliation. Using LWE for non-interactive key exchange (NIKE) has long been considered impractical, although the theoretical possibility of LWE-based NIKE has for a while been considered a ‘folklore’ idea, for instance by Lyubachevski in a 2017 Stack Exchange post [Lyu17].

In 2018, De Kock [dK18] made a first attempt at finding parameters that would make RLWE-based NIKE possible, but the work did not satisfy real-world security requirements. A 2020 work [GKRS20] showed the limitations of real-world lattice cryptography, and denotes that a non-interactive LWE-based scheme is impossible for many modulus-to-noise-ratios.

Paper [D]. SWOOSH: Practical Lattice-Based Non-Interactive Key Exchange

In this work we show that these inefficiency concerns are smaller than they were thought to be: we propose parameters for an LWE-based scheme, which we show to be secure and correct in both the passively and actively secure setting. The work comes with an implementation of the scheme’s passively secure version, which is competitive in terms of running time and message size. Moving the work into an active setting requires a transformation using

■ NIZKs: we have not implemented this in this version of the work.

2.2.2 Supersingular Isogeny-based cryptography

A different approach to post-quantum key exchange is cryptography based on isogeny problems. A supersingular isogeny is a relationship between two supersingular elliptic curves over finite fields: we can draw what is called an *isogeny graph* where every point represents a curve and every edge is an isogeny.

Until recently, isogeny-based cryptography was considered the most interesting direct translation of Diffie-Hellman to the post-quantum setting. In what is denoted *Supersingular Isogeny-based Diffie-Hellman (SIDH)*, one of the NIST candidates under the name *Supersingular Isogeny Key Exchange (SIKE)*, the two parties move between two curves in the isogeny graph, each using their own route (*walking* the graph). What made the scheme interesting is that such a graph walk is easy for a participant who, informally speaking, knows ‘which route to take’ — but computing the route between two given points was considered computationally infeasible. The resulting scheme closely mimics the structure of the Diffie-Hellman key exchange, and its interesting features like non-interactivity would therefore be easy to achieve in a post-quantum setting using SIKE. In addition, SIKE had among the smallest keys of any post-quantum candidate, making it extremely attractive for all kinds of use cases.

We speak in the past tense since a devastating attack on SIKE was published in summer 2022, and the scheme is now considered insecure [CD22].

In the work below we work with a variant of SIDH called CSIDH (Supersingular Isogeny-Based Diffie Hellman with Commutative group actions – pronounced *seaside*). Although CSIDH is an example of a scheme that does not fall victim to the new attack on SIKE, it is not as implementation-ready as SIKE was and is not a contender in the NIST competition: concretely we know that the parameters proposed in the original CSIDH paper are not secure enough for real-world use [Pei20].

More fundamentally, one can claim that the sudden appearance of an attack on SIKE has demonstrated how little developed this research field still is — without underselling the novelty of this new attack, it was fundamentally based on a theory that has existed since 1997, but that no one so far applied to these schemes [CD22]. Whether even a provably secure isogeny-based scheme will ever be considered reliable enough for real-world adoption is therefore a different question altogether.

Paper [A](#). Practical Isogeny-Based Key-Exchange with Optimal Tightness

The commutative property of CSIDH makes it possible to obtain a security proof based on random self-reducibility, which is what we use in this work: this essentially means transforming any hard instance of a problem to a random, average-hardness instance, performing the computation we want to, and then transforming the result back to the instance we were trying to look at. Because this is possible we obtain an optimal proof for CSIDH, that is, a proof with the smallest possible loss. This is important when implementing theoretical cryptographic protocols in a practical way, as the loss tells us how the real-world parameter size for a protocol relates to the hardness of the underlying mathematical problem. The smaller the loss, the smaller the parameters can be chosen, and the more efficient our system will become in practice.

It should be remarked that a paper by Kawashima et al. obtained an almost identical result in parallel with ours [\[KTAT20\]](#): a preprint version of that work was submitted to ePrint less than a week after our result was published there.

As mentioned above, a series of papers presenting attacks on isogeny-based cryptosystems were published summer 2022, however it should be remarked that these are not applicable to our work. There have been various other developments related to our work, for instance a 2022 paper which proves the security of our protocol in the quantum random oracle model (QROM) [\[DHK⁺22\]](#), while our proofs are in the ROM.

2.3 AKE with Symmetric Keys

In Section [2.1](#) we briefly discussed symmetric cryptography as a technology complementing public-key cryptography: it is computationally fast and its message complexity is small, but we lack some of the advantages we get when using a KEM, such as perfect forward secrecy: the guarantee that compromising a long-term secret will not enable an attacker to obtain the session keys that are used.

In 2019, Avoine et al. proposed a new protocol called *Symmetric-Key Authenticated Key Exchange* (SAKE) [\[ACF20\]](#): it aims to create a protocol based on symmetric-key cryptography which is able to guarantee the same security properties as we traditionally obtain using public-key cryptography, most notably perfect forward secrecy. In the same work they also introduce SAKE-AM, for *aggressive mode*, which inverts the roles of initiator and responder.

Paper B. Symmetric Key Exchange with Full Forward Security and Robust Synchronization

We introduce five new protocols in which the parties use evolving keys — keys can be updated either using a linear derivation method or by means of puncturable pseudorandom functions. Although this solves several security issues, it also introduces a challenges with respect to correctness, and new attack vectors. In the above-mentioned scenario with low-power IoT-devices, for which these protocols are very suitable, an adversary can cause problems by forcing the different parties' out of sync on purpose or by forcing a party to update their key continuously until the power is drained. We formalize these new properties in a new model, and prove our different protocols secure in these models. The protocols are in terms of both security and (message) efficiency an improvement compared to the state of the art.

The paper is very different than the others in terms of approach and result, but still fits within the scope of post-quantum key exchange: migrating to an (inherently post-quantum) symmetric algorithm instead of the usual pre-quantum key exchange algorithms is an effective way of securing ourselves against a quantum adversary in this setting.

SAKE and SAKE-like protocols have several interesting use cases: an example are Internet of Things (IoT) type settings with low-energy devices, where a public-key system would be considered too energy inefficient.

Another interesting application for the work is PSK-mode in TLS 1.3. In PSK-mode devices can be pre-loaded with a server communication key, but it is also used to agree on a 'next-session key' between a client and server after they performed the full authentication handshake the first time. Applying a SAKE / PSK-AKE protocol to this TLS mode is a relevant follow-up work to the paper introduced above.

2.4 Concluding remarks

The works above have improved our understanding of post-quantum cryptography. Here, we give some concrete suggestions for closely related future work, and conclude by briefly elaborating on how the theoretical results of this work should be seen in the larger context of obtaining more secure cryptographic systems for the real world.

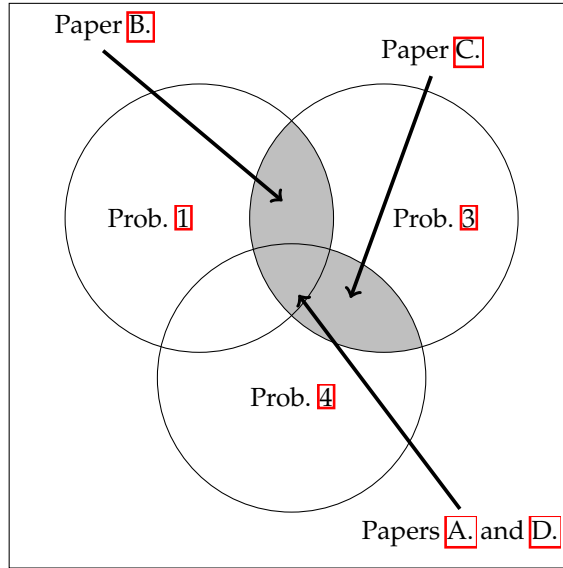


Figure 2.1: Relating the individual works to the various open problems.

2.4.1 Comparison to open problems in the field

In Section 1.2 we gave an overview of some open problems in our field. The contributions listed in this dissertation are all focused on the construction of new cryptographic schemes, but do not relate to the objectives in the same way. In Figure 2.1 we compare how the various papers stack up to the open questions as they were defined in the first chapter. At a first glance, we see that the papers are in a sense closely related — they all live in the intersection between two or three of the main questions: but in this chapter we have seen that the variation between the works is larger than at first appearance. For instance, Paper A. focuses on tight security to make better parameter choices (goal 4) possible — while Paper D. uses parameter choices in order to make new security properties (goal 3d) achievable. Where Paper A. uses isogenies, Paper D. uses lattices, and where A. uses a tight proof to convince the reader of its efficiency, Paper D. uses estimations and an implementation.

There are several open problems that did not end up being addressed in this project. In the list in Chapter 1, we for instance included cryptanalysis (Problem 2) and briefly mentioned constructions related to group key exchange and passwords (Problem 3c). Although these topics did not end up within the scope of

this thesis, research into them is important: there simply was not enough time to do it all.

2.4.2 Future work

There is still a lot to be done: as remarked at the start of this chapter, the papers in this dissertation contribute to the total sum of key exchange knowledge that we have, but none of them proposes a definitive solution for this problem.

The following open questions are perhaps the most natural ones, taking our contributions as the starting point:

- Our NIKE from Paper [D.] is provided with an implementation of only the passively secure version. As pointed out in the paper, an implementation of the NIZK proofs [LNP22] that we use to lift the protocol from passive to active security does currently not exist. Creating this actively secure version strengthens the case for our scheme (and the analysis of its efficiently).
- Papers [A.] and [B.] do not come with an implementation. For Paper [A.], a proof-of-concept implementation like the original CSIDH paper provides [CLM⁺18] would be helpful to say more about the real-world practicality of the scheme — although the caveats concerning isogeny-based schemes mentioned above certainly apply.
- The results of Paper [B.] are currently mostly focused towards small-scale internet of things devices, while there is certainly relevance for more generic application in TLS 1.3. The complexity of the TLS key schedule makes it challenging to simply “plug in” a proof-of-concept implementation as a non-expert. Collaborating with authors who have this expertise would be an interesting follow-up point.
- Perhaps the most theoretical work is Paper [C.] It provides several new protocols, that should be instantiated (for instance with concrete protocols from the NIST competition), implemented and compared with other works in the literature.

Additionally, there are many avenues we did not explore in this project. The potential of code-based cryptography is large, with several strong candidates in the NIST competition. Analyzing those proposals and comparing them with our results is a potential follow-up.

2.4.3 Regarding real-world security

An interesting discussion point with all of the results in this work, and with cryptographic research in general, is how theoretical results translate to real-world security. For Papers [\[B.\]](#) and [\[D.\]](#), this translation is perhaps the most obvious: they both provide a very specific solution to a specific problem. For Paper [\[C.\]](#) the distance to a real-world application is definitively the largest, as the contribution in this work will first of all *enable* the design of new protocols — if and when that happens, these will have to go through the normal kinds of scrutiny and will additionally have to be compared to existing protocols out there. For Paper [\[A.\]](#) the question of real-world security has perhaps the most paradoxical answer: the tight proof means that we understand the real-world security of the protocol very well, but only in relation to the security of CSIDH itself. What kinds of attacks and developments will appear in that direction is unclear — and so is the future of isogeny-based cryptography in general.

Having said that: cryptography remains one of perhaps few fields where highly theoretical mathematics is used to solve a very concrete, societal problem, and where the theoretical results we obtain can be used in practice within a time-frame of sometimes only months. Not only in a post-quantum context, we see that cryptographers regularly spark new life into decades or even centuries old problems and theories, and find use cases for obscure mathematics that no one ever thought would have a real-world application. In addition to this paradox, there is also the question of what cryptographic security really means. We have seen in the case of SIKE that the understanding we have of our underlying primitives is not always as good as we believe it to be — but sometimes the cryptographic problem isn't really cryptography-related either. An example of a large cryptographic vulnerability discovered in 2015 was the Logjam attack, which affected a significant part of all internet traffic: partially because of a new attack strategy, but mostly because of bad implementation choices [\[ABD⁺15\]](#).

Bibliography

- [AAB⁺22] Carlos Aguilar Melchor, Nicolas Aragon, Slim Bettaieb, Loïc Bidoux, Olivier Blazy, Jean-Christophe Deneuville, Philippe Gaborit, Edoardo Persichetti, Gilles Zémor, Jurjen Bos, Arnaud Dion, Jerome Lacan, Jean-Marc Robert, and Pascal Veron. HQC. Technical report, National Institute of Standards and Technology, 2022. available at <https://csrc.nist.gov/Projects/post-quantum-cryptography/round-4-submissions>.
- [ABB⁺22] Nicolas Aragon, Paulo Barreto, Slim Bettaieb, Loïc Bidoux, Olivier Blazy, Jean-Christophe Deneuville, Phillippe Gaborit, Shay Gueron, Tim Guneysu, Carlos Aguilar Melchor, Rafael Misoczki, Edoardo Persichetti, Nicolas Sendrier, Jean-Pierre Tillich, Gilles Zémor, Valentin Vasseur, Santosh Ghosh, and Jan Richter-Brokmann. BIKE. Technical report, National Institute of Standards and Technology, 2022. available at <https://csrc.nist.gov/Projects/post-quantum-cryptography/round-4-submissions>.
- [ABC⁺22] Martin R. Albrecht, Daniel J. Bernstein, Tung Chou, Carlos Cid, Jan Gilcher, Tanja Lange, Varun Maram, Ingo von Maurich, Rafael Misoczki, Ruben Niederhagen, Kenneth G. Paterson, Edoardo Persichetti, Christiane Peters, Peter Schwabe, Nicolas Sendrier, Jakub Szefer, Cen Jung Tjhai, Martin Tomlinson, and Wen Wang. Classic McEliece. Technical report, National Institute of Standards and Technology, 2022. available at <https://csrc.nist.gov/projects/post-quantum-cryptography/round-4-submissions>.
- [ABD⁺15] David Adrian, Karthikeyan Bhargavan, Zakir Durumeric, Pierriek Gaudry, Matthew Green, J. Alex Halderman, Nadia Heninger, Drew Springall, Emmanuel Thomé, Luke Valenta, Benjamin VanderSloot, Eric Wustrow, Santiago Zanella-Béguelin, and Paul Zim-

- mermann. Imperfect forward secrecy: How Diffie-Hellman fails in practice. In Indrajit Ray, Ninghui Li, and Christopher Kruegel, editors, *ACM CCS 2015: 22nd Conference on Computer and Communications Security*, pages 5–17, Denver, CO, USA, October 12–16, 2015. ACM Press.
- [ACF20] Gildas Avoine, Sébastien Canard, and Loïc Ferreira. Symmetric-key authenticated key exchange (SAKE) with perfect forward secrecy. In Stanislaw Jarecki, editor, *Topics in Cryptology – CT-RSA 2020*, volume 12006 of *Lecture Notes in Computer Science*, pages 199–224, San Francisco, CA, USA, February 24–28, 2020. Springer, Heidelberg, Germany.
- [ADPS16a] Erdem Alkim, Léo Ducas, Thomas Pöppelmann, and Peter Schwabe. Post-quantum key exchange - A new hope. In Thorsten Holz and Stefan Savage, editors, *USENIX Security 2016: 25th USENIX Security Symposium*, pages 327–343, Austin, TX, USA, August 10–12, 2016. USENIX Association.
- [ADPS16b] Erdem Alkim, Léo Ducas, Thomas Pöppelmann, and Peter Schwabe. Post-quantum Key Exchange-A New Hope. In *USENIX Security Symposium*, pages 327–343, 2016.
- [BDdK⁺21] Colin Boyd, Gareth T. Davies, Bor de Kock, Kai Gellert, Tibor Jager, and Lise Millerjord. Symmetric key exchange with full forward security and robust synchronization. In Mehdi Tibouchi and Huaxiong Wang, editors, *Advances in Cryptology – ASIACRYPT 2021, Part IV*, volume 13093 of *Lecture Notes in Computer Science*, pages 681–710, Singapore, December 6–10, 2021. Springer, Heidelberg, Germany.
- [BdKM23] Colin Boyd, Bor de Kock, and Lise Millerjord. Modular Design of KEM-Based Authenticated Key Exchange. *Cryptology ePrint Archive*, Paper 2023/167, 2023. <https://eprint.iacr.org/2023/167>.
- [Ber06] Daniel J. Bernstein. Curve25519: New Diffie-Hellman speed records. In Moti Yung, Yevgeniy Dodis, Aggelos Kiayias, and Tal Malkin, editors, *PKC 2006: 9th International Conference on Theory and Practice of Public Key Cryptography*, volume 3958 of *Lecture Notes in Computer Science*, pages 207–228, New York, NY, USA, April 24–26, 2006. Springer, Heidelberg, Germany.
- [BL17] Daniel J Bernstein and Tanja Lange. Post-quantum cryptography. *Nature*, 549(7671):188–194, 2017.

- [CD22] Wouter Castryck and Thomas Decru. An efficient key recovery attack on SIDH (preliminary version). *Cryptology ePrint Archive*, Report 2022/975, 2022. <https://eprint.iacr.org/2022/975>.
- [CLM⁺18] Wouter Castryck, Tanja Lange, Chloe Martindale, Lorenz Panny, and Joost Renes. CSIDH: An efficient post-quantum commutative group action. In Thomas Peyrin and Steven Galbraith, editors, *Advances in Cryptology – ASIACRYPT 2018, Part III*, volume 11274 of *Lecture Notes in Computer Science*, pages 395–427, Brisbane, Queensland, Australia, December 2–6, 2018. Springer, Heidelberg, Germany.
- [DH76] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.
- [DHK⁺22] Julien Duman, Dominik Hartmann, Eike Kiltz, Sabrina Kunzweiler, Jonas Lehmann, and Doreen Riepel. Group action key encapsulation and non-interactive key exchange in the QROM. In Shweta Agrawal and Dongdai Lin, editors, *Advances in Cryptology – ASIACRYPT 2022, Part II*, volume 13792 of *Lecture Notes in Computer Science*, pages 36–66, Taipei, Taiwan, December 5–9, 2022. Springer, Heidelberg, Germany.
- [dK18] Bor de Kock. A non-interactive key exchange based on ring-learning with errors. Master’s thesis, Eindhoven University of Technology, 2018.
- [dKGV20] Bor de Kock, Kristian Gjøsteen, and Mattia Veroni. Practical isogeny-based key-exchange with optimal tightness. In Orr Dunkelman, Michael J. Jacobson Jr., and Colin O’Flynn, editors, *SAC 2020: 27th Annual International Workshop on Selected Areas in Cryptography*, volume 12804 of *Lecture Notes in Computer Science*, pages 451–479, Halifax, NS, Canada (Virtual Event), October 21–23, 2020. Springer, Heidelberg, Germany.
- [DS04] Jintai Ding and Dieter Schmidt. Multivariable public-key cryptosystems. *Cryptology ePrint Archive*, Report 2004/350, 2004. <https://eprint.iacr.org/2004/350>.
- [EOS06] D. Engelbert, R. Overbeck, and A. Schmidt. A summary of McEliece-type cryptosystems and their security. *Cryptology ePrint Archive*, Report 2006/162, 2006. <https://eprint.iacr.org/2006/162>.

- [GdKQ⁺23] Phillip Gajland, Bor de Kock, Miguel Quaresma, Giulio Malavolta, and Peter Schwabe. Swoosh: Practical Lattice-Based Non-Interactive Key Exchange. *Cryptology ePrint Archive*, Paper 2023/271, 2023. <https://eprint.iacr.org/2023/271>.
- [GKRS20] Siyao Guo, Pritish Kamath, Alon Rosen, and Katerina Sotiraki. Limits on the efficiency of (ring) LWE based non-interactive key exchange. In Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas, editors, *PKC 2020: 23rd International Conference on Theory and Practice of Public Key Cryptography, Part I*, volume 12110 of *Lecture Notes in Computer Science*, pages 374–395, Edinburgh, UK, May 4–7, 2020. Springer, Heidelberg, Germany.
- [Gro96] Lov K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing, STOC '96*, page 212–219, New York, NY, USA, 1996. Association for Computing Machinery.
- [JAC⁺22] David Jao, Reza Azarderakhsh, Matthew Campagna, Craig Costello, Luca De Feo, Basil Hess, Amir Jalali, Brian Koziel, Brian LaMacchia, Patrick Longa, Michael Naehrig, Joost Renes, Vladimir Soukharev, David Urbanik, Geovandro Pereira, Koray Karabina, and Aaron Hutchinson. SIKE. Technical report, National Institute of Standards and Technology, 2022. available at <https://csrc.nist.gov/Projects/post-quantum-cryptography/round-4-submissions>.
- [JDF11] David Jao and Luca De Feo. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In *International Workshop on Post-Quantum Cryptography*, pages 19–34. Springer, 2011.
- [Kim16] Kevin Kimball. Announcing Request for Nominations for Public-Key Post-Quantum Cryptographic Algorithms. <https://www.federalregister.gov/documents/2016/12/20/2016-30615/announcing-request-for-nominations-for-public-key-post-quantum-cryptographic-algorithms>, December 2016.
- [KL14] Jonathan Katz and Yehuda Lindell. *Introduction to Modern Cryptography, Second Edition*. CRC Press, 2014.
- [Koc96] Paul C. Kocher. Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In Neal Koblitz, editor, *Advances in Cryptology – CRYPTO'96*, volume 1109 of *Lecture Notes*

- in Computer Science*, pages 104–113, Santa Barbara, CA, USA, August 18–22, 1996. Springer, Heidelberg, Germany.
- [KPG99] Aviad Kipnis, Jacques Patarin, and Louis Goubin. Unbalanced Oil and Vinegar signature schemes. In Jacques Stern, editor, *Advances in Cryptology – EUROCRYPT’99*, volume 1592 of *Lecture Notes in Computer Science*, pages 206–222, Prague, Czech Republic, May 2–6, 1999. Springer, Heidelberg, Germany.
- [KTAT20] Tomoki Kawashima, Katsuyuki Takashima, Yusuke Aikawa, and Tsuyoshi Takagi. An efficient authenticated key exchange from random self-reducibility on CSIDH. In Deukjo Hong, editor, *ICISC 20: 23rd International Conference on Information Security and Cryptology*, volume 12593 of *Lecture Notes in Computer Science*, pages 58–84, Seoul, Korea, December 2–4, 2020. Springer, Heidelberg, Germany.
- [Lan16a] Adam Langley. CECPQ1 results. <https://www.imperialviolet.org/2016/11/28/cecpq1.html>, November 2016.
- [Lan16b] Adam Langley. Intent to Implement and Ship: CECPQ1 for TLS. <https://groups.google.com/a/chromium.org/forum/#!topic/security-dev/DS9pp2U0SAc>, July 2016.
- [LNP22] Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Maxime Plançon. Lattice-based zero-knowledge proofs and applications: Shorter, simpler, and more general. In Yevgeniy Dodis and Thomas Shrimpton, editors, *Advances in Cryptology – CRYPTO 2022, Part II*, volume 13508 of *Lecture Notes in Computer Science*, pages 71–101, Santa Barbara, CA, USA, August 15–18, 2022. Springer, Heidelberg, Germany.
- [LP11] Richard Lindner and Chris Peikert. Better Key Sizes (and Attacks) for LWE-Based Encryption. In Aggelos Kiayias, editor, *Topics in Cryptology - CT-RSA 2011 - The Cryptographers’ Track at the RSA Conference 2011, San Francisco, CA, USA, February 14-18, 2011. Proceedings*, volume 6558 of *Lecture Notes in Computer Science*, pages 319–339. Springer, 2011.
- [Lyu17] Vadim Lyubashevsky. Converting NewHope/LWE key exchange to a Diffe-Hellman-like algorithm. Crypto Stack Exchange, 2017. [Online:] <https://crypto.stackexchange.com/questions/48146/converting-newhope-lwe-key-exchange-to-a-diffe-hellman-like-algorithm>.

- [O'N21] Patrick Howell O'Neill. The US is worried that hackers are stealing data today so quantum computers can crack it in a decade. *Technology Review*, 2021.
- [Pei20] Chris Peikert. He gives C-sieves on the CSIDH. In Anne Canteaut and Yuval Ishai, editors, *Advances in Cryptology – EUROCRYPT 2020, Part II*, volume 12106 of *Lecture Notes in Computer Science*, pages 463–492, Zagreb, Croatia, May 10–14, 2020. Springer, Heidelberg, Germany.
- [SAB⁺20] Peter Schwabe, Roberto Avanzi, Joppe Bos, Léo Ducas, Eike Kiltz, Tancrede Lepoint, Vadim Lyubashevsky, John M. Schanck, Gregor Seiler, and Damien Stehlé. CRYSTALS-KYBER. Technical report, National Institute of Standards and Technology, 2020. available at <https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-3-submissions>.
- [Sho94] P.W. Shor. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th Annual Symposium on Foundations of Computer Science*, pages 124–134, 1994.
- [vV18] Christine van Vredendaal. *Exploiting mathematical structures in cryptography*. PhD thesis, Mathematics and Computer Science, June 2018. Proefschrift.

Appendix A

Practical Isogeny-Based Key-Exchange with Optimal Tightness

Practical Isogeny-Based Key-exchange with Optimal Tightness

Bor de Kock, Kristian Gjøsteen, and Mattia Veroni

NTNU – Norwegian University of Science and Technology, Trondheim, Norway.

`{bor.dekock,kristian.gjosteen,mattia.veroni}@ntnu.no`

Abstract. We exploit the Diffie-Hellman-like structure of CSIDH to build a quantum-resistant authenticated key-exchange algorithm. Our security proof has optimal tightness, which means that the protocol is efficient even when instantiated with theoretically-sound security parameters. Compared to previous isogeny-based authenticated key-exchange protocols, our scheme is extremely simple, its security relies only on the underlying CSIDH-problem and it has optimal communication complexity for CSIDH-based protocols. Our security proof relies heavily on the re-randomizability of CSIDH-like problems and carries on in the ROM.

Keywords: Post-quantum, isogenies, key-exchange, provable-security, tightness, re-randomization.

1 Introduction

Authenticated key-exchange protocols allow two parties to collaborate in order to create a shared secret key, providing each of them with some assurance on the identity of the partner. Authentication can be achieved in two ways: *implicitly*, if the algebraic properties of the scheme imply that the only user who can compute the shared key is the intended one, or *explicitly*, by receiving a confirmation that the interlocutor has actually computed the key. The latter implies the use of a second mechanism which provides authentication, like a signature scheme, a KEM or a MAC. Even if explicit authentication might seem a stronger and preferable feature, in the real world it does not add much to the security of the protocol. First of all, it does not guarantee that the partner holds the shared key for all the time between the key confirmation and the use of the key. Moreover, the generation of signatures or the use of KEMs and MACs produces evidence of participation to a key-exchange, while implicit authentication does not. Finally, the schemes relying on implicit authentication typically require less computations and message exchanges compared to those involving an explicit authentication mechanism, with a significant profit in computational cost and communication efficiency.

The security proof limits the advantage of an adversary in breaking the scheme to the probability of solving some mathematical hard problem. Deploying a cryptographic algorithm should always be done in a *theoretically sound* way: the size of the concrete parameters must be large enough to guarantee the required λ bits of security. If on one hand any security proof asymptotically guarantees the desired security level, on the other hand we want to use the smallest parameters possible, in order to obtain the most efficient implementation under the given security constraints. It is therefore extremely relevant to measure the so-called *tightness* of the proof by computing its security loss $L(\lambda)$, which should be as small as possible. The parameters on which we focus are, in particular, the number of users running the protocol and the number of sessions per user; both quantities are typically approximated to 2^{16} . Note that, nowadays, security proofs [JKSS12, KPW13, BFK⁺14] for a widely deployed protocol such as TLS have a quadratic loss in the number of sessions, fact that is not taken into account for the implementation.

In 2019 Cohn-Gordon et al. [CCG⁺19] developed a key-exchange protocol with an nearly (but optimally) tight security proof. In particular, the security loss is linear in the number of users and constant in the number of sessions per user. The schemes in the latter paper base their security on the Strong-DH assumption and its variants, defined over cyclic groups of prime order. The re-randomization of Diffie-Hellman problems plays a fundamental role in achieving the optimal tightness of the proofs, and thus it is a desirable feature that we cannot disregard. The tightness and practicality of these schemes raise an interesting question: is it possible to adapt the protocols (together with their security proofs) in order to make them quantum-safe?

In 1997, Peter Shor [Sho97] published a quantum algorithm for integer factorization and one for computing discrete logarithms, both running in polynomial time. As soon as a large-scale quantum computer will become available, the information security based on primitives like the RSA cryptosystem and the Diffie-Hellman key-exchange will be breached. In order to address this quantum threat, many researchers have focused their attention on post-quantum cryptography. The goal is to find new cryptographic primitives which can be implemented on classical computers, still guaranteeing security against both classical and quantum adversaries. In 2016, NIST announced a world-wide competition for new post-quantum standards in public-key encryption and digital signature algorithms. 69 submissions were accepted in the first round, 26 made it to the second step, and 7 finalists were announced on July 22, 2020. The search for new post-quantum cryptographic standards is still ongoing.

Supersingular-Isogeny based Diffie-Hellman (SIDH) [JD11] is one of the promising candidates in the search for post-quantum cryptographic protocols. Key-exchange protocols based on isogenies are unique in the sense that they pro-

vide key-sizes roughly similar to those of pre-quantum alternatives, but they are also known for being more complex (algebraically) compared to some of the post-quantum alternatives. An example of a scheme that is based on SIDH is SIKE [JAC⁺19], which is one of the 26 candidates in the second round of NIST’s 2016 competition for post-quantum cryptographic protocols. Even if SIKE is not among the finalists announced in July 2020, NIST has shown high interest on isogeny-based cryptography, encouraging further research on this field [AASA⁺].

Although SIDH-based schemes have been around for a few years now, there are still open questions about the security behind them. In particular, random self-reducibility of SIDH problems seems very hard to achieve. A different isogeny-based scheme is CSIDH [CLM⁺18]: introduced in 2018, it offers a much more flexible and adaptable algebraic structure. In this paper we show how to obtain an optimally tight security proof for a CSIDH-based key-exchange protocol, making use of random self-reducibility. This kind of re-randomization plays a fundamental role in the tight proofs of, for instance, the classical Diffie-Hellman key-exchange, but is also used in modern schemes: Cohn-Gordon et al. [CCG⁺19] exploit this property to construct a tightly-secure AKE protocol.

The protocol we introduce is, to our knowledge, the best proven-secure result for isogeny-based key-exchange protocols. The proofs presented here draw on the proofs from Cohn-Gordon et al. [CCG⁺19], but with changes to the re-randomization strategy, since re-randomization in the isogeny case is different from the one in the cyclic group case. Both efficiency and tightness are a significant improvement over the state of the art, and can lead to the deployment of schemes with more efficient parameter choices obtaining high security at computational costs which are as low as possible.

1.1 Our contributions

In section 3.2 we adapt protocol *II* by Cohn-Gordon et al. [CCG⁺19] to the isogeny setting, obtaining the first implicitly authenticated CSIDH-like protocol with weak forward secrecy, under only the Strong-CSIDH assumption. This is the first scheme with a security proof (moreover with optimal tightness) in the same setting as CSIDH. The protocol requires each user to perform 4 ideal-class evaluations, and its security proof, shown in Appendix B, has a tightness loss which is linear in the number of sessions performed by a single user.

The adaptation we perform is, however, not entirely straightforward. In the new setting we have only one operation, namely the multiplication of ideal classes, while in the original protocol re-randomization is achieved via two operations (addition and multiplication of exponents). This leads to a different re-randomization technique which relies on the random self-reducibility of the computational CSIDH problem shown in appendix 4.1.

We obtain a significant improvement over the state of the art of isogeny-based key-exchange protocols. Compared to one of the latest scheme, from “Strongly Secure Authenticated Key Exchange from Supersingular Isogenies” [XXW⁺19], we obtain better efficiency and tightness. Moreover, unlike this latter scheme, our protocol does not require any authentication mechanism. This allows us to rely on the same class (and a smaller number) of hardness assumptions, and to avoid the use of signatures, which are tricky and expensive [DG19] to produce in the isogeny setting. Compared to the CSIDH protocol, which lacks a security proof and for which authentication seems hard to achieve, our *II*-SIDE protocol has implicit authentication at the cost of a few more ideal-class evaluations. As shown in section 6, our *II*-SIDE protocol is competitive with other post-quantum candidates, once instantiated with theoretically-sound parameters.

1.2 Related work

In the last years, a lot of research has been conducted on SIDH-based schemes. For example, Galbraith [Gal18] has shown how to adapt generic constructions to the SIDH setting, and he introduced two new SIDH-AKE protocols. Similar results were achieved by Longa [Lon18], except for the introduction of the two new schemes. Assuming a straightforward adaptation, a few other protocols have a non-quadratic tightness loss. For example KEA+ [LM06] has a linear loss in the number of participants multiplied by the number of sessions, assuming the hardness of the Gap-DH problem. Although, it does not achieve wPFS and takes $O(t \log t)$ time only when instantiated on pairing-friendly curves.

In their recent paper, Xu et al. [XXW⁺19] propose SIAKE₂ and SIAKE₃, a two-pass and a three-pass AKE respectively. SIAKE₂, whose security relies on the decisional SIDH assumption, has a rather convoluted construction: they design a strong One-Way CPA secure PKE scheme, which is then turned into a One-Way CCA KEM through the modified FO-transform and finally used as a building block for the AKE scheme. The three-pass AKE SIAKE₃ is obtained by modifying the previously designed KEM, once a new assumption (the 1-Oracle SI-DH, an analogue of the Oracle Diffie-Hellman assumption in which only one query is allowed) is made. Compared to this scheme, our result is simpler and it has a tighter security proof, smaller communication complexity and improved overall efficiency.

2 Preliminaries

In this section, we first recall the definition of tightness for security reductions. Then we provide the reader with key-concepts and results which are indispensable to understand the constructions of SIDH and CSIDH. Good references regarding

elliptic curves and isogenies are Silverman [HS09], Washington [Was08] and De Feo [Feo17]; the original papers introducing SIDH and CSIDH are Jao-De Feo [JD11] and Castryck et al. [CLM⁺18], respectively.

2.1 Tight reductions

When comparing schemes, one should always consider protocols once they have been instantiated with theoretically-sound parameters, which guarantee the desired level of security. These parameters (such as the bit-length of the prime defining a base field or the key size) strongly depend on the security proof correlated with the protocol. A security proof usually consists of

- a security model, in which we describe an adversary by listing a set of queries that it can make (and therefore specifying what it is allowed to do);
- a sequence of games leading to a *reduction*, in which an adversary \mathcal{A} against the protocol is turned into a solver \mathcal{B} for an allegedly hard problem.

The “quality” of a reduction can be measured by computing its security loss: if $t_{\mathcal{A}}$ and $\epsilon_{\mathcal{A}}$ are the running time and the success probability of \mathcal{A} respectively, and $t_{\mathcal{B}}$ and $\epsilon_{\mathcal{B}}$ respectively are the running time and the success probability of \mathcal{B} , then we define the *security loss* L as

$$\frac{t_{\mathcal{A}}}{\epsilon_{\mathcal{A}}} = L \frac{t_{\mathcal{B}}}{\epsilon_{\mathcal{B}}}. \quad (1)$$

If L is constant, then we say that the reduction is *tight*. Having a tight proof is as relevant as building an efficient protocol, because this leads to deploy the smallest possible parameters when concretely instantiating a protocol.

In some cases, however, it is impossible to obtain a tight reduction. In a *simple scheme* the adversary is run only once, in comparison to other protocols which use the Forking Lemma in order to run multiple copies of the adversary. A linear loss in the number of participants to the protocol is unavoidable for simple schemes, while applying the Forking Lemma leads to a non-tight proof. We therefore focus on *optimal tightness* whenever tightness is unachievable: the L in Equation (1) turns out to be not constant, but one proves that it is impossible to decrease its order. We rely on the same strategies adopted in the paper by Cohn-Gordon et al. [CCG⁺19] to prove the lower bound on the tightness loss, applying their variant of the meta-reduction techniques by Bader et al. [BJLS16].

Many available schemes, which are actually taken into account for standardization processes, have quite non-tight security reductions. Let μ be the number of users running the protocol and let k be the number of sessions per user. HMQV [Kra05], a classically secure protocol in the random-oracle model under the CDH

assumption, has security loss $O(\mu^2 k^2)$. If we consider a generic signed KEM approach, we get a $O(\mu^2 k^2)$ loss in addition to the signature scheme loss. In many cases, parameters are chosen in a non theoretically-sound way, while tightness loss should always be considered when comparing protocols.

2.2 Elliptic curves, isogenies and endomorphism rings

Let \mathbb{F}_p be a finite field for a large prime p and let E be an elliptic curve over \mathbb{F}_p . We say that E is *supersingular* if and only if it has order $\#E(\mathbb{F}_p) = p + 1$. Consider the isomorphisms of elliptic curves, i.e. all the invertible algebraic maps. Any two elliptic curves over the algebraic closure $\overline{\mathbb{F}_p}$ are *isomorphic* if and only if they have the same j -invariant. Thus we can use isomorphisms to define an equivalence relation between elliptic curves and identify an equivalence class by the j -invariant of the curves in the class.

Let E_1 and E_2 be two elliptic curves defined over \mathbb{F}_p and let $0_{E_1}, 0_{E_2}$ denote the respective points at infinity. An *isogeny* from E_1 to E_2 is a morphism $\phi : E_1 \rightarrow E_2$ such that $\phi(0_{E_1}) = 0_{E_2}$. For any isogeny $\phi : E_1 \rightarrow E_2$ there exists a *dual isogeny* $\hat{\phi} : E_2 \rightarrow E_1$ such that $\hat{\phi} \circ \phi = [\deg(\phi)]_{E_1}$ and $\phi \circ \hat{\phi} = [\deg(\phi)]_{E_2}$. An isogeny is essentially determined by its kernel: given a finite subgroup $G \subset E(\overline{\mathbb{F}_p})$ there exist a unique (up to isomorphisms) elliptic curve $E_2 \simeq E_1/G$ and a separable isogeny $\phi : E_1 \rightarrow E_2$ such that $\ker(\phi) = G$. The isogeny ϕ has *degree* ℓ equal to the cardinality of its kernel, and we call it an ℓ -isogeny. Given the kernel of an isogeny, we can exploit Vélu's formulae [Vél71] to compute the isogeny ϕ together with the codomain curve E_2 in $O(\ell \log(p)^2)$ bit operations. This is the best approach when ℓ is small enough and p is shorter than a few thousand bits. Any separable isogeny defined over \mathbb{F}_p can be written as the composition of isogenies of prime degrees.

An *endomorphism* is an isogeny from E to itself; the set of endomorphisms of E , together with the zero map and equipped with pointwise addition and composition, forms the *endomorphism ring* $End(E)$. We denote by $End_p(E)$ the ring of endomorphisms defined over \mathbb{F}_p . For ordinary curves $End_p(E) = End(E)$, while for supersingular curves $End_p(E) \subset End(E)$. In particular, $End(E)$ is an order in a quaternion algebra, whilst $End_p(E)$ is an order in the imaginary quadratic field $\mathbb{Q}(\sqrt{p})$. A classical result by Deuring [Deu41] reveals that $End(E)$ is a maximal order in $B_{p,\infty}$, the quaternion algebra ramified at p and at ∞ .

2.3 The ideal class group action

We hereafter provide the reader with the basic definitions and known results regarding ideal class group action. In particular, this section gravitates around a recurring sentence in isogeny-based cryptography:

“The ideal class group of an imaginary quadratic order \mathcal{O} acts freely via isogenies on the set of elliptic curves with $\text{End}_p(E) \simeq \mathcal{O}$.”

We will then focus on the computational aspects, essential to understand CSIDH.

Algebraic foundations. An *algebra* A is a vector space over a field \mathbb{K} equipped with a bilinear operation. If the bilinear operation is associative, then we say that A is an associative algebra. Given a unitary ring R , a left R -module ${}_R M$ consists of an abelian group $(M, +)$ and a scalar multiplication $R \times_R M \rightarrow_R M$ which satisfies left/right distributivity, associativity and neutrality of ring’s unit. Let R be an integral domain (a commutative unitary ring without zero-divisors) and let \mathbb{K} be its field of fractions; a left R -module ${}_R M$ is a *lattice* in the vector space V over \mathbb{K} if ${}_R M$ is finitely generated, R -torsion free and an R -submodule of V . An *order* is a subring \mathcal{O} of a ring A such that 1) A is a finite dimensional algebra over \mathbb{Q} , 2) \mathcal{O} spans A over \mathbb{Q} (i.e. $\mathbb{Q}\mathcal{O} = A$), 3) \mathcal{O} is an integer lattice in A .

The ideal class group. Let \mathbb{K} be a finite extension of \mathbb{Q} of degree 2, which is called a *quadratic number field*, and let $\mathcal{O} \subseteq \mathbb{K}$ be an order. The *norm* of an \mathcal{O} -ideal $\mathfrak{a} \subseteq \mathcal{O}$ is defined as $N(\mathfrak{a}) = |\mathcal{O}/\mathfrak{a}|$, which is equal to $\gcd(\{N(\alpha) \mid \alpha \in \mathfrak{a}\})$. Norms are multiplicative: $N(\mathfrak{a}\mathfrak{b}) = N(\mathfrak{a})N(\mathfrak{b})$. A *fractional ideal* of \mathcal{O} is an \mathcal{O} -submodule of \mathbb{K} of the form $\alpha\mathfrak{a}$, where $\alpha \in \mathbb{K}^*$ and \mathfrak{a} is an \mathcal{O} -ideal. Fractional ideals can be multiplied and conjugated in the obvious way, and the norm extends multiplicatively to fractional ideals. A fractional \mathcal{O} -ideal is *invertible* if there exists a fractional \mathcal{O} -ideal \mathfrak{b} such that $\mathfrak{a}\mathfrak{b} = \mathcal{O}$. If such \mathfrak{b} exists, we denote $\mathfrak{a}^{-1} = \mathfrak{b}$. All the principal fractional ideals $\alpha\mathcal{O}$ where $\alpha \in \mathbb{K}^*$ are invertible.

The *ideal class group* of \mathcal{O} , defined as $cl(\mathcal{O}) := I(\mathcal{O})/P(\mathcal{O})$, is the quotient of the set of invertible fractional ideals $I(\mathcal{O})$ by the set of principal invertible fractional ideals $P(\mathcal{O})$: For any $M \in \mathbb{Z} \setminus \{0\}$, every ideal class $[\mathfrak{a}]$ has an integral representative of norm coprime to M . There is a unique *maximal order* of \mathbb{K} with respect to inclusion, which is called the *ring of integers* and is denoted by $\mathcal{O}_{\mathbb{K}}$. The *conductor* of \mathcal{O} in $\mathcal{O}_{\mathbb{K}}$ is the index $f = [\mathcal{O}_{\mathbb{K}}/\mathcal{O}]$. Every \mathcal{O} -ideal of norm coprime to the conductor is invertible and factors uniquely into prime ideals.

The class group action. Let $\mathcal{E}ll_p(\mathcal{O})$ be the set of supersingular elliptic curves over \mathbb{F}_p with $\text{End}_p(E)$ isomorphic to an order \mathcal{O} in an imaginary quadratic field and let $E \in \mathcal{E}ll_p(\mathcal{O})$. Given an \mathcal{O} -ideal \mathfrak{a} , we define the *action* of \mathfrak{a} on E as follows:

1. we consider all the endomorphisms α in \mathfrak{a} ,
2. we compute the \mathfrak{a} -torsion subgroup $E[\mathfrak{a}] = \bigcap_{\alpha \in \mathfrak{a}} \ker(\alpha) = \{P \in E(\overline{\mathbb{F}}_p) : \alpha P = 0_E \forall \alpha \in \mathfrak{a}\}$,

3. we compute the isogeny $\phi_{\mathfrak{a}} : E \rightarrow E_{\mathfrak{a}} \simeq E/E[\mathfrak{a}]$.

It is common practice to denote the action of \mathfrak{a} on E by $\mathfrak{a} * E$.

A fundamental result in isogeny-based protocols is the *Deuring correspondence* between the set of maximal orders in $B_{p,\infty}$ and the set of elliptic curves: fixing a supersingular elliptic curve E_0 , every ℓ -isogeny $\alpha : E_0 \rightarrow E$ corresponds to an ideal \mathfrak{a} of norm ℓ , and vice-versa. Since $E_{\mathfrak{a}}$ is determined (up to isomorphism) by the ideal class of \mathfrak{a} , finding different representatives of an ideal class corresponds to finding different isogenies between two fixed curves.

We can rewrite any ideal \mathfrak{a} of \mathcal{O} as the product of \mathcal{O} -ideals $\mathfrak{a} = (\pi_p \mathcal{O})^r \mathfrak{a}_s$, where π_p is the p -th Frobenius endomorphism and $\mathfrak{a}_s \not\subseteq \pi_p \mathcal{O}$. This defines an elliptic curve $\mathfrak{a} * E$ and an isogeny $\phi_{\mathfrak{a}} : E \rightarrow \mathfrak{a} * E$ of degree $N(\mathfrak{a})$ as follows:

- the separable part of $\phi_{\mathfrak{a}}$ has kernel $\bigcap_{\alpha \in \mathfrak{a}_s} \ker(\alpha)$;
- the purely inseparable part consists of r iterations of Frobenius.

The isogeny $\phi_{\mathfrak{a}}$ and the codomain $\mathfrak{a} * E$ are both defined over \mathbb{F}_p and are unique up to \mathbb{F}_p -isomorphism. Directly from this construction it is clear that multiplying ideals and composing isogenies are equivalent operations.

Let $\mathcal{E}ll_p(\mathcal{O}, \pi)$ be the set of elliptic curves defined over \mathbb{F}_p whose endomorphism ring is isomorphic to \mathcal{O} such that the Frobenius endomorphism π_p corresponds to π . As explained by Castryck et al. [CLM⁺18], we get the following fundamental result:

Theorem 1. *Let \mathcal{O} be an order in an imaginary quadratic field and $\pi \in \mathcal{O}$ such that $\mathcal{E}ll_p(\mathcal{O}, \pi)$ is non-empty. Then the ideal class group $cl(\mathcal{O})$ acts freely and transitively on the set $\mathcal{E}ll_p(\mathcal{O}, \pi)$ via the map*

$$\begin{aligned} cl(\mathcal{O}) \times \mathcal{E}ll_p(\mathcal{O}, \pi) &\longrightarrow \mathcal{E}ll_p(\mathcal{O}, \pi) \\ ([\mathfrak{a}], E) &\longrightarrow [\mathfrak{a}] * E. \end{aligned}$$

From now on, we drop the class notation “[\mathfrak{a}]” in favor of a simpler “ \mathfrak{a} ” by considering any integral representative in the class.

The structure of the class group. The class group $cl(\mathcal{O})$ is a finite abelian group whose cardinality is asymptotically $\#cl(\mathcal{O}) \sim \sqrt{|\Delta|}$. As argued by CSIDH’s authors [CLM⁺18], computing the exact structure of the class group requires a lot of computational effort. The best known algorithm (by Hafner and McCurley [HM89]) for computing the structure of the class group is subexponential in Δ , which is typically very large for CSIDH (about the size of p). Therefore, the authors opt for heuristics which allow to find a very good approximation.

We are interested in the primes for which there exist distinct prime ideals $\mathfrak{l}, \bar{\mathfrak{l}}$ of \mathcal{O} such that $\ell \mathcal{O} = \mathfrak{l} \bar{\mathfrak{l}}$. If ℓ is such a prime, we say that it splits in \mathcal{O} ; ℓ is called an

Elkies primes in the point-counting setting. The ideal \mathfrak{l} is generated as $(\ell, \pi - \lambda)$, where $\lambda \in \mathbb{Z}/\ell\mathbb{Z}$ is an eigenvalue of π_p on the ℓ -torsion, and its conjugate is $\bar{\lambda} = (\ell, \pi - \pi/\lambda)$, where p/λ is any integral representative of that quotient modulo ℓ . The prime ℓ splits in \mathcal{O} if and only if Δ is a non-zero square modulo ℓ . The CSIDH protocol is carefully designed such that a long list of primes (74 in the 512-bit implementation) are Elkies primes.

Computing the group action. According to the heuristics which are assumed in CSIDH, any element of the group can be represented as the product of small primes ideals. We can compute $\mathfrak{l} * E$, the action of a prime ideal $\mathfrak{l} = (\ell, \pi - \lambda)$ on E , in three different ways:

- (a) by using the modular polynomials [?]:
 1. find \mathbb{F}_p -rational roots of the modular polynomial $\Phi_\ell(X, j(E))$, which are the j -invariants of the two possible codomains;
 2. compute the kernel polynomials $\chi(x) \in \mathbb{F}_p[x]$ for the corresponding isogenies;
 3. determine which of the options is the correct one by checking if $\pi_p(x, y) = [\lambda](x, y)$ modulo $\chi(x)$ over the curve;
- (b) by using the division polynomials [Was08, XI.3]:
 1. factor the ℓ -th division polynomial $\psi_\ell(E)$ over \mathbb{F}_p ;
 2. match the irreducible factors with the right Frobenius eigenvalues;
 3. use Kohel's formulae to compute the codomain;
- (c) by using Vélu's formulae:
 1. find a basis of the ℓ -torsion points and compute the eigenspaces of π_p ;
 2. apply Vélu's formulae to a basis point of the correct eigenspace to compute the codomain.

In CSIDH, the authors opt for the last method, which is the fastest when the necessary extension fields (in which the basis points lie) are small.

When $\lambda = 1$ the curve has a rational point defined over the base field \mathbb{F}_p . If we also have that $p/\lambda = -1$, the other eigenspace of Frobenius endomorphism modulo ℓ is defined over \mathbb{F}_{p^2} , so both codomains can be easily computed using Vélu's formulae over the base field, switching from a curve to its quadratic twist if necessary. The parameters of the implementation are decided such that $p \equiv -1 \pmod{\ell}$ for many different primes ℓ : in this case, $\lambda = 1$ automatically implies $p/\lambda = -1$.

3 Isogeny-based key-exchange protocols

Isogeny-based cryptography is a class of allegedly quantum-resistant schemes resulting from NIST's competition. Two of the most peculiar features that distinguish them from the other candidates are the use of shorter keys and the

deployment of more sophisticated algebraic structures. In this section, we first provide an overview of CSIDH (pronounced “seaside”) [CLM⁺18], a key-exchange protocol which does not take part in NIST’s competition but is extremely interesting and promising. Then we introduce our new protocol Π -SIDE (pronounced “pie-side”), a translation of the protocol Π [CCG⁺19] in the CSIDH setting.

3.1 CSIDH

What follows is an outline of the CSIDH protocol, whose underlying algebraic structures are briefly explained in section 2.3. We dwell in particular on the aspects which are relevant to our results.

Parameters. Fix a large prime $p = 4 \cdot \ell_1 \cdot \ell_2 \cdots \ell_n - 1$ where ℓ_i are small distinct odd primes. p is designed such that $p \equiv 3 \pmod{4}$, in order to

- easily write down supersingular elliptic curves over \mathbb{F}_p ;
- make use of the Montgomery form of elliptic curves in the implementation.

The starting curve for each execution of the protocol is the supersingular elliptic curve in Montgomery form $E_0 : y^2 = x^3 + x$ over \mathbb{F}_p . In this case the characteristic equation of the Frobenius endomorphism is $\pi_p^2 = -p$, which implies that the \mathbb{F}_p -rational endomorphism ring $\text{End}_p(E_0)$ is an order in the imaginary quadratic field $\mathbb{Q}(\sqrt{-p})$; in particular, $\text{End}_p(E_0) = \mathbb{Z}[\pi]$. The resulting ℓ_i -isogeny graph is a disjoint union of cycles. Moreover, since $\pi^2 - 1 \equiv 0 \pmod{\ell_i}$ for each $i = 1, \dots, n$, the ideals $\ell_i \mathcal{O}$ split as $\ell_i \mathcal{O} = \mathfrak{l}_i \bar{\mathfrak{l}}_i = (\ell_i, \pi - 1)(\ell_i, \pi + 1)$ (so all the ℓ_i are Elkies primes). Furthermore, the kernel of $\phi_{\mathfrak{l}_i}$ is the subgroup generated by a point P of order ℓ_i which lies in the kernel of $\pi - 1$. Analogously, the kernel of $\phi_{\bar{\mathfrak{l}}_i}$ is generated by a point Q of order ℓ_i that is defined over \mathbb{F}_{p^2} but not in \mathbb{F}_p and such that $\pi(Q) = -Q$.

Sampling ideals and computing their action. Although we want to sample uniformly at random from the ideal class group $cl(\mathcal{O})$, it is preferable not to compute its exact structure because of the large size of the discriminant Δ . By heuristically assuming that

- the ideals \mathfrak{l}_i do not have very small order,
- the ideals \mathfrak{l}_i are evenly distributed in the class group,

two ideals $\mathfrak{l}_1^{e_1} \mathfrak{l}_2^{e_2} \cdots \mathfrak{l}_n^{e_n}$ for small e_i will rarely lie in the same class. The e_i are sampled from a short range $\{-m, \dots, m\}$ for some integer m such that $2m + 1 \geq \sqrt[n]{\#cl(\mathcal{O})}$. Since the prime ideals \mathfrak{l}_i are fixed, we represent any ideal $\prod_i \mathfrak{l}_i^{e_i}$ (which will be the user’s secret key) as a vector $(e_1, e_2, \dots, e_n) \in [-m, m]^n$.

Since $\pi^2 \equiv -p \equiv 1 \pmod{\ell_i}$, the eigenvalues of all ℓ_i -torsion subgroups are $+1$ and -1 . This allows us to efficiently compute the action of \mathfrak{l}_i by using method 3. in section 2.3.

Representing and validating \mathbb{F}_p -isomorphism classes. SIDH misses a key-validation protocol, and countermeasures are expensive. We recall how the authors of CSIDH solve the problem for their protocol. First of all, they provide a result [CLM⁺18, Proposition 8]) which states that, for the chosen p and supersingular elliptic curve, the Montgomery coefficient uniquely represents the class of elliptic curves resulting from the evaluation of an ideal. Secondly, to prove that an elliptic curve is supersingular (and thus $\#E(\mathbb{F}_p) = p+1$), it is enough to find a point $Q \in E$ whose order is a divisor of $p+1$ greater than $4\sqrt{p}$ (by Hasse's theorem, we have only one multiple of that divisor in the interval $[p+1-2\sqrt{p}, p+1+2\sqrt{p}]$, which must be the group order by Lagrange's theorem). They therefore provide an algorithm which takes a point at random and computes its order. With high probability (increasing with ℓ_i), this will tell in only one step if the curve is supersingular or not. If x -only Montgomery arithmetic is used, a random point P is obtained by randomly picking $x \in \mathbb{F}_p$, and there is no need to differentiate points in \mathbb{F}_p and in \mathbb{F}_{p^2} (in the second case, the point will correspond to an \mathbb{F}_p -rational point in the quadratic twist, which is supersingular if and only if the original curve is supersingular).

The CSIDH protocol. We first describe how to perform the Setup and the key-generation, then we schematise the simple structure of key-exchange protocol.

Setup. In this phase we set up the global parameters of the key-exchange protocol. In particular, we fix:

- n distinct odd primes ℓ_i , corresponding to n isogeny-degrees;
- a large prime $p = 4 \cdot \ell_1 \cdot \ell_2 \cdots \ell_n - 1$;
- the supersingular elliptic curve $E_0 : y^2 = x^3 + x$ over \mathbb{F}_p with endomorphism ring $\mathcal{O} = \mathbb{Z}[\pi]$.

Key generation. The *private key* is an n -tuple (e_1, \dots, e_n) of integers, randomly sampled from a range $\{-m, \dots, m\}$ such that $2m+1 \geq \sqrt[n]{\#\text{cl}(\mathcal{O})}$, representing the ideal class $\mathfrak{a} = \mathfrak{l}_1^{e_1} \mathfrak{l}_2^{e_2} \cdots \mathfrak{l}_n^{e_n} \in \text{cl}(\mathcal{O})$. The *public key* is the Montgomery coefficient $A \in \mathbb{F}_p$ of the elliptic curve $\mathfrak{a} * E_0 : y^2 = x^3 + Ax^2 + x$, obtained by applying the action of \mathfrak{a} to the curve E_0 .

Algorithm 2: CSIDH, the non-interactive key-exchange protocol.

Alice	Bob
$ssk_A : \mathbf{a} \in cl(\mathcal{O})$	$ssk_B : \mathbf{b} \in cl(\mathcal{O})$
$spk_A : E_A = \mathbf{a} * E_0$	$spk_B : E_B = \mathbf{b} * E_0$
retrieve E_B and check its supersingularity;	retrieve E_A and check its supersingularity;
$K_A = \mathbf{a} * E_B$	$K_B = \mathbf{b} * E_A$
$K_A = \mathbf{ab} * E_0 = K_B$	

3.2 Our protocol: Π -SIDE

Algorithm 3: Π -SIDE protocol.

Alice: $\mathcal{P}_A \in \mathbb{F}_p$	Bob: $\mathcal{P}_B \in \mathbb{F}_p$
$ssk_A : \mathbf{a} \in cl(\mathcal{O})$	$ssk_B : \mathbf{b} \in cl(\mathcal{O})$
$spk_A : E_A = \mathbf{a} * E_0$	$spk_B : E_B = \mathbf{b} * E_0$
retrieve E_B and check its supersingularity;	retrieve E_A and check its supersingularity;
$esk_A : \mathbf{f} \xleftarrow{\$} cl(\mathcal{O})$	$esk_B : \mathbf{g} \xleftarrow{\$} cl(\mathcal{O})$
$epk_A : E_F = \mathbf{f} * E_0$	$epk_B : E_G = \mathbf{g} * E_0$
$\xrightarrow{E_F}$	
	$\xleftarrow{E_G}$
$ctxt = \mathcal{P}_A \parallel \mathcal{P}_B \parallel E_A \parallel E_B \parallel E_F \parallel E_G$	
$K_B = H(ctxt \parallel \mathbf{g} * E_A \parallel \mathbf{b} * E_F \parallel \mathbf{g} * E_F)$	
$K_A = H(ctxt \parallel \mathbf{a} * E_G \parallel \mathbf{f} * E_B \parallel \mathbf{f} * E_G)$	

Just like in CSIDH, we fix a large prime $p = 4 \cdot \ell_1 \cdot \ell_2 \cdots \ell_n - 1$ for odd and distinct primes ℓ_i . Then we consider the supersingular elliptic curve $E_0 : y^2 = x^3 + x$ defined over \mathbb{F}_p , with endomorphism ring $\mathcal{O} = \mathbb{Z}[\pi]$. We recall that a key-pair (\mathfrak{a}, E_A) can be correctly (with heuristic assumptions) formed as follows:

1. for $i = 1, 2, \dots, n$, sample the exponent $a_i \xleftarrow{\$} \{-m, \dots, m\}$, where m is the smallest integer such that $2m + 1 \geq \sqrt[n]{\#\text{cl}(\mathcal{O})}$;
2. construct the fractional ideal $\mathfrak{a} = \mathfrak{l}_1^{a_1} \cdot \mathfrak{l}_2^{a_2} \cdots \mathfrak{l}_n^{a_n}$. The ideal class \mathfrak{a} will play the role of secret key;
3. evaluate the action of the ideal class \mathfrak{a} on the elliptic curve E_0 , obtaining the curve $E_A = \mathfrak{a} * E_0$; E_A is the Montgomery curve defined by the equation $y^2 = x^3 + Ax^2 + x$ over \mathbb{F}_p and E_A will be the public part of the key pair.

The implementation-oriented reader should always remember that each elliptic curve should be represented using its Montgomery coefficient. For the sake of notation we will refer to the curve instead.

Let \mathcal{P} be the set of participants to the key-exchange protocol. We assume that each party in \mathcal{P} holds a *static secret key* ssk and a *static public key* spk , the latter registered at a certificate authority \mathcal{CA} . The certificate authority, upon registering a public key, does not require a proof of knowledge on the corresponding secret key. We do not demand that public keys differ from party to party, but we allow each party to register only one public key.

Suppose now that two parties Alice and Bob (uniquely identified as \mathcal{P}_A and \mathcal{P}_B) in the set \mathcal{P} want to establish a shared key. Here we have to distinguish between the initiator of the protocol (in our example Alice) and the responder. At the beginning of the session, upon retrieving Bob's public key, Alice samples an *ephemeral secret key* $esk_A = \mathfrak{f}$, computes the *ephemeral public key* $epk_A = E_F$ and sends the result to \mathcal{P}_B . Upon receiving E_F , Bob first checks that it is supersingular and that its Montgomery coefficient is not in $\{\pm 2\}$; if so, he in turn samples an ephemeral secret key $esk_B = \mathfrak{g}$, computes the ephemeral public key E_G and sends it to Alice. Alice herself verifies the validity of E_G . Each of them can now obtain the *session key* K : given access to an hash function H , they can locally compute

$$K = H(\mathcal{P}_A \parallel \mathcal{P}_B \parallel E_A \parallel E_B \parallel E_F \parallel E_G \parallel \mathfrak{a}\mathfrak{g} * E_0 \parallel \mathfrak{b}\mathfrak{f} * E_0 \parallel \mathfrak{f}\mathfrak{g} * E_0).$$

3.3 The SIDH case

A question naturally arises: if Π can be adapted to the CSIDH setting, why can't we do the same in the SIDH setting? On one hand, it is surely possible to translate the protocol itself, since SIDH has a Diffie-Hellman-like structure too. The adaptation would require a different parameter choice, allowing two extra

sets of basis points, and the exchange of four extra image points (the images of the peer’s basis points via the ephemeral isogeny) in order to allow the two parties to compute the common key.

On the other hand, in this case the security proof wouldn’t hit the optimality bound in the tightness loss. As it will be clarified in the next section, a property that plays a fundamental role in this sense is the random self-reducibility of the computational problem. In the next section we provide a formal proof of this feature in the CSIDH case. At our knowledge, there exists no evidence that SIDH shares this property, and it is rather unlikely to find a way to prove it.

4 Random self-reducibility

According to a fundamental definition by Blum and Micali, later rephrased by Naor [NR97], a problem f is *random self-reducible* if solving it at any given instance x can be reduced in polynomial time to the solution of f at one or more random instances y_i . In order to achieve random self-reducibility, there are two conditions that have to be satisfied:

- the generation of the random instances y_1, \dots, y_n has to be performed non-adaptively;
- the instances y_1, \dots, y_n must be uniformly distributed.

Random self-reducible problems are extremely relevant for cryptographic purposes. First of all, they are used in *worst-case to average-case reductions*: a worst-case instance of the problem can be used to generate a set of random instances, so that solving f on the random instances allows us to solve f at the worst-case instance in polynomial time. In the early ’80s, Goldwasser and Micali exploited random self-reducibility of mathematical problems to construct cryptographic algorithms for probabilistic encryption [GM82] and pseudorandom generation [BM82]. Even more, if the group G and its generator g are properly chosen, the random self-reducibility of the discrete logarithm problem guarantees passive security of the plain Diffie-Hellman key-exchange protocol.

4.1 Random self-reducibility on CSIDH

It is folklore that the key-recovery problem in CSIDH is random self-reducible, while SIDH-based problems are not. De Feo and Galbraith [DG19] provide a short proof of random self-reducibility of CSIDH; hereafter, we prove this property more verbosely, in a fashion that resembles the classical proof of re-randomizability for the Computational Diffie-Hellman problem. A fundamental role is played by the commutative action of $cl(\mathcal{O})$ on the set of elliptic curves with endomorphism ring

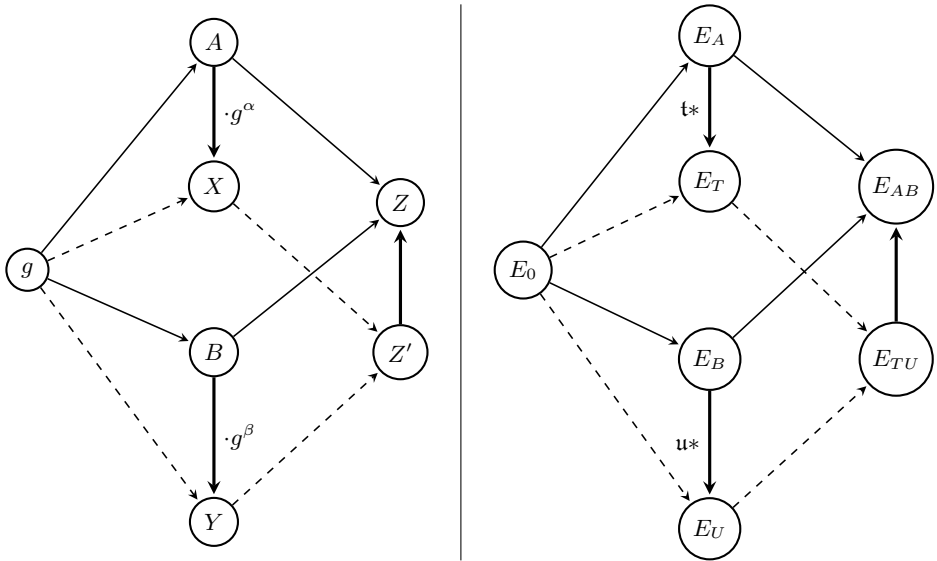


Fig. 1: Rerandomization graphs for Computational Diffie-Hellman and Computational-CSIDH problems.

isomorphic to \mathcal{O} . The presence of a commutative action is a very strong element of resemblance with the Diffie-Hellman protocol.

Let us start with the definition of the Computational CSIDH problem. Let \mathbb{G} be the set of elliptic curves defined over \mathbb{F}_p .

Problem 1 (Computational-CSIDH problem). *Given n distinct odd primes ℓ_i and a large prime $p = 4 \cdot \ell_1 \cdot \ell_2 \cdots \ell_n - 1$, let $E_0 \in \mathbb{G}$ be the supersingular elliptic curve in Montgomery form $y^2 = x^3 + x$. Given two valid CSIDH public keys $A, B \in \mathbb{F}_p$, where A is the Montgomery coefficient of the elliptic curve $E_A = \mathbf{a} * E_0$ and B is the one of $E_B = \mathbf{b} * E_0$, find the Montgomery coefficient $Z \in \mathbb{F}_p$ of the elliptic curve $E_{A,B} = \mathbf{ab} * E_0$.*

Theorem 2. *The computational-CSIDH problem is random self-reducible. In other words, given any two random elliptic curves $E_T = \mathbf{t} * E_0$ and $E_U = \mathbf{u} * E_0$, for any algorithm \mathcal{B} which solves the computational-CSIDH problem with advantage*

$$\text{Adv}_{\mathbb{G}}^{\text{Comp-CSIDH}}(\mathcal{B}) = \text{Prob}[\mathcal{B}(E_T, E_U) = Z' \mid E_T \xleftarrow{\$} \mathbb{G}, E_U \xleftarrow{\$} \mathbb{G}]$$

there exists an oracle algorithm $\mathcal{A}^{\mathcal{B}}$ that, for any input $E_A, E_B \in \mathbb{G}$, outputs the correct solution to the corresponding computational-CSIDH problem with advantage $\text{Adv}_{\mathbb{G}}^{\text{Comp-CSIDH}}(\mathcal{B})$, and has roughly the same running time.

