Future-Proofing Key Exchange Protocols

Vom Fachbereich Informatik der
Technischen Universität Darmstadt genehmigte

Dissertation

zur Erlangung des Grades
Doctor rerum naturalium (Dr. rer. nat.)
von

Jacqueline Brendel, M.Sc.
geboren in Deggendorf

Referenten:	Prof. Dr. Marc Fischlin
	Prof. Dr. Cas Cremers
Tag der Einreichung:	01.10.2019
Tag der mündlichen Prüfung:	28.11.2019

Darmstadt, 2019
D 17

Erklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit – abgesehen von den in ihr ausdrücklich genannten Hilfen – selbständig verfasst habe.

Wissenschaftlicher Werdegang

Oktober 2008 – Januar 2012

Studium der Mathematik mit Nebenfach Informatik
Ludwig-Maximilians-Universität München, **Bachelor of Science**

April 2012 – September 2012

Masterstudium der Mathematik mit Nebenfach Informatik
Ludwig-Maximilians-Universität München

Oktober 2012 – Januar 2016

Studium der Mathematik mit Vertiefung Informatik
Technische Universität Darmstadt, **Master of Science**

seit Februar 2016

Doktorandin der Informatik an der Technischen Universität Darmstadt

List of Publications

Note. Historically, cryptographic research has been a subfield of mathematics and thus it is still overwhelmingly common to sort author lists alphabetically such that in particular the concept of *lead authors* does not readily apply (cf. for example the culture statement of the American Mathematical Society [Soc04]).

Papers in Conferences and Workshops with Proceedings

[1] Jacqueline Brendel, Marc Fischlin, and Felix Günther. Breakdown Resilience of Key Exchange Protocols and the Cases of NewHope and TLS 1.3. In Kazue Sako, Steve Schneider, and Peter Y.A. Ryan, editors, *Computer Security – ESORICS 2019*, pages 521–541, Cham, 2019. Springer. Also available as Cryptology ePrint Archive Report 2017/1252, https://eprint.iacr.org/2017/1252. **Best Paper Award, Chapter 6.**

[2] 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*, pages 206–226, Cham, 2019. Springer. Also available as Cryptology ePrint Archive Report 2018/903, https://eprint.iacr.org/2018/903. **Chapter 5.**

[3] Jacqueline Brendel and Marc Fischlin. Zero Round-Trip Time for the Extended Access Control Protocol. In Simon N. Foley, Dieter Gollmann, and Einar Snekkenes, editors, *Computer Security – ESORICS 2017*, pages 297–314, Cham, 2017. Springer. Also available as Cryptology ePrint Archive Report 2017/060, https://eprint.iacr.org/2017/060.

[4] Jacqueline Brendel, Marc Fischlin, Felix Günther, and Christian Janson. PRF-ODH: Relations, Instantiations, and Impossibility Results. In *37th International Cryptology Conference (CRYPTO 2017, Part III)*, volume 10403 of *Lecture Notes in Computer Science*, pages 651–681, Cham, 2017. Springer. Also available as Cryptology ePrint Archive Report 2017/517, https://eprint.iacr.org/2017/517. **Chapter 7.**

[5] Jacqueline Brendel and Denise Demirel. Efficient Proactive Secret Sharing. In *14th Annual Conference on Privacy, Security and Trust (PST 2016)*, pages 543–550, 2016. Also available as Cryptology ePrint Archive Report 2017/719, https://eprint.iacr.org/2017/719.pdf.

Surveys

[6] Jacqueline Brendel and Nina Gerber. Sichere Instant Messaging Apps. *Datenschutz und Datensicherheit - DuD*, 43(5):276–280, May 2019.

Acknowledgments

This is how you do it:
you sit down at the keyboard and
you put one word after another
until it's done.
It's that easy, and that hard.

Neil Gaiman
(and every Ph.D. student ever)

While writing this thesis did indeed involve lots of dreadful hours spent alone with only an accusingly blank screen as company, the years leading up to it are composed of many, many wonderful memories that I got to share with amazing people, some of whom I would like to thank in the following.

First and foremost I am deeply indebted to my advisor Marc Fischlin. Thank you for taking me on as a Ph.D. student and consistently demonstrating faith in me as a researcher ever since. I was never afraid to speak my mind with you and you demonstrate by example how to be an excellent researcher, team leader, mentor, and teacher. Your advice and guidance over the years has been invaluable. Thank you also for providing me with the opportunity to extensively travel the world and meet wonderful people, not least of all my *Cryptoplexity* family in Darmstadt: Aishwarya Thiruvengadam, Andrea Püchner, Christian Janson, Felix Günther, Felix Rohrbach, Giorgia Azzurra Marson, Jean Paul Degabriele, Patrick Harasser, Sogol Mazaheri, and Tommaso Gagliardoni... what can I say? You know you have amazing colleagues when you are happy to spend your vacation with them and are excited to go to the office every morning... I cannot tell you how much I enjoyed (the many cakes and) all the inspiring, funny, gossipy, and at times serious conversations we had. A heartfelt *Thank You!* for traveling on this road with me (often quite literally) and supporting me throughout.

I especially thank all of my co-authors, Nina Bindel, Denise Demirel, Marc Fischlin, Nina Gerber, Brian Goncalves, Felix Günther, Christian Janson, and Douglas Stebila for tackling interesting problems with me and going through the (sometimes longer than planned) process from an idea to publication. Special thanks goes to Douglas Stebila for kindly hosting me for three months in wonderful Canada in 2018 and always making time to discuss research... or the relative benefits of various Sci-Fi TV shows.

I thank the GRK *Privacy and Trust for Mobile Users* and the CRC *CROSSING* for funding me during my Ph.D., as well as providing an environment with amazing fellow researchers to interact with. I especially thank Ágnes Kiss, Daniel Demmler, Giulia Traverso, Kris Shrishak, Matthias Geihs, Max Maass, and Nina Bindel for many, many nice chats at various occasions over the years.

I would further like to thank Thomas Schneider, Christian Reuter, and Felix Wolf for taking the time to serve on my defense committee. Special thanks goes to my second referee Cas Cremers. It was always good fun talking to you over the years and I am very much looking forward to working with you in Saarbrücken.

I am very grateful to Johannes Buchmann and Denise Demirel. Johannes Buchmann's lectures first introduced me to the magnificent world of cryptography and became my "happy place" during my Master study days which were otherwise filled with lectures containing too many partial differential equations. Denise Demirel who took me on as a working student and co-supervised my Master's thesis in those days has been nothing short of a role model and gave me the final encouragement needed to pursue an academic career. Representation does matter.

Last, but not least, I would like to thank my family and my partner Alex. Without your loving support and endless patience I wouldn't be where and who I am today. You keep me grounded. Alex, you have accompanied me on the hikes of my life for quite a few years now! Thank you so much for always being by my side—be it on tough climbs or leisurely strolls, and irrespective of the physical distance between us.

Jacqueline Brendel
Darmstadt, October 2019

Abstract

Key exchange protocols, first introduced by Diffie and Hellman in 1976, are one of the most widely-deployed cryptographic protocols. They allow two parties, that have never interacted before, to establish shared secrets. These shared cryptographic keys may subsequently be used to establish a secure communication channel. Use cases include the classic client-server setting that is for example at play when browsing the internet, but also chats via end-to-end-encrypted instant messaging applications.

Security-wise, we generally demand of key exchange protocols to achieve *key secrecy* and *authentication*. While, informally, authentication ensures that the communicating parties have confidence in the identity of their peers, key secrecy ensures that any shared cryptographic key that is established via the key exchange protocol is only known to the participants in the protocol and can be used securely in cryptographic protocols, i.e., is sufficiently random. In 1993, Bellare and Rogaway gave a first formalization of key exchange protocol security that captures these properties with respect to powerful adversaries with full control over the network. Their model constitutes the basis of the many subsequent treatments of authenticated key exchange security, including the models presented in this thesis.

The common methodological approach underlying all of these formalizations is the *provable security* paradigm, which has become a standard tool in assessing the security of cryptographic protocols and primitives. So-called *security models* specify the expected security guarantees of the scheme in question with regards to a well-defined class of adversaries. Proofs that validate these security claims do so by reducing the security of the overall scheme to the security of the underlying cryptographic primitives and hardness assumptions. However, advances in computational power and more sophisticated cryptanalytic capabilities often render exactly these components insecure. Especially the advent of quantum computers will have a devastating effect on much of today's public key cryptography. This is especially true for key exchange protocols since they rely crucially on public-key algorithms.

In this thesis, our focus in "future-proofing" key exchange protocols is two-fold. First, we focus on extending security models for key exchange protocols to capture the (un)expected break of cryptographic primitives and hardness assumptions. The aim is to gain assurances with respect to future adversaries and to investigate the effects of primitive failures on key exchange protocols. More specifically, we explore how key exchange protocols can be safely transitioned to new, post-quantum secure algorithms with hybrid techniques. Hybrids combine classical and post-quantum algorithms such that the overall key agreement scheme remains secure as long as one of the two base schemes remains secure. For this, we introduce security notions for key encapsulation mechanisms that account for adversaries with varying levels of quantum capabilities and present three new constructions for hybrid key encapsulation mechanisms. Our hybrid designs are practice-inspired and for example capture draft proposals for hybrid modes in the Transport Layer Security (TLS) protocol, which is one of the most widely-deployed cryptographic protocols that enables key agreement.

Furthermore, our notion of breakdown resilience for key exchange protocols allows to gauge the security of past session keys in the event of a failure of a cryptographic component in the key exchange. We exercise our model on variants of the post-quantum secure key exchange protocol NewHope by Alkim et al. Thereby, we confirm the intuition that, in order to guard against adversaries that only have access to quantum computing power in the (more distant) future, it is sufficient to use classically-secure authentication mechanisms alongside post-quantum key agreement to achieve authenticated key exchange.

As with any mathematical statement, theorems in the provable security paradigm are only as valid as the underlying assumptions. A careful consideration of any newly made assumption is thus essential to ensure the meaningfulness of the statement itself and make the assumption a viable tool for future analyses. Thus, secondly, we systematically classify the PRF-ODH assumption, a complexity-theoretic hardness assumption that has been used in key exchange security analyses of such prominent protocols as TLS, Signal, and Wireguard. In particular, we give a unified, parametrized definition of the assumption encompassing different variants that are present in the literature. We relate the resulting parametrized notions in terms of their strength and show where these assumptions fit in the collection of well-understood related hardness assumptions. We finally sketch our result on the impossibility of instantiating this assumption in the standard model, thereby disposing of the uncertainty in the community whether PRF-ODH is in fact a standard model assumption, i.e., removes the usage of some idealized assumptions in key exchange protocol proofs.

Zusammenfassung

Schlüsselaustauschverfahren wurden erstmals 1976 von Diffie und Hellman vorgestellt und gehören zu den weitverbreitesten kryptografischen Protokollen. Sie ermöglichen zwei Protokollteilnehmern, welche zuvor noch nicht miteinander in Kontakt standen, einen gemeinsamen geheimen kryptografischen Schlüssel abzuleiten. Dieser kann anschließend dazu verwendet werden, einen sicheren Kommunikationskanal zwischen ihnen aufzubauen. Zu den Hauptanwendungsfällen für kryptographischen Schlüsselaustausch zählt die klassische Client-Server-Kommunikation, wie sie beispielsweise beim Surfen im Internet auftritt. Aber auch Ende-zu-Ende-verschlüsselte Chats werden etwa mit Schlüsselaustauschprotokollen abgesichert.

Die Sicherheitseigenschaften, die wir von Schlüsselaustauschverfahren im Allgemeinen erwarten sind zum einen die *Zufälligkeit* und *Vertraulichkeit der ausgehandelten Schlüssel*, sodass diese nur den legitimen Protokollteilnehmern bekannt sind und sich für den sicheren Einsatz in kryptographischen Protokollen eignen. Andererseits soll auch die Identität (einzelner oder aller) Protokollteilnehmer mittels *Authentifizierung* zweifelsfrei sichergestellt werden. Die erste formale Betrachtung dieser zentralen Sicherheitseigenschaften für authentifizierten Schlüsselaustausch geht auf Bellare und Rogaway aus dem Jahre 1993 zurück. Sie setzten damit den Grundstein für viele weitere Sicherheitsdefinitionen für Schlüsselaustauschprotokolle in Gegenwart eines starken Angreifers, der die Kontrolle über das gesamte Netzwerk verfügt. Auch die Arbeiten in dieser Thesis stützen sich im Kern auf diese ursprüngliche Definition und erweitern diese.

Der zugrundeliegende wissenschaftliche Ansatz für all diese Betrachtungen ist das Konzept der *beweisbare Sicherheit*, welches nicht mehr wegzudenken ist aus modernen kryptographischen Sicherheitsanalysen. Sogenannte *Sicherheitsmodelle* spezifizieren die erwarteten Sicherheitseigenschaften des zu analysierenden Verfahrens gegenüber einer wohldefinierten Klasse von Angreifern. Beweise, dass ein Verfahren tatsächlich die definierten Sicherheitsziele erreicht, reduzieren die Sicherheit des Gesamtverfahrens auf die Sicherheit seiner kryptographischen Komponenten.

Unglücklicherweise ist jedoch genau die Sicherheit dieser Komponenten unentwegt durch leistungsfähigere Rechner und neue Techniken in der Kryptanalyse gefährdet. Insbesondere die erwartete technische Revolution durch rechenstarke Quantencomputer droht die moderne Kryptographie stark zu erschüttern. Ein großer Teil der heutzutage verwendeten kryptographischen Algorithmen, die sogenannte *Public-Key-Kryptographie*, die auch in Schlüsselaustauschverfahren zum Einsatz kommt, wird durch die gesteigerte Rechenleistung von Quantencomputern gebrochen. In dieser Arbeit beleuchten wir, wie Schlüsselaustauschverfahren und ihre Sicherheitsanalysen dieser und weiteren Anforderungen der Zukunft gerecht werden können.

Zum einen zeigen wir, wie bestehende Sicherheitsmodelle für Schlüsselaustauschverfahren so erweitert werden können, dass sie dem (un)erwarteten Bruch kryptographischer Primitive und komplexitätstheoretischer Annahmen standhalten können. Ziel ist es hier, die gewünschten Sicherheitsgarantien im Hinblick auf zukünftige Angreiferarten zu definieren und dann zu untersuchen, wie sich der Bruch bestimmter kryptographischer Bausteine auf die Sicherheit von der Protokolle auswirkt.

Konkreter untersuchen wir, wie man derzeit verwendete Algorithmen in Schlüsselaustauschverfahren mittels sogenannter Hybride durch quantencomputerresistente Algorithmen ersetzen kann. Hybride kombinieren die heuzutage verwendeten klassischen Verfahren mit quantencomputerresistenten Verfahren, sodass der Schlüsselaustausch sicher bleibt, solange zumindest eines der zugrundeliegenden Verfahren sicher bleibt. Zu diesem Zweck erweitern wir die bestehenden Sicherheitsmodelle so, dass sie unterschieldich starke Quantenangreifer abbilden können. Zusätzlich präsentieren wir drei praxisnahe Hybridkonstruktionen, die sich an Designvorschlägen für Hybride in einem der am weitesten verbreiteten kryptographischen Protokolle, dem Transport Layer Security (TLS) Protokoll, orientieren.

Unser Breakdown Resilience Modell erlaubt uns weiterhin die Sicherheit von bereits etablierten Schlüsseln zu untersuchen, falls kryptographische Komponenten im Schlüsselaustauschprotokoll in der Zukunft unsicher werden. Wir wenden dieses Modell dann auf Varianten des quantencomputerresistenten Schlüsselaustauschverfahrens NewHope von Alkim et al. an. Unsere Analyse unterstützt nun mit einem formalen Argument die weitherrschende Meinung, dass Schlüsselaustauschverfahren gegen Quantenangreifer in der fernen Zukunft ausreichend geschützt sind, wenn nur die Schlüsselerstellung quantencomputerresistent gestaltet ist, die Authentifizierungmechanismen jedoch nicht.

Wie bei jedem mathematischen Theorem, sind auch Theoreme über die beweisbare Sicherheit von kryptographischen Verfahren nur insofern gültig, als es die darin getroffenen Annahmen sind. Eine strukturierte Untersuchung von bisher unbekannten komplexitätstheoretischen Annahmen in Theoremen ist daher essentiell für deren Richtigkeit und Aussagekraft. Nur dies gewährleistet die bedenkenlose künftige Anwendbarkeit der Annahme selbst, als auch des Theorems.

Dieser Argumentation folgend klassifizieren wir im zweiten Teil der Thesis die komplexitätstheoretische PRF-ODH Annahme, welche sich in Sicherheitsanalysen bedeutender Schlüsselaustauschprotokolle wie TLS, Signal, und Wireguard wiederfindet. Zunächst geben wir hierzu eine vereinheitlichende, parametrisierte Definition der Annahme wieder, welche verschiedene Varianten von PRF-ODH aus der einschlägigen Literatur umfasst. Wir zeigen, wie sich die Annahmen je nach Parameterwahl in ihrer Stärke unterscheiden und wie sie in das Spektrum bereits bekannter, verwandter komplexitätsbasierter Annahmen einzuordnen sind. Seit Einführung von PRF-ODH wurde diskutiert, ob die Verwendung von PRF-ODH es erlaubt, Beweise für Schlüsselaustauschprotokolle ohne idealisierte Annahmen (im sogenannten Standardmodell) zu führen. Zu guter Letzt skizzieren wir unser Unmöglichkeitsresultat, wonach diese Betrachtungsweise unplausibel erscheint und es sich dementsprechend bei PRF-ODH um keine Annahme im Standardmodell handelt.

Contents

Chapter **1**

Introduction

Much of today's digital communication is protected by cryptography - from our everyday web browsing, to instant messaging chats or contactless payments with our banking cards. To efficiently enable secure data transmission, the involved devices must encrypt the confidential data with a so-called *symmetric* encryption scheme for which they need a shared secret key. However, in many scenarios, the parties that wish to communicate securely have no prior knowledge of each other and have, in particular, not communicated before. This is for example the case when your web browser connects to a new website or when you wish to send an encrypted message to a new contact in your messaging application. There is no common information from which a shared secret key could be derived.

Authenticated key exchange protocols, or AKE protocols, for short, solve this issue. On a high level, AKE protocols are run between parties whose goal it is to establish a secure communication channel. To do this, the parties use an unprotected channel to exchange public information from which they are then able to derive a shared secret key. This key can subsequently be used to secure their communication.

Security for AKE protocols essentially consists of two components: *key secrecy* and *authentication*. While, informally, authentication ensures that the two (or more) communicating parties are aware of and agree on the identity of their peer(s), key secrecy ensures that any shared cryptographic key (the *session key*) that is actually established in the interaction, is only known to the parties in the protocol. This latter property is formally captured by demanding that an adversary cannot distinguish an actual key established via the AKE protocol from a random value. The first to formalize these security requirements into a comprehensive model for authenticated key exchange with regards to an actively interfering malicious adversary were Bellare and Rogaway [BR94]. Their model has formed the basis for many AKE security models since, including the ones in this thesis. These models are used to analyse the security of concrete AKE protocol designs.

The most prominent examples for real-world key exchange protocols that have secured the internet and inter-server communications for decades are the Transport Layer Security protocol TLS [Res18] (formerly known as the Secure Sockets Layer protocol SSL) and the Internet Key Exchange protocol IKE [KHN+14] that is part of the IPsec protocol suite [KS05]. In recent years, with the introduction of ever more powerful smartphones, a new application field for key exchange protocols has emerged in the form of end-to-end encrypted instant messaging applications such as Signal or WhatsApp.

Diffie–Hellman key exchange. The first key exchange protocol was introduced by Diffie and Hellman in their 1976 seminal work *New Directions in Cryptography* [DH76], that laid the ground stone of modern public key cryptography. The so-called Diffie–Hellman (DH) key exchange still forms the core of most modern key exchange protocols today. It relies on the hardness of computing *discrete logarithms* in certain mathematical structures called *cyclic groups*. Informally, assume we have some set $\mathbb{G}$ which we call the *group* and this group has "size" q (the *order*), where q is a prime. In *cyclic* groups of order q, there exists an element g (the so-called *generator*) such that every element $X \in \mathbb{G}$ can be uniquely written as g^x, where x is an element in $\mathbb{Z}_q := \{0, 1, \ldots, q-1\}$. Furthermore, $\mathbb{G}$ is closed under multiplication and inversion, i.e., for any two elements $g^x, g^y \in \mathbb{G}$ their multiplicative product $g^x \cdot g^y$ also lies in the group and there exists an element $g^z \in \mathbb{G}$ such that $(g^x)^{-1} = g^z$.

In the basic DH protocol depicted in Figure 1.1, two parties, here called Alice and Bob, each sample a value from $\mathbb{Z}_q$ (their *secret key*). Let a be the value sampled by Alice and b the value sampled by Bob. They then exchange group elements of the form g^a and g^b (we call these their respective *public keys*) over an untrusted channel. From these they can derive a common shared secret $K \leftarrow g^{ab}$, computed by each party taking the other party's public key and raising it to the power of their secret key.

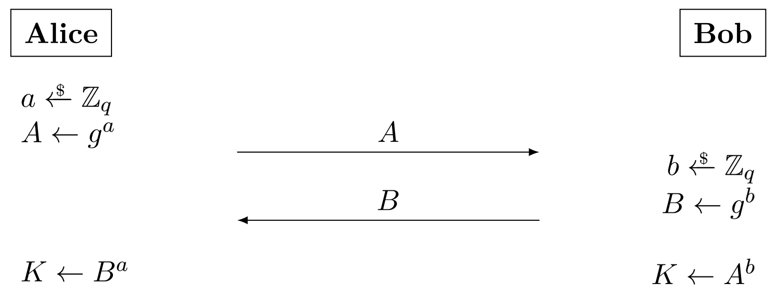

Figure 1.1: Unauthenticated Diffie–Hellman key exchange. Alice and Bob exchange public values A and B to compute the common shared secret $A^b = g^{ab} = B^a$.

This simple version of a key exchange protocol is unfortunately only secure against eavesdropping adversaries that remain passive during the execution of the protocol. Furthermore, the protocol as such does not provide any form of authentication; neither Alice nor Bob have any way of assuring that they are actually talking to each other.

However, modern key exchange protocols build on top of this key agreement scheme and complement it with suitable mechanisms such that both authentication and key secrecy are ensured even against actively interfering adversaries. Later in the thesis, we will discuss the SigMA compiler [Kra03], a popular example of how the basic DH protocol can be lifted to one that achieve security against active adversaries.

Ready for the Future?

Key exchange protocols employ a significant number of cryptographic algorithms and hardness assumptions, to achieve strong security guarantees. At the same time, the steady increase in computational power and advanced cryptanalytic capabilities often renders exactly those cryptographic algorithms insecure.

Quantum computers and cryptography. Especially the advent of large-scale *quantum computers* will have a devastating effect on many currently deployed cryptographic schemes and entire protocols. This is especially relevant for key exchange protocols as the key-generating backbone and thus source of security of many applications. We briefly describe the key concepts of quantum computation in the following. For an in-depth treatment of quantum computation and information we refer the interested reader to, e.g., the textbook by Nielsen and Chuang [NC00].

As opposed to classical computers which are built using digital electronic circuits, quantum computers take advantage of special quantum-mechanical properties of particles. The quantum computer analogue of the bits "0" and "1", that classical computers abstractly operate on, is a *qubit*—a small quantum-mechanical system that can collapse into two so-called *states*, either 0 or 1, when measured.

What is special about quantum computing is that, until such a measurement occurs, a single qubit is able to encode both classical bits *simultaneously*. For example, two qubits represent all two-bit strings "00","01","10", and "11". This *superposition* property allows for a single operation on n qubits to be applied to all n-bit strings in the same processing step.

The second outstanding property of quantum-mechanical systems is that of *entanglement*, which Einstein rather appropriately referred to as "spooky action at a distance". Informally, entanglement describes the process of correlating the states of multiple single qubits such that if the state of one of these qubits is altered (this happens, e.g., through measurement), the states of the other entangled qubits also change—even if they are separated by long distances.

Lastly, quantum states are subject to the so-called *no-cloning theorem* which states that, quite different from classical information, an unknown quantum state cannot be copied without "destroying" it, i.e., causing it to collapse.

The theoretical model of quantum computation was first introduced by Feynman [Fey82] in 1982. By now, quantum computers are no longer hypothetical constructs since small versions have been built successfully [Goo, IBM] and much research effort is invested into developing more powerful and stable systems. As hinted at before, especially for cryptography this ongoing progress in building large-scale quantum computers is rather bad news.

Over three decades ago, Shor [Sho94, Sho97] presented the first efficient algorithm to solve the integer factorization (and discrete logarithm) problem using hypothetical quantum computing power. He was able to transform the problem of integer factorization into a problem of finding a period in a function, i.e., finding the interval at which sequences of function outputs repeat. The above mentioned superposition property makes it possible to evaluate the function on all points simultaneously, thus yielding an efficient means to determine the period.

Unfortunately, most of today's public key cryptography is based on the intractability of either integer factorization or computing discrete logarithms and due to Shor's findings must thus be considered entirely broken once quantum computers operate reliably on sufficiently many qubits. For symmetric cryptography the situation is somewhat better, despite Grover [Gro96, Gro01] presenting a quantum algorithm for search in an unstructured database with quadratic speed-up over the best-known classical algorithms. From a cryptographic point of view this corresponds to speeding up the brute-force computation of an n-bit secret key input of a symmetric cryptographic primitive such as block cipher encryption from 2^n to $2^{\frac{n}{2}}$ operations. Simply doubling the key sizes retains the current security levels of most schemes (assuming Grover's algorithm is the only viable quantum attack against those primitives).

Sadly, such an easy fix is not available for public key cryptography and the vulnerable schemes must be replaced entirely by schemes that are secure even in the presence of quantum computing power. These cryptographic schemes have become known as *post-quantum* cryptography.[1] These problems are mostly based on lattice theory, codes, multivariate polynomial equations, or supersingular elliptic curves. Currently, the American National Institute of Standards and Technology (NIST) is heading the effort to standardize post-quantum cryptographic schemes [Nat15], with 26 "Round 2" candidates (of 69 accepted submissions in the first round) and standard documents expected in the early 2020s [Moo19].

[1]Note that post-quantum algorithms are merely classical algorithms (meaning they are implementable on classical computers just like our schemes today) but rely on problems that are hard to solve even for quantum computers. This stands in contrast to quantum cryptography that actively leverages the above mentioned principles of quantum physics.

Cryptanalysis. But not only quantum computers threaten our cryptography. Increasing classical computing power and advanced cryptanalysis render schemes insecure regularly. Examples include weak ciphers like RC4 [GMPS14, ABP$^+$13] or OCB2 [IIMP19] and collisions in still widely-deployed hash functions like MD5 [dB94, WY05, SLdW07] or SHA-1 [WYY05, Ste13, SKP16, SBK$^+$17].

This degradation in security over time is often expected and safe alternatives are available before the actual breakdown. Reality, however, sadly tells the story of very slow adoption rates for new algorithms and protocol versions which leads to widespread deployment of insecure algorithms long after they have been broken. The reasons for this are manifold but among the most prominent are (a combination of) virtually un-updatable legacy systems, apprehensive developers that must maintain backwards-compatibility, and security-illiterate decision-makers. With their many employed primitives and complex ecosystems in which they are embedded, key exchange protocols are especially prone to obsolete algorithms and protocol versions.

Provable security caveats. But even if key exchange protocols employ secure cryptographic algorithms as building blocks, this still does not mean that the protocol as a whole is secure. Especially modern key exchange protocols that aim for fine-grained security guarantees use intricate patterns to achieve both security and high efficiency with respect to communication and computation. This makes their security *analysis*—i.e., the rigorous argumentation that the protocol actually provides key secrecy and authentication—often complex and necessitates the usage of what is often referred to as *non-standard* cryptographic assumptions.

It has become common to subject (newly proposed) cryptographic schemes and protocols to a rigorous analysis in the *provable security* paradigm, which we will discuss in Chapter 2. While an indispensable tool in the development of new cryptographic schemes and algorithms, the resulting theorems about the security of a scheme inherit the caveat of any proven mathematical statement: the claim is only as viable as the underlying assumptions.

To capture complex protocols designs in the provable security framework, it often becomes necessary to deviate from what has become standard and is considered to be well-understood. While this is not problematic in itself, any new assumption requires a careful categorization into the realm of existing assumptions that have been deemed reasonable. Without relating the new hardness assumptions to tenable intractabilities, the entire security statement is at risk of becoming void if the validity of the new assumptions cannot be upheld. Furthermore it is beneficial to show that known cryptographic primitives do actually achieve this new assumption.

Organization of the Thesis and Contributions

The research efforts that constitute the basis of this thesis focus on the “future-proof” design and analysis of key exchange protocols that takes into account the above mentioned challenges and scenarios.

On the one hand, this includes advanced security models for the analysis of key exchange protocols that can capture future adversaries with increased computational and cryptanalytic power. In particular, we examine how key exchange protocols can be safely transitioned into a post-quantum setting and which security guarantees can be maintained for secret keys that were established before one (or more) of the cryptographic primitives employed in the KE was broken in the meantime. On the other hand, we consider a relatively novel non-standard intractability assumption that has founds its way into numerous security analyses of widely-deployed key exchange protocols. We strengthen these existing (and future) analyses by the systematic and careful categorization of this assumption.

In the following, we briefly outline the thesis and the main motivations and contributions. Each of the main chapters (Chapters 5, 6, and 7), contains more precise pointers to my own personal contributions, as well as an extensive discussion of related work, so we defer the discussion of these. Note that the joint paper with Denise Demirel on the unrelated topic of efficient proactive secret sharing [BD16], as well as the work on a provably-secure zero round-trip time extension to the Extended Access Control protocol with Marc Fischlin [BF17] have not entered this thesis.

Background. Chapters 2, 3, and 4 give the necessary background for the rest of the thesis. In Chapter 2 we introduce the methodological principle of *provable security* that builds the basis of many areas of cryptological research. We discuss how security of a cryptographic scheme can be defined in terms of the inability of a computationally-bounded adversary to win a so-called security game with greater than vanishingly small probability. We further introduce one of the most commonly used idealized models in cryptography, the *random oracle model*, which allows to prove security of schemes that otherwise escape analysis.

In Chapter 3 we specify the definitions and security games for the most relevant basic cryptographic building blocks that are used within this thesis. These include public key encryption and the related concept of key encapsulation mechanisms, as well as signatures, message authentication codes, hash functions, and pseudorandom functions. We furthermore introduce a class of complexity-theoretic hardness assumptions that are often applied to prove security of Diffie–Hellman-based key exchanges. These well-established assumptions will especially be relevant as reference points for the categorization of the interactive Diffie-Hellman-based $\mathsf{PRF\text{-}ODH}$ assumption in Chapter 7.

In Chapter 4, we finally introduce the Bellare-Rogaway model [BR94], which is one of the (if not the) most utilized models for analyzing security of authenticated key exchange protocols. It will form the core security model for our more advanced security notions of authenticated key exchange in the hybrid setting in Chapter 5, as well as for the breakdown resilience framework in Chapter 6.

Hybrid key exchange. In Chapter 5, we turn towards the question of how a smooth transition to key exchange protocols that are secure against quantum adversaries can be ensured without introducing unnecessary risks or even vulnerabilities. The circumstances that make such a transition challenging (apart from the previously mentioned slow adoption rates that typically come with new algorithms and protocol versions) are two-fold:

On the one hand, today's communication is already vulnerable to so-called *future-quantum* adversaries. These adversaries may record encrypted communications along with the preceding key exchanges today and will break confidentiality once quantum computers become available which allows them to extract the key from the stored (now broken) key exchange. On the other hand, with the NIST standardization process for post-quantum cryptography [Nat15] still ongoing, we are not yet confident in the appropriate choice of post-quantum schemes, and especially for lattice-based approaches, adequate parameter selection is unresolved [ACD+18].

So-called *hybrid* schemes offer a way out of this dilemma. These schemes combine today's classically-secure key exchange with experimental post-quantum secure schemes such that security is maintained even if one of these two base schemes breaks down completely. We model hybrid KE in the two-stage adversary setting introduced by Bindel et al. [BHMS17] and are thus able to achieve a fine-grained security analysis wrt. adversaries of varying quantum powers.

We propose three new hybrid key encapsulation mechanisms which are especially suitable for practice since they are either highly efficient or capture proposals for hybrid modes in TLS 1.3. Finally, we present a model for hybrid AKE based on the Bellare-Rogaway model and show how to generically build hybrid AKE from hybrid key encapsulation mechanisms.

Breakdown resilience of key exchange protocols. In Chapter 6 we introduce the notion of *breakdown resilience* for key exchange protocols. As already mentioned before, key exchange protocols rely on a significant number of cryptographic primitives and hardness assumptions. Unfortunately, failures of actively deployed primitives and hardness assumptions are common; be it weak ciphers, collisions in hash functions, or poor Diffie-Hellman parameter choices.

While the notions of *forward secrecy* [Gün90, DVOW92, CK01] and *post-compromise security* [CCG16] give security guarantees for key exchanges when (some of the) secret material of the participants is exposed, so far there has been no formal tool to analyze the security of a key exchange protocol in the face of the failure of an entire cryptographic component—neither retrospectively, nor in the anticipation of a future breakdown.

The breakdown resilience model closes this gap and presents an extension to the Bellare-Rogaway model which allows to generically model breakdowns and their effect on the security of past key exchanges. We furthermore present a stronger version which is able to argue about the (in)security of even ongoing and future sessions. While in general every component of a key exchange scheme is necessary for its security, the aforementioned hybrid designs have built-in redundancy to account for breakdowns.

To exercise our model we present an analysis of the post-quantum secure unauthenticated key exchange protocol NewHope [ADPS16b, AAB^{+}19] combined with classical authentication in the breakdown resilience framework. The analysis confirms the intuition that for key exchanges to remain secure against future-quantum adversaries, it is sufficient to replace the key exchange part with a post-quantum secure solution and rely on classical authentication mechanisms only. To show that the stronger notion of breakdown resilience can capture hybrid AKE, we analyze the security of a generic key exchange from hybrid KEMs and achieve results comparable to those presented in Chapter 5.

The PRF-ODH assumption. In Chapter 7 we turn our attention from key exchange security models to the underlying complexity-theoretic hardness assumptions. More specifically, we investigate the PRF-ODH assumption, which has found its way into many DH-based key exchange analyses since its usage in the TLS 1.2 security analysis by Jager et al. [JKSS12]. Informally, PRF-ODH guarantees pseudorandomness of a function value $\mathsf{PRF}(g^{uv}, x^{\star})$ even if the DH shares g^u and g^v, as well as related values $\mathsf{PRF}(S^u, x)$ and/or $\mathsf{PRF}(T^v, x)$ are known to the adversary.

Despite its widespread usage in different variations, a consolidating definition and a comprehensive classification of PRF-ODH into the realm of well-understood DH-type assumptions had been lacking. We closed this gap by providing a unifying definition, yielding nine different notions of the PRF-ODH assumption. We related these assumptions with respect to their strength and, perhaps most importantly, showed to which well-established DH problem they relate to.

We then show that HMAC [BCK96, KBC97, NIS08], the main building block of the popular key derivation function HKDF [Kra10, KE10], achieves the strongest form of PRF-ODH security that supports multiple related PRF values on both key shares.

We close the chapter by disposing of the uncertainty whether PRF-ODH is a standard model assumption. For a while it seemed like the usage of PRF-ODH might enable standard model proofs without random oracles for key exchange protocols that had previously necessitated them. Perhaps sadly, we were able to show via the meta-reduction technique (the proof of which we only sketch in this thesis) that even the mildest notions –where the adversary is allowed to learn only a single related PRF value– may not be based on general hard cryptographic problems without random oracles using standard techniques.

Chapter 2

Methodology

In this chapter we introduce our foundational method, the paradigm of *provable security*. We mostly focus on the notions and arguments most relevant to this thesis. For a more in-depth discussion of the paradigm itself (and its potential controversies) we refer the interested reader to, e.g., the technical overview by Alexander Dent [Den06a], on which this part of the chapter is loosely based, and the works of Koblitz and Menezes [KM04, KM06, KM19] and Goldreich [Gol06]. The introduction to the game-hopping technique at the end of this chapter is mostly based on the tutorial by Shoup [Sho04], which also contains many reference examples.

2.1 Notation

We start by fixing some notation. Let λ denote the security parameter. We assume that the security parameter is always known to the adversary $\mathcal{A}$ and will thus omit it as explicit input to $\mathcal{A}$. For algorithms we give the security parameter in its unary representation 1^λ as input. This allows to compute the complexity of the algorithm as a function of the input length and thus aids the definition of efficiency of algorithms.

For a bit string $x \in \{0,1\}^\star$, we denote the length of x by $|x|$. Furthermore, $x\|y$ denotes the concatenation of strings x and y. By $x \leftarrow y$ we describe the assignment of value y to some variable x and by $s \xleftarrow{\$} \mathcal{D}$ we denote the sampling of an element s from the probability distribution $\mathcal{D}$. Note that for a set S we denote the sampling from the uniform distribution on S simply by $s \xleftarrow{\$} S$.

For algorithms Alg, we denote by $y \xleftarrow{\$} \mathsf{Alg}(x)$ the probabilistic execution of algorithm Alg on input x with output y. If we want to stress that Alg is a deterministic algorithm, we write $y \leftarrow \mathsf{Alg}(x)$ instead. Furthermore, $\mathsf{Alg}^{\mathcal{O}}(x)$ denotes that algorithm Alg has access to oracle $\mathcal{O}$ during its execution on input x.

For a security game $\mathsf{G}^{\mathsf{sec\text{-}prop}}_{\Pi,\mathcal{A}}(\lambda)$ capturing the security property $\mathsf{sec\text{-}prop}$ of a cryptographic scheme Π against an adversary $\mathcal{A}$, we write $\mathsf{G}^{\mathsf{sec\text{-}prop}}_{\Pi,\mathcal{A}}(\lambda) = 1$ to denote the event in which $\mathcal{A}$ has won the game. By $[\![\mathsf{statement}]\!]$ we denote the Boolean evaluation of $\mathsf{statement}$, i.e., it corresponds to 1 if the statement is true and 0 otherwise.

2.2 Overview

In a nutshell, the aim of provable security is to show the security of cryptographic schemes in a mathematically rigorous way. The first formal definition of security of a cryptosystem has been given by Shannon [Sha49] in his 1949 *Communication Theory of Secrecy Systems*, where he specified what it means for a symmetric encryption scheme to be *perfectly secure*. The definition was inspired by his previous observations on information theory and entropy, published the year before [Sha48]. Unfortunately, information theory gives rise to a very strong notion of secrecy that is hard to fulfill for practical cryptographic schemes: security must be achieved against an unbounded adversary that has access to unlimited computational power and storage.

In 1976, Diffie and Hellman published the idea of public key cryptography [DH76] which was not based on the principles of information theory, but on computationally-intractable problems, i.e., problems that are hard to solve when given only limited resources. Their invention gave rise to a new conceptualization of security, where adversaries are not all-powerful as in Shannon's interpretation. Rabin gave a first security proof in this setting [Rab79], showing that their proposed schemes were as hard as the problem of factorization. The formal framework was then established by Goldwasser and Micali [GM82, GM84] when they introduced probabilistic encryption and thus broadened the concept from the deterministic to the probabilistic setting.

A lot of cryptographic schemes at that time and in the following years were built in an ad-hoc fashion. This resulted in an unsatisfactory cycle of a scheme being first attacked, finally broken, and then fixed, just to be attacked, broken, and fixed again. The provable security paradigm provided some relief to this situation, as the well-defined security goals and (somewhat) mathematical proofs in the very least offered some assurance about the basic soundness of the designs.

Let us look at how security analyses in this paradigm proceed. The statements that provable security analyses provide are essentially of the following form:

> If some cryptographic scheme Π can be *broken* by an *efficient* algorithm $\mathcal{A}$ with *non-negligible advantage*, then there exists an efficient *reduction* $\mathcal{B}$, that can break some computationally-intractable problem also with non-negligible advantage.

Thus, given an adequate description of the cryptographic scheme Π in question, we need the following basic ingredients to show that Π is indeed "provably secure":

- a well-defined class A of adversaries $\mathcal{A}$ that we aim to guard against.
- a definition of the security property $\mathsf{sec\text{-}prop}$ that shall be achieved by Π against A. In particular, this includes a clear winning condition, i.e., what it means for an adversary $\mathcal{A} \in \mathsf{A}$ to *break* $\mathsf{sec\text{-}prop}$.
- a proof that no adversary $\mathcal{A} \in \mathsf{A}$ can achieve the winning condition for Π unless $\mathcal{A}$ can solve some cryptographic problem(s) that is assumed to be hard from a complexity-theoretic viewpoint.

2.3 Classes of Adversaries

Following what has just been discussed, adversaries can roughly be divided into two subclasses, depending on whether the adversary is assumed to be computationally bounded or not. Schemes proven secure in the latter setting, i.e., for *information-theoretic* security as introduced by Shannon, are not prone to advances in cryptanalysis or computational power. However, as mentioned before, only very little (practical) schemes are able to achieve information-theoretic security.

Most security definitions today are thus *computational*, i.e., based on complexity-theoretic hardness assumptions that are intractable to solve for efficient adversaries. Efficiency is defined with respect to *probabilistic polynomial time algorithms*, and we will later, in Section 2.4, define what it means for a problem to be intractable for a given algorithm. But first, let us clarify the notions of algorithms and their efficiency.

2.3.1 Mathematical Models of Computation

In cryptography, there are two mathematical models of computation which are considered most relevant: *Turing machines* and *circuits*. We will focus our attention on these in accordance with the scope of the thesis, but note that further, equally powerful models exist.

Turing machines. Turing machines are an abstract mathematical model of machines that was introduced by Alan Turing in 1936. They consist of an infinitely long *memory tape*, that is subdivided into *cells*. Each cell can hold a symbol σ from an alphabet Σ. In our setting, we consider the alphabet $\Sigma = \{0, 1\}$, i.e., each symbol is either a bit 0 or 1. The cells can be written or read by a head that can pass over the tape by moving one position to the left or the right. The instructions, i.e., the movements and actions of the head are given by a finite *program*. In the beginning, a Turing machine $\mathcal{M}$ receives some finite input $x \in \Sigma^\star$, where $\Sigma^\star$ commonly denotes the set of all strings over the alphabet Σ. In case $\mathcal{M}$ is deterministic, the actions and thus the output $y \leftarrow \mathcal{M}(x)$ on input x are uniquely determined by the program. Probabilistic programs, where the output and actions not only depend on the input, but also on randomness, are modeled by introducing an additional, infinitely-long tape that is filled with random bits prior to the start of the execution.

Circuits. Turing machines are a so-called *uniform* algorithmic model, i.e., the machine executes the same instructions irrespective of the input length. Circuits on the other hand can be *non-uniform*, i.e., their computations may vary with the length of the input. This is why formally, we do not speak of a single circuit, but rather a family $\mathcal{C}$ of circuits $\{C_\lambda\}_\lambda$ over the input length λ. We say that a family of circuits $\mathcal{C} = \{C_\lambda\}_\lambda$ is *uniform*, if there exists a Turing machine $\mathcal{M}$ that on input 1^λ outputs a description of the circuit C_λ, i.e., $C_\lambda \leftarrow \mathcal{M}(1^\lambda)$. A circuit consists of a finite sequence of *gates*, which are operations on two input bits (with one of them potentially fixed) and one output bit. Typical examples of such gates are NOT, AND, OR, NAND, or XOR operations.

In the following discussions, we use the model of Turing machines (unless stated otherwise). Note that every family of circuits can be represented by a Turing machine that gets an additional input $z = z(|x|)$ that only depends on the length of the input x.

2.3.2 Efficient Algorithms

It must still be clarified, what it means for an algorithm (for example our adversary $\mathcal{A}$) to be *efficient*. Informally, an algorithm or a function is efficient if it only consumes a "reasonable" amount of resources. For Turing machines this is typically measured in *run time*, i.e., the number of operations until $\mathcal{M}$ stops with output y on input x. A reasonable run time is then specified with the help of upper-bounding polynomials in the input length. More formally:

Definition 2.1 (Efficient algorithm)**.** *We call a (probabilistic) Turing machine $\mathcal{M}$* efficient *or* (probabilistic) polynomial time (PPT)*, if the run time of $\mathcal{M}$ on input x is upper-bounded by a polynomial p in the input length $|x|$.*

This gives us the opportunity to define what an efficiently computable function is:

Definition 2.2 (Efficiently computable function)**.** *We call a deterministic function $f : \{0,1\}^\star \to \{0,1\}^\star$* efficiently computable*, if there exists an efficient deterministic Turing machine $\mathcal{M}$ such that $f(x) = \mathcal{M}(x)$ for all $x \in \{0,1\}^\star$.*

For probabilistic functions such as random variables we demand the existence of a probabilistic Turing machine whose output distribution is identical to the output distribution of the function.

2.4 Security Models

We are now ready to concretely define the security of a cryptographic scheme Π. In the realm of manual, computational proofs there are again two main approaches to formalize security: *game-based* and *simulation-based* definitions.

In the game-based setting, which we consider in this thesis, security is defined via a game that is played between two PPT algorithms, an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$. After the parameters of the game are set up by the challenger, the adversary receives inputs and may then interact further with the game via so-called *oracles* (thereby capturing executions of the scheme and possible interferences of the adversary). At some point, the adversary stops with some output. Along with the game description, there is a well-defined winning condition, which captures the cases where the adversary has (non-trivially) won the game.

In the simulation-based approach, the adversary $\mathcal{A}$ gets to interact with an environment $\mathcal{E}$ (to which for example other users belong) and through that accesses either the real scheme or an *idealized* version of the scheme that is per definition unbreakable. The adversary then needs to decide, which of the two it was interacting with. An example of this is the *universal composability* (UC) framework by Canetti [Can01]. This approach yields generally stronger security guarantees as it takes into account the larger picture of the environment of the scheme. However, as for example Canetti and Fischlin [CF01] showed for the UC framework, there exist classes of cryptographic functions (in more detail, two-party universally composable commitments) that cannot be proven secure in the UC framework without further assumptions.

Negligible success probability. In both versions, game-based and simulation-based, the security of a scheme is defined as the inability of the adversary to successfully either win the game or distinguish the two worlds, except with some vanishingly small probability, respectively. We define this vanishingly small probability via so-called *negligible* functions, which are functions that approach zero faster than the inverse of any polynomial. More formally:

Definition 2.3 (Negligible function)**.** *We call a function $\epsilon : \mathbb{N} \to \mathbb{R}$* negligible*, if for every polynomial $p : \mathbb{N} \to \mathbb{R}^+$ there exists an $N \in \mathbb{N}$, such that for all $n \geq N$ we have $\epsilon(n) \leq \frac{1}{p(n)}$.*

For cryptographic discussions we consider polynomials in the security parameter λ. In our proofs, we will implicitly make use of the fact that the sum of negligible functions is still negligible. We note that furthermore subtracting a negligible function from a non-negligible function still yields a non-negligible function, as summarized in the following proposition:

Proposition 2.4 (Properties of negligible functions)**.** *Let $\epsilon(n)$ and $\epsilon_i(n)$ be negligible functions and $\delta(n)$ be non-negligible. Let further $q, q_i : \mathbb{N} \to \mathbb{R}^+$ be polynomials for $i \in \mathbb{N}$. Then the following holds:*

1. *The sum $\sum_{i=1}^{k} q_i(n)\epsilon_i(n)$ is negligible for constant $k \in \mathbb{N}$.*
2. *The function $\delta(n) - \epsilon(n)$ is non-negligible.*

Proof. The proof is a straightforward application of the definitions of (non-)negligibility. □

2.5 Security Proofs

Lastly, given the adversarial model (PPT adversaries) as well as a definition of security (game-based), we want to show that a given scheme Π actually accomplishes this notion. Security is shown via *reductions*, which reduce the security of the scheme in question to some underlying hard cryptographic problem(s). The goal is to compute the success probability of the adversary in winning the game in relation to some target probability. For example, in a game that requires distinguishing two cases, the adversary can always win this game with probability $\frac{1}{2}$ by simply guessing. We are interested in how much *better* the adversary can do than guessing. We call this the adversary's *advantage*.

Usually, it is hard to accurately determine the adversary's advantage in a given security game due to its complexity. This is where the so-called *game-hopping technique* comes into play (cf., e.g., [Sho04, Den06b]). The idea is to gradually modify the security game that the adversary is playing until we arrive at a situation where the adversary can do no better than the target probability, i.e., where its advantage is essentially zero. For each modification of the game we upper-bound the adversary's advantage in detecting this change.

This technique is the key tool for most of the computational proofs presented in this thesis. There are multiple different ways to "hop" from one game to the next:

Based on indistinguishability. In this transition, we replace some value x that is given to the adversary $\mathcal{A}$ (either as initial input or as some oracle response) by a value $\widetilde{x}$ which is drawn from a different distribution that can however not be distinguished by either computationally-bounded or unbounded adversaries.[2]

It is then shown that if the adversary $\mathcal{A}$ could efficiently distinguish these two games, then this would imply the existence of a distinguisher $\mathcal{D}$ between the two probability distributions, thus contradicting the assumption.

Based on negligible failure events. Here, the two games proceed identically, unless a certain failure event E occurs. This means in particular, that the success probability of the adversary in both cases is the same if E does *not* occur and the difference between them is upper-bounded by the probability of the event E. It is then shown that this probability is in fact negligible. This is either done via an information-theoretic argument or by reduction to some hard complexity-theoretic problem.

Based on large failure events. This transition, pointed out by Dent [Den06b] can come up both in settings where the security notion is based on indistinguishability or on computational notions. Here, some event E may occur in Game G_i, but the probability of the adversary in succeeding in G_i is independent of the occurrence of it. The probability of E *not* occurring is non-negligible (in contrast to the previous game hop transition). In computational notions, we abort Game G_{i+1} if E occurs (otherwise the game proceeds as game G_i), thus causing the adversary to lose. For indistinguishability-based notions of security, we do not abort G_{i+1} whenever E occurs, but instead sample a random bit as the output of the adversary (thereby diminishing its advantage in distinguishing to zero). In both cases, the difference in advantage of the adversary is upper bounded by the probability of E not happening.

Bridging steps. These transitions are of a mere syntactical nature where some steps are replaced by equivalent computations. In particular, the success probability of the adversary remains the same in both games. These changes are mostly done to prepare for one of the aforementioned transitions.

[2]For the latter case of adversaries we then speak of *statistical indistinguishability*, whereas computationally-bounded adversaries give rise to the notion of *computational indistinguishability*.

2.6 The Random Oracle Model

Lastly, we introduce the random oracle model. Many constructions that could be proven with the above discussed method had one significant shortage: they were inefficient, which led to the unsatisfactory situation that schemes in practice had no security arguments whatsoever. By introducing idealized versions of primitives with additional features it became possible to give security arguments for efficient constructions that had so far eluded provable security analysis. When the schemes were implemented in practice, the random oracle would be instantiated by a “sufficiently good” hash function.

The random oracle model was introduced by Bellare and Rogaway in 1993 [BR93]. Intuitively, a *random oracle* RO implements a truly random function. It constitutes an idealized version of a hash function, in that it is a public object that is accessible to everybody. The outputs of random oracles are consistent (i.e., repeated queries elicit the same response) and the responses are uniformly distributed and independent of one another.

Random oracles can be modeled either through iterative filling of the input-output table (*lazy sampling*), i.e., on every new input x sample a new output value y, or by choosing a random function up front and thereby determining the entire input-output behavior at once. Some of the resulting properties of the random oracle make it especially useful for proofs via reductions, where a reduction algorithm has to simulate some game environment for the adversary, that it runs as sub-routine. These properties are the following:

- **Indistinguishability:** if some value x has not been queried to the random oracle, then $\mathsf{RO}(x)$ is indistinguishable from random.

- **Extractability:** the reduction sees all queries that the adversary poses to the random oracle.

- **Programmability:** the reduction may choose specific answers to the adversary’s queries as long as they are correctly (i.e., uniformly) distributed.

In [CGH98], Canetti, Goldreich, and Halevi showed that there exist signature and encryption schemes that are secure in the random oracle model but for which any practical instantiation of the random oracle results in insecure schemes. Despite this impossibility result, the random oracle model has become a success story and is by now a standard tool in achieving security proofs for practical schemes.

Proofs that do not employ random oracles are referred to as *standard model* proofs, if the distinction is to be made explicit. As for any strong assumption it is preferable to not employ them if not absolutely necessary, i.e., standard model proofs are what should be aimed for. However, it is widely acknowledged that proofs in the ROM are still preferable to having no argument about the soundness of a design at all.

It is no wonder that over the years, security models and proofs have become essential in the design and analysis of cryptographic primitives, schemes, and entire protocols. As a word of caution, one may add that provable security only offer security assurances when the schemes in question are used as analyzed, i.e., unmodified, in the considered environment, and against the specified class of adversaries.

It is an ongoing quest to push these mathematical security models and assumptions as close to what is happening in reality as possible and thus to arrive at increasingly insightful and relevant statements for practically deployed schemes and their users. The works presented in this thesis aim to contribute to this effort.

In the next section we will introduce the relevant cryptographic building blocks and Diffie–Hellman-type assumptions for our thesis. The security of the cryptographic primitives is defined in terms of games with respect to probabilistic polynomial time algorithms.

Chapter 3

Cryptographic Building Blocks

This chapter is dedicated to the collection of definitions for the basic cryptographic building blocks that we will be using throughout the thesis, mostly following the introductory textbook to cryptography by Katz and Lindell [KL14]. This chapter serves as a reference point mostly and can be skipped on first reading.

3.1 Public Key Encryption and Key Encapsulation

Definition 3.1 ((Public key) encryption (PKE)). *A* public key encryption scheme $\mathcal{E}$ *defined over some finite message space $\mathcal{M}$ and ciphertext space $\mathcal{C}$ is a triple of algorithms* $(\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ *such that*

- ***key generation*** KGen *takes as input the security parameter λ and outputs a public/secret-key pair $(pk, sk) \xleftarrow{\$} \mathsf{KGen}(1^\lambda)$,*
- ***encryption*** Enc *takes as input a public key pk and a message $m \in \mathcal{M}$ and outputs a ciphertext $c \in \mathcal{C}$, i.e., $c \xleftarrow{\$} \mathsf{Enc}(pk, m)$,*
- ***decryption*** Dec *takes a secret key sk and ciphertext $c \in \mathcal{C}$ and outputs either the decrypted message $m \leftarrow \mathsf{Dec}(sk, c)$ in $\mathcal{M}$ or the dedicated symbol $\perp$, in case of decryption failure.*

We say that a public key encryption scheme $\mathcal{E} = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ is ϵ-correct*, if*

$$\Pr\Big[m' \neq m : (pk, sk) \xleftarrow{\$} \mathsf{KGen}(1^\lambda), c \xleftarrow{\$} \mathsf{Enc}(pk, m), m' \leftarrow \mathsf{Dec}(sk, c)\Big] \leq \epsilon.$$

If $\epsilon = 0$, we call $\mathcal{E}$ (perfectly) correct*.*

Security of public key encryption schemes $\mathcal{E}$ is based on the *indistinguishability* of ciphertexts under either passive *chosen plaintext* adversaries ($\mathsf{IND\text{-}CPA}$ security) or active *chosen ciphertext* adversaries ($\mathsf{IND\text{-}CCA}$ security):

Definition 3.2 (Security of (public key) encryption). *Let $\mathcal{E} = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ be a public key encryption scheme and let $\mathcal{A}$ be a PPT algorithm. Consider the games $\mathsf{G}^{\mathsf{ind\text{-}atk}}_{\mathcal{E},\mathcal{A}}(\lambda)$ described in Figure 3.1, where $\mathsf{atk} \in \{\mathsf{cpa}, \mathsf{cca}\}$.*

We define the advantage function of an adversary $\mathcal{A}$ against game $\mathsf{G}^{\mathsf{ind\text{-}atk}}_{\mathcal{E},\mathcal{A}}$ as

$$\mathsf{Adv}^{\mathsf{ind\text{-}atk}}_{\mathcal{E},\mathcal{A}}(\lambda) := \left|\Pr\left[\mathsf{G}^{\mathsf{ind\text{-}atk}}_{\mathcal{E},\mathcal{A}}(\lambda) = 1\right] - \frac{1}{2}\right|.$$

We say that a public key encryption scheme $\mathcal{E}$ is $\mathsf{IND\text{-}ATK}$ secure *for $\mathsf{ATK} \in \{\mathsf{CPA}, \mathsf{CCA}\}$, if for any PPT adversary $\mathcal{A}$ this advantage function is negligible in the security parameter λ.*

```
$\underline{\mathsf{G}^{\mathsf{ind\text{-}cpa}}_{\mathcal{E},\mathcal{A}}(\lambda)}$:
$(pk, sk) \xleftarrow{\$} \mathsf{KGen}(1^\lambda)$
$m_0, m_1, st \xleftarrow{\$} \mathcal{A}(pk)$
$b \xleftarrow{\$} \{0,1\}$
$c^\star \xleftarrow{\$} \mathsf{Enc}(pk, m_b)$
$b' \xleftarrow{\$} \mathcal{A}(st, c^\star)$
$\textbf{return } [\![b' = b]\!]$
```

```
$\underline{\mathsf{G}^{\mathsf{ind\text{-}cca}}_{\mathcal{E},\mathcal{A}}(\lambda)}$:
$(pk, sk) \xleftarrow{\$} \mathsf{KGen}(1^\lambda)$
$m_0, m_1, st \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{\mathsf{Dec}}}(pk)$
$b \xleftarrow{\$} \{0,1\}$
$c^\star \xleftarrow{\$} \mathsf{Enc}(pk, m_b)$
$b' \xleftarrow{\$} \mathcal{A}^{\mathcal{O}^\perp_{\mathsf{Dec}}}(st, c^\star)$
$\textbf{return } [\![b' = b]\!]$
```

```
$\underline{\mathcal{O}_{\mathsf{Dec}}(c)}$:
$\textbf{return } \mathsf{Dec}(sk, c)$
$\underline{\mathcal{O}^\perp_{\mathsf{Dec}}(c)}$:
$\textbf{if } c = c^\star$
    $\textbf{return } \perp$
$\textbf{else}$
    $\textbf{return } \mathsf{Dec}(sk, c)$
```

Figure 3.1: IND-CPA and IND-CCA security of $\mathcal{E} = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ over message space $\mathcal{M}$ and ciphertext space $\mathcal{C}$ for $m_0, m_1 \in \mathcal{M}$ with $|m_0| = |m_1|$.