Proof. Let $E_A = \mathbf{a} * E_0$ and $E_B = \mathbf{b} * E_0$ be the two elliptic curves corresponding to the Montgomery coefficients A and B ; we can construct the following algorithm:

```

 $\mathcal{A}^{\mathcal{B}}(E_A, E_B)$ 
 $\mathbf{t}, \mathbf{u} \xleftarrow{\$} \text{cl}(\mathcal{O})$ 
 $E_T \leftarrow \mathbf{t} * E_A = \mathbf{t}' * E_0, E_U \leftarrow \mathbf{u} * E_B = \mathbf{u}' * E_0$ 
 $Z' \leftarrow \mathcal{B}(E_T, E_U)$ 
return  $Z$  of  $[\mathbf{t}^{-1}\mathbf{u}^{-1}] * E_{Z'}$ 
    
```

In other words, the algorithm proceeds as follows. First of all, we pick uniformly at random two isogeny classes $\mathbf{t}, \mathbf{u} \in \text{cl}(\mathcal{O})$: they are defined as $\mathbf{t} = \mathbf{t}_1^{t_1} \mathbf{t}_2^{t_2} \cdots \mathbf{t}_n^{t_n} \in \text{cl}(\mathcal{O})$ and $\mathbf{u} = \mathbf{u}_1^{u_1} \mathbf{u}_2^{u_2} \cdots \mathbf{u}_n^{u_n} \in \text{cl}(\mathcal{O})$ where each exponent t_i, u_j is picked uniformly at random from the set $\{-m, \dots, m\}$. Then we evaluate the action of \mathbf{t} on E_A and the action \mathbf{u} on E_B , obtaining two random elliptic curves $E_T, E_U \in \mathbb{G}$. Finally, we submit the new random instance to the algorithm \mathcal{B} , which outputs Z' , the Montgomery coefficient of the elliptic curve $E_{Z'}$. Since

$$\begin{aligned} E_{Z'} &= \mathbf{t}' \mathbf{u}' * E_0 \\ &= (\mathbf{ta})(\mathbf{ub}) * E_0 \\ &= (\mathbf{tu})(\mathbf{ab}) * E_0 \\ &= (\mathbf{tu}) * E_{A,B}, \end{aligned}$$

we can easily retrieve the Montgomery coefficient Z of the elliptic curve $E_{A,B} = \mathfrak{t}^{-1}\mathfrak{u}^{-1} * E_{Z'}$. The advantage of the algorithm $\mathcal{A}^{\mathcal{B}}$ can be calculated as follows:

$$\text{Prob}[\mathcal{A}^{\mathcal{B}}(E_A, E_B) = Z] = \text{Prob}\left[\mathfrak{t}, \mathfrak{u} \xleftarrow{\$} \text{cl}(\mathcal{O}) : \mathcal{B}(\mathfrak{t} * E_A, \mathfrak{u} * E_B) = (\mathfrak{t}\mathfrak{a})(\mathfrak{u}\mathfrak{b}) * E_0\right].$$

By construction, the ideal classes \mathfrak{t} and \mathfrak{u} can be considered as sampled uniformly at random from $\text{cl}(\mathcal{O})$ (for the heuristics assumed in CSIDH), and therefore the elliptic curves $E_T = \mathfrak{t} * E_A$ and $E_U = \mathfrak{u} * E_B$ are independent and uniformly distributed on \mathbb{G} . Therefore, the oracle consulted by $\mathcal{A}^{\mathcal{B}}$ receives a well formed instance, so we can conclude that

$$\begin{aligned} \text{Prob}[\mathcal{A}^{\mathcal{B}}(E_A, E_B) = Z] &= \text{Prob}\left[\mathcal{B}(E_T, E_U) = \mathfrak{t}\mathfrak{a}\mathfrak{u}\mathfrak{b} * E_0 \mid \mathfrak{t}, \mathfrak{u} \xleftarrow{\$} \text{cl}(\mathcal{O})\right] \\ &= \text{Adv}_{\mathbb{G}}^{\text{Comp-CSIDH}}(\mathcal{B}). \end{aligned}$$

As pointed out in section 2.3, we can efficiently compute the action of the ideal classes \mathfrak{l} and \mathfrak{l}^{-1} by using Vélu-type formulae. Therefore we can conclude that, if \mathcal{B} runs in t -time, then the algorithm $\mathcal{A}^{\mathcal{B}}$ runs in $(t + \delta)$ -time, where δ is the small running time required to sample elements and evaluate the action of ideal classes. □

5 Security of Π -SIDE

In this section, we define some allegedly hard problems in the CSIDH setting. The definition of our security model and the full proof can be found in Appendix B. The structure of the proof is similar to the one for protocol Π [CCG⁺19], but we have made a number of changes, mostly related to the new re-randomization technique. A straightforward adaption would have not been possible by simply substituting exponentiations with class group evaluations.

5.1 Hard problems

In section 4.1, we have seen that the Comp-CSIDH problem consists in finding the Montgomery coefficient $Z \in \mathbb{F}_p$ of the elliptic curve $\mathfrak{a}\mathfrak{b} * E_0$ given the Montgomery coefficients of the curves $E_A = \mathfrak{a} * E_0$ and $E_B = \mathfrak{b} * E_0$. In order to keep the notation as simple as possible, we will formulate the next problems referring to the elliptic curve itself, instead of its Montgomery coefficient. The reader should always keep in mind that, when it comes to the implementation, each elliptic curve will be represented by its Montgomery coefficient, which lies in \mathbb{F}_p . We start with defining a decisional problem:

Problem 2 (Decisional-CSIDH problem). *In the CSIDH setting, let $\mathbf{a}, \mathbf{b}, \mathbf{r} \xleftarrow{\$} cl(\mathcal{O})$ be three elements randomly sampled from $cl(\mathcal{O})$ and let $b \xleftarrow{\$} \{0, 1\}$ be the result of a fairly tossed coin. If $b = 0$ set $E_Z = \mathbf{r} * E_0$, otherwise set $E_Z = \mathbf{a}\mathbf{b} * E_0$ and run the adversary on input $(E_A = \mathbf{a} * E_0, E_B = \mathbf{b} * E_0, E_Z)$. We define the advantage of \mathcal{A} in solving the decisional CSIDH problem over $cl(\mathcal{O})$ as*

$$Adv_{cl(\mathcal{O})}^{Dec-CSIDH}(\mathcal{A}) := \left| Prob[\mathcal{A}(E_A, E_B, E_Z) = b] - \frac{1}{2} \right|.$$

In other words, the decisional problem is hard if the adversary succeeds with a negligible probability in distinguishing among a properly computed session key and a random key. Trivially, if we can solve the computational variant of problem then we can also solve its decisional variant. But does the opposite hold?

Problem 3 (Gap-CSIDH problem). *In the CSIDH setting, let $\mathbf{a}, \mathbf{b} \xleftarrow{\$} cl(\mathcal{O})$ be two elements randomly sampled from $cl(\mathcal{O})$, corresponding to the curves $E_A = \mathbf{a} * E_0$ and $E_B = \mathbf{b} * E_0$. Suppose that the adversary \mathcal{A} is given access to a Dec-CSIDH oracle $\mathcal{D}(\cdot, \cdot, \cdot)$, which outputs 1 if queried on a valid CSIDH triplet (E_A, E_B, E_{AB}) and 0 otherwise. We define the advantage of \mathcal{A} in solving the Gap-CSIDH problem over $cl(\mathcal{O})$ as*

$$Adv_{cl(\mathcal{O})}^{Gap-CSIDH}(\mathcal{A}) := Prob[\mathcal{A}(E_A, E_B) = E_{A,B}, \text{ providing } \mathcal{A} \text{ access to } \mathcal{D}(\cdot, \cdot, \cdot)]$$

The security of protocol *II* [CCG⁺19] relies on the Strong-DH problem [ABR01], a variant of the Gap problem in which the adversary is granted access to a more limited decisional oracle.

Problem 4 (Strong-CSIDH problem). *In the CSIDH setting, let $\mathbf{a}, \mathbf{b} \xleftarrow{\$} cl(\mathcal{O})$ be two elements randomly sampled from $cl(\mathcal{O})$, corresponding to the curves $E_A = \mathbf{a} * E_0$ and $E_B = \mathbf{b} * E_0$. Let \mathcal{D} be an oracle for the decisional CSIDH problem. Suppose that the adversary \mathcal{A} is given access to a decisional oracle with fixed first input $\mathcal{D}_X(\cdot, \cdot) := \mathcal{D}(E_X, \cdot, \cdot)$, which outputs 1 if queried on a valid CSIDH triplet (E_X, E_Y, E_{XY}) and 0 otherwise. We define the advantage of \mathcal{A} in solving the Strong-CSIDH problem over $cl(\mathcal{O})$ as*

$$Adv_{cl(\mathcal{O})}^{St-CSIDH}(\mathcal{A}) := Prob[\mathcal{A}(E_A, E_B) = E_{A,B}, \text{ providing } \mathcal{A} \text{ access to } \mathcal{D}_X(\cdot, \cdot)]$$

Rerandomizability of the Gap-CSIDH and the Strong-CSIDH problems follows directly from Theorem 4.1. The full security proof, which strongly relies on these problems, is provided in Appendix B. Based on the current state of the art, there is no reason to believe that the above problems can be easily solved.

6 Comparison

Comparing the efficiency of our scheme with other post-quantum schemes is hard. First of all, many schemes do not have a security proof [Ber19] (and thus we cannot define theoretically-sound parameters); secondly, it is highly non-trivial to convert the concrete analysis into security parameters for many schemes.

Castruck et al. [CLM⁺18] describe an implementation for a 128-bit security level that requires about $106 \cdot 10^6$ clock cycles to compute the group action. Since our protocol *II*-SIDE requires four group action computations, we have a total cost of about $400 \cdot 10^6$ clock cycles, ignoring hashing and other cheap operations.

The most natural target for comparison is SIKE [JAC⁺19]. The original *II*-protocol can also be generalized to SIKE, but one would probably not attempt to build it on top of the defined KEM, but use the underlying isogeny instead. Table 2.1 from SIKE [JAC⁺19] suggests that an isogeny computation using the optimized implementation (which probably matches the CSIDH implementation best) requires roughly $50 \cdot 10^6$ clock cycles for the 128 bit security level (SIKEp434), which becomes roughly $200 \cdot 10^6$ clock cycles for the generalized *II*-protocol, significantly faster than the CSIDH-based version.

Now suppose we instantiate the protocol with 2^{16} users and 2^{16} sessions per user. In this case, the apparent security level of our protocol falls to about 110 bits. The SIKE-based protocol with the standard security proof will have a quadratic security loss. This means that in order to get a similar theoretically-sound security level from the SIKE-based protocol, we need to switch to SIKEp610. Again, Table 2.1 from SIKE [JAC⁺19] suggests that an isogeny computation using the optimized implementation requires roughly $160 \cdot 10^6$ clock cycles. The generalized *II*-protocol then requires roughly $640 \cdot 10^6$ clock cycles, which is significantly slower than the CSIDH-based version. According to this approximate analysis, the CSIDH-based version is faster than the corresponding SIKE-based protocol when instantiated with theoretically-sound parameters. However, to properly determine which is faster, comparable optimized implementations would be needed.

Another natural comparison target is the Strongly secure AKE from Super-singular Isogenies by Xu et al. [XXW⁺19] referred to in section 1.2. For their two-pass protocol SIAKE₂ and their three-pass protocol SIAKE₃, the numbers of cycles are approximately $7 \cdot 10^9$ and $6 \cdot 10^9$, respectively [XXW⁺19, Table 6]. Our protocol is significantly faster, by about an order of magnitude.

7 Conclusions and open problems

In this paper we have shown that it is possible to construct post-quantum isogeny-based key-exchange protocols with optimal tightness, without compromising efficiency and key-size. The protocol is an easy adaptation of protocol *II* [CCG⁺19],

where we substitute exponentiations in cyclic groups with actions of ideal classes on elliptic curves. The adaptation of the proof, which requires random self-reducibility of the computational-CSIDH problem, could not be done trivially. Indeed, we have had to exploit a different re-randomization technique for the computational challenge, since we only have one group operation on ideal classes against two operations (addition and multiplication) on exponents. We have shown that the resulting scheme is competitive with other isogeny-based protocols, which lack a security proof or have a larger tightness loss.

Our protocol is proven secure in the Random Oracle Model. In a crucial step we use the Strong-CSIDH oracle to detect if the adversary queries the hashing oracle on an input which contains the solution to a given computational-CSIDH challenge. If we allow the adversary to make quantum queries, the target solution might be hidden in the superposition of states. We believe that collapsing the input state after the oracle's answer is not invalidating our security proof, since we do not need to reprogram the oracle. We leave the proof of security in the QROM as future work.

A stronger security notion can be achieved by adding the static-static term in the session-key computation, or by applying the NAXOS trick [LLM07]. But security against state compromise (ephemeral key reveal) increases the tightness loss, since we cannot tightly deal with state reveal queries. How to move to a stronger security model without losing in tightness is still an open problem.

We have seen how the flexible algebraic structure at the basis of CSIDH can be exploited to remodel protocol *II* in the isogeny setting. Nevertheless, the simplicity of this scheme might be further exploited. Other quantum-hard problems might be used to translate the scheme in other algebraic contexts. Adaptions in this direction are left for further research.

As a last remark, we would like to clarify that our scheme is not affected by the algorithm recently published by Castryck et al. [CSV20]. This attack, which breaks some instances of the Decisional CSIDH problem, does not work when $p \equiv 3 \pmod{4}$, as per our protocol.

References

- AASA⁺. Gorjan Alagic, Jacob Alperin-Sheriff, Daniel Apon, David Cooper, Quynh Dang, John Kelsey, Yi-Kai Liu, Carl Miller, Dustin Moody, Rene Peralta, Ray Perlner, Angela Robinson, and Daniel Smith-Tone. Nistir 8309. <https://doi.org/10.6028/NIST.IR.8309>.
- ABR01. Michel Abdalla, Mihir Bellare, and Phillip Rogaway. The oracle diffie-hellman assumptions and an analysis of dhies. volume 2020, pages 143–158, 04 2001.
- Ber19. Daniel J. Bernstein. Comparing proofs of security for lattice-based encryption. *IACR Cryptology ePrint Archive*, 2019:691, 2019.

- BFG⁺19. Jacqueline Brendel, Marc Fischlin, Felix Günther, Christian Janson, and Douglas Stebila. Towards post-quantum security for signal's x3dh handshake. Cryptology ePrint Archive, Report 2019/1356, 2019. <https://eprint.iacr.org/2019/1356>.
- BFK⁺14. Karthikeyan Bhargavan, Cédric Fournet, Markulf Kohlweiss, Alfredo Pironti, Pierre-Yves Strub, and Santiago Zanella Béguelin. Proving the TLS handshake secure (as it is). In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO 2014, Part II*, volume 8617 of *LNCS*, pages 235–255, Santa Barbara, CA, USA, August 17–21, 2014. Springer, Heidelberg, Germany.
- BJLS16. Christoph Bader, Tibor Jager, Yong Li, and Sven Schäge. On the impossibility of tight cryptographic reductions. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, pages 273–304, Vienna, Austria, May 8–12, 2016. Springer, Heidelberg, Germany.
- BM82. Manuel Blum and Silvio Micali. How to generate cryptographically strong sequences of pseudo random bits. In *23rd FOCS*, pages 112–117, Chicago, Illinois, November 3–5, 1982. IEEE Computer Society Press.
- CCG⁺19. Katriel Cohn-Gordon, Cas Cremers, Kristian Gjøsteen, Håkon Jacobsen, and Tibor Jager. Highly efficient key exchange protocols with optimal tightness. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part III*, volume 11694 of *LNCS*, pages 767–797, Santa Barbara, CA, USA, August 18–22, 2019. Springer, Heidelberg, Germany.
- CLM⁺18. Wouter Castryck, Tanja Lange, Chloe Martindale, Lorenz Panny, and Joost Renes. CSIDH: An efficient post-quantum commutative group action. In Thomas Peyrin and Steven Galbraith, editors, *ASIACRYPT 2018, Part III*, volume 11274 of *LNCS*, pages 395–427, Brisbane, Queensland, Australia, December 2–6, 2018. Springer, Heidelberg, Germany.
- CSV20. Wouter Castryck, Jana Sotáková, and Frederik Vercauteren. Breaking the decisional diffie-hellman problem for class group actions using genus theory. Cryptology ePrint Archive, Report 2020/151, 2020. <https://eprint.iacr.org/2020/151>.
- Deu41. Max Deuring. Die Typen der Multiplikatorenringe elliptischer Funktionenkörper. *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg*, 14:197–272, 1941.
- DG19. Luca De Feo and Steven D. Galbraith. SeaSign: Compact isogeny signatures from class group actions. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part III*, volume 11478 of *LNCS*, pages 759–789, Darmstadt, Germany, May 19–23, 2019. Springer, Heidelberg, Germany.
- Feo17. Luca De Feo. Mathematics of isogeny based cryptography. *CoRR*, abs/1711.04062, 2017. <http://arxiv.org/abs/1711.04062>.
- Gal18. Steven D. Galbraith. Authenticated key exchange for SIDH. Cryptology ePrint Archive, Report 2018/266, 2018. <https://eprint.iacr.org/2018/266>.
- GM82. Shafi Goldwasser and Silvio Micali. Probabilistic encryption and how to play mental poker keeping secret all partial information. In *14th ACM STOC*, pages 365–377, San Francisco, CA, USA, May 5–7, 1982. ACM Press.

- HM89. James Lee Hafner and Kevin S. McCurley. A rigorous subexponential algorithm for computation of class groups. *Journal of The American Mathematical Society*, 1989.
- HS09. Joseph H. Silverman. *The Arithmetic of Elliptic Curves*, volume 106. 2009.
- JAC⁺19. David Jao, Reza Azarderakhsh, Matthew Campagna, Craig Costello, Luca De Feo, Basil Hess, Amir Jalali, Brian Koziel, Brian LaMacchia, Patrick Longa, Michael Naehrig, Joost Renes, Vladimir Soukharev, David Urbanik, and Geovandro Pereira. SIKE. Technical report, National Institute of Standards and Technology, 2019. available at <https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions>.
- JD11. David Jao and Luca De Feo. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In Bo-Yin Yang, editor, *Post-Quantum Cryptography - 4th International Workshop, PQCrypto 2011*, pages 19–34, Tapei, Taiwan, November 29 – December 2 2011. Springer, Heidelberg, Germany.
- JKSS12. Tibor Jager, Florian Kohlar, Sven Schäge, and Jörg Schwenk. On the security of TLS-DHE in the standard model. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 273–293, Santa Barbara, CA, USA, August 19–23, 2012. Springer, Heidelberg, Germany.
- KPW13. Hugo Krawczyk, Kenneth G. Paterson, and Hoeteck Wee. On the security of the TLS protocol: A systematic analysis. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 429–448, Santa Barbara, CA, USA, August 18–22, 2013. Springer, Heidelberg, Germany.
- Kra05. Hugo Krawczyk. HMQV: A high-performance secure Diffie-Hellman protocol. In Victor Shoup, editor, *CRYPTO 2005*, volume 3621 of *LNCS*, pages 546–566, Santa Barbara, CA, USA, August 14–18, 2005. Springer, Heidelberg, Germany.
- KTAT20. Tomoki Kawashima, Katsuyuki Takashima, Yusuke Aikawa, and Tsuyoshi Takagi. An efficient authenticated key exchange from random self-reducibility on csidh. Cryptology ePrint Archive, Report 2020/1178, 2020. <https://eprint.iacr.org/2020/1178>.
- LLM07. Brian A. LaMacchia, Kristin Lauter, and Anton Mityagin. Stronger security of authenticated key exchange. In Willy Susilo, Joseph K. Liu, and Yi Mu, editors, *ProvSec 2007*, volume 4784 of *LNCS*, pages 1–16, Wollongong, Australia, November 1–2, 2007. Springer, Heidelberg, Germany.
- LM06. Kristin Lauter and Anton Mityagin. Security analysis of KEA authenticated key exchange protocol. In Moti Yung, Yevgeniy Dodis, Aggelos Kiayias, and Tal Malkin, editors, *PKC 2006*, volume 3958 of *LNCS*, pages 378–394, New York, NY, USA, April 24–26, 2006. Springer, Heidelberg, Germany.
- Lon18. Patrick Longa. A note on post-quantum authenticated key exchange from supersingular isogenies. Cryptology ePrint Archive, Report 2018/267, 2018. <https://eprint.iacr.org/2018/267>.

- NR97. Moni Naor and Omer Reingold. Number-theoretic constructions of efficient pseudo-random functions. In *38th FOCS*, pages 458–467, Miami Beach, Florida, October 19–22, 1997. IEEE Computer Society Press.
- Sho97. Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26(5):1484–1509, Oct 1997.
- Sut13. Andrew V. Sutherland. On the evaluation of modular polynomials. *Tenth Algorithmic Number Theory Symposium (ANTS X), MSP Open Book Series 1*, pages 531–555, 2013.
- Vél71. J. Vélú. Isogénies entre courbes elliptiques. *Comptes-Rendus de l'Académie des Sciences, Série I*, 273:238–241, 1971.
- Was08. Lawrence C. Washington. *Elliptic Curves: Number Theory and Cryptography, Second Edition*. Chapman & Hall/CRC, 2 edition, 2008.
- XXW⁺19. Xiu Xu, Haiyang Xue, Kumpeng Wang, Man Ho Au, and Song Tian. Strongly secure authenticated key exchange from supersingular isogenies. In Steven D. Galbraith and Shiho Moriai, editors, *ASIACRYPT 2019, Part I*, volume 11921 of *LNCS*, pages 278–308, Kobe, Japan, December 8–12, 2019. Springer, Heidelberg, Germany.

Appendix

A Security model

Suppose that we have a certificate authority \mathcal{CA} , a set of parties $\mathcal{P} := \{\mathcal{P}_i\}_{i=1}^{\mu}$ and an adversary \mathcal{M} . The parties can communicate with each other and with \mathcal{CA} by using an unauthenticated network. \mathcal{CA} can be seen as a globally trusted party, or register, who holds and distributes the static public keys of the parties in \mathcal{P} . At any time, a new player can join in \mathcal{P} by communicating his static public key to the \mathcal{CA} , and the register can grow indefinitely. As we mentioned before, we do not require different parties to hold different public keys, and neither we demand any proof of knowledge of the related secret key. Our protocol is implicitly authenticated and, as such, no identification or proof of knowledge of any secret information is required. The only constraint we impose is that each member can commit to only one static public key at a time.

Each \mathcal{P}_i is represented as a set of oracles $\{\pi_i^1, \pi_i^2, \dots, \pi_i^k\}$, one for each of the k sessions the user can participate to. Each oracle $\pi_i^s = (P_i^s, \psi_i^s, K_i^s, \text{sent}_i^s, \text{recv}_i^s, \text{role}_i^s)$ maintains an internal state consisting of:

- the identity of the intended peer P_i^s which is supposedly taking part to the key-exchange session;
- $\psi_i^s \in \{\emptyset, \text{accept}, \text{reject}\}$, which indicates whether the session key has not been computed yet, or if it has been accepted or rejected;
- the session key K_i^s , which is not empty if and only if $\psi_i^s = \text{accept}$;
- sent_i^s , the collection of all the messages sent by the oracle;
- recv_i^s , the collection of all the messages received by the oracle;
- the role role_i^s of the oracle (**init** or **resp**).

sent_i^s and recv_i^s together form the view view_i^s of \mathcal{P}_i of the session s .

We now define the attribute for indicating two oracles that allegedly participated to the same key-exchange session. Two oracles π_i^s and π_j^t are called *partner oracles* if

1. $P_i^s = P_j$ and $P_j^t = P_i$, i.e. if they are the intended peer of each other;
2. $\psi_i^s = \psi_j^t = \text{accept}$, i.e. they both accepted the session key;
3. $\text{view}_i^s = \text{view}_j^t$, i.e. the messages sent and received by P_i match with the ones respectively received and sent by P_j during the key-exchange session;
4. they have specular roles.

Slightly simplifying the definition, an oracle is *fresh* if and only if its session key has not been revealed, its partner oracle has not been corrupted or tested and the partner's session key has not been revealed. We will later constrain the adversary to test only fresh oracles. A party is *honest* if all its oracles are fresh, i.e. if it has not been corrupted yet.

In this model, the adversary \mathcal{A} has full control over the network and interacts with the oracles through queries that allow it to

- *activate* an oracle π_i^s and assign a role by sending it a message on behalf of a peer P_j ;
- *reveal the long-term secret key* of a user P_i . This query provides the target user with the attribute of *corrupted* and all its oracles will answer \perp to each later query;
- *register the long-term public key* for a new user. No knowledge of the corresponding secret key is required and the public key is distributed to all other users;
- *reveal the session key* k_i^s stored in the internal state of any oracle π_i^s . The target oracle is now said to be *revealed*.
- *test an oracle* π_i^s , which outputs \perp if $\psi_i^s \neq \text{accept}$. If $\psi_i^s = \text{accept}$ it then outputs a key, which is either the session-key or one picked at random, according to a previously defined random bit. The key, may it be real or the random, is consistently issued in case of further tests.

Note that the adversary cannot query on the ephemeral key of any session.

We work in the *Real-or-Random* model: when tested, each oracle will output a real session key or a random key, according to a bit sampled at the beginning of the security game. If $b = 0$ each oracle tested during the game will output a random key, while if $b = 1$ each tested oracle will output the real session key.

Once the environment has been set up, we run the following *AKE security game* $G_\Pi(\mu, k)$, with μ honest parties and at most k sessions per user:

1. at first we toss a coin $b \xleftarrow{\$} \{0, 1\}$. We also set up μ parties, providing each of them with a long-term key pair (sk_i, pk_i) and with k oracles;
2. we then run the adversary \mathcal{A} , which knows all the public keys and can make any number of the previously defined queries. The only restriction is that an oracle must be fresh when it is tested;
3. at some point, \mathcal{A} will eventually output b' , its guess on the initial bit b . If the tested oracles are fresh and $b' = b$, then \mathcal{A} wins the security game.

An adversary can try to break the system in three different ways: it can trick two oracles into computing different session keys (event $\mathbf{break}_{\text{Sound}}$), break the unicity of the partnership relation between two oracles (event $\mathbf{break}_{\text{Unique}}$) or successfully guess $b' = b$ (event $\mathbf{break}_{\text{KE}}$). We formalise these ideas in the following definition.

Definition 5. *In this security model, a protocol Π fails if at least one of $\mathbf{break}_{\text{Sound}}$, $\mathbf{break}_{\text{Unique}}$ and $\mathbf{break}_{\text{KE}}$ occurs while running game $G(\mu, k)$. Given*

an adversary \mathcal{A} , we define its advantage against the AKE security of protocol Π as

$$\text{Adv}_{\Pi}^{\text{AKE}}(\mathcal{A}) := \max \left\{ \text{Prob}[\text{break}_{\text{sound}}], \text{Prob}[\text{break}_{\text{unique}}], \text{Prob}[\text{break}_{\text{KE}}] - \frac{1}{2} \right\}$$

and we say that it $(t, \epsilon_{\mathcal{A}}, \mu, k)$ -breaks the AKE security of Π if it runs in time t and has advantage $\text{Adv}_{\Pi\text{-SIDE}}^{\text{AKE}}(\mathcal{A}) \geq \epsilon_{\mathcal{A}}$.

B The security proof

As in the proof by Cohn-Gordon et al. [CCG⁺19], we prove the following:

Theorem 3. *Consider an environment running Π -SIDE together with an adversary \mathcal{A} against AKE security of Π -SIDE. Then there exist 3 Strong-CSIDH adversaries $\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3$ such that \mathcal{A} 's advantage $\text{Adv}_{\Pi\text{-SIDE}}^{\text{AKE}}(\mathcal{A})$ is at most*

$$\mu \cdot \text{Adv}_{\text{cl}(\mathcal{O})}^{\text{St-CSIDH}}(\mathcal{B}_1) + \text{Adv}_{\text{cl}(\mathcal{O})}^{\text{St-CSIDH}}(\mathcal{B}_2) + \mu \cdot \text{Adv}_{\text{cl}(\mathcal{O})}^{\text{St-CSIDH}}(\mathcal{B}_3) + \frac{\mu k^2}{N}$$

where $\mu = |\mathcal{P}|$ is the number of parties, k is the maximal number of AKE-sessions per party and N is the order of $\text{cl}(\mathcal{O})$. The run-time of adversaries $\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3$ is almost the same as \mathcal{A} and they make at most as many queries to the Strong-CSIDH oracle as \mathcal{A} does to the hash oracle H .

The proof structure is analogous to the one of Π , rephrased and adapted to our setting. It consists of six different games: Game 0 is the AKE experiment, while the other five games involve the following oracle types:

- type I: an initiator oracle which has received the response from a responder oracle (honest when the response is received) and with which it agrees on the transcript `ctxt`;
- type II: an initiator oracle whose intended peer is honest until the oracle accepts;
- type III: a responder oracle triggered by an honest initiator, with which it agrees on `ctxt` and which is still honest when it receives the response;
- type IV: a responder oracle whose intended peer is honest until the oracle accepts;
- type V: an oracle (whether initiator or responder) whose intended peer is corrupted.

At the time of starting an AKE session, an initiator oracle cannot be entirely sure about the intended peer's honesty: we cannot tell if it is of type I or type II. This uncertainty vanishes when it receives the response and it comes the time to

Oracle	Init.	Resp.	Honest partner (before acceptance)	Honest partner (after acceptance)	Corrupted partner	Agreement on <code>ctxt</code>
Type I	✓		✓	✓		✓
Type II	✓		✓			
Type III		✓	✓	✓		✓
Type IV		✓	✓			
Type V	✓	✓			✓	

Table 1: Oracle types, defined by role, partner’s honesty and agreement on `ctxt`.

compute the session key. This aspect will be taken in account during the definition of the security games.

We now define six different security games, which will lead to the definition of the three adversaries $\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3$ in Theorem 3. In each game we will have to look at the input to the hash function; for future references, we indicate the general form of the input to the hash oracle involving a key-exchange session between parties $\mathcal{P}_A, \mathcal{P}_B$ as

$$\mathcal{P}_A \parallel \mathcal{P}_B \parallel E_A \parallel E_B \parallel E_F \parallel E_G \parallel W_1 \parallel W_2 \parallel W_3 \quad (2)$$

For $i = 0, 1, \dots, 5$ we denote with S_j the event “Game i outputs 1”, which will indicate a success for the adversary in breaking protocol *II*-SIDE (i.e. at least one of the events $\mathbf{break}_{\text{sound}}$, $\mathbf{break}_{\text{unique}}$ and $\mathbf{break}_{\text{KE}}$ happens during Game i).

Game 0. In this game, we simply run the usual AKE security game: the adversary can corrupt some players, reveal some session keys (but not any ephemeral secret key) and delay/redirect messages. When it will be ready, it will pick a fresh oracle and make a query test on its session key. Game 0 will output 1 whenever the adversary breaks the AKE security of protocol *II*-SIDE:

$$\text{Prob}[S_0] = \text{Prob}[\mathbf{break}_{\text{KE}}].$$

Game 1. In this game we abort if the same `ctxt` is computed by two non-partnered oracles. We can upper-bound the probability of this event with the probability that the following conditions are simultaneously verified:

1. two oracles π_i^s, π_i^t belong to the same user P_i ;
2. they pick the same ephemeral secret key during their respective sessions;
3. they are involved in two key-exchange sessions with the same user P_j (since the identity of the intended peer is part of the `ctxt`).

Recalling that we have μ users engaging in at most k sessions, we get that

$$|\text{Prob}[S_1] - \text{Prob}[S_0]| \leq \frac{\mu k^2}{N}$$

and thus, since in this game the unicity of the partner oracle cannot be broken, we can conclude that

$$\text{Prob}[\text{break}_{\text{Unique}}] \leq \frac{\mu k^2}{N}.$$

Game 2. In this game we modify how each oracle computes the session key: instead of computing the input to the hash oracle H , it checks if the adversary has queried the oracle on that same input, and behaves consequently: if the answer is yes, then it stores that hash value as the session key (i.e. it properly computes the key), otherwise it picks a key at random and stores that one instead. Note that, when it comes the time for an initiator oracle to compute the session key, the oracle type is fully determined.

A type I oracle (an initiator oracle with a definitely honest partner oracle with which it agrees on the `ctxt`) will store the key computed by the corresponding responder oracle.

Each type II or type V initiator oracles of party \mathcal{P}_A has to check if the input

$$\mathcal{P}_A \parallel \mathcal{P}_B \parallel E_A \parallel E_B \parallel E_F \parallel E_G \parallel \mathbf{a} * E_G \parallel \mathbf{f} * E_B \parallel \mathbf{f} * E_G$$

has been object of any oracle query. If so, it sets its session key to the corresponding hash value (previously stored by the responder oracle), otherwise it picks a session key at random (answering consistently to any following hash query on that same input).

Each type III, IV or V responder oracle of a party \mathcal{P}_B in a session with \mathcal{P}_A will check if any queries have been made on input

$$\mathcal{P}_A \parallel \mathcal{P}_B \parallel E_A \parallel E_B \parallel E_F \parallel E_G \parallel \mathbf{g} * E_A \parallel \mathbf{b} * E_F \parallel \mathbf{g} * E_F.$$

If so, it stores the same result; otherwise, it stores a random key. In any case, each later hash query is consistently answered with the stored session key.

We cannot observe the exact time in which the key derivation oracle is queried for the first time, thus Game 2 outputs 1 whenever Game 1 outputs 1, and vice versa. We can therefore conclude that

$$\text{Prob}[S_2] = \text{Prob}[S_1].$$

Game 3. In this game (which is a variant of Game 2) we modify how a type IV oracle (a responder oracle whose intended peer is honest until the oracle accepts) chooses the session key. What it does is 1) to pick a random key; 2) to wait for

the adversary to possibly corrupt the intended peer \mathcal{P}_A ; 3) only then modify the hash oracle with the random key k .

We can now define the following events:

- L (for Long-term key), in which the adversary queries the hash oracle on input

$$\mathcal{P}_A \parallel \mathcal{P}_B \parallel E_A \parallel E_B \parallel E_F \parallel E_G \parallel \mathfrak{g} * E_A \parallel \mathfrak{b} * E_F \parallel \mathfrak{g} * E_F$$

before the long-term secret key of any initiator oracle is revealed;

- L_A is the same event as L , but for a specific intended peer \mathcal{P}_A . Trivially $Prob[L] = \sum_i Prob[L_i]$;
- C_A (for Corruption), in which the adversary queries the hash oracle on input

$$\mathcal{P}_A \parallel \mathcal{P}_B \parallel E_A \parallel E_B \parallel E_F \parallel E_G \parallel \mathfrak{g} * E_A \parallel W_2 \parallel W_3$$

before peer \mathcal{P}_A is corrupted; therefore we have $Prob[L_A] \leq Prob[C_A]$.

In order to obtain a bound on $Prob[C_A]$ (and thus a bound on $Prob[L]$), we construct an adversary \mathcal{B}_1 against the Strong-CSIDH problem.

Definition 6. [*Adversary \mathcal{B}_1*] Consider now an adversary \mathcal{B}_1 which is given a Comp-CSIDH challenge (E_S, E_T) and is given access to a $\mathcal{D}_S(\cdot, \cdot)$ oracle. First of all, it chooses a user \mathcal{P}_A uniformly at random and sets its long-term public key to $E_A = E_S$. Then it sets the ephemeral public key of a type IV oracle to be $\mathfrak{r} * E_T$, for a random $\mathfrak{r} \xleftarrow{\$} cl(\mathcal{O})$. Finally, it runs Game 2. If \mathcal{B}_1 corrupts \mathcal{P}_A , the experiment aborts.

We need to recognise the hash queries that involve the user \mathcal{P}_A (happening in Game 2) and those involving the type IV oracle of any party \mathcal{P}_B . In particular,

1. consider hash queries of the form

$$\mathcal{P}_A \parallel \mathcal{P}_B \parallel E_A \parallel E_B \parallel E_F \parallel E_G \parallel W_1 \parallel \mathfrak{b} * E_F \parallel \mathfrak{f} * E_G$$

involving user \mathcal{P}_A as initiator. We do not know \mathcal{P}_A 's secret key $\mathfrak{a} = \mathfrak{s}$, so we have to recognise if W_1 is actually $E_{AG} = \mathfrak{s} * E_G$. This can be done by checking if $\mathcal{D}_S(E_G, W_1) = 1$;

2. consider hash queries of the form

$$\mathcal{P}_B \parallel \mathcal{P}_A \parallel E_B \parallel E_A \parallel E_F \parallel E_G \parallel \mathfrak{b} * E_G \parallel W_2 \parallel \mathfrak{f} * E_G$$

involving user \mathcal{P}_A as responder. Again, we do not know \mathcal{P}_A 's secret key $\mathfrak{a} = \mathfrak{s}$, but this time it is $W_2 = \mathfrak{a} * E_F$ that we cannot compute; thus we

have to recognise if W_2 is actually $\mathfrak{s} * E_F$. This can be done by checking if $\mathcal{D}_S(E_F, W_2) = 1$;

3. consider hash queries of the form

$$\mathcal{P}_A \parallel \mathcal{P}_B \parallel E_A \parallel E_B \parallel E_F \parallel E_G \parallel \mathfrak{g} * E_A \parallel W_2 \parallel W_3$$

involving the type IV oracle and user \mathcal{P}_A . We have to recognise if W_1 is actually $\mathfrak{rt} * E_A = \mathfrak{g} * E_S$. This can be done by checking if $\mathcal{D}_S(E_G, W_1) = 1$. Whenever we succeed and we find that $W_1 = E_{SG} = \mathfrak{s} * E_G$, since we computed $E_G = \mathfrak{r} * E_T$, we output

$$E_Z = \bar{\mathfrak{r}} * W_1 = \bar{\mathfrak{r}}\mathfrak{s} * E_G = \bar{\mathfrak{r}}\mathfrak{s}\mathfrak{r} * E_T = \bar{\mathfrak{r}}\mathfrak{r}\mathfrak{s}E_T = \mathfrak{s} * E_T = E_{ST}.$$

We have just described an adversary \mathcal{B}_1 which succeeds whenever event L_A occurs in Game 2. L_A can occur only before \mathcal{P}_A is corrupt, and thus \mathcal{B}_1 's game would have gone through. We can therefore define the upper bound

$$Adv_{cl(\mathcal{O})}^{St-CSIDH}(\mathcal{B}_1) \geq \frac{1}{\mu} \sum_{i=1}^{\mu} Prob[C_I] \geq \frac{1}{\mu} \sum_{i=1}^{\mu} Prob[L_I] = \frac{1}{\mu} Prob[L]$$

from which we get that

$$|Prob[S_3] - Prob[S_2]| \leq Prob[L] \leq \mu \cdot Adv_{cl(\mathcal{O})}^{St-CSIDH}(\mathcal{B}_1)$$

the first element at the right-hand side of the inequality in Theorem 3.

Game 4. In this game a type III oracle (a responder oracle triggered by an honest initiator, with which it agrees on the `ctxt` and which is still honest when it receives the response) chooses the session key at random without modifying the key derivation hash oracle. Consider an oracle belonging to user \mathcal{P}_B with static secret key \mathfrak{b} and ephemeral secret key \mathfrak{g} whose intended honest peer \mathcal{P}_A has static secret key \mathfrak{a} . The adversary can find out this change only if (call this event L) it makes a query of the form

$$\mathcal{P}_A \parallel \mathcal{P}_B \parallel E_A \parallel E_B \parallel E_F \parallel E_G \parallel W_1 \parallel W_2 \parallel \mathfrak{g} * E_F.$$

This leads us to the following inequality:

$$|Prob[S_4] - Prob[S_3]| \leq Prob[L].$$

Similarly to what we did in the previous game, we want to bound $Prob[L]$ by constructing an adversary \mathcal{B}_2 against the Strong-CSIDH problem.

Definition 7. [*Adversary \mathcal{B}_2*] Consider now an adversary \mathcal{B}_2 which is given a Comp-CSIDH challenge (E_S, E_T) and is given access to a $\mathcal{D}_S(\cdot, \cdot)$ oracle. It runs Game 3., re-randomizing the challenge as follows: 1) it sets the ephemeral public key of type I and II oracles to $E_F = \mathfrak{r} * E_S$ for a random $\mathfrak{r} \xleftarrow{\$} cl(\mathcal{O})$; 2) it sets the ephemeral public key of type III oracles to $E_G = \mathfrak{r}' * E_T$ for a random $\mathfrak{r}' \xleftarrow{\$} cl(\mathcal{O})$.

In this game, since we embed the challenge in two ephemeral keys, all the static secret keys are known to the adversary. We need therefore to recognise two types of hash oracle queries:

1. hash queries for type II oracles of the form

$$\mathcal{P}_A \parallel \mathcal{P}_B \parallel E_A \parallel E_B \parallel E_F \parallel E_G \parallel \mathfrak{a} * E_G \parallel \mathfrak{f} * E_B \parallel \mathfrak{f} * E_G$$

given the knowledge of the static secret keys, the only information to be detected is whether $W_3 = \mathfrak{f} * E_G = \mathfrak{r}\mathfrak{s} * E_G$ or not. The answer can be obtained by performing the oracle query $\mathcal{D}_S(E_G, \overline{\mathfrak{r}}W_3)$;

2. hash queries for type III oracles of the form

$$\mathcal{P}_A \parallel \mathcal{P}_B \parallel E_A \parallel E_B \parallel E_F \parallel E_G \parallel W_1 \parallel W_2 \parallel \mathfrak{g} * E_F$$

given the knowledge of the static secret keys, the only information to be detected is whether $W_3 = \mathfrak{g} * E_F = \mathfrak{r}'\mathfrak{t} * E_F$ or not. The answer can again be obtained by performing the oracle query $\mathcal{D}_S(E_G, [\overline{\mathfrak{r}}]W_3)$.

If the Strong-CSIDH oracle outputs 1, then we output

$$E_Z = \mathfrak{r}^{-1}\mathfrak{r}'^{-1}W_3 = \overline{\mathfrak{r}}\overline{\mathfrak{r}'}\mathfrak{f}\mathfrak{g} * E_0 = \overline{\mathfrak{r}}\overline{\mathfrak{r}'}\mathfrak{r}\mathfrak{s}\mathfrak{r}'\mathfrak{t} * E_0 = \overline{\mathfrak{r}}\overline{\mathfrak{r}'}\mathfrak{r}'\mathfrak{s}\mathfrak{t} * E_0 = E_{ST}.$$

We have just described an adversary \mathcal{B}_2 which succeeds whenever event L occurs in Game 2. From this fact we get that

$$|\text{Prob}[S_4] - \text{Prob}[S_3]| \leq \text{Prob}[L] \leq \text{Adv}_{cl(\mathcal{O})}^{\text{St-CSIDH}}(\mathcal{B}_2)$$

the second element at the right-hand side of the inequality in Theorem 3.

Game 5. In this game a type II oracle (an initiator oracle whose intended peer is honest until the oracle accepts) chooses a random key E_K and modifies the key derivation hash oracle only if the intended peer is corrupted. Consider an oracle belonging to user \mathcal{P}_A with static secret key \mathfrak{a} and ephemeral secret key \mathfrak{f} : if the adversary corrupts the intended peer \mathcal{P}_B , the hash oracle will output $E : k$ whenever it is queried on input

$$\mathcal{P}_A \parallel \mathcal{P}_B \parallel E_A \parallel E_B \parallel E_F \parallel E_G \parallel \mathfrak{a} * E_G \parallel \mathfrak{f} * E_B \parallel \mathfrak{f} * E_G.$$

Analogously to what we did in Game 3, we define the following events:

- L : a query on the above input happens before the long-term secret key of any responder oracle is revealed. It follows that

$$|\text{Prob}[S_5] - \text{Prob}[S_4]| \leq \text{Prob}[L];$$

- L_B : same as L , but for a specific intended peer \mathcal{P}_B . Trivially, $\text{Prob}[L] = \sum_i \text{Prob}[L_i]$;
- C_B : a query on input

$$\mathcal{P}_A \parallel \mathcal{P}_B \parallel E_A \parallel E_B \parallel E_F \parallel E_G \parallel W_1 \parallel W_2 \parallel W_3 \quad W_2 = \mathfrak{f} * E_B = \mathfrak{b} * E_F$$

happens before user \mathcal{P}_B is corrupted; therefore we have $\text{Prob}[L_B] \leq \text{Prob}[C_B]$.

As we did in the previous games, we want to find an upper bound on $\text{Prob}[L]$.

Definition 8. [*Adversary \mathcal{B}_3*] Consider now an adversary \mathcal{B}_3 which is given a Comp-CSIDH challenge (E_S, E_T) and is given access to a $\mathcal{D}_S(\cdot, \cdot)$ oracle. It runs Game 4., it embeds the challenge as follows: 1) it sets the static public key of a uniformly-at-random user \mathcal{P}_B to $E_B = E_S$; 2) it sets the ephemeral public key of type I and II oracles whose intended peer is \mathcal{P}_B to $E_F = \mathfrak{r} * E_T$ for a random $\mathfrak{r} \xleftarrow{\$} \text{cl}(\mathcal{O})$.

If the adversary corrupts party \mathcal{P}_B , the game aborts, since the corresponding static secret key is unknown. We need therefore to recognise three types of queries made to the hash oracle:

1. hash queries for which \mathcal{P}_B acts as responder

$$\mathcal{P}_A \parallel \mathcal{P}_B \parallel E_A \parallel E_B \parallel E_F \parallel E_G \parallel \mathfrak{g} * E_A \parallel \mathfrak{b} * E_F \parallel \mathfrak{g} * E_F.$$

Given that both $\mathfrak{b} = \mathfrak{s}$ and \mathfrak{t} are unknown, the only information we cannot compute and that has to be detected is whether $W_2 = \mathfrak{b} * E_F = \mathfrak{b} * E_S$. The answer can be obtained by performing the oracle query $\mathcal{D}_S(E_F, W_2)$;

2. hash queries for which \mathcal{P}_B acts as initiator:

$$\mathcal{P}_B \parallel \mathcal{P}_A \parallel E_B \parallel E_A \parallel E_F \parallel E_G \parallel \mathfrak{b} * E_G \parallel \mathfrak{f} * E_A \parallel \mathfrak{f} * E_G$$

(note that, in this case, the second part of the challenge has not been embedded in E_F). The only information to be detected is whether $W_1 = \mathfrak{b} * E_F = \mathfrak{b} * E_S$, and the answer can be obtained by performing the oracle query $\mathcal{D}_S(E_G, W_1)$;

3. hash queries defining event C_B , i.e. made before the user \mathcal{P}_B is corrupted:

$$\mathcal{P}_A \parallel \mathcal{P}_B \parallel E_A \parallel E_B \parallel E_F \parallel E_G \parallel W_1 \parallel W_2 \parallel W_3 \quad W_2 = \mathfrak{f} * E_B = \mathfrak{b} * E_F$$

We have to recognise if W_2 is actually $\mathfrak{f} * E_B = \mathfrak{rt} * E_B$, and this can be done by checking if $\mathcal{D}_S(E_F, W_2) = 1$.

If the Strong-CSIDH oracle outputs 1 and realise that $W_2 = \mathfrak{s} * E_F = \mathfrak{st} * E_0$, then we output

$$E_Z = \mathfrak{r}^{-1} W_2 = \bar{\mathfrak{r}} \mathfrak{st} * E_0 = \bar{\mathfrak{r}} \mathfrak{st} * E_0 = E_{ST}.$$

We have just described an adversary \mathcal{B}_3 which succeeds whenever event L_B occurs in Game 5. L_B can occur only before \mathcal{P}_B is corrupt, and thus \mathcal{B}_3 's game would have gone through. We can therefore upper bound

$$Adv_{cl(\mathcal{O})}^{St-CSIDH}(\mathcal{B}_3) \geq \frac{1}{\mu} \sum_{i=1}^{\mu} Prob[C_I] \geq \frac{1}{\mu} \sum_{i=1}^{\mu} Prob[L_I] = \frac{1}{\mu} Prob[L]$$

from which we get that

$$|Prob[S_5] - Prob[S_4]| \leq Prob[L] \leq \mu \cdot Adv_{cl(\mathcal{O})}^{St-CSIDH}(\mathcal{B}_3)$$

the third and last element at the right-hand side of the inequality in Theorem 3.

Concluding the proof. Following from how we constructed each game in the proof, whenever the games do not abort because of adversarial corruption, the adversary is provided with a random session key, completely independent of every key and sent message. Therefore

$$Pr[S_5] = \frac{1}{2}.$$

We have seen in Game 1. that

$$Prob[\mathbf{break}_{\text{unique}}] \leq \frac{\mu k^2}{N}$$

and, due to the perfect correctness of the scheme,

$$Prob[\mathbf{break}_{\text{sound}}] = 0.$$

We can therefore exploit the bounds on adversarial winning probabilities to prove Theorem 3: given an adversary \mathcal{A} against protocol Π -SIDE, we have built

three adversaries $\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3$ against Strong-CSIDH such that \mathcal{A} wins with advantage $Adv_{\Pi-SIDE}^{AKE}(\mathcal{A})$ at most

$$\mu \cdot Adv_{cl(\mathcal{O})}^{St-CSIDH}(\mathcal{B}_1) + Adv_{cl(\mathcal{O})}^{St-CSIDH}(\mathcal{B}_2) + \mu \cdot Adv_{cl(\mathcal{O})}^{St-CSIDH}(\mathcal{B}_3) + \frac{\mu k^2}{N}$$




where μ is the number of participants to the protocol.

The tightness loss $L = \mathcal{O}(\mu)$ that we achieve in this security proof is optimal for simple protocols such as ours. The arguments adopted by Cohn-Gordon et al. [CCG⁺19] still hold in our setting and the adaptation is straightforward.

Appendix B

Symmetric Key Exchange with Full Forward Security and Robust Synchronization

Symmetric Key Exchange with Full Forward Security and Robust Synchronization

Colin Boyd¹ Gareth T. Davies²  Bor de Kock¹ 
Kai Gellert²  Tibor Jager² Lise Millerjord¹

¹NTNU – Norwegian University of Science and Technology, Trondheim, Norway

²Bergische Universität Wuppertal, Wuppertal, Germany

December 1, 2021

Abstract

We construct lightweight authenticated key exchange protocols based on pre-shared keys, which achieve *full* forward security and rely only on simple and efficient symmetric-key primitives. All of our protocols have rigorous security proofs in a strong security model, all have low communication complexity, and are particularly suitable for resource-constrained devices.

We describe three protocols that apply *linear* key evolution to provide different performance and security properties. *Correctness* in parallel and concurrent protocol sessions is difficult to achieve for linearly key-evolving protocols, emphasizing the need for assurance of availability alongside the usual confidentiality and authentication security goals. We introduce *synchronization robustness* as a new formal security goal, which essentially guarantees that parties can re-synchronize efficiently. All of our new protocols achieve this property.

Since protocols based on linear key evolution cannot guarantee that all concurrently initiated sessions successfully derive a key, we also propose two constructions with *non-linear* key evolution based on puncturable PRFs. These are instantiable from standard hash functions and require $O(C \cdot \log(|\text{CTR}|))$ memory, where C is

This work was supported by Deutscher Akademischer Austauschdienst (DAAD) and Norges forskningsråd (NFR) under the PPP-Norwegen programme. Colin Boyd and Lise Millerjord have been supported by NFR project number 288545. Tibor Jager and Gareth T. Davies have been supported by the European Research Council (ERC) under the European Union’s Horizon 2020 research and innovation programme, grant agreement 802823.

the number of concurrent sessions and $|\text{CTR}|$ is an upper bound on the total number of sessions per party. These are the first protocols to simultaneously achieve full forward security, synchronization robustness, and concurrent correctness.

Contents

1	Introduction	74
2	Preliminaries	80
2.1	Message Authentication Codes	80
2.2	Pseudorandom Functions	81
3	Authenticated Key Exchange in the Symmetric Setting	82
3.1	Execution Environment	82
3.2	AKE Security	84
3.3	Concurrent Execution Synchronization Robustness	86
4	Linear Key Evolution	88
4.1	Key Derivation via Linear Evolution	89
4.2	LP3: a Three-Message Protocol	91
4.2.1	AKE-M of LP3	93
4.2.2	Bounded Gap: Non-Concurrent Setting.	97
4.2.3	Bounded Gap: Concurrent Setting.	99
4.2.4	wSR of LP3.	100
4.3	LP2: A Two-Message Protocol with Fixed Roles	102
4.3.1	AKE-M of LP2	103
4.3.2	wSR of LP2	107
4.4	LP1: A One-Message Protocol with Fixed Roles	110
4.4.1	AKE-R of LP1	111
4.4.2	wSR of LP1	116
5	Non-Linear Key Evolution	118
5.1	Puncturable Pseudorandom Functions	118
5.2	PPRF-based Symmetric AKE	119
5.3	PP2: a Two-Message Protocol with Fixed Roles	120
5.3.1	AKE-M of PP2	120
5.3.2	SR of PP2	124
5.4	PP1: a One-Message Protocol with Fixed Roles	127
5.4.1	AKE-R of PP1	128
5.4.2	SR of PP1	128
5.5	Instantiation	129

1 Introduction

Authenticated key exchange protocols based on pre-shared long-term symmetric keys (PSK-AKE) enable two parties to use a previously established symmetric key, agreed upon via out-of-band communication, to (mutually) authenticate and derive a shared session key. Prominent examples of such protocols are the PSK modes of TLS 1.3 and prior TLS versions, but these examples still make use of public-key techniques for key derivation, even if authentication uses symmetric keys. PSK-AKE protocols can be significantly more efficient than classical public-key AKE protocols, particularly when they can be constructed exclusively based on symmetric key primitives (“symmetric AKE”) for both authentication and key derivation. Therefore such protocols are especially desirable for performance-constrained devices, such as battery-powered wireless IoT devices, where every computation and every transmitted bit has a negative impact on battery life. More generally, such protocols may be preferable in “closed-world” applications, such as industrial settings, where pre-sharing keys may be easier and more practical than deploying a public-key infrastructure. Furthermore, protocols based purely on symmetric-key techniques, such as hash functions and symmetric encryption, also achieve security against quantum attacks by adjusting security parameters appropriately.

Forward Security in Symmetric AKE Protocols. *Forward security* is today a standard security goal of key exchange protocols. It requires that past session keys remain secure, even if the secret long-term key material is compromised. Note that this is only achievable if past session keys are *not* efficiently computable from a current long-term key. Forward security is comparatively easily achievable if *public key* cryptography is used. For instance, a classical approach is to use *ephemeral* keys for key establishment, such as the Diffie-Hellman protocol or, more generally, a key encapsulation mechanism (KEM). *Independent* long-term keys can then be used for authentication via digital signatures or another KEM.

The only currently known way to avoid public key techniques and use only symmetric key primitives is based on the “*derive-then-evolve*” approach, where first a session key is derived from a long-term key, and then the long-term key is evolved. This key evolution prevents efficient re-computation of prior session keys which yields forward security. Both steps can be implemented with simple key derivation functions. There are two common ways to use this approach:

1. *Synchronized key evolution.* In this case, both parties evolve their long-term keys in “epochs”, e.g., once per day. Note that this approach cannot achieve “*full*” forward security, but only a weaker “*delayed*” form. This is because all session keys of the current epoch can be computed from the current long-term secret, so forward security only holds for session keys of past epochs. Moreover, this approach requires synchronized clocks between parties, even to achieve correctness. For many applications this seems impractical, in particular for cheap low-

performance devices, for which symmetric AKE protocols are particularly relevant.

2. *Triggered key evolution.* In this case the protocol ensures that both parties advance their key material during protocol execution. This approach directly achieves “full” forward security for every session, and therefore seems preferable. However, this apparently simple approach turns out to be much less trivial to realize than might be expected, because both parties must remain “in sync”, such that correctness is guaranteed even in presence of *concurrent* sessions or message loss due to network failures or *active attacks*. This approach has similarities with *ratcheting* [ACD19], but there are significant differences in our setting as discussed under *Related Work* below.

Concurrency and Key-Evolving Protocols. The possibility of running concurrent protocol sessions in parallel is a standard correctness requirement for protocols, and reflected in all common AKE security models, such as the BR and CK models [BR94, CK01] and their countless variants and refinements. The main technical challenge of key evolution is to achieve full forward security while maintaining *correctness* in the presence of parallel and concurrent protocol sessions.

Even if we assume that all parties are *honest* and that all messages are transmitted *reliably* (i.e., without being dropped because of an unreliable network or influence from an adversary) this is already highly non-trivial and we do not know of any currently existing forward-secure symmetric AKE protocol which achieves *correctness* and full forward security in such a setting. The difficulty is essentially that one session might advance a key “too early” for another concurrent session to be completed, which breaks correctness. No such difficulty appears in classical forward-secure public key protocols, since long-term keys are usually *static* and different sessions use independent randomness. So it turns out that, somewhat surprisingly, forward security and correctness is more difficult to achieve for symmetric AKE.

To complicate matters even further, note that the assumption of honest parties and reliable message transmission is very strong and may not be realistic for many applications. Therefore we actually want to achieve forward security and “synchronization robustness” in the presence of an adversary which intentionally aims to *break* synchronization, e.g., by adaptively dropping or re-ordering messages. Such an adversary is attacking *availability* properties of the AKE protocol, an important aspect of security usually omitted from key exchange security models. The development of techniques to ensure availability for stateful key exchange is an unsolved foundational problem.

Our Contributions. In this work we develop several new lightweight forward-secure symmetric AKE protocols with different efficiency and correctness properties. Table 1 summarizes the main security and efficiency properties of our new protocols. This in-

Table 1: Overview of our protocols and comparison to SAKE [ACF20]. The number in the protocol name indicates the total number of messages per protocol run, “R only” means that only the responder authenticates its communication partner. The third column considers the communication complexity, where **C** is the number of counter values that are sent, **M** the number of MACs, and **N** the number of nonces. **Sync. Rob.** indicates the achieved level of synchronization robustness, **Bd. Gap** whether the gap between two parties is bounded (for non-concurrent executions), **CC** whether concurrent correctness is achieved, and **FS** whether full forward security is achieved.

Protocol	Auth.	(C, M, N)	Sync. Rob.	Bd. Gap	CC	FS
SAKE (5) [ACF20]	mutual	(0,4,2) + ID	✗	✓	✗	✓
SAKE-AM (4) [ACF20]	mutual	(0,4,2) + ID	✗	✓	✗	✓
LP3	mutual	(3,3,2)	weak	✓	✗	✓
LP2	mutual	(2,2,0)	weak	✗	✗	✓
LP1	R only	(1,1,0)	weak	✗	✗	✓
PP2	mutual	(1,2,0)	full	✓	✓	✓
PP1	R only	(1,1,0)	full	✓	✓	✓

cludes the first protocols that provably achieve synchronization robustness, a formal availability security notion we introduce, and correctness in the presence of concurrent sessions. More concretely we achieve the following.

Security model. We describe a security model suited to forward-secure symmetric AKE capturing entity authentication (one-sided and mutual), indistinguishability of established keys, and forward security. Our model follows a standard approach for AKE protocols based on the Bellare-Rogaway model [BR94], adapted to the requirements of symmetric AKE with evolving keys.

Synchronization robustness. We formalize a new property called *synchronization robustness* (SR), which is trivially achieved for traditional AKE protocols with fixed long-term keys, but turns out to be a crucial feature for key-evolving protocols such as forward-secure symmetric AKE. Essentially, SR captures whether parties in a protocol can efficiently re-synchronize their states in order to complete a successful protocol run. This should even hold if an adversary controls the network and/or some of the parties.

We define two flavours. Both consider an active adversary that may execute arbitrary protocol sessions to manipulate the state of parties, and whose goal is to manipulate the state such that a subsequent protocol execution fails.

In *weak SR* the ‘target’ protocol session must then be executed without adversarial interaction (similar to the corresponding requirement in Krawczyk’s weak forward security [Kra05]). “Full” SR allows the adversary arbitrary queries between messages of the ‘target’ session, even to parties of the oracles involved in the ‘target’ session.

Linear key evolving protocols. We define the notion of *linear key evolution*, which makes the classical “derive-then-evolve” approach concrete. We argue that protocols based on linear key evolution can only achieve weak SR and cannot achieve concurrent correctness.

We construct three different protocols (LP1, LP2, LP3, cf. Table 1), all of which achieve weak SR. Most interestingly, LP3 even achieves a “bounded gap” property, which means that no active adversary in control of the network is able to force the state of two parties to differ by more than one key evolving step, so that a party is always able to catch up quickly, if necessary. For all three protocols we show that in a setting where concurrent runs between two parties are allowed, this number of steps required to catch up is bounded in the number of concurrent runs. To this end, we apply a new approach to precisely analyze the state machine of a protocol. Furthermore, we also show two extremely lightweight protocols LP1 and LP2, which provide one-sided and mutual authentication, respectively, and where the communication complexity is only one (resp. two) MAC and one (resp. two) counter value.

Full SR and concurrent correctness. This leads to the question of whether and how full synchronization robustness and concurrent correctness (CC) can be achieved. We propose the use of puncturable pseudorandom functions (PPRFs) to apply a “non-linear” key evolving strategy, and we construct two protocols PP1 and PP2, which both achieve full SR and CC.

Since PPRFs can be efficiently instantiated from cryptographic hash functions, both protocols are extremely lightweight. PP1 achieves one-sided authentication with a single counter value and a single MAC, PP2 mutual authentication with one counter and two MACs. Furthermore, while repeated puncturing PPRFs may lead to large secret keys [AGJ19, AGJ21] we take advantage of the stateful nature of symmetric AKE protocols to instantiate the PPRF such that secret key size is at most *logarithmic* in the number of sessions.

Hence, we offer a versatile catalogue of lightweight and forward-secure symmetric AKE protocols with significantly stronger correctness and security properties. This includes the first protocols to achieve concurrent correctness and full synchronization robustness, or weak SR with bounded gap. Which of these protocols is best for a particular application depends on the nature of the security and functionality requirements. Further, in LP3 the parties exchange nonces: we recognize that in some applications sufficient

randomness will not be available and so we prove the protocol secure for any nonce generation procedure, which could be random selection or (stateful) use of a counter.

Related Work. Bellare and Yee [BY03] analyzed forward security for symmetric-key primitives, specifically pseudo-random generation, message authentication codes and symmetric encryption. They provide constructions using key evolution which are similar to the linear key evolution that we employ, and their protocols use some techniques from key-evolving schemes such as prior work on forward-secure signatures [BM99]. Their work does not deal with key exchange.

Brier and Peyrin [BP10] gave a tree-based protocol for key establishment, with the stated aim of improving the DUKPT scheme defined in ANSI X9.24 [ANS09]. The idea in DUKPT is that the client device (payment terminal) is highly constrained in terms of memory, yet needs to derive a unique key per transaction from an original pre-shared key, by applying a PRF (based on Triple-DES) to a counter and the base derivation key. Their work involves formalizing the specific problem faced in the payment terminal setting, and their scheme assumes an incorruptible server: a far weaker security model than the one that we consider. A similar security assumption was used by Le et al. [LBdM07], who presented a protocol for use in the context of Radio Frequency Identification (RFID), where the server keeps two values of the key derivation key to deal with potential synchronization loss.

Li et al. [LSY+14] analyzed the pre-shared key ciphersuites of TLS 1.2, using an adaption of the ACCE model of Jager et al. [JKSS12]. In this setting, Li et al. presented a formalization of the prior AKE-style models, but where parties could share PSK material with other parties in addition to their long-term key pairs.

Dousti and Jalili [DJ15] presented a key exchange protocol called FORSAKES, which is based on synchronized time-based key evolution. Their protocol requires 3 messages and assumes perfect synchronicity of parties to achieve correctness, and as we have already mentioned their approach can only obtain *delayed forward security*. A discussion of delayed forward security and more generally the various challenges involved in defining forward security was given Boyd and Gellert [BG20].

The concept of evolving symmetric keys is reminiscent of Signal’s double ratchet [ACD19], a well-known example of a symmetric protocol with evolving keys. Signal employs a Diffie-Hellman-ratchet, which adds new key material at every step through multiple Diffie-Hellman exchanges along the way. At every step of this main ratchet a separate linear key evolving ratchet is ‘branched off’, which is similar to how linear evolution works in our protocols — however, a critical difference is that in our scenario we evolve the key shared across different sessions as opposed to evolving a key within one session as happens in the Signal protocol. It is this difference which leads to the complexity of managing synchronization between sessions which run concurrently. In addition to this difference, which anyway makes Signal unusable for our setting, use of Diffie-Hellman in the Signal ratchet means that there is a vector for quantum attacks,

while our protocol is purely based on symmetric primitives.

Another primitive conceptually similar to PPRFs is *puncturable encryption*, which was introduced by Green and Miers in 2015 [GM15], and has since led to several follow-up constructions of puncturable encryption [GHJL17, DJSS18, CRSS20, SSS⁺20, DGJ⁺21]. However, all those constructions rely on expensive public-key techniques (such as bilinear pairings) and are thus impractical in the context of this work.

Comparison with Avoine et al. [ACF20]. In Table 1 above we have mentioned two protocols named SAKE and SAKE-AM that were presented by Avoine et al. [ACF20] (henceforth ACF20). Their paper was the first to provide key exchange protocols that attain forward security via linear evolution. Their system assumptions are largely the same as ours, with the crucial difference that our models are equipped to capture parallel executions. The security model of ACF20 explicitly disallows concurrent sessions, which not only yields a weak security notion, but also sidesteps the major difficulty of achieving even correctness in the presence of concurrent sessions in key-evolving symmetric-key protocols. Indeed, the protocols from ACF20 completely break down when executed concurrently, allowing an adversary to prevent the parties from computing any session keys in future sessions. We consider this an unrealistic and impractical restriction for many applications. Therefore we introduce the new notion of synchronization robustness, which formally defines the ability of key-evolving protocols to deal with concurrent executions, including in adversarial environments.

We embrace the use of (explicit) counters to acquire linear key evolving protocols that are conceptually simpler and require fewer messages than those provided by ACF20, in a way that additionally provides (weak) synchronization robustness. In any protocol that uses PSK evolution to achieve forward security a party must update the key state after a successful protocol run, and in embedded devices this requires writing to persistent storage. Our protocols require the updating (writing) of one key and one counter per session, while SAKE and SAKE-AM require updating two keys. Since a sequentially evolving key can also be seen as an implicit counter, conceptually the distinction between counters and evolving keys seems to be minor. The storage overhead of our protocols compared to ACF20's protocols is the (usually 8-byte) counter, while the linear key evolving protocols in our paper and ACF20 require storage of two keys (usually 16 or 32 bytes).

We note that ACF20 remarked that the parties could use separate PSKs for concurrent executions, however this solution requires an a priori bound on the number of possible concurrent sessions that could occur and a corresponding multiplication in key storage: none of our protocols require this. Further, implementing their approach would require a modification of their protocols, since parties need to know which PSK to use, and the security of these modified protocols is not proven.