Definition 3.3 (Key encapsulation mechanism (KEM))**.** *A key encapsulation mechanism $\mathcal{K}$ with ciphertext space $\mathcal{C}$ and key space $\mathscr{K}$ is a tuple of algorithms $\mathcal{K} = (\mathsf{KGen}, \mathsf{Encaps}, \mathsf{Decaps})$ such that*

- ***key generation*** KGen *takes as input the security parameter λ and outputs a public/secret-key pair, i.e., $(pk, sk) \xleftarrow{\$} \mathsf{KGen}(1^\lambda)$,*
- ***encapsulation*** Encaps *takes as input a public key pk and outputs a ciphertext $c \in \mathcal{C}$ and the therein encapsulated key $K \in \mathscr{K}$ i.e., $(c, K) \xleftarrow{\$} \mathsf{Encaps}(pk)$,*
- ***decapsulation*** Decaps *takes a ciphertext c and secret key sk and outputs either the decapsulated key $K \leftarrow \mathsf{Decaps}(sk, c)$ or the dedicated symbol $\perp$, in case of decapsulation failure.*

*We say that a key encapsulation mechanism $\mathcal{K} = (\mathsf{KGen}, \mathsf{Encaps}, \mathsf{Decaps})$ is ϵ-*correct *if*

$$\Pr\Big[K' \neq K : (pk, sk) \xleftarrow{\$} \mathsf{KGen}(1^\lambda), (c, K) \xleftarrow{\$} \mathsf{Encaps}(pk), K' \leftarrow \mathsf{Decaps}(sk, c)\Big] \leq \epsilon.$$

We call $\mathcal{K}$ (perfectly) correct *if $\epsilon = 0$.*

Definition 3.4 (Security of key encapsulation mechanisms)**.** *The security of a key encapsulation mechanism $\mathcal{K} = (\mathsf{KGen}, \mathsf{Encaps}, \mathsf{Decaps})$ with key space $\mathscr{K}$ is defined in terms of* indistinguishability of encapsulated keys from random *with respect to either passive* chosen plaintext *adversaries* (IND-CPA *security*) *or active* chosen ciphertext *adversaries* (IND-CCA *security*)*. The respective security games are depicted in Figure 3.2.*

We say that $\mathcal{K}$ is IND-ATK secure *with* ATK $\in \{\mathsf{CPA}, \mathsf{CCA}\}$*, if for every PPT adversary $\mathcal{A}$ the advantage function in winning the game $\mathsf{G}^{\mathsf{ind\text{-}atk}}_{\mathcal{K},\mathcal{A}}(\lambda)$ with* $\mathsf{atk} \in \{\mathsf{cpa}, \mathsf{cca}\}$ *defined as*

$$\mathsf{Adv}^{\mathsf{ind\text{-}atk}}_{\mathcal{K},\mathcal{A}}(\lambda) := \left|\Pr\left[\mathsf{G}^{\mathsf{ind\text{-}atk}}_{\mathcal{K},\mathcal{A}}(\lambda) = 1\right] - \frac{1}{2}\right|$$

is negligible in the security parameter λ.

Remark 3.5. In the proofs in this thesis, we will often encounter games, where the bit is not chosen randomly by the challenger during the game execution but fixed up front. This is an equivalent way of defining security by measuring the difference in adversarial behaviour depending whether the bit is 0 or 1. In the following, we exemplify this on the definition for IND-ATK security of key encapsulation mechanisms, but the same applies to all kinds of cryptographic experiments where the game execution is the same from the view of the adversary in both experiments, whereas the challenger behaves differently depending on the bit b. We will see that when transitioning between the two notions of security, we lose a factor of 2, which will be the case in many of our security proofs.

```
$\mathsf{G}^{\mathsf{ind\text{-}cpa}}_{\mathcal{K},\mathcal{A}}(\lambda)$:
    $(pk, sk) \xleftarrow{\$} \mathsf{KGen}(1^\lambda)$
    $(c^\star, K_0^\star) \xleftarrow{\$} \mathsf{Encaps}(pk)$
    $K_1^\star \xleftarrow{\$} \mathscr{K}$
    $b \xleftarrow{\$} \{0,1\}$
    $b' \xleftarrow{\$} \mathcal{A}(pk, c^\star, K_b^\star)$
    return $[\![b' = b]\!]$
```

```
$\mathsf{G}^{\mathsf{ind\text{-}cca}}_{\mathcal{K},\mathcal{A}}(\lambda)$:
    $(pk, sk) \xleftarrow{\$} \mathsf{KGen}(1^\lambda)$
    $(c^\star, K_0^\star) \xleftarrow{\$} \mathsf{Encaps}(pk)$
    $K_1^\star \xleftarrow{\$} \mathscr{K}$
    $b \xleftarrow{\$} \{0,1\}$
    $b' \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{\mathsf{Decaps}}}(pk, c^\star, K_b^\star)$
    return $[\![b' = b]\!]$
```

```
$\mathcal{O}_{\mathsf{Decaps}}(c)$:
    if $c = c^\star$
        return $\perp$
    else
        return $\mathsf{Decaps}(sk, c)$
```

Figure 3.2: IND-CPA and IND-CCA security of $\mathcal{K} = (\mathsf{KGen}, \mathsf{Encaps}, \mathsf{Decaps})$ with key space $\mathscr{K}$.

The security games with fixed bits are depicted in Figure 3.3. We then say that $\mathcal{K}$ is IND-ATK *secure* with $\mathsf{ATK} \in \{\mathsf{CPA}, \mathsf{CCA}\}$, if for every PPT adversary $\mathcal{A}$ the advantage function, denoted by $\mathsf{Adv}^{\mathsf{ind\text{-}atk\text{-}fixed}}_{\mathcal{K},\mathcal{A}}(\lambda)$, is defined as

$$\mathsf{Adv}^{\mathsf{ind\text{-}atk\text{-}fixed}}_{\mathcal{K},\mathcal{A}}(\lambda) := \left|\Pr\left[\mathsf{G}^{\mathsf{ind\text{-}atk\text{-}0}}_{\mathcal{K},\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathsf{G}^{\mathsf{ind\text{-}atk\text{-}1}}_{\mathcal{K},\mathcal{A}}(\lambda) = 1\right]\right|$$

is negligible in the security parameter λ, where $\mathsf{G}^{\mathsf{ind\text{-}atk\text{-}}b}_{\mathcal{K},\mathcal{A}}(\lambda) = 1$ denotes that the adversary's output and thus the output of the game is 1.

```
$\mathsf{G}^{\mathsf{ind\text{-}cpa\text{-}}b}_{\mathcal{K},\mathcal{A}}(\lambda)$:
    $(pk, sk) \xleftarrow{\$} \mathsf{KGen}(1^\lambda)$
    $(c^\star, K_0^\star) \xleftarrow{\$} \mathsf{Encaps}(pk)$
    $K_1^\star \xleftarrow{\$} \mathscr{K}$
    $b' \xleftarrow{\$} \mathcal{A}(pk, c^\star, K_b^\star)$
    return $b'$
```

```
$\mathsf{G}^{\mathsf{ind\text{-}cca\text{-}}b}_{\mathcal{K},\mathcal{A}}(\lambda)$:
    $(pk, sk) \xleftarrow{\$} \mathsf{KGen}(1^\lambda)$
    $(c^\star, K_0^\star) \xleftarrow{\$} \mathsf{Encaps}(pk)$
    $K_1^\star \xleftarrow{\$} \mathscr{K}$
    $b' \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{\mathsf{Decaps}}}(pk, c^\star, K_b^\star)$
    return $b'$
```

```
$\mathcal{O}_{\mathsf{Decaps}}(c)$:
    if $c = c^\star$
        return $\perp$
    else
        return $\mathsf{Decaps}(sk, c)$
```

Figure 3.3: Alternative definition with fixed-bit games of IND-CPA and IND-CCA security of $\mathcal{K} = (\mathsf{KGen}, \mathsf{Encaps}, \mathsf{Decaps})$ with key space $\mathscr{K}$.

Proposition 3.6. *Let* $\mathcal{K} = (\mathsf{KGen}, \mathsf{Encaps}, \mathsf{Decaps})$ *be a key encapsulation mechanism with key space* $\mathscr{K}$*. For every adversary* $\mathcal{A}$ *it holds that*

$$\mathsf{Adv}^{\mathsf{ind\text{-}atk\text{-}fixed}}_{\mathcal{K},\mathcal{A}}(\lambda) = 2 \cdot \mathsf{Adv}^{\mathsf{ind\text{-}atk}}_{\mathcal{K},\mathcal{A}}(\lambda).$$

Proof. To show this, we first note that if we condition the event that $\mathcal{A}$ wins Game $\mathsf{G}^{\mathsf{ind\text{-}atk}}_{\mathcal{K},\mathcal{A}}(\lambda)$ on the events that $b = 0$ and $b = 1$, respectively, we have

$$\Pr\left[b' = 1 \middle| b = 0\right] = \Pr\left[\mathsf{G}^{\mathsf{ind\text{-}atk\text{-}0}}_{\mathcal{K},\mathcal{A}}(\lambda) = 1\right]$$

and

$$\Pr\left[b' = 1 \middle| b = 1\right] = \Pr\left[\mathsf{G}^{\mathsf{ind\text{-}atk\text{-}1}}_{\mathcal{K},\mathcal{A}}(\lambda) = 1\right].$$

So we have

$$\begin{aligned}
\Pr\left[\mathsf{G}^{\text{ind-atk}}_{\mathcal{K},\mathcal{A}}(\lambda)=1\right] &= \Pr\left[b'=b\right] \\
&= \Pr\left[b=0\right]\cdot\Pr\left[b'=b\middle|b=0\right]+\Pr\left[b=1\right]\cdot\Pr\left[b'=b\middle|b=1\right] \\
&= \frac{1}{2}\cdot\Pr\left[b'=0\middle|b=0\right]+\frac{1}{2}\cdot\Pr\left[b'=1\middle|b=1\right] \\
&= \frac{1}{2}\cdot\left(1-\Pr\left[b'=1\middle|b=0\right]\right)+\frac{1}{2}\cdot\Pr\left[b'=1\middle|b=1\right] \\
&= \frac{1}{2}\cdot\left(1-\Pr\left[\mathsf{G}^{\text{ind-atk-}0}_{\mathcal{K},\mathcal{A}}(\lambda)=1\right]+\Pr\left[\mathsf{G}^{\text{ind-atk-}1}_{\mathcal{K},\mathcal{A}}(\lambda)=1\right]\right).
\end{aligned}$$

To summarize with advantages, we have

$$\begin{aligned}
\mathsf{Adv}^{\text{ind-atk}}_{\mathcal{K},\mathcal{A}}(\lambda) &= \left|\Pr\left[\mathsf{G}^{\text{ind-atk}}_{\mathcal{K},\mathcal{A}}(\lambda)=1\right]-\frac{1}{2}\right| \\
&= \left|\frac{1}{2}\cdot\left(1-\Pr\left[\mathsf{G}^{\text{ind-atk-}0}_{\mathcal{K},\mathcal{A}}(\lambda)=1\right]+\Pr\left[\mathsf{G}^{\text{ind-atk-}1}_{\mathcal{K},\mathcal{A}}(\lambda)=1\right]\right)-\frac{1}{2}\right| \\
&= \frac{1}{2}\cdot\left|\Pr\left[\mathsf{G}^{\text{ind-atk-}1}_{\mathcal{K},\mathcal{A}}(\lambda)=1\right]-\Pr\left[\mathsf{G}^{\text{ind-atk-}0}_{\mathcal{K},\mathcal{A}}(\lambda)=1\right]\right| \\
&= \frac{1}{2}\cdot\mathsf{Adv}^{\text{ind-atk-fixed}}_{\mathcal{K},\mathcal{A}}(\lambda),
\end{aligned}$$

which completes the proof. □

3.2 Signatures and Message Authentication Codes

Definition 3.7 (Signature scheme)**.** *A (digital)* signature scheme $\mathcal{S}$ *consists of three algorithms* $(\mathsf{KGen},\mathsf{Sign},\mathsf{Vfy})$ *such that*

- ***key generation*** KGen *takes as input the security parameter* λ *and outputs a public/secret-key pair* $(pk,sk)\xleftarrow{\$}\mathsf{KGen}(1^\lambda)$*,*
- ***signing*** Sign *takes as input a secret key* sk *as well as a message* $m\in\{0,1\}^\star$ *that is to be signed and outputs a signature* $\sigma\xleftarrow{\$}\mathsf{Sign}(sk,m)$*,*
- ***verification*** Vfy *takes as input a public key* pk*, signature* σ*, and message* m*. It outputs* 1*, if* σ *is a* valid *signature over* m *wrt.* pk *else it outputs* 0*, indicating failure to verify.*

We say that a signature scheme $\mathcal{S}=(\mathsf{KGen},\mathsf{Sign},\mathsf{Vfy})$ *is* correct *if*

$$\Pr\Big[0\leftarrow\mathsf{Vfy}(pk,\sigma,m):(pk,sk)\xleftarrow{\$}\mathsf{KGen}(1^\lambda),\sigma\xleftarrow{\$}\mathsf{Sign}(sk,m)\Big]$$

is negligible in the security parameter λ*.*

The security of signature schemes is defined as existential unforgeability of signatures under chosen message attacks:

Definition 3.8 (Security of signatures)**.** *Let* $\mathcal{S}=(\mathsf{KGen},\mathsf{Sign},\mathsf{Vfy})$ *be a signature scheme and let* $\mathcal{A}$ *be a PPT algorithm. Consider the security game* $\mathsf{G}^{\text{euf-cma}}_{\mathcal{S},\mathcal{A}}(\lambda)$ *as defined in Figure 3.4*

We say that a signature scheme $\mathcal{S}$ *is* EUF-CMA secure *or* existentially unforgeable under chosen message attacks*, if for any PPT adversary* $\mathcal{A}$ *the advantage function defined as*

$$\mathsf{Adv}^{\text{euf-cma}}_{\mathcal{S},\mathcal{A}}(\lambda):=\Pr\left[\mathsf{G}^{\text{euf-cma}}_{\mathcal{S},\mathcal{A}}(\lambda)=1\right]$$

is negligible in the security parameter λ*.*

```
$\mathsf{G}^{\text{euf-cma}}_{\mathcal{S},\mathcal{A}}(\lambda)$:
$\mathcal{L}_{\mathcal{O}_{\mathsf{Sign}}} \leftarrow \emptyset$
$(pk, sk) \xleftarrow{\$} \mathsf{KGen}(1^\lambda)$
$(\sigma', m') \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{\mathsf{Sign}}}(pk)$
return $[\![\mathsf{Vfy}(pk, \sigma', m') \wedge m' \notin \mathcal{L}_{\mathcal{O}_{\mathsf{Sign}}}]\!]$

$\mathcal{O}_{\mathsf{Sign}}(m)$:
$\mathcal{L}_{\mathcal{O}_{\mathsf{Sign}}} \leftarrow \mathcal{L}_{\mathcal{O}_{\mathsf{Sign}}} \cup \{m\}$
return $\mathsf{Sign}(sk, m)$
```

Figure 3.4: Existential unforgeability under chosen message attacks of $\mathcal{S} = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vfy})$.

Definition 3.9 (Message authentication code (MAC))**.** *A message authentication code scheme $\mathcal{M}$ with key space $\mathcal{K}$ consists of three algorithms* $\mathsf{KGen}, \mathsf{MAC}$*, and* Vfy *such that*

- ***key generation*** $\mathsf{KGen}(1^\lambda)$ *takes as input the security parameter λ and outputs a key $K \in \mathcal{K}$,*
- ***tag computation*** $\mathsf{MAC}(K, m)$ *takes as input a key $K \in \mathcal{K}$ and message $m \in \{0,1\}^\star$ and outputs a MAC (tag) $\tau \xleftarrow{\$} \mathsf{MAC}(K, m)$,*
- ***verification*** $\mathsf{Vfy}(K, \tau, m)$*, takes as input a key $K \in \mathcal{K}$, tag τ, and message m. It outputs* 1 *if τ is a* valid *MAC over m, else it outputs* 0*, indicating failure to verify.*

The unforgeability of a message authentication scheme $\mathcal{M} = (\mathsf{KGen}, \mathsf{MAC}, \mathsf{Vfy})$ is defined analogously to the unforgeability of signature schemes and is depicted in Figure 3.5.

```
$\mathsf{G}^{\text{euf-cma}}_{\mathcal{M},\mathcal{A}}(\lambda)$:
$\mathcal{L}_{\mathcal{O}_{\mathsf{MAC}}} \leftarrow \emptyset$
$K \xleftarrow{\$} \mathsf{KGen}(1^\lambda)$
$(m', \tau') \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{\mathsf{MAC}}}(\cdot)$
return $[\![\mathsf{Vfy}(K, \tau', m') \wedge m' \notin \mathcal{L}_{\mathcal{O}_{\mathsf{MAC}}}]\!]$

$\mathcal{O}_{\mathsf{MAC}}(m)$:
$\tau \xleftarrow{\$} \mathsf{MAC}(m)$
$\mathcal{L}_{\mathcal{O}_{\mathsf{MAC}}} \leftarrow \mathcal{L}_{\mathcal{O}_{\mathsf{MAC}}} \cup \{m\}$
return $\tau$
```

Figure 3.5: Existential unforgeability under chosen message attacks of $\mathcal{M} = (\mathsf{KGen}, \mathsf{MAC}, \mathsf{Vfy})$.

3.3 Hash Functions

Definition 3.10 ((Cryptographic) hash function)**.** *A keyed (cryptographic) hash funcion $\mathcal{H}$ with key space $\mathcal{K}$ and output length l consists of a pair of algorithms* $(\mathsf{KGen}, \mathsf{H})$ *such that*

- ***key generation*** $\mathsf{KGen}(1^\lambda)$ *takes as input the security parameter λ and outputs a key $K \in \mathcal{K}$, and*
- ***the hash function*** $\mathsf{H}(K, x)$ *takes as input a key K, string $x \in \{0,1\}^\star$ and outputs a string $y \in \{0,1\}^l$, i.e., $y \leftarrow \mathsf{H}(K, x)$.*

Definition 3.11 (Collision resistance)**.** *Let $\mathcal{H} = (\mathsf{KGen}, \mathsf{H})$ be a keyed hash function and let $\mathcal{A}$ be a PPT algorithm. Let $\mathsf{G}^{\text{coll-res}}_{\mathcal{H},\mathcal{A}}(\lambda)$ be the game defined in Figure 3.6. We say that a hash function $\mathcal{H}$ is* collision resistant*, if for any PPT adversary $\mathcal{A}$ the advantage function defined as*

$$\mathsf{Adv}^{\text{coll-res}}_{\mathcal{H},\mathcal{A}}(\lambda) := \Pr\left[\mathsf{G}^{\text{coll-res}}_{\mathcal{H},\mathcal{A}}(\lambda) = 1\right]$$

is negligible in the security parameter λ.

```
$\mathsf{G}^{\mathsf{coll\text{-}res}}_{\mathcal{H},\mathcal{A}}(\lambda)$:
1 $K \xleftarrow{\$} \mathsf{KGen}(1^\lambda)$
2 $(x, x') \xleftarrow{\$} \mathcal{A}(K)$
3 return $[\![x \neq x' \wedge \mathsf{H}(K, x) = \mathsf{H}(K, x')]\!]$
```

Figure 3.6: Collision resistance of a hash function $\mathcal{H} = (\mathsf{KGen}, \mathsf{H})$.

Remark 3.12. In the rest of the thesis we will mostly omit the key generation and keying of the hash function since many hash functions in practice are unkeyed. We simply identify the hash function $\mathcal{H}$ with H unless stated otherwise, i.e., we will for example write $\mathsf{H}(x)$ for the hash function evaluation at x instead of $\mathsf{H}(K, x)$, or talk about the collision resistance of H.

3.4 Pseudorandom Functions

Definition 3.13 (Pseudorandom function (PRF))**.** *Let* $\mathsf{F} : \{0,1\}^{\kappa(\lambda)} \times \{0,1\}^{\iota(\lambda)} \to \{0,1\}^{\omega(\lambda)}$ *be an efficient keyed function with key length* $\kappa(\lambda)$*, input length* $\iota(\lambda)$ *and output length* $\omega(\lambda)$*. Let* $\mathsf{G}^{\mathsf{prf\text{-}sec}}_{\mathsf{F},\mathcal{A}}(\lambda)$ *be defined as in Figure 3.7. We call* F *a* pseudorandom function *if for any PPT adversary the advantage function defined as*

$$\mathsf{Adv}^{\mathsf{prf\text{-}sec}}_{\mathsf{F},\mathcal{A}}(\lambda) := \left| \Pr\left[\mathsf{G}^{\mathsf{prf\text{-}sec}}_{\mathsf{F},\mathcal{A}}(\lambda) = 1\right] - \frac{1}{2} \right|$$

is negligible in the security parameter λ*.*

```
$\mathsf{G}^{\mathsf{prf\text{-}sec}}_{\mathsf{F},\mathcal{A}}(\lambda)$:
1 $K \xleftarrow{\$} \{0,1\}^{\kappa(\lambda)}$
2 $g \xleftarrow{\$} \{\text{functions } f : \{0,1\}^{\iota(\lambda)} \to \{0,1\}^{\omega(\lambda)}\}$
3 $b \xleftarrow{\$} \{0,1\}$
4 $b' \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{\mathsf{PRF}}}$
5 return $[\![b' = b]\!]$

$\mathcal{O}_{\mathsf{PRF}}(x)$:
6 if $b = 0$
7     return $\mathsf{F}(K, x)$
8 else
9     return $g(x)$
```

Figure 3.7: Pseudorandomness of a function F.

3.5 Diffie–Hellman Assumptions

In the course of the thesis, we will make use of various number-theoretic assumptions called *Diffie–Hellman* assumptions. These problems, underlying many of today's DH-based protocols, will prove especially relevant for categorizing the $\mathsf{PRF\text{-}ODH}$ assumption in Chapter 7. In the following, let p and q be large primes such that q divides $p-1$. Let $\mathbb{G}$ be a cyclic group of order q, i.e., there exists a generator g such that $\mathbb{G} = \{g^n : n \in \mathbb{Z}_q\}$, where $\mathbb{Z}_q = \{0, 1, \ldots, q-1\}$.

Roughly speaking, hard problems that we base security on, can be divided into three categories:

- **Computational problems:** given some problem instance P and a relation R, find a solution S such that $R(P, S) = 1$. An example of this is the computational DH problem (CDH) that asks to compute g^{uv}, given g^u and g^v.
- **Decisional problems:** given a pair (P, S) decide if $R(P, S) = 1$ or not. An example of this is the decisional DH problem (DDH), where given g^u, g^v, g^z it must be decided if $g^z = g^{uv}$ or $g^z \xleftarrow{\$} \mathbb{G}$.

- **Gap problems:** this class of problems was first introduced by Okamoto and Pointcheval [OP01] and is useful in analyzing schemes whose underlying hardness relies on a *computational problem in the presence of a decisional oracle*, that can leak one bit of information, i.e., that may leak whether a certain relation is fulfilled or not.

  In this thesis, we make use of two DH-based Gap assumptions. The first is the Gap Diffie–Hellman problem (GapDH) that asks to compute g^{uv} given g^u, g^v and an oracle $\mathsf{DDH}(\cdot,\cdot,\cdot)$ that on input g^x, g^y, g^z returns 1 if $g^z = g^{xy}$, and 0 otherwise. A slightly weaker version is the strong Diffie–Hellman problem (StDH), where the computational problem is the same as for GapDH, but the oracle is restricted by a fixed first input $\mathsf{DDH}(g^x,\cdot,\cdot)$, such that an adversary may only learn whether $g^z = (g^y)^x$.

The game-based descriptions for all the DH problems mentioned above are depicted in Figure 3.8. As usual, these problems are said to be *hard* if there exists no PPT adversary $\mathcal{A}$ that can solve them with non-negligible advantage.

```
$\mathsf{G}^{\mathsf{CDH}}_{\mathbb{G},\mathcal{A}}(\lambda)$:
$x, y \xleftarrow{\$} \mathbb{Z}_q$
$Z \xleftarrow{\$} \mathcal{A}(g^x, g^y)$
$[\![Z = g^{xy}]\!]$
```

```
$\mathsf{G}^{\mathsf{DDH}}_{\mathbb{G},\mathcal{A}}(\lambda)$:
$x, y, z \xleftarrow{\$} \mathbb{Z}_q$
$y^\star_0 \leftarrow g^{xy}$
$y^\star_1 \leftarrow g^{z}$
$b \xleftarrow{\$} \{0,1\}$
$b' \xleftarrow{\$} \mathcal{A}(g^x, g^y, y^\star_b)$
return $[\![b' = b]\!]$
```

```
$\mathsf{G}^{\mathsf{StDH}}_{\mathbb{G},\mathcal{A}}(\lambda)$:
$x, y \xleftarrow{\$} \mathbb{Z}_q$
$Z \xleftarrow{\$} \mathcal{A}^{\mathsf{DDH}(g^x,\cdot,\cdot)}(g^x, g^y)$
return $[\![Z = g^{xy}]\!]$

$\mathsf{DDH}(g^x, g^y, g^z)$:
return $[\![g^{xy} = g^z]\!]$
```

```
$\mathsf{G}^{\mathsf{GapDH}}_{\mathbb{G},\mathcal{A}}(\lambda)$:
$x, y \xleftarrow{\$} \mathbb{Z}_q$
$Z \xleftarrow{\$} \mathcal{A}^{\mathsf{DDH}(\cdot,\cdot,\cdot)}(g^x, g^y)$
return $[\![Z = g^{xy}]\!]$

$\mathsf{DDH}(g^x, g^y, g^z))$:
return $[\![g^{xy} = g^z]\!]$
```

Figure 3.8: Diffie-Hellman problems: computational DH, decisional DH, strong DH, and Gap DH.

Chapter 4

Key Exchange Security

The theoretical foundation for the study of key exchange security was laid by Bellare and Rogaway in 1993 in their seminal work *Entity Authentication and Key Distribution* [BR94]. After this first formalization of security for key exchange, many more models for key exchange followed: from the asymmetric setting [BWM98, BWJM97], over three-party protocols [BR95] and password-based key exchanges [BPR00, BMP00], to models capturing yet stronger classes of adversaries [BCK96, Sho99, CK02, LLM07]. Countless further refinements and extensions for more sophisticated security assurances in special settings have been proposed, e.g., [JKSS12, FG14, BBF$^+$16]. In this chapter, we focus on the presentation of the (variant of the) Bellare–Rogaway model for authenticated key exchange [BR94] (or BR model for short) that constitutes the basis for all AKE security models proposed in this thesis.

4.1 Security Requirements

As already mentioned, the two main requirements for authenticated key exchange protocols are key secrecy and authentication. Key secrecy (or sometimes *indistinguishability*) captures the notion that an adversary cannot efficiently distinguish a key which has been established between honest parties from a random key if it is still *fresh* (i.e., the key has not become trivially known to the adversary). Authentication, on the other hand, can come in multiple flavors: key exchanges can be *anonymous* (with no party authenticated), *unilaterally* authenticated (one party authenticated), or *mutually* authenticated (both authenticated).

The results in this thesis focus on the case of mutually authenticated key exchange protocols between two parties with *pre-specified peers* as introduced by Canetti and Krawczyk [CK02], i.e., the identity of the intended partner of a session is specified upon session creation.

We furthermore distinguish between protocols providing and not providing *forward secrecy* [Gün90, DVOW92]. Protocols that achieve forward secrecy guarantee security for sessions that have been established *before* the compromise of a party's long-term secrets. Forward secrecy has become a standard design goal of modern key exchange protocols and can be achieved by, e.g., not solely relying on long-term secrets to establish the session key but by mixing ephemeral secrets that are generated freshly for each session into the key derivation.