2 Preliminaries

We denote the security parameter as λ . For any $n \in \mathbb{N}$ let 1^n be the unary representation of n and let $[n] = \{1, \dots, n\}$ be the set of numbers between 1 and n . We write $x \xleftarrow{\$} \mathcal{S}$ to indicate that we choose element x uniformly at random from set \mathcal{S} . For a probabilistic polynomial-time algorithm \mathcal{A} we define $y \xleftarrow{\$} \mathcal{A}(a_1, \dots, a_n)$ as the execution of \mathcal{A} (with fresh random coins) on input a_1, \dots, a_n and assigning the output to y . The function $\text{NextOdd}(x)$ takes as input an integer and outputs the next odd integer greater than x , i.e. whichever element of $\{x+1, x+2\}$ is odd. Our protocols require the use of counters, and integer $|\text{CTR}|$ is the largest possible counter value. Furthermore, we write $[n] \times [n] \setminus (i^*, j^*)$ as a shorthand for $\{(i, j) \in [n]^2\} \setminus \{(i^*, j^*) \text{ with } i < j\}$.

2.1 Message Authentication Codes

Throughout this paper we assume that all MACs are deterministic. This is to reduce complexity in our proofs, however most MACs used in practice are deterministic [CMA05, GMA07, HMA08, ISO11, KMA16].

Definition 1 (Message Authentication Codes). A *message authentication code* consists of three probabilistic polynomial-time algorithms $\text{MAC} = (\text{KGen}, \text{Mac}, \text{Vrfy})$ with key space \mathcal{K}_{MAC} and the following properties:

- $\text{KGen}(1^\lambda)$ takes as input a security parameter λ and outputs a symmetric key $\mathbb{K}^{\text{MAC}} \in \mathcal{K}_{\text{MAC}}$;
- $\text{Mac}(\mathbb{K}^{\text{MAC}}, m)$ takes as input a key $\mathbb{K}^{\text{MAC}} \in \mathcal{K}_{\text{MAC}}$ and a message m . Output is a tag σ ;
- $\text{Vrfy}(\mathbb{K}^{\text{MAC}}, m, \sigma)$ takes as input a key $\mathbb{K}^{\text{MAC}} \in \mathcal{K}_{\text{MAC}}$, a message m , and a tag σ . Output is a bit $b \in \{0, 1\}$.

We call a message authentication code *correct* if for all m , we have

$$\Pr_{\mathbb{K}^{\text{MAC}} \xleftarrow{\$} \text{KGen}(1^\lambda)} [\text{Vrfy}(\mathbb{K}^{\text{MAC}}, m, \text{Mac}(\mathbb{K}^{\text{MAC}}, m)) = 1] = 1.$$

We define MAC security as strong existential unforgeability under chosen message attack, where the adversary has access to a verification oracle. In the more common version of this game, which we denote SEUF-CMA-1, the adversary must stop running after it submits its first verification query: this is a subcase of our more general definition. Bellare et al. [BGM04] showed that in the strong unforgeability case these definitions are equivalent up to a loss factor Q .

$G_{\text{MAC}}^{\text{SEUF-CMA-Q}}(\mathcal{A})$	$\mathcal{O}_{\text{Mac}}(m)$	$\mathcal{O}_{\text{Vrfy}}(m, \sigma)$
1: $\mathbb{K}^{\text{MAC}} \xleftarrow{\$} \text{KGen}(1^\lambda)$	7: $\sigma \leftarrow \text{Mac}(\mathbb{K}^{\text{MAC}}, m)$	10: $b \leftarrow \text{Vrfy}(m, \sigma)$
2: $\mathcal{Q}, \mathcal{V} \leftarrow \emptyset$	8: $\mathcal{Q} := \mathcal{Q} \cup \{(m, \sigma)\}$	11: if $b = 1$
3: $\mathcal{A}^{\mathcal{O}_{\text{Mac}}(\cdot), \mathcal{O}_{\text{Vrfy}}(\cdot, \cdot)}(1^\lambda)$	9: return σ	12: $\mathcal{V} := \mathcal{V} \cup \{(m, \sigma)\}$
4: if $\exists (m, \sigma) \in \mathcal{V} \setminus \mathcal{Q}$		13: return b
5: return 1		
6: return 0		

Figure 1: The SEUF-CMA- Q security experiment for message authentication code MAC. \mathcal{A} can make Q queries to $\mathcal{O}_{\text{Vrfy}}$.

Definition 2 (MAC Security). The advantage of an adversary \mathcal{A} in the SEUF-CMA- Q security experiment defined in Fig. 1 for message authentication code MAC is

$$\text{Adv}_{\text{MAC}}^{\text{SEUF-CMA-Q}}(\mathcal{A}) := \Pr \left[G_{\text{MAC}}^{\text{SEUF-CMA-Q}}(\mathcal{A}) = 1 \right].$$

2.2 Pseudorandom Functions

Definition 3 (Pseudorandom Functions). A *pseudorandom function* is a deterministic function $y = \text{PRF}(k, x)$ that takes as input some key $k \in \mathcal{K}_{\text{PRF}}$ and some element of a domain \mathcal{D}_{PRF} , and returns an element $y \in \mathcal{R}_{\text{PRF}}$.

$G_{\text{PRF}}^{\text{PRF-sec}}(\mathcal{A})$	$\mathcal{O}_f(x)$
1: $b \xleftarrow{\$} \{0, 1\}$	8: if $b = 1$
2: $k_{\text{PRF}} \xleftarrow{\$} \mathcal{K}_{\text{PRF}}$	9: $y \leftarrow f(k_{\text{PRF}}, x)$
3: $g \xleftarrow{\$} \{\mathcal{F} : \mathcal{D}_{\text{PRF}} \rightarrow \mathcal{R}_{\text{PRF}}\}$	10: else
4: $b^* \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_f(\cdot)}(1^\lambda)$	11: $y \leftarrow g(x)$
5: if $b^* = b$	12: return y
6: return 1	
7: return 0	

Figure 2: The PRF-sec security experiment for pseudorandom function PRF. $\{\mathcal{F} : \mathcal{D}_{\text{PRF}} \rightarrow \mathcal{R}_{\text{PRF}}\}$ is the set of all functions from \mathcal{D}_{PRF} to \mathcal{R}_{PRF} .

Definition 4 (PRF Security). The advantage of an adversary \mathcal{A} in the PRF-sec security

experiment defined in Fig. 2 for pseudorandom function PRF is

$$\text{Adv}_{\text{PRF}}^{\text{PRF-sec}}(\mathcal{A}) := \left| \Pr [G_{\text{PRF}}^{\text{PRF-sec}}(\mathcal{A}) = 1] - \frac{1}{2} \right|.$$

3 Authenticated Key Exchange in the Symmetric Setting

In this section we describe our model for authenticated key exchange with forward security in the symmetric setting. Our model follows the standard approach of AKE protocols based on the Bellare-Rogaway model [BR94], adapted to the requirements of symmetric AKE with evolving keys. This includes definitions for entity authentication (one-sided or mutual), key indistinguishability, and forward security. Furthermore, we define the property of synchronization robustness, which is a crucial feature for forward-secure symmetric key exchange protocols. Parts of our formalization take inspiration from the models of Jager et al. [JKSS12].

Differences to public-key AKE models. The most notable difference in the symmetric key setting is that each pair of parties is initialized with shared key material, which is specified before the actual protocol is run. This key material typically includes MAC keys or key derivation keys that have been established in an out-of-band communication (e.g., chosen during the manufacturing process of devices). In order to achieve forward-security via “key evolving techniques” in the symmetric key setting, we additionally have to provide (sessions of) parties with the ability to modify the party’s key material. As a consequence, the shared key material of two parties will not always be equal: While one party might evolve their key before preparing the first protocol message, the responder can (at the earliest) evolve after it has received that message.

3.1 Execution Environment

We consider a set of n parties $\{P_1, \dots, P_n\}$, where each party is a potential protocol participant. We refer to parties by P_i or by their label i if context is clear. Initially, each pair of parties (P_i, P_j) with $i \neq j$ share a common secret $\text{PSK}_{i,j}$, which is the initial key material generated during protocol initialization (e.g., MAC keys or key derivation keys). Note that this key material may evolve over time and that $\text{PSK}_{i,j}$ and $\text{PSK}_{j,i}$ may not necessarily be equal at all times.

We model parallel executions of a protocol by equipping each party i with $q \in \mathbb{N}$ session oracles π_i^1, \dots, π_i^q . Each session oracle represents a process that executes one single instance of the protocol. All oracles have access to the “global key material” PSK (including the ability to modify the key material PSK). Moreover, each oracle maintains an internal state consisting of the following variables:

Variable	Description
α	execution state $\in \{\text{uninitialized}, \text{negotiating}, \text{accept}, \text{reject}\}$
pid	identity of the intended partner $\in \{P_1, \dots, P_n\}$
ρ	role $\in \{\text{Initiator}, \text{Responder}\}$
sk	session key $\in \mathcal{K}_s \cup \perp$ for some session key space \mathcal{K}_s
κ	freshness of session key $\in \{\text{exposed}, \text{fresh}\}$
sid	session identifier
b	security bit $\in \{0, 1\}$

Additionally, we assume that each oracle has an additional temporary state variable, used to store ephemeral values or the transcript of messages. As initial state of the oracle, we have $\alpha = \text{uninitialized}$ and $\kappa = \text{fresh}$ and $b \stackrel{s}{\leftarrow} \{0, 1\}$. Note that pid and ρ are set when the adversary interacts with the respective oracles and that sid and sk are defined as the protocol/adversary progresses.

As usual, if an oracle derives a session key then it will enter the execution state `accept`. If an oracle reaches the execution state `reject`, then it will no longer accept any messages. Later on when we describe protocols, the event `Abort` will identify points at which this action would be triggered.

To begin any of the experiments in this section, the challenger initializes n parties $\{P_1, \dots, P_n\}$, with each pair of parties sharing symmetric key material PSK as specified by the protocol.

An adversary interacts with session oracles π_i^s by issuing the following queries. Several of these queries add output to an oracle transcript (defined below) which is available to the adversary.

- `NewSessionI`(π_i^s , pid) initializes a new initiator session for party P_i with intended partner pid. Specifically, this query assigns pid, $\rho = \text{Initiator}$ and $\alpha = \text{negotiating}$ to π_i^s , creates the first protocol message and adds this to transcript of π_i^s .
- `NewSessionR`(π_i^s , pid, m) initializes a new responder session for party P_i with $\rho = \text{Responder}$ and intended partner pid, and delivers a protocol message to this oracle. Specifically, it assigns pid and $\rho = \text{Responder}$ to π_i^s and processes message m . The message m and consequent protocol messages (if any) are added to its transcript, and the execution state is set to `negotiating`.
- `Send`(π_i^s , m) delivers message m to oracle π_i^s . This input message, and consequent protocol messages (if any), are added to this oracle's transcript.
- `RevealKey`(π_i^s) reveals session key sk_i^s and sets $\pi_i^s.\kappa$ to `exposed`.
- `Corrupt`(P_i, P_j) (issued to some oracle of P_i or P_j) returns $\text{PSK}_{i,j}$. If the query `Corrupt`(P_i, P_j) is the τ -th query issued by \mathcal{A} , we say that all oracles π_i with pid = j are τ -corrupted. (i.e., party P_i becomes τ -corrupted with respect to the other party P_j). An uncorrupted oracle is considered as $+\infty$ -corrupted.

- Test (π_i^s) chooses $sk_0 \xleftarrow{\$} \mathcal{K}_s$, sets $sk_1 = \pi_i^s.sk$ and returns sk_b . This oracle can only be queried once, and the query making this action is labelled τ_0 .

The adversary must call `NewSessionI` or `NewSessionR` in order to specify a role and intended partner identifier for each oracle it wishes to use. Afterwards, the adversary can use the `Send` query to convey messages to these oracles.

3.2 AKE Security

To define entity authentication we use *matching conversations* [BR94] for oracle partnering, which requires a definition of an oracle’s *transcript*: T_i^s is the sequence of all messages sent and received by π_i^s in chronological order. The standard definition of matching conversations, reflects that the party that sends the last message cannot be sure that the responder received that protocol message. We use this definition for entity authentication.

Note that an oracle π_i^s only has a transcript, T_i^s , if $\pi_i^s.\alpha \neq \text{uninitialized}$. Transcript T_j^t is a *prefix* of T_i^s if T_j^t contains at least one message and messages in T_j^t are identical to and in the same order as the first $|T_j^t|$ messages of T_i^s .

Definition 5 (Partial-transcript Matching conversations [JKSS12, Def. 3]). π_i^s has a partial-transcript matching conversation to π_j^t if

- T_j^t is a prefix of T_i^s and π_i^s has sent the last message(s), or
- $T_i^s = T_j^t$ and π_j^t has sent the last message(s).

However, standard matching conversations are not strong enough to define key indistinguishability in a symmetric setting and leave room for a trivial attack (intuitively, this is due to the “asynchronous evolution” of the global key material PSK). Consider an adversary that uses the above execution environment to execute some protocol between two (sessions of two) parties. The adversary forwards all messages but the last one between both parties. At this point the party that sent the last message must have reached the accept state and applied some one-way procedure to its key material PSK in order to achieve forward security. However, the other party still needs to receive the final message in order to derive the session key and update its version of the key material. If the adversary were now to use `Test` on the accepting party while using `Corrupt` on the other party, this leads to a trivial distinguishing attack in standard key indistinguishability games (e.g., in [JKSS12]). Hence, we need to introduce a slightly stronger notion of matching conversations to precisely capture when `Corrupt` queries are allowed: the conversation is only deemed to be matching if all messages were delivered.

Definition 6 (Guaranteed Delivery Matching conversations). π_i^s has a guaranteed delivery matching conversation to π_j^t if $T_i^s = T_j^t$.

As usual, we say that the adversary breaks entity authentication if it forces a fresh oracle to accept maliciously, and breaks key indistinguishability if it can distinguish from random an established key that it cannot trivially access.

Definition 7 (Entity Authentication). Let Π be a protocol. Let $G_{\Pi}^{\text{Ent-Auth}}(\mathcal{A})$ be the following game:

- The challenger initializes n parties and their keys;
- \mathcal{A} may issue queries to oracles `NewSessionI`, `NewSessionR`, `Send`, `RevealKey`, `Corrupt` and `Test` as defined above;
- Once \mathcal{A} has concluded, the experiment outputs 1 if and only if there exists an accepting oracle π_i^s such that the following conditions hold:
 1. both P_i (w.r.t. P_j) and intended partner P_j (w.r.t. P_i) were not corrupted before query τ_0 ;
 2. there is no unique π_j^t , with $\rho_i^s \neq \rho_j^t$, such that π_i^s has a partial-transcript matching conversation to π_j^t .

Define the advantage of an adversary \mathcal{A} in the Ent-Auth security experiment $G_{\Pi}^{\text{Ent-Auth}}(\mathcal{A})$ as

$$\text{Adv}_{\Pi}^{\text{Ent-Auth}}(\mathcal{A}) := \Pr [G_{\Pi}^{\text{Ent-Auth}}(\mathcal{A}) = 1].$$

An oracle π_i^s accepting in the above sense ‘accepts maliciously’.

Later on we separate the analysis of an initiator oracle accepting maliciously from a responder oracle accepting maliciously. Further, we will present protocols that only provide one-sided authentication: this requires separation of the AKE definition. To this end, we use the following notation:

$$\text{Adv}_{\Pi}^{\text{Ent-Auth}}(\mathcal{A}) = \text{Adv}_{\Pi}^{\text{Ent-Auth-I}}(\mathcal{A}) + \text{Adv}_{\Pi}^{\text{Ent-Auth-R}}(\mathcal{A}).$$

Definition 8 (Key Indistinguishability). Let Π be a protocol. Let $G_{\Pi}^{\text{Key-Ind}}(\mathcal{A})$ be the following game:

- The challenger initializes n parties and their keys;
- \mathcal{A} may issue queries to oracles `NewSessionI`, `NewSessionR`, `Send`, `RevealKey`, `Corrupt` and `Test` as defined above;
- Once \mathcal{A} has output (i, s, b') to indicate its conclusion, the experiment outputs 1 if and only if there exists an oracle π_i^s such that the following holds:
 1. π_i^s accepts, with a unique oracle π_j^t , such that π_i^s has a partial-transcript matching conversation to π_j^t , when \mathcal{A} issues its τ_0 -th query;

2. \mathcal{A} did not issue `RevealKey` to π_i^s nor π_j^t (so $\kappa_i^s = \text{fresh}$) and $\rho_i^s \neq \rho_j^t$;
3. P_i (w.r.t. P_j) is τ_i -corrupted and P_j (w.r.t. P_i) is τ_j -corrupted, with $\tau_i, \tau_j > \tau_0$;
4. at the point of query τ_j , oracle π_j^t had a guaranteed delivery matching conversation to π_i^s , and
5. $b' = \pi_i^s.b$.

Define the advantage of an adversary \mathcal{A} in the Key-Ind security experiment $G_{\Pi}^{\text{Key-Ind}}(\mathcal{A})$ as

$$\text{Adv}_{\Pi}^{\text{Key-Ind}}(\mathcal{A}) := \left| \Pr \left[G_{\Pi}^{\text{Key-Ind}}(\mathcal{A}) = 1 \right] - \frac{1}{2} \right|.$$

We assume that all adversaries in the Key-Ind game are valid, meaning that they terminate and provide an output in the correct format (i.e. $(i, s, b') \in [n] \times [q] \times \{0, 1\}$). Later on in our proofs we will follow the game-hopping strategy, and in doing so we will often simplify exposition by additionally assuming adversaries that do not trigger a trivial win (in the Key-Ind game or any subsequent modifications of this game).

We define AKE security in three flavors, distinguished by the level of entity authentication that is achieved by the protocol. An adversary breaks the AKE security of a protocol if it wins either the entity authentication game, or the key indistinguishability game.

Definition 9 (Authenticated Key Exchange). Let Π be a protocol. The advantage of an adversary \mathcal{A} in terms of AKE-M (mutual entity authentication), resp. AKE-I (initiator authenticates the responder), resp. AKE-R (responder authenticates the initiator) is defined as follows:

$$\begin{aligned} \text{Adv}_{\Pi}^{\text{AKE-M}}(\mathcal{A}) &:= \text{Adv}_{\Pi}^{\text{Key-Ind}}(\mathcal{A}) + \text{Adv}_{\Pi}^{\text{Ent-Auth-I}}(\mathcal{A}) + \text{Adv}_{\Pi}^{\text{Ent-Auth-R}}(\mathcal{A}). \\ \text{Adv}_{\Pi}^{\text{AKE-I}}(\mathcal{A}) &:= \text{Adv}_{\Pi}^{\text{Key-Ind}}(\mathcal{A}) + \text{Adv}_{\Pi}^{\text{Ent-Auth-I}}(\mathcal{A}). \\ \text{Adv}_{\Pi}^{\text{AKE-R}}(\mathcal{A}) &:= \text{Adv}_{\Pi}^{\text{Key-Ind}}(\mathcal{A}) + \text{Adv}_{\Pi}^{\text{Ent-Auth-R}}(\mathcal{A}). \end{aligned}$$

We do not specify any protocols that provide AKE-I alone in this paper, however it is defined here for completeness.

3.3 Concurrent Execution Synchronization Robustness

We now describe a novel property for key exchange protocols. The goal is to capture, in a formal manner, how *robust* a protocol is in the event of adversarial control of the network and/or some of the parties. We seek a definition that asks: after an adversary has had control of the communication network (by executing arbitrary `Send`

and NewSessionI/NewSessionR queries), can an honest protocol run be executed successfully? Specifically, if it is possible for the parties to lose synchronization (due to dropped messages or adversarial control) such that the parties cannot, in one protocol run, regain synchronization and compute the same key, then the protocol does not meet this property.

Our formalization follows the execution environment of the Ent-Auth and Key-Ind games described above, and allows an adversary to specify the protocol run (that it is attempting to ‘interrupt’) at the end of its execution by specifying two oracles. The challenger awards success if the two parties (specifically those two oracles) did not accept with the same session key. We define two flavours: a weaker version wSR in which the ‘target’ protocol run must be executed without any other messages interleaved, and a stronger version SR which allows arbitrary queries in between messages of the ‘target’ run, even to parties of the oracles involved in the ‘target’ run (though of course not to the two oracles).

We define an *honest protocol run* (via adversarial queries) between two oracles (with initial state set to *uninitialized*) as follows: a NewSessionI query was made that produced a protocol message m_1 , a NewSessionR query was made to the other oracle with input message m_1 , and if this query produced a protocol message m_2 then this value was given as a Send query to the other oracle, and so on, until all protocol messages have been created and delivered, if possible. In the event that any of these queries fails (returns \perp) the honest protocol run aborts. This honest protocol run can be thought of as a genuine attempt to execute a protocol execution.

Definition 10 ((weak) Synchronization Robustness). Let Π be a protocol. Let $G_{\Pi}^{\text{wSR}}(\mathcal{A})$ [with boxed text] or $G_{\Pi}^{\text{SR}}(\mathcal{A})$ [with dashed boxed text] be the following game:

- The challenger initializes n parties and their keys;
 - \mathcal{A} may issue queries NewSessionI, NewSessionR and Send as defined above;
 - Once \mathcal{A} has output (i, j, s, t) to indicate its conclusion, the experiment outputs 1 if and only if the following conditions hold:
 1. $\pi_i^s.\text{pid} = P_j$ and $\pi_j^t.\text{pid} = P_i$;
 2. $\pi_i^s.\text{sk} \neq \pi_j^t.\text{sk}$ or both values are \perp ;
 3. an honest protocol run was executed between π_i^s and π_j^t ;
 4. no queries were made by \mathcal{A} to interrupt the protocol execution between π_i^s and π_j^t .
- [with dashed boxed text]
4. no protocol messages in the transcripts of π_i^s and π_j^t were sent to any *other* oracles before they were delivered in the honest run.

Define the advantage of an adversary \mathcal{A} in the XX security experiment $G_{\Pi}^{\text{XX}}(\mathcal{A})$, for $\text{XX} \in \{\text{wSR}, \text{SR}\}$, as

$$\text{Adv}_{\Pi}^{\text{XX}}(\mathcal{A}) := \Pr [G_{\Pi}^{\text{XX}}(\mathcal{A}) = 1].$$

Notes on the definitions.

- Condition (4.) in the SR experiment states that for each genuine protocol message in the ‘target’ session, \mathcal{A} must not have provided this message to any other oracles before that message is delivered as part of the ‘target’ run. This prevents a trivial attack where \mathcal{A} delivers the final protocol message to two oracles: first to some other oracle than the ‘target’ oracle (but of the same party), then to the target oracle. When the (genuine) protocol message is delivered to the party for the second time the target oracle would abort. The parties have still created exactly one key for this genuine protocol run, and so condition (4.) essentially fixes the allowable output oracles as the ones that are processing protocol messages for the first time. (Replay attacks are not an issue in the wSR setting, since the execution must be uninterrupted and so any action made *after* that run has occurred has no impact on \mathcal{A} ’s chances of winning.)
- We do not allow Corrupt queries in this definition: in all of the protocols in this paper we assume pairwise shared key material (and specifically, no keys that are used by a party for communication with multiple other parties). This means that the adversary is not allowed to corrupt the parties in the target run with respect to each other, and that all other Corrupt queries will be of no benefit to an attacker. A similar argument follows for RevealKey queries. This simplifies the security experiment, while capturing the property that we wish to assess.
- In an alternative formulation of our definitions, the target protocol run could be performed by the challenger as an Execute query as seen in past literature [BPR00]. We avoid this approach for two reasons. First, in the SR case, in order to support interleaving, the adversary would have to call the challenger to initiate each stage of the execution (i.e. $k + 1$ times for a k -message protocol), and this is notationally awkward. Secondly and perhaps more importantly, our model allows the adversary to attempt to win its game in multiple protocol runs, and output the oracles which provides the best chance of success. Thus to retain the strength of the definition we would require *multiple* Execute queries, resulting in a model that looks very similar to what we have presented here.

4 Linear Key Evolution

In this section we present a number of protocols that use linear key evolution to derive session keys. All of these protocols achieve wSR. It is not hard to see that full robust-

ness (SR) is not achievable with linearly evolving protocols. To win the SR game the adversary makes a new complete protocol run after the target run has started and the session key is computed at one party, but before the session key is computed at the second party. This means that when the target session completes, the long-term key has already evolved and the key will be computed with the wrong version of the long-term key at the second party. Either the session will fail at the second party or the key will be different at the two parties. (There is a third case when the key is independent of the long-term key, but in that case the protocol fails to achieve key indistinguishability.)

The first linear key evolving protocol that we present, LP3, exchanges three short messages and has the attractive property of bounding the gap between the counters of the two parties. We present two further protocols which are even more efficient at the cost of some restrictions. LP2 is a two-message protocol but in order to maintain mutual authentication we insist that parties running LP2 have fixed their role as either initiator or responder (not an unreasonable assumption in many application scenarios). Our simplest protocol, LP1, has only a single message but, in addition to requiring fixed roles, like any other one-message protocol it can only achieve unilateral authentication. For all of our protocols we provide theorems guaranteeing authentication, key indistinguishability and weak synchronization robustness (wSR) security.

Syntax and Conventions. All protocols in this paper use message authentication codes to ensure that parties can only process messages that are meant for them. This means that party A stores a key K_{AB}^{MAC} (static) for MAC and key derivation key k_{AB}^{CTR} (evolving) to communicate with B , and K_{AC}^{MAC} and k_{AC}^{CTR} to communicate with C , etc. We describe the key derivation process in more detail in Sec. 4.1.

In LP2 and LP3, the party sending the protocol message includes its own identity in the MAC computation: this stops redirection/reflection attacks of protocol messages to the sending party. For LP1 this is not necessary since the sending party advances after sending its protocol message, meaning that its state is ahead and therefore it is unable to process messages that it has already sent.

4.1 Key Derivation via Linear Evolution

Before looking at specific protocols, we define what we mean by linear key evolution and present an abstract security definition for it. Party A holds a *key derivation key* k_{AB}^{CTR} for use in communication with party B , where the value CTR is an integer that defines the current key state, which is the number of times the key has evolved since its creation. After a party has participated in a key exchange run and computed a session key, it will apply a function Advnc to this key derivation key in order to obtain the next key derivation key and update the counter. This process is detailed in Fig. 3a. Looking ahead, forward security will be obtained if the function that computes $k_{AB}^{\text{CTR}+1}$ from k_{AB}^{CTR} is one-way: this stipulation ensures that an adversary corrupting a party has no way to

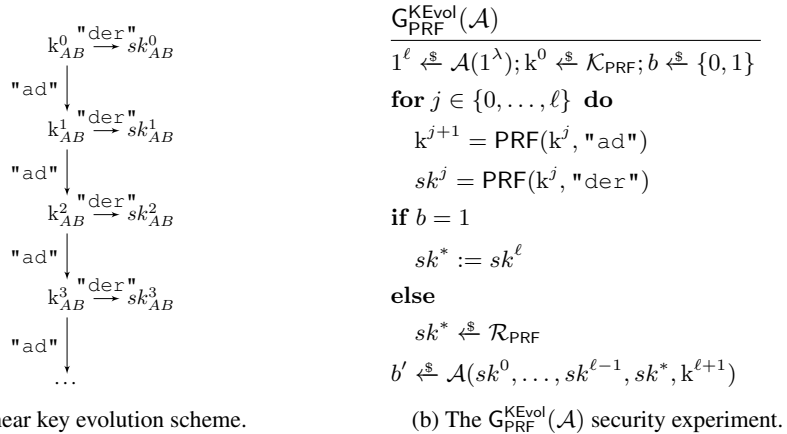


Figure 3: Linear key evolution and the corresponding experiment.

move upwards in the figure.

The initial “key derivation key” (KDK) is k_{AB}^0 . Subsequent KDKs are derived using a pseudorandom function PRF with $\mathcal{K}_{\text{PRF}} = \mathcal{R}_{\text{PRF}}$ as

$$k_{AB}^{i+1} = \text{PRF}(k_{AB}^i, \text{"ad"}) \quad (1)$$

and session keys are derived as

$$sk_{AB}^i = \text{PRF}(k_{AB}^i, \text{"der"})$$

where “ad” (“advance”) and “der” (“derive”) are constant labels used for domain separation.

Furthermore, for convenience, we define a function Advnc which performs multiple key derivations, if necessary. That is, $\text{Advnc}(k_{AB}^i, i, z)$ takes an i -th key derivation key for some counter i and an integer z , and applies PRF iteratively z times to obtain the $(i + z)$ -th KDK such that (1) is satisfied, and sets $i := i + z$. For example:

$$k_{AB}^{i+z}, i + z \leftarrow \text{Advnc}(k_{AB}^i, i, z).$$

Security. For the security proofs of our protocols it will be convenient to have an abstract security definition for such a key derivation scheme, which we will show to be implied by the security of the PRF. To this end, Fig. 3b represents a security experiment for the linear key evolution scheme that we describe. The adversary \mathcal{A} outputs an integer 1^ℓ (in unary, to ensure that the number ℓ is polynomially bounded for any efficient \mathcal{A}), and the adversary’s task is to distinguish sk^ℓ from random, when given all prior session keys $sk^0, \dots, sk^{\ell-1}$ and the ‘next’ key derivation key $k^{\ell+1}$.

Definition 11. The advantage of \mathcal{A} in the KEvol security experiment defined in Fig. 3b for pseudorandom function PRF is defined as

$$\text{Adv}_{\text{PRF}}^{\text{KEvol}}(\mathcal{A}) := \left| \Pr[b = b'] - \frac{1}{2} \right|.$$

We now prove the following straightforward theorem.

Theorem 1. *Let PRF be a pseudorandom function. For any adversary \mathcal{A} against the KEvol security of PRF, there exists an adversary \mathcal{B} against the PRF-sec of PRF such that*

$$\text{Adv}_{\text{PRF}}^{\text{KEvol}}(\mathcal{A}) \leq \ell \cdot \text{Adv}_{\text{PRF}}^{\text{PRF-sec}}(\mathcal{B}).$$

PROOF. The proof is a straightforward hybrid argument. Let H_0 be the original experiment. For $i \in \{1, \dots, \ell\}$ let H_i be an experiment which proceeds exactly like H_0 , except that k^j and sk^{j-1} are chosen uniformly random for all $j \leq i$, while all other keys are generated exactly as in the original experiment.

Let X_i denote the event that $b = b'$ in H_i . Then we have

$$\Pr[X_0] = \text{Adv}_{\text{PRF}}^{\text{KEvol}}(\mathcal{A}) \quad \text{and} \quad \Pr[X_\ell] = \frac{1}{2}$$

because the key sk^* is always uniformly random in H_ℓ , independent of the hidden bit b .

We now construct an adversary \mathcal{B} such that

$$|\Pr[X_i] - \Pr[X_{i+1}]| \leq \text{Adv}_{\text{PRF}}^{\text{PRF-sec}}(\mathcal{B}) \tag{2}$$

for all $i \in \{0, \dots, \ell-1\}$, which yields the claim. \mathcal{B} proceeds exactly like H_i , except that it defines $k^i = \mathcal{O}(\text{"ad"})$ and $sk^{i-1} = \mathcal{O}(\text{"deR"})$, where \mathcal{O} is the PRF oracle provided by the PRF security experiment. If \mathcal{A} outputs b' such that $b = b'$, then \mathcal{B} outputs 1, otherwise 0. Note that if \mathcal{O} implements a “real” PRF, then \mathcal{B} perfectly simulates H_i for \mathcal{A} , whereas if the oracle implements a “random” function, then it perfectly simulates H_{i+1} , which yields (2). \square

4.2 LP3: a Three-Message Protocol

Intuition. In Fig. 4 we present a three-message protocol called LP3, which puts a bound on how far initiator and responder can be out of sync, allows either party to initiate communications, and provides mutual authentication. After the first message is sent by an initiator, the responding party advances to catch up if they are behind. Then they respond, and the initiator does the same if they are behind. A third message confirms that both parties are now in sync again, and only after that a session key is established. We make use of state analysis proofs to show that the gap between the two states will be bounded even if messages are lost on the way (Lemma 6) and extend this proof to a

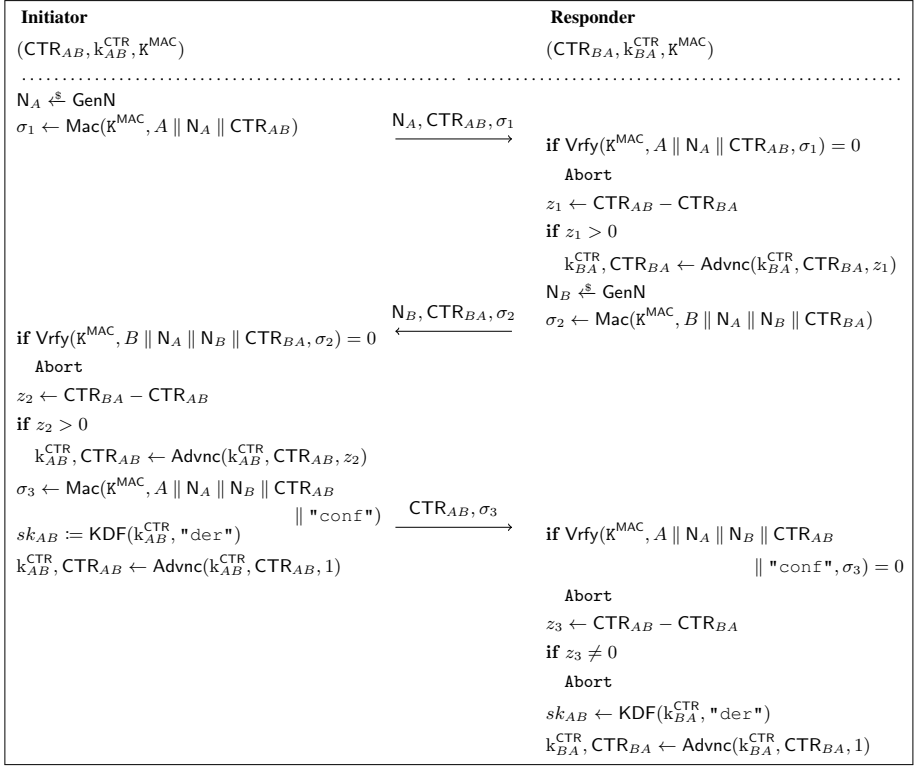


Figure 4: LP3, a three-message protocol.

scenario where concurrent runs are allowed (Lemma 7). We then show that the number of concurrent runs is a bound on the gap that can occur. We show in Theorem 8 that this also implies that the protocol achieves weak synchronization robustness (wSR). The protocol uses MACs and nonces to achieve mutual authentication (AKE-M). The functions Advnc and KDF, for PSK advancement and session key derivation respectively, are implemented using a PRF as described in Fig. 3a and Sec. 4.1.

State. The protocol uses nonces on both the initiating (N_A) and responding (N_B) sides. Local session state keeps track of these, and so it is only necessary to send N_A in the first protocol message and only N_B in the second message. The nonce generation procedure is denoted GenN, and this process could be, for example, random selection of a bitstring of some fixed length, or a (per-recipient) counter maintained by the party (note however that this counter is distinct from CTR, which tracks the key derivation key's

evolution stage). This choice depends on the application scenario, and this abstraction is for cleaner proofs. In the absence of a hardware RNG, random nonces require memory to be allocated for code of a software CSPRNG, while maintaining a counter requires writing to persistent storage (though such writes must be made anyway in linear key evolving protocols). The probability of a collision in random selection from \mathcal{NS} can be bounded by $\text{coll}[q_N, \text{GenN}] \leq \frac{q_N^2}{2|\mathcal{NS}|}$, and the collision probability of a (per-recipient) counter of size $|\mathcal{NS}|$ that is called q_N times is

$$\text{coll}[q_N, \text{GenN}] = \begin{cases} 0 & \text{for } 0 \leq q_N \leq |\mathcal{NS}| - 1, \\ 1 & \text{for } q_N \geq |\mathcal{NS}|. \end{cases}$$

We do not specify the additional counters required to make LP3 deterministic, so it is specified here as a protocol with random nonces.

4.2.1 AKE-M of LP3

Theorem 2 (AKE-M of LP3). *Let Π be the two-message protocol in Fig. 4, built using $\text{MAC} = \{\text{KGen}, \text{Mac}, \text{Vrfy}\}$ and PRF, with n parties. Then for any adversary \mathcal{A} against the AKE-M security of Π that makes a maximum of q queries that initiate new sessions for each party (with $q < |\text{CTR}|$), there exists an adversary $\mathcal{B}_{2.1}$ against the SEUF-CMA-Q of MAC and an adversary $\mathcal{B}_{2.2}$ against the KEvol security of KDF such that*

$$\text{Adv}_{\Pi}^{\text{AKE-M}}(\mathcal{A}) \leq n^2 \cdot \left(4\text{Adv}_{\text{MAC}}^{\text{SEUF-CMA-Q}}(\mathcal{B}_{2.1}) + 4\text{coll}[q, \text{GenN}] + q \cdot \text{Adv}_{\text{PRF}}^{\text{KEvol}}(\mathcal{B}_{2.2}) \right).$$

We form a bound for each of the three ways in which an adversary can break AKE-M security, namely Ent-Auth-R, Ent-Auth-I and Key-Ind, and then sum these bounds. There are two MAC security terms, for entity authentication of responder and initiator, and so $2 \cdot \text{Adv}_{\text{MAC}}^{\text{SEUF-CMA-Q}}(\mathcal{B}_{2.1})$ bounds these two terms by fixing $\mathcal{B}_{2.1}$ to be whichever of $\mathcal{B}_{2.1r}$ and $\mathcal{B}_{2.1i}$ has greater advantage.

We now give intuition regarding the Ent-Auth proofs. Note that since $q < |\text{CTR}|$, if there is no collision in the generation of \mathbb{N}_A for party A then all first protocol messages are unique. Further, after receiving a first protocol message a responder always generates a nonce, thus if no nonce collisions occur, sending the same first protocol message to two (or more) different responder oracles will result in distinct transcripts. This rules out the possibility of *multiple* oracles with non-unique matching conversations accepting, in either Ent-Auth-R or Ent-Auth-I. To conclude our proofs, we need to show that the only way an adversary can force an oracle to accept and for there not to exist any other oracle with a matching transcript, the adversary must forge a MAC message.

For Lemma 3 (Ent-Auth-R), a responder oracle accepts when it receives (what it believes to be) a third protocol message, and so to win there must not exist any oracle with the same partial transcript as this accepting oracle. This accepting oracle's transcript

must consist of (i) an input first protocol message, (ii) the resulting second protocol message, and (iii) the third protocol message that resulted in accept being reached. Since the MAC is calculated on both nonce values and the counter value, the first and third input messages to the accepting oracle must have been generated by an oracle of the communication partner. So the only viable route to victory if a forgery has not occurred is if these messages came from different oracles: and since the nonces are stored as part of the session state, this victory could only occur in the event of a nonce collision.

A similar argument applies for Lemma 4 (Ent-Auth-I), except now the accepting oracle computes a session key after receiving a valid second protocol message. Again that second protocol message must have come from a genuine invocation of NewSessionR by the intended partner if no forgery has occurred, and thus that oracle's transcript is a prefix of the accepting oracle's transcript.

Lemma 3 (Ent-Auth-R of LP3). *For any adversary \mathcal{A} , the probability that there exists an oracle with $\rho = \text{Responder}$ that accepts maliciously can be bounded by*

$$\text{Adv}_{\Pi}^{\text{Ent-Auth-R}}(\mathcal{A}) \leq n^2 \cdot \text{Adv}_{\text{MAC}}^{\text{SEUF-CMA-Q}}(\mathcal{B}_{2.1r}) + \text{coll}[q, \text{GenN}]$$

where all quantities are defined as stated in Theorem 2.

PROOF. We proceed using a sequence of games.

Game 0. This is the original Ent-Auth game.

$$\Pr [G_{\Pi}^{\text{Ent-Auth}}(\mathcal{A}) = 1] = \Pr [G_0^{\mathcal{A}} = 1]$$

Game 1. This game is the same as Game 0 except that we assume all nonces generated by NewSessionI and NewSessionR queries are different, and abort otherwise. The only way an adversary can notice the difference between these games is by making nonces collide, so we can write the following:

$$\Pr [G_0^{\mathcal{A}} = 1] \leq \Pr [G_1^{\mathcal{A}} = 1] + \text{coll}[q, \text{GenN}].$$

Game 2. In this game we guess which responder will be the first to accept maliciously and its partner identity, and abort if this guess is wrong. The game is the same as Game 1 except that the challenger guesses $(i^*, j^*) \xleftarrow{\$} [n] \times [n]$, and if an oracle π_i^s (for some s) accepts maliciously with $\pi_i^s.\rho \neq \text{Responder}$ or $i^* \neq i$ or $j^* \neq \pi_i^s.\text{pid}$, then the challenger aborts.

$$\Pr [G_1^{\mathcal{A}} = 1] = n^2 \cdot \Pr [G_2^{\mathcal{A}} = 1]$$

In order for an adversary to win without an abort in Game 2, it must make a responder oracle (of party i^*) accept maliciously, with no initiator oracle (of party j^*) having a matching conversation.

We construct a reduction $\mathcal{B}_{2.1r}$ that is playing against the SEUF-CMA- Q security of MAC that simulates the environment for an underlying adversary \mathcal{A} that attempts to win in game Game 2.

The reduction generates (initial) key derivation keys for all pairs of parties, and authentication keys for all pairs of parties except i^* and j^* . When responding to queries by \mathcal{A} regarding all other parties, the reduction will honestly provide messages as specified in the protocol specification and the Ent-Auth game. For any query made between oracles of parties i^* and j^* , the reduction will use its \mathcal{O}_{Mac} oracle and provide the received value in its simulation for \mathcal{A} . For example, to initialize initiator oracles π_{j^*} , the reduction calls $N \leftarrow \text{GenN}$, checks the current state counter $\text{CTR}_{j^*i^*}$ and calls $\mathcal{O}_{\text{Mac}}(j^* \| N \| \text{CTR}_{j^*i^*})$. If \mathcal{A} provides any value that it has not been given by a prior protocol message as input to a Send query from i^* to j^* or j^* to i^* , then the reduction sends this to its $\mathcal{O}_{\text{Vrfy}}$ oracle in the SEUF-CMA- Q game. The simulation of Game 2 is perfect and any win for \mathcal{A} directly corresponds to a valid signature forgery, so we can write

$$\Pr [G_2^{\mathcal{A}} = 1] \leq \text{Adv}_{\text{MAC}}^{\text{SEUF-CMA-}Q}(\mathcal{B}_{2.1r}).$$

Summing these terms gives the bound in the statement of Lemma 3. □

Lemma 4 (Ent-Auth-I of LP3). *For any adversary \mathcal{A} , the probability that there exists an oracle with $\rho = \text{Initiator}$ that accepts maliciously can be bounded by*

$$\text{Adv}_{\Pi}^{\text{Ent-Auth-I}}(\mathcal{A}) \leq n^2 \cdot \text{Adv}_{\text{MAC}}^{\text{SEUF-CMA-}Q}(\mathcal{B}_{2.1i}) + \text{coll}[q, \text{GenN}]$$

where all quantities are defined as stated in Theorem 2.

PROOF. Games 0, 1 and 2 are as in the proof of Lemma 3. The reduction $\mathcal{B}_{2.1i}$ is as in the proof of Lemma 3, except that now we are guessing which initiator will be the first to accept maliciously (and its intended partner), and so the abort occurs if a responder oracle accepts maliciously. The loss of n^2 incurred by selection of parties is the same. □

The proof of key indistinguishability is very similar to that of Lemma 12 and the game hops proceed following the same strategy. Again we use a reduction to the KEvol security of PRF.

Lemma 5 (Key-Ind of LP3). *For any adversary \mathcal{A} and (any fixed) entity authentication adversary $\mathcal{B}_{2.1}$, the probability that \mathcal{A} answers the Test challenge correctly can be bounded by*

$$\text{Adv}_{\Pi}^{\text{Key-Ind}}(\mathcal{A}) \leq \text{Adv}_{\Pi}^{\text{Ent-Auth}}(\mathcal{B}_{2.1}) + n^2 \cdot q \cdot \text{Adv}_{\text{PRF}}^{\text{KEvol}}(\mathcal{B}_{2.2})$$

where all quantities are defined as stated in Theorem 2.

PROOF. Let b' be the bit output by \mathcal{A} in each game, and b be the bit sampled as part of the Test query.

Game 0. This is the original Key-Ind game.

$$\Pr \left[G_{\Pi}^{\text{Key-Ind}}(\mathcal{A}) = 1 \right] = \Pr \left[G_0^{\mathcal{A}} = 1 \right]$$

Game 1. This game is the same as Game 0, except the challenger aborts and chooses $b' \xleftarrow{\$} \{0, 1\}$ if any oracle accepts maliciously.

$$\Pr \left[G_0^{\mathcal{A}} = 1 \right] \leq \Pr \left[G_1^{\mathcal{A}} = 1 \right] + \text{Adv}_{\Pi}^{\text{Ent-Auth-R}}(\mathcal{B}_{2.1})$$

At this stage, the oracle to which \mathcal{A} asks its Test query has a unique partner oracle with a matching conversation.

Game 2. This game is the same as Game 1, except the challenger guesses the two parties involved in the Test query via $(i^*, j^*) \xleftarrow{\$} [n] \times [n]$, and additionally guesses the counter value which identifies the key derivation key state of the session key in the Test query. If \mathcal{A} issues a $\text{Test}(\pi_{i^*}^s)$ query with $i^* \neq i$ or $j^* \neq \pi_{i^*}^s.\text{pid}$ (for some s), or the session key computed via $\text{Test}(\pi_{i^*}^s)$, i.e. $sk_{i^*j^*}$ is not equal to $\text{KDF}(k_{i^*j^*}^{\text{CTR}}, \text{"der"})$, then the challenger aborts. Note that the challenger does not guess which oracle of i^* the Test query will be made to, only the counter value linked to the session key in that query.

$$\Pr \left[G_1^{\mathcal{A}} = 1 \right] \leq n^2 \cdot q \cdot \Pr \left[G_2^{\mathcal{A}} = 1 \right]$$

At this stage, if the challenger has guessed correctly then the Test query will be asked to an oracle after the key derivation counter has been advanced a fixed number of times, and this oracle has unique partner with a matching conversation.

Game 3. This game is the same as Game 2, except that when the challenger runs KDF on the key derivation values used in the Test query, the challenger instead responds with a random key from the session key space. Noticing this change results in an adversary that is successful in the KEvol game for PRF, so we can write

$$\Pr \left[G_2^{\mathcal{A}} = 1 \right] = \Pr \left[G_3^{\mathcal{A}} = 1 \right] + \text{Adv}_{\text{PRF}}^{\text{KEvol}}(\mathcal{B}_{2.2}).$$

The reduction $\mathcal{B}_{2.2}$ is detailed in Fig. 5. $\mathcal{B}_{2.2}$ initially guesses the target parties in the Test session and the counter value associated with the Test session, as per the previous game hop.

For this reduction the challenger determines how an oracle should process each $\text{Send}(\pi_i^s)$ query using a label $\text{MsgNr} \in \{1, 2, 3\}$ as follows. If $\pi_i^s.\rho = \text{Responder}$ and the message is of the form N, CTR, σ then $\text{MsgNr} \leftarrow 1$. If $\pi_i^s.\rho = \text{Initiator}$ then $\text{MsgNr} \leftarrow 2$. If $\pi_i^s.\rho = \text{Responder}$ and the message is of the form CTR, σ then $\text{MsgNr} \leftarrow 3$. For all other query types or improperly formatted messages, the challenger returns \perp and sets $\pi_i^s.\alpha \leftarrow \text{reject}$

In the event that $\mathcal{B}_{2.2}$ is in the ‘real’ version of its own game, where it receives a genuine evaluation of the function KDF, $\mathcal{B}_{2.2}$ perfectly simulates Game 2 for \mathcal{A} , and otherwise it perfectly simulates Game 3.

At this stage, the Test query is asked on a key that is randomly chosen, and thus independent of the protocol and the security game. Consequently,

$$\Pr [G_3^{\mathcal{A}} = 1] = \frac{1}{2} \Rightarrow \text{Adv}_{\Pi}^3(\mathcal{A}) = 0.$$

□

4.2.2 Bounded Gap: Non-Concurrent Setting.

We will now prove that the “gap” between the state of the two parties in LP3 is bounded in the non-concurrent setting, that is:

Lemma 6. *Let A and B be respectively the initiator and the responder of a single — non-concurrent — LP3-run. Let δ_{AB} be the gap between A and B with respect to the evolution of the master keys of both parties. Then $\delta_{AB} \in \{-1, 0, 1\}$, assuming MAC-security.*

The messages in LP3 are counted in a natural way, as indicated in Fig. 6a. For this non-concurrent setting the proof is similar to [ACF20, Lemma 1]. Then the notation “ $(\text{CTR}_{AB}, \text{CTR}_{BA})$ ” means that, when the run ends, the last valid message received by A has index CTR_{AB} , and the last valid message received by B has index CTR_{BA} . We call a $(\text{CTR}_{AB}, \text{CTR}_{BA})$ -run a run where the last message received by A is message CTR_{AB} , and the last message received by B is message CTR_{BA} . By convention $\text{CTR}_{AB} = 0$ means that no message has been received by A . In Fig. 6b, we define the states to be the different values of δ_{AB} . The transitions are the possible messages. An example: if our protocol instance is in state $\delta_{AB} = -1$, and B responds to message 1 with message 2, i.e. transition $(2, 1)$ in the state diagram, the initiator will advance twice and the state will be $\delta_{AB} = 1$. A then sends the third message: transition $(2, 3)$ takes place and we end up in state $\delta_{AB} = 0$ since this third message will cause the responder to advance.

PROOF. We prove Lemma 6. The protocol is initialized with $\delta_{AB} = 0$ and the first step is sending message 1: either the message never reaches the responder, or the message is received correctly. In either case neither party advances, so $\delta_{AB} = 0$ — i.e. transition $(0, 1)$ in Fig. 6b is fired. If the protocol now terminates we end up in state 0, while sending and receiving message 2 would cause the initiator to advance, or in terms of the state diagram, fire $(2, 1)$ and transition to $\delta_{AB} = 1$.

Because we restrict ourselves to non-concurrent executions, the only possible option no matter the state is to advance with one message or terminate and start from $(0, 1)$.

Reduction $\mathcal{B}_{2.2}$ playing $G_{\text{KDF}}^{\text{KEvol}}(\mathcal{B}_{2.2})$

```

1 :  $i^*, j^* \xleftarrow{\$} [n]$ ;  $\text{CTR}^* \xleftarrow{\$} [q]$ 
2 : for  $i, j \in [n]$  do
3 :    $\text{CTR}_{ij} = \text{CTR}_{ji} \leftarrow 0$ 
4 :    $K_{ij}^{\text{MAC}} = K_{ji}^{\text{MAC}} \xleftarrow{\$} \mathcal{K}_{\text{MAC}}$ 
5 :   for  $n \times [n] \setminus (i^*, j^*)$  do
6 :      $k_{ij}^0 = k_{ji}^0 \xleftarrow{\$} \mathcal{K}_{\text{PRF}}$ 
7 :   output  $\text{CTR}^*$ 
8 :   receive( $sk^0, \dots, sk^{\text{CTR}^*-1}, sk^*, k^{\text{CTR}^*+1}$ )
9 :    $k_{i^*j^*}^{\text{CTR}^*+1} \leftarrow k^{\text{CTR}^*+1}$ 
10 :  $\mathcal{A}^{\text{oracles}}$ 
11 : When  $\mathcal{A}$  calls  $\text{Test}(\pi_i^s)$  do
12 :   if  $i \neq i^*$  or  $j^* \neq \pi_{i^*}^{s^*}.\text{pid}$ 
13 :     return Abort
14 :   return  $\pi_i^s.sk$ 
15 :    $(i^*, s^*, b') \xleftarrow{\$} \mathcal{A}(\pi_{i^*}^{s^*}.sk)$ 
16 :   return  $b'$ 

```

NewSessionI(π_i^s, pid)

```

17 :  $\pi_i^s.\rho \leftarrow \text{Initiator}$ 
18 :  $\pi_i^s.\alpha \leftarrow \text{negotiating}$ 
19 :  $\pi_i^s.\text{pid} \leftarrow \text{pid} \quad / \text{=j}$ 
20 :  $N \leftarrow \text{GenN}$ 
21 :  $\sigma_1 \leftarrow \text{Mac}(K^{\text{MAC}}, i \parallel N \parallel \text{CTR}_{ij})$ 
22 :  $m' \leftarrow N, \text{CTR}_{ij}, \sigma_1$ 
23 : return  $m'$ 

```

NewSessionR(π_i^s, pid, m)

```

24 :  $\pi_i^s.\rho \leftarrow \text{Responder}$ 
25 :  $\pi_i^s.\alpha \leftarrow \text{negotiating}$ 
26 :  $\pi_i^s.\text{pid} \leftarrow \text{pid}$ 
27 : do  $\text{Send}(\pi_i^s, m)$ 

```

$\text{Send}(\pi_i^s, m) \quad / \text{pid=j}$

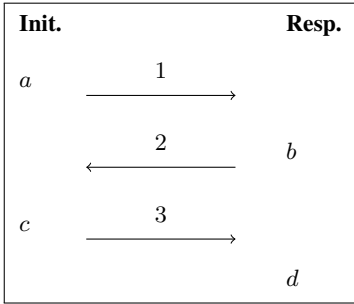
```

28 : if  $\text{MsgNr} = 1$ 
29 :   if  $\text{Vrfy}(K^{\text{MAC}}, j \parallel N_j \parallel \text{CTR}_{ji}, \sigma_1) = 0$ 
30 :     return Abort
31 :    $z_1 \leftarrow \text{CTR}_{ji} - \text{CTR}_{ij}$ 
32 :   if  $z_1 > 0$ 
33 :      $k_{ij}, \text{CTR}_{ij} \leftarrow \text{Advnc}(k_{ij}, \text{CTR}_{ij}, z_1)$ 
34 :    $N_i \leftarrow \text{GenN}$ 
35 :    $\sigma_2 \leftarrow \text{Mac}(K^{\text{MAC}}, i \parallel N_j \parallel N_i \parallel \text{CTR}_{ij})$ 
36 :    $m' \leftarrow N_i, \text{CTR}_{ij}, \sigma_2$ 
37 :   return  $m'$ 
38 : if  $\text{MsgNr} = 2$ 
39 :   if  $\text{Vrfy}(K^{\text{MAC}}, j \parallel N_i \parallel N_j \parallel \text{CTR}_{ji}, \sigma_2) = 0$ 
40 :     return Abort
41 :    $z_2 \leftarrow \text{CTR}_{ji} - \text{CTR}_{ij}$ 
42 :   if  $z_2 > 0$ 
43 :      $k_{ij}, \text{CTR}_{ij} \leftarrow \text{Advnc}(k_{ij}, \text{CTR}_{ij}, z_2)$ 
44 :    $\sigma_3 \leftarrow \text{Mac}(K^{\text{MAC}}, i \parallel N_i \parallel N_j \parallel \text{CTR}_{ij} \parallel \text{"conf"})$ 
45 :    $m' \leftarrow \text{CTR}_{ij}, \sigma_3$ 
46 :   return  $m'$ 
47 : else  $/ \text{MsgNr} = 3$ 
48 :   if  $\text{Vrfy}(K^{\text{MAC}}, j \parallel N_j \parallel N_i \parallel \text{CTR}_{ji} \parallel \text{"conf"}, \sigma_3) = 0$ 
49 :     return Abort
50 :    $z_3 \leftarrow \text{CTR}_{ji} - \text{CTR}_{ij}$ 
51 :   if  $z_3 \neq 0$ 
52 :     return Abort
53 :   if  $\text{MsgNr} \in \{2, 3\}$ 
54 :     if  $(i, j) = (i^*, j^*) \vee (j^*, i^*)$ 
55 :       if  $\text{CTR}_{ij} < \text{CTR}^*$ 
56 :          $\pi_i^s.sk \leftarrow sk^{\text{CTR}_{ij}}$ 
57 :       if  $\text{CTR}_{ij} = \text{CTR}^*$ 
58 :          $\pi_i^s.sk \leftarrow sk^*$ 
59 :          $s^* \leftarrow s$ 
60 :        $\text{CTR}_{ij} \leftarrow \text{CTR}_{ij} + 1$ 
61 :     else
62 :        $\pi_i^s.sk \leftarrow \text{KDF}(k_{ij}^{\text{CTR}}, \text{"der"})$ 
63 :        $k_{ij}^{\text{CTR}}, \text{CTR}_{ij} \leftarrow \text{Advnc}(k_{ij}^{\text{CTR}}, \text{CTR}_{ij}, 1)$ 
64 :        $\pi_i^s.\alpha \leftarrow \text{accept}$ 

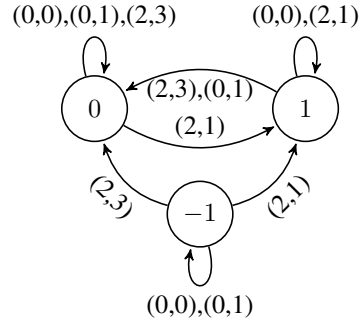
```

100

Figure 5: Reduction $\mathcal{B}_{2.2}$ for the proof of Lemma 5. If at any time \mathcal{A} causes an oracle to accept maliciously, then $\mathcal{B}_{2.2}$ simply does Abort. $\mathcal{B}_{2.2}$ provides \mathcal{A} with access to $\text{oracles} = \text{NewSessionI}(\cdot, \cdot, \cdot), \text{NewSessionR}(\cdot, \cdot, \cdot), \text{Send}(\cdot, \cdot, \cdot), \text{RevealKey}(\cdot), \text{Corrupt}(\cdot, \cdot)$, however RevealKey and Corrupt are omitted for space reasons: they are exactly as in Fig. 9 (Key-Ind proof of LP2).



(a) Numbering of states for the proofs of Lemmas 6 (1, 2, 3) and 7 (a, b, c, d).



(b) Synchronization state for LP3 in the non-concurrent setting.

Figure 6: Different states for LP3, and transitions between them.

Adding all possible transitions to the state diagram, we observe that there are no reachable states other than 0 and 1. Since the protocol does not have fixed roles we can reach a state -1 by changing roles after we reached state 1. From there, there are two transitions that bring us back to states 0 and 1. Since we assume that MACs cannot be forged, these are the only reachable states, thus $\delta_{AB} \in \{-1, 0, 1\}$ always holds. \square

4.2.3 Bounded Gap: Concurrent Setting.

We will now extend Lemma 6 to the concurrent setting.

Lemma 7. *Let A and B be respectively the initiator and the responder of C concurrent LP3-runs. Let δ_{AB} be the gap between A and B with respect to the evolution of the master keys of both parties. Then $-C \leq \delta_{AB} \leq 1 + C$, assuming MAC-security.*

To illustrate the (in a sense) multidimensional effect of concurrent runs on the protocol, we will now use a different message labelling convention. Fig. 6a defines the different states the protocol execution can be in. The state diagram in Fig. 7 now uses these four possible protocol states as diagram states — a message between state a and b is thus necessarily message 1. The internal state of the four ‘macro states’ in the diagram now represents the value of δ_{AB} .

Observe that for the transitions from a to b and from b to c , i.e. the sending of messages 1 and 2, respectively, the evolution of δ_{AB} depends on the actual value of a . For all transitions caused by message 3, the change is systematic:

1. Any transition from c to d will decrease δ_{AB} by 1;

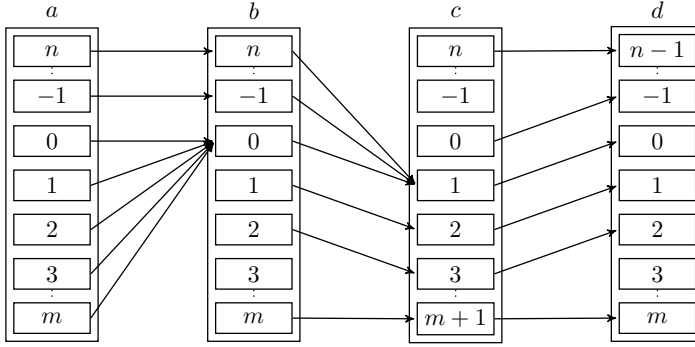


Figure 7: Synchronization state for LP3 in the concurrent setting.

2. any transition from b to c will increase δ_{AB} by at least 1.

Additionally there are two ‘resets’, since

3. any transition from a to b will set δ_{AB} to 0, if the gap is 1 or more;
4. any transition from b to c will set δ_{AB} to 1, if the gap is 0 or less.

PROOF. We prove Lemma 7. In Lemma 6, the normal range is shown to be $\delta_{AB} \in \{-1, 0, 1\}$. Extensions beyond this range are possible when the condition in 1. or 2. above occurs during a run, so each consecutive run can influence δ_{AB} with -1 or $+1$ at most. Since we assume MAC-security, the adversary cannot influence the protocol with messages other than those authentically sent during one of the runs. Inductively, we conclude $-C \leq \delta_{AB} \leq 1 + C$. \square

4.2.4 wSR of LP3.

We now argue that LP3 obtains weak synchronization robustness (wSR), the property that captures how well a protocol can recover from network errors and interleaving of protocol runs. In the wSR game the adversary can make arbitrary `NewSessionI`, `NewSessionR` and `Send` queries, and at its conclusion it outputs the identifiers of two oracles: it is said to win the wSR game if these oracles engaged in an uninterrupted protocol run but did not compute the same session key. As such, a proof of wSR must argue that whatever values of party state exist before the target protocol run occurs, neither of the parties will abort and both will arrive at the same session key.

Our general approach for proving robustness of all of the protocols in this paper is to separate adversaries that win the wSR game via forging a MAC value, and those that do

not produce a forgery during their execution. LP1 (Fig. 11) and LP2 (Fig. 8) have fixed roles and as a result the initiator's counter value must always be at least the size of the responder's counter value for the protocol to have correctness. Thus a MAC forgery can force the responding party's counter value to be arbitrarily large, and the target protocol run will cause at least one party to abort, and the adversary wins the wSR game. LP3, on the other hand, is actually not vulnerable in the sense of synchronization robustness if a MAC forgery does occur. This is due to LP3 being designed to have correctness for all starting (integer) counter values, since in any session, both parties can catch up from being arbitrarily far behind.

We formally prove this below, however to see this visually, consider Fig. 7 for the execution of a single protocol run, i.e. from a to d . For any initial state difference a , the state c after the second protocol message has been processed is always 1 (the initiator computes a session key and advances once), leading to state difference 0 after the responder processes the final protocol message (deriving a session key and advancing once).

Theorem 8 (wSR of LP3). *Let Π be the three-message protocol in Fig. 4, built using $\text{MAC} = \{\text{KGen}, \text{Mac}, \text{Vrfy}\}$ and PRF with n parties. Then for any adversary \mathcal{A} against the wSR security of Π , $\text{Adv}_{\Pi}^{\text{wSR}}(\mathcal{A}) = 0$.*

PROOF. The only places where Abort occurs in the protocol description (Fig. 4) are after MAC verification failures: in the target protocol session all messages are honestly generated so this cannot occur (assuming perfect correctness of MAC). As a result, the only route to victory in the wSR game for an adversary is to make the parties compute different session keys. This occurs if the parties compute session keys but have different counter values once all three protocol messages have been delivered and processed: following the notation and arguments in Lemma 7, this is the same as showing that $\delta = 0$ after a (2, 3) session for any starting delta value. More precisely, let A and B be the parties involved in the target session where A sends the first protocol message, let δ_{AB}^{pre} be the gap between A and B with respect to the evolution of the master keys of both parties and the point *before* the target session begins (i.e. before the adversary calls `NewSessionI` for the target session), and let $\delta_{AB}^{\text{post}}$ be the gap *after* the target session has occurred. Fig. 6b shows that $\delta_{AB}^{\text{post}} = 0$ for $\delta_{AB}^{\text{pre}} \in \{-1, 0, -1\}$, so to complete the proof we need to show that this also holds for arbitrary δ_{AB}^{pre} .

If $\delta_{AB}^{\text{pre}} \in \{1, 2, \dots\}$, i.e. CTR_{AB} is ahead of CTR_{BA} by $\delta_{AB}^{\text{pre}} = z_1$ steps, then the first protocol message processing by B results in B advancing its counter CTR_{BA} by δ_{AB}^{pre} steps, leading to state difference 0. This means that A will not advance on receiving the second protocol message and both parties will compute a session key for state CTR_{AB} and then advance once, and so $\delta_{AB}^{\text{post}} = 0$.

If $\delta_{AB}^{\text{pre}} \in \{-1, -2, \dots\}$, i.e. CTR_{BA} is ahead of CTR_{AB} by $-\delta_{AB}^{\text{pre}} = z_2$ steps, B does not advance in processing the first message, however A does advance by $-\delta_{AB}^{\text{pre}} = z_2$ steps on receiving the second protocol message. Again this leads to state difference

0 and here a session key is computed for state CTR_{BA} and then both parties advance once, so $\delta_{AB}^{\text{post}} = 0$.

This concludes the proof, since any initial state will lead to the target protocol run computing the same session key for the involved parties. \square

4.3 LP2: A Two-Message Protocol with Fixed Roles

In Fig. 8 we present a two-message protocol, LP2, with linear key evolution. The roles of initiator and responder are fixed, so the same party initiates every session: this is enforced by $\text{CTR}_{AB} \geq \text{CTR}_{BA}$ (for A initiating).

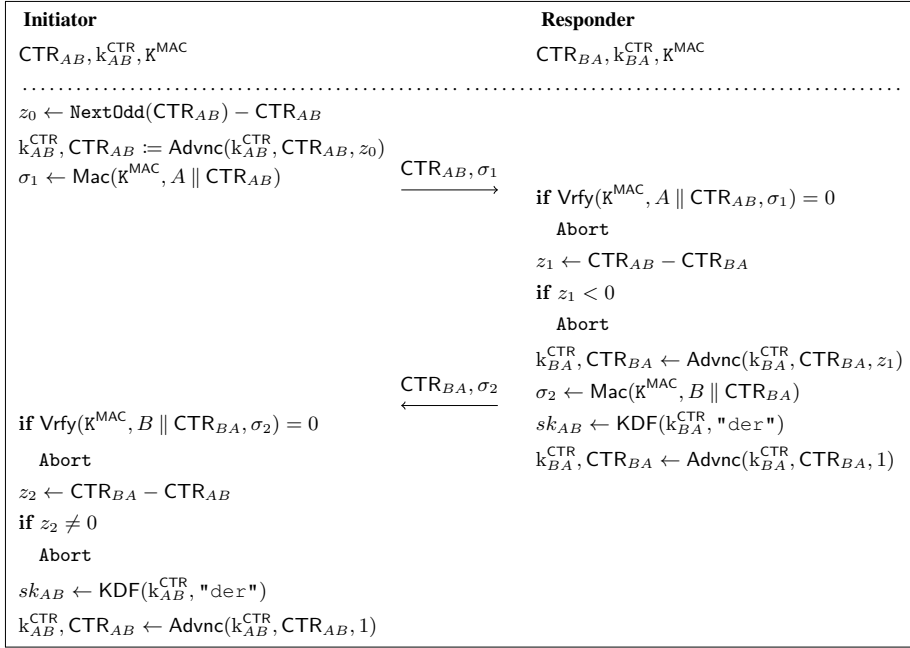


Figure 8: LP2, a two-message protocol with fixed roles.

Achieving weak synchronization robustness (wSR) is slightly more complicated in LP2 than it was in LP3. If we were to adapt LP3 to a two-message protocol by simply dropping the last message and having the responder accept (thus, deriving a session key and advancing its state), we could end up in a situation where we break the requirement that the responder should never advance past the state of the initiator. In this hypothetical protocol, the initiator will initiate the key exchange, but will not derive a session key until it has authenticated the responder. The responder, however, will authenticate the

initiator upon receiving the first protocol message (rather than waiting for a key confirmation message as in LP3) and produce the second protocol message, after which it will immediately derive a session key and advance its state. Thus, if this second protocol message is not delivered, the responder will have advanced its state, but the initiator has not, contradicting our requirement that $\text{CTR}_{AB} \geq \text{CTR}_{BA}$.

In order to avoid this in LP2, the initiator A will always advance to the next *odd* value of its counter at the beginning of each session. How many steps the initiator advances depends on what has happened earlier. If a complete session has been executed as A 's previous action, A starts by advancing once, so that its state counter is ahead of B . If in the previous session A never processed the second protocol message, A will advance twice at the beginning of the next session, in order to catch up to B and move ahead. The reasoning behind this is the separation of A 's counter set: if the counter is an even integer then A has most recently received a message (and derived a key), whereas if it is an odd integer then A most recently sent a (session opening) protocol message. In both cases, advancing to $\text{NextOdd}(\text{CTR}_{AB})$ will have the desired effect.

With this simpler protocol we are able to achieve most of the desired properties from SP3, but with a more lightweight protocol. Fixing the roles makes this possible, and this demonstrates the fine balance between forward security and (weak) synchronization robustness. In the event that the reduced communication complexity of LP2 compared to LP3 is desirable when choosing a protocol, but if the application demands that either party can initiate, it is possible to run LP2 in *duplex mode*. In duplex mode, both parties keep separate key derivation keys and counters for initiating and responding such that both parties can have both roles without violating the condition $\text{CTR}_{AB} \geq \text{CTR}_{BA}$.

4.3.1 AKE-M of LP2

Theorem 9 (AKE-M of LP2). *Let Π be the two-message protocol in Fig. 8, built using $\text{MAC} = \{\text{KGen}, \text{Mac}, \text{Vrfy}\}$, and PRF, with n parties. Then for any adversary \mathcal{A} against the AKE-M security of Π that makes a maximum of q queries that initiate new sessions for each party (with $q < \frac{|\text{CTR}|}{2}$), there exists an adversary $\mathcal{B}_{9,1}$ against the SEUF-CMA- Q of MAC and an adversary $\mathcal{B}_{9,2}$ against the KEvol security of PRF such that*

$$\text{Adv}_{\Pi}^{\text{AKE-M}}(\mathcal{A}) \leq n^2 \cdot \left(4 \cdot \text{Adv}_{\text{MAC}}^{\text{SEUF-CMA-}Q}(\mathcal{B}_{9,1}) + q \cdot \text{Adv}_{\text{PRF}}^{\text{KEvol}}(\mathcal{B}_{9,2}) \right).$$

We form a bound for each of the three ways in which an adversary can break AKE-M security, namely Ent-Auth-R, Ent-Auth-I and Key-Ind, and then sum these bounds. There are two MAC security terms, for entity authentication of responder and initiator, and so $2 \cdot \text{Adv}_{\text{MAC}}^{\text{SEUF-CMA-}Q}(\mathcal{B}_{9,1})$ bounds these two terms by fixing $\mathcal{B}_{9,1}$ to be whichever of $\mathcal{B}_{9,1r}$ and $\mathcal{B}_{9,1i}$ has greater advantage.

We now give intuition regarding the Ent-Auth proofs. Note that since $q < \frac{|\text{CTR}|}{2}$, all first protocol messages are unique, and thus the first message in every transcript is unique. Further, after receiving a first protocol message a responder always advances its

state (at least) once, and so delivering that same first protocol message to any oracle of the same responder party will trigger the $\text{if } z_1 < 0$ branch and result in **Abort**. This rules out the possibility of *multiple* oracles with non-unique matching conversations accepting. To conclude our proofs, we need to show that the only way an adversary can force an oracle to accept and for there not to exist any other oracle with a matching transcript, the adversary must forge a MAC message.

For Lemma 10 (Ent-Auth-R), a responder oracle accepts when it receives a first protocol message, and so to win there must not exist any oracle with the same partial transcript as this accepting oracle. Since the MAC is calculated on the counter value and the communicating parties' identities, the input message to the accepting oracle must have been generated by an oracle of the communication partner (in which case there is a matching conversation and the adversary has not won) or the adversary has produced a MAC forgery (in the SEUF-CMA- Q sense).

A similar argument applies for Lemma 11 (Ent-Auth-I), except now the accepting oracle computes a session key after receiving a valid second protocol message. Again if no forgery has occurred that second protocol message must have come from a genuine invocation of `NewSessionR` by the intended partner (with the first protocol message of the accepting oracle provided as input), and thus that oracle's transcript is a prefix of the accepting oracle's transcript. Specifically, if an initiator session was instantiated after the first protocol message of the accepting oracle was produced then $z_2 \neq 0$ and we have a contradiction since this oracle could not then reach **accept**.

Lemma 10 (Ent-Auth-R of LP2). *For any adversary \mathcal{A} , the probability that there exists an oracle with $\rho = \text{Responder}$ that accepts maliciously can be bounded by*

$$\text{Adv}_{\Pi}^{\text{Ent-Auth-R}}(\mathcal{A}) \leq n^2 \cdot \text{Adv}_{\text{MAC}}^{\text{SEUF-CMA-Q}}(\mathcal{B}_{9.1r})$$

where all quantities are defined as stated in Theorem 9.

PROOF. We proceed using a sequence of games.

Game 0. This is the original Ent-Auth game.

$$\Pr [G_{\Pi}^{\text{Ent-Auth}}(\mathcal{A}) = 1] = \Pr [G_0^{\mathcal{A}} = 1]$$

Game 1. In this game we guess which responder will be the first to accept maliciously and its partner identity, and abort if this guess is wrong. The game is the same as Game 0 except that the challenger guesses $(i^*, j^*) \xleftarrow{\$} [n] \times [n]$, and if an oracle π_i^s (for some s) accepts maliciously with $\pi_i^s.\rho \neq \text{Responder}$ or $i^* \neq i$ or $j^* \neq \pi_i^s.\text{pid}$, then the challenger aborts.

$$\Pr [G_0^{\mathcal{A}} = 1] = n^2 \cdot \Pr [G_1^{\mathcal{A}} = 1]$$

We construct a reduction $\mathcal{B}_{9.1r}$ that is playing against the SEUF-CMA- Q security of MAC that simulates the environment for an underlying adversary \mathcal{A} that attempts to win

in game Game 1. The reduction generates (initial) key derivation keys for all pairs of parties, and authentication keys for all pairs of parties except i^* and j^* . When responding to queries by \mathcal{A} regarding all other parties, the reduction will honestly provide messages as specified in the protocol specification and the Ent-Auth game. For any query made between oracles of parties i^* and j^* , the reduction will use its \mathcal{O}_{Mac} oracle and provide the received value in its simulation for \mathcal{A} . For example, to initialize initiator oracles π_{j^*} , the reduction checks the current state counter $\text{CTR}_{j^*i^*}$ and calls $\mathcal{O}_{\text{Mac}}(j^* \parallel \text{CTR}_{j^*i^*})$. If \mathcal{A} provides any value that it has not been given as an initialization query as input to a `NewSessionR` query from i^* to j^* , then the reduction sends this to its $\mathcal{O}_{\text{Vrfy}}$ oracle in the SEUF-CMA- Q game. The simulation of Game 1 is perfect and any win for \mathcal{A} directly corresponds to a valid signature forgery, so we can write

$$\Pr [G_1^{\mathcal{A}} = 1] \leq \text{Adv}_{\text{MAC}}^{\text{SEUF-CMA-}Q}(\mathcal{B}_{9.1r}).$$

Summing these terms gives the bound in the statement of Lemma 10. □

The second proof is very similar, and considers malicious acceptance by an initiator, i.e. as a result of a full protocol run of two messages. We only detail significant changes and note that our term collection is exactly the same.

Lemma 11 (Ent-Auth-I of LP2). *For any adversary \mathcal{A} , the probability that there exists an oracle with $\rho = \text{Initiator}$ that accepts maliciously can be bounded by*

$$\text{Adv}_{\Pi}^{\text{Ent-Auth-I}}(\mathcal{A}) \leq n^2 \cdot \text{Adv}_{\text{MAC}}^{\text{SEUF-CMA-}Q}(\mathcal{B}_{9.1i})$$

where all quantities are defined as stated in Theorem 9.

PROOF. Games 0 and 1 are exactly as in the proof of Lemma 10. The reduction is as in the proof of Lemma 10, except that now we are guessing which initiating party will be the first to accept maliciously (and its intended partner), and so the abort occurs if a responder oracle accepts maliciously. The loss of n^2 incurred by selection of parties is the same.

Again, the next step is a reduction that plays against SEUF-CMA- Q of the MAC scheme MAC. However, the reduction of course must behave slightly differently, since it must send to its own $\mathcal{O}_{\text{Vrfy}}$ oracle any message that was called as a `Send` query for the targeted initiator oracle, but which was not given as an output protocol message by a `NewSessionR` query (to responder oracle j^*). Further, we need to ensure that the forgery attempt is on an oracle that does not have a matching conversation with any others: in the proof of Lemma 10 this was straightforward since there was only one unique message in question, but here the transcripts that we are interested in consist of (up to) two flows between each oracle. This is just a matter of bookkeeping, and as before $\mathcal{B}_{9.1i}$ forwards all attempted forgeries to its own verification oracle, so any query that would have caused \mathcal{A} to win the entity authentication game (in the simulation that \mathcal{A} is experiencing) also corresponds to success in the game that $\mathcal{B}_{9.1i}$ is playing. □

Lemma 12 (Key-Ind of LP2). *For any adversary \mathcal{A} and (any fixed) entity authentication adversary $\mathcal{B}_{9.1}$, the probability that \mathcal{A} answers the Test challenge correctly can be bounded by*

$$\text{Adv}_{\Pi}^{\text{Key-Ind}}(\mathcal{A}) \leq \text{Adv}_{\Pi}^{\text{Ent-Auth}}(\mathcal{B}_{9.1}) + n^2 q \cdot \text{Adv}_{\text{PRF}}^{\text{KEvol}}(\mathcal{B}_{9.2}).$$

where all quantities are defined as stated in Theorem 9.

PROOF. Let b' be the bit output by \mathcal{A} in each game, and b be the bit sampled as part of the Test query.

Game 0. This is the original Key-Ind game.

$$\Pr [G_{\Pi}^{\text{Key-Ind}}(\mathcal{A}) = 1] = \Pr [G_0^{\mathcal{A}} = 1]$$

Game 1. This game is the same as Game 0, except the challenger aborts and chooses $b' \xleftarrow{\$} \{0, 1\}$ if any oracle accepts maliciously.

$$\Pr [G_0^{\mathcal{A}} = 1] \leq \Pr [G_1^{\mathcal{A}} = 1] + \text{Adv}_{\Pi}^{\text{Ent-Auth}}(\mathcal{B}_{9.1})$$

At this stage, the oracle to which \mathcal{A} asks its Test query has a unique partner oracle with a matching conversation.

Game 2. This game is the same as Game 1, except the challenger guesses the two parties involved in the Test query via $(i^*, j^*) \xleftarrow{\$} [n] \times [n]$, and additionally guesses the counter value which identifies the key derivation key state of the session key in the Test query. Note that session keys are only derived for counters equal to odd integers in $\{1, 3, \dots, q\}$, so the challenger chooses $\text{CTR}_{i^* j^*}^*$ from this set. If \mathcal{A} issues a $\text{Test}(\pi_{i^*}^s)$ query with $i^* \neq i$ or $j^* \neq \pi_{i^*}^s.\text{pid}$ (for some s), or the session key computed via $\text{Test}(\pi_{i^*}^s)$, i.e. $sk_{i^* j^*}$ is not equal to $\text{KDF}(k_{i^* j^*}^{\text{CTR}^*}, \text{"der"})$, then the challenger aborts. Note that the challenger does not guess which oracle of i^* the Test query will be made to, only the counter value linked to the session key in that query.

$$\Pr [G_1^{\mathcal{A}} = 1] \leq n^2 \cdot q \cdot \Pr [G_2^{\mathcal{A}} = 1]$$

At this stage, if the challenger has guessed correctly then the Test query will be asked to an oracle after the key derivation counter has been advanced a fixed number of times, and this oracle has unique partner with a matching conversation.

Game 3. This game is the same as Game 2, except that when the challenger runs KDF on the key derivation values used in the Test query, the challenger instead responds with

a random key from the session key space. Noticing this change results in an adversary that is successful in the KEvol game for PRF, so we can write

$$\Pr [G_2^{\mathcal{A}} = 1] = \Pr [G_3^{\mathcal{A}} = 1] + \text{Adv}_{\text{PRF}}^{\text{KEvol}}(\mathcal{B}_{9.2}).$$

The reduction $\mathcal{B}_{9.2}$ is detailed in Fig. 9. $\mathcal{B}_{9.2}$ initially guesses the target parties in the Test session and the counter value associated with the Test session, as per the previous game hop. In this reduction, instances of **return Abort** indicate that the adversary has done something that the reduction cannot respond to (since it has been ruled out by earlier game hops) and thus the reduction must abort. Instances of **return \perp** indicate that the adversary has done something that it is not allowed to do, for example doing **RevealKey** on a non-existent session key or delivering an invalid message to a(n oracle of a) party. In this case the reduction stops the action of the query, and if instructed by the model in Section 3, sets the execution state α of the queried oracle to **reject**. Finally, **return x** indicates that $\mathcal{B}_{9.2}$ gives value x to \mathcal{A} as a result of one of \mathcal{A} 's queries.

In the event that $\mathcal{B}_{9.2}$ is in the ‘real’ version of its own game, where it receives a genuine evaluation of the function KDF, $\mathcal{B}_{9.2}$ perfectly simulates Game 2 for \mathcal{A} , and otherwise it perfectly simulates Game 3.

At this stage, the Test query is asked on a key that is randomly chosen, and thus independent of the protocol and the security game. Consequently,

$$\Pr [G_3^{\mathcal{A}} = 1] = \frac{1}{2} \Rightarrow \text{Adv}_{\Pi}^3(\mathcal{A}) = 0.$$

□

4.3.2 wSR of LP2

We now argue that LP2 obtains weak synchronization robustness (wSR). A proof of wSR must argue that whatever the adversary does before the target protocol run occurs, during the target protocol run itself neither of the parties will abort and both will arrive at the same session key. Protocol LP2, like LP1 (Fig. 11), only has correctness when the initiator’s counter is at least the size of the responder’s counter, i.e. $\text{CTR}_{AB} \geq \text{CTR}_{BA}$ – this inequality is guaranteed in our protocol by MAC security. By inspection, if an adversary forges a MAC on $A \parallel \text{CTR}_{AB}$ for some CTR_{AB} larger than the current value of CTR_{BA} and delivers this MAC as part of a protocol message to an oracle of B , then any subsequent protocol run will cause B to **Abort** and thus will be a winning target protocol run for this adversary.

The formal proof is below, but here we outline the proof idea. We first define an event E_{13} that is triggered if the adversary in the wSR game forges a MAC, i.e. produces a message-tag pair that verifies correctly that it has not seen before, and the challenger aborts if this occurs: bounding this event is of course straightforward. Then, we must argue that if a MAC forgery has not occurred then there are in fact no viable routes to

Reduction $\mathcal{B}_{9,2}$ playing $G_{\text{KDF}}^{\text{KEvol}}(\mathcal{B}_{9,2})$

```

1 :  $i^*, j^* \xleftarrow{\$} [n]$ ;  $\text{CTR}^* \xleftarrow{\$} \{1, 3, \dots, q\}$ 
2 : for  $i, j \in [n]$  do
3 :    $\text{CTR}_{ij} = \text{CTR}_{ji} \leftarrow 0$ 
4 :    $K_{ij}^{\text{MAC}} = K_{ji}^{\text{MAC}} \xleftarrow{\$} \mathcal{K}_{\text{MAC}}$ 
5 :   for  $x \in [n] \setminus \{i^*, j^*\}$  do
6 :      $k_{ij}^0 = k_{ji}^0 \xleftarrow{\$} \mathcal{K}_{\text{PRF}}$ 
7 :   output  $\text{CTR}^*$ 
8 :   receive  $(sk^0, \dots, sk^{\text{CTR}^*-1}, sk^*, k^{\text{CTR}^*+1})$ 
9 :    $k_{i^*j^*}^{\text{CTR}^*+1} \leftarrow k^{\text{CTR}^*+1}$ 
10 :   $\mathcal{A}^{\text{oracles}}$ 
11 :  When  $\mathcal{A}$  calls  $\text{Test}(\pi_i^s)$  do
12 :    if  $i \neq i^*$  or  $j^* \neq \pi_i^{s*}.\text{pid}$ 
13 :      return Abort
14 :    return  $\pi_i^s.sk$ 
15 :   $(i^*, s^*, b') \xleftarrow{\$} \mathcal{A}(\pi_i^s.sk)$ 
16 :  return  $b'$ 

```

NewSessionI(π_i^s, pid)

```

17 :  $\pi_i^s.\rho \leftarrow \text{Initiator}$ 
18 :  $\pi_i^s.\alpha \leftarrow \text{negotiating}$ 
19 :  $\pi_i^s.\text{pid} \leftarrow \text{pid} \quad / = j$ 
20 :  $z_0 \leftarrow \text{NextOdd}(\text{CTR}_{ij}) - \text{CTR}_{ij}$ 
21 :  $k_{ij}^{\text{CTR}}, \text{CTR}_{ij} \leftarrow \text{Advnc}(k_{ij}^{\text{CTR}}, \text{CTR}_{ij}, z_0)$ 
22 :  $\sigma_1 \leftarrow \text{Mac}(K_{ij}^{\text{MAC}}, i \parallel \text{CTR}_{ij})$ 
23 :  $m' \leftarrow \text{CTR}_{ij}, \sigma_1$ 
24 : return  $m'$ 

```

NewSessionR(π_i^s, pid, m)

```

25 :  $\pi_i^s.\rho \leftarrow \text{Responder}$ 
26 :  $\pi_i^s.\alpha \leftarrow \text{negotiating}$ 
27 :  $\pi_i^s.\text{pid} \leftarrow \text{pid}$ 
28 : do  $\text{Send}(\pi_i^s, m)$ 

```

Corrupt(P_i, P_j)

```

29 : if  $(i, j) = (i^*, j^*) \vee (j^*, i^*)$ 
30 :   if  $\text{CTR}_{ij} \leq \text{CTR}^*$ 
31 :     return Abort
32 :   return  $k_{ij}$ 

```

RevealKey(π_i^s)

```

33 : if  $\pi_i^s.\alpha \neq \text{accept}$ 
34 :   return  $\perp$ 
35 : if  $(i, s) = (i^*, s^*)$ 
36 :   return Abort
37 :  $\pi_i^s.\kappa \leftarrow \text{exposed}$ 
38 : return  $\pi_i^s.sk$ 

```

Send(π_i^s, m) $/ \text{pid} = j$

```

39 : Parse  $m$  as  $\text{CTR}_{ji}, \sigma$ 
40 : if  $\text{Vrfy}(K_{ij}^{\text{MAC}}, j \parallel \text{CTR}_{ji}, \sigma) = 0$ 
41 :   return  $\perp$ 
42 : if  $\pi_i^s.\rho = \text{Responder}$ 
43 :    $z_1 \leftarrow \text{CTR}_{ji} - \text{CTR}_{ij}$ 
44 :   if  $z_1 < 0$ 
45 :     return  $\perp$ 
46 :    $k_{ij}^{\text{CTR}}, \text{CTR}_{ij} \leftarrow \text{Advnc}(k_{ij}^{\text{CTR}}, \text{CTR}_{ij}, z_1)$ 
47 :    $\sigma_2 \leftarrow \text{Mac}(K_{ij}^{\text{MAC}}, i \parallel \text{CTR}_{ij})$ 
48 :    $m' \leftarrow \text{CTR}_{ij}, \sigma_2$ 
49 :   return  $m'$ 
50 : else
51 :    $z_2 \leftarrow \text{CTR}_{ij} - \text{CTR}_{ji}$ 
52 :   if  $z_2 \neq 0$ 
53 :     return  $\perp$ 
54 :   if  $(i, j) = (i^*, j^*) \vee (j^*, i^*)$ 
55 :     if  $\text{CTR}_{ij} < \text{CTR}^*$ 
56 :        $\pi_i^s.sk \leftarrow sk^{\text{CTR}_{ij}}$ 
57 :     if  $\text{CTR}_{ij} = \text{CTR}^*$ 
58 :        $\pi_i^s.sk \leftarrow sk^*$ 
59 :      $s^* \leftarrow s$ 
60 :      $\text{CTR}_{ij} \leftarrow \text{CTR}_{ij} + 1$ 
61 :   else
62 :      $\pi_i^s.sk \leftarrow \text{KDF}(k_{ij}^{\text{CTR}}, \text{"der"})$ 
63 :      $k_{ij}^{\text{CTR}}, \text{CTR}_{ij} \leftarrow \text{Advnc}(k_{ij}^{\text{CTR}}, \text{CTR}_{ij}, 1)$ 
64 :      $\pi_i^s.\alpha \leftarrow \text{accept}$ 

```

Figure 9: Reduction $\mathcal{B}_{9,2}$ for the proof of Lemma 12. If at any time \mathcal{A} causes an oracle to Accept maliciously, then $\mathcal{B}_{9,2}$ simply does **Abort**. $\mathcal{B}_{9,2}$ provides \mathcal{A} with access to oracles $= \text{NewSessionI}(\cdot, \cdot, \cdot), \text{NewSessionR}(\cdot, \cdot), \text{Send}(\cdot, \cdot, \cdot), \text{RevealKey}(\cdot), \text{Corrupt}(\cdot, \cdot)$.

victory in the wSR game. To see this, note that for the (uninterrupted) target session, if $z_1 = \text{CTR}_{AB} - \text{CTR}_{BA} \geq 0$ then B will always catch up to the counter value of A (i.e. advance by z_1 steps) and both parties will compute a session key for counter value CTR_{AB} . Note also that in the target session, it is not possible for the if $z_2 \neq 0$ to be triggered after A receives the second protocol message since the counter value CTR_{BA} that B sends will always have caught up to CTR_{AB} in the processing of the first protocol message. Thus to conclude, we just need to show that, if a forgery has not occurred, it is not possible for the adversary to force $\text{CTR}_{AB} < \text{CTR}_{BA}$. Every time the adversary creates a new initiator session, the initiator's counter is incremented by either 1 or 2 steps, whereas B can advance an arbitrary number of times to catch up to (what B thinks is) A 's current counter state. Since the MAC includes party identification information and the initiator's counter value, in the absence of MAC forgeries the adversary cannot produce a valid protocol message with verifying MAC for any counter larger than the ones that it has seen as a result of genuine invocations of new protocol sessions.

Theorem 13 (wSR of LP2). *Let Π be the two-message protocol in Fig. 8, built using $\text{MAC} = \{\text{KGen}, \text{Mac}, \text{Vrfy}\}$ and PRF, with n parties. Then for any adversary \mathcal{A} against the wSR security of Π that makes a maximum of q queries that initiate new sessions for each party (with $q < \frac{|\text{CTR}|}{2}$), there exists an adversary \mathcal{B}_{13} against the SEUF-CMA- Q of MAC such that*

$$\text{Adv}_{\Pi}^{\text{wSR}}(\mathcal{A}) \leq n^2 \cdot \text{Adv}_{\text{MAC}}^{\text{SEUF-CMA-}Q}(\mathcal{B}_{13}).$$

PROOF. We proceed using a sequence of games.

Game 0. This is the original wSR game.

$$\Pr [G_{\Pi}^{\text{wSR}}(\mathcal{A}) = 1] = \Pr [G_0^A = 1]$$

Game 1. This game is the same as Game 0 except that we define an event E_{13} , that is said to occur if the adversary successfully forges a MAC while it is running, and the challenger aborts if E_{13} occurs.

$$\Pr [G_0^A = 1] = \Pr [G_1^A = 1] + \Pr [E_{13}]$$

We now bound the probability that E_{13} occurs. We construct a reduction \mathcal{B}_{13} to the SEUF-CMA- Q security of MAC with a verification oracle. First, the reduction guesses which parties will be involved in the forgery that triggers E_{13} , and then simulates the environment of Game 0. To do this, \mathcal{B}_{13} must select (initial) key derivation keys from the appropriate keyspace and randomly pick MAC keys for all pairs of parties except for the guessed parties i^* and j^* . It is then simple to simulate all queries to oracles of parties except communication between the guessed pair. Note that this choice of parties involved in the forgery is independent of the parties that the underlying adversary \mathcal{A} will eventually output for the target protocol run.

For any query to an oracle of party i^* with $\text{pid} = j^*$ or an oracle of j^* with $\text{pid} = i^*$, the reduction needs to forward queries to its own MAC and verification oracles. However note that \mathcal{B}_{13} knows the key derivation keys even for these parties, so responding to queries is straightforward except for its calls to \mathcal{O}_{Mac} and $\mathcal{O}_{\text{Vrfy}}$. The full reduction is detailed in Fig. 10.

$$\Pr[E_{13}] \leq n^2 \cdot \text{Adv}_{\text{MAC}}^{\text{SEUF-CMA-Q}}(\mathcal{B}_{13})$$

At this point, the adversary cannot win via a MAC forgery.

A win can be obtained if either party in the target session aborts, or if the parties compute different keys. If the target session begins with $\text{CTR}_{AB} \geq \text{CTR}_{BA}$, then the parties will always compute the same key via $sk_{AB} \leftarrow \text{KDF}(k_{AB}^{\text{CTR}}, \text{"der"})$ so we only need to argue that the adversary cannot force the state $\text{CTR}_{BA} > \text{CTR}_{AB}$ without forging the MAC. Note that every initiator session advances its counter (and thus key state) once either once or twice, while responder sessions (oracles) can advance an arbitrary number of times. However, the responder only advances if it has received and accepted a protocol message. For the responder to advance without the initiator having done so, the adversary must deliver a valid message to the responder without having called `NewSessionI`. This message must also have a counter value CTR'_{AB} that is greater than CTR_{AB} (the actual counter value of A with respect to B , and a valid MAC. Producing such a message that will be processed by B requires it to have a valid MAC on $A \parallel \text{CTR}'_{AB}$. Since we assume that the adversary does not produce a MAC forgery it must have seen this message before, which means it must have been output by a `NewSessionI` query, and we have a contradiction. This concludes the proof. \square

4.4 LP1: A One-Message Protocol with Fixed Roles

In Fig. 11 we present a one-message protocol, LP1, with linear key evolution. Like in LP2, the roles of initiator and responder are fixed, so the same party initiates every session: i.e. $\text{CTR}_{AB} \geq \text{CTR}_{BA}$ (for A initiating). Ensuring that the counter states of the communicating parties is slightly simpler in LP1 than LP2, since we do not have to worry about the responder advancing before the initiator. The initiator simply advances once every time it participates in a session, and both parties advance exactly once after computing a session key. If protocol messages are dropped then it may be necessary for the responder to advance before it can compute the session key.

In Theorem 14 we show that LP1 achieves one-sided authentication (responder authenticates initiator). Achieving weak synchronization robustness (wSR, Theorem 17) is similar in LP1 and LP2, and is guaranteed by MAC security. Like with LP2, if both parties need to be able to initiate then LP1 can be run in *duplex mode*.

<p>Reduction \mathcal{B}_{13} playing $\mathcal{G}_{\text{MAC}}^{\text{SEUF-CMA-Q}}(\mathcal{B}_{13})$</p> <hr/> <pre> 1 : $i^*, j^* \xleftarrow{\\$} [n]$ 2 : for $i, j \in [n]$ do 3 : $k_{ij}^{\text{CTR}} = k_{ji}^{\text{CTR}} \xleftarrow{\\$} \mathcal{K}_{\text{PRF}}$ 4 : for $n \times [n] \setminus (i^*, j^*)$ do 5 : $k_{ij}^{\text{MAC}} = k_{\text{MAC}}^{j,i} \xleftarrow{\\$} \mathcal{K}_{\text{MAC}}$ 6 : $\text{CTR}_{ij} = \text{CTR}_{ji} \leftarrow 0$ 7 : $\mathcal{A}^{\text{oracles}}$ </pre> <p>NewSessionI(π_i^s, pid)</p> <hr/> <pre> 8 : $\pi_i^s.\rho \leftarrow \text{Initiator}$ 9 : $\pi_i^s.\alpha \leftarrow \text{negotiating}$ 10 : $\pi_i^s.\text{pid} \leftarrow \text{pid} \quad / = j$ 11 : $z_0 \leftarrow \text{NextOdd}(\text{CTR}_{ij}) - \text{CTR}_{ij}$ 12 : $k_{ij}^{\text{CTR}}, \text{CTR}_{ij} \leftarrow \text{Advnc}(k_{ij}^{\text{CTR}}, \text{CTR}_{ij}, z_0)$ 13 : if $(i, j) = (i^*, j^*) \vee (j^*, i^*)$ 14 : $\sigma_1 \leftarrow \text{call } \mathcal{O}_{\text{Mac}}(i \parallel \text{CTR}_{ij})$ 15 : else 16 : $\sigma_1 \leftarrow \text{Mac}(k_{ij}^{\text{MAC}}, i \parallel \text{CTR}_{ij})$ 17 : $m' \leftarrow \text{CTR}_{ij}, \sigma_1$ 18 : return m' </pre> <p>NewSessionR(π_i^s, pid, m)</p> <hr/> <pre> 19 : $\pi_i^s.\rho \leftarrow \text{Responder}$ 20 : $\pi_i^s.\alpha \leftarrow \text{negotiating}$ 21 : $\pi_i^s.\text{pid} \leftarrow \text{pid}$ 22 : do $\text{Send}(\pi_i^s, m)$ </pre>	<pre> Send(π_i^s, m) $/ \text{pid} = j$ <hr/> 23 : Parse m as CTR_{ji}, σ 24 : if $(i, j) = (i^*, j^*) \vee (j^*, i^*)$ 25 : $b \leftarrow \text{call } \mathcal{O}_{\text{Vrfy}}(j \parallel \text{CTR}_{ji}, \sigma)$ 26 : if $b = 0$ 27 : return \perp 28 : else 29 : if $\text{Vrfy}(k_{ij}^{\text{MAC}}, j \parallel \text{CTR}_{ij}, \sigma) = 0$ 30 : return \perp 31 : if $\pi_i^s.\rho = \text{Responder}$ 32 : $z_1 \leftarrow \text{CTR}_{ji} - \text{CTR}_{ij}$ 33 : if $z_1 < 0$ 34 : return \perp 35 : $k_{ij}^{\text{CTR}}, \text{CTR}_{ij} \leftarrow \text{Advnc}(k_{ij}^{\text{CTR}}, \text{CTR}_{ij}, z_1)$ 36 : if $(i, j) = (i^*, j^*) \vee (j^*, i^*)$ 37 : $\sigma_2 \leftarrow \text{call } \mathcal{O}_{\text{Mac}}(i \parallel \text{CTR}_{ij})$ 38 : else 39 : $\sigma_2 \leftarrow \text{Mac}(k_{ij}^{\text{MAC}}, i \parallel \text{CTR}_{ij})$ 40 : return $m' \leftarrow \text{CTR}_{ij}, \sigma_2$ 41 : else 42 : $z_2 \leftarrow \text{CTR}_{ij} - \text{CTR}_{ji}$ 43 : if $z_2 \neq 0$ 44 : return \perp 45 : $\pi_i^s.sk \leftarrow \text{KDF}(k_{ij}^{\text{CTR}}, \text{"der"})$ 46 : $\pi_i^s.\alpha \leftarrow \text{accept}$ 47 : $k_{ij}^{\text{CTR}}, \text{CTR}_{ij} \leftarrow \text{Advnc}(k_{ij}^{\text{CTR}}, \text{CTR}_{ij}, 1)$ </pre>
--	---

Figure 10: Reduction \mathcal{B}_{13} for the proof of Theorem 13. \mathcal{B}_{13} provides \mathcal{A} with access to $\text{oracles} = \text{NewSessionI}(\cdot, \cdot, \cdot), \text{NewSessionR}(\cdot, \cdot), \text{Send}(\cdot, \cdot, \cdot)$.

4.4.1 AKE-R of LP1

Theorem 14 (AKE-R of LP1). *Let Π be the one-message protocol in Fig. 11, built using $\text{MAC} = \{\text{KGen}, \text{Mac}, \text{Vrfy}\}$ and PRF, with n parties. Then for any adversary \mathcal{A} against the AKE-R security of Π that makes a maximum of q queries that initiate new*

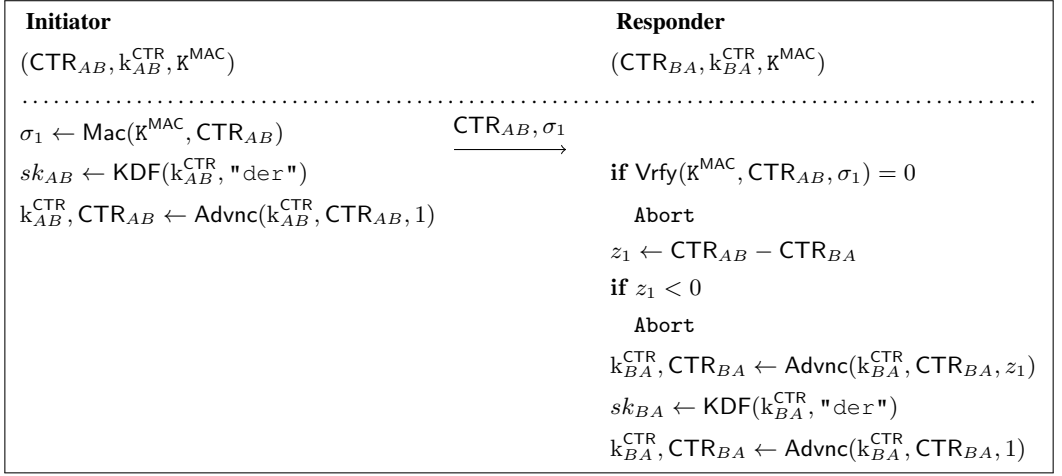


Figure 11: LP1, a one-message protocol with fixed roles.

sessions for each party (with $q < |CTR|$), there exists an adversary $\mathcal{B}_{14.1}$ against the SEUF-CMA- Q security of MAC and an adversary $\mathcal{B}_{14.2}$ against the KEvol security of PRF such that

$$\text{Adv}_{\text{II}}^{\text{AKE-R}}(\mathcal{A}) \leq n^2 \cdot \left(2 \cdot \text{Adv}_{\text{MAC}}^{\text{SEUF-CMA-}Q}(\mathcal{B}_{14.1}) + q \cdot \text{Adv}_{\text{PRF}}^{\text{KEvol}}(\mathcal{B}_{14.2}) \right).$$

We form a bound for each of the two ways in which an adversary can break AKE-R security, namely Ent-Auth-R and Key-Ind, and then sum these bounds.

We now give intuition regarding the Ent-Auth-R proof. Note that since $q < \frac{|CTR|}{2}$, all first protocol messages are unique, and thus the first message in every transcript is unique. This rules out the possibility of *multiple* oracles with non-unique matching conversations accepting. To conclude our proofs, we need to show that the only way an adversary can force an oracle to accept and for there not to exist any other oracle with a matching transcript, the adversary must forge a MAC message.

A responder oracle accepts when it receives a first protocol message, and so to win there must not exist any oracle with the same partial transcript as this accepting oracle. Since the MAC is calculated on the counter value and the communicating parties' identities, the input message to the accepting oracle must have been generated by an oracle of the communication partner (in which case there is a matching conversation and the adversary has not won) or the adversary has produced a MAC forgery (in the SEUF-CMA- Q sense).

Lemma 15 (Ent-Auth-R of LP1). *For any adversary \mathcal{A} , the probability that there exists*

an oracle with $\rho = \text{Responder}$ that accepts maliciously can be bounded by

$$\text{Adv}_{\Pi}^{\text{Ent-Auth-R}}(\mathcal{A}) \leq n^2 \cdot \text{Adv}_{\text{MAC}}^{\text{SEUF-CMA-Q}}(\mathcal{B}_{14.1}).$$

PROOF. We proceed using a sequence of games.

Game 0. This is the original Ent-Auth game.

$$\Pr [G_{\Pi}^{\text{Ent-Auth}}(\mathcal{A}) = 1] = \Pr [G_0^{\mathcal{A}} = 1]$$

Game 1. In this game we guess which responder will be the first to accept maliciously and its partner identity, and abort if this guess is wrong. The game is the same as Game 0 except that the challenger guesses $(i^*, j^*) \leftarrow_{\$} [n] \times [n]$, and if an oracle π_i^s (for some s) accepts maliciously with $\pi_i^s.\rho \neq \text{Responder}$ or $i^* \neq i$ or $j^* \neq \pi_i^s.\text{pid}$, then the challenger aborts.

$$\Pr [G_0^{\mathcal{A}} = 1] = n^2 \cdot \Pr [G_1^{\mathcal{A}} = 1]$$

We construct a reduction $\mathcal{B}_{14.1}$ that is playing against the SEUF-CMA- Q security of MAC that simulates the environment for an underlying adversary \mathcal{A} that attempts to win in game Game 1. The reduction generates (initial) key derivation keys for all pairs of parties, and authentication keys for all pairs of parties except i^* and j^* . When responding to queries by \mathcal{A} regarding all other parties, the reduction will honestly provide messages as specified in the protocol specification and the Ent-Auth game. For any query made between oracles of parties i^* and j^* , the reduction will use its \mathcal{O}_{Mac} oracle and provide the received value in its simulation for \mathcal{A} . For example, to initialize initiator oracles π_{j^*} , the reduction checks the current state counter $\text{CTR}_{j^*i^*}$ and calls $\mathcal{O}_{\text{Mac}}(\text{CTR}_{j^*i^*})$. If \mathcal{A} provides any value that it has not been given as an initialization query as input to a `NewSessionR` query from i^* to j^* , then the reduction sends this to its $\mathcal{O}_{\text{Vrfy}}$ oracle in the SEUF-CMA- Q game. The simulation of Game 1 is perfect and any win for \mathcal{A} directly corresponds to a valid signature forgery, so we can write

$$\Pr [G_1^{\mathcal{A}} = 1] \leq \text{Adv}_{\text{MAC}}^{\text{SEUF-CMA-Q}}(\mathcal{B}_{14.1}).$$

Summing these terms gives the bound in the statement of Lemma 15. □

The proof of key indistinguishability is very similar to that of Lemma 12 and the game hops proceed following the same strategy. Again we use a reduction to the KEvol security of PRF.

Lemma 16 (Key-Ind of LP1). *For any adversary \mathcal{A} and (any fixed) entity authentication adversary $\mathcal{B}_{14.1}$, the probability that \mathcal{A} answers the Test challenge correctly can be bounded by*

$$\text{Adv}_{\Pi}^{\text{Key-Ind}}(\mathcal{A}) \leq \text{Adv}_{\Pi}^{\text{Ent-Auth}}(\mathcal{B}_{14.1}) + n^2 \cdot q \cdot \text{Adv}_{\text{PRF}}^{\text{KEvol}}(\mathcal{B}_{14.2})$$

where all quantities are defined as stated in Theorem 14.

PROOF. Let b' be the bit output by \mathcal{A} in each game, and b be the bit sampled as part of the Test query.

Game 0. This is the original Key-Ind game.

$$\Pr [G_{\Pi}^{\text{Key-Ind}}(\mathcal{A}) = 1] = \Pr [G_0^{\mathcal{A}} = 1]$$

Game 1. This game is the same as Game 0, except the challenger aborts and chooses $b' \stackrel{\$}{\leftarrow} \{0, 1\}$ if any oracle accepts maliciously.

$$\Pr [G_0^{\mathcal{A}} = 1] \leq \Pr [G_1^{\mathcal{A}} = 1] + \text{Adv}_{\Pi}^{\text{Ent-Auth-R}}(\mathcal{B}_{14.1})$$

At this stage, the oracle to which \mathcal{A} asks its Test query has a unique partner oracle with a matching conversation.

Game 2. This game is the same as Game 1, except the challenger guesses the two parties involved in the Test query via $(i^*, j^*) \stackrel{\$}{\leftarrow} [n] \times [n]$, and additionally guesses the counter value which identifies the key derivation key state of the session key in the Test query. If \mathcal{A} issues a $\text{Test}(\pi_{i^*}^s)$ query with $i^* \neq i$ or $j^* \neq \pi_{i^*}^s.\text{pid}$ (for some s), or the session key computed via $\text{Test}(\pi_{i^*}^s)$, i.e. $sk_{i^*j^*}$ is not equal to $\text{KDF}(k_{i^*j^*}^{\text{CTR}^*}, \text{"der"})$, then the challenger aborts. Note that the challenger does not guess which oracle of i^* the Test query will be made to, only the counter value linked to the session key in that query.

$$\Pr [G_1^{\mathcal{A}} = 1] \leq n^2 \cdot q \cdot \Pr [G_2^{\mathcal{A}} = 1]$$

At this stage, if the challenger has guessed correctly then the Test query will be asked to an oracle after the key derivation counter has been advanced a fixed number of times, and this oracle has a unique partner with a matching conversation.

Game 3. This game is the same as Game 2, except that when the challenger runs KDF on the key derivation values used in the Test query, the challenger instead responds with a random key from the session key space. Noticing this change results in an adversary that is successful in the KEvol game for PRF, so we can write

$$\Pr [G_2^{\mathcal{A}} = 1] = \Pr [G_3^{\mathcal{A}} = 1] + \text{Adv}_{\text{PRF}}^{\text{KEvol}}(\mathcal{B}_{9.2}).$$

The reduction $\mathcal{B}_{14.2}$ is detailed in Fig. 12. $\mathcal{B}_{14.2}$ initially guesses the target parties in the Test session and the counter value associated with the Test session, as per the previous game hop.

Reduction $\mathcal{B}_{14.2}$ playing $G_{\text{KDF}}^{\text{KEvol}}(\mathcal{B}_{14.2})$

```

1 :  $i^*, j^* \xleftarrow{\$} [n]$ ;  $\text{CTR}^* \xleftarrow{\$} [q]$ 
2 : for  $i, j \in [n]$  do
3 :    $\text{CTR}_{ij} = \text{CTR}_{ji} \leftarrow 0$ 
4 :    $K_{ij}^{\text{MAC}} = K_{ji}^{\text{MAC}} \xleftarrow{\$} \mathcal{K}_{\text{MAC}}$ 
5 :   for  $n \times [n] \setminus (i^*, j^*)$  do
6 :      $k_{ij}^0 = k_{ji}^0 \xleftarrow{\$} \mathcal{K}_{\text{PRF}}$ 
7 :   output  $\text{CTR}^*$ 
8 :   receive( $sk^0, \dots, sk^{\text{CTR}^*-1}, sk^*, k^{\text{CTR}^*+1}$ )
9 :    $k_{i^*j^*}^{\text{CTR}^*+1} \leftarrow k^{\text{CTR}^*+1}$ 
10 :  $\mathcal{A}^{\text{oracles}}$ 
11 : When  $\mathcal{A}$  calls  $\text{Test}(\pi_i^s)$  do
12 :   if  $i \neq i^*$  or  $j^* \neq \pi_{i^*}^{s^*}.\text{pid}$ 
13 :     return Abort
14 :   return  $\pi_i^s.sk$ 
15 :    $(i^*, s^*, b') \xleftarrow{\$} \mathcal{A}(\pi_{i^*}^{s^*}.sk)$ 
16 :   return  $b'$ 

```

NewSessionI(π_i^s, pid)

```

17 :  $\pi_i^s.\rho \leftarrow \text{Initiator}$ 
18 :  $\pi_i^s.\alpha \leftarrow \text{negotiating}$ 
19 :  $\pi_i^s.\text{pid} \leftarrow \text{pid} \neq j$ 
20 :  $\sigma_1 \leftarrow \text{Mac}(K^{\text{MAC}}, \text{CTR}_{ij})$ 
21 :  $m' \leftarrow \text{CTR}_{ij}, \sigma_1$ 
22 :  $k_{ij}^{\text{CTR}}, \text{CTR}_{ij} \leftarrow \text{Advnc}(k_{ij}^{\text{CTR}}, \text{CTR}_{ij}, 1)$ 
23 : return  $m'$ 

```

NewSessionR(π_i^s, pid, m)

```

24 :  $\pi_i^s.\rho \leftarrow \text{Responder}$ 
25 :  $\pi_i^s.\alpha \leftarrow \text{negotiating}$ 
26 :  $\pi_i^s.\text{pid} \leftarrow \text{pid}$ 
27 : do  $\text{Send}(\pi_i^s, m)$ 

```

Corrupt(P_i, P_j)

```

28 : if  $(i, j) = (i^*, j^*) \vee (j^*, i^*)$ 
29 :   if  $\text{CTR}_{ij} \leq \text{CTR}^*$ 
30 :     return Abort
31 :   return  $k_{ij}$ 

```

RevealKey(π_i^s)

```

32 : if  $\pi_i^s.\alpha \neq \text{accept}$ 
33 :   return  $\perp$ 
34 : if  $(i, s) = (i^*, s^*)$ 
35 :   return Abort
36 :  $\pi_i^s.\kappa \leftarrow \text{exposed}$ 
37 : return  $\pi_i^s.sk$ 

```

Send(π_i^s, m) / pid=j

```

38 : Parse  $m$  as  $\text{CTR}_{ji}, \sigma_1$ 
39 : if  $\text{Vrfy}(K_{ij}^{\text{MAC}} \parallel \text{CTR}_{ji}, \sigma_1) = 0$ 
40 :   return  $\perp$ 
41 :  $z_1 \leftarrow \text{CTR}_{ji} - \text{CTR}_{ij}$ 
42 : if  $z_1 < 0$ 
43 :   return  $\perp$ 
44 :  $k_{ij}^{\text{CTR}}, \text{CTR}_{ij} \leftarrow \text{Advnc}(k_{ij}^{\text{CTR}}, \text{CTR}_{ij}, z_1)$ 
45 : if  $(i, j) = (i^*, j^*) \vee (j^*, i^*)$ 
46 :   if  $\text{CTR}_{ij} < \text{CTR}^*$ 
47 :      $\pi_i^s.sk \leftarrow sk^{\text{CTR}_{ij}}$ 
48 :   if  $\text{CTR}_{ij} = \text{CTR}^*$ 
49 :      $\pi_i^s.sk \leftarrow sk^*$ 
50 :      $s^* \leftarrow s$ 
51 :    $\text{CTR}_{ij} \leftarrow \text{CTR}_{ij} + 1$ 
52 : else
53 :    $\pi_i^s.sk \leftarrow \text{KDF}(k_{ij}^{\text{CTR}}, \text{"der"})$ 
54 :    $k_{ij}^{\text{CTR}}, \text{CTR}_{ij} \leftarrow \text{Advnc}(k_{ij}^{\text{CTR}}, \text{CTR}_{ij}, 1)$ 
55 :    $\pi_i^s.\alpha \leftarrow \text{accept}$ 

```

Figure 12: Reduction $\mathcal{B}_{14.2}$ for the proof of Lemma 16. If at any time \mathcal{A} causes an oracle to accept maliciously, then $\mathcal{B}_{14.2}$ simply does Abort. $\mathcal{B}_{14.2}$ provides \mathcal{A} with access to oracles = $\text{NewSessionI}(\cdot, \cdot, \cdot), \text{NewSessionR}(\cdot, \cdot), \text{Send}(\cdot, \cdot, \cdot), \text{RevealKey}(\cdot), \text{Corrupt}(\cdot, \cdot)$.

In the event that $\mathcal{B}_{14.2}$ is in the ‘real’ version of its own game, where it receives a genuine evaluation of the function KDF, $\mathcal{B}_{14.2}$ perfectly simulates Game 2 for \mathcal{A} , and otherwise it perfectly simulates Game 3.

At this stage, the Test query is asked on a key that is randomly chosen, and thus independent of the protocol and the security game. Consequently,

$$\Pr [G_3^{\mathcal{A}} = 1] = \frac{1}{2} \Rightarrow \text{Adv}_{\Pi}^3(\mathcal{A}) = 0.$$

□

4.4.2 wSR of LP1

The wSR security of LP1 is achieved in a similar manner to LP2 and we proceed to prove the theorem.

Theorem 17 (wSR of LP1). *Let Π be the one-message protocol in Fig. 11, built using $\text{MAC} = \{\text{KGen}, \text{Mac}, \text{Vrfy}\}$ and PRF, with n parties. Then for any adversary \mathcal{A} against the wSR security of Π that makes a maximum of q queries that initiate new sessions for each party (with $q < |\text{CTR}|$), there exists an adversary \mathcal{B}_{17} against the SEUF-CMA- Q security of MAC such that*

$$\text{Adv}_{\Pi}^{\text{wSR}}(\mathcal{A}) \leq n^2 \cdot \text{Adv}_{\text{MAC}}^{\text{SEUF-CMA-}Q}(\mathcal{B}_{17}).$$

We now argue that LP1 obtains weak synchronization robustness (wSR). A proof of wSR must argue that whatever the adversary does before the target protocol run occurs, neither of the parties will abort during the target protocol run itself and both will arrive at the same session key. Protocol LP1, like LP2 (Fig. 8), only has correctness when the initiator’s counter is at least the size of the responder’s counter, i.e. $\text{CTR}_{AB} \geq \text{CTR}_{BA}$ — this inequality is guaranteed in our protocol by MAC security. By inspection, if an adversary forges a MAC on CTR_{AB} for some CTR_{AB} larger than the current value of CTR_{BA} and delivers this MAC as part of a protocol message to an oracle of B , then any subsequent protocol run will cause B to Abort and thus will be a winning target protocol run for this adversary.

The formal proof is below, but here we outline the proof idea. We first define an event E_{17} that is triggered if the adversary in the wSR game forges a MAC, i.e. produces a message-tag pair that verifies correctly and that it has not seen before, and the challenger aborts if this occurs: bounding this event is of course straightforward. Then, we must argue that if a MAC forgery has not occurred then there are in fact no viable routes to victory in the wSR game. To see this, note that for the (uninterrupted) target session, if $z_1 = \text{CTR}_{AB} - \text{CTR}_{BA} \geq 0$ then B will always catch up to the counter value of A (i.e. advance by z_1 steps) and both parties will compute a session key for counter value CTR_{AB} . Thus to conclude, we just need to show that, if a forgery has not occurred, it

is not possible for the adversary to force $\text{CTR}_{AB} < \text{CTR}_{BA}$. Every time the adversary creates a new initiator session, the initiator's counter is incremented by 1, whereas B can advance an arbitrary number of times to catch up to (what B thinks is) A 's current counter state. Since the MAC includes party identification information and the initiator's counter value, in the absence of MAC forgeries the adversary cannot produce a valid protocol message with verifying MAC for any counter larger than the ones that it has seen as a result of genuine invocations of new protocol sessions.

PROOF. We proceed using a sequence of games.

Game 0. This is the original wSR game.

$$\Pr [G_{\Pi}^{\text{wSR}}(\mathcal{A}) = 1] = \Pr [G_0^{\mathcal{A}} = 1]$$

Game 1. This game is the same as Game 0 except that we define an event E_{17} , that is said to occur if the adversary successfully forges a MAC while it is running, and the challenger aborts if E_{17} occurs.

$$\Pr [G_0^{\mathcal{A}} = 1] = \Pr [G_1^{\mathcal{A}} = 1] + \Pr [E_{17}]$$

We now bound the probability that E_{17} occurs. We construct a reduction \mathcal{B}_{17} to the SEUF-CMA- Q security of MAC with a verification oracle. First, the reduction guesses which parties will be involved in the forgery that triggers E_{17} , and then simulates the environment of Game 0. To do this, \mathcal{B}_{17} must select (initial) key derivation keys from the appropriate key space and randomly pick MAC keys for all pairs of parties except for the guessed parties i^* and j^* . It is then easy to simulate all queries to the parties' oracles, except communication between the guessed pair. Note that this choice of parties involved in the forgery is independent of the parties that the underlying adversary \mathcal{A} will eventually output for the target protocol run.

For any query to an oracle of party i^* with $\text{pid} = j^*$ or an oracle of j^* with $\text{pid} = i^*$, the reduction needs to forward queries to its own MAC and verification oracles. However note that \mathcal{B}_{17} knows the key derivation keys even for these parties, so responding to queries is straightforward except for its calls to \mathcal{O}_{Mac} and $\mathcal{O}_{\text{Vrfy}}$. The full reduction is detailed in Fig. 13.

$$\Pr [E_{17}] \leq n^2 \cdot \text{Adv}_{\text{MAC}}^{\text{SEUF-CMA-}Q}(\mathcal{B}_{17})$$

At this point, the adversary cannot win via MAC forgery.

A win can be obtained if either party in the target session aborts, or if the parties compute different keys. If the target session begins with $\text{CTR}_{AB} \geq \text{CTR}_{BA}$, then the parties will always compute the same key via $sk_{AB} \leftarrow \text{KDF}(k_{AB}^{\text{CTR}}, \text{"der"})$, so we only need to argue that the adversary cannot force the state $\text{CTR}_{BA} > \text{CTR}_{AB}$

without forging the MAC. Note that every initiator session advances its counter (and thus key state) exactly once, while responder sessions (oracles) can advance an arbitrary number of times. However, the responder only advances if it has received and accepted a protocol message. For the responder to advance without the initiator having done so, the adversary must deliver a valid message to the responder without having called `NewSessionI`. This message must also have a counter value CTR'_{AB} that is greater than CTR_{AB} (the actual counter value of A with respect to B , and a valid MAC. Producing such a message that will be processed by B requires it to have a valid MAC on CTR'_{AB} . Since we assume that the adversary does not produce a MAC forgery it must have seen this message before, which means it must have been output by a `NewSessionI` query, and we have a contradiction. This concludes the proof. \square

5 Non-Linear Key Evolution

In the previous section, we have considered protocols that deploy a linear key evolving mechanism. We have seen that the linearity of these mechanisms has significant downsides when the protocol runs multiple times in parallel between the same two parties. Especially interleaving of messages might cause all but one protocol execution to abort, which is an undesirable behavior.

In this section, we present a protocol that uses puncturable pseudorandom functions (PPRFs) as a “non-linear” key evolution mechanism. We show that this protocol can establish many parallel sessions between two parties, while only requiring some additional storage (logarithmic in the supported maximum number sessions) and computations (in practice hash function evaluations logarithmic in the supported maximum number of sessions).

5.1 Puncturable Pseudorandom Functions

We briefly recall the basic definition of puncturable pseudorandom functions (PPRF). A PPRF is a special case of a pseudorandom function, where it is possible to compute punctured keys, which do not allow evaluation on inputs that have been punctured. We recall the definition of a PPRF and its security [SW14].

Definition 12 (PPRF). A *puncturable pseudorandom function* with key space $\mathcal{K}_{\text{PPRF}}$, domain $\mathcal{D}_{\text{PPRF}}$, and range $\mathcal{R}_{\text{PPRF}}$ consists of three probabilistic polynomial-time algorithms $\text{PPRF} = (\text{Setup}, \text{Eval}, \text{Punct})$, which are described as follows:

- $\text{Setup}(1^\lambda)$: This algorithm takes as input the security parameter λ and outputs a description of a key $k \in \mathcal{K}_{\text{PPRF}}$.

- $\text{Eval}(k, x)$: This algorithm takes as input a key $k \in \mathcal{K}_{\text{PPRF}}$ and a value $x \in \mathcal{D}_{\text{PPRF}}$, and outputs a value $y \in \mathcal{R}_{\text{PPRF}}$, or a failure symbol \perp .
- $\text{Punct}(k, x)$: This algorithm takes as input a key $k \in \mathcal{K}_{\text{PPRF}}$ and a value $x \in \mathcal{D}_{\text{PPRF}}$, and returns a punctured key $k' \in \mathcal{K}_{\text{PPRF}}$.

Note that the puncturing procedure can also output an unmodified key (i.e. $k' = k$). This is for example reasonable if the procedure is called on an already-punctured value.

Definition 13 (PPRF Correctness). A PPRF is *correct* if for every subset $\{x_1, \dots, x_t\} = \mathcal{S} \subseteq \mathcal{D}_{\text{PPRF}}$ and all $x \in \mathcal{D}_{\text{PPRF}} \setminus \mathcal{S}$, it holds that

$$\Pr \left[\text{Eval}(k_0, x) = \text{Eval}(k_t, x) : \begin{array}{l} k_0 \stackrel{\$}{\leftarrow} \text{Setup}(1^\lambda); \\ k_i = \text{Punct}(k_{i-1}, x_i) \text{ for } i \in [t]; \end{array} \right] = 1.$$

The security experiment asks that an adversary cannot distinguish an evaluation of a real input (provided by the adversary) from a random output range element, even if the adversary has access to an evaluation oracle and the key that results from puncturing on the challenge input.

Definition 14 (PPRF Security). The advantage of an adversary \mathcal{A} in the rand security experiment $\text{G}_{\text{PPRF}}^{\text{rand}}(\mathcal{A})$ defined in Fig. 14 is

$$\text{Adv}_{\text{PPRF}}^{\text{rand}}(\mathcal{A}) := \left| \Pr [\text{G}_{\text{PPRF}}^{\text{rand}}(\mathcal{A}) = 1] - \frac{1}{2} \right|.$$

5.2 PPRF-based Symmetric AKE

Intuition. The main idea of our PPRF-based protocol is to derive the session key via an evaluation of the PPRF. That is, both parties share a PPRF evaluation key k , which is used to derive session keys by computing $\text{Eval}(k, N_A)$ for some value N_A (in our protocols this will be a counter). After derivation of a session key, the PPRF key will also be punctured at the value N_A by computing $k \leftarrow \text{Punct}(k, N_A)$. Note that the new key k cannot recompute $\text{Eval}(k, N_A)$ as it has been punctured for N_A . This will be our leverage to achieve forward security.

Additionally, the PPRF is an essential building block to achieve full synchronization robustness in our protocols. Intuitively, the puncturing procedure of a PPRF does not evolve its key “linearly” but rather enables fine-grained removal of evaluation capabilities. This guarantees that every protocol run with some fresh value N_A for $\text{Eval}(k, N_A)$ will be completed successfully, even if other protocol runs with some value $N'_A \neq N_A$ are executed in-between.

Our protocols. We present a one-message and a two-message protocol, based on PPRFs. Both protocols have fixed roles, meaning the same party will always initiate (and only this party is required to store the counter). The two-message protocol implicitly authenticates both parties (and thus achieves mutual authentication), while the one-message protocol inherently only achieves responder-only authentication (responder authenticates initiator).

Another important aspect of our protocols is that they use counters to systematically “exhaust” the PPRF. We will later discuss that this approach assists the efficiency of tree-based PPRFs as discussed in Aviram et al. [AGJ19]. The number of session keys that can be derived is equal to the size of the counter space.

5.3 PP2: a Two-Message Protocol with Fixed Roles

5.3.1 AKE-M of PP2

Theorem 18 (AKE-M of PP2). *Let Π be the two-message protocol in Fig. 15, built using $\text{MAC} = \{\text{KGen}, \text{Mac}, \text{Vrfy}\}$ and $\text{PPRF} = (\text{Setup}, \text{Eval}, \text{Punct})$ with n parties. Then for any adversary \mathcal{A} against the AKE-M security of Π that makes a maximum of q queries that initiate new sessions for each party (with $q < |\text{CTR}|$), there exists an adversary $\mathcal{B}_{18.1}$ against the SEUF-CMA- Q of MAC and an adversary $\mathcal{B}_{18.2}$ against the rand security of PPRF such that*

$$\text{Adv}_{\Pi}^{\text{AKE-M}}(\mathcal{A}) \leq 4n^2 \cdot \text{Adv}_{\text{MAC}}^{\text{SEUF-CMA-}Q}(\mathcal{B}_{18.1}) + n^2 \cdot q \cdot \text{Adv}_{\text{PPRF}}^{\text{rand}}(\mathcal{B}_{18.2}).$$

We form a bound for each of the three ways in which an adversary can break AKE security, namely Ent-Auth-R, Ent-Auth-I and Key-Ind, and then sum these bounds.

There are two MAC security terms, for entity authentication of responder and initiator, and so $2 \cdot \text{Adv}_{\text{MAC}}^{\text{SEUF-CMA-}Q}(\mathcal{B}_{18.1})$ bounds these two terms by fixing $\mathcal{B}_{18.1}$ to be whichever of $\mathcal{B}_{18.1r}$ and $\mathcal{B}_{18.1i}$ has greater advantage.

Lemma 19 (Ent-Auth-R of PP2). *For any adversary \mathcal{A} , the probability that there exists an oracle with $\rho = \text{Responder}$ that accepts maliciously can be bounded by*

$$\text{Adv}_{\Pi}^{\text{Ent-Auth-R}}(\mathcal{A}) \leq n^2 \cdot \text{Adv}_{\text{MAC}}^{\text{SEUF-CMA-}Q}(\mathcal{B}_{18.1r})$$

where all quantities are defined as stated in Theorem 18.

PROOF. We proceed using a sequence of games.

Game 0. This is the original Ent-Auth game. We have

$$\Pr [\text{G}_{\Pi}^{\text{Ent-Auth}}(\mathcal{A}) = 1] = \Pr [G_0^{\mathcal{A}} = 1].$$

Note that that Theorem 18 requires that the adversary only initiates $q < |\text{CTR}|$ sessions (which can be guaranteed by choosing $|\text{CTR}|$ exponential in the security parameter),

implying that the counter is never exhausted and thus, the initial protocol message sent from initiator to responder is unique. This rules out the possibility of *multiple* oracles with non-unique matching conversations accepting.

Game 1. In this game we guess which responder will be the first to accept maliciously and its partner identity, and abort if this guess is wrong. The game is the same as Game 0 except that the challenger guesses $(i^*, j^*) \leftarrow_{\$} [n] \times [n]$, and if an oracle π_i^s (for some s) accepts maliciously with $\pi_i^s.\rho \neq \text{Responder}$ or $i^* \neq i$ or $j^* \neq \pi_i^s.\text{pid}$, then the challenger aborts. We have

$$\Pr [G_0^A = 1] = n^2 \cdot \Pr [G_1^A = 1].$$

We conclude our reduction by constructing a reduction $\mathcal{B}_{18.1r}$ that is playing against the SEUF-CMA- Q security of MAC that simulates the environment for an underlying adversary \mathcal{A} that attempts to win in game Game 1. Note that in Game 1, the only way that an adversary can win without an abort occurring is if it makes a responder oracle (of party i^*) accept maliciously, and no initiator oracle (of party j^*) has a matching conversation. To do this, it must initialize an initiator oracle, resulting in some initial protocol message, and then provide a different message to the responder oracle that causes the responder oracle to accept. (If the messages were not different and the adversary forwarded the genuine initial message, then the conversations would match.)

The reduction generates (initial) key derivation keys for all pairs of parties, and authentication keys for all pairs of parties except i^* and j^* . When responding to queries by \mathcal{A} regarding all other parties, the reduction will honestly provide messages as specified in the protocol specification and the Ent-Auth game. For any query made between oracles of parties i^* and j^* , the reduction will use its \mathcal{O}_{Mac} oracle and provide the received value in its simulation for \mathcal{A} . For example, to initialize initiator oracles π_{j^*} , the reduction sets $N := \text{CTR}_{j^*i^*}$, increments $\text{CTR}_{j^*i^*} := \text{CTR}_{j^*i^*} + 1$, and calls $\mathcal{O}_{\text{Mac}}(j^* \parallel N)$. If \mathcal{A} provides a value that it has not been given as an initialization query as input to a NewSessionR query from i^* to j^* , then the reduction sends this to its $\mathcal{O}_{\text{Vrfy}}$ oracle in the SEUF-CMA- Q game. The simulation of Game 1 is perfect and any win for \mathcal{A} directly corresponds to a valid signature forgery, so we can write

$$\Pr [G_1^A = 1] \leq \text{Adv}_{\text{MAC}}^{\text{SEUF-CMA-}Q}(\mathcal{B}_{18.1r}).$$

Summing these terms gives the bound in the statement of Lemma 19. \square

The second proof is very similar, and considers malicious acceptance by an initiator, i.e. as a result of a full protocol run of two messages. We only detail significant changes and note that our term collection is exactly the same.

Lemma 20 (Ent-Auth-I of PP2). *For any adversary \mathcal{A} , the probability that there exists an oracle with $\rho = \text{Initiator}$ that accepts maliciously can be bounded by*

$$\text{Adv}_{\Pi}^{\text{Ent-Auth-I}}(\mathcal{A}) \leq n^2 \cdot \text{Adv}_{\text{MAC}}^{\text{SEUF-CMA-Q}}(\mathcal{B}_{18.1i})$$

where all quantities are defined as stated in Theorem 18.

PROOF. Games 0 is exactly as in the proof of Lemma 19. Game 1 is as in the proof of Lemma 19, except that now we are guessing which initiator oracle will be the first to accept maliciously (and its intended partner), and so the abort occurs if a responder oracle accepts maliciously. The loss of n^2 incurred by selection of parties is the same.

Again, the next step is a reduction that plays against SEUF-CMA-Q of the MAC scheme MAC. However, the reduction of course must behave slightly differently, since it must send to its own $\mathcal{O}_{\text{Vrfy}}$ oracle any message that was called as a Send query for the targeted initiator oracle, but which was not given as an output protocol message by a NewSessionR query (to responder oracle j^*). Further, we need to ensure that the forgery attempt is on an oracle that does not have a matching conversation with any others: in the proof of Lemma 19 this was straightforward since there was only one unique message in question, but here the transcripts that we are interested in consist of (up to) two flows between each oracle. This is just a matter of bookkeeping, and as before $\mathcal{B}_{18.1i}$ forwards all attempted forgeries to its own verification oracle, so any query that would have caused \mathcal{A} to win the entity authentication game (in the simulation that \mathcal{A} is experiencing) also corresponds to success in the game that $\mathcal{B}_{18.1i}$ is playing. \square

Lemma 21 (Key-Ind of PP2). *For any adversary \mathcal{A} and (any fixed) entity authentication adversary $\mathcal{A}_{18.2}$, the probability that \mathcal{A} answers the Test challenge correctly can be bounded by*

$$\text{Adv}_{\Pi}^{\text{Key-Ind}}(\mathcal{A}) \leq \text{Adv}_{\Pi}^{\text{Ent-Auth}}(\mathcal{A}_{18.2}) + n^2 \cdot q \cdot \text{Adv}_{\text{PPRF}}^{\text{rand}}(\mathcal{B}_{18.2})$$

where all quantities are defined as stated in Theorem 18.

PROOF. Let b' be the bit output by \mathcal{A} in each game, and b be the bit sampled as part of the Test query.