The spiritually related notion of *post-compromise security* (PCS) has recently been put forward by Cohn-Gordon, Cremers, and Garratt [CCG16]. PCS describes the ability of a key exchange protocol to *recover* from a party's compromise, i.e., security of session keys can be regained. We do not incorporate post-compromise security in our models here, but note that PCS is gaining recognition as an important feature in fine-grained key exchange security. For example, it has been stated as an explicit design goal in the development of a standard for secure (group) messaging by the IETF working group on Message Layer Security [BBM$^+$19] (albeit achieving strong PCS guarantees presents some difficulties in the group messaging setting as pointed out by Cremers, Hale, and Kohbrok [CHK19]).

4.2 The Bellare–Rogaway Model for Authenticated Key Exchange

The BR security model [BR94] (and its variants) provide strong security guarantees for authenticated key exchange in the presence of an active adversary. As formalized in the following, the adversary interacts with protocol instances via oracle queries. The goal is to distinguish the real session key established in a so-called test session from a randomly chosen one; the test session is chosen by the adversary via the Test oracle and must fulfill certain criteria such that it is not trivial for the adversary to distinguish the respective key (more on this *freshness* criteria later). The adversary is considered to have full control over the network, in particular it may inject, drop, and alter messages. This is modeled via a Send oracle that delivers messages specified by the adversary from and to key exchange sessions. The adversary is furthermore able to corrupt some of the parties' long-term secrets (via a $\mathsf{Corrupt}$ oracle) and to reveal some of the established session keys (via a Reveal oracle). The following model description here is largely taken verbatim from Brendel, Fischlin, and Günther [BFG17].

Notation and overview. The participants in a key exchange protocol KE are given by elements U from the set of users $\mathcal{U}$, each of whom holds a long-term public key pk_U with corresponding secret key sk_U. Each participant can act as initiator or responder of a protocol execution and may run multiple instances (*sessions*) of the key exchange protocol in parallel. To uniquely refer to the k-th session owned by user $U \in \mathcal{U}$ with intended communication partner $V \in \mathcal{U}$ on an administrative level, we use the notation $\pi^k_{U,V}$. Each such session is associated with the following set of variables:

- $\mathsf{role} \in \{\mathsf{initiator}, \mathsf{responder}\}$ indicates the session owner's role in this session.
- $\mathsf{st}_{\mathsf{exec}} \in \{\mathsf{running}, \mathsf{accepted}, \mathsf{rejected}\}$ indicates the current state of execution. The default value upon creation of the session is $\mathsf{running}$.
- $\mathsf{sid} \in \{0,1\}^\star \cup \{\bot\}$ indicates the session identifier. The default value is $\bot$.
- $\mathsf{st}_{\mathsf{key}} \in \{\mathsf{fresh}, \mathsf{revealed}\}$ indicates the state of the session key K. The default value is fresh.
- $\mathsf{K} \in \{0,1\}^\star \cup \{\bot\}$ denotes the established session key. The default value is $\bot$.
- $\mathsf{tested} \in \{\mathsf{true}, \mathsf{false}\}$ indicates whether the session key K has been tested or not. The default value for each key is false.

To be able to refer to a specific entry for a session $\pi^k_{U,V}$, we use the notation $\pi^k_{U,V}.\mathsf{entry}$. For example, $\pi^k_{U,V}.\mathsf{role}$ specifies the session owner U's role in session $\pi^k_{U,V}$. For simplicity, we sometimes simply write π and π' to refer to sessions in a general context where the specific indices do not matter.

Partnering of Sessions. The partnering of sessions is defined via session identifiers [BPR00]. More precisely, we call the session $\pi^k_{U,V}$ owned by U *partnered* with the session $\pi^{k'}_{V',U'}$ owned by V' (and vice versa), if the sessions share the same session identifier, i.e., whenever

$$\pi^k_{U,V}.\mathsf{sid} = \pi^{k'}_{V',U'}.\mathsf{sid} \neq \bot.$$

We require that any execution between honest instances that has not been tampered with by the adversary is partnered.

4.2.1 Adversary Model

As usual, we model the adversary as a probabilistic polynomial time Turing machine denoted by $\mathcal{A}$. The adversary is active and in full control over the network. This implies in particular that—additional to the interception of messages—the adversary can schedule when (and if) message delivery occurs. Furthermore, the adversary may alter and inject messages. We assume the adversary learns if a participant in the protocol has terminated and/or accepted.

Adversarial Queries. Recall that in order to break key secrecy, the goal of the adversary is to distinguish real from random session keys. However, not all interactions of the adversary with the protocol are admissible at any given point in time. In particular, there are conditions under which the adversary trivially loses the game, e.g., when both revealing and testing session keys of partnered sessions as mentioned before. To keep track if one of these special cases has occurred, we introduce a flag lost, which is initialized to false. The adversary interacts with the protocol via the following oracle queries:

$\mathsf{NewSession}(U, V, role)$**:** Establishes a new session $\pi^k_{U,V}$ for U (with k being the next counter value for sessions owned by U with intended partner V), stores the given role $role \in \{\mathsf{initiator}, \mathsf{responder}\}$ in $\pi^k_{U,V}.\mathsf{role} \leftarrow role$, and returns the identifier $\pi^k_{U,V}$.

$\mathsf{Send}(\pi^k_{U,V}, m)$**:** Causes the message m to be sent to the session with label $\pi^k_{U,V}$. If there exists no session $\pi^k_{U,V}$, the query outputs $\perp$. Else the response of the session owner U upon receipt of message m is returned, and the state of execution $\mathsf{st}_{\mathsf{exec}}$ is updated. If $\mathsf{st}_{\mathsf{exec}}$ changes to $\mathsf{accepted}$ with an intended communication partner V that was previously corrupted, then set $\mathsf{st}_{\mathsf{key}} \leftarrow \mathsf{revealed}$.

$\mathsf{Reveal}(\pi^k_{U,V})$**:** Returns the session key K of session $\pi^k_{U,V}$. If there exists no session $\pi^k_{U,V}$ or if $\pi^k_{U,V}.\mathsf{st}_{\mathsf{exec}} \neq \mathsf{accepted}$, then return $\perp$. Otherwise, $\mathsf{st}_{\mathsf{key}}$ is set to $\mathsf{revealed}$ and K is returned to the adversary.

$\mathsf{Corrupt}(U)$**:** Returns the long-term secret key sk_U of U to the adversary. No further queries may be issued to sessions owned by U. If forward secrecy is not considered, $\mathsf{st}_{\mathsf{key}}$ is set to $\mathsf{revealed}$ in all sessions $\pi^k_{V,W}$ where $V = U$ or $W = U$.

$\mathsf{Test}(\pi^k_{U,V})$**:** Tests the session key K of session $\pi^k_{U,V}$. The oracle uses a test bit $b_{\mathsf{test}} \xleftarrow{\$} \{0, 1\}$ chosen at the outset and then fixed during the game execution.

For simplicity, we restrict the adversary to ask a single Test query only. If there exists no session $\pi^k_{U,V}$ or if $\pi^k_{U,V}.\mathsf{st}_{\mathsf{exec}} \neq \mathsf{accepted}$, the query returns $\perp$. Otherwise, $\pi^k_{U,V}.\mathsf{tested}$ is set to true. If $b_{\mathsf{test}} = 0$, K_{test} is set to the actual session key $\pi^k_{U,V}.\mathsf{K}$. If $b_{\mathsf{test}} = 1$, a key $K_{\mathsf{test}} \xleftarrow{\$} \mathcal{D}$ is sampled at random from the session key space $\mathcal{D}$. Finally K_{test} is returned.

4.2.2 Bellare–Rogaway AKE Security Games

We adopt the approach of Brzuska et al. [BFWW11, Brz13] and separate the overall BR security properties into the notions of $\mathsf{BR\text{-}Match}$ security and BR key secrecy. The conditions of $\mathsf{BR\text{-}Match}$ security guarantee that

- partnered sessions hold the same key,

- the session identifiers sid ensure an appropriate identification of partnered sessions,
- and that at most two sessions are partnered.

BR key secrecy then ensures that the protocol in question establishes session keys that are indistinguishable from random keys and (implicitly) mutually authenticated. The definition excludes trivial attacks of the adversary such as distinguishing revealed session keys from random keys.

Definition 4.1 (BR-Match security)**.** *Let* KE *be a key exchange protocol, and* $\mathcal{A}$ *be a PPT adversary interacting with* KE *via the queries defined in Section 4.2.1 in the following game* $\mathsf{G}^{\mathsf{BR\text{-}Match}}_{\mathsf{KE},\mathcal{A}}(\lambda)$*:*

Setup. *The challenger generates long-term public/secret-key pairs for each participant* $U \in \mathcal{U}$*.*

Query. *The adversary* $\mathcal{A}$ *receives the generated public keys and has access to the oracle queries* $\mathsf{NewSession}$*,* Send*,* Reveal*,* $\mathsf{Corrupt}$*, and* Test*.*

Stop. *At some point, the adversary stops with no output.*

We say that $\mathcal{A}$ *wins the game* $\mathsf{G}^{\mathsf{BR\text{-}Match}}_{\mathsf{KE},\mathcal{A}}(\lambda)$ *if at least one of the following conditions holds:*

1. Different session keys in partnered sessions:

   There exist two distinct sessions π *and* π' *with* $\pi.\mathsf{sid} = \pi'.\mathsf{sid} \neq \bot$*, and* $\pi.\mathsf{st}_{\mathsf{exec}}, \pi'.\mathsf{st}_{\mathsf{exec}} \neq \mathsf{rejected}$*, but* $\pi.\mathsf{K} \neq \pi'.\mathsf{K}$*.*

2. Different intended partner in partnered sessions:

   There exist two sessions $\pi := \pi^{k}_{U,V}$ *and* $\pi' := \pi^{k'}_{V',U'}$ *such that* $\pi.\mathsf{sid} = \pi'.\mathsf{sid} \neq \bot$*,* $\pi.\mathsf{role} = \mathsf{initiator}$*, and* $\pi'.\mathsf{role} = \mathsf{responder}$*, but* $U \neq U'$ *or* $V \neq V'$*.*

3. More than two sessions share the same session identifier:

   There exist at least three sessions π*,* π'*, and* π'' *such that* π*,* π'*,* π'' *are pairwise distinct, but* $\pi.\mathsf{sid} = \pi'.\mathsf{sid}' = \pi''.\mathsf{sid} \neq \bot$*.*

We say KE *is* $\mathsf{BR\text{-}Match}$ secure *if for all PPT adversaries* $\mathcal{A}$ *the advantage function defined as*

$$\mathsf{Adv}^{\mathsf{BR\text{-}Match}}_{\mathsf{KE},\mathcal{A}}(\lambda) := \Pr\left[\mathsf{G}^{\mathsf{BR\text{-}Match}}_{\mathsf{KE},\mathcal{A}}(\lambda) = 1\right]$$

is negligible in the security parameter λ*.*

Definition 4.2 (BR key secrecy)**.** *Let* KE *be a key exchange protocol with key distribution* $\mathcal{D}$*, and* $\mathcal{A}$ *be a PPT adversary interacting with* KE *via the queries defined in Section 4.2.1 in the following game* $\mathsf{G}^{\mathsf{BR},\mathcal{D}}_{\mathsf{KE},\mathcal{A}}(\lambda)$*:*

Setup. *The challenger generates long-term public/secret-key pairs for each participant* $U \in \mathcal{U}$*, chooses the test bit* $b_{\mathsf{test}} \xleftarrow{\$} \{0,1\}$ *at random, and sets* $\mathsf{lost} \leftarrow \mathsf{false}$*.*

Query. *The adversary* $\mathcal{A}$ *receives the generated public keys and has access to the oracle queries* $\mathsf{NewSession}$*,* Send*,* Reveal*,* $\mathsf{Corrupt}$*, and* Test*.*

Guess. *At some point,* $\mathcal{A}$ *stops and outputs a guess* b_{guess}*.*

Finalize. *The challenger sets* $\mathsf{lost} \leftarrow \mathsf{true}$ *if the following condition holds:*

Adversary has tested and revealed the key in a single session or in two partnered sessions:

There exist two (not necessarily distinct) sessions π*,* π' *such that* $\pi.\mathsf{sid} = \pi'.\mathsf{sid}$*,* $\pi.\mathsf{st}_{\mathsf{key}} = \mathsf{revealed}$*, and* $\pi'.\mathsf{tested} = \mathsf{true}$*.*

$\mathcal{A}$ wins the game $\mathsf{G}^{\mathsf{BR},\mathcal{D}}_{\mathsf{KE},\mathcal{A}}(\lambda)$ if $b_{\mathsf{guess}} = b_{\mathsf{test}}$ and $\mathsf{lost} = \mathsf{false}$.

We say that KE *provides* BR key secrecy with/without forward secrecy *if for all PPT adversaries $\mathcal{A}$ the advantage function defined as*

$$\mathsf{Adv}^{\mathsf{BR},\mathcal{D}}_{\mathsf{KE},\mathcal{A}}(\lambda) := \Pr\left[\mathsf{G}^{\mathsf{BR},\mathcal{D}}_{\mathsf{KE},\mathcal{A}}(\lambda) = 1\right] - \frac{1}{2}$$

is negligible in the security parameter λ.

Definition 4.3 (BR security). *We finally say a key exchange protocol* KE *is* BR-secure (with/without forward secrecy) *if* KE *provides* BR-Match *security and* BR *key secrecy (with/without forward secrecy), according to Definitions 4.1 and 4.2.*

Remark 4.4. Note that forward secrecy (if modeled) is incorporated into the Corrupt query and need thus not be stated in the *Finalize* step of Definition 4.2.

In the following, we come to the main contributions of the thesis and begin with the treatment of so-called hybrid key exchange protocols that promise a safe transition to a post-quantum world. Enjoy.

Chapter 5

Hybrid Key Exchange

By now, the fact that quantum computers will have serious implications on the security of the currently deployed cryptographic primitives is widely acknowledged and research into developing quantum-resistant solutions is well under way. However, history shows that the biggest hurdle is often not in the development of new, more secure algorithms, but in the widespread deployment of these algorithms. Backwards compatibility must be ensured while, at the same time, downgrade-attacks must be avoided at all costs. This makes the transition to new cryptographic primitives or protocol versions a highly non-trivial and daunting task, leading to slow adoption rates; even if some of the deployed building blocks are fundamentally broken. For post-quantum solutions yet another obstacle stands in the way of widespread immediate deployment. Due to their relative novelty, there exists a non-negligible chance that currently proposed schemes are, in fact, *not* quantum resistant and may not even withstand advanced classical cryptanalysis. Especially appropriate parameter selection for post-quantum schemes is not yet reliable: Albrecht et al. showed that for LWE- and NTRU-based schemes in the “Round 1” submissions to the NIST Post-Quantum Cryptography Standardization process [Nat15], the differences in bit hardness can measure up to several hundred bits [ACD+18] and Bernstein [Ber19] conducted a survey, comparing proofs and finding weaknesses in the lattice-based submissions to the NIST process.

Hence, the current situation presents as follows: On the one hand, with the expected long transition period, quantum-resistant schemes should be deployed as soon as possible to protect today’s communication from the potential threat posed by quantum computing; on the other hand we are not sufficiently confident in the concrete security of post-quantum schemes for immediate deployment. Both industry experts and academics explore ways how to best transition existing applications in light of this predicament. A promising approach to preserve today’s common security guarantees while still mitigating the risk of (future-)quantum attacks, is the use of so-called *hybrid* schemes. These schemes combine classically-secure and quantum-resistant cryptographic schemes in such a manner that the overall scheme remains secure as long as at least one of the two base schemes remains secure.

There have already been experimental deployments and multiple draft standards related to the Transport Layer Security and Internet Key Exchange protocol that propose some form of hybrid key exchange [Bra16, Lan18, CEL+16, CC19, SS17, WFZGM17, KK18, TTB+19]. However, despite the strong interest by industry in hybrid key exchange, there has been limited academic treatment of how to actually design and prove such schemes, especially with regards to quantum adversaries.

Our Contributions

Since early prototypes often become de facto standards, it is crucial to establish solid theoretical foundations for hybrid key exchange and KEMs at an early stage. Our work complements the existing work on classically-secure hybrid KEMs (a.k.a. KEM *combiners*[3]) to achieve provably-secure constructions in the post-quantum setting. Furthermore, it extends the foundations of authenticated key exchange security to allow treatment of hybrid constructions. Our contributions are three-fold:

- In Section 5.2 and Section 5.4, we introduce security models for KEMs and AKE protocols that account for adversaries with different levels of quantum capabilities. We start from the two-stage adversarial model of Bindel et al. [BHMS17] that was introduced in the context of hybrid signatures and transfer it to the relevant notions for hybrid key exchange.
- In Section 5.3, we present several KEM combiners and rigorously show their robustness with respect to the aforementioned security models, achieving the preservation of post-quantum security. These include a combiner, called XOR-then-MAC combiner, which is based on minimal cryptographic assumptions. We also discuss two combiners, the dual-PRF combiner and the nested dual-PRF combiner, that are closely related to the key schedule used in TLS 1.3 [Res18]. All proofs are in the standard model and do not rely on classical (or therefore quantum) random oracles.
- Lastly, in Section 5.5, we show how to generically build provably-secure hybrid authenticated key exchange from hybrid KEMs. Our construction relies on Krawczyk's SigMA-compiler [Kra03] using signatures and MACs to authenticate and lift the protocol to one that is secure in the strong adversarial model of authenticated key exchange.

Personal scientific contribution in this chapter. All the material from this section appeared in [BBF+19]. The full version of the paper can be found on ePrint [BBF+18]. While Nina, Douglas, and Brian mostly focused on the aspects of the paper involving quantum (two-stage) adversaries, Marc's and my focus lay on the transfer of security notions both for KEM combiners and AKE, as well as the proofs of these constructions. To reflect this, the following discussion of the work omits fully-quantum adversaries as they are outside of the scope of hybrid key exchange.

My main contribution lies in the development of the two-stage security notions for AKE and the proof of the generic hybrid AKE construction. Furthermore, I have considerably contributed to the construction and proofs of the dual-PRF and nested dual-PRF combiner in the standard model and have aided the transfer of security notions to the two-stage adversarial setting.

Note that, for completeness, this chapter will include the two-stage adversary framework and the proofs of the KEM combiners in the two-stage adversary model (cf. Section 5.3) which were also included in the Ph.D. thesis by Nina Bindel [Bin18]. The presentation of the proofs was at times slightly modified for clarity. For novelty, we will however re-prove the hybrid AKE construction with the dual-PRF combiner $\mathsf{dualPRF}$ in the alternative framework of strong breakdown resilience in Chapter 6.

5.1 Related Work

Before coming to the main contributions, we would like to pause and briefly review the current state of the art in related areas of research:

[3]We will use the terms "hybrid" and "combiner" interchangeably.

Robust combiners. In the existing literature, hybrid schemes have commonly been referred to as *(robust) combiners*. The study of such schemes in the symmetric setting dates back to the 80s to work by Asmuth and Blakely [AB81] and Even and Goldreich [EG85]. In the public key setting, Zhang et al. [ZHSI04], Herzberg [Her05], and Dodis and Katz [DK05] examined the security of combining multiple IND-CCA-secure public key encryption schemes. Harnik et al. [HKN+05] defined the term *robust combiner* and treated combiners for oblivious transfer, with a sketch of a combiner for key agreement. Combiners for other primitives have since followed, including Bindel et al. [BHMS17] on hybrid digital signatures. Most relevant to our setting of key exchange and KEMs is the recent work by Giacon, Heuer, and Poettering [GHP18] which considers various KEM combiners. Their combiner constructions show how hybrid KEMs can be built in the presence of solely classical adversaries.

Since the advent of quantum computing, and thus the introduction of more powerful adversaries, security analyses in the quantum setting are however not to be neglected. This is especially relevant for the constructions in [GHP18], as most of their proofs use idealized assumptions such as random oracles or ideal ciphers that are not guaranteed to transfer to the quantum setting (cf., e.g., [BDF+11]). Moreover, the (quantum) security of hybrid authenticated key exchange remains unresolved in [GHP18]. An alternative recent approach to model security of protocols in which a component fails is the breakdown resilience model of Brendel, Fischlin, and Günther [BFG17, BFG19] which in its stronger formulation can be used to analyze hybrid key exchanges. This model does not treat quantum adversaries explicitly but we demonstrate the applicability of this model to hybrid key exchanges in Section 6.4 by an exemplary proof of the dualPRF KEM combiner presented in this chapter as key agreement component in an authenticated key exchange protocol.

Real-world hybrid key exchange. The interest in hybrid key exchange has foremost been driven by industry players. Already in 2016, Google temporarily tested a hybrid key exchange cipher suite named CECPQ1 which combined elliptic curve Diffie–Hellman (ECDH) and the Ring-Learning-with-Errors-based key exchange scheme NewHope [ADPS16b] in the TLS stack on an experimental build of their Chrome browser [Bra16, Lan16]. Recently, Adam Langley announced the follow-up project CECPQ2 on his blog *Imperial Violet* [Lan18]. Experiments with this modification to TLS 1.3 have not only been run in Google's Chrome browser but also at Cloudflare [Kwi19, KSL+19]. Microsoft Research [CEL+16], Amazon [CC19], Mozilla [KK18], and Cloudflare [dV17] have gotten involved in developing further hybrid key exchange schemes, with a focus on supersingular isogeny-based schemes. Furthermore, along general outlines on how to transition to post-quantum secure cryptography (cf., e.g., [ETS15, Hof19]), Crockett, Paquin, and Stebila [CPS19] have put forward a survey on case studies for post-quantum and hybrid integration in TLS and SSH. Stebila, Fluhrer, and Gueron [SFG19] gave a detailed report on the expected challenges in incorporating hybrid modes in TLS 1.3, specifically. Several "Internet-Drafts" for concrete hybrid modes in TLS [SS17, WFZGM17, KK18] and IKE [TTB+19] have already been submitted to the Internet Engineering Task Force (IETF). Drucker and Gueron [DG19] have proposed a hybrid scheme for continuous key agreement that may be applicable to secure messaging applications.

5.2 Modeling Hybrid Key Encapsulation Mechanisms

In this section, we present the security model that we use in the rest of the chapter to analyze hybrid key encapsulation mechanisms. We first review the two-stage notion of (partially) quantum adversaries from Bindel et al. [BHMS17]. In a next step, we then apply this hierarchy of adversaries to the indistinguishability-based security notions of key encapsulation mechanisms.

5.2.1 Two-stage Adversaries for KEM Security

When analyzing the security of KEMs with respect to quantum adversaries we opt for the approach taken in [BHMS17] to analyze hybrid signature schemes. In contrast to previous works considering quantum adversaries, their notion allows to model adversaries whose quantum capabilities evolve over time. In particular, this fine-grained adversary model captures the difference between the following two settings:

Future-quantum adversaries. In this scenario, we assume that during the deployment of the scheme in question no sufficiently powerful quantum computers exist and it is expected to be decades before a full-fledged quantum computer is built. However, we want to protect today's communications against attacks in which the (currently classical) attacker records encrypted communications. Once a quantum computer becomes available, the adversary is then able to extract the session keys used for encryption from the corresponding collected key exchange transcripts.

Post-quantum adversaries. Alternatively, one might not feel comfortable excluding the possibility that powerful quantum computers exist already today or in the nearer future. In this case, stronger security guarantees are needed, as a locally quantum adversary may already interfere with the deployed protocols and may break employed primitives and hardness assumptions.

In the following, we adapt the two-stage notion from [BHMS17] to the indistinguishability-based security notions of key encapsulation mechanisms. The main difference when formally modeling the aforementioned scenarios is in *how* and *when* the oracles within the game description are accessed by the adversary. Thus, security notions that allow for no oracle access beyond the (quantum) random oracle, as, e.g., $\mathsf{IND\text{-}CPA}$ security, do not require the full formalism of two-stage adversaries $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$. They can be modeled by the usual (single stage) adversary notion, solely indicating whether the adversary is classical or quantum. Consequently, we will limit the following motivation of two-stage adversaries for KEMs to the active $\mathsf{IND\text{-}CCA}$ setting.

Analogously to the notion of unforgeability of signature schemes in [BHMS17], the first stage of the adversary may interact with the oracles in the game, while the second stage does not. This stipulation ties back to the distinction between future-quantum and post-quantum adversaries mentioned before. We identify three different aspects in the two-stage adversary's capabilities during the $\mathsf{IND\text{-}CCA}$ game with decapsulation oracle access:

- While the adversary has access to the decapsulation oracle, the adversary may locally have access to either classical or quantum computing power.
- The adversary's interaction with the decapsulation oracle in that phase can either be classical or quantum (i.e., queries may be in superposition).
- After access to the decapsulation oracle has been revoked, the adversary may again either have classical or quantum computing power available for local computations.

To model this, a two-stage adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ is introduced, where $\mathcal{A}_1$ has access to the decapsulation oracle. $\mathcal{A}_1$ then terminates and passes a state *st* to the second-stage adversary $\mathcal{A}_2$, which does no longer have access to the decapsulation oracle and may run local computations on *st* before terminating with its guess.

As in [BHMS17], we say that the adversary is *of type* $\mathsf{X^yZ}$ or an $\mathsf{X^yZ}$ *adversary*, where $\mathsf{X}, \mathsf{Z} \in \{\mathsf{C}, \mathsf{Q}\}$ and $\mathsf{y} \in \{\mathsf{c}, \mathsf{q}\}$. The X indicates whether the local computing power of the adversary in its first stage is classical ($\mathsf{X} = \mathsf{C}$) or quantum ($\mathsf{X} = \mathsf{Q}$). Analogously, Z indicates the available local computing power in the second stage. Lastly, y gives information about the way the first-stage adversary accesses the decapsulation oracle. Note that the random oracle is not influenced by the value of y. The adversary may query the random oracle in superposition (i.e., the *quantum* random oracle), whenever it is quantum, i.e., whenever $\mathsf{X} = \mathsf{Q}$ and/or $\mathsf{Z} = \mathsf{Q}$.

Not all combinations of classical and quantum adversaries in the two-stage setting are meaningful in a real-world context. We consider the following configurations of two-stage $\mathsf{X}^{\mathsf{y}}\mathsf{Z}$ adversaries to be relevant:

- $\mathsf{C}^{\mathsf{c}}\mathsf{C}$ security models a purely **classical** adversary with classical access to all oracles. This corresponds to the usual IND-CCA security notion for KEMs (cf. Def. 3.3).
- $\mathsf{C}^{\mathsf{c}}\mathsf{Q}$ security models a classical adversary at time of execution, that may however become quantum at a later point in time. In particular, that means that the adversary is classical as long as it has access to the decapsulation oracle. Eventually, the adversary gains local quantum computing power, but by this time, the ability to interact with the system and decapsulate are no longer available. We refer to these kind of adversaries as **future-quantum** adversaries.
- $\mathsf{Q}^{\mathsf{c}}\mathsf{Q}$ security models an adversary that has local quantum computing power at every stage, but interacts with the decapsulation oracle via classical queries. This is commonly referred to as the **post-quantum** setting and is subject of most works considering quantum adversaries in the context of key encapsulation mechanisms, e.g., [HHK17, SXY18, JZC+18, HKSU18, JZM19].
- $\mathsf{Q}^{\mathsf{q}}\mathsf{Q}$ security corresponds to **fully-quantum** adversaries. This adversary is quantum at every stage and queries the available decapsulation oracle in superposition.

It is notation-wise convenient to define an order for the notions, to reflect the relative strength of the adversary, where $\mathsf{C} < \mathsf{Q}$ and $\mathsf{c} < \mathsf{q}$. This implies a partial order $\mathsf{X}^{\mathsf{y}}\mathsf{Z} \leq \mathsf{U}^{\mathsf{v}}\mathsf{W}$ if $\mathsf{X} \leq \mathsf{U}$, $\mathsf{y} \leq \mathsf{v}$, and $\mathsf{Z} \leq \mathsf{W}$, i.e.,

$$\mathsf{C}^{\mathsf{c}}\mathsf{C} \leq \mathsf{C}^{\mathsf{c}}\mathsf{Q} \leq \mathsf{Q}^{\mathsf{c}}\mathsf{Q} \leq \mathsf{Q}^{\mathsf{q}}\mathsf{Q}.$$

Let $\max S$ (resp., $\min S$) denote the set of maximal (resp., minimal) elements of some set S according to this partial order. Since in our case $S \subseteq \{\mathsf{C}^{\mathsf{c}}\mathsf{C}, \mathsf{C}^{\mathsf{c}}\mathsf{Q}, \mathsf{Q}^{\mathsf{c}}\mathsf{Q}, \mathsf{Q}^{\mathsf{q}}\mathsf{Q}\}$ with the order on $\mathsf{X}^{\mathsf{y}}\mathsf{Z}$ notions as above, we often then speak of *the* maximal element; for example it holds that $\mathsf{Q}^{\mathsf{c}}\mathsf{Q} = \max\{\mathsf{C}^{\mathsf{c}}\mathsf{Q}, \mathsf{Q}^{\mathsf{c}}\mathsf{Q}\}$.