Game 0. This is the original Key-Ind game. We have

$$\Pr [G_{\Pi}^{\text{Key-Ind}}(\mathcal{A}) = 1] = \Pr [G_0^{\mathcal{A}} = 1].$$

Game 1. This game is the same as Game 0, except the challenger aborts and chooses $b' \stackrel{\$}{\leftarrow} \{0, 1\}$ if any oracle accepts maliciously. We have

$$\Pr [G_0^{\mathcal{A}} = 1] \leq \Pr [G_1^{\mathcal{A}} = 1] + \text{Adv}_{\Pi}^{\text{Ent-Auth}}(\mathcal{A}_{21}).$$

At this stage, the oracle to which \mathcal{A} asks its Test query has a unique partner oracle with a matching conversation. (Recall that we only consider adversaries that terminate with valid outputs, and further if \mathcal{A} does anything that would trigger a trivial loss in the original Key-Ind game then it loses in all games in this proof.)

Game 2. This game is the same as Game 1, except the challenger guesses $(i^*, j^*, s^*) \leftarrow^{\$} [n] \times [n] \times [q]$, and if \mathcal{A} issues a Test(π_i^s) query with $(i^*, s^*) \neq (i, s)$ or $j^* \neq \pi_{i^*}^{s^*}$.pid then the challenger aborts. We have

$$\Pr [G_1^{\mathcal{A}} = 1] = n^2 \cdot q \cdot \Pr [G_2^{\mathcal{A}} = 1].$$

Now $\pi_{i^*}^{s^*}$ is the oracle to which the Test query will be asked, and this oracle has unique partner $\pi_{j^*}^{t^*}$ with a matching conversation.

Game 3. In this game, the challenger responds to the Test query with a random element of $\mathcal{K}_{\text{PPRF}}$, the output space of PPRF.

We construct a reduction $\mathcal{B}_{18.2}$, detailed in Fig. 16 that runs an adversary that attempts to distinguish Game 3 from Game 2, while $\mathcal{B}_{18.2}$ is playing the rand game.

$$\Pr [G_2^{\mathcal{A}} = 1] = \Pr [G_3^{\mathcal{A}} = 1] + \text{Adv}_{\text{PPRF}}^{\text{rand}}(\mathcal{B}_{18.2})$$

If $b = 1$ in the rand game then $\mathcal{B}_{18.2}$ perfectly simulates Game 2 for \mathcal{A} , while if $b = 0$ in the rand game then $\mathcal{B}_{18.2}$ perfectly simulates Game 3 for \mathcal{A} .

The presentation in Fig. 16 is given for clarity, and omits a number of bookkeeping tasks performed by $\mathcal{B}_{18.2}$, such as managing and updating execution state values, partner identifiers, session key freshness values, security bit values and transcripts for all oracles. Further, if \mathcal{A} makes an invalid query, such as a Send query to an oracle that has already entered $\alpha \in \{\text{accept}, \text{reject}\}$ then $\mathcal{B}_{18.2}$ replies with \perp . Likewise if the adversary submits a Test query to an oracle that has not entered into an accept state or either it or its partner were corrupted before acceptance occurred, then $\mathcal{B}_{18.2}$ will do Abort. Recall that since the MAC keys do not update, $K_{ij}^{\text{MAC}} = K_{ji}^{\text{MAC}}$ throughout, while the key derivation keys k_{ij} and k_{ji} initially start as the same value but may differ as the protocol progresses.

In Game 3 the adversary's advantage of winning is zero, since a Test query will always return a random key that is independent of the protocol,

$$\Pr [G_3^{\mathcal{A}} = 1] = 0.$$

□

5.3.2 SR of PP2

We will now prove that PP2 achieves full synchronization robustness (SR). Intuitively we want to show that any adversary, making arbitrary message delivery queries between any of the parties (and their session oracles), cannot cause an adversarially chosen but honestly executed target protocol run to break down.

The robustness proof essentially needs three arguments: 1) the adversary cannot forge protocol messages without breaking the security of the MAC, 2) replaying messages from the target protocol run to other oracles is not beneficial to the adversary, and 3) the correctness of the PPRF ensures that interleaving queries with nonce values different to the one used in the target session will not influence the successful computation of a session key in the target session.

Theorem 22 (SR of PP2). *Let Π be the two-message protocol in Fig. 15, built using $\text{MAC} = \{\text{KGen}, \text{Mac}, \text{Vrfy}\}$ and PPRF, with n parties. Then for any adversary \mathcal{A} against the SR security of Π that makes a maximum of q queries that initiate new sessions for each party (with $q < |\text{CTR}|$), there exists an adversary \mathcal{B}_{22} against the SEUF-CMA- Q of MAC such that*

$$\text{Adv}_{\Pi}^{\text{SR}}(\mathcal{A}) \leq n^2 \cdot \text{Adv}_{\text{MAC}}^{\text{SEUF-CMA-}Q}(\mathcal{B}_{22}).$$

PROOF. We proceed using a sequence of games.

Game 0. This is the original SR game. We have

$$\Pr [G_{\Pi}^{\text{SR}}(\mathcal{A}) = 1] = \Pr [G_0^{\mathcal{A}} = 1].$$

Note that that Theorem 22 requires that the adversary only initiates $q < |\text{CTR}|$ sessions (which can be guaranteed by choosing $|\text{CTR}|$ exponential in the security parameter), implying that the counter is never exhausted and thus, the value N^* contained in the first protocol message is *unique*.

Game 1. This game is the same as Game 0 except that we define an event E_{22} , that is said to occur if the adversary successfully forges a MAC while it is running, and the challenger aborts if E_{22} occurs. We have

$$\Pr [G_0^{\mathcal{A}} = 1] = \Pr [G_1^{\mathcal{A}} = 1] + \Pr [E_{22}].$$

We now bound the probability that E_{22} occurs. We construct a reduction \mathcal{B}_{22} to the SEUF-CMA- Q security of MAC with a verification oracle. First, the reduction guesses which parties will be involved in the forgery that triggers E_{22} , and then simulates the environment of Game 0. To do this, \mathcal{B}_{22} must select (initial) PPRF keys from the appropriate keyspace and randomly pick MAC keys for all pairs of parties except for the

guessed parties i^* and j^* . It is then simple to simulate all queries to oracles of parties except communication between the guessed pair. For any query to an oracle of party i^* with $\text{pid} = j^*$ or an oracle of j^* with $\text{pid} = i^*$, the reduction needs to forward queries to its own MAC and verification oracles. The full reduction is detailed in Fig. 17.

We have

$$\Pr [E_{22}] \leq n^2 \cdot \text{Adv}_{\text{MAC}}^{\text{SEUF-CMA-Q}}(\mathcal{B}_{22}).$$

Game 2. This game is the same as Game 1 except that we add an additional requirement to its winning condition: the adversary may not issue queries to interrupt the target protocol execution between π_i^s and π_j^s . Note that this is the additional winning condition required by the weak synchronization robustness experiment. We claim

$$\Pr [G_1^A = 1] = \Pr [G_2^A = 1].$$

To prove the above equality, we need to take a closer look at the sequence of queries made by \mathcal{A} . Let π_i^s and π_j^s be the oracles of the targeted protocol execution. That is, the adversary needs to query

$$\text{NewSessionI}(\pi_i^s, j) \rightarrow m_1, \quad \text{NewSessionR}(\pi_j^t, i, m_1) \rightarrow m_2, \quad \text{Send}(\pi_i^s, m_2)$$

during its runtime, where m_1 contains some nonce N^* . While the adversary has to make those three queries in order, it may interleave the queries with queries of different protocol executions. We make the following three observations:

- Sending the message m_1 to any oracle apart from π_j^t will either make the oracle abort without any modification to k_{ij} , k_{ji} or CTR_{ij} , or make the adversary immediately lose. We distinguish three cases. (i) m_1 is sent to an oracle $\pi_{j'}^{t'}$ with $j \neq j'$. In this case the MAC cannot be verified and that oracle will abort without any modification to k_{ij} , k_{ji} or CTR_{ij} . (ii) m_1 is sent to an oracle $\pi_{j'}^{t'}$ with $t' \neq t$ before it is sent to π_j^t . In this case the oracle $\pi_{j'}^{t'}$ would accept the message, however, the adversary is now unable to output its intended oracles, since the transcripts involve a message that was sent earlier. (iii) m_1 is sent to an oracle $\pi_{j'}^{t'}$ with $t' \neq t$ after it was sent to π_j^t . In this case $\pi_{j'}^{t'}$ will abort after receiving m_1 ($x_B = \perp$ since π_j^t already computed $k_{ji} \leftarrow \text{Punct}(k_{ji}, N^*)$), without any modification to k_{ji} .
- Sending the message m_2 to any oracle apart from π_i^s will make the oracle abort without any modification to k_{ij} , k_{ji} or CTR_{ij} . We distinguish two cases. (i) m_2 is sent to an oracle $\pi_{i'}^{s'}$ with $i \neq i'$. In this case the MAC cannot be verified and that oracle will abort without any modification to k_{ij} , k_{ji} or CTR_{ij} . (ii) m_2 is sent to an oracle $\pi_{i'}^{s'}$ with $s' \neq s$. In this case the uniqueness of N^* guarantees that the MAC verification will fail and the oracle will abort without any modification to k_{ij} or CTR_{ij} .

- Note that the adversary may not ‘replace’ m_1 or m_2 in the target protocol run as this would immediately result in a loss for \mathcal{A} , as the target protocol run would not consist of an honest protocol run anymore.

We conclude that sending m_1 or m_2 to any oracles apart from \mathcal{A} ’s respective target oracles will either have no impact on the keys of the target parties or the counter CTR_{ij} and thus have no impact on the target protocol run, or make the target run ineligible in the SR game.

It now remains to show that the adversary cannot use any queries of different protocol runs to win the robustness experiment. We do this by “isolating” queries made during the queries related to the target protocol run. We start with the observation that any queries related to protocol runs with parties $i' \neq i$ and $j' \neq j$ does not have any influence over the target protocol run. Hence, the adversary does not gain any advantage issuing those queries in a way that interleaves with the target protocol run.

At this point, we need to consider the remaining possible queries for any oracle $\pi_i^{s'}$ with $s' \neq s$ or $\pi_j^{t'}$ with $t' \neq t$, which can be interleaved with the target protocol run. The adversary gains an advantage if it is capable of modifying the global session variables, here $(k_{ij}, k_{ji}, K_{ij}^{\text{MAC}} = K_{ji}^{\text{MAC}}, \text{CTR}_{ij})$, in such a way that the target protocol run aborts. We show that there is no such possible query that influences the outcome of the target session by a case distinction:

NewSessionI queries. Any NewSessionI query will cause the counter CTR of the initiator i to be incremented. However, since the counter is only relevant during the generation of the first protocol message, and since the initial message m_1 of the target run has already been generated, this has no impact on the target protocol run.

NewSessionR queries. Any NewSessionR query for $\pi_j^{t'}$ with $t' \neq t$ will either result in the receiving oracle aborting the protocol run (either due to an invalid MAC or a replayed first protocol message), or in puncturing k_{BA} at some position N . Since all values of N are unique and the adversary cannot re-use the first message of the target run m_1 , the adversary cannot cause a puncturing operation on the value N^* contained in the target protocol run. The correctness of the PPRF then guarantees that a consistent evaluation for N^* is possible as long as only values $N \neq N^*$ have been punctured.

Send queries. Any Send query for $\pi_i^{s'}$ with $s' \neq s$, if it does not abort due to an invalid MAC, will puncture k_{ij} (or another key belonging to i , though this does not assist the adversary) at some position N . Any oracle $\pi_i^{s'}$ with $s' \neq s$ will always puncture the PPRF key for some value N , which is not equal to the value N^* contained in the target session. This is ensured by the uniqueness of the counter during the adversary’s runtime and by the session storing its respective value N during initialization. Hence, only the target initiator oracle π_i^s is able to puncture the PPRF key

for the value N^* . The correctness of the PPRF then guarantees that a consistent evaluation for N^* is possible as long as only values $N \neq N^*$ have been punctured.

We have now exhausted all possible options for the adversary to cause a disturbance of the target protocol run, either via re-using values of the target protocol run before it is concluded, or via interleaving any other protocol message during the target protocol run. Both of which yield no advantage for the adversary. We hence have

$$\Pr [G_1^{\mathcal{A}} = 1] = \Pr [G_2^{\mathcal{A}} = 1].$$

Bounding the advantage of \mathcal{A} . It remains to bound the adversary's advantage in Game 2. Recall that 1) the adversary can now only execute a complete protocol run between the target session, which has full matching transcripts and is not interrupted by any other queries. Furthermore, for any protocol run between two fixed parties, the value N^* is unique, which ensures that for any protocol run, the PPRF key remains not punctured at N^* . In this case, the correctness of PP2 ensures that the target protocol runs will not abort and, in particular, will successfully derive the same session key. We get

$$\Pr [G_2^{\mathcal{A}} = 1] = 0.$$

□

5.4 PP1: a One-Message Protocol with Fixed Roles

In Figure 18 we give a one-message protocol that uses a PPRF in a very similar way to our two-message protocol. Here we only get one-sided entity authentication, but the proofs are very similar to the ones for the two-message protocol.

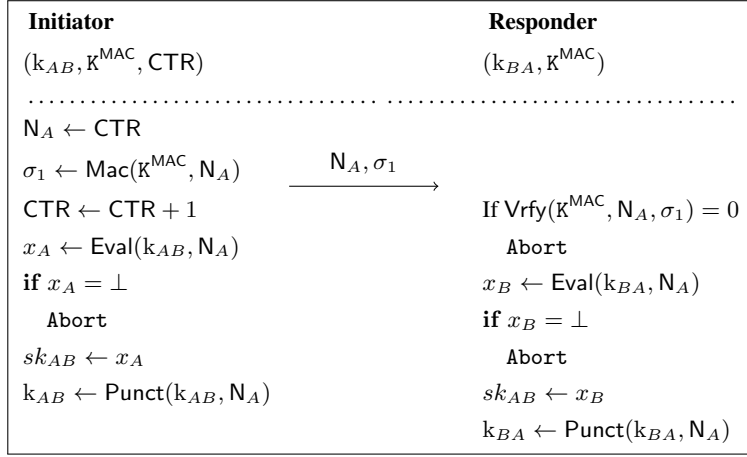


Figure 18: One-message symmetric AKE protocol that tolerates concurrent sessions, using a Puncturable PRF $\text{PPRF} = (\text{Setup}, \text{Eval}, \text{Punct})$.

5.4.1 AKE-R of PP1

Theorem 23 (AKE-R of PP1). *Let Π be the one-message protocol in Fig. 18, built using $\text{MAC} = \{\text{KGen}, \text{Mac}, \text{Vrfy}\}$ and $\text{PPRF} = (\text{Setup}, \text{Eval}, \text{Punct})$ with n parties. Then for any adversary \mathcal{A} against the AKE-R security of Π that makes a maximum of q queries that initiate new sessions for each party (with $q < |\text{CTR}|$), there exists an adversary $\mathcal{B}_{23.1}$ against the SEUF-CMA- Q of MAC and an adversary $\mathcal{B}_{23.2}$ against the rand security of PPRF such that*

$$\text{Adv}_{\Pi}^{\text{AKE-R}}(\mathcal{A}) \leq 2n^2 \cdot \text{Adv}_{\text{MAC}}^{\text{SEUF-CMA-}Q}(\mathcal{B}_{23.1}) + n^2 \cdot q \cdot \text{Adv}_{\text{PPRF}}^{\text{rand}}(\mathcal{B}_{23.2}).$$

The Ent-Auth-R analysis is essentially the same as in Lemma 19 and leads to the same term collection, since in PP1 the responder performs the same notable actions as in PP2 (the creation of the second protocol message is not necessary).

Proving Key-Ind is also similar to that of Lemma 21, leading to the same term collection. The final reduction is slightly more straightforward than $\mathcal{B}_{18.2}$ since the only Send queries that $\mathcal{B}_{23.2}$ needs to deal with are those that are part of NewSessionR queries.

5.4.2 SR of PP1

Theorem 24 (SR of PP1). *Let Π be the two-message protocol in Fig. 18, built using $\text{MAC} = \{\text{KGen}, \text{Mac}, \text{Vrfy}\}$ and PPRF, with n parties. Then for any adversary \mathcal{A} against the SR security of Π that makes a maximum of q queries that initiate new sessions*

for each party (with $q < |\text{CTR}|$), there exists an adversary \mathcal{B}_{24} against the SEUF-CMA- Q of MAC such that

$$\text{Adv}_{\Pi}^{\text{SR}}(\mathcal{A}) \leq n^2 \cdot \text{Adv}_{\text{MAC}}^{\text{SEUF-CMA-}Q}(\mathcal{B}_{24}).$$

PROOF. The strategy for this proof is very similar to that of Theorem 22, and the term collection is the same: only the reduction logic is different (since there are no second protocol messages to deal with). The reduction \mathcal{B}_{24} is detailed in Fig. 19. Note that for one-message protocols, there are no Send queries that were not initially called by NewSessionR, since those queries will always result in the oracle going to state `accept` or `reject` (in other words, an oracle cannot ever be `negotiating` at the point an adversary makes a query). However, formatting is maintained to provide easier comparison with reduction \mathcal{B}_{22} . □

5.5 Instantiation

It remains to discuss how PP2 can be instantiated with a PPRF and what impact the PPRF has on its efficiency. A promising candidate is the Goldreich–Goldwasser–Micali PRF [GGM86], which can be transformed to a PPRF [BW13, KPTZ13, BGI14]. We give an intuitive explanation of the construction and refer the reader to [AGJ19] for a more detailed description and analysis. This construction is especially suitable, as both the PPRF evaluation and puncturing are solely based on hash function evaluations in practice.

Intuition. The tree-based PPRF uses two functions H_0 and H_1 both mapping from $\{0, 1\}^\lambda$ to $\{0, 1\}^\lambda$. For every input $x \in \{0, 1\}^\lambda$ of the PPRF, the binary representation of x prescribes the sequence in which H_0 and H_1 have to be repeatedly applied to x . For example, $\text{Eval}(01) = H_1(H_0(x))$. Note that the evaluation of x corresponds to a path through a binary tree, where each bit in x tells you whether to take a “left” or “right” path. The result of an evaluation always corresponds to a leaf in the binary tree.

The initial PPRF key consists of the root node, which is initialized during key generation as a randomly chosen string. To puncture values (i.e., to puncture leaves of the tree), we precompute and store all nodes on the co-path between the root and the leaf, before deleting all parent nodes (including the root node) of the leaf. Note that this procedure can be repeated for any of the leaves and note that it satisfies all puncturing-relevant properties (i.e., re-computation of $\text{Eval}(x)$ is not possible but the correctness of the PPRF remains intact).

Memory Consumption. We briefly discuss the memory consumed by the PPRF during the lifetime of PP2 (and PP1). First, note that the PPRF-based protocols deploy

counters, which (if all messages are delivered in sequence) ensure a systematic puncturing from the leftmost leaf to the rightmost leaf of the binary tree. This yields the need to store at most $\log(|\text{CTR}|)$ tree nodes (i.e., at most one node per layer of the tree) at any point in time. For C concurrent sessions, this bound increases to a maximum of $C \cdot \log(|\text{CTR}|)$ tree nodes.

The analysis gets slightly more difficult if an adversary *actively drops* protocols messages. Each dropped message will either cause the initiator or both parties to not puncture at some position. One approach to tame the memory consumption in this case, would be to always puncture on all values which are smaller than $\text{CTR} - C$.¹ As we never expect more than C sessions in parallel, this reduces additional memory caused by lost messages. In this case, the memory consumption is again upper-bounded by $C \cdot \log(|\text{CTR}|)$ tree nodes.

Finally, note that in the one-message protocol PP1 (Sec. 5.4) the initiator always punctures strictly in order and thus has to store at most $\log(|\text{CTR}|)$ tree nodes. This may be particularly useful in an application where many low-end devices communicate with a central server.

Case Study. To provide some intuition on how efficient our PPRF-based protocols are, we present a brief toy example. Consider a sensor device that commences communication with a central hub on average six times per hour, with an expected lifetime of 15 years. We can upper bound the expected number of sessions $|\text{CTR}|$ by 2^{20} (since $6 \cdot 24 \cdot 365 \cdot 15 \approx 2^{19.6}$), so an instantiation of the GGM-based PPRF with H_0 and H_1 as the left and right halves of SHA-256 outputs respectively (a tree node thus has size 16 bytes), produces a punctured key with size upper-bounded by $\log(|\text{CTR}|) \cdot 16 \cdot C = 320 \cdot C$ bytes. (More generally, $|\text{CTR}| = 2^{64}$ should suffice for any conceivable application, in this case the upper bound on the key size is $1024 \cdot C$ bytes.)

For computation, in the worst case $\log(|\text{CTR}|) = 20$ SHA computations are required per evaluation/puncturing operation, and fewer on average (this depends on the position in the tree; some puncture operations require no computations but only a deletion).

¹Interestingly, the tree-based PPRF can puncture multiple values in one go by “chopping off” whole branches of the tree, instead of puncturing all values one after another.

Reduction \mathcal{B}_{17} playing $G_{\text{MAC}}^{\text{SEUF-CMA-Q}}(\mathcal{B}_{17})$

```

1:  $i^*, j^* \xleftarrow{\$} [n]$ 
2: for  $i, j \in [n]$  do
3:    $k_{ij}^{\text{CTR}} = k_{ji}^{\text{CTR}} \xleftarrow{\$} \mathcal{K}_{\text{PRF}}$ 
4:   for  $n \times [n] \setminus (i^*, j^*)$  do
5:      $K_{ij}^{\text{MAC}} = k_{\text{MAC}}^{i,i} \xleftarrow{\$} \mathcal{K}_{\text{MAC}}$ 
6:      $\text{CTR}_{ij} = \text{CTR}_{ji} \leftarrow 0$ 
7:    $\mathcal{A}^{\text{oracles}}$ 

```

$\text{NewSessionI}(\pi_i^s, \text{pid})$

```

8:  $\pi_i^s.\rho \leftarrow \text{Initiator}$ 
9:  $\pi_i^s.\alpha \leftarrow \text{negotiating}$ 
10:  $\pi_i^s.\text{pid} \leftarrow \text{pid} \quad / = j$ 
11: if  $(i, j) = (i^*, j^*) \vee (j^*, i^*)$ 
12:    $\sigma_1 \leftarrow \text{call } \mathcal{O}_{\text{Mac}}(\text{CTR}_{ij})$ 
13: else
14:    $\sigma_1 \leftarrow \text{Mac}(K_{ij}^{\text{MAC}}, \text{CTR}_{ij})$ 
15:  $m' \leftarrow \text{CTR}_{ij}, \sigma_1$ 
16: return  $m'$ 
17:  $\pi_i^s.sk \leftarrow \text{KDF}(k_{ij}^{\text{CTR}}, \text{"der"})$ 
18:  $\pi_i^s.\alpha \leftarrow \text{accept}$ 
19:  $k_{ij}^{\text{CTR}}, \text{CTR}_{ij} \leftarrow \text{Advnc}(k_{ij}^{\text{CTR}}, \text{CTR}_{ij}, 1)$ 

```

$\text{NewSessionR}(\pi_i^s, \text{pid}, m)$

```

20:  $\pi_i^s.\rho \leftarrow \text{Responder}$ 
21:  $\pi_i^s.\alpha \leftarrow \text{negotiating}$ 
22:  $\pi_i^s.\text{pid} \leftarrow \text{pid}$ 
23: do  $\text{Send}(\pi_i^s, m)$ 

```

$\text{Send}(\pi_i^s, m) \quad / \text{pid} = j$

```

24: Parse  $m$  as  $\text{CTR}_{ji}, \sigma_1$ 
25: if  $(i, j) = (i^*, j^*) \vee (j^*, i^*)$ 
26:    $b \leftarrow \text{call } \mathcal{O}_{\text{Vrfy}}(\text{CTR}_{ji}, \sigma_1)$ 
27:   if  $b = 0$ 
28:     return  $\perp$ 
29:   else
30:     if  $\text{Vrfy}(K_{ij}^{\text{MAC}}, \text{CTR}_{ij}, \sigma_1) = 0$ 
31:       return  $\perp$ 
32:     if  $\pi_i^s.\rho = \text{Initiator}$ 
33:       return  $\perp$ 
34:      $z_1 \leftarrow \text{CTR}_{ji} - \text{CTR}_{ij}$ 
35:     if  $z_1 < 0$ 
36:       return  $\perp$ 
37:      $k_{ij}^{\text{CTR}}, \text{CTR}_{ij} \leftarrow \text{Advnc}(k_{ij}^{\text{CTR}}, \text{CTR}_{ij}, z_1)$ 
38:      $\pi_i^s.sk \leftarrow \text{KDF}(k_{ij}^{\text{CTR}}, \text{"der"})$ 
39:      $\pi_i^s.\alpha \leftarrow \text{accept}$ 
40:      $k_{ij}^{\text{CTR}}, \text{CTR}_{ij} \leftarrow \text{Advnc}(k_{ij}^{\text{CTR}}, \text{CTR}_{ij}, 1)$ 

```

Figure 13: Reduction \mathcal{B}_{17} for the proof of Theorem 17. \mathcal{B}_{17} provides \mathcal{A} with access to $\text{oracles} = \text{NewSessionI}(\cdot, \cdot, \cdot), \text{NewSessionR}(\cdot, \cdot), \text{Send}(\cdot, \cdot, \cdot)$.

$\text{Grand}_{\text{PPRF}}(\mathcal{A})$	$\mathcal{O}_{\text{Eval}}(x)$
$k \xleftarrow{\$} \text{Setup}(1^\lambda)$	$y \leftarrow \text{Eval}(k, x)$
$b \xleftarrow{\$} \{0, 1\}; \mathcal{Q} := \emptyset$	$\mathcal{Q} := \mathcal{Q} \cup \{x\}$
$x^* \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{\text{Eval}}(\cdot)}(1^\lambda)$	$k \leftarrow \text{Punct}(k, x)$
$y_0 \xleftarrow{\$} \mathcal{R}_{\text{PPRF}}; y_1 \leftarrow \text{Eval}(k, x^*)$	return y
$k \leftarrow \text{Punct}(k, x^*)$	
$b^* \xleftarrow{\$} \mathcal{A}(k, y_b)$	
return 1 if $b = b^*$ and $x^* \notin \mathcal{Q}$	
return 0	

Figure 14: The rand security experiment for puncturable PRF PPRF.

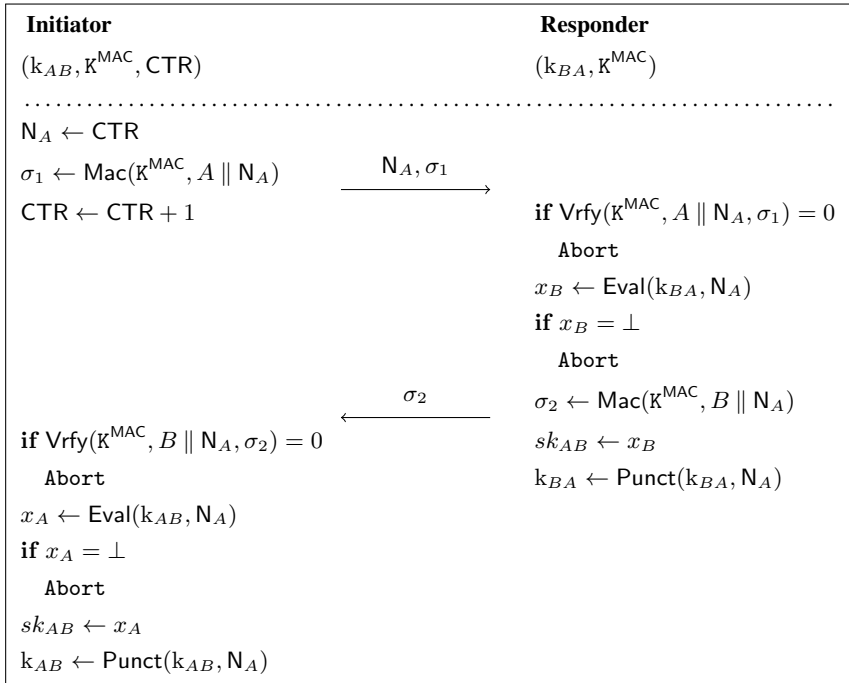


Figure 15: A symmetric AKE protocol PP2 that tolerates concurrent sessions, using a puncturable PRF PPRF = (Setup, Eval, Punct).

Reduction $\mathcal{B}_{18.2}$ playing $\text{Grand}_{\text{PPRF}}(\mathcal{B}_{18.2})$

```

1 :  $i^*, j^* \xleftarrow{\$} [n]; s^* \xleftarrow{\$} [q]$ 
2 :  $\text{CTR}_i = 0$  for all  $i \in \{1, \dots, n\}$ 
3 : for  $i, j \in [n]$  do
4 :    $K_{ij}^{\text{MAC}} = K_{ji}^{\text{MAC}} \xleftarrow{\$} \mathcal{K}_{\text{MAC}}$ 
5 :    $\text{CTR}_{ij} \leftarrow 0$ 
6 : for  $n \times [n] \setminus (i^*, j^*)$  do
7 :    $k_{ij} = k_{ji} \xleftarrow{\$} \mathcal{K}_{\text{PPRF}}$ 
8 :  $\mathcal{A}^{\text{oracles}}$ 
9 : When  $\mathcal{A}$  calls  $\text{Test}(\pi_i^s)$  do
10 :   if  $(i, s) \neq (i^*, s^*)$  or  $j^* \neq \pi_{i^*}^{s^*}.\text{pid}$ 
11 :     return Abort
12 :   submit  $N^*$ , receive  $(k^*, y^*)$ 
13 :    $(i^*, s^*, b') \xleftarrow{\$} \mathcal{A}(y^*)$ 
14 :   return  $b'$ 

```

NewSessionI(π_i^s, pid)

```

15 :  $\pi_i^s.\rho \leftarrow \text{Initiator}$ 
16 :  $\pi_i^s.\alpha \leftarrow \text{negotiating}$ 
17 :  $\pi_i^s.\text{pid} \leftarrow \text{pid}$ 
18 :  $N \leftarrow \text{CTR}_{ij}$ 
19 :  $\text{CTR}_{ij} \leftarrow \text{CTR}_{ij} + 1$ 
20 :  $\sigma_1 \leftarrow \text{Mac}(K_{ij}^{\text{MAC}}, i \parallel N)$ 
21 :  $m' \leftarrow N, \sigma_1$ 
22 : return  $m'$ 

```

NewSessionR(π_i^s, pid, m)

```

23 :  $\pi_i^s.\rho \leftarrow \text{Responder}$ 
24 :  $\pi_i^s.\alpha \leftarrow \text{negotiating}$ 
25 :  $\pi_i^s.\text{pid} \leftarrow \text{pid}$ 
26 : do Send( $\pi_i^s, m$ )

```

RevealKey(π_i^s)

```

27 : if  $\pi_i^s.\alpha \neq \text{accept}$ 
28 :   return  $\perp$ 
29 : if  $(i, s) = (i^*, s^*)$ 
30 :   return Abort
31 :  $\pi_i^s.\kappa \leftarrow \text{exposed}$ 
32 : return  $\pi_i^s.sk$ 

```

Corrupt(P_i, P_j)

```

33 : if  $i \in \{i^*, j^*\}$  or  $j \in \{i^*, j^*\}$ 
34 :   if  $\pi_{i^*}^{s^*}.\alpha \neq \text{accept}$ 
35 :     return Abort
36 :   return  $k_{ij}$ 

```

Send($\pi_i^s, m, \text{pid} = j$)

```

37 : Parse  $m$  as  $N, \sigma_1$ 
38 : if  $\text{Vrfy}(K_{ij}^{\text{MAC}}, j \parallel N, \sigma_1) = 0$ 
39 :   return  $\perp$ 
40 : if  $(i, j) \neq (i^*, j^*) \vee (j^*, i^*)$ 
41 :    $x_i \leftarrow \text{Eval}(k_{ij}, N)$ 
42 : else / embed
43 :   if Test has not occurred
44 :      $x_i \leftarrow \text{call } \mathcal{O}_{\text{Eval}}(N)$ 
45 :   else
46 :      $x_i \leftarrow \text{Eval}(k_{ij}, N)$ 
47 :    $k_{ij} \leftarrow \text{Punct}(k_{ij}, N)$ 
48 : endif
49 : if  $x_i = \perp$ 
50 :   return  $\perp$ 
51 :  $sk_{ij} \leftarrow x_i$ 
52 :  $\pi_i^s.\alpha \leftarrow \text{accept}$ 
53 : if  $\pi_i^s.\rho = \text{Responder}$ 
54 :    $\sigma_2 \leftarrow \text{Mac}(K_{ij}^{\text{MAC}}, i \parallel N)$ 
55 :    $m' \leftarrow N, \sigma_2$ 
56 :   return  $m'$ 
57 : if  $(i, j) \neq (i^*, j^*) \vee (j^*, i^*)$ 
58 :    $k_{ij} \leftarrow \text{Punct}(k_{ij}, N)$ 
59 : endif

```

Figure 16: Reduction $\mathcal{B}_{18.2}$ for the proof of Lemma 21. N^* is the initial nonce sent in the transcript between $\pi_{i^*}^{s^*}$ and $\pi_{j^*}^{t^*}$. If at any time \mathcal{A} causes an oracle to accept maliciously, then $\mathcal{B}_{18.2}$ simply does Abort. $\mathcal{B}_{18.2}$ provides \mathcal{A} with access to $\text{oracles} = \text{NewSessionI}(\cdot, \cdot, \cdot), \text{NewSessionR}(\cdot, \cdot), \text{Send}(\cdot, \cdot, \cdot), \text{RevealKey}(\cdot), \text{Corrupt}(\cdot, \cdot)$.

<p>Reduction \mathcal{B}_{22} playing $G_{\text{MAC}}^{\text{SEUF-CMA-Q}}(\mathcal{B}_{22})$</p> <hr/> 1 : $i^*, j^* \xleftarrow{\$} [n]$ 2 : for $i, j \in [n]$ do 3 : $k_{ij} = k_{ji} \xleftarrow{\$} \mathcal{K}_{\text{PPRF}}$ 4 : $\text{CTR}_{ij} \leftarrow 0$ 5 : for $n \times [n] \setminus (i^*, j^*)$ do 6 : $K_{ij}^{\text{MAC}} = K_{ji}^{\text{MAC}} \xleftarrow{\$} \mathcal{K}_{\text{MAC}}$ 7 : $\mathcal{A}^{\text{oracles}}$	<p>NewSessionR(π_i^s, pid, m)</p> <hr/> 19 : $\pi_i^s.\rho \leftarrow \text{Responder}$ 20 : $\pi_i^s.\alpha \leftarrow \text{negotiating}$ 21 : $\pi_i^s.\text{pid} \leftarrow \text{pid}$ 22 : do Send(π_i^s, m)
<p>NewSessionI(π_i^s, pid)</p> <hr/> 8 : $\pi_i^s.\rho \leftarrow \text{Initiator}$ 9 : $\pi_i^s.\alpha \leftarrow \text{negotiating}$ 10 : $\pi_i^s.\text{pid} \leftarrow \text{pid} \quad / = j$ 11 : $N \leftarrow \text{CTR}_{ij}$ 12 : $\text{CTR}_{ij} \leftarrow \text{CTR}_{ij} + 1$ 13 : if $(i, j) = (i^*, j^*) \vee (j^*, i^*)$ 14 : $\sigma_1 \leftarrow \text{call } \mathcal{O}_{\text{Mac}}(i \parallel N)$ 15 : else 16 : $\sigma_1 \leftarrow \text{Mac}(K_{ij}^{\text{MAC}}, i \parallel N)$ 17 : $m' \leftarrow N, \sigma_1$ 18 : return m'	<p>Send(π_i^s, m) $/ \text{pid} = j$</p> <hr/> 23 : Parse m as N, σ_1 24 : if $(i, j) = (i^*, j^*) \vee (j^*, i^*)$ 25 : $b \leftarrow \text{call } \mathcal{O}_{\text{Vrfy}}(j \parallel N, \sigma_1)$ 26 : if $b = 0$ 27 : return \perp 28 : if $\pi_i^s.\rho = \text{Responder}$ 29 : $\sigma_2 \leftarrow \text{call } \mathcal{O}_{\text{Mac}}(i \parallel N)$ 30 : return $m' \leftarrow N, \sigma_2$ 31 : else $/ \text{simulate}$ 32 : if $\text{Vrfy}(K_{ij}^{\text{MAC}}, j \parallel N, \sigma_1) = 0$ 33 : return \perp 34 : if $\pi_i^s.\rho = \text{Responder}$ 35 : $\sigma_2 \leftarrow \text{Mac}(K_{ij}^{\text{MAC}}, i \parallel N)$ 36 : return $m' \leftarrow N, i, \sigma_2$ 37 : $sk_{ji} \leftarrow \text{Eval}(k_{ij}, N_j)$ 38 : $k_{ij} \leftarrow \text{Punct}(k_{ij}, N)$

Figure 17: Reduction \mathcal{B}_{22} for the proof of Theorem 22. \mathcal{B}_{22} provides \mathcal{A} with access to $\text{oracles} = \text{NewSessionI}(\cdot, \cdot, \cdot), \text{NewSessionR}(\cdot, \cdot), \text{Send}(\cdot, \cdot, \cdot)$.

<p>Reduction \mathcal{B}_{24} playing $G_{\text{MAC}}^{\text{SEUF-CMA-Q}}(\mathcal{B}_{24})$</p> <hr/> <pre> 1 : $i^*, j^* \xleftarrow{\\$} [n]$ 2 : for $i, j \in [n]$ do 3 : $k_{ij} = k_{ji} \xleftarrow{\\$} \mathcal{K}_{\text{PPRF}}$ 4 : $\text{CTR}_{ij} \leftarrow 0$ 5 : for $n \times [n] \setminus (i^*, j^*)$ do 6 : $K_{ij}^{\text{MAC}} = K_{ji}^{\text{MAC}} \xleftarrow{\\$} \mathcal{K}_{\text{MAC}}$ 7 : $\mathcal{A}^{\text{oracles}}$ </pre> <hr/> <p>NewSessionI(π_i^s, pid)</p> <hr/> <pre> 8 : $\pi_i^s.\rho \leftarrow \text{Initiator}$ 9 : $\pi_i^s.\alpha \leftarrow \text{negotiating}$ 10 : $\pi_i^s.\text{pid} \leftarrow \text{pid} \quad / = j$ 11 : $\text{N} \leftarrow \text{CTR}_{ij}$ 12 : if $(i, j) = (i^*, j^*) \vee (j^*, i^*)$ 13 : $\sigma_1 \leftarrow \text{call } \mathcal{O}_{\text{Mac}}(\text{N})$ 14 : else 15 : $\sigma_1 \leftarrow \text{Mac}(K_{ij}^{\text{MAC}}, \text{N})$ 16 : $m' \leftarrow \text{N}, \sigma_1$ 17 : $\text{CTR}_{ij} \leftarrow \text{CTR}_{ij} + 1$ 18 : $sk_{ji} \leftarrow \text{Eval}(k_{ij}, \text{N}_j)$ 19 : $k_{ij} \leftarrow \text{Punct}(k_{ij}, \text{N})$ 20 : return m' </pre>	<p>NewSessionR(π_i^s, pid, m)</p> <hr/> <pre> 21 : $\pi_i^s.\rho \leftarrow \text{Responder}$ 22 : $\pi_i^s.\alpha \leftarrow \text{negotiating}$ 23 : $\pi_i^s.\text{pid} \leftarrow \text{pid}$ 24 : do $\text{Send}(\pi_i^s, m)$ </pre> <hr/> <p>Send($\pi_i^s, m, \text{pid} = j$)</p> <hr/> <pre> 25 : Parse m as N, σ_1 26 : if $\pi_i^s.\rho = \text{Initiator}$ 27 : return Abort 28 : if $(i, j) = (i^*, j^*) \vee (j^*, i^*) \quad / \text{embed}$ 29 : $b \leftarrow \text{call } \mathcal{O}_{\text{Vrfy}}(\text{N}, \sigma_1)$ 30 : if $b = 0$ 31 : return \perp 32 : else $/ \text{simulate}$ 33 : If $\text{Vrfy}(K_{ij}^{\text{MAC}}, \text{N}, \sigma_1) = 0$ 34 : return \perp 35 : $sk_{ji} \leftarrow \text{Eval}(k_{ij}, \text{N}_j)$ 36 : $k_{ij} \leftarrow \text{Punct}(k_{ij}, \text{N})$ </pre>
---	---

Figure 19: Reduction \mathcal{B}_{24} for the proof of Theorem 24. \mathcal{B}_{24} provides \mathcal{A} with access to $\text{oracles} = \text{NewSessionI}(\cdot, \cdot, \cdot), \text{NewSessionR}(\cdot, \cdot), \text{Send}(\cdot, \cdot, \cdot)$.

Acknowledgements. In addition to the funding bodies acknowledged on page 1, we would also like to thank Luke Mather for numerous helpful comments.

References

- [ACD19] Joël Alwen, Sandro Coretti, and Yevgeniy Dodis. The double ratchet: Security notions, proofs, and modularization for the Signal protocol. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2019, Part I*, volume 11476 of *Lecture Notes in Computer Science*, pages 129–158, Darmstadt, Germany, May 19–23, 2019. Springer, Heidelberg, Germany.
- [ACF20] Gildas Avoine, Sébastien Canard, and Loïc Ferreira. Symmetric-key authenticated key exchange (SAKE) with perfect forward secrecy. In Stanislaw Jarecki, editor, *Topics in Cryptology – CT-RSA 2020*, volume 12006 of *Lecture Notes in Computer Science*, pages 199–224, San Francisco, CA, USA, February 24–28, 2020. Springer, Heidelberg, Germany.
- [AGJ19] Nimrod Aviram, Kai Gellert, and Tibor Jager. Session resumption protocols and efficient forward security for TLS 1.3 0-RTT. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2019, Part II*, volume 11477 of *Lecture Notes in Computer Science*, pages 117–150, Darmstadt, Germany, May 19–23, 2019. Springer, Heidelberg, Germany.
- [AGJ21] Nimrod Aviram, Kai Gellert, and Tibor Jager. Session resumption protocols and efficient forward security for TLS 1.3 0-RTT. *Journal of Cryptology*, 34(3):1–57, 2021.
- [ANS09] Retail Financial Services Symmetric Key Management Part 1: Using Symmetric Techniques (ANSI x9.24). Standard, American National Standards Institute, New York, USA, 2009.
- [BG20] Colin Boyd and Kai Gellert. A Modern View on Forward Security. *The Computer Journal*, 08 2020. <https://doi.org/10.1093/comjnl/bxaa104>.
- [BGI14] Elette Boyle, Shafi Goldwasser, and Ioana Ivan. Functional signatures and pseudorandom functions. In Hugo Krawczyk, editor, *PKC 2014: 17th International Conference on Theory and Practice of Public Key Cryptography*, volume 8383 of *Lecture Notes in Computer Science*, pages 501–519, Buenos Aires, Argentina, March 26–28, 2014. Springer, Heidelberg, Germany.

- [BGM04] Mihir Bellare, Oded Goldreich, and Anton Mityagin. The power of verification queries in message authentication and authenticated encryption. *Cryptology ePrint Archive*, Report 2004/309, 2004. <https://eprint.iacr.org/2004/309>.
- [BM99] Mihir Bellare and Sara K. Miner. A forward-secure digital signature scheme. In Michael J. Wiener, editor, *Advances in Cryptology – CRYPTO’99*, volume 1666 of *Lecture Notes in Computer Science*, pages 431–448, Santa Barbara, CA, USA, August 15–19, 1999. Springer, Heidelberg, Germany.
- [BP10] Eric Brier and Thomas Peyrin. A forward-secure symmetric-key derivation protocol - how to improve classical DUKPT. In Masayuki Abe, editor, *Advances in Cryptology – ASIACRYPT 2010*, volume 6477 of *Lecture Notes in Computer Science*, pages 250–267, Singapore, December 5–9, 2010. Springer, Heidelberg, Germany.
- [BPR00] Mihir Bellare, David Pointcheval, and Phillip Rogaway. Authenticated key exchange secure against dictionary attacks. In Bart Preneel, editor, *Advances in Cryptology – EUROCRYPT 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 139–155, Bruges, Belgium, May 14–18, 2000. Springer, Heidelberg, Germany.
- [BR94] Mihir Bellare and Phillip Rogaway. Entity authentication and key distribution. In Douglas R. Stinson, editor, *Advances in Cryptology – CRYPTO’93*, volume 773 of *Lecture Notes in Computer Science*, pages 232–249, Santa Barbara, CA, USA, August 22–26, 1994. Springer, Heidelberg, Germany.
- [BW13] Dan Boneh and Brent Waters. Constrained pseudorandom functions and their applications. In Kazue Sako and Palash Sarkar, editors, *Advances in Cryptology – ASIACRYPT 2013, Part II*, volume 8270 of *Lecture Notes in Computer Science*, pages 280–300, Bangalore, India, December 1–5, 2013. Springer, Heidelberg, Germany.
- [BY03] Mihir Bellare and Bennet S. Yee. Forward-security in private-key cryptography. In Marc Joye, editor, *Topics in Cryptology – CT-RSA 2003*, volume 2612 of *Lecture Notes in Computer Science*, pages 1–18, San Francisco, CA, USA, April 13–17, 2003. Springer, Heidelberg, Germany.
- [CK01] Ran Canetti and Hugo Krawczyk. Analysis of key-exchange protocols and their use for building secure channels. In Birgit Pfitzmann, editor, *Advances in Cryptology – EUROCRYPT 2001*, volume 2045 of *Lecture Notes in Computer Science*, pages 453–474, Innsbruck, Austria, May 6–10, 2001. Springer, Heidelberg, Germany.

- [CMA05] (NIST SP)-800-38B. recommendation for block cipher modes of operation: The CMAC mode for authentication. Special Publication. Standard, NIST, 2005.
- [CRSS20] Valerio Cini, Sebastian Ramacher, Daniel Slamanig, and Christoph Striecks. CCA-secure (puncturable) KEMs from encryption with non-negligible decryption errors. In Shiho Moriai and Huaxiong Wang, editors, *Advances in Cryptology – ASIACRYPT 2020, Part I*, volume 12491 of *Lecture Notes in Computer Science*, pages 159–190, Daejeon, South Korea, December 7–11, 2020. Springer, Heidelberg, Germany.
- [DGJ⁺21] David Derler, Kai Gellert, Tibor Jager, Daniel Slamanig, and Christoph Striecks. Bloom filter encryption and applications to efficient forward-secret 0-RTT key exchange. *Journal of Cryptology*, 34(2):1–59, 2021.
- [DJ15] Mohammad Sadeq Dousti and Rasool Jalili. FORSAKES: A forward-secure authenticated key exchange protocol based on symmetric key-evolving schemes. *Adv. Math. Commun.*, 9(4):471–514, 2015.
- [DJSS18] David Derler, Tibor Jager, Daniel Slamanig, and Christoph Striecks. Bloom filter encryption and applications to efficient forward-secret 0-RTT key exchange. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2018, Part III*, volume 10822 of *Lecture Notes in Computer Science*, pages 425–455, Tel Aviv, Israel, April 29 – May 3, 2018. Springer, Heidelberg, Germany.
- [GGM86] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *Journal of the ACM*, 33(4):792–807, October 1986.
- [GHJL17] Felix Günther, Britta Hale, Tibor Jager, and Sebastian Lauer. 0-RTT key exchange with full forward secrecy. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *Advances in Cryptology – EUROCRYPT 2017, Part III*, volume 10212 of *Lecture Notes in Computer Science*, pages 519–548, Paris, France, April 30 – May 4, 2017. Springer, Heidelberg, Germany.
- [GM15] Matthew D. Green and Ian Miers. Forward secure asynchronous messaging from puncturable encryption. In *2015 IEEE Symposium on Security and Privacy*, pages 305–320, San Jose, CA, USA, May 17–21, 2015. IEEE Computer Society Press.
- [GMA07] (NIST SP)-800-38D. Recommendation for block cipher modes of operation: Galois/Counter Mode (GCM) and GMAC. Special Publication. Standard, NIST, 2007.

- [HMA08] FIPS 198-1. the Keyed-Hash Message Authentication Code (HMAC). Standard, NIST, 2008.
- [ISO11] ISO/IEC 9797-1:2011. Message Authentication Codes (MACs) – part 1: Mechanisms using a block cipher. Standard, International Organization for Standardization / International Electrotechnical Commission, 2011.
- [JKSS12] Tibor Jager, Florian Kohlar, Sven Schäge, and Jörg Schwenk. On the security of TLS-DHE in the standard model. In Reihaneh Safavi-Naini and Ran Canetti, editors, *Advances in Cryptology – CRYPTO 2012*, volume 7417 of *Lecture Notes in Computer Science*, pages 273–293, Santa Barbara, CA, USA, August 19–23, 2012. Springer, Heidelberg, Germany.
- [KMA16] (NIST SP)-800-185. SHA-3 derived functions: cSHAKE, KMAC, Tuple-Hash and ParallelHash. Special Publication. Standard, NIST, 2016.
- [KPTZ13] Aggelos Kiayias, Stavros Papadopoulos, Nikos Triandopoulos, and Thomas Zacharias. Delegatable pseudorandom functions and applications. In Ahmad-Reza Sadeghi, Virgil D. Gligor, and Moti Yung, editors, *ACM CCS 2013: 20th Conference on Computer and Communications Security*, pages 669–684, Berlin, Germany, November 4–8, 2013. ACM Press.
- [Kra05] Hugo Krawczyk. HMQV: A high-performance secure Diffie-Hellman protocol. In Victor Shoup, editor, *Advances in Cryptology – CRYPTO 2005*, volume 3621 of *Lecture Notes in Computer Science*, pages 546–566, Santa Barbara, CA, USA, August 14–18, 2005. Springer, Heidelberg, Germany.
- [LBdM07] Tri Van Le, Mike Burmester, and Breno de Medeiros. Universally composable and forward-secure RFID authentication and authenticated key exchange. In Feng Bao and Steven Miller, editors, *ASIACCS 07: 2nd ACM Symposium on Information, Computer and Communications Security*, pages 242–252, Singapore, March 20–22, 2007. ACM Press.
- [LSY⁺14] Yong Li, Sven Schäge, Zheng Yang, Florian Kohlar, and Jörg Schwenk. On the security of the pre-shared key ciphersuites of TLS. In Hugo Krawczyk, editor, *PKC 2014: 17th International Conference on Theory and Practice of Public Key Cryptography*, volume 8383 of *Lecture Notes in Computer Science*, pages 669–684, Buenos Aires, Argentina, March 26–28, 2014. Springer, Heidelberg, Germany.
- [SSS⁺20] Shifeng Sun, Amin Sakzad, Ron Steinfeld, Joseph K. Liu, and Dawu Gu. Public-key puncturable encryption: Modular and compact constructions. In Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas, editors, *PKC 2020: 23rd International Conference on Theory and Practice of*

Public Key Cryptography, Part I, volume 12110 of *Lecture Notes in Computer Science*, pages 309–338, Edinburgh, UK, May 4–7, 2020. Springer, Heidelberg, Germany.

- [SW14] Amit Sahai and Brent Waters. How to use indistinguishability obfuscation: deniable encryption, and more. In David B. Shmoys, editor, *46th Annual ACM Symposium on Theory of Computing*, pages 475–484, New York, NY, USA, May 31 – June 3, 2014. ACM Press.

Appendix C

Modular Design of KEM-Based Authenticated Key Exchange

Modular Design of KEM-Based Authenticated Key Exchange

Colin Boyd Bor de Kock Lise Millerjord*

NTNU – Norwegian University of Science and Technology, Trondheim, Norway.

`colin.boyd@ntnu.no`, `bor.dekock@ntnu.no`, `lise.millerjord@ntnu.no`

Abstract

A key encapsulation mechanism (KEM) is a basic building block for key exchange which must be combined with long-term keys in order to achieve authenticated key exchange (AKE). Although several KEM-based AKE protocols have been proposed, KEM-based modular building blocks are not available. We provide a KEM-based authenticator and a KEM-based protocol in the Authenticated Links model (AM), in the terminology of Canetti and Krawczyk (2001). Using these building blocks we achieve a set of generic AKE protocols. By instantiating these with post-quantum secure primitives we are able to propose several new post-quantum secure AKE protocols.

1 Introduction

Authenticated key exchange (AKE) is a fundamental tool for establishing secure communications. An important component in the design of AKE protocols is Diffie–Hellman (DH) key exchange, due to its versatility and potential for providing security properties such as forward secrecy. Today many real-world AKE protocols are based on DH implementations, typically in elliptic curve groups; examples include TLS, IPsec, WireGuard and the generic Noise Framework.

*Millerjord is supported by the Research Council of Norway under Project No. 288545. Author list in alphabetical order; see <https://www.ams.org/profession/leaders/culture/CultureStatement04.pdf>

The looming threat of quantum computers has brought about an increasingly pressing need to find post-quantum secure replacements for DH, which itself is well known to be broken by Shor’s quantum algorithm for finding discrete logarithms [BL17]. In the absence of many promising candidates for a post-quantum secure direct DH replacement, designs for post-quantum AKE have tended to make use of key encapsulation mechanisms (KEM). This approach aligns well with the research literature where many post-quantum candidate KEMs have been proposed and also with the prominent NIST post-quantum cryptography competition [AAC⁺22] which requests primitives of only two types, namely a KEM¹ or a digital signature. Although DH can be framed as a KEM, DH has special properties which prevent KEMs from being used as a drop-in replacement for DH. For example, DH has the property that two parties can generate their DH shares completely independently; this cannot be achieved in general with KEMs, but rather one party must wait for the other party’s input.

Achieving the *authenticated* part of AKE has traditionally been done by applying a digital signature scheme on the messages of a key exchange protocol, but authentication can also be achieved in different ways, which can be advantageous for several reasons: for instance to achieve a speed-up or use less memory, as other works have demonstrated [SSW20].

Although the NIST competition for post-quantum secure cryptography (PQ or PQ-crypto) has in 2022 led to the standardization of several KEMs, only the CRYSTALS-Dilithium signature scheme was standardized along with an open call for more schemes to be proposed [AAC⁺22]. To achieve authentication without depending on this one scheme is a desirable property. One of the main motivations for our work is to be flexible in the use of cryptographic primitives so that as the security of post-quantum KEMs or signatures becomes better understood, and as new primitives are designed, it is easy to swap in and out different ones.

1.1 Modular design of AKE

Over the past decade, several key exchange protocols using post-quantum candidate KEMs have been proposed, both authenticated [DAL⁺17] and

¹Note that in some documents of the NIST competition [AASA⁺20], KEM is used as an abbreviation for *key establishment mechanism* but we stick to the traditional name in cryptographic research.

unauthenticated [DXL12, Pei15, BCD⁺16]. Some of these protocols have been proposed based on specific KEM constructions and the security proofs (where available) relate to specific computational assumptions. These are essential constructions for instantiating protocols using abstract primitives, but when using specific constructions as the basis of security for AKE there is a loss of cryptographic agility. Our goal in this work is to design generic AKE protocols where we can be as flexible as possible with regard to choice of specific KEM instantiations and how they are used.

Our protocol designs are based on the modular approach of Bellare, Canetti and Krawczyk [BCK98, HBN06] (hereafter referred to as BCK98) and Canetti and Krawczyk [CK01] (hereafter referred to as CK01). This approach entails defining protocols which are secure in a world which is ideally authenticated and then compiling these protocols with *authenticators* to achieve protocols secure in a world where adversaries completely control the network. A brief introduction to this modular approach is given in Sec. 2.2.

A significant benefit of the modular approach is the ability to “mix-and-match” different components and to use different concrete instances of the same component within one protocol instantiation. This leads to a plethora of different concrete protocols with varying performance characteristics. For example, in our abstract protocol combining our KEM-based ideal-world protocol with our KEM-based authenticator, choosing from a collection of, say, five different concrete KEMs for each usage, leads to 5^3 different concrete protocols. Not all of these will be of independent interest, but many will have distinctive properties.

We remark that there already exist several protocol designs which are generic in the sense that they can use any specific (secure) instance of different cryptographic primitives such as non-interactive key exchange (NIKE), signatures and/or KEMs [FSXY12, BJS15, JKRS21]. However, such designs do not allow generic mixing of different generic primitives as can be done with the modular approach. For example, the modular approach can be used to replace a digital signature authentication method which a MAC-based method in the case that a pre-shared key is available in a particular application.

While the motivation for this work comes from the goal of achieving post-quantum secure AKE, we note that there is no requirement to

use (only) post-quantum secure components in instantiations. Interim solutions combining post-quantum secure ideal KEMs with conventional established primitives for authentication, will lead to more efficient AKEs secure against active attackers in the short-term and post-quantum adversaries into the future. Furthermore, we can provide *hybrid* secure AKEs which remain secure until both conventional and post-quantum secure components are broken. By using signature combiners or KEM combiners [BHMS17, GHP18, BBF⁺19] we can plug hybrid-secure primitives into any of our generic constructions.

1.2 Contributions

We regard the following as the main contributions of this paper:

1. We develop a new KEM-based authenticator and prove its security as a valid authenticator in the CK01 definition, relying on the established CCA-security definition for KEMs.
2. We frame the well-known method of using an ephemeral KEM as a DH replacement as a protocol in the authenticated-links model (AM) of CK01 and prove its security in that model, relying on the established CPA-security definition for KEMs.
3. We derive efficient and secure generic AKE protocols which can be instantiated with any appropriately secure KEMs and also matched with other primitives such as signatures. Some of these generic protocols are completely new, allowing new instantiations of concrete protocols.

1.3 Related work

In 2017, De Saint Guilhem, Smart and Warinschi [dSW17] presented a generic transformation to convert any two-round forward-secret, but only passively secure, key agreement protocol into a three-round authenticated key agreement protocol. Recognising the value of avoiding signatures for authentication in the post-quantum setting, their transformation makes use of generic CCA-secure public key encryption and a secure MAC. While the approach of De Saint Guilhem et al. has clear parallels with ours, they

rely on encryption rather than the often more efficient notion of KEMs. Moreover, they do not allow mixing of different authentication methods as we do, nor provide KEM-based concrete passively secure protocols. Furthermore, their proofs require a key derivation function modelled as a random oracle. Interestingly, they dismiss the CK01 modular approach stating that it necessarily results in increased number of rounds; below we will explain why this is not the case.

Several recent works show that KEM-based approaches are suitable for replacing signatures in real-world applications. The KEMTLS protocol of Schwabe, Stebila and Wiggers [SSW20] is for instance a complete reworking of the TLS 1.3 handshake without using signatures, showing that this would in theory require only half the bandwidth compared to a classical approach — with additional improvements to be gained if the public keys are exchanged in advance [SSW21]. Some of these theoretical improvements turned out to be less impactful when looking at a real-world implementation [CFS⁺21].

Using KEM as a building block for AKE is also done in some other purpose-specific works: examples of this include Post-Quantum Noise [ADH⁺22], FSXY [FSXY12] and Post-Quantum WireGuard [HNS⁺20]. These are generic in the sense that any suitable KEMs can be used, but they do not allow the flexibility of different authenticators that we obtain. Specifically, they do not provide re-usable and interchangeable components for passive security and for authentication.

There exist many formal security models for AKE, amongst which several are incomparable [Cre11] in the sense that any one model is often neither stronger nor weaker than another. The modular approach that we use [CK01] achieves security in the well-established model known widely as the CK-model. This encompasses fundamental security properties of session key indistinguishability against active attackers who can obtain non-target session keys and adaptively corrupt non-target parties. Forward secrecy is also captured. This model can be adapted [Kra05] if other security properties, such as ephemeral key leakage, are desirable.

1.4 Outline

After presenting necessary definitions, Sec. 2 gives an overview of the CK01 modular design methodology, including definitions such as message trans-

mission (MT) and session key indistinguishability (SK security), which we use extensively throughout this work. This also includes an explanation of how to optimise compiled protocols securely through the application of what we call *compressed* authenticators.

In Sec. 3 we define a new KEM-based MT-authenticator and prove it is a valid authenticator as long as the underlying KEM is CCA-secure, and define a new KEM-based AM protocol, Π which we prove SK-secure.

We then apply compressed authenticators to the authentic messages in Π which results in a 3-message protocol Π' secure in the unauthenticated links model (UM), in Sec. 4. By removing repeated message fields from Π' and combining parallel messages in the same direction, we also obtain efficient generic UM-secure protocols in this section.

Finally, in Sec. 5, we compare the new protocols with existing KEM-based protocols from other works, and examine concrete instantiations of our generic protocols by plugging in concrete KEMs.

2 Background

The main goal of this section is to present the background necessary to understand the modular approach of (Bellare), Canetti and Krawczyk [BCK98, CK01]. This includes our method to optimise, in a rigorous way, protocols obtained through the approach.

2.1 Basic definitions

We use the following standard definitions for KEM, MAC, signatures etc. throughout the paper. These can be found in, for example, the textbook of Katz and Lindell [KL14].

Definition 1. *A key encapsulation mechanism (KEM) is a tuple of PPT algorithms (Gen, Encap, Decap) such that:*

1. *The key generation algorithm Gen takes the security parameter 1^n and outputs a public-/private-key pair (sk, pk) : $(sk, pk) \leftarrow \text{Gen}(1^n)$.*
2. *The encapsulation algorithm Encap takes as input a public key pk . It outputs a ciphertext c and a key $k \in \{0, 1\}^{l(n)}$: $(c, k) \leftarrow \text{Encap}(pk)$.*

3. The deterministic decapsulation algorithm Decap takes as input a private key sk and a ciphertext c , and outputs a key k or the special symbol \perp denoting failure: $k \leftarrow \text{Decap}(sk, c)$.

We require correctness from the KEM: If $(sk, pk) \leftarrow \text{Gen}(1^n)$ and $(c, k) \leftarrow \text{Encap}_{pk}(1^n)$, and $k' \leftarrow \text{Decap}_{sk}(c)$, then $k' = k$ except with negligible probability.

Furthermore we show the CPA (resp. CCA) indistinguishability experiment(s) for KEMs.

Definition 2. The CPA resp. CCA indistinguishability experiment proceeds as follows:

1. The key generation algorithm is run: $(pk, sk) \leftarrow \text{Gen}(1^n)$.
2. The encapsulation algorithm is run: $(c, k) \leftarrow \text{Encap}(pk)$, with $k \in \{0, 1\}^n$.
3. A uniform bit $b \in \{0, 1\}$ is chosen. If $b = 0$, set $k' = k$. Otherwise, if $b = 1$, choose a uniform $k' \in \{0, 1\}^n$.
4. The experiments outputs (pk, c, k') to \mathcal{A} .

\mathcal{A} is also given access to a decapsulation oracle, $\text{Decap}_{sk}(\cdot)$, but cannot query the decapsulation oracle on the ciphertext c .

5. \mathcal{A} outputs a bit b' . If $b' = b$, \mathcal{A} wins and the experiment outputs 1. Otherwise, \mathcal{A} loses and the experiment outputs 0.

The advantage of the adversary \mathcal{A} in the CPA CCA experiment is defined to be:

$$\text{Adv}_{\text{KEM}}^{\text{CPA} \text{ CCA}}(\mathcal{A}) = 2 \cdot |\Pr [b' = b] - 1/2|.$$

Definition 3. A message authentication code or MAC consists of three probabilistic polynomial-time algorithms $(\text{Gen}, \text{Mac}, \text{MacVer})$ such that:

1. The key-generation algorithm Gen takes as input the security parameter 1^n and outputs a key k with $|k| \geq n$.
2. The tag-generation algorithm Mac takes as input a key k and a message $m \in \{0, 1\}^*$, and outputs a tag t . Since this algorithm may be randomized, we write this as $t \leftarrow \text{Mac}_k(m)$.
3. The deterministic verification algorithm MacVer takes as input a key k , a message m and a tag t . It outputs a bit b , with $b = 1$ meaning valid and $b = 0$ meaning invalid. We write this as $b := \text{MacVer}_k(m, t)$.

It is required that for every n , every key k and output by $\text{Gen}(1^n)$ and every $m \in \{0, 1\}$, it holds that $\text{MacVer}_k(m, \text{Mac}(m)) = 1$.

Definition 4. *The existential unforgeability under chosen message attacks (EUF-CMA) experiment for $\text{MAC}(\text{Gen}, \text{Mac}, \text{MacVer})$ proceeds as follows:*

1. A key is generated: $k \leftarrow \text{Gen}(1^n)$.
2. The adversary \mathcal{A} gets oracle access to $\text{Mac}_k(\cdot)$. Let Q be the set of all queries \mathcal{A} made to the oracle. The adversary eventually outputs (m, t) .
3. \mathcal{A} wins if and only if
 - (a) $\text{MacVer}_k(m, t) = 1$, and
 - (b) $m \notin Q$.

In that case the experiment outputs 1. Otherwise, the experiment outputs 0.

The advantage of the adversary \mathcal{A} in the EUF-CMA experiment is defined to be:

$$\text{Adv}_{\text{MAC}}^{\text{EUF-CMA}}(\mathcal{A}) = \Pr \left[\text{G}_{\text{MAC}}^{\text{EUF-CMA}}(\mathcal{A}) = 1 \right].$$

Definition 5. *A (digital) signature scheme is a tuple of three PTT-algorithms $(\text{Gen}, \text{Sign}, \text{SigVer})$ such that:*

1. The key generation algorithm Gen takes the security parameter 1^n and outputs a public-/private-key pair (sk, pk) : $(sk, pk) \leftarrow \text{Gen}(1^n)$.

2. The signing algorithm Sign takes as input a private key sk and a message m from some message space (that may depend on pk). It outputs the signature σ and we write this as $\sigma \leftarrow \text{Sign}_{sk}(m)$.
3. The deterministic verification algorithm SigVer takes as input a public key pk , a message m and a signature σ . It outputs a bit b , with $b = 1$ meaning valid and $b = 0$ meaning invalid. We write this as $b := \text{SigVer}_{pk}(m, \sigma)$.

We require correctness from the scheme: If $(sk, pk) \leftarrow \text{Gen}(1^n)$ then, except with negligible probability, $\text{SigVer}_{pk}(m, \text{Sign}_{sk}(m)) = 1$.

For brevity we often denote the key generation algorithms from the various primitives as $\text{Gen}()$, omitting the security parameter.

2.2 Canetti–Krawczyk modular design

The *modular approach*, arising originally in a 1998 paper of Bellare, Canetti and Krawczyk [BCK98], is to first define protocols secure against a limited adversary which can then be promoted to protocols secure against a realistic adversary using a generic compiler. In the *authenticated links model* (AM) the adversary is not permitted to fabricate messages, but can otherwise control the network and deliver messages out of order or to different parties from those intended. Compilers, or *authenticators*, can be applied to messages in an AM protocol to obtain protocols in the *unauthenticated links model* (UM) where the adversary can alter or fabricate messages limited only by the available computational power.

In both the UM and AM, adversaries control the execution of protocols by initiating parties and then invoking parties with available queries, including with message inputs. (In Sec. 2.4 we describe the available adversarial queries.) Parties respond to input messages by following the protocol definition and to other queries as defined by the query. Each party computes *local output* which is available to the adversary. The local output includes protocol decisions, such as whether a message is accepted (see Sec. 2.4 for details).

Bellare et al. [BCK98] provide a theorem showing that a secure protocol, Π_{AM} , in the AM maps to a secure protocol in the UM, Π_{UM} , if the mapping is defined by a valid authenticator. An authenticator is valid if

an observer, or distinguisher, is unable to distinguish between the world where an adversary \mathcal{A} is interacting with the Π_{AM} and the world where an adversary \mathcal{U} is interacting with the protocol Π_{UM} . This is captured in the notion of protocol emulation in Definition 7.