Remark 5.1. Note, that in the rest of this chapter, we concentrate on providing security against at most post-quantum, i.e., $\mathsf{Q}^{\mathsf{c}}\mathsf{Q}$ adversaries, omitting fully-quantum $\mathsf{Q}^{\mathsf{q}}\mathsf{Q}$ adversaries. This "limitation" is natural as hybrid solutions are solely intended to secure the transition to the post-quantum setting.

Finally, we are ready to define the notions for two-stage KEM security that we will be using throughout this chapter:

Definition 5.2 (Two-stage security of key encapsulation mechanisms)**.** *Analogously to the solely classical setting, the two-stage security of a key encapsulation mechanism $\mathcal{K} = (\mathsf{KGen}, \mathsf{Encaps}, \mathsf{Decaps})$ with key space $\mathscr{K}$ is defined in terms of* indistinguishability of encapsulated keys from random *with respect to either passive* chosen plaintext *adversaries* (IND-CPA *security*) *or active* chosen ciphertext *adversaries* (IND-CCA *security*)*. The respective security games are depicted in Figure 5.1.*

We say that $\mathcal{K}$ is $\mathsf{Z\text{-}IND\text{-}CPA}$ secure*, if for every quantum polynomial time*[4] *(QPT) adversary $\mathcal{A}$ of type Z the advantage function in winning the game $\mathsf{G}^{\mathsf{Z\text{-}ind\text{-}cpa}}_{\mathcal{K},\mathcal{A}}(\lambda)$ defined as*

$$\mathsf{Adv}^{\mathsf{Z\text{-}ind\text{-}cpa}}_{\mathcal{K},\mathcal{A}}(\lambda) := \left|\Pr\left[\mathsf{G}^{\mathsf{Z\text{-}ind\text{-}cpa}}_{\mathcal{K},\mathcal{A}}(\lambda) = 1\right] - \frac{1}{2}\right|$$

is negligible in the security parameter λ. We further say that $\mathcal{K}$ is $\mathsf{X^cZ\text{-}IND\text{-}CCA}$ secure*, if for every QPT $\mathsf{X^cZ}$ adversary $\mathcal{A}$ the advantage function in winning the game $\mathsf{G}^{\mathsf{X^cZ\text{-}ind\text{-}cca}}_{\mathcal{K},\mathcal{A}}(\lambda)$ defined as*

$$\mathsf{Adv}^{\mathsf{X^cZ\text{-}ind\text{-}cca}}_{\mathcal{K},\mathcal{A}}(\lambda) := \left|\Pr\left[\mathsf{G}^{\mathsf{X^cZ\text{-}ind\text{-}cca}}_{\mathcal{K},\mathcal{A}}(\lambda) = 1\right] - \frac{1}{2}\right|$$

is negligible in the security parameter λ.

```
$\mathsf{G}^{\mathsf{Z\text{-}ind\text{-}cpa}}_{\mathcal{K},\mathcal{A}}(\lambda)$:
1 $(pk, sk) \xleftarrow{\$} \mathsf{KGen}(1^\lambda)$
2 $(c^\star, K_0^\star) \xleftarrow{\$} \mathsf{Encaps}(pk)$
3 $K_1^\star \xleftarrow{\$} \mathscr{K}$
4 $b \xleftarrow{\$} \{0,1\}$
5 $b' \xleftarrow{\$} \mathcal{A}(pk, c^\star, K_b^\star)$
6 return $[\![b' = b]\!]$
```

```
$\mathsf{G}^{\mathsf{X^cZ\text{-}ind\text{-}cca}}_{\mathcal{K},\mathcal{A}}(\lambda)$:
1 $(pk, sk) \xleftarrow{\$} \mathsf{KGen}(1^\lambda)$
2 $(c^\star, K_0^\star) \xleftarrow{\$} \mathsf{Encaps}(pk)$
3 $K_1^\star \xleftarrow{\$} \mathscr{K}$
4 $b \xleftarrow{\$} \{0,1\}$
5 $st \xleftarrow{\$} \mathcal{A}_1^{\mathcal{O}_{\mathsf{Decaps}}}(pk, c^\star, K_b^\star)$
6 $b' \xleftarrow{\$} \mathcal{A}_2(st)$
7 return $[\![b' = b]\!]$
```

```
$\mathcal{O}_{\mathsf{Decaps}}(c)$:
8  if $c = c^\star$
9      return $\bot$
10 else
11     return $\mathsf{Decaps}(sk, c)$
```

Figure 5.1: $\mathsf{IND\text{-}CPA}$ and $\mathsf{IND\text{-}CCA}$ security of $\mathcal{K} = (\mathsf{KGen}, \mathsf{Encaps}, \mathsf{Decaps})$ with key space $\mathscr{K}$ with respect to two-stage adversaries $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$.

Remark 5.3. For consistency with the $\mathsf{IND\text{-}CCA}$ case, where we need to distinguish in which stage the adversary has quantum power, we occasionally also use the notation $\mathsf{X^cZ\text{-}IND\text{-}CPA}$ instead of $\mathsf{Z\text{-}IND\text{-}CPA}$ in the $\mathsf{IND\text{-}CPA}$ case. In such cases we sometimes refer to both as $\mathsf{X^cZ\text{-}IND\text{-}ATK}$ security with $\mathsf{ATK} \in \{\mathsf{CPA}, \mathsf{CCA}\}$. We stress, however, that X and c in the CPA case are superfluous, and are stated solely for notational uniformity.

5.2.2 Relations Between Indistinguishability Security Notions

The various indistinguishability notions for KEMs are related to each other through a series of implications and separations which are depicted in Figure 5.2 and proven in the following section.

Proposition 5.4 (Implications)**.** *Let $\mathcal{K}$ be a key encapsulation mechanism. Then the following implications hold:*

If $\mathcal{K}$ is $\mathsf{Q^cQ\text{-}IND\text{-}CCA}$ *secure, then $\mathcal{K}$ is also* $\mathsf{C^cQ\text{-}IND\text{-}CCA}$ *secure.*

If $\mathcal{K}$ is $\mathsf{C^cQ\text{-}IND\text{-}CCA}$ *secure, then $\mathcal{K}$ is also* $\mathsf{C^cC\text{-}IND\text{-}CCA}$ *secure and* $\mathsf{Q\text{-}IND\text{-}CPA}$ *secure.*

If $\mathcal{K}$ is $\mathsf{Q\text{-}IND\text{-}CPA}$ *secure or* $\mathsf{C^cC\text{-}IND\text{-}CCA}$ *secure, then $\mathcal{K}$ is also* $\mathsf{C\text{-}IND\text{-}CPA}$ *secure.*

Proof. Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be a two-stage adversary. The proof is straightforward since every classical adversary can be seen as a quantum adversary that forgoes its additional quantum power. It thus holds that

$$\mathsf{Adv}^{\mathsf{Q^cQ\text{-}ind\text{-}cca}}_{\mathcal{K},\mathcal{A}}(\lambda) \geq \mathsf{Adv}^{\mathsf{C^cQ\text{-}ind\text{-}cca}}_{\mathcal{K},\mathcal{A}}(\lambda) \geq \mathsf{Adv}^{\mathsf{C^cC\text{-}ind\text{-}cca}}_{\mathcal{K},\mathcal{A}}(\lambda)$$

[4]We say a quantum algorithm runs in *quantum polynomial time*, if it is a uniform family of quantum circuits of size polynomial in the security parameter.

$$\begin{tikzcd}
\text{C-IND-CPA} & \text{Q-IND-CPA} \arrow[l, "(5.4)"'] \arrow[dl, bend left, "/" marking, "(5.8)"] & \\
\text{C}^{\text{c}}\text{C-IND-CCA} \arrow[u, "(5.4)"] \arrow[ur, bend left, "/" marking, "(5.5)"] \arrow[r, bend left, "/" marking, "(5.5)"] & \text{C}^{\text{c}}\text{Q-IND-CCA} \arrow[u, "(5.4)"'] \arrow[l, bend left, "(5.4)"] \arrow[r, bend left, "/" marking, "(5.7)"] & \text{Q}^{\text{c}}\text{Q-IND-CCA} \arrow[l, bend left, "(5.4)"]
\end{tikzcd}$$

Figure 5.2: Implications ($\rightarrow$) and separations ($\not\rightarrow$) between indistinguishability-based security notions for KEMs wrt. two-stage adversaries.

and

$$\mathsf{Adv}^{\text{Q-ind-cpa}}_{\mathcal{K},\mathcal{A}}(\lambda) \geq \mathsf{Adv}^{\text{C-ind-cpa}}_{\mathcal{K},\mathcal{A}}(\lambda).$$

Similarly, active security implies passive security, i.e., $\mathsf{Adv}^{\text{C}^{\text{c}}\text{C-ind-cca}}_{\mathcal{K},\mathcal{A}}(\lambda) \geq \mathsf{Adv}^{\text{C-ind-cpa}}_{\mathcal{K},\mathcal{A}}(\lambda)$. □

In fact, these implications are strict. In the following, we show separations between the different notions. We start with Proposition 5.5 which essentially states that there exist KEMs that are secure against classical C$^{\text{c}}$C adversaries, but that become insecure once adversaries gain quantum power, i.e., are of type C$^{\text{c}}$Q or Q:

Proposition 5.5 (C$^{\text{c}}$C-IND-CCA $\not\Rightarrow$ Q-IND-CPA, C$^{\text{c}}$Q-IND-CCA)**.** *Let $\mathcal{E} = (\mathcal{E}.\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ be a public key encryption scheme that is* C-IND-CPA *secure. Then there exists a* C$^{\text{c}}$C-IND-CCA *secure KEM $\mathcal{K}$ in the random oracle model that is neither* Q-IND-CPA *secure nor* C$^{\text{c}}$Q-IND-CCA *secure.*

Proof. We construct $\mathcal{K} = (\mathsf{KGen}, \mathsf{Encaps}, \mathsf{Decaps})$ from $\mathcal{E}$ via the Fujisaki-Okamoto transform [FO99, FO13, HHK17] which turns a weakly-secure public key encryption scheme into an IND-CCA-secure key encapsulation mechanism when modeling the hash functions G and H as random oracles. The resulting KEM is depicted in Figure 5.3.

```
KGen(1^λ):
  $(pk, sk) \xleftarrow{\$} \mathcal{E}.\mathsf{KGen}(1^\lambda)$
  return $(pk, sk)$

Encaps(pk):
  $m \xleftarrow{\$} \mathcal{M}$
  $c \xleftarrow{\$} \mathsf{Enc}(pk, m; \mathsf{G}(m))$
  $K \leftarrow \mathsf{H}(c, m)$
  return $c, K$

Decaps(sk, c):
  $m' \leftarrow \mathsf{Dec}(sk, c)$
  if $c \neq \mathsf{Enc}(pk, m'; \mathsf{G}(m'))$ or if $m' = \bot$
      return $\bot$
  else
      return $\mathsf{H}(c, m')$
```

Figure 5.3: Description of KEM $\mathcal{K}$ that separates C$^{\text{c}}$C-IND-CCA from Q-IND-CPA and C$^{\text{c}}$Q-IND-CCA. (Prop. 5.5).

While $\mathcal{K}$ is secure against classical C$^{\text{c}}$C-IND-CCA adversaries by construction, it is not secure against an adversary that has local quantum capabilities, i.e., Q-IND-CPA or C$^{\text{c}}$Q-IND-CCA adversaries. The adversary then simply breaks the encryption scheme $\mathcal{E}$ (which is not resistant to quantum algorithms) to retrieve m from the ciphertext c. With this, the key can be re-computed as $\mathsf{H}(c, m)$ and indistinguishability can no longer be guaranteed.

□

Next, we show that there exist KEMs that are secure as long as only classical adversaries interact with the decapsulation oracle ($\mathsf{C^cQ}$ adversaries), but that become insecure in the post-quantum setting ($\mathsf{Q^cQ}$ adversaries). For this, we need the two-stage notion of $\mathsf{OW\text{-}CPA}$ security:

Definition 5.6 (Two-stage one-way security of key encapsulation mechanisms)**.** *The two-stage security of a key encapsulation mechanism $\mathcal{K} = (\mathsf{KGen}, \mathsf{Encaps}, \mathsf{Decaps})$ with key space $\mathscr{K}$ can also be defined in terms of* one-wayness *with respect to either passive* chosen plaintext *adversaries* ($\mathsf{OW\text{-}CPA}$ *security*) *or active* chosen ciphertext *adversaries* ($\mathsf{OW\text{-}CCA}$ *security*)*. The respective security games are depicted in Figure 5.1.*

We say that $\mathcal{K}$ is $\mathsf{Z\text{-}OW\text{-}CPA}$ secure*, if for every quantum polynomial time (QPT) adversary $\mathcal{A}$ of type Z the advantage function in winning the game $\mathsf{G}^{\mathsf{Z\text{-}ow\text{-}cpa}}_{\mathcal{K},\mathcal{A}}(\lambda)$ defined as*

$$\mathsf{Adv}^{\mathsf{Z\text{-}ow\text{-}cpa}}_{\mathcal{K},\mathcal{A}}(\lambda) := \Pr\left[\mathsf{G}^{\mathsf{Z\text{-}ow\text{-}cpa}}_{\mathcal{K},\mathcal{A}}(\lambda) = 1\right]$$

is negligible in the security parameter λ. We further say that $\mathcal{K}$ is $\mathsf{X^cZ\text{-}OW\text{-}CCA}$ secure*, if for every QPT $\mathsf{X^cZ}$ adversary $\mathcal{A}$ the advantage function in winning the game $\mathsf{G}^{\mathsf{X^cZ\text{-}ow\text{-}cca}}_{\mathcal{K},\mathcal{A}}(\lambda)$ defined as*

$$\mathsf{Adv}^{\mathsf{X^cZ\text{-}ow\text{-}cca}}_{\mathcal{K},\mathcal{A}}(\lambda) := \Pr\left[\mathsf{G}^{\mathsf{X^cZ\text{-}ow\text{-}cca}}_{\mathcal{K},\mathcal{A}}(\lambda) = 1\right]$$

is negligible in the security parameter λ.

```
$\mathsf{G}^{\mathsf{Z\text{-}ow\text{-}cpa}}_{\mathcal{K},\mathcal{A}}(\lambda)$:
1 $(pk, sk) \xleftarrow{\$} \mathsf{KGen}(1^\lambda)$
2 $(c^\star, K^\star) \xleftarrow{\$} \mathsf{Encaps}(pk)$
3 $K' \xleftarrow{\$} \mathcal{A}(pk, c^\star)$
4 return $[\![K' = K^\star]\!]$
```

```
$\mathsf{G}^{\mathsf{X^cZ\text{-}ow\text{-}cca}}_{\mathcal{K},\mathcal{A}}(\lambda)$:
1 $(pk, sk) \xleftarrow{\$} \mathsf{KGen}(1^\lambda)$
2 $(c^\star, K^\star) \xleftarrow{\$} \mathsf{Encaps}(pk)$
3 $st \xleftarrow{\$} \mathcal{A}_1^{\mathcal{O}_{\mathsf{Decaps}}}(pk, c^\star)$
4 $K' \xleftarrow{\$} \mathcal{A}_2(st)$
5 return $[\![K' = K^\star]\!]$
```

```
$\mathcal{O}_{\mathsf{Decaps}}(c)$:
6 if $c = c^\star$
7     return $\perp$
8 else
9     return $\mathsf{Decaps}(sk, c)$
```

Figure 5.4: $\mathsf{OW\text{-}CPA}$ and $\mathsf{OW\text{-}CCA}$ security of $\mathcal{K} = (\mathsf{KGen}, \mathsf{Encaps}, \mathsf{Decaps})$ with key space $\mathscr{K}$ wrt. two-stage adversaries $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$.

Proposition 5.7 ($\mathsf{C^cQ\text{-}IND\text{-}CCA} \not\Longrightarrow \mathsf{Q^cQ\text{-}IND\text{-}CCA}$)**.** *Let $\mathcal{K} = (\mathsf{KGen}, \mathsf{Encaps}, \mathsf{Decaps})$ be a* $\mathsf{C^cQ\text{-}IND\text{-}CCA}$*-secure KEM and $\mathcal{K}' = (\mathsf{KGen}', \mathsf{Encaps}', \mathsf{Decaps}')$ be a* $\mathsf{C\text{-}OW\text{-}CPA}$*-secure KEM for which there is an efficient quantum algorithm that recovers the session key from the ciphertexts, i.e., it is not* $\mathsf{Q\text{-}OW\text{-}CPA}$*. Then there exists a key encapsulation mechanism $\overline{\mathcal{K}}$ that is* $\mathsf{C^cQ\text{-}IND\text{-}CCA}$ *secure, but not* $\mathsf{Q^cQ\text{-}IND\text{-}CCA}$ *secure.*

Proof. The construction of the separating KEM $\overline{\mathcal{K}} = (\overline{\mathsf{KGen}}, \overline{\mathsf{Encaps}}, \overline{\mathsf{Decaps}})$, which is secure against future-quantum adversaries but not against post-quantum adversaries, can be found in Figure 5.5. The idea is to insert a backdoor in the decapsulation oracle that requires local quantum computing power to be exploited.

<u>$\overline{\mathcal{K}}$ is $\mathsf{C^cQ\text{-}IND\text{-}CCA}$ secure.</u> Assume it is not, i.e., there exists an efficient future-quantum adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ that can break the $\mathsf{C^cQ\text{-}IND\text{-}CCA}$ security of $\overline{\mathcal{K}}$. However, this immediately yields an efficient adversary $\mathcal{B}$ that can break the $\mathsf{C^cQ\text{-}IND\text{-}CCA}$ security of $\mathcal{K}$. This adversary is defined as follows:

The reduction $\mathcal{B}$ receives its challenge $(pk, c^\star, K_b^\star)$ for KEM $\mathcal{K}$. It then runs lines 2 and 3 of $\overline{\mathsf{KGen}}$ by itself to create the backdoor (c', K'). It sends $(\overline{pk} \leftarrow (pk, c'), c^\star, K_b^\star)$ as

```
$\overline{\mathsf{KGen}}(1^\lambda)$:
1  $(pk, sk) \xleftarrow{\$} \mathsf{KGen}(1^\lambda)$
2  $(pk', sk') \xleftarrow{\$} \mathsf{KGen}'(1^\lambda)$
3  $(c', K') \xleftarrow{\$} \mathsf{Encaps}'(pk')$
4  $\overline{pk} \leftarrow (pk, c')$
5  $\overline{sk} \leftarrow sk$
6  return $(\overline{pk}, \overline{sk})$

$\overline{\mathsf{Encaps}}(\overline{pk})$:
7  $(c, K) \leftarrow \mathsf{Encaps}(pk)$
8  return $(c, K)$

$\overline{\mathsf{Decaps}}(\overline{sk}, c)$:
9  if $c = K'$
10     return $sk$
11 else
12     return $\mathsf{Decaps}(sk, c)$
```

Figure 5.5: Description of KEM $\overline{\mathcal{K}}$ that separates $\mathsf{C^cQ\text{-}IND\text{-}CCA}$ from $\mathsf{Q^cQ\text{-}IND\text{-}CCA}$ (Prop. 5.7).

input to $\mathcal{A}$. Whenever $\mathcal{A}_1$ queries the decapsulation oracle $\overline{\mathsf{Decaps}}$ on some ciphertext $c \neq K'$, the reduction $\mathcal{B}$ forwards the query to its own decapsulation oracle Decaps. If the adversary queries the oracle on K', then $\mathcal{B}$ returns $\perp$. Note that this only occurs with negligible probability, as this would immediately contradict the one-wayness of $\mathcal{K}'$ against classical adversaries.

Once $\mathcal{A}_2$ outputs its guess b', $\mathcal{B}$ outputs the same guess. It is easy to see, that if $\mathcal{A}$ wins $\mathsf{G}^{\mathsf{C^cQ\text{-}ind\text{-}cca}}_{\overline{\mathcal{K}},\mathcal{A}}(\lambda)$ with non-negligible advantage, then $\mathcal{B}$ also wins its game $\mathsf{G}^{\mathsf{C^cQ\text{-}ind\text{-}cca}}_{\mathcal{K},\mathcal{B}}(\lambda)$ with the same advantage.

$\overline{\mathcal{K}}$ is not $\mathsf{Q^cQ\text{-}IND\text{-}CCA}$ secure. Contrary to before, the first stage adversary $\mathcal{A}_1$ now has access to local quantum computing power. This means in particular, that it can obtain K' from c', which is part of its public key $\overline{pk}$. By construction of $\overline{\mathcal{K}}$, the decapsulation oracle queried on $c = K'$ returns the secret key sk of $\mathcal{K}$. This can then be used to recover the key $K^\star$ encapsulated in the challenge $c^\star$ and therefore distinguish the corresponding challenge key $K_b^\star$ from random.

□

Lastly, we observe that passive security in the quantum setting ($\mathsf{Q\text{-}IND\text{-}CPA}$) is not necessarily enough to show classical security against active adversaries ($\mathsf{C^cC\text{-}IND\text{-}CCA}$).

Proposition 5.8 ($\mathsf{Q\text{-}IND\text{-}CPA} \not\Longrightarrow \mathsf{C^cC\text{-}IND\text{-}CCA}$)**.** *Assume there exists a* $\mathsf{Q\text{-}IND\text{-}CPA}$*-secure KEM* $\mathcal{K}$*. Then there exists a KEM* $\mathcal{K}'$ *that is* $\mathsf{Q\text{-}IND\text{-}CPA}$ *secure but not* $\mathsf{C^cC\text{-}IND\text{-}CCA}$ *secure.*

Proof. Let $\mathcal{K} = (\mathsf{KGen}, \mathsf{Encaps}, \mathsf{Decaps})$ be a $\mathsf{Q\text{-}IND\text{-}CPA}$-secure KEM. Then $\mathcal{K}' = (\mathsf{KGen}', \mathsf{Encaps}', \mathsf{Decaps}')$ as constructed in Figure 5.6 is passively secure against quantum adversaries but not actively secure against classical adversaries.

```
$\mathsf{KGen}'(1^\lambda)$:
1  $(pk, sk) \leftarrow \mathsf{KGen}(1^\lambda)$
2  return $(pk, sk)$

$\mathsf{Encaps}'(pk)$:
3  $(c, K) \xleftarrow{\$} \mathsf{Encaps}(pk)$
4  return $(c, K)$

$\mathsf{Decaps}'(sk, c)$:
5  if $c = pk$
6    return $sk$
7  else
8    return $\mathsf{Decaps}(sk, c)$
```

Figure 5.6: Description of KEM $\mathcal{K}'$ that separates $\mathsf{Q\text{-}IND\text{-}CPA}$ from $\mathsf{C^cC\text{-}IND\text{-}CCA}$ (Prop. 5.8).

Clearly $\mathcal{K}'$ is not $\mathsf{C^cC\text{-}IND\text{-}CCA}$ secure: when the public key is asked to the decapsulation oracle, the secret key for decapsulation is returned. Recall that for honest executions in secure encryption schemes the probability that a ciphertext c equals the public key pk "accidentally" is negligible. However, as long as no queries to the decapsulation oracle are allowed, $\mathcal{K}'$ is as secure as $\mathcal{K}$.

□

5.3 Practical Hybrid Key Encapsulation Mechanisms

We now turn towards the question of how to construct robust combiners for key encapsulation mechanisms with respect to (partially) quantum adversaries. The three combiners proposed in this section aim at applicability, either through their optimality with regards to employed cryptographic hardness assumptions and primitives (XOR-then-MAC combiner XtM), or by their applicability to drafts of real-world hybrid protocols (dual-PRF combiner $\mathsf{dualPRF}$ and nested dual-PRF combiner N).

In the following constructions, let $\mathcal{K}_1 = (\mathsf{KGen}_1, \mathsf{Encaps}_1, \mathsf{Decaps}_1)$ and $\mathcal{K}_2 = (\mathsf{KGen}_2, \mathsf{Encaps}_2, \mathsf{Decaps}_2)$ be two KEMs with key space $\mathscr{K}_1$ and $\mathscr{K}_2$, respectively. We write $\mathcal{C}[\mathcal{K}_1, \mathcal{K}_2] = (\mathsf{KGen}_\mathcal{C}, \mathsf{Encaps}_\mathcal{C}, \mathsf{Decaps}_\mathcal{C})$ for the hybrid KEM constructed by one of the three proposals $\mathcal{C} \in \{\mathsf{XtM}, \mathsf{dualPRF}, \mathsf{N}\}$. Its key space will be denoted by $\mathscr{K}$.

We start with the XOR-then-MAC combiner. After a brief introduction to hybrid modes in TLS, we present the remaining two combiners, the dual-PRF and the nested dual-PRF combiners

5.3.1 XtM: XOR-then-MAC Combiner

Ideally, we would like to build a KEM combiner which can do without further cryptographic operations than those already employed in the two underlying schemes. One of the most natural combiners that is conceivable is a plain XOR-combiner, where the ciphertext consists of the concatenation of the two ciphertexts and the encapsulated key is simply the XOR of the individual keys.

Unfortunately, it is well-known that such a combiner does not preserve $\mathsf{IND\text{-}CCA}$ security for encryption. Giacon et al. [GHP18] reiterated this result in the context of KEM combiners. We additionally note that this even holds, if *both* KEMs are $\mathsf{IND\text{-}CCA}$ secure: Given a challenge ciphertext $c^\star = (c_1^\star, c_2^\star)$ the adversary can make two decapsulation requests for $c' = (c_1^\star, c_2)$ and $c'' = (c_1, c_2^\star)$ with fresh ciphertexts $c_1 \neq c_1^\star$, $c_2 \neq c_2^\star$, for which it knows the encapsulated keys. This allows the adversary to easily recover the challenge key from the responses of the oracle.

Combiner description. To prevent the adversary from these "mix-and-match attacks", we augment the combined ciphertext by adding a MAC tag over the individual ciphertexts c_1, c_2. The combined KEM key as well as the key for the MAC scheme are derived as the XOR of the two encapsulated keys K_1 and K_2. Note that the underlying KEMs must thus potentially stretch their key output pseudorandomly to allow for two keys of sufficient length to be derived. For the ease of the discussion of the XOR-then-MAC combiner, let $\mathscr{K}_1$ denotes the key space of $\mathcal{K}_1$ and let it be such that $\mathscr{K}_1 = \{0,1\}^{2 \cdot l(\lambda)}$ where the $l(\lambda)$ most significant bits constitute the actual KEM key K_1 and the other $l(\lambda)$ bits the MAC key k_1; analogously for $\mathscr{K}_2$ of $\mathcal{K}_2$. The resulting hybrid KEM with key space $\mathscr{K} = \{0,1\}^{l(\lambda)}$ is depicted in Figure 5.7.

Security of the MAC scheme. The MAC scheme must be robust such that it provides unforgeability, even if one of the keys is chosen adversarially. I.e., the adversary $\mathcal{A}$ tries to win for a key $k = (k_1, k_2)$, where either k_1 or k_2 is chosen by itself. We allow $\mathcal{A}$ to specify one of the two keys for computing the challenge, as well as for each verification query and in the forgery attempt.

It suffices to use one-time unforgeable MACs with multiple verification queries, where the adversary can initially choose a message, receives the MAC tag, and can then make multiple verification attempts for other messages. We require strong unforgeability, meaning the adversary wins if it is able to output a valid message-tag pair, which may even be for the same initial message.

```
$\underline{\mathsf{KGen}_{\mathsf{XtM}}(1^\lambda):}$
 1 $(pk_1, sk_1) \xleftarrow{\$} \mathsf{KGen}_(1^\lambda)$
 2 $(pk_2, sk_2) \xleftarrow{\$} \mathsf{KGen}_2(1^\lambda)$
 3 $pk \leftarrow (pk_1, pk_2)$
 4 $sk \leftarrow (sk_1, sk_2)$
 5 **return** $(pk, sk)$

$\underline{\mathsf{Encaps}_{\mathsf{XtM}}(pk):}$
 6 $(c_1, K_1\|k_1) \xleftarrow{\$} \mathsf{Encaps}_1(pk_1)$
 7 $(c_2, K_2\|k_2) \xleftarrow{\$} \mathsf{Encaps}_2(pk_2)$
 8 $k \leftarrow (k_1, k_2)$
 9 $c' \leftarrow (c_1, c_2)$
10 $\tau \leftarrow \mathsf{MAC}(k, c')$
11 $c \leftarrow (c', \tau)$
12 $K \leftarrow K_1 \oplus K_2$
13 **return** $(c, K)$

$\underline{\mathsf{Decaps}_{\mathsf{XtM}}(sk, c):}$
14 $K_1'\|k_1' \leftarrow \mathsf{Decaps}_1(sk_1, c_1)$,
15 $K_2'\|k_2' \leftarrow \mathsf{Decaps}_2(sk_2, c_2)$
16 $k' \leftarrow (k_1', k_2')$
17 **if** $\mathsf{Vfy}(k', \tau, (c_1, c_2)) = 0$
18    **return** $\bot$
19 **else**
20    **return** $K_1' \oplus K_2'$
```

Figure 5.7: Hybrid KEM constructed by the XOR-then-MAC combiner $\mathsf{XtM}[\mathcal{K}_1, \mathcal{K}_2, \mathcal{M}]$.

The security game $\mathsf{G}^{\mathsf{X^cZ\text{-}ot\text{-}s\text{-}euf}}_{\mathcal{M},\mathcal{A}}(\lambda)$ for two-stage one-time strong existential unforgeability of such MACs with two keys is given in Figure 5.8.

Definition 5.9 (Two-stage one-time strong existential unforgeability)**.** *Let* $\mathcal{M} = (\mathsf{KGen}, \mathsf{MAC}, \mathsf{Vfy})$ *be a message authentication scheme, where the key generation algorithm* KGen *outputs a key* k *that is comprised of two keys* k_1, k_2*.*

We say that $\mathcal{M}$ *is* two-stage one-time strong existential unforgeable *or* $\mathsf{X^cZ\text{-}OT\text{-}s\text{-}EUF}$-secure *if for every QPT adversary* $\mathcal{A}$ *of type* $\mathsf{X^cZ}$ *the advantage function in winning the game* $\mathsf{G}^{\mathsf{X^cZ\text{-}ot\text{-}s\text{-}euf}}_{\mathcal{M},\mathcal{A}}(\lambda)$ *defined as*

$$\mathsf{Adv}^{\mathsf{X^cZ\text{-}ot\text{-}s\text{-}euf}}_{\mathcal{M},\mathcal{A}}(\lambda) := \Pr\Big[\mathsf{G}^{\mathsf{X^cZ\text{-}ot\text{-}s\text{-}euf}}_{\mathcal{M},\mathcal{A}}(\lambda) = 1\Big]$$

is negligible in the security parameter λ*.*

```
$\underline{\mathsf{G}^{\mathsf{X^cZ\text{-}ot\text{-}s\text{-}euf}}_{\mathcal{M},\mathcal{A}}(\lambda):}$
 1 $(k_1, k_2) \xleftarrow{\$} \mathsf{KGen}(1^\lambda)$
 2 $(b, k, m^\star, st) \xleftarrow{\$} \mathcal{A}_1$
 3 **if** $b = 1$
 4    $k^\star \leftarrow (k, k_2)$
 5 **else**
 6    $k^\star \leftarrow (k_1, k)$
 7 $\tau^\star \leftarrow \mathsf{MAC}(k^\star, m^\star)$
 8 $st \xleftarrow{\$} \mathcal{A}_1^{\mathcal{O}_{\mathsf{Vfy}}}(\tau^\star)$
 9 $(k', \tau', m') \xleftarrow{\$} \mathcal{A}_2(st)$
10 **if** $b = 1$
11    $k'' \leftarrow (k', k_2)$
12 **else**
13    $k'' \leftarrow (k_1, k')$
14 **if** $(\mathsf{Vfy}(k'', \tau', m') = 1) \wedge ((m', \tau') \neq (m^*, \tau^*))$
15    **return** 1
16 **else**
17    **return** 0

$\underline{\mathcal{O}_{\mathsf{Vfy}}(k, \tau, m):}$
18 **if** $b = 1$
19    $\tilde{k} \leftarrow (k, k_2)$
20 **else**
21    $\tilde{k} \leftarrow (k_1, k)$
22 **return** $\mathsf{Vfy}(\tilde{k}, \tau, m)$
```

Figure 5.8: Security game for one-time strong existential unforgeability with multiple verifications of a two-key MAC $\mathcal{M} = (\mathsf{KGen}, \mathsf{MAC}, \mathsf{Vfy})$ with respect to two-stage adversaries $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$.

The simplest ways to build secure MAC combiners of this type is to concatenate two MACs, where each one is computed under one of the keys. For specific constructions various improvements may apply. For instance, for deterministic MACs, one may aggregate the two MACs via XOR [KL08] to reduce the communication overhead. Post-quantum secure MACs that satisfy the $\mathsf{Q^cQ\text{-}OT\text{-}s\text{-}EUF}$ notion can be constructed from universal hash functions based on the Carter-Wegman paradigm [WC81]. This construction does not rely on cryptographic assumptions and thus makes the XOR-then-MAC combiner optimal with respect to needed cryptographic assumptions.

Security of the XOR-then-MAC combiner. We can now show that the XOR-then-MAC combiner constitutes a hybrid KEM: the combined KEM $\mathcal{K} = \mathsf{XtM}[\mathcal{K}_1, \mathcal{K}_2, \mathcal{M}]$ preserves the security offered by the strongest of the input KEMs $\mathcal{K}_1, \mathcal{K}_2$, if $\mathcal{M}$ offers the same level of security. In fact, the security of the MAC scheme is only required in case of $\mathsf{IND\text{-}CCA}$ attacks when the adversary has access to the decapsulation oracle, yielding an even better bound for the $\mathsf{IND\text{-}CPA}$ case.

Theorem 5.10 (XOR-then-MAC is robust). *Let $\mathcal{K}_1$ be an $\mathsf{X^cZ\text{-}IND\text{-}ATK}$-secure KEM or $\mathcal{K}_2$ be a $\mathsf{U^cW\text{-}IND\text{-}ATK}$-secure KEM. Furthermore, let $\mathcal{M}$ be an $\mathsf{R^cT\text{-}OT\text{-}s\text{-}EUF}$-secure MAC scheme, where $\mathsf{R^cT} = \max\{\mathsf{X^cZ}, \mathsf{U^cW}\}$.*

Then $\mathsf{XtM}[\mathcal{K}_1, \mathcal{K}_2, \mathcal{M}]$ as defined in Figure 5.7 is also $\mathsf{R^cT\text{-}IND\text{-}ATK}$ secure. More precisely, for any efficient QPT adversary $\mathcal{A}$ of type $\mathsf{R^cT}$ against the combined KEM $\mathcal{K} := \mathsf{XtM}[\mathcal{K}_1, \mathcal{K}_2, \mathcal{M}]$, there exist efficient adversaries $\mathcal{B}_1$, $\mathcal{B}_2$, and $\mathcal{B}_3$ such that

$$\mathsf{Adv}^{\mathsf{R^cT\text{-}ind\text{-}atk}}_{\mathcal{K},\mathcal{A}}(\lambda) \quad \leq \quad 2 \cdot \min\left\{\mathsf{Adv}^{\mathsf{R^cT\text{-}ind\text{-}atk}}_{\mathcal{K}_2,\mathcal{B}_1}(\lambda), \mathsf{Adv}^{\mathsf{R^cT\text{-}ind\text{-}atk}}_{\mathcal{K}_1,\mathcal{B}_2}(\lambda)\right\} + \ \mathsf{Adv}^{\mathsf{R^cT\text{-}ot\text{-}s\text{-}euf}}_{\mathcal{M},\mathcal{B}_3}(\lambda).$$

Proof. We consider the scenario where $\mathcal{K}_1$ becomes insecure; the case for $\mathcal{K}_2$ breaking proceeds analogously. Furthermore, we focus on the stronger $\mathsf{IND\text{-}CCA}$ case here as this immediately implies $\mathsf{IND\text{-}CPA}$ security.

In the proof, we show that the existence of an efficient adversary $\mathcal{A}$ against the $\mathsf{R^cT\text{-}IND\text{-}CCA}$ security of the combiner $\mathcal{K} = \mathsf{XtM}[\mathcal{K}_1, \mathcal{K}_2, \mathcal{M}]$ necessarily implies that there exist efficient adversaries against the $\mathsf{R^cT\text{-}IND\text{-}CCA}$ security of either of the two KEMs or the $\mathsf{R^cT\text{-}OT\text{-}s\text{-}EUF}$ security of $\mathcal{M}$, respectively. The game hops are depicted in Figure 5.9.

Game$_0(\lambda)$**:** The original $\mathsf{R^cT\text{-}IND\text{-}CCA}$ game $\mathsf{G}^{\mathsf{R^cT\text{-}ind\text{-}cca}}_{\mathcal{K},\mathcal{A}}(\lambda)$ against the combiner $\mathcal{K}$.

Game$_1(\lambda)$**:** First, we replace the part of the real challenge key corresponding to $\mathcal{K}_2$, i.e., K_2 and k_2, with uniformly random and independent values $\widetilde{K_2}$ and $\widetilde{k_2}$, each of length $l(\lambda)$.

We use these randomly generated values from then onwards in all computations involving the respective keys, such as the establishment of the final MAC key $k^\star$ and the decapsulation of ciphertext parts $c_2^\star$.

We show that if $\mathcal{A}$ can efficiently distinguish *Game*$_0(\lambda)$ from *Game*$_1(\lambda)$, then there exists an efficient adversary $\mathcal{B}_1$ against the $\mathsf{R^cT\text{-}IND\text{-}CCA}$ security of $\mathcal{K}_2$. The reduction works as follows:

Algorithm $\mathcal{B}_1$ receives as input $(pk_2, c_2^\star, \kappa_{b'}^\star)$, where pk_2 is the public key, $c_2^\star$ the challenge ciphertext, and $\kappa_{b'}^\star = K_{b'} \| k_{b'}$ the real or random challenge key.

To initialize the environment for $\mathcal{A}$, the reduction $\mathcal{B}_1$ generates the key pair (pk_1, sk_1) for $\mathcal{K}_1$ and sets $pk \leftarrow (pk_1, pk_2)$. Furthermore, it computes the first challenge ciphertext portion $c_1^\star$ as well as the key share $K_1 \| k_1$ itself by running $\mathsf{Encaps}_1(pk_1)$.

$\mathcal{B}_1$ then sets $K_0^\star \leftarrow K_1 \oplus K_{b'}$ and $k \leftarrow (k_1, k_{b'})$. The challenge ciphertext for $\mathcal{A}$ is set as $c^\star \leftarrow ((c_1^\star, c_2^\star), \tau^\star)$, where $\tau^\star \leftarrow \mathsf{MAC}(k, (c_1^\star, c_2^\star))$. The reduction then initializes $\mathcal{A}$ on input $(pk, c^\star, K^\star)$.

Decapsulation queries by $\mathcal{A}$ of the form $c = c^\star$ are immediately answered with $\perp$. For any decapsulation query $c = ((c_1, c_2), \tau)$ with $c_2 \neq c_2^\star$, the reduction first computes $K_1' \| k_1' \leftarrow \mathsf{Decaps}_1(sk_1, c_1)$ by itself. $\mathcal{B}_1$ then queries c_2 to its own decapsulation oracle for

```
$Game_0(\lambda)$:
$(pk_1, sk_1) \xleftarrow{\$} \mathsf{KGen}_1(1^\lambda)$
$(pk_2, sk_2) \xleftarrow{\$} \mathsf{KGen}_2(1^\lambda)$
$pk \leftarrow (pk_1, pk_2)$
$sk \leftarrow (sk_1, sk_2)$
$(c_1^\star, K_1 \| k_1) \xleftarrow{\$} \mathsf{Encaps}_1(pk_1)$
$(c_2^\star, K_2 \| k_2) \xleftarrow{\$} \mathsf{Encaps}_2(pk_2)$
$k \leftarrow (k_1, k_2)$
$c' \leftarrow (c_1^\star, c_2^\star)$
$\tau \leftarrow \mathsf{MAC}(k, c')$
$c^\star \leftarrow (c', \tau)$
$K_0^\star \leftarrow K_1 \oplus K_2$
$K_1^\star \xleftarrow{\$} \mathscr{K}$
$b \xleftarrow{\$} \{0,1\}$
$st \xleftarrow{\$} \mathcal{A}_1^{\mathcal{O}_{\mathsf{Decaps}}}(pk, c^\star, K_b^\star)$
$b' \xleftarrow{\$} \mathcal{A}_2(st)$
return $[\![b' = b]\!]$
```

```
$Game_1(\lambda)$:
$(pk_1, sk_1) \xleftarrow{\$} \mathsf{KGen}_1(1^\lambda)$
$(pk_2, sk_2) \xleftarrow{\$} \mathsf{KGen}_2(1^\lambda)$
$pk \leftarrow (pk_1, pk_2)$
$sk \leftarrow (sk_1, sk_2)$
$(c_1^\star, K_1 \| k_1) \xleftarrow{\$} \mathsf{Encaps}_1(pk_1)$
$(c_2^\star, K_2 \| k_2) \xleftarrow{\$} \mathsf{Encaps}_2(pk_2)$
$\widetilde{K_2}, \widetilde{k_2} \xleftarrow{\$} \{0,1\}^{l(\lambda)}$
$k \leftarrow (k_1, \widetilde{k_2})$
$c' \leftarrow (c_1^\star, c_2^\star)$
$\tau \leftarrow \mathsf{MAC}(k, c')$
$c^\star \leftarrow (c', \tau)$
$K_0^\star \leftarrow K_1 \oplus \widetilde{K_2}$
$K_1^\star \xleftarrow{\$} \mathscr{K}$
$b \xleftarrow{\$} \{0,1\}$
$st \xleftarrow{\$} \mathcal{A}_1^{\mathcal{O}_{\mathsf{Decaps}}}(pk, c^\star, K_b^\star)$
$b' \xleftarrow{\$} \mathcal{A}_2(st)$
return $[\![b' = b]\!]$
```

```
$Game_2(\lambda)$:
$(pk_1, sk_1) \xleftarrow{\$} \mathsf{KGen}_1(1^\lambda)$
$(pk_2, sk_2) \xleftarrow{\$} \mathsf{KGen}_2(1^\lambda)$
$pk \leftarrow (pk_1, pk_2)$
$sk \leftarrow (sk_1, sk_2)$
$(c_1^\star, K_1 \| k_1) \xleftarrow{\$} \mathsf{Encaps}_1(pk_1)$
$(c_2^\star, K_2 \| k_2) \xleftarrow{\$} \mathsf{Encaps}_2(pk_2)$
$\widetilde{K_2}, \widetilde{k_2} \xleftarrow{\$} \{0,1\}^{l(\lambda)}$
$k \leftarrow (k_1, \widetilde{k_2})$
$c' \leftarrow (c_1^\star, c_2^\star)$
$\tau \leftarrow \mathsf{MAC}(k, c')$
$c^\star \leftarrow (c', \tau)$
$K_0^\star \leftarrow K_1 \oplus K_1 \oplus \widetilde{K_2}$
$K_1^\star \xleftarrow{\$} \mathscr{K}$
$b \xleftarrow{\$} \{0,1\}$
$st \xleftarrow{\$} \mathcal{A}_1^{\mathcal{O}_{\mathsf{Decaps}}}(pk, c^\star, K_b^\star)$
$b' \xleftarrow{\$} \mathcal{A}_2(st)$
return $[\![b' = b]\!]$
```

```
$Game_3(\lambda)$:
$(pk_1, sk_1) \xleftarrow{\$} \mathsf{KGen}_1(1^\lambda)$
$(pk_2, sk_2) \xleftarrow{\$} \mathsf{KGen}_2(1^\lambda)$
$pk \leftarrow (pk_1, pk_2)$
$sk \leftarrow (sk_1, sk_2)$
$(c_1^\star, K_1 \| k_1) \xleftarrow{\$} \mathsf{Encaps}_1(pk_1)$
$(c_2^\star, K_2 \| k_2) \xleftarrow{\$} \mathsf{Encaps}_2(pk_2)$
$\widetilde{K_2}, \widetilde{k_2} \xleftarrow{\$} \{0,1\}^{l(\lambda)}$
$k \leftarrow (k_1, \widetilde{k_2})$
$c' \leftarrow (c_1^\star, c_2^\star)$
$\tau \leftarrow \mathsf{MAC}(k, c')$
$c^\star \leftarrow (c', \tau)$
$K_0^\star \leftarrow \widetilde{K_2}$
$K_1^\star \xleftarrow{\$} \mathscr{K}$
$b \xleftarrow{\$} \{0,1\}$
$st \xleftarrow{\$} \mathcal{A}_1^{\mathcal{O}_{\mathsf{Decaps}}}(pk, c^\star, K_b^\star)$
$b' \xleftarrow{\$} \mathcal{A}_2(st)$
return $[\![b' = b]\!]$
```

```
$\mathcal{O}_{\mathsf{Decaps}}(c = ((c_1, c_2), \tau))$:
if $c = c^\star$
    return $\bot$
else
    $K_1' \| k_1' \leftarrow \mathsf{Decaps}_1(sk_1, c_1)$
    $K_2' \| k_2' \leftarrow \mathsf{Decaps}_2(sk_2, c_2)$
    $k' \leftarrow (k_1', k_2')$
    if $\mathsf{Vfy}(k', \tau, (c_1, c_2)) = 0$
        return $\bot$
    else
        return $K_1' \oplus K_2'$
```

```
$\mathcal{O}_{\mathsf{Decaps}}(c = ((c_1, c_2), \tau))$:
if $c = c^\star$
    return $\bot$
elseif $c_2 = c_2^\star$
    $K_1' \| k_1' \leftarrow \mathsf{Decaps}_1(sk_1, c_1)$
    $k' \leftarrow (k_1', \widetilde{k_2})$
    if $\mathsf{Vfy}(k', \tau, (c_1, c_2)) = 0$
        return $\bot$
    else
        return $K_1' \oplus \widetilde{K_2}$
else
    $K_1' \| k_1' \leftarrow \mathsf{Decaps}_1(sk_1, c_1)$
    $K_2' \| k_2' \leftarrow \mathsf{Decaps}_2(sk_2, c_2)$
    $k' \leftarrow (k_1', k_2')$
    if $\mathsf{Vfy}(k', \tau, (c_1, c_2)) = 0$
        return $\bot$
    else
        return $K_1' \oplus K_2'$
```

```
$\mathcal{O}_{\mathsf{Decaps}}(c = ((c_1, c_2), \tau))$:
if $c = c^\star$
    return $\bot$
elseif $c_2 = c_2^\star$
    $K_1' \| k_1' \leftarrow \mathsf{Decaps}_1(sk_1, c_1)$
    $k' \leftarrow (k_1', \widetilde{k_2})$
    if $\mathsf{Vfy}(k', \tau, (c_1, c_2)) = 0$
        return $\bot$
    else
        return $K_1' \oplus K_1 \oplus \widetilde{K_2}$
else
    $K_1' \| k_1' \leftarrow \mathsf{Decaps}_1(sk_1, c_1)$
    $K_2' \| k_2' \leftarrow \mathsf{Decaps}_2(sk_2, c_2)$
    $k' \leftarrow (k_1', k_2')$
    if $\mathsf{Vfy}(k', \tau, (c_1, c_2)) = 0$
        return $\bot$
    else
        return $K_1' \oplus K_2'$
```

```
$\mathcal{O}_{\mathsf{Decaps}}(c = ((c_1, c_2), \tau))$:
if $c = c^\star$
    return $\bot$
elseif $c_2 = c_2^\star$
    return $\bot$
else
    $K_1' \| k_1' \leftarrow \mathsf{Decaps}_1(sk_1, c_1)$
    $K_2' \| k_2' \leftarrow \mathsf{Decaps}_2(sk_2, c_2)$
    $k' \leftarrow (k_1', k_2')$
    if $\mathsf{Vfy}(k', \tau, (c_1, c_2)) = 0$
        return $\bot$
    else
        return $K_1' \oplus K_2'$
```

Figure 5.9: Game hops for proof of Theorem 5.10.

$\mathcal{K}_2$ to receive the answer $K_2' \| k_2'$. Set $k' \leftarrow (k_1', k_2')$. If $\mathsf{Vfy}(k', \tau, (c_1, c_2)) = 0$, $\mathcal{B}_1$ returns $\bot$ to the adversary. Otherwise it returns $K_1' \oplus K_2'$.

If the decapsulation query is such that $c = ((c_1, c_2^\star), \tau)$, then $\mathcal{B}_1$ uses $K_{b'}$ as the decapsulation of $c_2^\star$, $k_{b'}$ as the corresponding MAC key, and then continues as in the previous case.

At some point, $\mathcal{A}$ terminates and outputs a guess bit b_{guess}. The reduction $\mathcal{B}_1$ outputs the same bit b_{guess}.

Clearly, $\mathcal{B}_1$ perfectly simulates the environment for $\mathcal{A}$, corresponding to $Game_0(\lambda)$ if the challenge key $\kappa_{b'}^\star$ is the actual key ($b' = 0$), and corresponding to $Game_1(\lambda)$ if $\kappa_{b'}^\star$ is random ($b' = 1$).

Furthermore, $\mathcal{B}_1$ is of the same two-stage type as $\mathcal{A}$. Hence, we have

$$\mathsf{Adv}^{\mathsf{G}_0}_{\mathcal{K},\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{G}_1}_{\mathcal{K},\mathcal{A}}(\lambda) + 2 \cdot \mathsf{Adv}^{\mathsf{R^cT\text{-}ind\text{-}cca}}_{\mathcal{K}_2,\mathcal{B}_1}(\lambda).$$

The game hop works completely analogous for $\mathcal{K}_1$ being secure and $\mathcal{K}_2$ insecure, yielding an adversary $\mathcal{B}_2$.

$Game_2(\lambda)$**:** In a syntactical change, we replace the now random value $\widetilde{K_2}$ by $K_1 \oplus \widetilde{K_2}$, where K_1 is the encapsulated key contributed from $\mathcal{K}_1$ in the challenge generation. We leave $\widetilde{k_2}$

unaltered. Effectively, the modification means that the encapsulated key in the challenge ciphertext is now $\widetilde{K_2} = K_1 \oplus (K_1 \oplus \widetilde{K_2})$.

Since $\widetilde{K_2}$ and K_1 are independent, the distributions of $\widetilde{K_2}$ and $K_1 \oplus \widetilde{K_2}$ are identical, so the adversary's advantage does not change and we have:

$$\mathsf{Adv}^{\mathsf{G}_1}_{\mathcal{K},\mathcal{A}}(\lambda) = \mathsf{Adv}^{\mathsf{G}_2}_{\mathcal{K},\mathcal{A}}(\lambda).$$

In $Game_2(\lambda)$ the adversary now receives a random value as the challenge key and the MAC is also computed independently of the challenge bit b over a random key part $\widetilde{k_2}$. To finalize the proof, we need to argue that the adversary cannot gain any advantage via queries to the decapsulation oracle of the form $c = ((\cdot, c_2^\star), \cdot)$.

However, we show in the next game hop that the difference in advantage that can be gained by such queries is negligible, as this could only be exploited if the adversary were able to forge a MAC.

$Game_3(\lambda)$**:** Thus, lastly, we modify the decapsulation oracle such that it rejects all ciphertexts of the form $c = ((\cdot, c_2^\star), \cdot)$ by outputting $\perp$.

We argue that any efficient distinguisher $\mathcal{A}$ between $Game_2(\lambda)$ and $Game_3(\lambda)$ immediately implies an efficient adversary $\mathcal{B}_3$ against the one-time strong unforgeability of the MAC scheme $\mathcal{M}$. This is due to the fact that, in order to enforce decapsulation of such ciphertexts, the adversary will have to provide a ciphertext $c = ((c_1, c_2^\star), \tau)$ with $c_1 \neq c_1^\star$ and τ a valid MAC tag. The reduction $\mathcal{B}_3$ works as follows:

In order to initialize $\mathcal{A}$, the reduction $\mathcal{B}_3$ first generates the key pairs (pk_1, sk_1) and (pk_2, sk_2), as well as encapsulations $(c_1^\star, K_1 \| k_1)$ and $(c_2^\star, K_2 \| k_2)$ under the respective public keys itself. It then outputs $1, k_1, (c_1^\star, c_2^\star)$ as input to its challenger to then receive a MAC tag $\tau^\star$. The reduction then runs $\mathcal{A}$ on input $(c^\star, K^\star)$, where $c^\star \leftarrow ((c_1^\star, c_2^\star), \tau)$ and $K^\star \xleftarrow{\$} \{0,1\}^{l(\lambda)}$ is a uniformly random key.

For all of $\mathcal{A}$'s decapsulation queries that are not identical to the challenge ciphertext and where $c_2 \neq c_2^\star$, the reduction $\mathcal{B}_3$ computes the appropriate responses itself, using its knowledge of the corresponding secrets. Whenever $\mathcal{A}$ queries the challenge ciphertext, $\mathcal{B}_3$ returns $\perp$. For decapsulation queries of the form $c = ((c_1, c_2^\star), \tau)$, $\mathcal{B}_3$ records these queries in a list $\mathcal{L}$ and returns $\perp$ to the adversary.

$\mathcal{B}_3$ runs all queries $((c_1, c_2^\star), \tau) \in \mathcal{L}$ by its verification oracle on input $(k_1, \tau, (c_1, c_2^\star))$, where $K_1 \| k_1 \leftarrow \mathsf{Decaps}_1(sk_1, c_1)$. If for some query, say, $((c_1', c_2^\star), \tau')$ the response of $\mathcal{O}_{\mathsf{Vfy}}$ is 1, $\mathcal{B}_3$ outputs $(k_1', \tau', (c_1', c_2^\star))$ with $K_1' || k_1' \leftarrow \mathsf{Decaps}_1(sk_1, c_1')$ as its forgery.

At some point $\mathcal{A}$ terminates, potentially outputting a bit b_{guess}.

In $Game_2(\lambda)$, both the challenge key as well as the internal MAC keys are each uniformly random strings of length $l(\lambda)$, independent of the actual bit b in the game. This is also the case in $\mathcal{B}_3$'s simulation for $\mathcal{A}$. In particular, the second part of the MAC key is provided as a uniformly random value by the challenger in the $\mathsf{OT\text{-}sEUF}$ game. The only way $\mathcal{B}_3$'s simulation could be detected by $\mathcal{A}$ is if $\mathcal{A}$ made a query for a fresh ciphertext $(c_1, c_2^\star)$ with a valid MAC tag that would require a response different from $\perp$. But then, $\mathcal{B}_3$ has already found its forgery and can abort the simulation.

Since $\mathcal{B}_3$ is of the same type $\mathsf{R^cT}$ as $\mathcal{A}$, we have that

$$\mathsf{Adv}^{\mathsf{G}_2}_{\mathcal{K},\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{G}_3}_{\mathcal{K},\mathcal{A}}(\lambda) + \mathsf{Adv}^{\mathsf{R^cT\text{-}ot\text{-}s\text{-}euf}}_{\mathcal{M},\mathcal{B}_3}(\lambda).$$

At this point, the secret bit b is perfectly hidden from $\mathcal{A}$. Both the challenge key as well as the keys used within $\mathcal{O}_{\mathsf{Decaps}}$ are uniformly random strings, independent of b. Thus, the adversary can do no better than guessing and we have:

$$\mathsf{Adv}^{\mathsf{G}_3}_{\mathcal{K},\mathcal{A}}(\lambda) \leq 0,$$

which yields the final bound. □

Application of XOR-then-MAC in protocols. Another viable approach to protect against "mix-and-match" attacks against the combined key would be to make the key derivation depend on both keys and both ciphertexts. However, many real-world protocols already include a MAC over the transcript to provide integrity and authenticity. The key for this MAC is often derived from the session key and the transcript includes the information exchanged for key agreement such as the KEM ciphertexts. This already present MAC may then be leveraged to forgo the additional MAC over the ciphertext in the XOR-then-MAC combiner. An example of this can be found in the `Finished` message in the Transport Layer Security protocol [Res18].

5.3.2 Hybrid Modes in TLS 1.3

The last two combiners, the dual-PRF combiner and the nested dual-PRF combiner, are inspired by drafts by Whyte et al. [WFZGM17] and Schank and Stebila [SS17] that were put forward to enable hybrid modes in TLS 1.3 [Res18].

In TLS 1.3, HKDF [Kra10, KE10] is used as the key derivation function. HKDF can be split into an extraction and an expansion step. In TLS 1.3 extraction is applied to the raw (EC)DH shared secret. The output is then expanded, where the (hashed) transcript is included within the label. In the presence of a second key share, at least two approaches are feasible: one is to extract from the concatenation of the two secrets (this is the approach taken by Whyte et al. [WFZGM17] and [KK18, OQS18]). The second approach is to insert an additional extract-then-expand step (this is the approach taken by Schank and Stebila [SS17]). Both abstracted key schedules are depicted in Figure 5.10 and are treated in more detail in the following sections.

5.3.3 dualPRF: Dual-PRF Combiner

Our second combiner is based on the primitive of dual pseudorandom functions [BCK96, Bel06, BL15] and is motivated by Whyte et al.'s proposal for enabling a hybrid key exchange mode in TLS 1.3 [WFZGM17]. The dualPRF combiner captures their proposal by modeling the HKDF extraction as a dual PRF dPRF and the HKDF expansion step as a pseudorandom function F.

We will define dual PRFs shortly in the two-stage setting, but for now we want to give an informal intuition: a dual PRF $\mathsf{dPRF}(K, x)$ is a PRF, whenever either the key material K is random (i.e., $\mathsf{dPRF}(K, \cdot)$ is a PRF), or alternatively when the input label x is random (i.e., $\mathsf{dPRF}(\cdot, x)$ is a PRF). This means, that pseudorandomness can be extracted if either of the two inputs carries sufficient entropy. In the standard PRF scenario this applies only to the first input component.