Definition 6. *The AM-UM distinguishing experiment, $G_{\Pi_{\text{AM}}-\Pi_{\text{UM}}}^{\text{AM-UM-dist}}(\mathcal{D})$ proceeds as follows:*

1. *A uniform bit $b \in \{0, 1\}$ is chosen. If $b = 0$, \mathcal{D} will interact with \mathcal{A} and the AM protocol Π_{AM} . Otherwise, if $b = 1$, \mathcal{D} will interact with \mathcal{U} and the UM protocol Π_{UM} .*
2. *To conclude the experiment, \mathcal{D} will halt and output b' .*
3. *The experiment will output 1 if and only if $b = b'$.*

We define the advantage of the distinguisher \mathcal{D} to be

$$\text{Adv}_{\Pi_{\text{AM}}-\Pi_{\text{UM}}}^{\text{AM-UM-dist}}(\mathcal{D}) = 2 \cdot \left| \Pr \left[G_{\Pi_{\text{AM}}-\Pi_{\text{UM}}}^{\text{AM-UM-dist}}(\mathcal{D}) = 1 \right] - \frac{1}{2} \right|.$$

Definition 7. *Let Π_{UM} and Π_{AM} be protocols in the UM and AM models respectively. We say that Π_{UM} ϵ -emulates Π_{AM} in unauthenticated networks if for any UM-adversary \mathcal{U} interacting with Π_{UM} , there exists an AM-adversary \mathcal{A} interacting with Π_{AM} such that for any distinguisher \mathcal{D} playing the AM-UM distinguishing game,*

$$\text{Adv}_{\Pi_{\text{AM}}-\Pi_{\text{UM}}}^{\text{AM-UM-dist}}(\mathcal{D}) \leq \epsilon.$$

An authenticator is a specific type of *protocol compiler* transforming one protocol into another. The modularity of the approach relies on the observation that an authenticator will actually preserve protocol security as we will see in Sec. 2.4.

Definition 8 ([BCK98]). *An authenticator is a compiler, C , that takes an AM protocol Π_{AM} as input and outputs a UM protocol Π_{UM} , such that Π_{UM} emulates Π_{AM} .*

2.3 MT-authenticators

Defining an authenticator for any protocol, regardless of the number of messages, seems at first a difficult problem. To deal with this, BCK98 [BCK98] define a simpler notion of an MT-authenticator, designed to authenticate a single arbitrary message. They also showed that repeated use of a valid MT-authenticator is a valid authenticator, so that protocol messages can be treated separately.

A bit more formally we define MT as a *message transmission* protocol in authenticated networks that works as follows: when P_i is activated with (P_j, m) , party P_i sends the message (P_i, P_j, m) to party P_j and outputs “ P_i sent m to P_j ”. Upon receiving (P_i, P_j, m) , P_j outputs “ P_j received m from P_i ”. Note that the quoted outputs are local outputs of the parties and are critical in proving proper emulation; however, when we later show compiled protocols we omit mention of these local outputs.

An MT-authenticator, λ , is a protocol that emulates MT in unauthenticated networks. Given a sequence of MT-authenticators, $\Lambda = (\lambda_1, \lambda_2, \dots, \lambda_t)$, the derived protocol compiler, C_Λ , uses the next MT-authenticator to authenticate the next message. More precisely, given a protocol Π in the AM with t messages, m_1, m_2, \dots, m_t the protocol $\Pi' = C_\Lambda(\Pi)$ in the UM is defined as follows. For each message, m_k , sent in Π , λ_k is run to send the same message from the same initiator to the same recipient. Whenever a party, P_j , outputs “ P_j received m from P_i ” in λ_k , then Π is activated at P_j with message m_k from P_i . If Λ is a sequence of t MT-authenticators then C_Λ is an authenticator. We restate this in Thm. 1 and give a sketch of the proof, which is given in full in [BCK98].

Theorem 1 ([BCK98]). *Let $\Lambda = (\lambda_1, \lambda_2, \dots, \lambda_t)$ be a sequence of t MT-authenticators so that each λ_k ϵ -emulates MT. Then the compiler, C_Λ , will be an authenticator such that for any protocol Π in the AM, $C_\Lambda(\Pi)$ $(t \cdot \epsilon)$ -emulates Π in the UM.*

Proof. Let Π be an AM protocol. Let \mathcal{U} be a UM-adversary interacting with $C_\Lambda(\Pi)$. \mathcal{A} runs \mathcal{U} on a simulated interaction. Action requests from \mathcal{U} to parties in the UM can be mimicked by \mathcal{A} in the AM and \mathcal{A} relays its results back to \mathcal{U} . The only problem with the simulation could occur in the case that \mathcal{U} specifies that a message is received by some party P_j from some party P_i in the UM, but that message is not in the set of messages

waiting for delivery in the AM. But this can happen with probability bounded by ϵ . Such an event could occur for any of the t messages and so the probability that the simulation is correct is at least $(1 - \epsilon)^t \geq 1 - t \cdot \epsilon$. Finally, any observer will be able to distinguish between the run of Π in the AM and $C_\Lambda(\Pi)$ in the UM with advantage at most $\epsilon' = t \cdot \epsilon$. \square

Note that although we have assumed that each MT-authenticator has the same security level ϵ , the theorem is still true if the MT-authenticators have different security levels $\epsilon_1, \epsilon_2, \dots, \epsilon_t$ and we take $\epsilon = \max_k(\epsilon_k)$. In the cases we are interested in, we will always have $t = 2$.

An example of an MT-authenticator is the encryption-based authenticator [BCK98] shown in Fig. 1, where N_B denotes a nonce and (e_A, d_A) an encryption-decryption keypair. This is a valid MT-authenticator as long as the public key encryption used is CCA-secure and the MAC is secure. The protocol can be optimized in various ways, as we will show later.

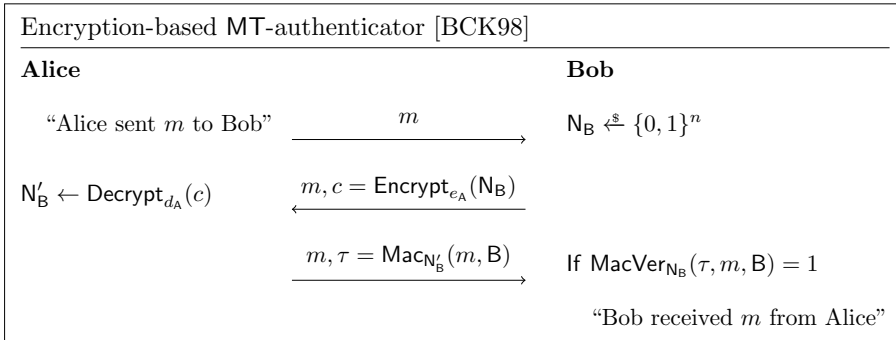


Figure 1: Bob authenticates message m from Alice.

There exist several other MT-authenticators defined in the literature including signature-based [BCK98], MAC-based [CK01] and password-based [HBN06]. We show, in compressed form, the signature-based MT-authenticator later (Fig. 9). The modular approach allows combination of any MT-authenticator together with any AM-protocol, resulting in automatically secure UM protocols. Therefore adding any new building block, either an MT-authenticator or an AM-protocol, results in several new protocols of potential interest.

2.4 SK-security

SK-security is the AKE security notion of CK01 [CK01], capturing session key indistinguishability and correctness of the protocol. To define this notion we need to state the capabilities of the adversary and the indistinguishability experiment. Each protocol run at a party A is associated with a session identifier s . In the AM the value s is an input at the start of a run to the initiator party. Later we will see that session identifiers can be replaced by protocol messages as long as parties can verify that no incomplete sessions between the same parties have the same session identifier. The state of a session consists of the following information:

- status – whether or not the session is complete, aborted, or still in progress;
- any ephemeral key material needed to complete the protocol;
- the session key, sk , if the protocol is completed and has not expired.

The global state of a party may include long-term authentication keys pk_A, sk_A . As in most AKE models, we do not explicitly model distribution of long-term keys. Furthermore, we assume that long-term public keys are immediately available to any party that needs them. This may be too strong an assumption in some real-world protocols, such as TLS, and we remark further on this issue when we examine concrete protocols in Sec. 5.

Definition 9 (Matching sessions [CK01]). *The two sessions $(A, B, s, role)$ and $(A', B', s', role')$ are matching if $A = B', B = A'$ and $s = s'$.*

The adversary may issue the following queries, subject to certain restrictions we will see later.

- $\text{NewSession}(A, B, s, r)$: the adversary issues the NewSession query to party A , specifying its intended partner B , the session identifier² s , and the role r (initiator or responder) of A in the session. A will follow the protocol definition and may return an output message intended for B .

²We remark that instantiation of session identifiers differs between the models. In UM, s can be blank as the session identifier need not be determined by the adversary.

- $\text{Send}(A, B, m)$: represents activation of A by an incoming message m (possibly including a session identifier) from party B . A will follow the protocol and may reject, accept, or return an output message intended for B .
- $\text{Corrupt}(A)$: the adversary learns the whole state of A including any long-term keys. The corruption event is recorded in the local output of A . Subsequently A can never be activated but the adversary can take the role of A in the protocol.
- $\text{RevealKey}(A, B, s)$: the adversary learns the session key accepted in the session s by A with partner B , if it exists. The reveal event is recorded in the local output of A .
- $\text{RevealState}(A, B, s)$: the adversary learns the state information associated with session s at A , such as ephemeral keys. The reveal state event is recorded in the local output of A .
- $\text{Expire}(A, B, s)$: if there is a completed session s at A with B then any session key associated with that session is deleted from the memory of A . The Expire event is recorded in the local output of A .
- $\text{Test}(A, B, s)$: this query can be asked only once and can only be made to a completed session s at A with partner B . Furthermore there cannot have been any of the following queries made: $\text{RevealKey}(A, B, s)$ or $\text{RevealState}(A, B, s)$ or $\text{Corrupt}(A)$ or $\text{Corrupt}(B)$. If the bit b specified by the challenger is $b = 1$ then the session key is returned. Otherwise $b = 0$ and a random key from the keyspace are returned.

Now we are in a position to define the SK-security experiment.

Definition 10. *The key indistinguishability experiment, $G_{\Pi}^{\text{Key-Ind}}(\mathcal{A})$ is defined as follows:*

1. *The challenger chooses a random bit b needed to define the Test query response.*
2. *The challenger initialises n parties and any long-term keys.*
3. *\mathcal{A} may issue queries as defined above.*

4. Eventually \mathcal{A} halts and outputs a bit b' to indicate its guess for b , based on the response to the `Test` query. The experiment outputs 1 if and only if $b' = b$.

Definition 11. A key exchange protocol Π is ϵ – SK-secure if the following holds for any adversary \mathcal{A} :

1. two honest parties (i.e. uncorrupted parties who faithfully execute the protocol instructions) completing matching sessions of the protocol Π will output the same key, except with negligible probability, and
2. the advantage of the adversary \mathcal{U} in the key indistinguishability experiment is:

$$\text{Adv}_{\Pi}^{\text{Key-Ind}}(\mathcal{A}) = 2 \cdot |\Pr [b' = b] - 1/2| \leq \epsilon.$$

The final step needed to bring the modular approach together is to show that emulation preserves SK-security. This was proven in CK01 and we restate it as Thm. 2 including concrete bounds. Note that using emulation of an ideal key exchange process as a definition of security, the original idea of BCK98, results in too strong a definition to allow some well-known protocols to be proven secure [CK01, Appendix A].

Theorem 2 ([CK01]). *Let Π be an ϵ – SK-secure protocol in the AM with t messages. Let C_{Λ} be the compiler based on MT-authenticators $\lambda_1, \lambda_2, \dots, \lambda_t$ such that for any protocol Π in the AM, $C_{\Lambda}(\Pi)$ α -emulates Π in the UM. Then protocol $\Pi' = C_{\Lambda}(\Pi)$ is an ϵ' – SK-secure protocol in the UM with $\epsilon' = \epsilon + \alpha$.*

Proof. Assume to the contrary that there exists a UM adversary \mathcal{U} that has advantage ϵ' in the UM. Using \mathcal{U} , we build an AM adversary, \mathcal{A} , playing the game of Defn. 10. When \mathcal{A} receives its setup information consisting of system parameters and public keys from its challenger, \mathcal{A} sends the same information to \mathcal{U} . Then \mathcal{A} invokes \mathcal{U} and mimics its behaviour in the AM, using its challenger to respond to the action requests when any party is exposed.

When \mathcal{U} sends a message to a party P_j from a party P_i in the UM, \mathcal{A} sends the same message between the same parties in the AM. The

emulation will be perfect unless \mathcal{U} successfully sends a message m to some party P_j from P_i but m was never sent by P_i . In this case we will say that \mathcal{U} made a forgery and we let forge be the event that a forgery happens at any time during the run of \mathcal{U} .

If forge occurs then \mathcal{A} will abort the simulation and return a random bit to its challenger. Note that this also defines a distinguisher \mathcal{D} which will always win in the case that forge occurs. If forge does not occur then at some point \mathcal{U} will ask its Test query for a session s . \mathcal{A} then announces session s for its own Test query in the AM, receives a real or random key, and returns it to \mathcal{U} . Eventually \mathcal{U} will halt and output its bit which \mathcal{A} copies as its response. In this case, \mathcal{A} wins whenever \mathcal{U} wins.

$$\begin{aligned} \Pr[\mathcal{A} \text{ wins}] &= \Pr[\mathcal{A} \text{ wins}|\text{forge}] \cdot \Pr[\text{forge}] + \Pr[\mathcal{A} \text{ wins}|\neg\text{forge}] \cdot \Pr[\neg\text{forge}] \\ &= 1/2 \cdot \Pr[\text{forge}] + \Pr[\mathcal{B} \text{ wins}] \cdot (1 - \Pr[\text{forge}]) \\ &\geq \Pr[\mathcal{B} \text{ wins}] - 1/2 \cdot \Pr[\text{forge}] \end{aligned}$$

We also implicitly defined a distinguisher, \mathcal{D} , which wins when forge occurs or wins with probability at least $1/2$ when forge does not occur: $\Pr[\mathcal{D} \text{ wins}] \geq \Pr[\text{forge}]/2 + 1/2$. Putting this together we get:

$$\text{Adv}_{\Pi'}^{\text{Key-Ind}}(\mathcal{U}) \leq \text{Adv}_{\Pi}^{\text{Key-Ind}}(\mathcal{A}) + \text{Adv}_{\Pi-\Pi'}^{\text{AM-UM-dist}}(\mathcal{D}).$$

□

2.5 Optimising the UM protocol

Simple application of an MT-authenticator to each message of an SK-secure AM protocol results in an SK-secure UM protocol as proven in Thm. 2. However, such a protocol is far from optimal. The most obvious drawback is that a two-message protocol, such as Diffie–Hellman, compiles to a six-message protocol. The obvious way to optimise such a protocol is to “piggyback” messages going in the same direction. The resulting protocol may be secure, but formally this process may break the security proof because it may alter the order of the local output of the parties, allowing trivial distinguishability outputs of the AM protocol from the outputs of the compiled UM protocol [HBN06].

Because of such issues, the modular approach of CK01 has been criticised [dSW17] for not achieving efficient protocols. There is some truth

in such criticisms — for example, when using signature- or encryption-based authenticators it is not possible to achieve secure 2-message AKE protocols which are often seen in the literature. Fortunately, rigorous optimisations are not difficult to achieve, typically resulting in 3-message protocols as efficient as real-world protocols. Indeed, 3-message AKE protocols are necessary in any case to achieve desirable security properties such as mutual entity authentication or key confirmation.

Hitchcock et al. [HBN06] designed a general technique for altering message ordering in a security-preserving way. This involved defining an intermediate model between the AM and UM, which they call the *hybrid model*. Rather than use this more comprehensive approach, here we apply simple techniques to allow optimisation of the number of messages and re-use message components as session identifiers. Consequently, the drawbacks of practical application of authenticators are removed resulting in generic protocols as efficient as standalone protocols.

Compressed authenticators. The first step is to compress the authenticator to remove redundant elements. Notice that use of the authenticator in Fig. 1 expands each message m from the AM into three messages in the UM. However, sending m in all three messages is not actually needed (to achieve security), so we can simplify the encryption-based authenticator into a compressed version shown in Fig. 2. It is not hard to see [BCK98, HBN06] that removal of the repeated m fields does not affect the security of the MT-authenticator. Depending on the application scenario, the version in Fig. 1 may remain appropriate. The version in Fig. 2 is useful in a situation where Bob knows that some message, as yet unknown, will be authentically received from Alice; this case typically occurs in AKE protocols. Later we will see that to apply optimisation it is important that the first message in Fig. 2 is independent of the message to be authenticated, so that it can be generated and sent early in the protocol.

Session identifiers. In the original formulation of CK01, session identifiers are sent in each protocol run in the AM. These must be unique for each active protocol run between the same parties, but it is not defined how they should be obtained in practice. Although the only property required of session identifiers is uniqueness, a natural way of obtaining them is to

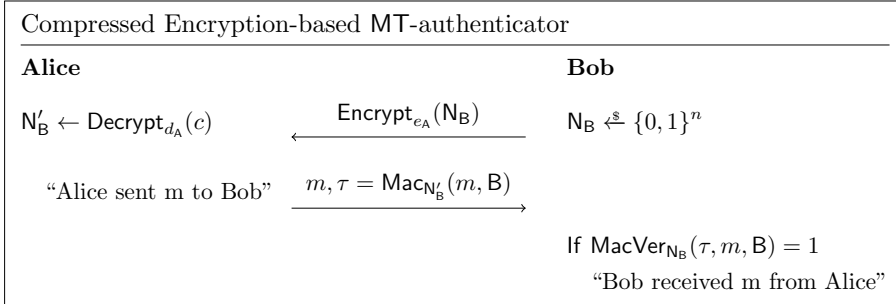


Figure 2: Compressed version of MT-authenticator in Fig. 1.

use random values chosen by each party; in that case the probability that session identifiers are not unique is negligible. In practice it may not be a burden for each party to ensure that there are no other incomplete sessions with the same identifier so that uniqueness is unconditionally guaranteed.

We assume that higher communication layers will provide a mechanism to ensure that messages get delivered to the correct session. They can also be explicitly added to the protocol messages if desired.

3 KEM-based building blocks

This section defines and proves security for the basic KEM-based MT-authenticator and AM protocol, which will be brought together in Sec. 4 as components in defining generic efficient KEM-based AKE.

3.1 KEM-based MT-authenticator

Fig. 3 illustrates our KEM-based MT-authenticator. The construction is closely related to the encryption-based authenticator of BCK98.

Next we give a theorem that λ_{KEM} is secure, meaning that it emulates MT in unauthenticated networks, as long as the KEM used achieves CCA security.

Theorem 3. *The KEM-based MT-authenticator, λ_{KEM} , in Fig. 3, when instantiated with a CCA-secure KEM and a secure MAC scheme, ϵ -emulates protocol MT in unauthenticated networks such that $\epsilon \leq l \cdot (\text{Adv}_{\text{KEM}}^{\text{CCA}}(\mathcal{D}) +$*

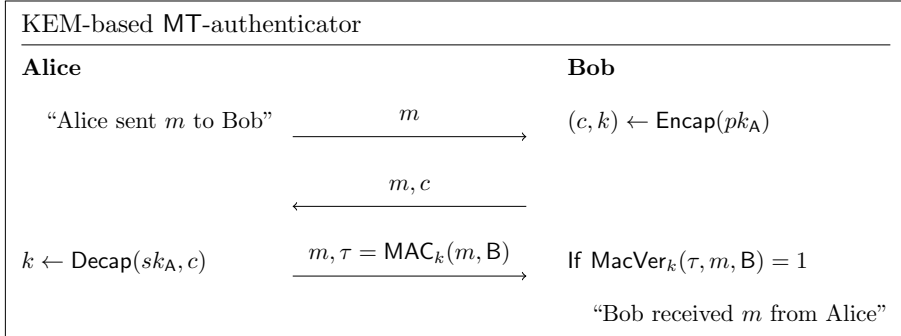


Figure 3: KEM-based MT-authenticator, λ_{KEM} : Bob authenticates m from Alice.

$\text{Adv}_{\text{MAC}}^{\text{EUF-CMA}}(\mathcal{F})$ where $l = n_P^2 \times n_M$, n_P is the number of parties that run the protocol and n_M is the maximum number of challenge messages that can be sent by any party.

The proof of Thm. 3 follows closely the proof of Bellare et al. [BCK98, Proposition 5]. We have included some extra details which we hope may be useful to the reader.

Proof. Let \mathcal{U} be a UM-adversary that interacts with λ_{KEM} . We construct an AM-adversary \mathcal{A} such that for any distinguisher \mathcal{D} observing, $\text{View}_{\mathcal{A}}^{\text{MT}}(\mathcal{D}) \stackrel{s}{=} \text{View}_{\mathcal{U}}^{\lambda_{\text{KEM}}}(\mathcal{D})$.

First note that \mathcal{A} can simply copy all of the outputs of \mathcal{U} unless the following event **bad** happens.

bad: \mathcal{U} outputs “Q received m from P” for some parties P and Q, but there was no previous output “P sent m to Q”.

If we assume that **bad** happens with probability ϵ then for any distinguisher \mathcal{D} we must have $\text{Adv}(\mathcal{D}) \leq \epsilon$. Thus from now on we focus on bounding ϵ . We construct an experiment involving multiple adversaries as follows and illustrated in Fig. 4. We start by describing the purpose and connections between the different entities. Afterwards we give details of the security game.

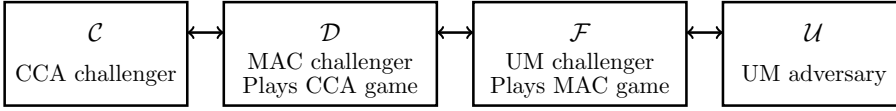


Figure 4: Overview of the experiment in the proof of Thm. 3.

- \mathcal{U} is the UM adversary running λ_{KEM} in such a way that event **bad** will occur with probability ϵ . When **bad** happens \mathcal{U} will produce a valid MAC for some session without knowing the decapsulation key of the party P which should be producing the MAC.
- \mathcal{F} is going to interact with \mathcal{U} in order to simulate all of the parties in the run of \mathcal{U} . We will see that \mathcal{F} can make the simulation perfect in case **bad** happens only in a chosen target session. \mathcal{F} also is playing a forging game against the MAC scheme. Since \mathcal{F} also receives a ciphertext from its challenger it may not be playing the usual EUF-CMA game, but in cases where that ciphertext is independent of the MAC key the game \mathcal{F} plays is equivalent to the EUF-CMA game.
- \mathcal{D} is acting as the challenger for \mathcal{F} in the EUF-CMA game, providing a MAC oracle. \mathcal{D} is also playing the IND-CCA game from Def. 2 against the KEM. \mathcal{D} will provide its challenge encapsulation to \mathcal{F} .
- \mathcal{C} is the IND-CCA challenger, providing the challenge encapsulation and the decapsulation oracle.

The experiment proceeds as follows.

1. \mathcal{D} starts the CCA indistinguishability experiment with \mathcal{C} . The first 4 steps in Def. 2 are run between \mathcal{D} and \mathcal{C} so that \mathcal{D} gets (pk, c^*, k') where k' is either encapsulated in c^* ($b = 0$) or is a key chosen randomly in the range of Encap ($b = 1$).
2. \mathcal{D} starts \mathcal{F} by passing \mathcal{F} the encapsulation c^* . From now on \mathcal{D} will answer any MAC oracle queries by using the key k' to compute the correct MAC tag and returning it to \mathcal{F} . The goal of \mathcal{F} is to produce a forgery (m^*, t) which is a valid MAC (for key k') which was never returned by \mathcal{D} following a MAC oracle query from \mathcal{F} .

\mathcal{F} is also provided with a decapsulation oracle for the key encapsulated in c^* . \mathcal{F} will need this oracle in order to properly simulate the parties for \mathcal{U} . This oracle is available to \mathcal{D} through the game that \mathcal{D} is playing with \mathcal{C} so \mathcal{D} can also simulate the oracle for \mathcal{F} .

Note that there are two cases.

- (a) The encapsulation c contains key k' , the key for the MAC that \mathcal{F} is attempting to forge. In this case \mathcal{F} has received some additional help in finding a forgery by knowing c and having the decapsulation oracle for k' . Thus \mathcal{F} is an *aided oracle* [BCK98] and even though a correct forgery is produced and accepted by \mathcal{D} , this does not win the MAC security game in the EUF-CMA sense.
 - (b) The encapsulation c contains a random key, independent of k' . In this case \mathcal{F} does not receive any useful help in finding a forgery for k' . (Note that \mathcal{F} could easily simulate the “help” itself in this case.) Thus a forgery wins the EUF-CMA MAC game in Def. 4.
3. \mathcal{F} starts \mathcal{U} and chooses two random parties P^* and P and a randomly selected session s at P . This is the session where \mathcal{F} guesses that event **bad** will happen. Note that the probability that **bad** happens in session s is at least ϵ/l where $l = n_P^2 \times n_M$ where n_P is the number of parties and n_M is the maximum number of challenge messages that can be sent by party P .

\mathcal{F} chooses public key pairs for all parties except P^* and assigns pk to party P^* . \mathcal{F} can now simulate all parties in the run of \mathcal{U} as long as \mathcal{U} does not choose to corrupt P^* or asks for a MAC for the message used in the target session s with partner P ; however if either of these two things happen then \mathcal{F} will abort and \mathcal{D} will return a random bit as its guess for b' .

When simulating a session involving party P^* , but different from s , \mathcal{F} has to make use of the decapsulation oracle from \mathcal{D} since it does not have the private decapsulation key of P^* . Then \mathcal{F} can obtain the correct MAC key and provide the correct third message for P . Also if \mathcal{F} is asked by \mathcal{U} to simulate a session where P^* receives m, c^*

from party Q with $m \neq m^*$ or $Q \neq P$, then \mathcal{F} asks \mathcal{D} for the MAC tag in order to continue the simulation.

4. If **bad** does not happen in session s then \mathcal{F} halts with no output and then \mathcal{D} returns a random bit at its guess for b' . With probability at least ϵ/l , event **bad** does happen in session s . Since party P^* was never activated in this session, \mathcal{U} will directly activate party P with message m^* to send the second (challenge) message to party P^* . Then \mathcal{F} responds with m^*, c^* . When **bad** happens \mathcal{U} will send a MAC value t for message m^* to party P which is then passed to \mathcal{F} (who is simulating P). \mathcal{F} then sends (m^*, t) to \mathcal{D} as its forgery attempt. Note that \mathcal{U} cannot ask for the MAC tag from P^* directly since P^* is not involved in the run when **bad** happens, so the forgery must originate from \mathcal{U} .
5. \mathcal{U} receives the putative forgery (m^*, t) from \mathcal{F} and checks its correctness using key k' . If $\text{MacVer}_{k'}(m^*, t) = 1$ then \mathcal{U} decides that encapsulation c contains key k' and returns its guess $b' = 0$ to \mathcal{C} . Otherwise \mathcal{U} returns $b' = 1$.

Analysis We consider two cases, each of which occurs with probability $1/2$.

- If $b = 0$ then \mathcal{F} will always return a correct MAC tag for key k' when **bad** happens. In this case \mathcal{D} will return $b' = 0$ and so will win the CCA game with \mathcal{C} .
- If $b = 1$ then we have two possibilities.
 1. If \mathcal{F} returns an incorrect MAC tag for key k' when **bad** happens then \mathcal{D} will return $b' = 1$ and so will again win its CCA game with \mathcal{C} .
 2. If \mathcal{F} returns a correct MAC tag for key k' then \mathcal{D} will return $b' = 0$ and so will lose the CCA game with \mathcal{C} . However, \mathcal{F} got no help in its forgery game and so can win the EUF-CMA game against the MAC.

First note that

$$\Pr[\mathcal{F} \text{ wins}] = \Pr[\mathcal{F} \text{ wins}|\text{bad}] \cdot \Pr[\text{bad}] = \Pr[\mathcal{F} \text{ wins}|\text{bad}] \cdot \frac{\epsilon}{l}$$

since $\Pr[\mathcal{F} \text{ wins}|\overline{\text{bad}}] = 0$. Also

$$\Pr[\mathcal{D} \text{ wins}|\text{bad}] = \frac{1}{2} + \frac{1}{2} \cdot (1 - \Pr[\mathcal{F} \text{ wins}|\text{bad}]) = 1 - \Pr[\mathcal{F} \text{ wins}] \cdot \frac{l}{2\epsilon}.$$

Putting these together we get:

$$\Pr[\mathcal{D} \text{ wins}] \geq \left(1 - \frac{l}{2\epsilon} \Pr[\mathcal{F} \text{ wins}]\right) \cdot \frac{\epsilon}{l} + \frac{1}{2} \left(1 - \frac{\epsilon}{l}\right) = \frac{1}{2} + \frac{\epsilon}{2l} - \frac{1}{2} \Pr[\mathcal{F} \text{ wins}].$$

Finally we re-arrange to obtain: $2 \cdot (\Pr[\mathcal{D} \text{ wins}] - \frac{1}{2}) + \Pr[\mathcal{F} \text{ wins}] \geq \frac{\epsilon}{l}$, and then by Def. 2 and 4, $\epsilon \leq l \cdot (\text{Adv}_{\text{KEM}}^{\text{CCA}}(\mathcal{D}) + \text{Adv}_{\text{MAC}}^{\text{EUF-CMA}}(\mathcal{F}))$. \square

Now that λ_{KEM} is proven to be an MT-authenticator we can invoke Thm. 2 to conclude that λ_{KEM} can be used to authenticate messages in an SK-secure AM protocol and results in a SK-secure UM protocol. In order to optimise the resulting protocol we will want to use a compressed version of the authenticator (see Sec. 2.5) as shown in Fig. 5.

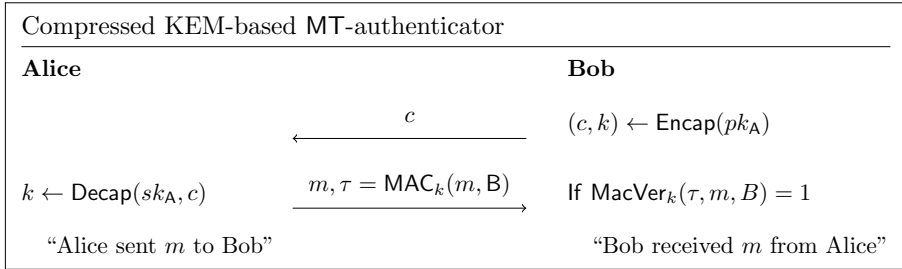


Figure 5: λ_{KEM} , the compressed KEM-based MT-authenticator.

The security proof for the compressed authenticator is identical to the proof for the full authenticator since the only difference is the deletion of plaintext messages in the UM which are ignored in the security proof.

Corollary 1. *Theorem 3 still holds if the authenticator in Fig. 3 is replaced by the compressed KEM-based MT-authenticator, λ_{KEM} , in Fig. 5.*

3.2 KEM-based AM protocol

In Fig. 6 we present a KEM-based protocol Π that is SK-secure in the AM. The protocol is a generalisation of the basic Diffie–Hellman AM protocol of CK01 [CK01]. We assume that a setup with parameters for the KEM are known already to all parties. The initiator A will be invoked by the $\text{NewSession}(A, B, s, r)$ query and responds with a new ephemeral KEM public key pk_e . Upon receipt of (pk_e, s) the responder encapsulates a new session key sk in c , and returns it to party A.

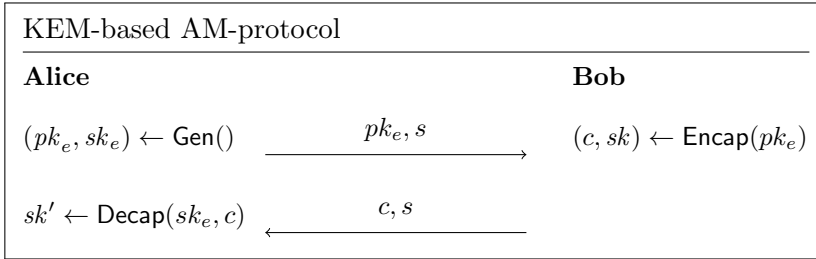


Figure 6: KEM-based protocol with any CPA-secure KEM (see Def. 2).

Theorem 4. *Let \mathcal{A} be an adversary against the SK-security of protocol Π shown in Fig. 6. Let \mathcal{A} interact with at most q sessions of Π for each pair of parties. Let n be the maximum number of parties involved in the protocol run. Then the advantage of \mathcal{A} can be bounded by: $\text{Adv}_{\Pi}^{\text{SK}}(\mathcal{A}) \leq n^2 q \cdot \text{Adv}_{\text{KEM}}^{\text{CPA}}(\mathcal{B})$.*

Proof. The definition of SK-security has two requirements. The first requirement is achieved by the correctness of the KEM. For the second requirement we use adversary \mathcal{A} against the sk -security of the protocol Π to construct an adversary \mathcal{B} against the CPA-security of the KEM. Adversary \mathcal{B} interacts with its challenger \mathcal{C} in the $\mathbf{G}_{\text{KEM}}^{\text{CPA}}(\cdot)$ experiment.

First \mathcal{B} is given (pk^*, c^*, k^*) by \mathcal{C} and should output $b \in \{0, 1\}$ guessing if k^* is the real key ($b = 1$) or a random key ($b = 0$). Next \mathcal{A} chooses random parties A^* and B^* from the set of all (n possible) parties and for these chooses a *target session* identifier s^* by choosing a unique value in $[1..q]$, where q is the maximum number of sessions at any party. Then \mathcal{B} invokes \mathcal{A} and simulates all responses as follows.

- **NewSession**(A, B, r, s): if r is initiator then \mathcal{B} checks whether $A = A^*$, $B = B^*$, and $s = s^*$ and if so \mathcal{B} returns (pk^*, s^*) to \mathcal{A} . Otherwise \mathcal{B} runs $(pk_e, sk_e) \leftarrow \text{Gen}()$, returns (pk_e, s) to \mathcal{A} and stores (s, A, B, sk_e) for answering later queries about session s .
- **Send**($A, B, m \parallel s$): if A is the responder in the run with session identifier s then m is a public encapsulation key pk_e which can be used by \mathcal{B} to compute the correct response. If A is the initiator in the run with session identifier s then \mathcal{B} will accept. Note that if $A = A^*$, $B = B^*$ and $s = s^*$ then \mathcal{B} does not have the decapsulation key and cannot compute the session key.
- **Corrupt**(A): if $A \in \{A^*, B^*\}$ then \mathcal{B} aborts and returns a random bit to \mathcal{C} . Otherwise \mathcal{B} returns any session key and sk_e value allocated to A . Note that for this protocol there are no long-term keys.
- **RevealKey**(A, B, s): if $A \in \{A^*, B^*\}$ and $s = s^*$ then \mathcal{B} aborts and returns a random bit to \mathcal{C} . Otherwise \mathcal{B} returns the session key value allocated to A with session identifier s (or \emptyset if it is undefined).
- **RevealState**(A, B, s): if $A = A^*$, $B = B^*$ and $s = s^*$ then \mathcal{B} aborts and returns a random bit to \mathcal{C} . Otherwise, if A is the initiator in the run with session identifier s then \mathcal{B} returns the private ephemeral key, sk_e , or \emptyset if it is undefined.
- **Expire**(A, B, s): if there is a completed session s at A with B then \mathcal{B} will return a success flag to \mathcal{A} or otherwise return a failure flag to \mathcal{A} .
- **Test**(A, B, s): if $A = A^*$, $B = B^*$ and $s = s^*$ then \mathcal{B} returns k^* to \mathcal{A} . Otherwise \mathcal{B} aborts and returns a random guess for b .

As long as \mathcal{B} does not abort then \mathcal{A} will eventually halt and output its bit b' . Then \mathcal{B} sets $b \leftarrow b'$ and returns b to its challenger \mathcal{C} .

If \mathcal{A} chooses the target session as its test session then \mathcal{B} wins whenever \mathcal{A} wins. This happens with probability at least $1/n^2q$. Note that this

necessarily means that \mathcal{B} does not abort on any query. Then we have

$$\Pr[\mathcal{B} \text{ wins}] \geq \frac{1}{2} \cdot \left(1 - \frac{1}{n^2q}\right) + \Pr[\mathcal{A} \text{ wins}] \cdot \frac{1}{n^2q}$$

$$\text{or } \Pr[\mathcal{B} \text{ wins}] - \frac{1}{2} \geq \frac{1}{n^2q} \left(\Pr[\mathcal{A} \text{ wins}] - \frac{1}{2}\right).$$

Thus the theorem statement follows. □

4 Generic KEM-based AKE protocols

With the building blocks from Sec. 3 we now apply MT-authenticators to AM protocols and optimise them to obtain protocols which are both SK-secure in the realistic UM security model and efficient in comparison with other protocols in the literature. There is no restriction to apply the new MT-authenticator in Fig. 5 only to the new AM protocol in Fig. 6; the authenticator can be applied to any SK-secure AM protocol and any authenticator can be applied to the KEM-based AM protocol. Furthermore, we may apply different MT-authenticators to each of the messages in an AM protocol [HBN06, Thm. 6] resulting in yet more ways to construct different secure protocols.

Due to our field’s focus on post-quantum security in recent years, we emphasise KEM-based and signature-based components in this section, allowing us to apply any of the primitives from the NIST competition library. We illustrate this usage with several different examples in this section, applying both our new KEM-based authenticator and the existing signature-based authenticator to achieve a variety of protocols. Another example, also with potential for post-quantum security, is to apply the MAC-based authenticator of CK01 to our KEM-based AM protocol. This results in a protocol suitable for pre-shared key environments which is a common scenario, for example in TLS and IPsec. Details of a MAC-based generic protocol construction are available in Appendix 4.3.

4.1 Compiled KEM-based protocol and optimization

We start with the AM-secure protocol from Fig. 6 and then apply the compiler consisting of application of the compressed MT-authenticator to

each of its two messages. This leads to the 4-message protocol of Fig. 7.

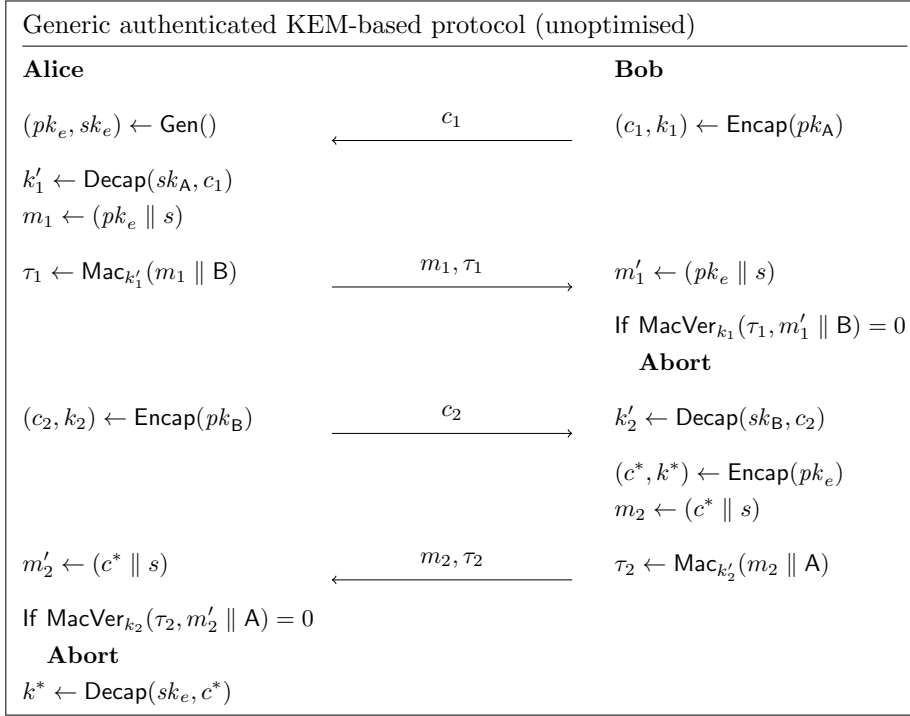


Figure 7: Generic 4-message protocol obtained by compiling the KEM-based AM protocol with the compressed KEM-based MT-authenticator.

Messages 1 and 2 are the result of applying the compressed MT-authenticator to authenticate the ephemeral public key pk_e generated by Alice. Messages 3 and 4 are the result of applying the compressed MT-authenticator to authenticate the encapsulated shared key c^* generated by Bob. The difference between m_1 and m'_1 (resp. m_2 and m'_2) in Fig. 7 is that both players have their own version of s — the MAC verifies the integrity of both the message and the session.

To optimise the 4-message protocol in Fig. 7 we take four simple steps:

1. The messages that were numbered 2 and 3 will be sent in parallel as a new message with number 2. This does not change the order or

contents of any messages.

2. The session identifier, s , will be constructed by the parties as part of the protocol, instead of taking it as an external input to the protocol. Recall that the only requirements on s are to be unique between the parties amongst any incomplete protocol session between the two parties. We choose $s = c_1 \parallel c_2$ where c_1 and c_2 are the (randomised) encapsulations (ciphertexts) generated by each party.
3. Repeated message fields and fields previously generated by message receivers are removed from messages.
4. The protocol parties are re-labelled so that Alice becomes the protocol initiator.

Combining all of these steps we obtain the optimised protocol shown in Fig. 8.

As far as we are aware, the precise protocol of Fig. 8 is new in the literature. There are several existing protocols aimed also at achieving AKE based only on KEMs [FSXY12, SSW20, HNS⁺20] or encryption [dSW17]. Several of these are motivated by the desire to avoid signatures, which tend to suffer efficiency disadvantages compared with KEMs in the post-quantum examples from the NIST competition. The security varies between of each these protocols. For example, the FSXY protocol [FSXY12] provides security against ephemeral key leakage whereas KEMTLS [SSW20], like the protocol of Fig. 8, lacks this property. On the other hand, our protocol does allow state reveals from non-target sessions. KEMTLS is also designed to provide only one-way (server) authentication. Making a judgement on which of these protocols is “better” is therefore difficult since it depends on the security requirements and implementation details. In Sec. 5 we compare efficiency using concrete KEMs and signature schemes to get a better feel for the relative efficiencies.

4.2 Generic protocols using signatures

We now look at two further generic protocols which we can obtain by using signatures in combination with our KEM-based AM protocol. We will need to apply the compressed signature-based authenticator shown in Fig. 9.

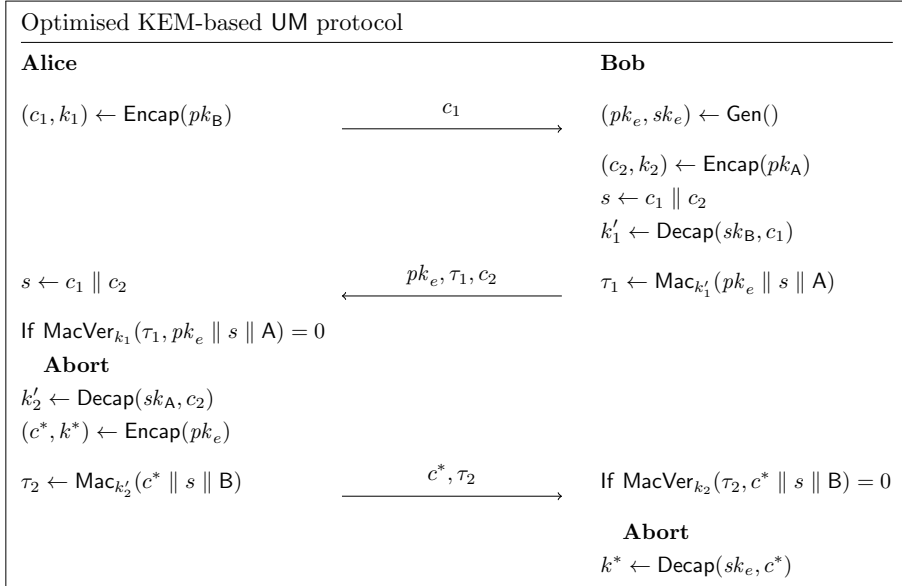


Figure 8: Optimised UM protocol from the KEM-based AM protocol and the KEM-based MT-authenticator.

The authenticator λ_{Sign} is derived from the authenticator of BCK98 by removing the unnecessary message components in exactly the same way as for the encryption- and KEM-based authenticators. As before, the existing proof that the full authenticator is a valid MT-authenticator [BCK98] still holds.

The optimised protocol for the KEM-based AM protocol compiled with the signature-based authenticator is shown in Fig. 10. The optimisation follows the same process as described in Sec. 4.1. Although more general, the resulting protocol has much in common with the signed Diffie–Hellman protocol which has been widely known and deployed for many years and is today the usual AKE in the latest version of TLS (though with only one-sided authentication).

We have another way to authenticate the KEM-based AM protocol, namely to authenticate its two messages with different MT-authenticators. As far as we are aware there are no examples of such a protocol in the

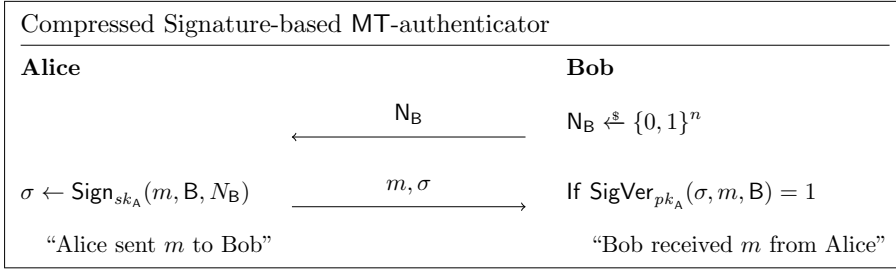


Figure 9: λ_{Sign} , a compressed signature-based MT-authenticator.

existing literature. There can be practical usages, for example when signatures are very expensive to generate but very cheap to verify. In such a case, when a powerful server authenticates its AM message it can shift computation away from a lightweight client by using the signature-based authenticator, while the client can avoid generating signatures by using a different KEM-based authenticator. In Fig. 14 we show the optimised protocol using the KEM-based authenticator for the first message and the signature-based authenticator for the second message. A mirror protocol results from using the two authenticators the other way around. For completeness this optimised protocol is given as Fig. 11.

4.3 MAC-based MT-authenticator

Canetti and Krawczyk [CK01] present also a MAC-based MT-authenticator (interestingly described only in compressed form) as shown in Fig. 12. This authenticator can be useful in scenarios where pre-shared keys exist such as in many use-cases of TLS with lightweight entities and also in session resumption in the latest TLS 1.3 version.

Since MACs are expected to remain secure in the post-quantum setting it makes sense to combine this authenticator with our KEM-based AM protocol to obtain a post-quantum secure AKE protocol suitable for pre-shared key applications. Fig. 13 shows the resulting optimised protocol.

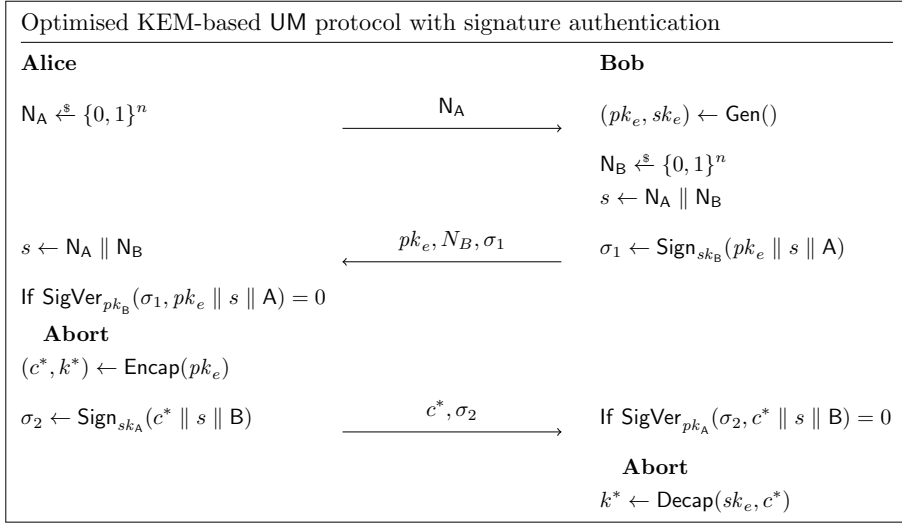


Figure 10: Optimised UM protocol from the KEM-based AM protocol and the signature-based MT-authenticator.

5 Concrete post-quantum secure AKE protocols

In the previous section we have presented optimised generic AKE protocols which will be secure as long as the KEM, signature and MAC primitives are instantiated with secure instances. Even restricting to a handful of currently best-trusted post-quantum primitives, this leads to hundreds of potential concrete protocols, bearing in mind that we have shown that different KEMs and signatures can be mixed in the same protocol and observing that the generic protocols are not symmetric between initiator and responder. The question of whether the concrete instantiated protocols are practical in terms of computational efficiency and message size is a natural one.

5.1 Comparison of different implementations

Tables 1a and 1b present the computational efficiency of various KEMs and signatures from the NIST competition. The figures are taken from

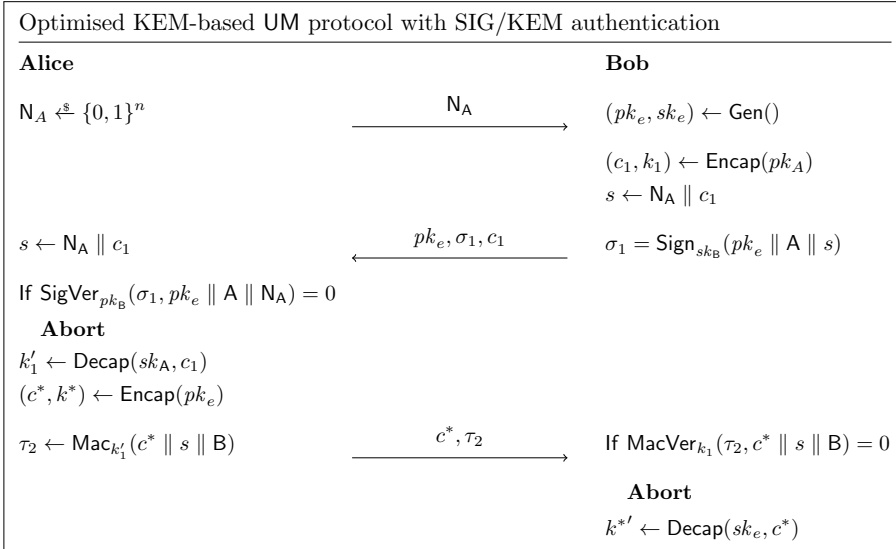


Figure 11: Optimised UM protocol from the KEM-based AM protocol and the signature-based MT-authenticator used for authenticating the first message, and the kem-based MT-authenticator authenticating the second message.

two recent reports from ETSI [Sec21a, Sec21b]. They are not intended as definitive efficiency comparisons — indeed some of the figures have already been improved upon — but rather to illustrate typical ballpark figures and highlight the big variation between many of the existing proposals.

5.2 Computational cost

To give an impression of the computational costs of our new protocols we summarize the number of public key operations needed in each of our optimised protocols in the upper part of Table 2. The lower part of the same table includes the number of similar operations for some prominent existing protocols.

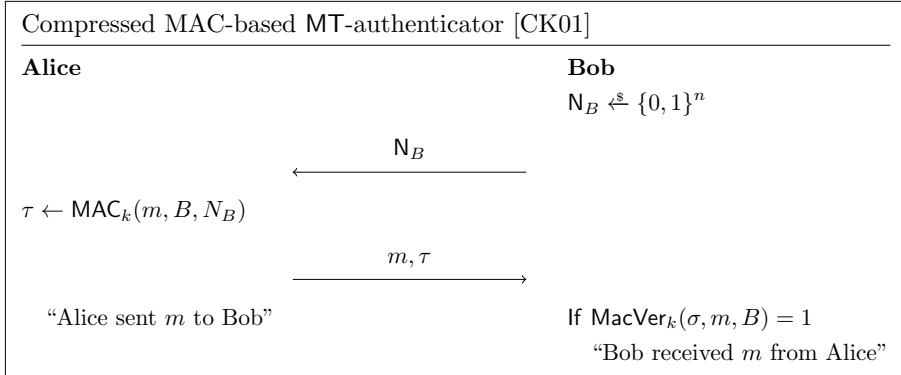


Figure 12: λ_{MAC} , a compressed MAC-based MT-authenticator with shared key k .

All of the protocols in Table 2 use three passes and three rounds. However, they do not all have the same goals or assumptions. TLS and KEMTLS only aim for server-side authentication while our protocol in Fig. 13 assumes pre-shared keys. PQ-WireGuard [HNS⁺20] is a variant of the WireGuard protocol using only KEMs. Its design is based on the FSXY protocol [FSXY12]. All of the protocols in the lower half of Table 2 use only KEMs, both for authentication and key exchange. When comparing with our KEM-only protocol of Fig. 8 we see that the main computational effort is the same as in the three bottom protocols which are all KEM-only protocols. We conclude that our protocols are comparable in computation to existing ones. Another difference between the various protocols is on which side most computations are performed, e.g. in Fig. 14 the initiator Alice encapsulates twice while Bob performs computationally heavier decapsulations and generation of the ephemeral key.

The most obvious difference between the upper and lower part of the table is that our designs have the responder generating the ephemeral KEM key while all existing protocols shown give this task to the initi-

³Using Diffie-Hellman as an ephemeral KEM. Unilateral authentication

⁴Unilateral authentication

⁵Assuming that our KEM-based AM protocol is used as the base protocol.

⁶Encryption is needed in the full protocol, not encapsulation

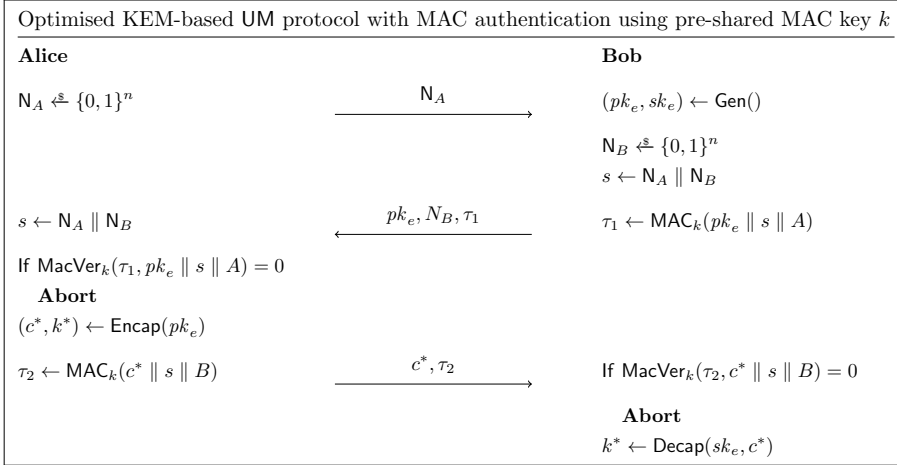


Figure 13: Optimised UM protocol from the KEM-based AM protocol and the MAC-based MT-authenticator.

ating party. We do not believe that either option is inherently better, rather it depends on the relative costs of generation, encapsulation and decapsulation of the instantiating ephemeral KEM. For some well-known KEMs (Table 1a), key generation is far more costly than encapsulation or decapsulation. To minimise the overall protocol cost to both parties it seems better to use an algorithm with more uniform cost for the three KEM operations, but if it is desired to reduce the cost of one party at the expense of the other then different algorithms can be better.

It can be argued that implementation is most efficient when the same concrete KEM is used for all three of the KEM instances in the all-KEM protocols. This should be true at least with regard to the codebase needed in any implementation. However, this may not be the case when it comes to counting computation cycles. Recall that the AM protocol includes generation of an ephemeral public key, while the long-term keys are generated only once before the protocol runs. Therefore it can make sense to use a KEM with an efficient key generation algorithm for the ephemeral KEM, and a different one with a much less efficient key generation algorithm for the KEM using the long-term keys. PQ-WireGuard [HNS⁺20] does

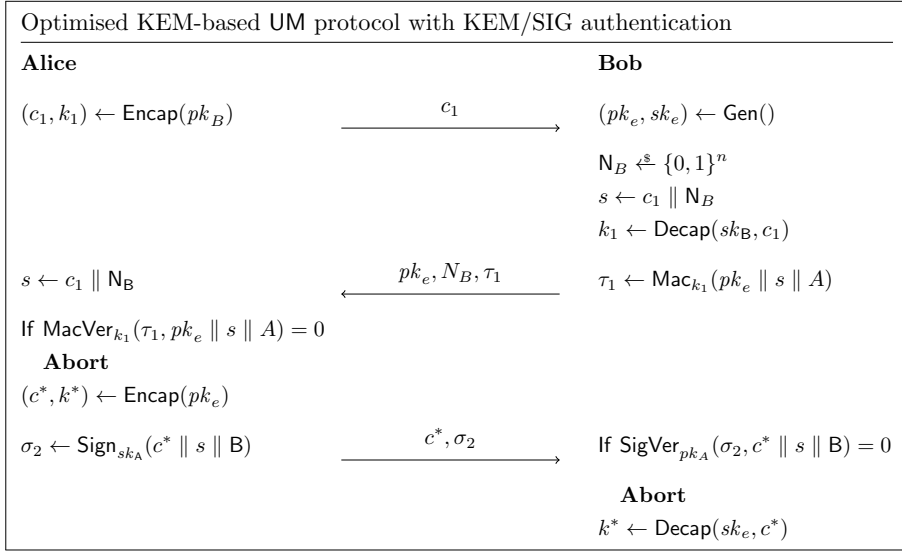


Figure 14: Optimised UM protocol from the KEM-based AM protocol and the KEM-based MT-authenticator used for the first message, and the signature-based MT-authenticator for the second message.

exactly this, using Classic McEliece for the long-term KEM and a variant of Saber for the ephemeral KEM. The size of its public key (Table 1a) shows why using Classic McEliece for the ephemeral KEM seems to be a bad idea.

Current known post-quantum signatures tend to be computationally less efficient than KEM constructions (Table 1b) where signing is much more expensive than decapsulation in known algorithms. It is therefore natural that KEM-based authentication currently is seen favourably. This can change in the future, and the NIST focus on new post-quantum signature proposals may well lead to more efficient post-quantum secure signature algorithms. To our knowledge, there is no analog to our KEM/Sig or Sig/KEM protocols in the literature, neither are we aware of post-quantum proposals for the pre-shared key case.

Table 1: The efficiency of selected post-quantum algorithm proposals. Algorithms **Gen**, **Encap**, **Decap**, **Sign** and **SigVer**, are measured in clock cycles on a standard processor. Parameters public key size (pk), ciphertexts (encapsulations) (ct) and signatures (σ) are measured in bytes.

(a) The efficiency of various KEMs [Sec21a].

	Gen	Encap	Decap	pk	ct	NIST security category
mceliece348864	36641040	44 350	134 745	261120	128	1
mceliece460896	117067765	117 782	271 694	524160	188	3
KYBER512	33856	45 200	59 088	800	768	1
KYBER1024	73544	97 324	115 332	1568	1568	3
ntruhs2048677	309216	83 519	59 729	930	930	1
ntruhs4096821	431667	98 809	75 384	1230	1230	3
LightSaber	45152	49 948	47 852	672	736	1
Saber	66727	79 064	76 612	992	1088	3

(b) The efficiency of various signature schemes [Sec21b].

	Sign	SigVer	σ	NIST security category
Dilithium-3	269 000	118 000	3293	3
Dilithium-5	429 000	179 000	4595	5
FALCON-512	386 678	82 340	666	1
FALCON-1025	789 564	168 498	1280	5

Remark. We find it interesting to remark on a major difference regarding the *symmetry* of the computation requirements between Diffie–Hellman and the AM protocol (Fig. 6) which can be regarded as a generalisation. The computational requirements for Diffie–Hellman are the same for both initiator and responder. In the AM protocol the initiator runs **Gen** and **Decap** while the responder runs only **Encap**. Of itself this is not significant, since **Encap** really has two purposes: to generate the new key for the responder and to generate the encapsulation for the initiator. Thus in the Diffie–Hellman case the cost of **Encap** is the same as the cost of **Gen** plus the cost of **Decap**. However, in all the examples in Table 1a this is nowhere close to being true. Indeed **Encap** is always significantly cheaper

Table 2: The number of public key operations for ours and existing protocols.

	Initiator					Responder				
	Gen _e	Encap	Decap	Sign	SigVer	Gen _e	Encap	Decap	Sign	SigVer
Fig 8. KEM/KEM	0	2	1	0	0	1	1	2	0	0
Fig 10. Sig/Sig	0	1	0	1	1	1	1	0	1	1
Fig 14. KEM/Sig	0	2	0	1	0	1	0	2	0	1
Fig 11. Sig/KEM	0	1	1	0	1	1	1	1	1	0
Fig 13. MAC/MAC	0	1	0	0	0	1	0	1	0	0
TLS 1.3 ³	1	0	1	0	1	0	1	0	1	0
KEMTLS ⁴ [SSW20]	1	1	1	0	0	0	1	1	0	0
KEMTLS-pdk [SSW21]	1	1	2	0	0	0	2	1	0	0
PQ-WireGuard [HNS ⁺ 20]	1	1	2	0	0	0	2	1	0	0
SSW17 ⁵ [dSW17]	1	1 ⁶	2	0	0	0	2	1	0	0

Table 3: What comprises the messages sent in each protocol.

	Message 1	Message 2	Message 3
Fig 8. KEM/KEM	ct	pk, ct, MAC	ct, MAC
Fig 10. Sig/Sig	N	pk, N, σ	ct, σ
Fig 14. KEM/Sig	ct	pk, N, MAC	pk, MAC
Fig 11. Sig/KEM	N	pk, σ, ct	ct, MAC
Fig 13. MAC/MAC	N	pk, N, MAC	ct, MAC

than Gen plus Decap, which may be important when deciding which party take the role of initiator in a protocol run.

5.3 Communications cost

In Table 3 we take an inventory of the message fields in each of our abstract protocols. Due to the optimisation techniques explained earlier, the number of fields sent and received by each party is three in all cases. Informally, at least, this is a minimum since the ephemeral public key needs to be communicated and then used in the response, and each of these two messages must be authenticated using a fresh value chosen by the other party.

The size of these fields depends on the parameters of the concrete primitives chosen. In July 2022, NIST announced a first list of selected candidates as a result of its Post-Quantum Cryptography competition [AAC⁺22], pointing out CRYSTALS-Kyber as their selected KEM and CRYSTALS-Dilithium as their selected signature algorithm. Using the real-world efficiency of the Kyber KEM and the Dilithium signature scheme, in Tables 1a and 1b and naively adding up these numbers, all messages in our Fig. 14 protocol would be under under 5 kB for Kyber-1024, which definitely is practical. Another look at the ephemeral public key sizes in Table 1a shows why the choice of Saber in PQ-WireGuard [HNS⁺20] is an obvious one. We note that before its recent demise, SIKE looked an even more promising candidate to minimise the ephemeral key size.

Just as for computation efficiency, currently accepted post-quantum secure signature candidates do not look attractive for communications efficiency as shown in Table 1b. To minimise signature size FALCON is a better choice than Dilithium, but requires a trade-off with computation.

We reiterate that Table 3 assumes that authentic long-term public keys are available to all parties by some external channel. This fits some real-world protocols (such as WireGuard) but not others (such as TLS). Post-quantum signatures used to certify the long-term public keys can be chosen independently of other concrete choices in the protocol. This choice will obviously affect both the computation for each party and the size of the protocol messages. Although registration of public keys can avoid use of post-quantum signatures [GHL⁺22], it seems necessary to use signatures to achieve usual certificate properties.

6 Conclusion and future work

As summarized in Sec. 1.2, the main contributions of this work are the new KEM-based authenticator and corresponding security proof, the new proofs in the AM-model and the derivation of several new generic AKE protocols. We hope that the flexibility of protocol designs can be useful in fitting AKE protocols to different application use cases and that as new concrete KEMs and signature schemes are developed the generic protocols will yield new and interesting instantiations. Some of the ways to extend the work are the following.

- Adding additional security properties; for example, application of the twisted-PRF trick [FSXY12] can likely be added to an AM-protocol to secure against ephemeral leakage.
- Check whether use of hybrid-KEMs (secure against conventional adversaries based on traditional assumptions and secure against post-quantum adversaries based on new assumptions) can be usefully applied to obtain hybrid-secure AKE [BBF⁺19].
- Since the modular approach does not naturally lead to tight reductions, it would be useful to improve on this, although in the end it may be necessary to complement new protocol designs obtained from the modular approach with a monolithic proof in a stronger model for some concrete protocol.
- Applying an authenticator just to one message (from a two-message AM protocol) will allow for unilateral authentication such as is common in TLS. The security and efficiency of such a generic protocol deserves analysis.
- Real-world implementations of the generic protocols is also left to future work — obviously this would be a prerequisite for future adaptation, and give us a more concrete comparison of their efficiency in the real world.

References

- [AAC⁺22] Gorjan Alagic, Daniel Apon, David Cooper, Quynh Dang, Thinh Dang, John Kelsey, Jacob Lichtinger, Carl Miller, Dustin Moody, Rene Peralta, Ray Perlner, Angela Robinson, Daniel Smith-Tone, and Yi-Kai Liu. Status Report on the Third Round of the NIST Post-Quantum Cryptography Standardization Process. Technical report, National Institute of Standards and Technology, 2022. available at <https://csrc.nist.gov/publications/detail/nistir/8413/final>.
- [AASA⁺20] Gorjan Alagic, Jacob Alperin-Sheriff, Daniel Apon, David Cooper, Quynh Dang, John Kelsey, Yi-Kai Liu, Carl Miller,

Dustin Moody, Rene Peralta, Ray Perlner, Angela Robinson, and Daniel Smith-Tone. Status Report on the Second Round of the NIST Post-Quantum Cryptography Standardization Process. Technical report, National Institute of Standards and Technology, 2020. available at <https://csrc.nist.gov/publications/detail/nistir/8309/final>.