It has been shown that HMAC is a secure MAC if its compression function satisfies the dual-PRF property. Bellare and Lysyanskaya [BL15] gave a confirmation of this dual-PRF assumption for HMAC and therefore also HKDF.

Combiner description. Unfortunately, to achieve robustness, it is not enough to simply use the dual PRF on the two input keys, i.e., to compute the combined KEM key as $\mathsf{dPRF}(K_1, K_2)$. To see this, assume $\mathcal{K}_1$ is broken such that an adversary may be able to maul the challenge

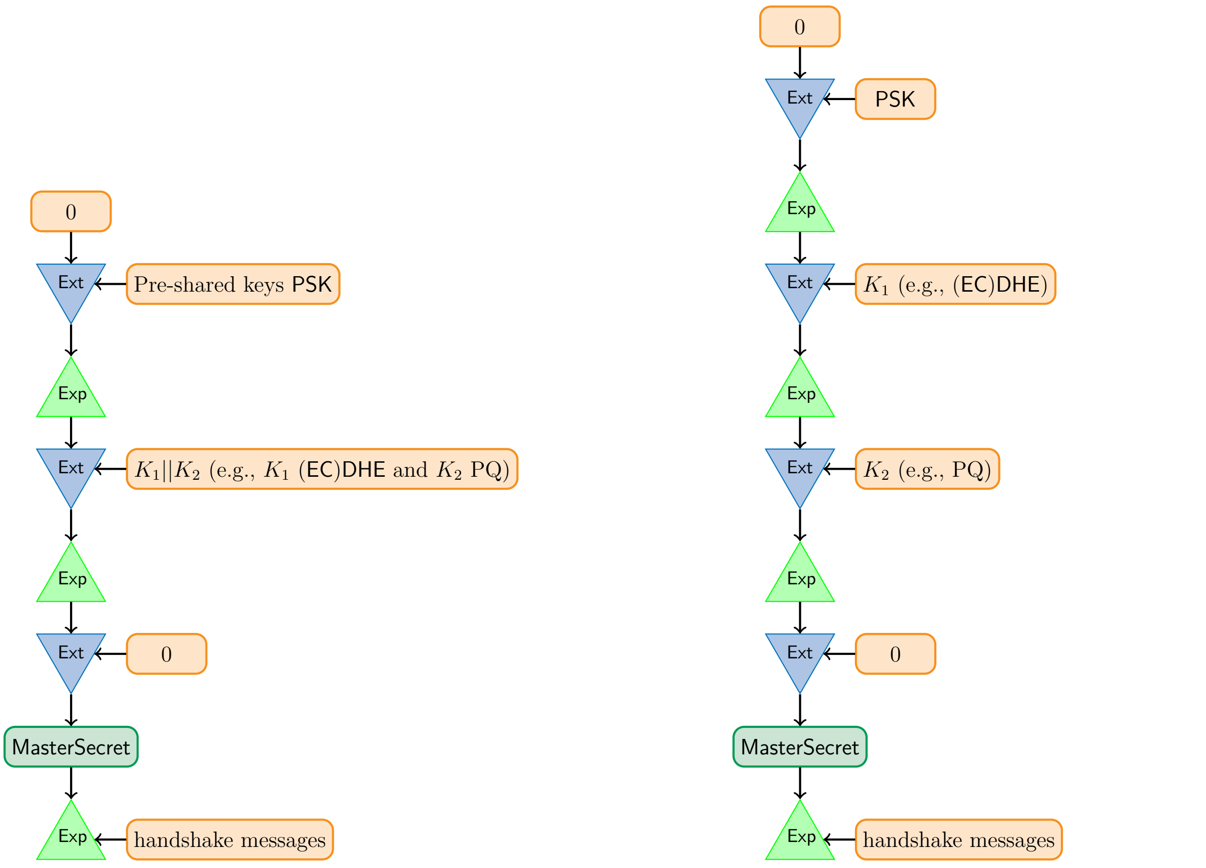

Figure 5.10: Excerpt from altered TLS 1.3 key schedule as proposed in [WFZGM17] (left) and as proposed in [SS17] (right) to enable a hybrid mode.

ciphertext $(c_1^\star, c_2^\star)$ into $(c_1, c_2^\star)$, where $c_1 \neq c_1^\star$ but they both encapsulate the same key K_1. Even if $\mathcal{K}_2$ remains completely secure, an adversary then only needs a single decapsulation query (with the mauled ciphertext) to the decapsulation oracle to recover the key $\mathsf{dPRF}(K_1, K_2)$ and thus distinguish the challenge key from random.

To avoid this, our dual-PRF combiner, depicted in Figure 5.11, runs the output of dPRF through a pseudorandom function, where it acts as the PRF key and the ciphertext pairs constitute the label, i.e., the combined key is given as $\mathsf{F}(\mathsf{dPRF}(K_1, K_2), (c_1, c_2))$.

```
$\mathsf{KGen}_{\mathsf{dualPRF}}(1^\lambda)$:
1  $(pk_1, sk_1) \xleftarrow{\$} \mathsf{KGen}_1(1^\lambda)$
2  $(pk_2, sk_2) \xleftarrow{\$} \mathsf{KGen}_2(1^\lambda)$
3  $pk \leftarrow (pk_1, pk_2)$
4  $sk \leftarrow (sk_1, sk_2)$
5  $\mathbf{return}\ (pk, sk)$

$\mathsf{Encaps}_{\mathsf{dualPRF}}(pk)$:
6  $(c_1, K_1) \xleftarrow{\$} \mathsf{Encaps}_1(pk_1)$
7  $(c_2, K_2) \xleftarrow{\$} \mathsf{Encaps}_2(pk_2)$
8  $c \leftarrow (c_1, c_2)$
9  $K' \leftarrow \mathsf{dPRF}(K_1, K_2)$
10 $K \leftarrow \mathsf{F}(K', c)$
11 $\mathbf{return}\ (c, K)$

$\mathsf{Decaps}_{\mathsf{dualPRF}}(sk, c)$:
12 $K_1' \leftarrow \mathsf{Decaps}_1(sk_1, c_1)$
13 $K_2' \leftarrow \mathsf{Decaps}_2(sk_2, c_2)$
14 $K'' \leftarrow \mathsf{dPRF}(K_1', K_2')$
15 $\mathbf{return}\ \mathsf{F}(K'', c)$
```

Figure 5.11: KEM constructed by the dual-PRF combiner $\mathsf{dualPRF}[\mathcal{K}_1, \mathcal{K}_2, \mathsf{dPRF}, \mathsf{F}]$.

Dual-PRF security. In the following, we give the two-stage equivalents of the security definitions for PRF and dual-PRF security:

Definition 5.11 (Two-stage PRF security)**.** *Let* $\mathsf{F} : \{0,1\}^{\kappa(\lambda)} \times \{0,1\}^{\iota(\lambda)} \to \{0,1\}^{\omega(\lambda)}$ *be an efficient keyed function with key length* $\kappa(\lambda)$*, input length* $\iota(\lambda)$ *and output length* $\omega(\lambda)$*. Let* $\mathsf{G}^{\mathsf{X^cZ\text{-}prf\text{-}sec}}_{\mathsf{F},\mathcal{A}}$ *be defined as in Figure 5.12. We call* F *an* $\mathsf{X^cZ}$-secure pseudorandom function *if for all QPT adversaries* $\mathcal{A}$ *of type* $\mathsf{X^cZ}$ *the advantage function defined as*

$$\mathsf{Adv}^{\mathsf{X^cZ\text{-}prf\text{-}sec}}_{\mathsf{F},\mathcal{A}}(\lambda) := \left|\Pr\left[\mathsf{G}^{\mathsf{X^cZ\text{-}prf\text{-}sec}}_{\mathsf{F},\mathcal{A}}(\lambda) = 1\right] - \frac{1}{2}\right|$$

is negligible in the security parameter λ*.*

```
$\mathsf{G}^{\mathsf{X^cZ\text{-}prf\text{-}sec}}_{\mathsf{F},\mathcal{A}}(\lambda)$:
$K \xleftarrow{\$} \{0,1\}^{\kappa(\lambda)}$
$g \xleftarrow{\$} \{\text{functions } f : \{0,1\}^{\iota(\lambda)} \to \{0,1\}^{\omega(\lambda)}\}$
$b \xleftarrow{\$} \{0,1\}$
$st \xleftarrow{\$} \mathcal{A}_1^{\mathcal{O}_{\mathsf{PRF}}}$
$b' \xleftarrow{\$} \mathcal{A}_2(st)$
return $[\![b' = b]\!]$

$\mathcal{O}_{\mathsf{PRF}}(x)$:
if $b = 0$
    return $\mathsf{F}(K, x)$
else
    return $g(x)$
```

Figure 5.12: Definition of pseudorandom function F with respect to two-stage adversaries $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$.

The two-stage definition of dual-PRF security follows easily from this. Recall that a function F is said to be a dual PRF if it is a pseudorandom function when keyed with either of its two inputs. More formally:

Definition 5.12 (Two-stage dual-PRF security)**.** *Let* $\mathsf{F} : \{0,1\}^{\kappa(\lambda)} \times \{0,1\}^{\iota(\lambda)} \to \{0,1\}^{\omega(\lambda)}$ *be an efficient keyed function key length* $\kappa(\lambda)$*, input length* $\iota(\lambda)$ *and output length* $\omega(\lambda)$*. Define* $\mathsf{F}' : \{0,1\}^{\iota(\lambda)} \times \{0,1\}^{\kappa(\lambda)} \to \{0,1\}^{\omega(\lambda)}$ *such that* $\mathsf{F}'(x, K) := \mathsf{F}(K, x)$*. Let* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *be a two-stage QPT adversary of type* $\mathsf{X^cZ}$ *interacting with* F, F' *in the Game* $\mathsf{G}^{\mathsf{X^cZ\text{-}dprf\text{-}sec}}_{\mathsf{F},\mathcal{A}}(\lambda)$ *given in Figure 5.13.*

We say that F *is an* $\mathsf{X^cZ}$-secure dual pseudorandom function *if both* F *and* F' *are* $\mathsf{X^cZ}$*-secure pseudorandom functions according to Definition 5.11.*

In particular, for all QPT $\mathsf{X^cZ}$ *adversaries* $\mathcal{A}$ *the advantage function defined as*

$$\begin{aligned}\mathsf{Adv}^{\mathsf{X^cZ\text{-}dprf\text{-}sec}}_{\mathsf{F},\mathcal{A}}(\lambda) &:= \left|\Pr\left[\mathsf{G}^{\mathsf{X^cZ\text{-}dprf\text{-}sec}}_{\mathsf{F},\mathcal{A}}(\lambda) = 1\right] - \frac{1}{2}\right| \\ &= \max\left\{\mathsf{Adv}^{\mathsf{X^cZ\text{-}prf\text{-}sec}}_{\mathsf{F},\mathcal{A}}(\lambda), \mathsf{Adv}^{\mathsf{X^cZ\text{-}prf\text{-}sec}}_{\mathsf{F}',\mathcal{A}}(\lambda)\right\}\end{aligned}$$

is negligible in the security parameter λ*.*

```
$\mathsf{G}^{\mathsf{X^cZ\text{-}dprf\text{-}sec}}_{\mathsf{F},\mathcal{A}}(\lambda)$:
$K \xleftarrow{\$} \{0,1\}^{\kappa(\lambda)}$
$x \xleftarrow{\$} \{0,1\}^{\iota(\lambda)}$
$g \xleftarrow{\$} \{\text{functions } f : \{0,1\}^{\iota(\lambda)} \to \{0,1\}^{\omega(\lambda)}\}$
$g' \xleftarrow{\$} \{\text{functions } f : \{0,1\}^{\kappa(\lambda)} \to \{0,1\}^{\omega(\lambda)}\}$
$b \xleftarrow{\$} \{0,1\}$
$st \xleftarrow{\$} \mathcal{A}_1^{\mathcal{O}^{\mathsf{F}}_{\mathsf{PRF}}, \mathcal{O}^{\mathsf{F}'}_{\mathsf{PRF}}}$
$b' \xleftarrow{\$} \mathcal{A}_2(st)$
return $[\![b' = b]\!]$

$\mathcal{O}^{\mathsf{F}}_{\mathsf{PRF}}(x)$:
if $b = 0$
    return $\mathsf{F}(K, x)$
else
    return $g(x)$

$\mathcal{O}^{\mathsf{F}'}_{\mathsf{PRF}}(K)$:
if $b = 0$
    return $\mathsf{F}'(x, K)$
else
    return $g'(K)$
```

Figure 5.13: Definition of a dual pseudorandom function F with respect to two-stage adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$.

Security of the dual-PRF combiner. We can now show that the dual-PRF combiner constitutes a hybrid KEM: the combined KEM $\mathcal{K} = \mathsf{dualPRF}[\mathcal{K}_1, \mathcal{K}_2, \mathsf{dPRF}, \mathsf{F}]$ preserves the security offered by the strongest of the input KEMs $\mathcal{K}_1, \mathcal{K}_2$, if dPRF and F offer the same level of security.

Theorem 5.13 (Dual-PRF is robust)**.** *Let $\mathcal{K}_1$ be an* $\mathsf{X^cZ\text{-}IND\text{-}ATK}$*-secure KEM or $\mathcal{K}_2$ be a* $\mathsf{U^cW\text{-}IND\text{-}ATK}$*-secure KEM. Set* $\mathsf{R^cT} = \max\{\mathsf{X^cZ}, \mathsf{U^cW}\}$*. Furthermore, let* $\mathsf{dPRF} : \mathscr{K}_1 \times \mathscr{K}_2 \to \mathscr{K}'$ *be an* $\mathsf{R^cT}$*-secure dual PRF, and* $\mathsf{F} : \mathscr{K}' \times \{0,1\}^\star \to \mathscr{K}$ *be an* $\mathsf{R^cT}$*-secure PRF. Then the combiner* $\mathsf{dualPRF}[\mathcal{K}_1, \mathcal{K}_2, \mathsf{dPRF}, \mathsf{F}]$ *as defined in Figure 5.11 is* $\mathsf{R^cT\text{-}IND\text{-}ATK}$ *secure.*

More precisely, for any QPT adversary $\mathcal{A}$ of type $\mathsf{R^cT}$ *against the combined KEM* $\mathcal{K} = \mathsf{dualPRF}[\mathcal{K}_1, \mathcal{K}_2, \mathsf{dPRF}, \mathsf{F}]$*, we derive efficient adversaries $\mathcal{B}_1$, $\mathcal{B}_2$, $\mathcal{B}_3$, and $\mathcal{B}_4$ such that*

$$\mathsf{Adv}^{\mathsf{R^cT\text{-}ind\text{-}atk}}_{\mathcal{K},\mathcal{A}}(\lambda) \le 2 \cdot \Big(\min \Big\{ \mathsf{Adv}^{\mathsf{R^cT\text{-}ind\text{-}atk}}_{\mathcal{K}_2,\mathcal{B}_1}(\lambda), \mathsf{Adv}^{\mathsf{R^cT\text{-}ind\text{-}atk}}_{\mathcal{K}_1,\mathcal{B}_2}(\lambda) \Big\} + \mathsf{Adv}^{\mathsf{R^cT\text{-}dprf\text{-}sec}}_{\mathsf{dPRF},\mathcal{B}_3}(\lambda) + \mathsf{Adv}^{\mathsf{R^cT\text{-}prf\text{-}sec}}_{\mathsf{F},\mathcal{B}_4}(\lambda) \Big).$$

Proof. As for the XOR-then-MAC combiner, we prove the theorem by considering a sequence of game hops. The game hops are depicted in Figure 5.14. We focus on the case that $\mathcal{K}_1$ becomes insecure. The case for when $\mathcal{K}_2$ becomes insecure proceeds similarly. Again, we show the theorem statement for $\mathsf{IND\text{-}CCA}$ security, which implies the $\mathsf{IND\text{-}CPA}$ case.

```
$Game_0(\lambda)$:
$(pk_1, sk_1) \xleftarrow{\$} \mathsf{KGen}_1(1^\lambda)$
$(pk_2, sk_2) \xleftarrow{\$} \mathsf{KGen}_2(1^\lambda)$
$pk \leftarrow (pk_1, pk_2)$
$sk \leftarrow (sk_1, sk_2)$
$(c_1^\star, K_1) \xleftarrow{\$} \mathsf{Encaps}_1(pk_1)$
$(c_2^\star, K_2) \xleftarrow{\$} \mathsf{Encaps}_2(pk_2)$
$c^\star \leftarrow (c_1^\star, c_2^\star)$
$K' \leftarrow \mathsf{dPRF}(K_1, K_2)$
$K_0^\star \leftarrow \mathsf{F}(K', c^\star)$
$K_1^\star \xleftarrow{\$} \mathscr{K}$
$b \xleftarrow{\$} \{0,1\}$
$st \xleftarrow{\$} \mathcal{A}_1^{\mathcal{O}_{\mathsf{Decaps}}}(pk, c^\star, K_b^\star)$
$b' \xleftarrow{\$} \mathcal{A}_2(st)$
return $[\![b' = b]\!]$

$\mathcal{O}_{\mathsf{Decaps}}(c = (c_1, c_2))$:
if $c = c^\star$
    return $\bot$
else
    $K_1' \leftarrow \mathsf{Decaps}_1(sk_1, c_1)$
    $K_2' \leftarrow \mathsf{Decaps}_2(sk_2, c_2)$
    $K'' \leftarrow \mathsf{dPRF}(K_1', K_2')$
    return $\mathsf{F}(K'', c)$
```

```
$Game_1(\lambda)$:
$(pk_1, sk_1) \xleftarrow{\$} \mathsf{KGen}_1(1^\lambda)$
$(pk_2, sk_2) \xleftarrow{\$} \mathsf{KGen}_2(1^\lambda)$
$pk \leftarrow (pk_1, pk_2)$
$sk \leftarrow (sk_1, sk_2)$
$(c_1^\star, K_1) \xleftarrow{\$} \mathsf{Encaps}_1(pk_1)$
$(c_2^\star, K_2) \xleftarrow{\$} \mathsf{Encaps}_2(pk_2)$
$c^\star \leftarrow (c_1^\star, c_2^\star)$
$\widetilde{K_2} \xleftarrow{\$} \mathscr{K}_2$
$K' \leftarrow \mathsf{dPRF}(K_1, \widetilde{K_2})$
$K_0^\star \leftarrow \mathsf{F}(K', c^\star)$
$K_1^\star \xleftarrow{\$} \mathscr{K}$
$b \xleftarrow{\$} \{0,1\}$
$st \xleftarrow{\$} \mathcal{A}_1^{\mathcal{O}_{\mathsf{Decaps}}}(pk, c^\star, K_b^\star)$
$b' \xleftarrow{\$} \mathcal{A}_2(st)$
return $[\![b' = b]\!]$

$\mathcal{O}_{\mathsf{Decaps}}(c = (c_1, c_2))$:
if $c = c^\star$
    return $\bot$
elseif $c_2 = c_2^\star$
    $K_1' \leftarrow \mathsf{Decaps}_1(sk_1, c_1)$
    $K'' \leftarrow \mathsf{dPRF}(K_1', \widetilde{K_2})$
    return $\mathsf{F}(K'', c)$
else
    $K_1' \leftarrow \mathsf{Decaps}_1(sk_1, c_1)$
    $K_2' \leftarrow \mathsf{Decaps}_2(sk_2, c_2)$
    $K'' \leftarrow \mathsf{dPRF}(K_1', K_2')$
    return $\mathsf{F}(K'', c)$
```

```
$Game_2(\lambda)$:
$(pk_1, sk_1) \xleftarrow{\$} \mathsf{KGen}_1(1^\lambda)$
$(pk_2, sk_2) \xleftarrow{\$} \mathsf{KGen}_2(1^\lambda)$
$pk \leftarrow (pk_1, pk_2)$
$sk \leftarrow (sk_1, sk_2)$
$(c_1^\star, K_1) \xleftarrow{\$} \mathsf{Encaps}_1(pk_1)$
$(c_2^\star, K_2) \xleftarrow{\$} \mathsf{Encaps}_2(pk_2)$
$c^\star \leftarrow (c_1^\star, c_2^\star)$
$\widetilde{K'} \xleftarrow{\$} \mathscr{K}'$
$K_0^\star \leftarrow \mathsf{F}(\widetilde{K'}, c^\star)$
$K_1^\star \xleftarrow{\$} \mathscr{K}$
$b \xleftarrow{\$} \{0,1\}$
$st \xleftarrow{\$} \mathcal{A}_1^{\mathcal{O}_{\mathsf{Decaps}}}(pk, c^\star, K_b^\star)$
$b' \xleftarrow{\$} \mathcal{A}_2(st)$
return $[\![b' = b]\!]$

$\mathcal{O}_{\mathsf{Decaps}}(c = (c_1, c_2))$:
if $c = c^\star$
    return $\bot$
elseif $c_2 = c_2^\star$
    $K_1' \leftarrow \mathsf{Decaps}_1(sk_1, c_1)$
    $K'' \leftarrow \mathsf{dPRF}(K_1', \widetilde{K_2})$
    if $K_1' = K_1$
        return $\mathsf{F}(\widetilde{K'}, c)$
    else
        return $\mathsf{F}(K'', c)$
else
    $K_1' \leftarrow \mathsf{Decaps}_1(sk_1, c_1)$
    $K_2' \leftarrow \mathsf{Decaps}_2(sk_2, c_2)$
    $K'' \leftarrow \mathsf{dPRF}(K_1', K_2')$
    return $\mathsf{F}(K'', c)$
```

```
$Game_3(\lambda)$:
$(pk_1, sk_1) \xleftarrow{\$} \mathsf{KGen}_1(1^\lambda)$
$(pk_2, sk_2) \xleftarrow{\$} \mathsf{KGen}_2(1^\lambda)$
$pk \leftarrow (pk_1, pk_2)$
$sk \leftarrow (sk_1, sk_2)$
$(c_1^\star, K_1) \xleftarrow{\$} \mathsf{Encaps}_1(pk_1)$
$(c_2^\star, K_2) \xleftarrow{\$} \mathsf{Encaps}_2(pk_2)$
$c^\star \leftarrow (c_1^\star, c_2^\star)$
$\widetilde{K_2} \xleftarrow{\$} \mathscr{K}$
$\widetilde{K'} \xleftarrow{\$} \mathscr{K}'$
$\widetilde{K} \leftarrow \mathscr{K}$
$K_0^\star \leftarrow \widetilde{K}$
$K_1^\star \xleftarrow{\$} \mathscr{K}$
$b \xleftarrow{\$} \{0,1\}$
$st \xleftarrow{\$} \mathcal{A}_1^{\mathcal{O}_{\mathsf{Decaps}}}(pk, c^\star, K_b^\star)$
$b' \xleftarrow{\$} \mathcal{A}_2(st)$
return $[\![b' = b]\!]$

$\mathcal{O}_{\mathsf{Decaps}}(c = (c_1, c_2))$:
if $c = c^\star$
    return $\bot$
elseif $c_2 = c_2^\star$
    $K_1' \leftarrow \mathsf{Decaps}_1(sk_1, c_1)$
    $K'' \leftarrow \mathsf{dPRF}(K_1', \widetilde{K_2})$
    if $K_1' = K_1$
        return $\mathsf{F}(\widetilde{K'}, c)$
    else
        return $\mathsf{F}(K'', c)$
else
    $K_1' \leftarrow \mathsf{Decaps}_1(sk_1, c_1)$
    $K_2' \leftarrow \mathsf{Decaps}_2(sk_2, c_2)$
    $K'' \leftarrow \mathsf{dPRF}(K_1', K_2')$
    return $\mathsf{F}(K'', c)$
```

Figure 5.14: Game hops for proof of Theorem 5.13.

Game$_0(\lambda)$**:** The original $\mathsf{R^cT\text{-}IND\text{-}CCA}$ game $\mathsf{G}^{\mathsf{R^cT\text{-}ind\text{-}cca}}_{\mathcal{K},\mathcal{A}}$ against the combiner $\mathcal{K} = \mathsf{dualPRF}[\mathcal{K}_1, \mathcal{K}_2, \mathsf{dPRF}, \mathsf{F}]$.

Game$_1(\lambda)$**:** First, we replace the value K_2 by a uniformly random element $\widetilde{K_2}$ from the same key space $\mathscr{K}_2$. The real key for challenge $b = 0$ is then computed as $K_0^\star \leftarrow \mathsf{F}(\mathsf{dPRF}(K_1, \widetilde{K_2}), c^\star)$. If the adversary asks a decapsulation query $c = (c_1, c_2^\star)$, where $c_1 \neq c_1^\star$, then the value $\widetilde{K_2}$ is used instead of $\mathsf{Decaps}_2(sk_2, c_2^\star)$.

We argue that if there exists an efficient adversary $\mathcal{A}$ that can distinguish *Game*$_0(\lambda)$ from *Game*$_1(\lambda)$, this implies the existence of an efficient adversary $\mathcal{B}_1$ against the $\mathsf{IND\text{-}CCA}$ security of $\mathcal{K}_2$. The reduction works as follows:

Algorithm $\mathcal{B}_1$ receives as input $(pk_2, c_2^\star, k_{b'}^\star)$, where pk_2 is the public key, $c_2^\star$ the challenge ciphertext, and $k_{b'}^\star$ the real or random challenge key in the two-stage $\mathsf{IND\text{-}CCA}$ game for $\mathcal{K}_2$.

To initialize $\mathcal{A}$, the reduction $\mathcal{B}_1$ generates the key pair (pk_1, sk_1) for $\mathcal{K}_1$ and sets $pk \leftarrow (pk_1, pk_2)$. It computes the first challenge ciphertext portion $c_1^\star$ as well as the key share K_1 itself by running $\mathsf{Encaps}_1(pk_1)$. $\mathcal{B}$ then sets $K^\star \leftarrow \mathsf{F}(\mathsf{dPRF}(K_1, k_{b'}^\star), c^\star)$, where $c^\star = (c_1^\star, c_2^\star)$. The reduction then initializes $\mathcal{A}$ on input $(pk, c^\star, K^\star)$.

Decapsulation queries by $\mathcal{A}$ of the form $c = c^\star$ are immediately answered with $\perp$.

For any decapsulation query $c = (c_1, c_2)$ such that $c_2 \neq c_2^\star$, the reduction first computes $K_1 \leftarrow \mathsf{Decaps}_1(sk_1, c_1)$ by itself. $\mathcal{B}_1$ then queries c_2 to its own decapsulation oracle for $\mathcal{K}_2$ to receive the answer K_2. $\mathcal{B}_1$ then returns $\mathsf{F}(\mathsf{dPRF}(K_1, K_2), c)$ to the adversary.

If the decapsulation query is such that $c = (c_1, c_2^\star)$, then $\mathcal{B}_1$ uses $k_{b'}^\star$ in place of the decapsulation of $c_2^\star$. It then proceeds to compute the response as in the previous case, i.e., it computes the decapsulation K_1 of c_1 itself and derives the final response as $\mathsf{F}(\mathsf{dPRF}(K_1, k_{b'}^\star), c)$.

At some point, $\mathcal{A}$ terminates and outputs a guess bit b_{guess}. The reduction $\mathcal{B}_1$ then outputs the same bit b_{guess}.

Clearly, $\mathcal{B}_1$ simulates the environment for $\mathcal{A}$ corresponding to *Game*$_0(\lambda)$ if the challenge key $k_{b'}^\star$ is the actual key (i.e., if $b' = 0$), and corresponding to *Game*$_1(\lambda)$ if $k_{b'}^\star$ is random (i.e., $b' = 1$).

Hence, we have

$$\mathsf{Adv}_{\mathcal{K},\mathcal{A}}^{\mathsf{G_0}}(\lambda) \leq \mathsf{Adv}_{\mathcal{K},\mathcal{A}}^{\mathsf{G_1}}(\lambda) + 2 \cdot \mathsf{Adv}_{\mathcal{K}_2,\mathcal{B}_1}^{\mathsf{X^cZ\text{-}ind\text{-}cca}}(\lambda).$$

The game hop works completely analogous for $\mathcal{K}_1$ being secure and $\mathcal{K}_2$ insecure, yielding an adversary $\mathcal{B}_2$ against the $\mathsf{IND\text{-}CCA}$ security of $\mathcal{K}_1$.

Game$_2(\lambda)$**:** Next, we replace the value $\mathsf{dPRF}(K_1, \widetilde{K_2})$ used for computing the challenge value $K_0^\star$ by a uniformly random value $\widetilde{K'} \xleftarrow{\$} \mathscr{K}'$. Furthermore, for any query of $\mathcal{A}$ to the decapsulation oracle of the form $c = (c_1', c_2^\star)$, where $c_1' \neq c_1^\star$ but $K_1 \leftarrow \mathsf{Decaps}_1(sk_1, c_1')$ we change the decapsulation oracle to immediately return $\mathsf{F}(\widetilde{K'}, c)$.