- [ADH⁺22] Yawning Angel, Benjamin Dowling, Andreas Hülsing, Peter Schwabe, and Florian Weber. Post quantum noise. Cryptology ePrint Archive, Report 2022/539, 2022. <https://eprint.iacr.org/2022/539>.
- [BBF⁺19] Nina Bindel, Jacqueline Brendel, Marc Fischlin, Brian Goncalves, and Douglas Stebila. Hybrid key encapsulation mechanisms and authenticated key exchange. In Jintai Ding and Rainer Steinwandt, editors, *Post-Quantum Cryptography - 10th International Conference, PQCrypto 2019*, pages 206–226. Springer, Heidelberg, 2019. doi:10.1007/978-3-030-25510-7_12.
- [BCD⁺16] Joppe W. Bos, Craig Costello, Léo Ducas, Ilya Mironov, Michael Naehrig, Valeria Nikolaenko, Ananth Raghunathan, and Douglas Stebila. Frodo: Take off the ring! Practical, quantum-secure key exchange from LWE. In Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi, editors, *ACM CCS 2016*, pages 1006–1018. ACM Press, October 2016. doi:10.1145/2976749.2978425.
- [BCK98] Mihir Bellare, Ran Canetti, and Hugo Krawczyk. A modular approach to the design and analysis of authentication and key exchange protocols (extended abstract). In *30th ACM STOC*, pages 419–428. ACM Press, May 1998. doi:10.1145/276698.276854.
- [BHMS17] Nina Bindel, Udyani Herath, Matthew McKague, and Douglas Stebila. Transitioning to a quantum-resistant public key infrastructure. In Tanja Lange and Tsuyoshi Takagi, editors,

- Post-Quantum Cryptography - 8th International Workshop, PQCrypto 2017*, pages 384–405. Springer, Heidelberg, 2017. doi:10.1007/978-3-319-59879-6_22.
- [BJS15] Florian Bergsma, Tibor Jager, and Jörg Schwenk. One-round key exchange with strong security: An efficient and generic construction in the standard model. In Jonathan Katz, editor, *PKC 2015*, volume 9020 of *LNCS*, pages 477–494. Springer, Heidelberg, March / April 2015. doi:10.1007/978-3-662-46447-2_21.
- [BL17] Daniel J Bernstein and Tanja Lange. Post-quantum cryptography. *Nature*, 549(7671):188–194, 2017.
- [CFS⁺21] Sofía Celi, Armando Faz-Hernández, Nick Sullivan, Goutam Tamvada, Luke Valenta, Thom Wiggers, Bas Westerbaan, and Christopher A. Wood. Implementing and measuring KEMTLS. *Cryptology ePrint Archive*, Report 2021/1019, 2021. <https://eprint.iacr.org/2021/1019>.
- [CK01] Ran Canetti and Hugo Krawczyk. Analysis of key-exchange protocols and their use for building secure channels. In Birgit Pfitzmann, editor, *EUROCRYPT 2001*, volume 2045 of *LNCS*, pages 453–474. Springer, Heidelberg, May 2001. doi:10.1007/3-540-44987-6_28.
- [Cre11] Cas Cremers. Examining indistinguishability-based security models for key exchange protocols: the case of CK, CK-HMQV, and eCK. In Bruce S. N. Cheung, Lucas Chi Kwong Hui, Ravi S. Sandhu, and Duncan S. Wong, editors, *ASIACCS 11*, pages 80–91. ACM Press, March 2011.
- [DAL⁺17] Jintai Ding, Saed Alsayigh, Jean Lancrenon, Saraswathy RV, and Michael Snook. Provably secure password authenticated key exchange based on RLWE for the post-quantum world. In Helena Handschuh, editor, *CT-RSA 2017*, volume 10159 of *LNCS*, pages 183–204. Springer, Heidelberg, February 2017. doi:10.1007/978-3-319-52153-4_11.

- [dSW17] Cyprien Delpech de Saint Guilhem, Nigel P. Smart, and Bogdan Warinschi. Generic forward-secure key agreement without signatures. In Phong Q. Nguyen and Jianying Zhou, editors, *ISC 2017*, volume 10599 of *LNCS*, pages 114–133. Springer, Heidelberg, November 2017.
- [DXL12] Jintai Ding, Xiang Xie, and Xiaodong Lin. A simple provably secure key exchange scheme based on the learning with errors problem. Cryptology ePrint Archive, Paper 2012/688, 2012. <https://eprint.iacr.org/2012/688>. URL: <https://eprint.iacr.org/2012/688>.
- [FSXY12] Atsushi Fujioka, Koutarou Suzuki, Keita Xagawa, and Kazuki Yoneyama. Strongly secure authenticated key exchange from factoring, codes, and lattices. In Marc Fischlin, Johannes Buchmann, and Mark Manulis, editors, *PKC 2012*, volume 7293 of *LNCS*, pages 467–484. Springer, Heidelberg, May 2012. doi:10.1007/978-3-642-30057-8_28.
- [GHL⁺22] Tim Güneysu, Philip Hodges, Georg Land, Mike Ounsworth, Douglas Stebila, and Greg Zaverucha. Proof-of-possession for KEM certificates using verifiable generation. Cryptology ePrint Archive, Report 2022/703, 2022. <https://eprint.iacr.org/2022/703>.
- [GHP18] Federico Giacon, Felix Heuer, and Bertram Poettering. KEM combiners. In Michel Abdalla and Ricardo Dahab, editors, *PKC 2018, Part I*, volume 10769 of *LNCS*, pages 190–218. Springer, Heidelberg, March 2018. doi:10.1007/978-3-319-76578-5_7.
- [HBN06] Yvonne Hitchcock, Colin Boyd, and Juan Manuel González Nieto. Modular proofs for key exchange: rigorous optimizations in the Canetti-Krawczyk model. *Appl. Algebra Eng. Commun. Comput.*, 16(6):405–438, 2006. doi:10.1007/s00200-005-0185-9.
- [HNS⁺20] Andreas Hülsing, Kai-Chun Ning, Peter Schwabe, Florian Weber, and Philip R. Zimmermann. Post-quantum Wire-

- Guard. Cryptology ePrint Archive, Report 2020/379, 2020. <https://eprint.iacr.org/2020/379>.
- [JKRS21] Tibor Jager, Eike Kiltz, Doreen Riepel, and Sven Schäge. Tightly-secure authenticated key exchange, revisited. In Anne Canteaut and François-Xavier Standaert, editors, *EURO-CRYPT 2021, Part I*, volume 12696 of *LNCS*, pages 117–146. Springer, Heidelberg, October 2021. doi:10.1007/978-3-030-77870-5_5.
- [KL14] Jonathan Katz and Yehuda Lindell. *Introduction to Modern Cryptography, Second Edition*. CRC Press, 2014. URL: <https://www.crcpress.com/Introduction-to-Modern-Cryptography-Second-Edition/Katz-Lindell/p/book/9781466570269>.
- [Kra05] Hugo Krawczyk. HMQR: A high-performance secure Diffie-Hellman protocol. In Victor Shoup, editor, *CRYPTO 2005*, volume 3621 of *LNCS*, pages 546–566. Springer, Heidelberg, August 2005. doi:10.1007/11535218_33.
- [Pei15] Chris Peikert. A decade of lattice cryptography. Cryptology ePrint Archive, Paper 2015/939, 2015. <https://eprint.iacr.org/2015/939>. URL: <https://eprint.iacr.org/2015/939>.
- [Sec21a] ETSI Technical Committee Cyber Security. Quantum-safe public-key encryption and key encapsulation. ETSI TR 103823, ETSI, October 2021. URL: https://www.etsi.org/deliver/etsi_tr/103800_103899/103823/01.01.01_60/tr_103823v010101p.pdf.
- [Sec21b] ETSI Technical Committee Cyber Security. Quantum-safe signatures. ETSI TR 103616, ETSI, September 2021. URL: https://www.etsi.org/deliver/etsi_tr/103600_103699/103616/01.01.01_60/tr_103616v010101p.pdf.
- [SSW20] Peter Schwabe, Douglas Stebila, and Thom Wiggers. Post-quantum TLS without handshake signatures. In Jay Ligatti,

Xinming Ou, Jonathan Katz, and Giovanni Vigna, editors, *ACM CCS 2020*, pages 1461–1480. ACM Press, November 2020. doi:10.1145/3372297.3423350.

- [SSW21] Peter Schwabe, Douglas Stebila, and Thom Wiggers. More efficient post-quantum KEMTLS with pre-distributed public keys. In Elisa Bertino, Haya Shulman, and Michael Waidner, editors, *ESORICS 2021, Part I*, volume 12972 of *LNCS*, pages 3–22. Springer, Heidelberg, October 2021. doi:10.1007/978-3-030-88418-5_1.

Appendix D

SWOOSH: Practical Lattice-Based Non-Interactive Key Exchange

Swoosh: Practical Lattice-Based Non-Interactive Key Exchange

PHILLIP GAJLAND, Max Planck Institute for Security and Privacy and Ruhr-University Bochum

BOR DE KOCK, NTNU – Norwegian University of Science and Technology

MIGUEL QUARESMA, Max Planck Institute for Security and Privacy

GIULIO MALAVOLTA, Max Planck Institute for Security and Privacy

PETER SCHWABE, Max Planck Institute for Security and Privacy and Radboud University

The advent of quantum computers has generated a wave of interest for post-quantum cryptographic schemes, as a replacement for currently used cryptographic primitives. In this context, lattice-based cryptography has emerged as the leading paradigm to build post-quantum cryptography. However, all viable replacements of the classical Diffie-Hellman key exchange require additional rounds of interactions, thus failing to achieve all the benefits of this protocol. Although earlier work has shown that lattice-based Non-Interactive Key Exchange (NIKE) is theoretically possible, it has been considered too inefficient for real-life applications.

In this work, we provide the first evidence against this folklore belief. We construct a practical lattice-based NIKE whose security is based on the standard module learning with errors (M-LWE) problem in the quantum random oracle model. Our scheme is obtained in two steps: (i) A passively-secure construction that achieves a strong notion of correctness, coupled with (ii) a generic compiler that turns any such scheme into an actively-secure one. To substantiate our efficiency claim, we present an optimised implementation of our construction in Rust and Jasmin, demonstrating its applicability to real-world scenarios. For this we obtain public keys of approximately 220 KBs and the computation of shared keys takes than 12 million cycles on an Intel Skylake CPU at a post-quantum security level of more than 120 bits.

1 INTRODUCTION

A key exchange is a cryptographic primitive that allows two users to agree on a common secret key over an insecure channel, such as the Internet. If the protocol consists of a single, asynchronous message from each party, then we refer to it as a *Non-Interactive Key Exchange* (NIKE). The seminal work of Diffie and Hellman [34] introduced the well-known NIKE scheme that marked the birth of public-key cryptography; each party sends a single group element g^x (or g^y , respectively) and the shared key can be derived by computing $(g^y)^x = (g^x)^y$. From a theoretical stand-point NIKE implies the existence of public key encryption (PKE), key encapsulation mechanism (KEM), and even authenticated key-exchange (AKE) when combining the results of [23] with [41]. Moreover, in practice, the Diffie-Hellman key exchange lies at the heart of protocols such as Transport Layer Security (TLS) [71], the Signal protocol, or the Noise protocol framework [66].

The looming threat of quantum computers, combined with the discovery of efficient quantum algorithms for factoring integers and computing discrete logarithms [75], has motivated the cryptographic community to explore solutions based on new mathematical structures, departing from protocols based on the Diffie-Hellman key exchange. In particular, lattice-based cryptography [70] has emerged as the leading paradigm for constructing *post-quantum* cryptographic schemes. As a result of NIST’s recent standardisation process, three, of the four algorithms that were selected for standardisation, are lattice-based [58, 69, 72].

While efficient lattice-based key exchange protocols exist [5, 21, 72], they are all qualitatively different from the standard Diffie-Hellman-style key exchange, in the sense that they require *additional rounds of interaction*. For many applications, where interaction is already built-in, these protocols are perfectly fine substitutes for the Diffie-Hellman (that is not post-quantum secure). However, in many scenarios of interest, the non-interactive nature of NIKE protocols is crucial (we discuss concrete examples in further detail in Section 1.1). Unfortunately, despite almost two decades of research on the subject, an efficient lattice-based NIKE remains elusive. Perhaps more worryingly, a recent work [46] has shown theoretical barriers on the efficiency of lattice-based NIKE, calling into question whether it is even possible to build a practical scheme at all. Thus, the current state of affairs, leaves open the following question:

**Is lattice-based *non-interactive*
key exchange feasible in practice?**

In our work we seek to answer this question in the affirmative, and show that lattice-based NIKE can be made efficient enough to be used in practice, whilst maintaining post-quantum security.

1.1 NIKE vs. KEMs

While the Diffie-Hellman (DH) key exchange happens to be non-interactive, most post-quantum approaches to key exchange are interactive key-encapsulation mechanisms (KEMs). Intuitively, the difference is as follows. In a NIKE, any user A can use their secret key sk_A together with the public key pk_B of a user B to derive a shared key k_{AB} . At the same time, and without interaction with A , user B can compute the same shared key k_{AB} by combining their secret key sk_B with the public key pk_A of user A . However, in a KEM, this key-derivation becomes a two-stage, inherently asymmetric and interactive process. First, A invokes an encapsulation routine that accepts pk_B as input and produces as output the shared key k_{AB} , and a ciphertext ct , which they send to B . User B then invokes the decapsulation routine that takes as input ct and secret key sk_B to produce the same shared secret k_{AB} .

Some protocols employing DH do not actually make use of the non-interactive nature and can thus be migrated to post-quantum KEMs in a straight-forward manner. Probably the most prominent example is TLS, which uses the DH key exchange with ephemeral keys on both sides

for forward secrecy and has been updated to offer post-quantum security by using KEMs in multiple papers [15, 22, 64] and real-world deployments [53–55, 78].

Other protocols *do* make use of the non-interactive nature of DH and their migration to post-quantum primitives is thus much more involved. A common pattern in these protocols is that they use static DH keys for authentication. One example is OPTLS by Krawczyk and Wee [52], a proposal that eliminates the need for handshake signatures in TLS. The idea was picked up in the post-quantum setting in the KEMTLS proposal by Schwabe, Stebila, and Wiggers [73]. Like OPTLS, also KEMTLS eliminates the need for handshake signatures, but unlike OPTLS uses static KEM keys for authentication. This comes at the expense of requiring more communication round-trips until full server authentication is achieved. This can be problematic for some protocols like HTTPS that allow the server to send early payload data before the handshake is finished. Similar issues with delayed authentication when moving from DH to KEMs were identified in the migration of WireGuard to the post-quantum setting in [48] and in the the recently proposed post-quantum version of the Noise protocol framework [9].

These examples all manage to migrate from the DH setting to the KEM setting at the cost of further communication round trips, and without having to move to signature-based authentication or engaging even more involved cryptographic primitives. If communicating parties cannot be assumed to be online at the same time, this approach is doomed to fail. A prominent example of precisely this asynchronous communication setting is the Signal secure-messaging protocol and specifically the X3DH protocol [60] that is invoked when a user A starts their communication with a (possibly offline) user B . The X3DH protocol uses a combination of ephemeral, static, and semi-static DH keys to achieve forward secrecy, mutual authentication, and offline deniability without the need for direct interaction between A and B . There have been multiple attempts to migrate X3DH to the post-quantum setting [25, 26, 36, 47, 76] but they all either assume the existence of a reasonably efficient post-quantum NIKE, or fail to achieve the same security and privacy as the pre-quantum version from a single simple asymmetric primitive.

1.2 Our Contributions

In this work, we demonstrate the practical feasibility of lattice-based *non-interactive key exchange*. We propose a new scheme, that we call “Swoosh”, based on the hardness of the M-LWE problem. We show a proof of its security, both in the passive and active setting, and provide parameter sets for the former with over 120-bits of security against quantum adversaries (using the best known attacks that incorporate recent advances in lattice cryptanalysis). Our contributions can be succinctly summarised as follows.

- (1) We propose a new construction of NIKE based on the hardness of the M-LWE problem. Our construction is based on the standard template [35, 57], but with a new tweak that

allows us to prove a strong notion of correctness (which, in turn, is necessary to achieve active security) in the quantum random oracle model (QROM). Somewhat interestingly, our use of the random oracle appears to be different from the Fiat-Shamir [40] and the Fujisaki-Okamoto [42, 43] transformations, and may thus be of independent interest.

- (2) We propose a compiler to generically lift a passively secure NIKE to an actively secure scheme, using non-interactive zero-knowledge (NIZK) proofs. While this approach is folklore, to the best of our knowledge it has never appeared explicitly in the literature. Furthermore, the exact notion of passive security needed for the proof to go through, turns out to be surprisingly subtle to identify.
- (3) We carefully select parameters for the passively secure NIKE and instantiate the scheme with parameters achieving 120 bits of security against quantum adversaries. The resulting scheme we call “Passive-Swoosh” and the full scheme including the NIZK “Swoosh”.
- (4) We provide an implementation of Passive-Swoosh written in a combination of Rust and Jasmin, and show that the public keys of Passive-Swoosh are smaller than the ones of the smallest parameter set of Classic McEliece [1], an interactive KEM selected for round 4 of the NIST-PQC competition. We also demonstrate that in terms of speed, Passive-Swoosh outperforms CSIDH [30], the only currently known (and realistic) post-quantum NIKE, by orders of magnitude.

1.3 Related work

Post-quantum NIKE. While interactive KEMs appear to be much more efficient in a post-quantum world than NIKEs, this does not mean that there are no previous proposals for post-quantum NIKE. In [19], Boneh and Zhandry show a construction using iO to construct a multiparty NIKE from pseudorandom generators. Given the impractical performance of iO, the result is mainly of theoretical interest. Much more practical was supersingular-isogeny Diffie-Hellman (SIDH) [32, 50]. However, in 2016, this construction was shown to be susceptible to active attacks [44]. This could be solved by employing the Fujisaki-Okamoto transform [42] in the NIST PQC candidate SIKE [49], but this came at the expense of turning the NIKE into an interactive KEM. Another approach to restoring the active security of SIKE was presented in [10]. This approach preserved the non-interactive nature of SIDH, but required many parallel protocol executions and thus massively increased computation time and message sizes. In 2022, all of these approaches based on SIDH were made obsolete by the Castryck-Decru attack against SIDH [29].

In 2018, Castryck, Lange, Martindale, Panny, and Renes proposed CSIDH, a different approach for constructing an isogeny-based NIKE [30]. CSIDH is not affected by the Castryck-Decru attack, and is arguably the most plausible candidate for practical post-quantum NIKE thus far, although the post-quantum security of concrete parameters is subject of debate [16, 20, 65]. Multiple works

have considered the efficient and secure implementation of CSIDH, currently the fastest approach is a variant called CTIDH [11]. We provide a performance comparison of our proposal to CTIDH in Section 6.2.

Lattice-based NIKE. The idea of lattice-based NIKE using the approach we use for Swoosh is not new; in [57] Lyubashevsky calls it “folklore (since at least 2010)”. An attempt at selecting parameters was made in [33]. However, the proposed scheme did not formally consider passive security, nor active security. Furthermore, the selected parameters resulted in a correctness error that would not even allow the transformation into an actively secure scheme through the use of NIZK proofs that we use for Swoosh.

In fact, prior to our work, lattice-based NIKE was widely considered impractical and this was even substantiated by theoretical evidence. The work of [46] discovered information theoretic barriers in constructing lattice-based NIKE with non-interactive reconciliations. In particular, they showed that any natural candidate of lattice-based NIKE with polynomial modulus-to-noise ratio would necessarily incur an inverse-polynomial correctness error. However, we stress that our work does not contradict the theorem of [46]. As the authors of [46] observe, non-interactive reconciliation is possible, if we consider (M-)LWE instances with *super-polynomial* modulus-to-noise ratio. This is indeed the regime of parameters that we adopt in our work.

2 TECHNICAL OUTLINE

We give a self-contained overview of our approach for constructing a fast lattice-based NIKE. The following is somewhat informal and glosses over many important details, as it is only intended for an intuitive understanding of our approach. The reader is referred to the respective technical sections for precise statements.

The Basic Blueprint. Before delving into the specifics of our approach, it is useful to recall the folklore construction of lattice-based key exchange between Alice and Bob. Let A be a random public $N \times N$ matrix over some ring \mathcal{R}_q and χ a noise distribution. The protocol proceeds as follows; Alice samples \vec{s}_1 and \vec{e}_1 from χ^N , and computes her public key as $\vec{s}_1^\top A + \vec{e}_1^\top$. Bob samples an independent \vec{s}_2 and \vec{e}_2 from χ^N , and computes his public key as $A\vec{s}_2 + \vec{e}_2$. After asynchronously obtaining each other’s public keys, Alice and Bob can compute an approximate shared key as

$$\vec{s}_1^\top (A\vec{s}_2 + \vec{e}_2) \approx (\vec{s}_1^\top A + \vec{e}_1^\top) \vec{s}_2.$$

A simple calculation shows that the shared keys computed by both parties are identical with the exception of the error terms $\vec{s}_1^\top \vec{e}_2$ and $\vec{e}_1^\top \vec{s}_2$ for Alice and Bob, respectively. To correct these errors, known schemes in the literature are based on encryption or interactive *reconciliation*, which can be realised quite efficiently. However, if we insist on a NIKE protocol, no further

interaction is allowed, and Alice and Bob must correct the errors locally. That is, we need to devise a *non-interactive* reconciliation function Rec such that

$$\text{Rec}(\vec{s}_1^\top (A\vec{s}_2 + \vec{e}_2)) = \text{Rec}((\vec{s}_1^\top A + \vec{e}_1^\top) \vec{s}_2).$$

Note that, thus far, we have assumed that both Alice and Bob compute their keys according to the specification of the protocol, i.e., we implicitly only considered passive attacks. However, for the security of the final scheme, it will be necessary to handle parties that may behave arbitrarily. In what follows, we show how we tackle these two challenges separately, in a way that preserves the efficiency and security of the scheme.

Challenge I: Non-Interactive Reconciliation. A natural approach for correcting the errors introduced by the noise terms, is to derive the key by *rounding* the coefficients of the resulting ring element. In fact this is the approach that we adopt in this work, however there are still new ideas required to simultaneously achieve all of the following objectives: (i) security from the hardness of the standard module learning with errors (M-LWE) problem, (ii) reducing the correctness error to negligible, and (iii) maintaining the concrete efficiency of the construction. Here, we stress that a negligible correctness error is not just a matter of convenience, but that a non-negligible correctness error translates to an attack against the scheme: Loosely speaking, this is because the attacker can observe whenever the key agreement fails, therefore learning some information about the secret key of the honest party. Let us now focus on making the rounding approach work for non-interactive reconciliation. A simple calculation shows that the error terms cause a correctness error, only when the term $\vec{s}_1^\top A\vec{s}_2$ falls into a *danger interval*

$$S^* = \left[\frac{q}{4} \pm \beta^2 dN \right] \cup \left[\frac{3q}{4} \pm \beta^2 dN \right],$$

where β is a bound on the norm of the noise distribution and d is the degree of \mathcal{R}_q . It is tempting to conclude that, if q is sufficiently large, then this event only happens with negligible probability. However, this analysis is imprecise as it does not take into account *adaptive* attacks, where the adversary chooses their secret key intentionally to make this event more likely. To prevent this, and obtain a provably secure scheme, we add a *random shift* \mathbf{r} to the term $\vec{s}_1^\top A\vec{s}_2$ to ensure that their sum $\vec{s}_1^\top A\vec{s}_2 + \mathbf{r}$ is indeed uniformly distributed in \mathcal{R}_q . Note that such \mathbf{r} does not need to be kept private, although it is important that it is sampled independently of the keys. Our idea is to sample \mathbf{r} as the output of a hash function (modelled as a random oracle) on input the two public keys. This allows us to achieve two goals simultaneously:

- Both parties can recompute the shift \mathbf{r} without the need of further interaction.
- We can show that $\vec{s}_1^\top A\vec{s}_2 + \mathbf{r}$ is indeed uniformly sampled, even if the adversary has quantum access to the random oracle.

In summary, we are able to build a non-interactive reconciliation mechanism so that the scheme is provably secure (in the passive settings) against the standard M-LWE assumption, in the QROM. In fact, we are also able to show a strong notion of correctness, namely that the adversary cannot cause a reconciliation error, *even if it is allowed to choose both secret keys*. This strong notion of correctness will be useful when lifting the scheme to the active setting.

Challenge II: From Passive to Active Security. The above discussion concerns keys that are guaranteed to be well-formed (passive security). However, in real-world scenarios we have to deal with attackers that can behave arbitrarily. In the stronger notion of *active security* [28, 41] the adversary is given access to various oracles that allow him to register honest keys, register corrupt keys (ones to which he does not know the corresponding public key), or reveal the shared key between an honest key and a corrupted one. Ultimately the adversary wins if he can distinguish between a random key and a shared key, that was derived from two honestly generated key pairs.

In order to prove the active security of our scheme we present a *compiler* that generically lifts our scheme to the active setting using non-interactive zero-knowledge (NIZK) proofs. Here it is crucial that our scheme satisfies the aforementioned strong notion of correctness, since the only thing that the NIZK guarantees is that the keys are in the support of the honest distributions, but otherwise they may be chosen arbitrarily. For technical reasons, we require a NIZK that satisfies the strong property of simulation-sound online-extractability. We refer the reader to Section 5 for more details.

Putting Everything Together. Overall, we obtain a passively secure construction in the QROM assuming the hardness of the Module-LWE (M-LWE) problem (for the active settings, we additionally require a NIZK proof). Compared to Ring-LWE (R-LWE), M-LWE gives us greater flexibility over the choice of parameters, when implementing our scheme. However, this introduces an additional complication: Unlike the case for R-LWE, where single polynomials are considered and their multiplication is commutative, in the case of M-LWE we work with matrices where the matrix multiplication is not generally commutative. For the general case of two parties without predefined roles in a protocol, there is no way to know ahead of time whether to left multiply or right multiply. This means that each public key is effectively duplicated by adding a left multiplied key and right multiplied key. However, we argue that in many cases, when parties have predefined roles in a protocol, such as a server or client, this issue can be resolved (the server could “go right” and the client “left” or vice versa). We defer a more detailed discussion of this to Section 5.3.

Our parameters are selected as to provide more than 120 bits of post-quantum security, taking into account recent advances in lattice cryptanalysis. We work over the ring $\mathcal{R}_q := \mathbb{Z}_q[X]/(X^d + 1)$ with $d = 256$. Along with our public matrix $A \in \mathcal{R}_q^{N \times N}$, where $N = 32$, this gives us a lattice

dimension of 8192. In order to reduce the correctness error to reasonable levels, q had to be sufficiently large. We choose $q = 2^{214} - 255$, a prime that is simultaneously NTT-friendly and close to a power-of-two making for more efficient field arithmetic. Furthermore, we use ternary noise sampled from a centred binomial distribution, for the sake of efficiency.

Finally, we provide an open-source implementation of Passive-Swoosh in Rust and Jasmin, which employs numerous optimisations rendering competitive benchmarks. Due to the modular fashion of our implementation we note that it can easily be tailored to use different parameters or be incorporated with suitable NIZKs. We defer a more detailed discussion to Section 6.

3 PRELIMINARIES

In this Section we introduce our notation and review some quantum preliminaries along with the relevant lattice-based hardness assumptions.

3.1 Notation

We start by defining some standard notation used throughout the paper.

Sets, Vectors, Polynomials and Norms. For integers a, b , where $a < b$, $[a, b]$ denotes the set $\{a, a + 1, \dots, b\}$. For any positive $\beta \in \mathbb{Z}$, we define the set $[\beta] := \{-\beta, \dots, -1, 0, 1, \dots, \beta\}$, and let $x \xleftarrow{\$} \mathcal{S}$ denote the uniform sampling of x from the set \mathcal{S} . Let \mathbb{Z}_q denote the ring of integers modulo a prime q . We define $\mathcal{R} := \mathbb{Z}[X]/(X^d + 1)$ to be the ring of integer polynomials modulo $X^d + 1$, for d a power of 2, and $\mathcal{R}_q := \mathbb{Z}_q[X]/(X^d + 1)$ the ring of integer polynomials modulo $X^d + 1$ where each coefficient is reduced modulo q . We assume that that, for any N , a uniformly sampled N -dimensional square matrix over \mathcal{R}_q is invertible with probability c , for some constant $0 < c \leq 1$. For concreteness, we conservatively set this constant to be $c = 0.5$. Bold upper case letters \mathbf{A} and bold lower case letters with arrows $\vec{\mathbf{a}}$ denote matrices and column vectors over \mathcal{R}_q , respectively; for row vectors we use the transpose $\vec{\mathbf{b}}^\top$.

For a polynomial $f \in \mathcal{R}_q$, let $\vec{f} \in \mathbb{Z}_q^d$ denote the coefficient vector of f , and $f_i \in \mathbb{Z}_q$ the i^{th} coefficient. However, we denote the constant coefficient by $\tilde{f} := f_0 \in \mathbb{Z}_q$. For an element $f_i \in \mathbb{Z}_q$, we write $|f_i|$ to mean $|f_i \bmod q|$. Let the ℓ_∞ and ℓ_p norms for $f = f_0 + f_1X + \dots + f_{d-1}X^{d-1} \in \mathcal{R}_q$ be defined as

$$\|f\|_\infty := \max_{0 \leq i \leq d-1} |f_i| \quad \text{and} \quad \|f\|_p := \sqrt[p]{\sum_{i=0}^{d-1} |f_i|^p},$$

respectively. If $\vec{f} = (f_1, \dots, f_k) \in \mathcal{R}_q^k$, then

$$\|\vec{f}\|_\infty := \max_{1 \leq i \leq k} \|f_i\|_\infty \quad \text{and} \quad \|\vec{f}\|_p := \sqrt[p]{\sum_{i=1}^k \|f_i\|_p^p}.$$

By default $\|\vec{f}\| := \|\vec{f}\|_2$.

Probabilities, Algorithms and Games. The support of a discrete random variable X is defined as

$$\text{sup}(X) := \{x \in \mathbb{R} : \Pr[X = x] > 0\}.$$

Algorithms are denoted by upper-case letters in sans-serif font, such as A and B . Unless otherwise stated all algorithms are probabilistic and $(x_1, \dots) \stackrel{s}{\leftarrow} A(y_1, \dots)$ is used to denote that A returns (x_1, \dots) when run on input (y_1, \dots) . When A has oracle access to B during its execution, this is denoted by A^B . For a probabilistic algorithm A , the notation $x \in A(y)$ denotes that x is a possible output of A on input y . We use code-based security games [13], where $\Pr[G \Rightarrow 1]$ denotes the probability that the final output of game G is 1. The notation $\llbracket B \rrbracket$, where B is a Boolean statement, refers to a bit that is 1 if the statement is true and 0 otherwise.

3.2 Quantum Preliminaries

We review some quantum preliminaries as stated in [37].

Qubits, n -qubit States and Measurement. A qubit $|x\rangle := \alpha_0 |0\rangle + \alpha_1 |1\rangle$ is a unit vector in some Hilbert space \mathcal{H} . When $\alpha_0 \neq 1$ and $\alpha_1 \neq 1$, we say that $|x\rangle$ is in *superposition*. An n -bit quantum register $|x\rangle := \sum_{i=0}^{2^n-1} \alpha_i |i\rangle$ is a unit vector in $\mathcal{H}^{\otimes n} \cong \mathbb{C}^{2^n}$, that is $\sum_{i=0}^{2^n-1} |\alpha_i|^2 = 1$ for $\alpha_i \in \mathbb{C}$. We call the set $\{|0\rangle, |1\rangle, \dots, |2^n - 1\rangle\}$ the *computational basis* and say that $|x\rangle$ is *entangled* when $|x\rangle$ cannot be written as the tensor product of single qubits. Unless otherwise stated, measurements are done in the computational basis. After measuring a quantum register $|x\rangle = \sum_{i=0}^{2^n-1} \alpha_i |i\rangle$ in the computational basis, the state *collapses* and $|x\rangle = \pm |i\rangle$ with probability $|\alpha_i|^2$.

Quantum Algorithms. A quantum algorithm A is a sequence of unitary operations U_i , where unitary operations are defined to map unit vectors to unit vectors, while preserving the normalisation constraint of quantum registers. A quantum oracle algorithm A^O is defined analogously, and can additionally query the oracle O before (or after) executing a unitary U_i . As quantum computations need to be reversible, we model an oracle $O : X \rightarrow Y$ by a unitary U_O that maps $|x\rangle |y\rangle \mapsto |x\rangle |y \oplus O(x)\rangle$. For an oracle O , we write $|O\rangle$ to denote that an algorithm has quantum-access to U_O .

Quantum Random Oracle Model. In the random oracle model [12], all parties have access to a uniformly sampled random function H . Since quantum adversaries can evaluate hash functions in superposition, we model quantum adversaries to have quantum access to random oracles [18]. Specifically, we assume that all algorithms have access to the unitary implementing the mapping:

$$|x\rangle |y\rangle \mapsto |x\rangle |y \oplus H(x)\rangle$$

where H is a uniformly sampled random function.

Query Depth and Query Parallelism. As in the work of [8] we consider the query depth D of an adversary making a total of Q_H random oracle queries. This is important in practice because for highly parallel adversaries we have $D \ll Q_H$. By setting $D := Q_H$ we obtain the bounds for sequential adversaries. We will use the following technical Lemma from [8].

LEMMA 3.1 (SEARCH IN UNSTRUCTURED FUNCTIONS [8, LEM. 2]). *Let H be a random function drawn from a distribution such that $\Pr[H(x) = 1] \leq \lambda$ for all x . Let A be an adversary with query depth D , making at most Q_H many queries to H . Then*

$$\Pr \left[H(x) = 1 : b \stackrel{\$}{\leftarrow} A^H \right] \leq 4 \cdot (D + 2) \cdot (Q_H + 1) \cdot \lambda.$$

3.3 Hardness Assumptions

The security of our scheme relies on the following variant of the Module-Learning With Errors (M-LWE) [56, 70].

Definition 3.2 (M-LWE $_{q,n,m,\chi}$). The decisional *Module-Learning With Errors* problem (in its Hermite normal form) with parameters $n, m > 0$ and an error distribution χ over \mathcal{R}_q is defined via the game **M-LWE $_{q,n,m,\chi}^b$** depicted in Figure 1. Here, **M-LWE $_{q,n,m,\chi}^b$** is parameterised by a bit b . We define A 's advantage in **M-LWE $_{q,n,m,\chi}^b$** as

$$\text{Adv}_{q,n,m,\chi}^{\text{M-LWE}}(A) := \left| \begin{array}{l} \Pr[\text{M-LWE}_{q,n,m,\chi}^{0,A} \Rightarrow 1] \\ - \Pr[\text{M-LWE}_{q,n,m,\chi}^{1,A} \Rightarrow 1] \end{array} \right|,$$

and say that **M-LWE $_{q,n,m,\chi}$** is ϵ -hard for all adversaries A satisfying $\text{Adv}_{q,n,m,\chi}^{\text{M-LWE}}(A) \leq \epsilon$.

Game M-LWE$_{q,n,m,\chi}^b$	Oracle RoR(b)	/ Once
01 $b' \stackrel{\$}{\leftarrow} A^{\text{RoR}(b)}$	03 if $b = 0$:	
02 return $[[b = b']]$	04 $A \stackrel{\$}{\leftarrow} \mathcal{R}_q^{n \times m}$	
	05 $\vec{s} \stackrel{\$}{\leftarrow} \chi^m$	
	06 $\vec{e} \stackrel{\$}{\leftarrow} \chi^n$	
	07 return $(A, A\vec{s} + \vec{e})$	
	08 elseif $b = 1$:	
	09 $A \stackrel{\$}{\leftarrow} \mathcal{R}_q^{n \times m}$	
	10 $\vec{u} \stackrel{\$}{\leftarrow} \mathcal{R}_q^n$	
	11 return (A, \vec{u})	

Fig. 1. Game defining **M-LWE $_{q,n,m,\chi}^b$** with adversary A .

Theoretic treatments of LWE-based schemes typically consider the modulus to be polynomial in n and χ to be the discrete Gaussian on $D_{\mathbb{Z}, \alpha \cdot q}$ over \mathbb{Z} with mean 0 and standard deviation $\sigma = \alpha \cdot q / \sqrt{2\pi}$ for some $\alpha < 1$. For these choices the work of [24, 70] showed that if $\alpha q > 2\sqrt{n}$ then worst-case GapSVP- $\tilde{O}(n/\alpha)$ reduces to average-case LWE. As such, many early implementations sampled from a discrete Gaussian distribution, which turns out to be either fairly inefficient [22] or vulnerable to timing attacks [27, 39, 67]. Furthermore, the performance of the best known attacks against LWE-based schemes does not depend on the exact distribution of noise, but rather on the standard deviation (and potentially the entropy). This motivates the use of noise distributions that we can easily, efficiently, and securely sample from. One example is the centred binomial distribution used by CRYSTALS-Kyber [72] and in [5].

4 DEFINITIONS

In this section we present a formal definition of a non-interactive key exchange along with its security notions. A precise definition of non-interactive zero-knowledge proofs can be found in Appendix A.1.

4.1 Non-Interactive Key Exchange

Following the work of [28, 41], we formally define a non-interactive key exchange (NIKE). Through the use of IDs, the security model proposed in [28] abstracts away all considerations concerning certification and public-key infrastructure.

Definition 4.1 (Non-Interactive Key Exchange). A non-interactive key exchange NIKE is defined as a tuple $\text{NIKE} := (\text{Stp}, \text{Gen}, \text{SdK})$ of the following PPT algorithms. Furthermore, we define an identity space \mathcal{IDS} and a shared key space \mathcal{SKS} .

$par \xleftarrow{\$} \text{Stp}(1^\lambda)$: Given the security parameter 1^λ (encoded in unary), the probabilistic setup algorithm returns a set of system parameters par .

$(sk, pk) \xleftarrow{\$} \text{Gen}(\text{ID})$: Given an identity $\text{ID} \in \mathcal{IDS}$, the probabilistic key generation algorithm Gen returns a secret/public key pair (sk, pk) .

$k \leftarrow \text{SdK}(\text{ID}_1, pk_1, \text{ID}_2, sk_2)$: Given an identity $\text{ID}_1 \in \mathcal{IDS}$ and its corresponding public key pk_1 along with another identity $\text{ID}_2 \in \mathcal{IDS}$ and its corresponding secret key sk_2 , the deterministic shared key establishment algorithm SdK returns a shared-key $k \in \mathcal{SKS}$, or a failure symbol \perp . We assume that SdK always returns \perp if $\text{ID}_1 = \text{ID}_2$.

Correctness. Informally, *honest correctness* states that shared keys derived by two honest parties should be the same with overwhelming probability. Although our subsequent definition of correctness implies honest correctness, we state both definitions here for completeness.

Definition 4.2 (Honest Correctness). A non-interactive key exchange $\text{NIKE} := (\text{Stp}, \text{Gen}, \text{Sdk})$ has *correctness error* δ (or is said to be δ -correct), if for all $par \in \text{Stp}(1^\lambda)$ and $\text{ID}_1, \text{ID}_2 \in \mathcal{IDS}$ it holds that,

$$\Pr \left[\text{Sdk}(\text{ID}_1, pk_1, \text{ID}_2, sk_2) \neq \text{Sdk}(\text{ID}_2, pk_2, \text{ID}_1, sk_1) \mid \begin{array}{l} (sk_1, pk_1) \xleftarrow{\$} \text{Gen}(\text{ID}_1) \\ (sk_2, pk_2) \xleftarrow{\$} \text{Gen}(\text{ID}_2) \end{array} \right] \leq \delta,$$

where the probability is taken over the random choices of Stp and Gen .

In this work we define a stronger notion, *semi-malicious correctness* that captures the property that two maliciously chosen key pairs (that are in the support of the key-generation algorithm) will not cause the key exchange to fail. Since this property clearly implies honest correctness, throughout the rest of this work we only focus on semi-malicious correctness. We formalise *semi-malicious correctness* for NIKE relative to a random oracle H via the game $\text{SM-COR}_{\text{NIKE}}$ depicted in Figure 2 and define the advantage of an adversary A in $\text{SM-COR}_{\text{NIKE}}$ as

$$\text{Adv}_{\text{NIKE}, par}^{\text{SM-COR}}(A) := \Pr[\text{SM-COR}_{\text{NIKE}}^A \Rightarrow 1].$$

Definition 4.3 (Semi-malicious Correctness). Let $\text{NIKE} := (\text{Stp}, \text{Gen}, \text{Sdk})$ be a non-interactive key exchange. In the quantum random oracle model, we say that NIKE is $\delta(Q_H, D)$ - SM-COR if for all $\text{ID}_1, \text{ID}_2 \in \mathcal{IDS}$ and for all (possibly unbounded) adversaries A of depth at most D , making at most Q_H queries (possibly in superposition) to the random oracle H , we have $\text{Adv}_{\text{NIKE}, par}^{\text{SM-COR}}(A) \leq \delta(Q_H, D)$.¹

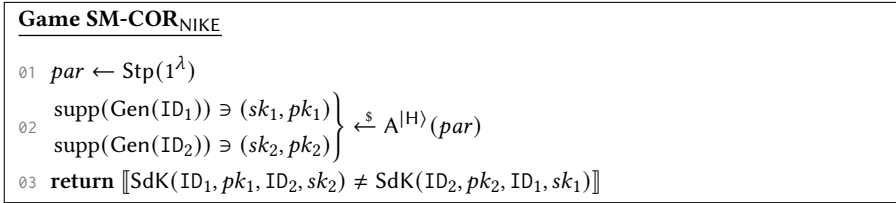


Fig. 2. Correctness game $\text{SM-COR}_{\text{NIKE}}$ for a non-interactive key exchange NIKE defined relative to a random oracle H with adversary A .

Passive Security. We formalise the notion of key indistinguishability with *passive security* for a non-interactive key exchange NIKE , with respect to system parameters $par \in \text{Stp}(1^\lambda)$ via

¹Note that in the standard model our correctness definition can be considered a special case where the number of random oracle queries is zero and hence $\delta(Q_H, D)$ is a constant.

the game $\mathbf{PasSec}_{\text{NIKE},par}^b$ depicted in Figure 3. In $\mathbf{PasSec}_{\text{NIKE},par}^b$, the adversary A provides two identities ID_1 and ID_2 for which the public and secret keys are derived honestly. Given both public keys, A has to distinguish the shared key from a random key. We define the advantage of adversary A in $\mathbf{PasSec}_{\text{NIKE},par}^b$ as

$$\text{Adv}_{\text{NIKE},par}^{\mathbf{PasSec}}(A) := \left| \begin{array}{l} \Pr[\mathbf{PasSec}_{\text{NIKE},par}^{0,A} \Rightarrow 1] \\ - \Pr[\mathbf{PasSec}_{\text{NIKE},par}^{1,A} \Rightarrow 1] \end{array} \right|.$$

Definition 4.4 (Passive Security). Let $\text{NIKE} := (\text{Stp}, \text{Gen}, \text{SdK})$ be a non-interactive key exchange. We say that NIKE is (ϵ, Q_H) - \mathbf{PasSec} relative to $par \in \text{Stp}(1^\lambda)$ if for all $\text{ID}_1, \text{ID}_2 \in \mathcal{ID}\mathcal{S}$ and for all PPT adversaries A , making at most Q_H queries (possibly in superposition) to the random oracle H , we have $\text{Adv}_{\text{NIKE},par}^{\mathbf{PasSec}}(A) \leq \epsilon(Q_H)$.

Game $\mathbf{PasSec}_{\text{NIKE},par}^b$

```

01  $(sk_1, pk_1) \xleftarrow{\$} \text{Gen}(\text{ID}_1)$ 
02  $(sk_2, pk_2) \xleftarrow{\$} \text{Gen}(\text{ID}_2)$ 
03  $k_0 := \text{SdK}(\text{ID}_1, pk_1, \text{ID}_2, sk_2)$ 
04  $k_1 \xleftarrow{\$} \mathcal{SKS}$ 
05  $b' \leftarrow A^{|\mathcal{H}}(pk_1, pk_2, k_b)$ 
06 return  $\llbracket b = b' \rrbracket$ 

```

Fig. 3. Passive security game $\mathbf{PasSec}_{\text{NIKE},par}^b$ for a non-interactive key exchange NIKE defined relative to a random oracle H with adversary A .

Active Security. We formalise the notion of key indistinguishability with *active security* for a non-interactive key exchange NIKE , with respect to system parameters $par \in \text{Stp}(1^\lambda)$ via the game $\mathbf{ActSec}_{\text{NIKE},par}^b$ depicted in Figure 4. Observe that the \mathbf{ActSec} notion defined here corresponds to *CKS-light* which is polynomially equivalent to *CKS* and *m-CKS-heavy* in the work of [41]. The original *CKS* notion was defined in [28]. Unsurprisingly our definition of active security implies the former notion of passive security. The game starts by selecting a bit b uniformly at random after which the adversary A is given access to four oracles. A 's queries may be made adaptively and are arbitrary in number. The RegHonUtr and RegCorUtr oracles let A register honest and corrupted user public keys, respectively. A may make multiple queries to RegCorUtr , in which case only the most recent (*corrupt*, ID , \perp , pk) entry is kept. The RevCorQue oracle provides A with a shared key between a pair of registered identities, subject only to the restriction that at least one of the two identities was registered as honest. Depending on the bit b , the TestQue

oracle returns either a random key or a shared key between two identities registered as honest. Finally, the adversary outputs a guess bit b' and wins the game if and only if $b = b'$. We define the advantage of adversary A in $\text{ActSec}_{\text{NIKE},par}^b$ as

$$\text{Adv}_{\text{NIKE},par}^{\text{ActSec}}(A) := \left| \begin{array}{l} \Pr[\text{ActSec}_{\text{NIKE},par}^{0,A} \Rightarrow 1] \\ - \Pr[\text{ActSec}_{\text{NIKE},par}^{1,A} \Rightarrow 1] \end{array} \right|.$$

Definition 4.5 (Active Security [28]). Let $\text{NIKE} := (\text{Stp}, \text{Gen}, \text{SdK})$ be a non-interactive key exchange. We say that NIKE is $(\epsilon, Q_H, Q_{\text{RHU}}, Q_{\text{RCU}}, Q_{\text{RCQ}}, Q_{\text{TQ}})$ - ActSec relative to $par \in \text{Stp}(1^\lambda)$ if for all PPT adversaries A making at most Q_H queries (possibly in superposition) to the random oracle H , Q_{RHU} queries to RegHonUsr , Q_{RCU} queries to RegCorUsr , Q_{RCQ} queries to RevCorQue , and Q_{TQ} queries to TestQue , we have $\text{Adv}_{\text{NIKE},par}^{\text{ActSec}}(A) \leq \epsilon$.

Game $\text{ActSec}_{\text{NIKE},par}^b$	Oracle $\text{RevCorQue}(\text{ID}_1, \text{ID}_2)$
01 $\mathcal{D} := \perp$	14 if $(\text{honest}, \text{ID}_1, \cdot, \cdot) \in \mathcal{D} \wedge (\text{corrupt}, \text{ID}_2, \cdot, \cdot) \in \mathcal{D} :$
02 $\mathcal{K} := \perp$	15 return $\text{SdK}(\text{ID}_2, pk_2, \text{ID}_1, sk_1)$
03 $b' \leftarrow A^{[H], \text{RegHonUsr}(\cdot), \text{RegCorUsr}(\cdot), \text{RevCorQue}(\cdot), \text{TestQue}(\cdot)}$	16 elseif $(\text{corrupt}, \text{ID}_1, \cdot, \cdot) \in \mathcal{D} \wedge (\text{honest}, \text{ID}_2, \cdot, \cdot) \in \mathcal{D} :$
04 return $\llbracket b = b' \rrbracket$	17 return $\text{SdK}(\text{ID}_1, pk_1, \text{ID}_2, sk_2)$
Oracle $\text{RegHonUsr}(\text{ID} \in \mathcal{IDS})$ / Twice in CKS-light	Oracle $\text{TestQue}(\text{ID}_1, \text{ID}_2)$ / Once in CKS-light
05 if $(\text{corrupt}, \text{ID}, \perp, \cdot) \in \mathcal{D} :$	18 if $\text{ID}_1 = \text{ID}_2 :$
06 return \perp	19 return \perp
07 $(sk, pk) \xleftarrow{\$} \text{Gen}(\text{ID})$	20 if $(\text{honest}, \text{ID}_1, \cdot, \cdot) \in \mathcal{D} \wedge (\text{honest}, \text{ID}_2, \cdot, \cdot) \in \mathcal{D} :$
08 $\mathcal{D} \cup \{(\text{honest}, \text{ID}, sk, pk)\}$	21 if $b = 0 :$
09 return pk	22 $k := \text{SdK}(\text{ID}_1, pk_1, \text{ID}_2, sk_2)$
Oracle $\text{RegCorUsr}(\text{ID} \in \mathcal{IDS}, pk)$	23 if $b = 1 :$
10 if $(\text{corrupt}, \text{ID}, \perp, \cdot) \in \mathcal{D} :$	24 if $(\text{ID}_1, \text{ID}_2, k) \in \mathcal{K} \vee (\text{ID}_2, \text{ID}_1, k) \in \mathcal{K} :$
11 $(\text{corrupt}, \text{ID}, \perp, \cdot) := (\text{corrupt}, \text{ID}, \perp, pk)$	25 return k
12 else :	26 $k \xleftarrow{\$} \mathcal{SKS}$
13 $\mathcal{D} \cup \{(\text{corrupt}, \text{ID}, \perp, pk)\}$	27 $\mathcal{K} \cup \{(\text{ID}_1, \text{ID}_2, k)\}$
	28 return k
	29 return \perp

Fig. 4. Game defining $\text{ActSec}_{\text{NIKE},par}^b$ for a non-interactive key exchange NIKE with adversary A .

5 CONSTRUCTION

We present our NIKE construction in two steps by introducing a scheme that only satisfies passive security followed by a generic transformation that turns it into a scheme with active security.

Stp(1^λ)	SdK(ID_1, pk_1, ID_2, sk_2)	Rec(k)
01 $\mathcal{R}_q := \mathbb{Z}_q[X]/(X^d + 1)$	12 if $ID_1 \leq ID_2$:	24 for $i \in \{0, \dots, d-1\}$:
02 $A \xleftarrow{\$} \text{GL}(N, \mathcal{R}_q)$	13 $r := H(ID_1, pk_1, ID_2, pk_2) \in \mathcal{R}_q$	25 $k_i := \text{Rnd}(k_i) \in \{0, 1\}^d$
03 $par := (q, d, \mathcal{R}_q, N, A)$	14 parse $pk_1 \rightarrow (pk_L, \perp) =: \bar{u}_L^\top \in \mathcal{R}_q^{1 \times N}$	26 return $k \in \{0, 1\}^d$
04 return par	15 parse $sk_2 \rightarrow (\perp, sk_R) =: \bar{s}_R \in \mathcal{R}_q^N$	27 if $\frac{q}{4} \leq k_i \leq \frac{3q}{4}$:
Gen(ID)	16 $k' := \bar{u}_L^\top \bar{s}_R + r \in \mathcal{R}_q$	28 return 1
05 $\bar{s}_L, \bar{s}_R \leftarrow \text{Cbd}(\cdot)$ / Samples $\bar{s} \in \mathcal{R}_q^N$ from \mathcal{X}^N	17 else :	29 return 0
06 $\bar{e}_L, \bar{e}_R \leftarrow \text{Cbd}(\cdot)$ / Samples $\bar{e} \in \mathcal{R}_q^N$ from \mathcal{X}^N	18 $r := H(ID_2, pk_2, ID_1, pk_1) \in \mathcal{R}_q$	31 for $i \in \{1, \dots, N\}$:
07 $sk_L := \bar{s}_L^\top \in \mathcal{R}_q^{1 \times N}$	19 parse $pk_1 \rightarrow (\perp, pk_R) =: \bar{u}_R \in \mathcal{R}_q^N$	32 for $j \in \{0, \dots, d-1\}$:
08 $sk_R := \bar{s}_R \in \mathcal{R}_q^N$	20 parse $sk_2 \rightarrow (sk_L, \perp) =: \bar{s}_R^\top \in \mathcal{R}_q^{1 \times N}$	33 $a, b \xleftarrow{\$} \{0, 1\}$
09 $pk_L := \bar{s}_L^\top A + \bar{e}_L^\top \in \mathcal{R}_q^{1 \times N}$	21 $k' := \bar{s}_R^\top \bar{u}_R + r \in \mathcal{R}_q$	34 $f_j := a - b$
10 $pk_R := A \bar{s}_R + \bar{e}_R \in \mathcal{R}_q^N$	22 $k := \text{Rec}(k') \in \{0, 1\}^d$	35 $f_i := \sum_{j=0}^{d-1} f_j X^j$
11 return $(sk_{ID} := (sk_L, sk_R), pk_{ID} := (pk_L, pk_R))$	23 return k	36 return $\vec{f} := (f_1, \dots, f_N)$

Fig. 5. Construction of passively secure non-interactive key exchange NIKE := (Stp, Gen, SdK) with functions Rec : $\mathcal{R}_q \rightarrow \{0, 1\}^d$, Rnd : $\mathbb{Z}_q \rightarrow \{0, 1\}$ and Cbd : $\emptyset \rightarrow \mathcal{R}_q^N$, and random oracle $H : \mathcal{IDS} \times (\mathcal{R}_q^{1 \times N} \times \mathcal{R}_q^N) \times \mathcal{IDS} \times (\mathcal{R}_q^{1 \times N} \times \mathcal{R}_q^N) \rightarrow \mathcal{R}_q$. Here $\text{GL}(N, \mathcal{R}_q)$ denotes the set of invertible matrices over \mathcal{R}_q .

5.1 Passive Setting

In this section we present our construction of a non-interactive key exchange with semi-malicious correctness that satisfies key indistinguishability for honestly registered public keys (passive security) in the random-oracle model. The scheme is depicted in Figure 5.

Correctness. In order to achieve better bounds in our proof of security, we show that our scheme satisfies both honest correctness as well as the stronger notion of semi-malicious correctness of Definition 4.2 and Definition 4.3, respectively. Although Theorem 5.1 implies Lemma 1, we will use the latter and state its proof in Appendix B for sake of completeness.

LEMMA 1 (HONEST CORRECTNESS). For all (possibly unbounded) adversaries A the non-interactive key exchange NIKE := (Stp, Gen, SdK) construction depicted in Figure 5 has *honest correctness error*

$$\delta \leq \frac{4\beta^2 d^2 N}{q}$$

as per Definition 4.2.

We show that the scheme satisfies the stronger notion of semi-malicious correctness in the quantum random-oracle model.

THEOREM 5.1 (SM-COR OF NIKE). For all (possibly unbounded) adversaries A of depth D making at most Q_H queries (possibly in superposition) to the random oracle H , the non-interactive key exchange NIKE := (Stp, Gen, SdK) construction depicted in Figure 5 has *semi-malicious correctness error*

$$\delta(Q_H, D) \leq 16 \cdot (D + 2) \cdot (Q_H + 1) \cdot \frac{\beta^2 d^2 N}{q}$$

as per Definition 4.3, where β is a bound on the maximum absolute value of the support of χ .

PROOF. We are going to prove that the adversary cannot cause an error in the key-derivation, i.e., a mismatch between the derived keys, even if he is allowed to choose both secret keys from the support of the key generation algorithm. This trivially implies semi-malicious correctness. Let (sk_1, pk_1) and (sk_2, pk_2) be the pairs returned by the adversary. Without loss of generality we can consider $sk_1 = sk_L$ and $pk_2 = pk_R$, i.e., only “one side” of the key. A key mismatch occurs whenever

$$\begin{aligned} & \text{Rec}(pk_L^\top sk_R + \mathbf{r}) \neq \text{Rec}(sk_L^\top pk_R + \mathbf{r}) \\ & \text{Rec}((\vec{s}_L^\top A + \vec{e}_L^\top) \vec{s}_R + \mathbf{r}) \neq \text{Rec}(\vec{s}_L^\top (A\vec{s}_R + \vec{e}_R) + \mathbf{r}) \\ & \text{Rec}\left(\underbrace{\vec{s}_L^\top A\vec{s}_R + \mathbf{r} + \vec{e}_L^\top \vec{s}_R}_{\mathbf{k}^* \in \mathcal{R}_q}\right) \neq \text{Rec}\left(\underbrace{\vec{s}_L^\top A\vec{s}_R + \mathbf{r} + \vec{s}_L^\top \vec{e}_R}_{\mathbf{k}^* \in \mathcal{R}_q}\right), \end{aligned}$$

where \mathbf{r} is the output of the random oracle on both public keys and \vec{e}_L and \vec{e}_R are sampled from the noise distribution χ^N . By definition of the Rec function, this means that the term $\vec{e}_L^\top \vec{s}_R$ (or, equivalently, the term $\vec{s}_L^\top \vec{e}_R$) is causing a rounding error on one of the coefficients of \mathbf{k}^* . We now bound the size of the largest coefficient of $\vec{e}_L^\top \vec{s}_R$ as

$$\begin{aligned} \|\vec{e}_L^\top \vec{s}_R\|_\infty &= \left\| \sum_{i=1}^N e_{L,i} s_{R,i} \right\|_\infty \\ &\leq \sum_{i=1}^N \|e_{L,i} s_{R,i}\|_\infty \\ &\leq \beta^2 dN, \end{aligned}$$

where the first inequality follows from the triangle inequality. The norm of $\vec{s}_L^\top \vec{e}_R$ can be bounded similarly. It follows that, in order for a key-derivation error to occur, at least one coefficient of \mathbf{k}^* must be in the following interval

$$S^* = \left[\frac{q}{4} \pm \beta^2 dN \right] \cup \left[\frac{3q}{4} \pm \beta^2 dN \right].$$

Next we define a function F that, on input two public keys and two identities samples a uniform \mathbf{r} , it returns 1 if a key mismatch occurs, i.e.,

$$\text{Rec}(pk_L^\top sk_R + \mathbf{r}) \neq \text{Rec}(sk_L^\top pk_R + \mathbf{r})$$

and 0 otherwise. The function checks this by (inefficiently) recovering the secret keys and comparing the results of the Rec functions (see equation above). Note that, since \mathbf{A} is invertible, the secret key is uniquely determined by the public key, and therefore this (inefficient) function is well defined on all inputs. Furthermore, note that the element

$$\mathbf{k}^\star = sk_L^\top \mathbf{A} sk_R + \mathbf{r}$$

is uniformly distributed in \mathcal{R}_q , since $\mathbf{r} \leftarrow \mathcal{R}_q$. It follows that for any given input x :

$$\Pr[F(x) = 1] \leq \frac{4\beta^2 d^2 N}{q}.$$

Finally, observe that by definition a key mismatch happens if and only if the function F outputs 1 and consequently the adversary is able to find such accepting input. By Lemma 3.1, this happens with probability at most $16 \cdot (D + 2) \cdot (Q_H + 1) \cdot \beta^2 d^2 N / q$ for an adversary of depth D , making at most Q_H quantum query to the random oracle. ■

On the Need for Random Oracles. An astute reader may wonder whether the usage of the random oracle is needed at all to prove the above notion of correctness, since there does not appear to be an immediate attack even if we omit the random oracle completely from the scheme. It is plausible to conjecture that semi-malicious correctness holds even without the random oracle. Informally, semi-malicious correctness boils down to showing that, for a given public key $pk \in \mathcal{R}_q^N$, it is hard to find an $s \in \mathcal{R}_q^N$ such that for no coefficient of the product $s^\top pk$ lies in the interval S^\star . Thus, the a bound in these settings would require one to estimate the hardness of this version of the (inhomogenous) 1-dimensional short-integer-solution (SIS) problem. By relying on the random-oracle heuristic, we are able to bypass this problem and obtain a construction in the QROM that is: (i) unconditionally correct in *any ring* and (ii) whose security is based on the well-established M-LWE problem. We leave the precise study of the hardness of this 1-dimensional variant of the SIS problem as ground for future work.

Passive Security. Assuming the hardness of M-LWE, Definition 3.2, we show that the scheme satisfies *passive security*, Definition 4.4, in the QROM.

THEOREM 5.2 (PASSIVE SECURITY). For any PPT adversary \mathbf{A} against NIKE $:= (\text{Stp}, \text{Gen}, \text{SdK})$, depicted in Figure 7, making at most Q_H queries (possibly in superposition) to \mathbf{H} , there exist PPT

adversaries B_1, B_2 such that

$$\text{Adv}_{\text{NIKE}, \text{par}}^{\text{PasSec}}(A) \leq 6 \cdot \text{Adv}_{q, N, N, \chi}^{\text{M-LWE}}(B_1) + 2 \cdot \text{Adv}_{q, N, N+1, \chi}^{\text{M-LWE}}(B_2) + \frac{4\beta d}{q}.$$

PROOF OF THEOREM 5.2. Let A be an adversary against NIKE in the **PasSec** game. Consider the sequence of games in Figure 6.

Game PasSec _{NIKE, par} ^b	
01 $(sk_1, pk_1) \xleftarrow{\$} \text{Gen}(\text{ID}_1)$	
02 $pk_1 \xleftarrow{\$} \mathcal{R}_q^{1 \times N} \times \mathcal{R}_q^N$	/ G ₁
03 $(sk_1, pk_1) \xleftarrow{\$} \text{Gen}(\text{ID}_1)$	/ G ₄
04 $(sk_2, pk_2) \xleftarrow{\$} \text{Gen}(\text{ID}_2)$	
05 $pk_2 \xleftarrow{\$} \mathcal{R}_q^{1 \times N} \times \mathcal{R}_q^N$	/ G ₃
06 $(sk_2, pk_2) \xleftarrow{\$} \text{Gen}(\text{ID}_2)$	/ G ₄
07 if $\text{ID}_1 \leq \text{ID}_2$:	
08 $r := \text{H}(\text{ID}_1, pk_1, \text{ID}_2, pk_2) \in \mathcal{R}_q$	
09 parse $pk_1 \rightarrow (pk_L, \perp) =: \vec{u}_L^T \in \mathcal{R}_q^{1 \times N}$	
10 parse $sk_2 \rightarrow (\perp, sk_R) =: \vec{s}_R \in \mathcal{R}_q^N$	
11 $k' := \vec{u}_L^T \vec{s}_R + r \in \mathcal{R}_q$	
12 else :	
13 $r := \text{H}(\text{ID}_2, pk_2, \text{ID}_1, pk_1) \in \mathcal{R}_q$	
14 parse $pk_1 \rightarrow (\perp, pk_R) =: \vec{u}_R \in \mathcal{R}_q^N$	
15 parse $sk_2 \rightarrow (sk_L, \perp) =: \vec{s}_R^T \in \mathcal{R}_q^{1 \times N}$	
16 $k' := \vec{s}_R^T \vec{u}_R + r \in \mathcal{R}_q$	
17 $k_0 := \text{Rec}(k') \in \{0, 1\}^d$	/ G ₀
18 $e \xleftarrow{\$} \chi$	/ G ₂
19 $k_0 := \text{Rec}(k' + e) \in \{0, 1\}^d$	/ G ₂
20 $u \xleftarrow{\$} \mathcal{R}_q$	/ G ₃
21 $k_0 := \text{Rec}(u) \in \{0, 1\}^d$	/ G ₃
22 $k_1 \xleftarrow{\$} \text{SKS}$	
23 $b' \leftarrow A^{(H)}(pk_1, pk_2, k_b)$	
24 return $\llbracket b = b' \rrbracket$	

Fig. 6. Games G₀, G₁, G₂, G₃, G₄ for the proof of **PasSec** of NIKE in Figure 5.

Game G₀. This is the original $\text{PasSec}_{\text{NIKE},par}^b$ game so by definition

$$\text{Adv}_{\text{NIKE},par}^{\text{PasSec}}(\mathcal{A}) \leq \left| \Pr \left[\mathbf{G}_0^{\mathbf{A}} \Rightarrow 1 \right] - \frac{1}{2} \right|.$$

Game G₁. In this game the half of pk_1 that is used in the key-derivation is replaced with a uniform key. Without loss of generality we can consider either half of the key. Indistinguishability follows from a reduction against the M-LWE problem, conditioned on the matrix \mathbf{A} being invertible. Since this happens with probability at least $1/2$, we have that

$$\left| \Pr \left[\mathbf{G}_0^{\mathbf{A}} \Rightarrow 1 \right] - \Pr \left[\mathbf{G}_1^{\mathbf{A}} \Rightarrow 1 \right] \right| \leq 2 \cdot \text{Adv}_{q,N,N,\chi}^{\text{M-LWE}}(\mathcal{B}_1).$$

Game G₂. In this hybrid we modify the way we compute the shared key. Consider \mathbf{k}' as computed in the Sdk algorithm, we define the shared key as $\text{Rec}(\mathbf{k}' + \mathbf{e})$ where $\mathbf{e} \xleftarrow{\$} \chi$ is a freshly sampled ring element from the noise distribution. Note that the adversary can only detect a change in this hybrid if

$$\text{Rec}(\mathbf{k}' + \mathbf{e}) \neq \text{Rec}(\mathbf{k}).$$

Since \mathbf{k}' is uniformly sampled from \mathcal{R}_q , the probability that any coefficient is rounded to a different term is at most $4\beta d/q$, which is also an upper bound on the distinguishing advantage of the adversary. Thus we get

$$\left| \Pr \left[\mathbf{G}_1^{\mathbf{A}} \Rightarrow 1 \right] - \Pr \left[\mathbf{G}_2^{\mathbf{A}} \Rightarrow 1 \right] \right| \leq \frac{4\beta d}{q}.$$

Game G₃. In this game the half of pk_2 used in the key-derivation is replaced with a uniform key, along with $\mathbf{k}' + \mathbf{e}$ that is replaced with a uniform ring element \mathbf{u} . By another invocation of the M-LWE assumption, again conditioning on \mathbf{A} being invertible, we have that

$$\left| \Pr \left[\mathbf{G}_2^{\mathbf{A}} \Rightarrow 1 \right] - \Pr \left[\mathbf{G}_3^{\mathbf{A}} \Rightarrow 1 \right] \right| \leq 2 \cdot \text{Adv}_{q,N,N+1,\chi}^{\text{M-LWE}}(\mathcal{B}_2).$$

Game G₄. In this game we revert the changes made to pk_1 and pk_2 , and appealing again to Definition 3.2 we get

$$\left| \Pr \left[\mathbf{G}_3^{\mathbf{A}} \Rightarrow 1 \right] - \Pr \left[\mathbf{G}_4^{\mathbf{A}} \Rightarrow 1 \right] \right| \leq 4 \cdot \text{Adv}_{q,N,N,\chi}^{\text{M-LWE}}(\mathcal{B}_1).$$

Observe that k_0 and k_1 are identically distributed and the adversary can only guess b' . Hence,

$$\Pr \left[\mathbf{G}_4^{\mathbf{A}} \Rightarrow 1 \right] = \frac{1}{2}.$$

Collecting all probabilities yields the bound stated in Theorem 5.2. ■

5.2 Active Setting

Here we show how a non-interactive key exchange with passive security can be generically transformed to one with active security. The transformation, depicted in Figure 7, requires a simulation-sound NIZK with a straight-line extractor. The proof is deferred to Appendix B.

<u>Stp(1^λ)</u>	<u>Gen(ID)</u>	<u>SdK(ID₁, pk₁, ID₂, sk₂)</u>
01 $par \xleftarrow{\$} \text{Stp}'(1^\lambda)$	03 $(sk'_{ID}, pk'_{ID}) \xleftarrow{\$} \text{Gen}'(\text{ID})$	08 parse $pk_1 \rightarrow (pk'_1, \pi)$
02 return par	04 $\pi \xleftarrow{\$} \text{ZK.Prv}(pk'_{ID}, sk'_{ID})$	09 if $\text{ZK.Ver}(pk'_1, \pi) = 0$: return \perp
	05 $sk_{ID} := sk'_{ID}$	10 $k' := \text{SdK}'(\text{ID}_1, pk'_1, \text{ID}_2, sk_2)$
	06 $pk_{ID} := (pk'_{ID}, \pi)$	11 return k'
	07 return (sk_{ID}, pk_{ID})	

Fig. 7. Compiler for transforming a passively secure non-interactive key exchange $\text{NIKE}' := (\text{Stp}', \text{Gen}', \text{SdK}')$ with semi-malicious correctness into an actively secure non-interactive key exchange $\text{NIKE} := (\text{Stp}, \text{Gen}, \text{SdK})$.

THEOREM 5.3 (PasSec AND SM-COR OF $\text{NIKE}' \xrightarrow[\text{ZKPoK}]{\text{QROM}} \text{ActSec}$ OF NIKE). Let $H : \{0, 1\}^* \rightarrow \mathcal{R}_q$ be a random oracle and $\text{NIKE}' := (\text{Stp}', \text{Gen}', \text{SdK}')$ a passively secure non-interactive key exchange with semi-malicious correctness defined relative to $par' \in \text{Stp}'(1^\lambda)$. Further, let $\text{ZKPoK} := (\text{ZK.Prv}, \text{ZK.Ver})$ be a simulation-sound online extractable zero-knowledge proof of knowledge for the NP relation $R = (pk_{ID}, sk_{ID})$. Then, for any **ActSec** adversary A against $\text{NIKE} := (\text{Stp}, \text{Gen}, \text{SdK})$, depicted in Figure 7, there exist PPT adversaries $B_1, B_2, B_{3,i}, B'_{3,i}, B_4$ such that

$$\begin{aligned}
 \text{Adv}_{\text{NIKE}, par}^{\text{ActSec}}(A) &\leq Q_{\text{RCU}} \cdot \text{Adv}_{\text{ZKPoK}}^{\text{SSND}}(B_1) + 2 \cdot \text{Adv}_{\text{NIKE}', par'}^{\text{SM-COR}}(B_2) \\
 &\quad + Q_{\text{TQ}} \cdot Q_{\text{RHU}}^2 \cdot \text{Adv}_{\text{NIKE}', par'}^{\text{PasSec}}(B_{3,i}) \\
 &\quad + 2 \cdot Q_{\text{TQ}} \cdot \text{Adv}_{\text{NIKE}', par'}^{\text{SM-COR}}(B'_{3,i}) \\
 &\quad + 2 \cdot Q_{\text{RHU}} \cdot \text{Adv}_{\text{ZKPoK}}^{\text{ZK}}(B_4),
 \end{aligned}$$

where Q_{RCU} and Q_{RHU} , are the number of queries made by A to RegCorUsr and RegHonUsr , respectively, and Q_{TQ} denotes the number of queries made by B_i to TestQue for $i \in \{0, \dots, Q_{\text{TQ}} - 1\}$.

5.3 Practical considerations

Halving the Key Size. Observe that the “left” and “right” components pk_L and pk_R of the public key of the NIKE as specified in Figure 5 are necessary because we work in the non-commutative M-LWE setting. An easy way to halve the size of the public key would be to set $N = 1$, i.e., to

work in the R-LWE setting; this also eliminates the need for the case distinction in Sdk. We argue that for essentially all relevant applications of a NIKE, we can halve the public-key size even *without* moving to the R-LWE setting. All that is required is that protocol participants (and their associated NIKE keys) have different *roles*, typically called initiator and responder or client and server, and that these roles are clear from protocol context. This is certainly the case for the application examples sketched in Section 1.1: The OPTLS handshake, like the TLS handshake, clearly distinguishes the roles of client and server, so does the handshake in (post-quantum) WireGuard. Also in X3DH the critical static-semistatic key exchange has clear roles that can be used to distinguish between the “left” and “right” participant instead of transmitting both halves of the key and using comparison of IDs. Note that this setting of a NIKE using keys with different roles is very similar to the ℓ_A and ℓ_B keys of SIDH [50, § 3.2], when it was still considered as a replacement for DH, i.e., before it was shown to not be actively secure in [44] and completely broken in [29].

Based on these considerations, we stick to the M-LWE setting for the construction of Swoosh; in our performance evaluation in Section 6 we report the size of only one public-key component.

Security of the NIZK. We highlight that our proof of active security, Theorem 5.3, requires the strong property of simulation-sound online-extractability. Although constructions satisfying such a strong notion exist [77], they tend to be less efficient than alternatives satisfying weaker notions of security. For instance, a proof of knowledge of an M-LWE secret satisfying simulation soundness, but without *online*-extractability, using state of the art techniques [59] and appropriate parameters is around 70 KB in size.

It appears likely that the need for the stronger notion is an artefact of the proof, and we conjecture that our construction remains secure even if we use NIZKs that are simulation-sound and extractable, although not online-extractable (such as the protocol in [59]). We are not the first to make this additional assumption, in favour of a more efficient scheme and similar heuristics have already appeared in the literature, e.g., in [31]. While we cannot exclude that contrived examples of NIZKs could make our compiler fail, we believe that all natural candidates of NIZKs would lead to secure schemes.

Tangentially, we also mention that for some applications, the performance of the NIZK does not affect the efficiency of the shared-key computation, since it can be verified once and for all for a given public key: In any scenario where the public keys are distributed by some PKI, the NIZK proof can be simply verified by the PKI upon the registration of the key, and then immediately discarded. The users would then trust the PKI to have verified the NIZK on their behalf. Note that this does not introduce any extra trust assumption, since the PKI is anyway trusted to provide the correct public key. In these scenarios, the efficiency of the NIZK only marginally impacts

the overall system performance, and thus justifies ignoring the costs of the NIZK for shared-key computation.

5.4 Parameter selection

Selecting parameters for the scheme influences several aspects, most notably the correctness error and the hardness of M-LWE. In order to evaluate the security of our scheme we use the *Lattice-Estimator* [2, 4, 68], to estimate the memory and CPU operations required to perform various lattice attacks, including dual attacks, uSVP, the Coded-BKW attack, and solving using Gröbner bases with the Arora-Ge attack. The estimator has been used to estimate the concrete security for all LWE and NTRU based candidates of the NIST competition [3], and is regularly updated to include the latest developments in lattice cryptanalysis². However, we also take into account practical considerations for the implementation when selecting our parameters, such as the use of ternary secrets and noise sampled from a centred binomial distribution. For our scheme with parameters $n = 8192$, $q = 2^{214} - 255$ and X a ternary distribution, we estimate the hardness of the M-LWE problem underlying Swoosh at 120 bits³.

The other way to attack Swoosh is, for an active attacker, to try to produce failures. We consider a quantum attacker with a bounded query depth of $D = 2^{64}$ (i.e., what NIST considers to be “the approximate number of gates that current classical computing architectures can perform serially in a decade” [63, Sec. 4.A]) and a bound on the number of queries of 2^{120} (i.e., matching the hardness of the underlying lattice problem). Applying Theorem 5.1 yields a success probability (correctness error), after this amount of computation, of

$$16 \cdot \left(2^{64} + 2\right) \cdot \left(2^{120} + 1\right) \cdot \frac{256^2 \cdot 32}{2^{214}} < \frac{1}{2^4} = \delta(Q_H, D),$$

i.e., considerably smaller than 1/2. Note that this analysis is conservative as it ignores the circuit depth for the Grover oracle that an attacker would need to implement.

Generating an Invertible Matrix. In order to justify our conservative estimate that at least 50% of all matrices in $\mathcal{R}_q^{N \times N}$ are invertible, we used Sage to generate 2000 random matrices and checked if they are invertible. They were all invertible. We additionally verified that the concrete matrix used by our implementation (see Section 6.1) is invertible.

In order to demonstrate the practicality of Swoosh in terms of performance, we implement the core part of the scheme, Passive-Swoosh, present benchmarks of this implementation, and compare to other KEMs and (pre- and post-quantum) NIKEs. We caution the reader that all implementation details and numbers we present in this section are for Passive-Swoosh only. To

²An up-to-date list of implemented works can be found <https://lattice-estimator.readthedocs.io/en/latest/references.html>.

³These numbers can be reproduced with the estimator — the version used in this work is at commit 96875622c6b0e6f98a91ddecaaa17b66dbc5a87.

Parameter	Description	Value
β	upper bound on $\ \vec{s}\ _\infty = \ \vec{e}\ _\infty$	1
q	prime modulus	$2^{214} - 255$
d	dim of $\mathcal{R}_q := \mathbb{Z}_q[X]/(X^d + 1)$	256
l	# factors $X^d + 1$ splits into mod q	128
N	height of the A matrix	32
n	lattice dimension	8192
χ	noise distribution	$p(-1) = 25\%$ $p(0) = 50\%$ $p(1) = 25\%$

Table 1. Parameter selection for non-interactive key exchange NIKE.

obtain a full picture of the performance of Swoosh, the implementation will need to be augmented with a future implementation of the NIZKP from [59]. As outlined in Section 5.3, the performance impact of adding the NIZKP in terms of both size and computational effort depends on the concrete application scenario and may be negligible if key-generation performance is not critical and if NIZKP verification can be outsourced to the PKI. The source code of our implementation is publicly available⁴.

6.1 Implementation

Scheme (variant)	Assumption	Non-int.	PQ	Sizes (in bytes)	
				Ciphertext	Public Key
CRYSTALS-Kyber [72] (Kyber-512)	M-LWE	✗	✓	768	800
Classic McEliece [1] (mceliece348864)	Binary Goppa Codes	✗	✓	96	261120
X25519 [14]	DLOG	✓	✗	–	32
CTIDH [11] (CTIDH-1024)	Supersingular Isogenies	✓	✓	–	128
Passive-Swoosh (this work)	M-LWE	✓	✓	–	221184

Table 2. Public-key sizes for select NIKes and public-key and ciphertext sizes of select post-quantum KEMs

As a NIKE, Swoosh is composed of two major functions, the key generation procedure and the shared-key computation, the performance of which dictates the practicality of Swoosh.

In the case of the key generation, the matrix A is fixed and assumed to be in the NTT domain, so performance is dictated by the sampling of the secret and error vectors, as well as the computation of the public key which involves two NTT transformations, and a matrix multiplication followed by a polynomial addition. As for the shared key computation, its performance is mainly dictated by the random offset computation, which requires the use of cSHAKE [51] and the polynomial base multiplication required to calculate k' (see Fig. 5). Similar to other schemes, the shared-key derivation also performs rounding of the shared key, however its execution time is negligible. At

⁴<https://github.com/MQuaresma/pswoosh>, commit ID a580876 at the time of writing.

a high level, the architecture of our implementation is divided into two distinct parts: low-level field arithmetic over \mathbb{F}_q that is implemented using the Jasmin language [6, 7], and polynomial arithmetic in \mathcal{R}_q as well as the scheme itself, both of which are implemented in Rust.

The structure largely mimics the abstract specification in Figure 5. The main difference is that, like other lattice-based schemes [5, 72], we encode and transmit public keys in NTT domain. This massively reduces the number of cycles required for shared-key computation. In addition, as discussed in Section 5.3, we assume that the role of each party is well defined and thus only compute one half of the key. We implement this by passing a Boolean flag as an argument to key generation and shared-key derivation to indicate which party is calling the respective function. Finally, we implement the noise sampling in a slightly different way than one might expect; we will discuss this later in this section.

Zooming in on the low-level field arithmetic, the operations on integers modulo $2^{214} - 255$ require multiple-precision integers since native scalar registers (64 bits in AMD64) are not large enough to store a single field element. This arithmetic is implemented through `libjbn`⁵, a Jasmin library that exposes big-integer arithmetic.

Polynomial Arithmetic. On top of this layer, operations in polynomial rings are implemented using Rust, in addition to other functions such as reconciliation, matrix and noise generation. Similar to other lattice-based schemes, one of the more critical (and easier) operations to optimise (from a performance perspective) is polynomial multiplication. The naive algorithm for multiplying two polynomials in \mathcal{R}_q , sometimes called Schoolbook multiplication, involves multiplying all pairs of coefficients, calculating their sum and reducing modulo $X^d + 1$. However, the complexity of this approach is quadratic in the number of coefficients and thus quite costly.

The *Number Theoretic Transform* (NTT) provides a more efficient approach for polynomial multiplication with quasi-logarithmic time complexity $\mathcal{O}(d \log(d))$ instead of $\mathcal{O}(d^2)$. For a detailed discussion on the NTT refer to [74].

As is the case for other implementations [5, 72], we implement an in-place NTT which requires bit-reversal operations in the forward and inverse transforms but uses less memory. Another optimisation is to make the NTT a part of our scheme, which means the matrix A is sampled in the NTT domain, and the secret and public keys are stored in the NTT domain. This results in the NTT only being used three times, once for the shared key derivation and twice in the key generation to convert the secret and error vectors, which are sampled in the normal domain to the NTT domain before computing the public key. A common trick to speed-up the NTT transformation when using Montgomery reduction [61], as is the case for `libjbn`, is the pre-computed constants in Montgomery form $\zeta \cdot R \pmod{q}$.

⁵See <https://github.com/formosa-crypto/libjbn>.

Noise sampling and matrix generation. Both the matrix generation and noise sampling procedures use a seed, either set as a system parameter for A or as a secret input to a PRG in the case of \vec{s} and \vec{e} , to produce a stream bytes from which the distributions are sampled. In the case of matrix generation this is achieved via rejection sampling on the stream of bytes produced by an extendable output function (XOF). The noise sampling procedure, used for generating the secret key and the error vector, samples these vectors from a centred binomial distribution using the output of a PRF with a random seed. As with other schemes where multiplication is optimised using the NTT, the choice of (symmetric) primitive that underlies these functions tends to be a deciding factor for the performance. We chose cSHAKE [51] based on Keccak [38] as the underlying primitive for the XOF and AES256-CTR for the PRF used in noise sampling.

Similar to the NewHope scheme [5], for efficiency reasons the secret and error vectors are sampled from a centred binomial distribution rather than a discrete Gaussian distribution. Using ternary noise means that each coefficient can be generated from only 2 bits and thus, the generation of a polynomial in \mathcal{R}_q only requires $(32 \cdot 256 \cdot 2)/8 = 2048$ (pseudo-random) bytes. Intuitively, our CBD definition in Figure 5 when a and b are sourced from a PRG, maps 00_b and 11_b to $0 \pmod q$ with 50% probability, 10_b to $1 \pmod q$ and 01_b to $-1 \pmod q$ with 25% probability each. Our implementation differs from the specification by applying signed reduction modulo 3 to each two bit block and converting it to a congruent value in \mathbb{F}_q , as opposed to using big integer field arithmetic to map bits a and b to an element in \mathbb{F}_q . Although this approach produces a different mapping (11_b to $-1 \pmod q$, 00_b and 10_b to $0 \pmod q$ and 01_b to $1 \pmod q$), the distribution of the outputs is identical. Due to the size of our field elements, this approach results in a considerable speed up in the noise sampling.

The random offset used in our scheme is generated by performing rejection sampling on the output of cSHAKE-256 [51].

6.2 Performance Evaluation

In this section we evaluate the performance of our scheme and compare it to others. We also provide a comparison of key sizes and the properties of each scheme such as post-quantum security, and whether they are non-interactive.

The benchmark results for Passive-Swoosh were obtained on an Intel Core i7-6500U (Skylake) running on a single core with Hyper-threading and TurboBoost disabled. The Rust compiler version used for the benchmarks was 1.62.1⁶ and the Jasmin compiler version was 2022.09.0. We report the median cycle counts of 10000 runs. In Table 3 we list the results and compare to the cycle counts of CTIDH-1024 as reported in [11, Sec. 8] and of lib25519 [62], on Intel Skylake CPUs.

⁶The following build configuration options/values were used: `opt-level=3` and `target-cpu="native"`.

As expected, the pre-quantum X25519 [14] scheme is orders of magnitude faster than Passive-Swoosh for key generation. However, in many applications of NIKes, keys are re-used many times and what is more critical is the performance of shared-key computation. Here the gap to pre-quantum X25519 is considerably smaller and Passive-Swoosh outperforms the only real post-quantum competitor CTIDH by a factor of 48.

Operation	X25519	CTIDH-1024	Passive-Swoosh
NTT	—	—	217 430
NTT ⁻¹	—	—	262 992
Noise generation	—	—	89 776
Key generation	28 187	469 520 000	146 920 890
Shared key	87 942	511 190 000	10 612 666

Table 3. Cycle counts on Intel Skylake.

However, as shown in Table 2, CTIDH, Kyber, and X25519 have a public-key size several orders of magnitude smaller than Passive-Swoosh. In this aspect, only Classic McEliece has a public key size comparable to that of Passive-Swoosh, even when taking into account the expected size of the proof of knowledge (see Section 5.3).

7 CONCLUSIONS

In this work, we constructed a NIKE based on the M-LWE problem, with a proof in the QROM. Our scheme is based on the standard blueprint, but with an additional twist to guarantee provable security for arbitrary rings. Our optimised implementation shows that our scheme offers reasonable computational performance and key sizes that should be acceptable for most applications. We view our work as the first evidence contradicting the folklore belief that lattice-based NIKE is too inefficient to be used in practice. As future work, we plan an implementation of the full Swoosh scheme, i.e., including the NIZK proof. We also plan to explore applications of our scheme to more complex protocols and to formally verify the correctness of (parts of) our implementation.

REFERENCES

- [1] Martin R. Albrecht, Daniel J. Bernstein, Tung Chou, Carlos Cid, Jan Gilcher, Tanja Lange, Varun Maram, Ingo von Maurich, Rafael Misoczki, Ruben Niederhagen, Kenneth G. Paterson, Edoardo Persichetti, Christiane Peters, Peter Schwabe, Nicolas Sendrier, Jakub Szefer, Cen Jung Tjhai, Martin Tomlinson, and Wen Wang. 2022. *Classic McEliece*. Technical Report. National Institute of Standards and Technology. available at <https://csrc.nist.gov/projects/post-quantum-cryptography/round-4-submissions>.
- [2] Martin R. Albrecht, Benjamin R. Curtis, Amit Deo, Alex Davidson, Rachel Player, Eamonn W. Postlethwaite, Fernando Virdia, and Thomas Wunderer. 2018. Estimate All the LWE, NTRU Schemes!. In *SCN 18: 11th International Conference on Security in Communication Networks (Lecture Notes in Computer Science, Vol. 11035)*, Dario Catalano and Roberto De Prisco (Eds.). Springer, Heidelberg, Germany, Amalfi, Italy, 351–367. https://doi.org/10.1007/978-3-319-98113-0_19

- [3] Martin R. Albrecht, Benjamin R. Curtis, Amit Deo, Alex Davidson, Rachel Player, Eamonn W. Postlethwaite, Fernando Virdia, and Thomas Wunderer. 2018. Estimate all the LWE, NTRU schemes! Cryptology ePrint Archive, Report 2018/331. <https://eprint.iacr.org/2018/331>.
- [4] Martin R. Albrecht, Rachel Player, and Sam Scott. 2015. On The Concrete Hardness Of Learning With Errors. Cryptology ePrint Archive, Report 2015/046. <https://eprint.iacr.org/2015/046>.
- [5] Erdem Alkim, Léoucas, Thomas Pöppelmann, and Peter Schwabe. 2016. Post-quantum Key Exchange - A New Hope. In *USENIX Security 2016: 25th USENIX Security Symposium*, Thorsten Holz and Stefan Savage (Eds.). USENIX Association, Austin, TX, USA, 327–343.
- [6] José Baccelar Almeida, Manuel Barbosa, Gilles Barthe, Arthur Blot, Benjamin Grégoire, Vincent Laporte, Tiago Oliveira, Hugo Pacheco, Benedikt Schmidt, and Pierre-Yves Strub. 2017. Jasmin: High-Assurance and High-Speed Cryptography. In *ACM CCS 2017: 24th Conference on Computer and Communications Security*, Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu (Eds.). ACM Press, Dallas, TX, USA, 1807–1823. <https://doi.org/10.1145/3133956.3134078>
- [7] José Baccelar Almeida, Manuel Barbosa, Gilles Barthe, Benjamin Grégoire, Adrien Koutsos, Vincent Laporte, Tiago Oliveira, and Pierre-Yves Strub. 2020. The Last Mile: High-Assurance and High-Speed Cryptographic Implementations. In *2020 IEEE Symposium on Security and Privacy*. IEEE Computer Society Press, San Francisco, CA, USA, 965–982. <https://doi.org/10.1109/SP40000.2020.00028>
- [8] Andris Ambainis, Mike Hamburg, and Dominique Unruh. 2019. Quantum Security Proofs Using Semi-classical Oracles. In *Advances in Cryptology – CRYPTO 2019, Part II (Lecture Notes in Computer Science, Vol. 11693)*, Alexandra Boldyreva and Daniele Micciancio (Eds.). Springer, Heidelberg, Germany, Santa Barbara, CA, USA, 269–295. https://doi.org/10.1007/978-3-030-26951-7_10
- [9] Yawning Angel, Benjamin Dowling, Andreas Hülsing, Peter Schwabe, and Florian Weber. 2022. Post Quantum Noise. , 97–109 pages. <http://cryptojedi.org/papers/#pqnoise>.
- [10] Reza Azarderakhsh, David Jao, and Christopher Leonardi. 2017. Post-Quantum Static-Static Key Agreement Using Multiple Protocol Instances. In *SAC 2017: 24th Annual International Workshop on Selected Areas in Cryptography (Lecture Notes in Computer Science, Vol. 10719)*, Carlisle Adams and Jan Camenisch (Eds.). Springer, Heidelberg, Germany, Ottawa, ON, Canada, 45–63. https://doi.org/10.1007/978-3-319-72565-9_3
- [11] Gustavo Banegas, Daniel J. Bernstein, Fabio Campos, Tung Chou, Tanja Lange, Michael Meyer, Benjamin Smith, and Jana Sotáková. 2021. CTIDH: faster constant-time CSIDH. *IACR Transactions on Cryptographic Hardware and Embedded Systems* 2021, 4 (2021), 351–387. <https://doi.org/10.46586/tches.v2021.i4.351-387> <https://tches.iacr.org/index.php/TCHES/article/view/9069>.
- [12] Mihir Bellare and Phillip Rogaway. 1993. Random Oracles are Practical: A Paradigm for Designing Efficient Protocols. In *ACM CCS 93: 1st Conference on Computer and Communications Security*, Dorothy E. Denning, Raymond Pyle, Ravi Ganesan, Ravi S. Sandhu, and Victoria Ashby (Eds.). ACM Press, Fairfax, Virginia, USA, 62–73. <https://doi.org/10.1145/168588.168596>
- [13] Mihir Bellare and Phillip Rogaway. 2006. The Security of Triple Encryption and a Framework for Code-Based Game-Playing Proofs. In *Advances in Cryptology – EUROCRYPT 2006 (Lecture Notes in Computer Science, Vol. 4004)*, Serge Vaudenay (Ed.). Springer, Heidelberg, Germany, St. Petersburg, Russia, 409–426. https://doi.org/10.1007/11761679_25
- [14] Daniel J. Bernstein. 2006. Curve25519: New Diffie-Hellman Speed Records. In *PKC 2006: 9th International Conference on Theory and Practice of Public Key Cryptography (Lecture Notes in Computer Science, Vol. 3958)*, Moti Yung, Yevgeniy Dodis, Aggelos Kiayias, and Tal Malkin (Eds.). Springer, Heidelberg, Germany, New York, NY, USA, 207–228. https://doi.org/10.1007/11745853_14
- [15] Daniel J. Bernstein, Billy Bob Brumley, Ming-Shing Chen, and Nicola Tuveri. 2022. OpenSSLNTRU: Faster post-quantum TLS key exchange. In *31st USENIX Security Symposium (USENIX Security 22)*. USENIX Association, Boston, MA, 845–862. <https://www.usenix.org/conference/usenixsecurity22/presentation/bernstein>
- [16] Daniel J. Bernstein, Tanja Lange, Chloe Martindale, and Lorenz Panny. 2019. Quantum Circuits for the CSIDH: Optimizing Quantum Evaluation of Isogenies. In *Advances in Cryptology – EUROCRYPT 2019, Part II (Lecture Notes in Computer Science, Vol. 11477)*, Yuval Ishai and Vincent Rijmen (Eds.). Springer, Heidelberg, Germany, Darmstadt, Germany, 409–441. https://doi.org/10.1007/978-3-030-17656-3_15

- [17] Manuel Blum, Paul Feldman, and Silvio Micali. 1988. Non-Interactive Zero-Knowledge and Its Applications (Extended Abstract). In *20th Annual ACM Symposium on Theory of Computing*. ACM Press, Chicago, IL, USA, 103–112. <https://doi.org/10.1145/62212.62222>
- [18] Dan Boneh, Özgür Dagdelen, Marc Fischlin, Anja Lehmann, Christian Schaffner, and Mark Zhandry. 2011. Random Oracles in a Quantum World. In *Advances in Cryptology – ASIACRYPT 2011 (Lecture Notes in Computer Science, Vol. 7073)*, Dong Hoon Lee and Xiaoyun Wang (Eds.). Springer, Heidelberg, Germany, Seoul, South Korea, 41–69. https://doi.org/10.1007/978-3-642-25385-0_3
- [19] Dan Boneh and Mark Zhandry. 2014. Multiparty Key Exchange, Efficient Traitor Tracing, and More from Indistinguishability Obfuscation. In *Advances in Cryptology – CRYPTO 2014, Part I (Lecture Notes in Computer Science, Vol. 8616)*, Juan A. Garay and Rosario Gennaro (Eds.). Springer, Heidelberg, Germany, Santa Barbara, CA, USA, 480–499. https://doi.org/10.1007/978-3-662-44371-2_27
- [20] Xavier Bonnetain and André Schrottenloher. 2020. Quantum Security Analysis of CSIDH. In *Advances in Cryptology – EUROCRYPT 2020, Part II (Lecture Notes in Computer Science, Vol. 12106)*, Anne Canteaut and Yuval Ishai (Eds.). Springer, Heidelberg, Germany, Zagreb, Croatia, 493–522. https://doi.org/10.1007/978-3-030-45724-2_17
- [21] Joppe W. Bos, Craig Costello, Léo Ducas, Ilya Mironov, Michael Naehrig, Valeria Nikolaenko, Ananth Raghunathan, and Douglas Stebila. 2016. Frodo: Take off the Ring! Practical, Quantum-Secure Key Exchange from LWE. In *ACM CCS 2016: 23rd Conference on Computer and Communications Security*, Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi (Eds.). ACM Press, Vienna, Austria, 1006–1018. <https://doi.org/10.1145/2976749.2978425>
- [22] Joppe W. Bos, Craig Costello, Michael Naehrig, and Douglas Stebila. 2015. Post-Quantum Key Exchange for the TLS Protocol from the Ring Learning with Errors Problem. In *2015 IEEE Symposium on Security and Privacy*. IEEE Computer Society Press, San Jose, CA, USA, 553–570. <https://doi.org/10.1109/SP.2015.40>
- [23] Colin Boyd, Yvonne Cliff, Juan González Nieto, and Kenneth G. Paterson. 2008. Efficient One-Round Key Exchange in the Standard Model. In *ACISP 08: 13th Australasian Conference on Information Security and Privacy (Lecture Notes in Computer Science, Vol. 5107)*, Yi Mu, Willy Susilo, and Jennifer Seberry (Eds.). Springer, Heidelberg, Germany, Wollongong, Australia, 69–83.
- [24] Zvika Brakerski, Adeline Langlois, Chris Peikert, Oded Regev, and Damien Stehlé. 2013. Classical hardness of learning with errors. In *45th Annual ACM Symposium on Theory of Computing*, Dan Boneh, Tim Roughgarden, and Joan Feigenbaum (Eds.). ACM Press, Palo Alto, CA, USA, 575–584. <https://doi.org/10.1145/2488608.2488680>
- [25] Jacqueline Brendel, Rune Fiedler, Felix Günther, Christian Janson, and Douglas Stebila. 2022. Post-quantum Asynchronous Deniable Key Exchange and the Signal Handshake. In *Public-Key Cryptography – PKC 2022*, Goichiro Hanaoka, Junji Shikata, and Yohei Watanabe (Eds.). Springer International Publishing, Cham, 3–34.
- [26] Jacqueline Brendel, Marc Fischlin, Felix Günther, Christian Janson, and Douglas Stebila. 2020. Towards Post-Quantum Security for Signal’s X3DH Handshake. In *SAC 2020: 27th Annual International Workshop on Selected Areas in Cryptography (Lecture Notes in Computer Science, Vol. 12804)*, Orr Dunkelman, Michael J. Jacobson Jr., and Colin O’Flynn (Eds.). Springer, Heidelberg, Germany, Halifax, NS, Canada (Virtual Event), 404–430. https://doi.org/10.1007/978-3-030-81652-0_16
- [27] Leon Groot Bruinderink, Andreas Hülsing, Tanja Lange, and Yuval Yarom. 2016. Flush, Gauss, and Reload - A Cache Attack on the BLISS Lattice-Based Signature Scheme. In *Cryptographic Hardware and Embedded Systems – CHES 2016 (Lecture Notes in Computer Science, Vol. 9813)*, Benedikt Gierlichs and Axel Y. Poschmann (Eds.). Springer, Heidelberg, Germany, Santa Barbara, CA, USA, 323–345. https://doi.org/10.1007/978-3-662-53140-2_16
- [28] David Cash, Eike Kiltz, and Victor Shoup. 2008. The Twin Diffie-Hellman Problem and Applications. In *Advances in Cryptology – EUROCRYPT 2008 (Lecture Notes in Computer Science, Vol. 4965)*, Nigel P. Smart (Ed.). Springer, Heidelberg, Germany, Istanbul, Turkey, 127–145. https://doi.org/10.1007/978-3-540-78967-3_8
- [29] Wouter Castryck and Thomas Decru. 2022. An efficient key recovery attack on SIDH (preliminary version). Cryptology ePrint Archive, Report 2022/975. <https://eprint.iacr.org/2022/975>.
- [30] Wouter Castryck, Tanja Lange, Chloe Martindale, Lorenz Panny, and Joost Renes. 2018. CSIDH: An Efficient Post-Quantum Commutative Group Action. In *Advances in Cryptology – ASIACRYPT 2018, Part III (Lecture Notes in Computer Science, Vol. 11274)*, Thomas Peyrin and Steven Galbraith (Eds.). Springer, Heidelberg, Germany, Brisbane, Queensland,

- Australia, 395–427. https://doi.org/10.1007/978-3-030-03332-3_15
- [31] Melissa Chase, David Derler, Steven Goldfeder, Claudio Orlandi, Sebastian Ramacher, Christian Rechberger, Daniel Slamanig, and Greg Zaverucha. 2017. Post-Quantum Zero-Knowledge and Signatures from Symmetric-Key Primitives. In *ACM CCS 2017: 24th Conference on Computer and Communications Security*, Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu (Eds.). ACM Press, Dallas, TX, USA, 1825–1842. <https://doi.org/10.1145/3133956.3133997>
- [32] Craig Costello, Patrick Longa, and Michael Naehrig. 2016. Efficient Algorithms for Supersingular Isogeny Diffie-Hellman. In *Advances in Cryptology – CRYPTO 2016, Part I (Lecture Notes in Computer Science, Vol. 9814)*, Matthew Robshaw and Jonathan Katz (Eds.). Springer, Heidelberg, Germany, Santa Barbara, CA, USA, 572–601. https://doi.org/10.1007/978-3-662-53018-4_21
- [33] Bor de Kock. 2018. *A non-interactive key exchange based on ring-learning with errors*. Master’s thesis. Master’s thesis, Eindhoven University of Technology.
- [34] Whitfield Diffie and Martin E. Hellman. 1976. New Directions in Cryptography. *IEEE Transactions on Information Theory* 22, 6 (1976), 644–654.
- [35] Jintai Ding, Xiang Xie, and Xiaodong Lin. 2012. A Simple Provably Secure Key Exchange Scheme Based on the Learning with Errors Problem. Cryptology ePrint Archive, Report 2012/688. <https://eprint.iacr.org/2012/688>.
- [36] Samuel Dobson and Steven D. Galbraith. 2022. Post-Quantum Signal Key Agreement from SIDH. In *Post-Quantum Cryptography*, Jung Hee Cheon and Thomas Johansson (Eds.). Springer International Publishing, Cham, 422–450.
- [37] Julien Duman, Dominik Hartmann, Eike Kiltz, Sabrina Kunzweiler, Jonas Lehmann, and Doreen Riepel. 2022. Group Action Key Encapsulation and Non-Interactive Key Exchange in the QROM. Cryptology ePrint Archive, Paper 2022/1230. <https://eprint.iacr.org/2022/1230> <https://eprint.iacr.org/2022/1230>.
- [38] Morris J. Dworkin. 2015. *SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions*. Technical Report. National Institute of Standards and Technology. <https://doi.org/10.6028/nist.fips.202>
- [39] Thomas Espitau, Pierre-Alain Fouque, Benoît Gérard, and Mehdi Tibouchi. 2017. Side-Channel Attacks on BLISS Lattice-Based Signatures: Exploiting Branch Tracing against strongSwan and Electromagnetic Emanations in Microcontrollers. In *ACM CCS 2017: 24th Conference on Computer and Communications Security*, Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu (Eds.). ACM Press, Dallas, TX, USA, 1857–1874. <https://doi.org/10.1145/3133956.3134028>
- [40] Amos Fiat and Adi Shamir. 1987. How to Prove Yourself: Practical Solutions to Identification and Signature Problems. In *Advances in Cryptology – CRYPTO’86 (Lecture Notes in Computer Science, Vol. 263)*, Andrew M. Odlyzko (Ed.). Springer, Heidelberg, Germany, Santa Barbara, CA, USA, 186–194. https://doi.org/10.1007/3-540-47721-7_12
- [41] Eduarda S. V. Freire, Dennis Hofheinz, Eike Kiltz, and Kenneth G. Paterson. 2013. Non-Interactive Key Exchange. In *PKC 2013: 16th International Conference on Theory and Practice of Public Key Cryptography (Lecture Notes in Computer Science, Vol. 7778)*, Kaoru Kurosawa and Goichiro Hanaoka (Eds.). Springer, Heidelberg, Germany, Nara, Japan, 254–271. https://doi.org/10.1007/978-3-642-36362-7_17
- [42] Eiichiro Fujisaki and Tatsuki Okamoto. 1999. Secure Integration of Asymmetric and Symmetric Encryption Schemes. In *Advances in Cryptology – CRYPTO’99 (Lecture Notes in Computer Science, Vol. 1666)*, Michael J. Wiener (Ed.). Springer, Heidelberg, Germany, Santa Barbara, CA, USA, 537–554. https://doi.org/10.1007/3-540-48405-1_34
- [43] Eiichiro Fujisaki and Tatsuki Okamoto. 2013. Secure Integration of Asymmetric and Symmetric Encryption Schemes. *Journal of Cryptology* 26, 1 (Jan. 2013), 80–101. <https://doi.org/10.1007/s00145-011-9114-1>
- [44] Steven D. Galbraith, Christophe Petit, Barak Shani, and Yan Bo Ti. 2016. On the Security of Supersingular Isogeny Cryptosystems. In *Advances in Cryptology – ASIACRYPT 2016, Part I (Lecture Notes in Computer Science, Vol. 10031)*, Jung Hee Cheon and Tsuyoshi Takagi (Eds.). Springer, Heidelberg, Germany, Hanoi, Vietnam, 63–91. https://doi.org/10.1007/978-3-662-53887-6_3
- [45] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. 1985. The Knowledge Complexity of Interactive Proof-Systems (Extended Abstract). In *17th Annual ACM Symposium on Theory of Computing*. ACM Press, Providence, RI, USA, 291–304. <https://doi.org/10.1145/22145.22178>
- [46] Siyao Guo, Pritish Kamath, Alon Rosen, and Katerina Sotiraki. 2020. Limits on the Efficiency of (Ring) LWE Based Non-interactive Key Exchange. In *PKC 2020: 23rd International Conference on Theory and Practice of Public Key Cryptography, Part I (Lecture Notes in Computer Science, Vol. 12110)*, Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas (Eds.). Springer, Heidelberg, Germany, Edinburgh, UK, 374–395. https://doi.org/10.1007/978-3-030-45374-9_13

- [47] Keitaro Hashimoto, Shuichi Katsumata, Kris Kwiatkowski, and Thomas Prest. 2021. An Efficient and Generic Construction for Signal’s Handshake (X3DH): Post-Quantum, State Leakage Secure, and Deniable. In *PKC 2021: 24th International Conference on Theory and Practice of Public Key Cryptography, Part II (Lecture Notes in Computer Science, Vol. 12711)*, Juan Garay (Ed.). Springer, Heidelberg, Germany, Virtual Event, 410–440. https://doi.org/10.1007/978-3-030-75248-4_15
- [48] Andreas Hülsing, Kai-Chun Ning, Peter Schwabe, Florian Weber, and Philip R. Zimmermann. 2021. Post-quantum WireGuard. In *2021 IEEE Symposium on Security and Privacy*. IEEE Computer Society Press, San Francisco, CA, USA, 304–321. <https://doi.org/10.1109/SP40001.2021.00030>
- [49] David Jao, Reza Azarderakhsh, Matthew Campagna, Craig Costello, Luca De Feo, Basil Hess, Amir Jalali, Brian Koziel, Brian LaMacchia, Patrick Longa, Michael Naehrig, Joost Renes, Vladimir Soukharev, and David Urbanik. 2017. *SIKE*. Technical Report. National Institute of Standards and Technology. available at <https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-1-submissions>.
- [50] David Jao and Luca De Feo. 2011. Towards Quantum-Resistant Cryptosystems from Supersingular Elliptic Curve Isogenies. In *Post-Quantum Cryptography - 4th International Workshop, PQCrypto 2011*, Bo-Yin Yang (Ed.). Springer, Heidelberg, Germany, Taipei, Taiwan, 19–34. https://doi.org/10.1007/978-3-642-25405-5_2
- [51] John Kelsey, Shu jen Change, and Ray Perlner. 2016. *SHA-3 derived functions: cSHAKE, KMAC, TupleHash and ParallelHash*. Technical Report. National Institute of Standards and Technology. <https://doi.org/10.6028/nist.sp.800-185>
- [52] Hugo Krawczyk and Hoeteck Wee. 2016. The OPTLS Protocol and TLS 1.3. In *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*, IEEE, Saarbruecken, Germany, 81–96. <https://doi.org/10.1109/eurosp.2016.18>
- [53] Kris Kwiatkowski and Luke Valenta. 2019. The TLS Post-Quantum Experiment. Post on the Cloudflare blog. <https://blog.cloudflare.com/the-tls-post-quantum-experiment/>.
- [54] Adam Langley. 2016. CECPQ1 results. Blog post. <https://www.imperialviolet.org/2016/11/28/cecpq1.html>.
- [55] Adam Langley. 2018. CECPQ2. Blog post. <https://www.imperialviolet.org/2018/12/12/cecpq2.html>.
- [56] Adeline Langlois and Damien Stehlé. 2015. Worst-Case to Average-Case Reductions for Module Lattices. *Designs, Codes and Cryptography* 75, 3 (2015), 565–599.
- [57] Vadim Lyubashevsky. 2017. Converting NewHope/LWE key exchange to a Diffe-Hellman-like algorithm. Crypto Stack Exchange. <https://crypto.stackexchange.com/questions/48146/converting-newhope-lwe-key-exchange-to-a-diffe-hellman-like-algorithm> [Online]; <https://crypto.stackexchange.com/questions/48146/converting-newhope-lwe-key-exchange-to-a-diffe-hellman-like-algorithm>.
- [58] Vadim Lyubashevsky, Léo Ducas, Eike Kiltz, Tancrede Lepoint, Peter Schwabe, Gregor Seiler, Damien Stehlé, and Shi Bai. 2022. *CRYSTALS-DILITHIUM*. Technical Report. National Institute of Standards and Technology. available at <https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022>.
- [59] Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Maxime Plançon. 2022. Lattice-Based Zero-Knowledge Proofs and Applications: Shorter, Simpler, and More General. In *Advances in Cryptology – CRYPTO 2022, Part II (Lecture Notes in Computer Science, Vol. 13508)*, Yevgeniy Dodis and Thomas Shrimpton (Eds.). Springer, Heidelberg, Germany, Santa Barbara, CA, USA, 71–101. https://doi.org/10.1007/978-3-031-15979-4_3
- [60] Moxie Marlinspike and Trevor Perrin. 2016. The X3DH Key Agreement Protocol (Revision 1). Part of the Signal Protocol Documentation. <https://signal.org/docs/specifications/x3dh/x3dh.pdf>.
- [61] Peter L. Montgomery. 1985. Modular Multiplication without Trial Division. *Math. Comp.* 44, 170 (1985), 519–521.
- [62] Kaushik Nath and Daniel J. Bernstein. 2022. lib25519. <https://lib25519.cr.yp.to/>.
- [63] NIST. 2016. Submission Requirements and Evaluation Criteria for the Post-Quantum Cryptography Standardization Process. <https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/call-for-proposals-final-dec-2016.pdf>.
- [64] Christian Paquin, Douglas Stebila, and Goutam Tamvada. 2020. Benchmarking Post-quantum Cryptography in TLS. In *Post-Quantum Cryptography - 11th International Conference, PQCrypto 2020*, Jintai Ding and Jean-Pierre Tillich (Eds.). Springer, Heidelberg, Germany, Paris, France, 72–91. https://doi.org/10.1007/978-3-030-44223-1_5
- [65] Chris Peikert. 2020. He Gives C-Sieves on the CSIDH. In *Advances in Cryptology – EUROCRYPT 2020, Part II (Lecture Notes in Computer Science, Vol. 12106)*, Anne Canteaut and Yuval Ishai (Eds.). Springer, Heidelberg, Germany, Zagreb, Croatia, 463–492. https://doi.org/10.1007/978-3-030-45724-2_16

- [66] Trevor Perrin. 2018. Noise Protocol Framework. <https://noiseprotocol.org/noise.pdf> (Revision 34 vom 2018-07-11).
- [67] Peter Pessl, Leon Groot Bruinderink, and Yuval Yarom. 2017. To BLISS-B or not to be: Attacking strongSwan’s Implementation of Post-Quantum Signatures. In *ACM CCS 2017: 24th Conference on Computer and Communications Security*, Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu (Eds.). ACM Press, Dallas, TX, USA, 1843–1855. <https://doi.org/10.1145/3133956.3134023>
- [68] Rachel Player. 2018. *Parameter selection in lattice-based cryptography*. Ph. D. Dissertation. Royal Holloway, University of London.
- [69] Thomas Prest, Pierre-Alain Fouque, Jeffrey Hoffstein, Paul Kirchner, Vadim Lyubashevsky, Thomas Pornin, Thomas Ricosset, Gregor Seiler, William Whyte, and Zhenfei Zhang. 2022. *FALCON*. Technical Report. National Institute of Standards and Technology. available at <https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022>.
- [70] Oded Regev. 2005. On lattices, learning with errors, random linear codes, and cryptography. In *37th Annual ACM Symposium on Theory of Computing*, Harold N. Gabow and Ronald Fagin (Eds.). ACM Press, Baltimore, MA, USA, 84–93. <https://doi.org/10.1145/1060590.1060603>
- [71] Eric Rescorla. 2018. The Transport Layer Security (TLS) Protocol Version 1.3. RFC 8446. <https://doi.org/10.17487/RFC8446>
- [72] Peter Schwabe, Roberto Avanzi, Joppe Bos, Léo Ducas, Eike Kiltz, Tancrede Lepoint, Vadim Lyubashevsky, John M. Schanck, Gregor Seiler, Damien Stehlé, and Jintai Ding. 2022. *CRYSTALS-KYBER*. Technical Report. National Institute of Standards and Technology. available at <https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022>.
- [73] Peter Schwabe, Douglas Stebila, and Thom Wiggers. 2020. Post-Quantum TLS Without Handshake Signatures. In *ACM CCS 2020: 27th Conference on Computer and Communications Security*, Jay Ligatti, Xinming Ou, Jonathan Katz, and Giovanni Vigna (Eds.). ACM Press, Virtual Event, USA, 1461–1480. <https://doi.org/10.1145/3372297.3423350>
- [74] Gregor Seiler. 2018. Faster AVX2 optimized NTT multiplication for Ring-LWE lattice cryptography. Cryptology ePrint Archive, Report 2018/039. <https://eprint.iacr.org/2018/039>.
- [75] Peter W. Shor. 1994. Algorithms for Quantum Computation: Discrete Logarithms and Factoring. In *35th Annual Symposium on Foundations of Computer Science*. IEEE Computer Society Press, Santa Fe, NM, USA, 124–134. <https://doi.org/10.1109/SFCS.1994.365700>
- [76] Sara Stadler, Vitor Sakaguti, Harjot Kaur, and Anna Lena Fehlhäber. 2021. Hybrid Signal protocol for post-quantum email encryption. Cryptology ePrint Archive, Report 2021/875. <https://eprint.iacr.org/2021/875>.
- [77] Dominique Unruh. 2015. Non-Interactive Zero-Knowledge Proofs in the Quantum Random Oracle Model. In *Advances in Cryptology – EUROCRYPT 2015, Part II (Lecture Notes in Computer Science, Vol. 9057)*, Elisabeth Oswald and Marc Fischlin (Eds.). Springer, Heidelberg, Germany, Sofia, Bulgaria, 755–784. https://doi.org/10.1007/978-3-662-46803-6_25
- [78] Bas Westerbaan and Cefan Daniel Rubin. 2019. Defending against future threats: Cloudflare goes post-quantum. Post on the Cloudflare blog. <https://blog.cloudflare.com/post-quantum-for-all/>.

A PROOFS FOR SECTION 4 (DEFINITIONS)

A.1 Non-Interactive Zero-Knowledge Proofs

Zero-Knowledge proofs [45] allow a verifier to convince a prover of the validity of a statement without revealing anything beyond that. In the random oracle model [12] zero-knowledge proofs can be made non-interactive [17] by applying the Fiat-Shamir transformation [40].

Definition A.1 (Zero-Knowledge Proof of Knowledge). A zero-knowledge proof of knowledge ZKPoK for an NP language \mathcal{L} ⁷ is defined as a tuple ZKPoK := (ZK.Priv, ZK.Ver) of the following oracle algorithms.

⁷The language \mathcal{L} is defined as the set of all yes-instances of the relation R , i.e. $\mathcal{L} = \{x : \exists w \text{ s.t. } R(x, w) = 1\}$.

$\pi \xleftarrow{\$} \text{ZK.Prv}^H(x, w)$: Given a statement x and a witness w , the probabilistic prover algorithm ZK.Prv returns a proof π .

$1/0 \leftarrow \text{ZK.Ver}^H(x, \pi)$: Given a statement x and a proof π , the deterministic verifier algorithm returns either 1 for accept or 0 for reject.

Similar to the work of [77] we assume a distribution RODist on functions, modelling the distribution of our random oracle. That is, given a random oracle $H : \{0, 1\}^* \rightarrow \{0, 1\}^n$, RODist would be the uniform distribution on $\{0, 1\}^* \rightarrow \{0, 1\}^n$.

ZKPoK Security Notions. Besides *completeness*, which captures that valid proofs are accepted by the verifier, a zero-knowledge proof of knowledge should fulfil two additional properties; *soundness* ensures a cheating prover cannot convince the verifier of a false proof, and *zero-knowledge* conveys that the verifier learns nothing from its interaction with the prover beyond the fact that he knows a valid witness to the proof. We make this more precise with the following definitions and note that we require the strong notion of *simulation soundness with a straight-line extractor* [77], sometimes referred to as “online extractability” in the literature.

Definition A.2 (Completeness). *Completeness* for a zero-knowledge proof of knowledge ZKPoK of an NP language \mathcal{L} is defined via the game $\text{CMPLT}_{\text{ZKPoK}}$ depicted in Figure 8. For an adversary A , we define A ’s advantage in $\text{CMPLT}_{\text{ZKPoK}}$ as

$$\text{Adv}_{\text{ZKPoK}}^{\text{CMPLT}}(A) := \Pr[\text{CMPLT}_{\text{ZKPoK}}^A \Rightarrow 1],$$

and say that ZKPoK is (ϵ, Q_H) - CMPLT if for all quantum-polynomial-time adversaries A , making at most Q_H queries (possibly in superposition) to the random oracle H , we have $\text{Adv}_{\text{ZKPoK}}^{\text{CMPLT}}(A) \leq \epsilon(Q_H)$.

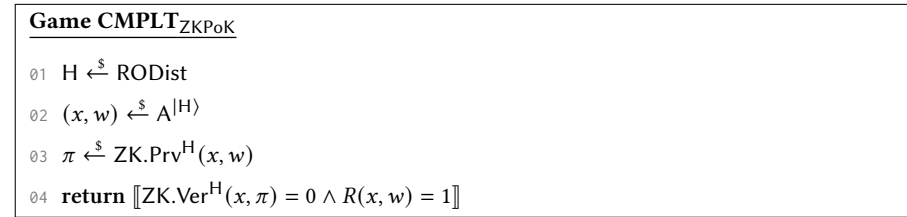


Fig. 8. Game defining $\text{CMPLT}_{\text{ZKPoK}}$ for a zero-knowledge proof of knowledge ZKPoK with adversary A .

For the following notions we additionally require a simulator $\text{ZK.Sim} := (\text{ZK.Sim}_1, \text{ZK.Sim}_2)$ that is split into two classical algorithms ZK.Sim_1 and ZK.Sim_2 , where:

$H \xleftarrow{\$} \text{ZK.Sim}_1$: The probabilistic simulator algorithm ZK.Sim_1 returns a circuit H which represents the initial simulated random oracle.

Game $\mathbf{ZK}_{\text{ZKPoK}}^0$	Procedure $\mathbf{ZK.Sim}'_2(x, w)$
01 $H \stackrel{s}{\leftarrow} \text{RODist}$	07 if $R(x, w) = 0$:
02 $b' \stackrel{s}{\leftarrow} A^{H, \text{ZK.Priv}(\cdot, \cdot)}$	08 return \perp
03 return $\llbracket b' = 0 \rrbracket$	09 else :
Game $\mathbf{ZK}_{\text{ZKPoK}}^1$	10 return $\text{ZK.Sim}_2(x)$
04 $H \stackrel{s}{\leftarrow} \text{ZK.Sim}_1$	
05 $b' \stackrel{s}{\leftarrow} A^{H, \text{ZK.Sim}'_2(\cdot, \cdot)}$	
06 return $\llbracket b' = 1 \rrbracket$	

Fig. 9. Games defining $\mathbf{ZK}_{\text{ZKPoK}}^b$ for a zero-knowledge proof of knowledge ZKPoK with adversary A and simulator $\text{ZK.Sim} := (\text{ZK.Sim}_1, \text{ZK.Sim}_2)$. The purpose of $\text{ZK.Sim}'_2(\cdot, \cdot)$ is merely to serve as an interface for the adversary who expects a prover taking two arguments x and w .

$\pi \stackrel{s}{\leftarrow} \text{ZK.Sim}_2(x)$: Given a statement x the stateful simulator algorithm ZK.Sim_2 returns a proof π . Additionally, ZK.Sim_2 is given access to the description of H and may replace it with a different description (i.e. it can program the random oracle).

Definition A.3 (Zero-Knowledge [45]). Zero-knowledge for a zero-knowledge proof of knowledge ZKPoK of an NP language \mathcal{L} is defined via the game $\mathbf{ZK}_{\text{ZKPoK}}^b$, depicted in Figure 9, where $\mathbf{ZK}_{\text{ZKPoK}}^b$ is parametrised by a bit b . For an adversary A, we define A's advantage in $\mathbf{ZK}_{\text{ZKPoK}}^b$ as

$$\text{Adv}_{\text{ZKPoK}}^{\mathbf{ZK}}(A) := \left| \Pr[\mathbf{ZK}_{\text{ZKPoK}}^{0,A} \Rightarrow 1] - \Pr[\mathbf{ZK}_{\text{ZKPoK}}^{1,A} \Rightarrow 1] \right|,$$

and say that ZKPoK is (ϕ, Q_H) -**ZK**, if there exists a PPT simulator $\text{ZK.Sim} := (\text{ZK.Sim}_1, \text{ZK.Sim}_2)$, such that for all quantum-polynomial-time adversaries A, making at most Q_H queries (possibly in superposition) to the random oracle H , we have $\text{Adv}_{\text{ZKPoK}}^{\mathbf{ZK}}(A) \leq \phi(Q_H)$.

Definition A.4 (Simulation-Sound Online-Extractability [77]). Simulation-sound online-extractability⁸ for a zero-knowledge proof of knowledge ZKPoK of an NP language \mathcal{L} is defined via the game $\mathbf{SSND}_{\text{ZKPoK}}$, depicted in Figure 10. For an adversary A, we define A's advantage in $\mathbf{SSND}_{\text{ZKPoK}}$ as

$$\text{Adv}_{\text{ZKPoK}}^{\mathbf{SSND}}(A) := \Pr[\mathbf{SSND}_{\text{ZKPoK}}^A \Rightarrow 1],$$

and say that ZKPoK is (ψ, Q_H) -**SSND** relative to a simulator $\text{ZK.Sim} := (\text{ZK.Sim}_1, \text{ZK.Sim}_2)$, if there exists a PPT extractor ZK.Ext such that for all quantum-polynomial-time adversaries A, making at most Q_H queries to the random oracle H , we have $\text{Adv}_{\text{ZKPoK}}^{\mathbf{SSND}}(A) \leq \psi(Q_H)$.

⁸Online-extractability is sometimes referred to as straight line extractability in the literature.

Game SSND_{ZKPoK}

```

01  $H \xleftarrow{\$} \text{ZK.Sim}_1$ 
02  $(x, \pi) \xleftarrow{\$} A^{|H|}, \text{ZK.Sim}_2(\cdot)$ 
03  $w \xleftarrow{\$} \text{ZK.Ext}(H, x, \pi)$ 
04 return  $\llbracket \text{ZK.Ver}^H(x, \pi) = 1 \wedge R(x, w) = 0 \wedge (x, \pi) \notin \tilde{\pi} \rrbracket$ 

```

Fig. 10. Games defining **SSND_{ZKPoK}** for a zero-knowledge proof of knowledge ZKPoK with adversary A , simulator $\text{ZK.Sim} := (\text{ZK.Sim}_1, \text{ZK.Sim}_2)$ and extractor ZK.Ext . Here, $\tilde{\pi}$ denotes the set of all proofs returned by $\text{ZK.Sim}_2(\cdot)$ (together with the corresponding statements).

B PROOFS FOR SECTION 5 (CONSTRUCTION)

LEMMA 1 (HONEST CORRECTNESS). For all (possibly unbounded) adversaries A the non-interactive key exchange $\text{NIKE} := (\text{Stp}, \text{Gen}, \text{SdK})$ construction depicted in Figure 5 has *honest correctness error*

$$\delta \leq \frac{4\beta^2 d^2 N}{q}$$

as per Definition 4.2.

PROOF. The proof strategy is similar to the proof of Theorem 5.1, except that we can bound the probability of any coefficient of k^* being in the interval

$$S^* = \left[\frac{q}{4} \pm \beta^2 dN \right] \cup \left[\frac{3q}{4} \pm \beta^2 dN \right]$$

by $\frac{4\beta^2 d^2 N}{q}$, with a union bound over all coefficients. ■

PROOF OF THEOREM 5.3. Let A be an adversary against NIKE in the **ActSec** game. Consider the sequence of games in Figure 11, where Q_{TQ} is the number of queries to TestQue .

Game G_0 . This is the original $\text{ActSec}_{\text{NIKE}, \text{par}}^0$ game, where the bit b is fixed to 0, hence

$$\Pr \left[G_0^A \Rightarrow 1 \right] = \Pr \left[\text{ActSec}_{\text{NIKE}, \text{par}}^{0,A} \Rightarrow 1 \right].$$

Game G_1 . In this game we modify the RegCorUsr oracle so that the secret key $\widetilde{sk}'_{\text{ID}}$ is extracted from the proof π of a public key pk on Line 18, and stored with the identity ID . This requires the strong notion of simulation-sound online-extractability, from Definition A.4. If the extraction fails, then by default $\widetilde{sk}'_{\text{ID}} = \perp$. I.e. the secret key is not stored, as in the original game. Therefore,

$$\left| \Pr \left[G_0^A \Rightarrow 1 \right] - \Pr \left[G_1^A \Rightarrow 1 \right] \right| \leq Q_{\text{RCU}} \cdot \text{Adv}_{\text{ZKPoK}}^{\text{SSND}}(B_1).$$

Game G₂. In this game we introduce a new condition for aborting: If at any point in the simulation the adversary asks a query to the TestQue or to the RevCorQue oracles on two public keys that cause a key mismatch, then abort the simulation. Note that this condition is efficiently testable, as the game knows all the secret keys. We can bound the probability of this event happening with a reduction to the semi-malicious correctness property, Definition 4.3, of NIKE' by

$$\left| \Pr \left[G_1^A \Rightarrow 1 \right] - \Pr \left[G_2^A \Rightarrow 1 \right] \right| \leq \text{Adv}_{\text{NIKE}', \text{par}'}^{\text{SM-COR}}(B_2).$$

Game G₃. In this game we modify the RevCorQue oracle on Line 29 and Line 34 to use the secret key that was extracted when the corresponding public key was registered as a corrupt key. Since the derived key is always the same for both secret keys, we get

$$\left| \Pr \left[G_2^A \Rightarrow 1 \right] - \Pr \left[G_3^A \Rightarrow 1 \right] \right| = 0.$$

Game G_{4.0}. In this game we modify the RegHonUsr oracle and replace the zero-knowledge proof of knowledge on Line 11 with a simulated proof. By the zero-knowledge property, Definition A.3, we get

$$\left| \Pr \left[G_3^A \Rightarrow 1 \right] - \Pr \left[G_{4.0}^A \Rightarrow 1 \right] \right| \leq Q_{\text{RHU}} \cdot \text{Adv}_{\text{ZKPoK}}^{\text{ZK}}(B_4).$$

Game G_{4.i}. This is identical to the previous game, except that the i^{th} query to TestQue is answered with the bit b fixed to 1. We now state the following Lemma.

CLAIM (REDUCTION). There exists a pair of adversaries $B_{3,i}$ and $B'_{3,i}$ such that

$$\left| \Pr \left[G_{4,i}^A \Rightarrow 1 \right] - \Pr \left[G_{4,i+1}^A \Rightarrow 1 \right] \right| \leq \epsilon, \quad (1)$$

where

$$\epsilon = Q_{\text{RHU}}^2 \cdot \text{Adv}_{\text{NIKE}', \text{par}'}^{\text{PasSec}}(B_{3,i}) + 2 \cdot \text{Adv}_{\text{NIKE}', \text{par}'}^{\text{SM-COR}}(B'_{3,i}).$$

PROOF. To prove the Claim, we modify the game to guess two identities ID^* and ID^{**} that were queried to the RegHonUsr oracle. As a first modification, we no longer use the secret keys corresponding to ID^* and ID^{**} to answer any oracle queries, except for the query involving both ID^* and ID^{**} . Since key mismatches cannot happen, the resulting game is in fact identical to the previous one.

Next, we further modify the game to no longer check for key mismatches in that involve the public keys associated with ID^* and ID^{**} , this change is indistinguishable by another invocation of the semi-malicious correctness.

Next, switch the bit for the i^{th} query. To show that this change is indistinguishable, we construct a reduction $B_{3,i}$ (for $i \in \{0, \dots, Q_{\text{TQ}} - 1\}$) against PasSec of NIKE'. As a first step, the reduction sets the public keys given by the challenger to be the keys associated with ID^* and ID^{**} . RegHonUsr,

RegCorUsr, and RevCorQue remain unchanged, and the only difference is in the case where the TestQue oracle is queried on ID^* and ID^{**} . For the latter queries, we consider two cases:

- The query involving both ID^* and ID^{**} is the i^{th} query. In this case we simply answer with the key k_b provided by the reduction.
- The query involving both ID^* and ID^{**} is *not* the i^{th} query. In this case we abort the execution.

Note that, if the reduction correctly guesses ID^* and ID^{**} as being the identities queried in the i^{th} query of the TestQue oracle, then the second case does not happen and the reduction perfectly reproduces the view of the adversary with the bit $b = 0$ (in case k_b is the real key) or with the bit $b = 1$ (in case k_b is random). Since the reduction guesses correctly with probability at least $1/Q_{\text{RHU}}^2$, the bound follows.

As the final change to the experiment, we undo the first and second modifications done above, namely, we check again for key mismatches for all keys and we no longer randomly sample two identities. Indistinguishability follows by a similar argument. Overall, this yields Equation (1). \blacksquare

Game $G_{4.Q_{\text{TQ}}-1}$. In this game the bit b is fixed to 1. Applying a standard hybrid argument, yields

$$\left| \Pr \left[G_{4.Q_{\text{TQ}}-1}^A \Rightarrow 1 \right] - \Pr \left[G_{4.0}^A \Rightarrow 1 \right] \right| \leq Q_{\text{TQ}} \cdot \epsilon.$$

Game G_5 . In this game we undo the changes made Games G_2 and G_3 . Appealing to the semi-malicious correctness, we obtain

$$\left| \Pr \left[G_5^A \Rightarrow 1 \right] - \Pr \left[G_{4.Q_{\text{TQ}}-1}^A \Rightarrow 1 \right] \right| \leq \text{Adv}_{\text{NIKE}', \text{par}'}^{\text{SM-COR}}(B_2).$$

Game G_6 . In this game we undo the changes made to the RegHonUsr oracle in Game G_4 and replace the simulated proof with a real proof on Line 12. Appealing to Definition A.3 again, we get

$$\left| \Pr \left[G_6^A \Rightarrow 1 \right] - \Pr \left[G_5^A \Rightarrow 1 \right] \right| \leq Q_{\text{RHU}} \cdot \text{Adv}_{\text{ZKPoK}}^{\text{ZK}}(B_4).$$

Game G_7 . In this game we undo the changes made Game G_1 . Using a similar argument, we obtain

$$\left| \Pr \left[G_7^A \Rightarrow 1 \right] - \Pr \left[G_6^A \Rightarrow 1 \right] \right| \leq Q_{\text{RCU}} \cdot \text{Adv}_{\text{ZKPoK}}^{\text{SSND}}(B_1).$$

Observe that this game is the original $\text{ActSec}_{\text{NIKE}, \text{par}}^1$ game. Hence,

$$\Pr \left[G_7^A \Rightarrow 1 \right] = \Pr \left[\text{ActSec}_{\text{NIKE}, \text{par}}^{1,A} \Rightarrow 1 \right].$$

Collecting all probabilities yields the bound stated in Theorem 5.3.

Game ActSec_{NIKE,par}^b	Oracle RevCorQue(ID₁, ID₂)
01 $\mathcal{D} := \perp$	25 if (<i>honest</i> , ID ₁ , ·, ·) ∈ $\mathcal{D} \wedge$ (<i>corrupt</i> , ID ₂ , ·, ·) ∈ \mathcal{D} :
02 $\mathcal{K} := \perp$	26 if SdK(ID ₂ , pk_2 , ID ₁ , sk_1) ≠ SdK(ID ₁ , pk_1 , ID ₂ , sk_2) : / G ₂ -
03 $b := 0$	27 abort / G ₂ -G _{4,Q_{TQ}-1}
04 $b := 1$	28 return SdK(ID ₂ , pk_2 , ID ₁ , sk_1) / G ₀ G _{4,Q_{TQ}-1}
05 $b' \leftarrow \mathcal{A}^{\text{RegHonUsr}(\cdot), \text{RegCorUsr}(\cdot, \cdot), \text{RevCorQue}(\cdot, \cdot), \text{TestQue}(\cdot, \cdot)}$	29 return SdK(ID ₁ , pk_1 , ID ₂ , $\widetilde{sk_2}$) / G ₃
06 return $\llbracket b = b' \rrbracket$	30 elseif (<i>corrupt</i> , ID ₁ , ·, ·) ∈ $\mathcal{D} \wedge$ (<i>honest</i> , ID ₂ , ·, ·) ∈ \mathcal{D} :
Oracle RegHonUsr (ID ∈ \mathcal{IDS})	31 if SdK(ID ₁ , pk_1 , ID ₂ , sk_2) ≠ SdK(ID ₂ , pk_2 , ID ₁ , sk_1) : / G ₂ -
07 if (<i>corrupt</i> , ID, \perp , ·) ∈ \mathcal{D} :	32 abort / G ₂ -G _{4,Q_{TQ}-1}
08 return \perp	33 return SdK(ID ₁ , pk_1 , ID ₂ , sk_2)
09 ($sk'_{ID} \leftarrow \mathcal{S} \text{Gen}(ID)$)	34 return SdK(ID ₂ , pk_2 , ID ₁ , $\widetilde{sk_1}$) / G ₃
10 $\pi \leftarrow \mathcal{S} \text{ZK.Priv}(pk'_{ID}, sk'_{ID})$	35 if ID ₁ = ID ₂ :
11 $\pi \leftarrow \mathcal{S} \text{ZK.Sim}_2(pk'_{ID})$	36 return \perp
12 $\pi \leftarrow \mathcal{S} \text{ZK.Priv}(pk'_{ID}, sk'_{ID})$	37 if (<i>honest</i> , ID ₁ , ·, ·) ∈ $\mathcal{D} \wedge$ (<i>honest</i> , ID ₂ , ·, ·) ∈ \mathcal{D} :
13 $sk_{ID} := sk'_{ID}$	38 $b := 1$ / G _{4,i}
14 $pk_{ID} := (pk'_{ID}, \pi)$	39 if $b = 0$:
15 $\mathcal{D} \cup \{(honest, ID, sk_{ID}, pk_{ID})\}$	40 if SdK(ID ₁ , pk_1 , ID ₂ , sk_2) ≠ SdK(ID ₂ , pk_2 , ID ₁ , sk_1) :
16 return pk_{ID}	41 abort / G ₂ -G _{4,Q_{TQ}-1}
Oracle RegCorUsr (ID ∈ \mathcal{IDS} , pk)	42 $k := \text{SdK}(ID_1, pk_1, ID_2, sk_2)$
17 parse $pk \rightarrow (pk'_{ID}, \pi)$	43 if $b = 1$:
18 $\widetilde{sk'_{ID}} \leftarrow \mathcal{S} \text{ZK.Ext}(H, pk'_{ID}, \pi)$ / G ₁ -G ₅	44 if (ID ₁ , ID ₂ , k) ∈ $\mathcal{K} \vee$ (ID ₂ , ID ₁ , k) ∈ \mathcal{K} :
19 if (<i>corrupt</i> , ID, ·, ·) ∈ \mathcal{D} :	45 return k
20 (<i>corrupt</i> , ID, \perp , ·) := (<i>corrupt</i> , ID, \perp , pk)	46 $k \leftarrow \mathcal{S} \mathcal{SKS}$
21 (<i>corrupt</i> , ID, ·, ·) := (<i>corrupt</i> , ID, $\widetilde{sk'_{ID}}$, pk) / G ₁ -G ₅	47 $\mathcal{K} \cup \{(ID_1, ID_2, k)\}$
22 else :	48 return k
23 $\mathcal{D} \cup \{(corrupt, ID, \perp, pk)\}$	49 return \perp
24 $\mathcal{D} \cup \{(corrupt, ID, \widetilde{sk'_{ID}}, pk)\}$ / G ₁ -G ₅	

Fig. 11. Games G₀, G₁, G₃, G_{4,i} (for $i \in \{0 \leq i \leq Q_{TQ} - 1\}$), G₆ for the proof of ActSec of NIKE in Figure 7. ■

ISBN 978-82-326-7104-5 (printed ver.)
ISBN 978-82-326-7103-8 (electronic ver.)
ISSN 1503-8181 (printed ver.)
ISSN 2703-8084 (online ver.)