We argue that if an adversary can efficiently distinguish *Game*$_1(\lambda)$ from *Game*$_2(\lambda)$, this would imply an efficient adversary $\mathcal{B}_3$ against the PRF security of dPRF' and thus contradict the dual-PRF security of dPRF. The reduction works as follows:

The reduction $\mathcal{B}_3$ generates key pairs (pk_1, sk_1), (pk_2, sk_2), as well as $c_1^\star, c_2^\star$ (with encapsulated keys K_1 and K_2, respectively). It then queries its PRF oracle $\mathcal{O}_{\mathsf{PRF}}$ about K_1 to receive a value $y_{b'}^\star$ which is either $\mathsf{dPRF}'(K, K_1) = \mathsf{dPRF}(K_1, K)$ with $K \xleftarrow{\$} \mathscr{K}_2$ chosen by its challenger ($b' = 0$), or a uniformly random element computed as $g(K_1)$ from $\mathscr{K}_2$ ($b' = 1$) in the PRF security game for dPRF'.

$\mathcal{B}_3$ then initializes $\mathcal{A}$ on input $pk \leftarrow (pk_1, pk_2)$, challenge ciphertext $c^\star \leftarrow (c_1^\star, c_2^\star)$ and challenge key $K^\star \leftarrow \mathsf{F}(y_{b'}^\star, c^\star)$.

Queries of $\mathcal{A}$ on the challenge ciphertext are immediately rejected by returning $\perp$. Each decapsulation query for $c' = (c_1', c_2')$ with $c_2' \neq c_2^\star$ is answered with the help of sk_1 and sk_2 by first decapsulating the ciphertexts to K_1' and K_2', respectively, and then computing the final key as $\mathsf{F}(\mathsf{dPRF}(K_1', K_2'), c')$.

If $\mathcal{A}$ queries a ciphertext $c' = (c_1', c_2^\star)$, then $\mathcal{B}_3$ first decapsulates c_1' to K_1'. If $K_1' = K_1$, $\mathcal{B}_3$ returns $\mathsf{F}(y_{b'}^\star, c')$. Otherwise, $\mathcal{B}_3$ queries its oracle on K_1' to receive the answer y which is either $\mathsf{dPRF}'(K, K_1') = \mathsf{dPRF}(K_1', K)$ in case $b' = 0$ or a uniformly random element computed as $g(K_1')$ if $b' = 1$. Then, the answer for $\mathcal{A}$ is computed as $\mathsf{F}(y, c')$.

At some point, $\mathcal{A}$ terminates with output bit b_{guess}. $\mathcal{B}_3$ outputs the same guess.

Observe that if $\mathcal{B}_3$ receives the real value, i.e., if $b' = 0$, then $K^\star \leftarrow \mathsf{F}(\mathsf{dPRF}(K_1, K), c^\star)$ for random K and the situation is as in $Game_1(\lambda)$ since K is distributed as $\widetilde{K}_2$. If on the other hand $\mathcal{B}_3$ receives $y_{b'}^\star \leftarrow g(K_1)$, i.e., if $b' = 1$, then $K^\star \leftarrow \mathsf{F}(y_{b'}^\star, c^\star)$ is distributed as in $Game_2(\lambda)$.

Hence, if $\mathcal{A}$ is able to efficiently distinguish the two games, $\mathcal{B}_3$ can win the PRF game against dPRF' and thus the dual-PRF game against dPRF with non-negligible advantage.

For $\mathcal{K}_1$ being secure and $\mathcal{K}_2$ insecure, the argument is analogous, as dPRF is a dual PRF, i.e., $\mathsf{dPRF}(K, \cdot)$ is also a pseudorandom function.

We have:

$$\mathsf{Adv}^{\mathsf{G}_1}_{\mathcal{K},\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{G}_2}_{\mathcal{K},\mathcal{A}}(\lambda) + 2 \cdot \mathsf{Adv}^{\mathsf{R^cT\text{-}dprf\text{-}sec}}_{\mathsf{dPRF},\mathcal{B}_3}(\lambda).$$

Game$_3(\lambda)$**:** In the last step, we replace the value $\mathsf{F}(\widetilde{K'}, c^\star)$ by a uniformly random value $\widetilde{K} \xleftarrow{\$} \mathscr{K}$. Note, that there is no change in the description of the decapsulation oracle.

We argue that any efficient distinguisher $\mathcal{A}$ between $Game_2(\lambda)$ and $Game_3(\lambda)$ immediately yields an efficient adversary $\mathcal{B}_4$ against the pseudorandomness of F:

The reduction $\mathcal{B}_4$ first generates key pairs (pk_1, sk_1), (pk_2, sk_2) and then the challenge ciphertext parts $c_1^\star$ and $c_2^\star$ via $\mathsf{Encaps}_1(pk_1)$ and $\mathsf{Encaps}_2(pk_2)$, respectively. It then queries its oracle $\mathcal{O}_{\mathsf{PRF}}$ on $c^\star \leftarrow (c_1^\star, c_2^\star)$ and receives some element $y_{b'}^\star$ as output which is either $F(K, c^\star)$ for random $K \leftarrow \mathscr{K}'$ chosen by its challenger $(b' = 0)$, or a uniformly random element from $\mathscr{K}$ computed as some $g(c^\star)$ $(b' = 1)$. Let $pk \leftarrow (pk_1, pk_2)$ and $c^\star \leftarrow (c_1^\star, c_2^\star)$. $\mathcal{B}_4$ then initializes $\mathcal{A}$ on input $(pk, c^\star, K^\star)$, where $K^\star \leftarrow y_{b'}^\star$.

Decapsulation queries by $\mathcal{A}$ of the form $c = c^\star$ are immediately answered with $\perp$. Whenever $\mathcal{A}$ asks a decapsulation query of the form (c_1, c_2), where $c_1 \neq c_1^\star$ and $c_2 \neq c_2^\star$, $\mathcal{B}_4$ simply decapsulates with the help of its secret keys sk_1 and sk_2 and returns the same answer as the real decapsulation oracle.

For queries of the form $c' = (c_1', c_2^\star)$, $\mathcal{B}_4$ decapsulates c_1' using via $K_1' \leftarrow \mathsf{Decaps}_1(sk_1, c_1')$. If $K_1' \neq K_1$, $\mathcal{B}_4$ uses $\mathsf{dPRF}(K_1', \widetilde{K_2})$ to compute the answer $\mathsf{F}(\mathsf{dPRF}(K_1', \widetilde{K_2}), c')$. Otherwise, i.e., if $\mathsf{Decaps}_1(sk_1, c_1') = K_1$, $\mathcal{B}_4$ queries its oracle on c' to receive either $F(K, c')$ in case $b' = 0$ or $g(c')$ if $b' = 1$ and forwards this to the adversary.

At some point, $\mathcal{A}$ terminates and outputs a guess bit b_{guess}. $\mathcal{B}_4$ then outputs the same guess bit.

We see that if $b' = 0$, $\mathcal{B}_4$ faithfully simulates $Game_2(\lambda)$ and if $b' = 1$, $\mathcal{B}_4$ simulates $Game_3(\lambda)$. Thus, if $\mathcal{A}$ can distinguish between the two games, $\mathcal{B}_4$ also wins its PRF game with non-negligible advantage.

We have:

$$\mathsf{Adv}^{\mathsf{G}_2}_{\mathcal{K},\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{G}_3}_{\mathcal{K},\mathcal{A}}(\lambda) + 2 \cdot \mathsf{Adv}^{\mathsf{R^cT\text{-}prf\text{-}sec}}_{\mathsf{F},\mathcal{B}_4}(\lambda).$$

The analogous argument applies when $\mathcal{K}_1$ is secure.

To conclude the proof, we note that in $Game_3(\lambda)$ neither the challenge ciphertext and challenge key nor the decapsulation oracle depend on the secret bit b. Thus, the adversary can do no better than guessing and we have

$$\mathsf{Adv}^{\mathsf{G}_3}_{\mathcal{K},\mathcal{A}}(\lambda) \leq 0$$

which yields the final bound and shows that $\mathcal{K} = \mathsf{dualPRF}[\mathcal{K}_1, \mathcal{K}_2, \mathsf{dPRF}, \mathsf{F}]$ is a hybrid KEM that achieves $\mathsf{R^cT\text{-}IND\text{-}CCA}$ security. □

5.3.4 N: Nested Dual-PRF Combiner

In a different approach, the hybrid TLS 1.3 draft by Schank and Stebila [SS17] extends the key schedule by an additional extraction to accommodate the second key share (cf. Figure 5.10). The main structure of extraction followed by expansion is however the same in both proposals.

Combiner description. We can thus simply revise the dualPRF combiner from the previous section with a further preprocessing step for the key K_1 such that we get an intermediate key $k_e \xleftarrow{\$} \mathsf{Ext}(0, K_1)$, where Ext is a PRF modeling the extraction step of HKDF. We refer to this combiner as the nested dual-PRF combiner N and it is depicted in Figure 5.15.

```
KGen_N(1^λ):
 1  $(pk_1, sk_1) \xleftarrow{\$} \mathsf{KGen}_1(1^\lambda)$
 2  $(pk_2, sk_2) \xleftarrow{\$} \mathsf{KGen}_2(1^\lambda)$
 3  $pk \leftarrow (pk_1, pk_2)$
 4  $sk \leftarrow (sk_1, sk_2)$
 5  return $(pk, sk)$

Encaps_N(pk):
 6  $(c_1, K_1) \xleftarrow{\$} \mathsf{Encaps}_1(pk_1)$
 7  $(c_2, K_2) \xleftarrow{\$} \mathsf{Encaps}_2(pk_2)$
 8  $c = (c_1, c_2)$
 9  $k_e = \mathsf{Ext}(0, K_1)$
10  $k_d = \mathsf{dPRF}(k_e, K_2)$
11  $K = \mathsf{F}(k_d, c)$
12  return $(c, K)$

Decaps_N(sk, c):
13  $K_1' \leftarrow \mathsf{Decaps}_1(sk_1, c_1)$
14  $K_2' \leftarrow \mathsf{Decaps}_2(sk_2, c_2)$
15  $k_e' = \mathsf{Ext}(0, K_1')$
16  $k_d' = \mathsf{dPRF}(k_e', K_2')$
17  return $\mathsf{F}(k_d', (c_1, c_2))$
```

Figure 5.15: Hybrid KEM constructed by the nested dual-PRF combiner $\mathsf{N}[\mathcal{K}_1, \mathcal{K}_2, \mathsf{Ext}, \mathsf{dPRF}, \mathsf{F}]$.

Security of the nested dual-PRF combiner. We confirm that the nested dual-PRF combiner N is indeed a robust KEM combiner in our desired sense:

Theorem 5.14 (Nested dual-PRF is robust)**.** *Let $\mathcal{K}_1$ be an $\mathsf{X^cZ\text{-}IND\text{-}ATK}$-secure KEM or $\mathcal{K}_2$ be a $\mathsf{U^cW\text{-}IND\text{-}ATK}$-secure KEM. Set $\mathsf{R^cT} = \max\{\mathsf{X^cZ}, \mathsf{U^cW}\}$. Then let $\mathsf{dPRF} : \mathscr{K}' \times \mathscr{K}_2 \to \mathscr{K}''$ be an $\mathsf{R^cT}$-secure dual PRF and $\mathsf{F} : \mathscr{K}'' \times \{0,1\}^\star \to \mathscr{K}$ and $\mathsf{Ext} : \{0,1\}^* \times \mathscr{K}_1 \to \mathscr{K}'$ be $\mathsf{R^cT}$-secure PRFs. Then the nested dual-PRF combiner $\mathcal{K} = \mathsf{N}[\mathcal{K}_1, \mathcal{K}_2, \mathsf{Ext}, \mathsf{dPRF}, \mathsf{F}]$ as defined in Figure 5.15 is $\mathsf{R^cT\text{-}IND\text{-}ATK}$ secure.*

More precisely, for any QPT adversary $\mathcal{A}$ of type $\mathsf{R^cT}$ against the combined KEM $\mathcal{K} = \mathsf{N}[\mathcal{K}_1, \mathcal{K}_2, \mathsf{Ext}, \mathsf{dPRF}, \mathsf{F}]$, we derive efficient adversaries $\mathcal{B}_1$, $\mathcal{B}_2$, ..., $\mathcal{B}_5$ such that

$$\begin{aligned}\mathsf{Adv}^{\mathsf{R^cT\text{-}ind\text{-}atk}}_{\mathcal{K},\mathcal{A}}(\lambda) \quad \leq \quad & 2 \cdot \Big(\min\Big\{ \mathsf{Adv}^{\mathsf{R^cT\text{-}ind\text{-}atk}}_{\mathcal{K}_2,\mathcal{B}_1}(\lambda), \mathsf{Adv}^{\mathsf{R^cT\text{-}ind\text{-}atk}}_{\mathcal{K}_1,\mathcal{B}_2}(\lambda) \Big\} \\ & + \mathsf{Adv}^{\mathsf{R^cT\text{-}prf\text{-}sec}}_{\mathsf{Ext},\mathcal{B}_3}(\lambda) + \mathsf{Adv}^{\mathsf{R^cT\text{-}dprf\text{-}sec}}_{\mathsf{dPRF},\mathcal{B}_4}(\lambda) + \mathsf{Adv}^{\mathsf{R^cT\text{-}dprf\text{-}sec}}_{\mathsf{F},\mathcal{B}_5}(\lambda) \Big).\end{aligned}$$

Proof. The proof proceeds as for the $\mathsf{dualPRF}$ combiner (Thm. 5.13), except that there is an additional step in which we replace the output of $\mathsf{Ext}(0, K_1)$ by a uniformly random string. As done before, we argue that an adversary cannot detect such a change, as otherwise this would immediately imply an efficient adversary against the pseudorandomness of Ext. The proof then proceeds as for the $\mathsf{dualPRF}$ combiner. □

5.4 Modeling Hybrid Authenticated Key Exchange

The rest of this chapter deals with the question of how to build authenticated key exchange from hybrid key encapsulation mechanisms. After introducing the relevant security definitions in this section, in Section 5.5 we present a generic compiler that uses a hybrid key encapsulation mechanism alongside SigMA-style authentication [Kra03]. This is a well-known compiler to build authenticated key exchange from an unauthenticated key agreement protocol. It is for example used in the design of TLS 1.3 [Res18] and the Internet Key Exchange protocol [KHN+14].

5.4.1 Two-stage Bellare–Rogaway Security Definitions

The model we use for hybrid authenticated key exchange is based on the Bellare–Rogaway model as introduced in Chapter 4.2. Recall that the goal of the adversary is to distinguish session keys in so-called *fresh* sessions of its choice from random. The adversary interacts with honest participants in the authenticated key exchange protocol executions via oracle queries which allow the adversary to fully control all network communications. Additionally, these queries enable the adversary to learn long-term secret values of participants and session keys. We adjust the Bellare–Rogaway model to accommodate two-stage adversaries in order to allow for a security analysis with respect to adversaries of different quantum capabilities. As before, we only allow adversaries that are at most post-quantum, i.e., whose interaction with parties is through classical oracle queries.

The following two definitions transfer the properties of $\mathsf{BR\text{-}Match}$ security and BR key secrecy to the two-stage setting. As for the other two-stage security definitions, the first stage adversary is actively interacting with the protocol, whereas the second stage adversary only has access to oracle queries that are not tied to the execution of the protocol, i.e., it may for example still learn secret key values.

Definition 5.15 (Two-stage $\mathsf{BR\text{-}Match}$ security). *Let KE be an authenticated key exchange protocol and $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be a QPT two-stage adversary of type $\mathsf{X^cZ}$ interacting with KE via the queries defined in Section 4.2.1 in the following game $\mathsf{G}^{\mathsf{X^cZ\text{-}BR\text{-}Match},\mathcal{D}}_{\mathsf{KE},\mathcal{A}}(\lambda)$:*

Setup. *The challenger generates long-term public/secret-key pairs for each participant $U \in \mathcal{U}$.*

Query Phase 1. *The adversary $\mathcal{A}_1$ receives the generated public keys and has access to the queries $\mathsf{NewSession}$, Send, Reveal, $\mathsf{Corrupt}$, and Test.*

Stage Change. *The adversary $\mathcal{A}_1$ passes some state st to the second stage adversary $\mathcal{A}_2$ and terminates.*

Query Phase 2. *$\mathcal{A}_2$ may now perform local computations on state st and only has access to queries Reveal and $\mathsf{Corrupt}$.*

Stop. *At some point, the adversary $\mathcal{A}_2$ stops with no output.*

We say that $\mathcal{A}$ wins the game $\mathsf{G}^{\mathsf{X^cZ\text{-}BR\text{-}Match},\mathcal{D}}_{\mathsf{KE},\mathcal{A}}(\lambda)$ if at least one of the following conditions holds:

1. Different session keys in partnered sessions:

   There exist two distinct sessions π *and* π' *with* $\pi.\mathsf{sid} = \pi'.\mathsf{sid} \neq \bot$, *and* $\pi.\mathsf{st}_{\mathsf{exec}}, \pi'.\mathsf{st}_{\mathsf{exec}} \neq \mathsf{rejected}$, *but* $\pi.\mathsf{K} \neq \pi'.\mathsf{K}$.

2. Different intended partner in partnered sessions:

   There exist two sessions $\pi := \pi^{k}_{U,V}$ *and* $\pi' := \pi^{k'}_{V',U'}$ *such that* $\pi.\mathsf{sid} = \pi'.\mathsf{sid} \neq \bot$, $\pi.\mathsf{role} = \mathsf{initiator}$, *and* $\pi'.\mathsf{role} = \mathsf{responder}$, *but* $U \neq U'$ *or* $V \neq V'$.

3. More than two sessions share the same session identifier:

   There exist at least three sessions π, π', *and* π'' *such that* π, π', π'' *are pairwise distinct, but* $\pi.\mathsf{sid} = \pi'.\mathsf{sid}' = \pi''.\mathsf{sid} \neq \bot$.

We say KE *is* X^cZ-BR-Match secure *or simply* two-stage BR-Match secure *if for all QPT* $\mathsf{X^cZ}$ *adversaries* $\mathcal{A}$ *the advantage function*

$$\mathsf{Adv}^{\mathsf{X^cZ\text{-}BR\text{-}Match}}_{\mathsf{KE},\mathcal{A}}(\lambda) := \Pr\left[\mathsf{G}^{\mathsf{X^cZ\text{-}BR\text{-}Match},\mathcal{D}}_{\mathsf{KE},\mathcal{A}}(\lambda) = 1\right]$$

is negligible in the security parameter λ.

Definition 5.16 (Two-stage BR key secrecy)**.** *Let* KE *be a key exchange protocol with key distribution* $\mathcal{D}$. *Let* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *be a QPT adversary of type* $\mathsf{X^cZ}$ *interacting with* KE *via the queries defined in Section 4.2.1 in the following game* $\mathsf{G}^{\mathsf{X^cZ\text{-}BR},\mathcal{D}}_{\mathsf{KE},\mathcal{A}}(\lambda)$:

Setup. *The challenger generates long-term public/secret-key pairs for each participant* $U \in \mathcal{U}$, *chooses the test bit* $b_{\mathsf{test}} \xleftarrow{\$} \{0,1\}$ *at random, and sets* $\mathsf{lost} \leftarrow \mathsf{false}$.

Query Phase 1. *Adversary* $\mathcal{A}_1$ *receives the generated public keys and may query* NewSession, Send, Reveal, Corrupt, *and* Test.

Stage Change. *At some point,* $\mathcal{A}_1$ *terminates and outputs some state st to be passed to the second stage adversary* $\mathcal{A}_2$.

Query Phase 2. $\mathcal{A}_2$ *may now perform local computations on state st, but may query only* Reveal *and* Corrupt.

Guess. *At some point,* $\mathcal{A}_2$ *terminates and outputs a guess bit* b_{guess}.

Finalize. *The challenger sets* $\mathsf{lost} \leftarrow \mathsf{true}$ *if the following condition holds.*

Adversary has tested and revealed the key in a single session or in two partnered sessions:

There exist two (not necessarily distinct) sessions π, π' *such that* $\pi.\mathsf{sid} = \pi'.\mathsf{sid}$, $\pi.\mathsf{st}_{\mathsf{key}} = \mathsf{revealed}$, *and* $\pi'.\mathsf{tested} = \mathsf{true}$.

$\mathcal{A}$ *wins the game if* $b_{\mathsf{guess}} = b_{\mathsf{test}}$ *and* $\mathsf{lost} = \mathsf{false}$. *We say* KE *provides* X^cZ-BR key secrecy *(with/without forward secrecy) if for all QPT* $\mathsf{X^cZ}$ *adversaries* $\mathcal{A}$ *the advantage function*

$$\mathsf{Adv}^{\mathsf{X^cZ\text{-}BR},\mathcal{D}}_{\mathsf{KE},\mathcal{A}}(\lambda) := \left|\Pr\left[\mathsf{G}^{\mathsf{X^cZ\text{-}BR},\mathcal{D}}_{\mathsf{KE},\mathcal{A}}(\lambda) = 1\right] - \frac{1}{2}\right|$$

is negligible in the security parameter.

Finally, these two notions constitute what it means for a key exchange protocol to be two-stage BR secure:

Definition 5.17 (Two-stage BR security)**.** *We say a key exchange protocol* KE *is* X^cZ-BR secure *(with/without forward secrecy) if* KE *provides* BR-Match *security (Def. 5.15) and* X^cZ-BR *key secrecy (with/without forward secrecy) (Def. 5.16).*

Implications between two-stage BR notions

Similarly to the two-stage security notions of indistinguishability for KEMs, the following implications hold for two-stage BR security.

Proposition 5.18 ($\mathsf{Q^cQ\text{-}BR} \implies \mathsf{C^cQ\text{-}BR} \implies \mathsf{C^cC\text{-}BR}$)**.** *Let* KE *be an authenticated key exchange protocol. If* KE *is* $\mathsf{Q^cQ\text{-}BR}$ *secure, then* KE *is also* $\mathsf{C^cQ\text{-}BR}$ *secure. If* KE *is* $\mathsf{C^cQ\text{-}BR}$ *secure, then it is also* $\mathsf{C^cC\text{-}BR}$ *secure.*

Proof. Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be a two-stage adversary. The proof is straightforward since every classical adversary can be seen as a quantum adversary that forgoes its additional quantum power. It thus holds that $\mathsf{Adv}^{\mathsf{Q^cQ\text{-}BR}}_{\mathsf{KE},\mathcal{A}}(\lambda) \geq \mathsf{Adv}^{\mathsf{C^cQ\text{-}BR}}_{\mathsf{KE},\mathcal{A}}(\lambda) \geq \mathsf{Adv}^{\mathsf{C^cC\text{-}BR}}_{\mathsf{KE},\mathcal{A}}(\lambda)$.

□

Separations between two-stage BR notions

We give separations based on the compiler $\mathcal{C}_{\mathsf{SigMA}}$ that we will prove secure in the next section. The full proofs of the security are very much like the proof given in that section for the generic construction, thus we only give a proof sketch here, highlighting the relevant points.

Theorem 5.19 ($\mathsf{C^cC\text{-}BR} \not\Longrightarrow \mathsf{C^cQ\text{-}BR} \not\Longrightarrow \mathsf{Q^cQ\text{-}BR}$)**.** *There exists an authenticated key exchange protocol* KE' *that is* $\mathsf{C^cC\text{-}BR}$ *secure but not* $\mathsf{C^cQ\text{-}BR}$ *secure. Furthermore, there exists an authenticated key exchange protocol* KE'' *that is* $\mathsf{C^cQ\text{-}BR}$ *secure but not* $\mathsf{Q^cQ\text{-}BR}$ *secure.*

Proof Sketch. We show both separations separately and focus on the case of BR key secrecy.

$\underline{\mathsf{C^cC\text{-}BR}\text{ key secrecy } \not\Longrightarrow \mathsf{C^cQ\text{-}BR}\text{ key secrecy.}}$ Consider the Diffie–Hellman KEM depicted in Figure 5.16. It is well known that $\mathsf{KE}' = \mathcal{C}_{\mathsf{SigMA}}[\mathcal{K}, \mathcal{S}, \mathcal{M}, \mathsf{KDF}]$ with DH key agreement $\mathcal{K}$ and secure signatures and MACs is a classically BR-secure protocol. However, a $\mathsf{C^cQ\text{-}BR}$ adversary can extract the secret key $K = g^{xy}$ from the transcript by using its local quantum power to break the discrete logarithm of, e.g., either $pk = g^x$ or $c = g^y$, hence trivially breaking key secrecy.

```
KGen(1^λ):
    $x \xleftarrow{\$} \mathbb{Z}_q$
    $pk \leftarrow g^x$
    $sk \leftarrow x$
    return $(pk, sk)$
```

```
Encaps(pk):
    $y \xleftarrow{\$} \mathbb{Z}_q$
    $c \leftarrow g^y$
    $K \leftarrow pk^y$
    return $(c, K)$
```

```
Decaps(sk, c):
    $K' \leftarrow c^{sk}$
    return $K'$
```

Figure 5.16: Diffie-Hellman KEM $\mathcal{K} = (\mathsf{KGen}, \mathsf{Encaps}, \mathsf{Decaps})$.

$\underline{\mathsf{C^cQ\text{-}BR}\text{ key secrecy } \not\Longrightarrow \mathsf{Q^cQ\text{-}BR}\text{ key secrecy.}}$ Let $\mathcal{K} = (\mathsf{KGen}, \mathsf{Encaps}, \mathsf{Decaps})$ be a $\mathsf{Q\text{-}IND\text{-}CPA}$-secure KEM, let $\mathcal{S}$ be an $\mathsf{C^cC}$-unforgeable signature scheme, $\mathcal{M}$ be a $\mathsf{C^cC}$-unforgeable message authentication scheme and KDF be a $\mathsf{C^cQ}$-secure key derivation function modeled as a PRF.

Theorem 5.21 establishes that the compiler $\mathsf{KE}'' = \mathcal{C}_{\mathsf{SigMA}}[\mathcal{K}, \mathcal{S}, \mathcal{M}, \mathsf{KDF}]$ achieves $\mathsf{C^cQ\text{-}BR}$ security, and in particular key secrecy. However we see that the compiler $\mathcal{C}_{\mathsf{SigMA}}$ does not achieve $\mathsf{Q^cQ\text{-}BR}$ key secrecy. A locally quantum adversary $\mathcal{A}_1$ in the first stage can for example forge signatures in $Game_3(\lambda)$ of the proof. Thus, in the end, the correct identification of the associated session can no longer be guaranteed. This enables the adversary to ask a Reveal query on this session and hence trivially break key secrecy.

□

5.4.2 Further two-stage definitions

The compiler for the hybrid AKE that will be presented in the next Section 5.5 makes use of signatures and message authentication codes. In the following we give the security definitions of these primitives for two-stage adversaries.

Definition 5.20 (Two-stage $\mathsf{EUF\text{-}CMA}$ security of signatures)**.** *Let $\mathcal{S} = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vfy})$ be a signature scheme and let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be a two-stage QPT $\mathsf{X^cZ}$ adversary interacting with $\mathcal{S}$ in the Game $\mathsf{G}^{\mathsf{X^cZ\text{-}euf\text{-}cma}}_{\mathcal{S},\mathcal{A}}$ given in Figure 5.17.*

We say that $\mathcal{S}$ is $\mathsf{X^yZ\text{-}euf\text{-}cma}$-secure if for any QPT $\mathsf{X^cZ}$ adversary $\mathcal{A}$ the advantage function

$$\mathsf{Adv}^{\mathsf{X^cZ\text{-}euf\text{-}cma}}_{\mathcal{S},\mathcal{A}}(\lambda) := \Pr\left[\mathsf{G}^{\mathsf{X^cZ\text{-}euf\text{-}cma}}_{\mathcal{S},\mathcal{A}}(\lambda) = 1\right]$$

is negligible in the security parameter λ.

```
$\mathsf{G}^{\mathsf{X^cZ\text{-}euf\text{-}cma}}_{\mathcal{S},\mathcal{A}}(\lambda)$:
1 $\mathcal{L}_{\mathcal{O}_{\mathsf{Sign}}} \leftarrow \emptyset$
2 $(pk, sk) \xleftarrow{\$} \mathsf{KGen}(1^\lambda)$
3 $(st) \xleftarrow{\$} \mathcal{A}_1^{\mathcal{O}_{\mathsf{Sign}}}(pk)$
4 $(\sigma', m') \xleftarrow{\$} \mathcal{A}_2(st)$
5 return $[\![\mathsf{Vfy}(pk, \sigma', m') \wedge m' \notin \mathcal{L}_{\mathcal{O}_{\mathsf{Sign}}}]\!]$

$\mathcal{O}_{\mathsf{Sign}}(m)$:
6 $\mathcal{L}_{\mathcal{O}_{\mathsf{Sign}}} \leftarrow \mathcal{L}_{\mathcal{O}_{\mathsf{Sign}}} \cup \{m\}$
7 return $\mathsf{Sign}(sk, m)$
```

Figure 5.17: Existential unforgeability under chosen message attacks of $\mathcal{S} = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vfy})$ with respect to two-stage adversaries $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$.

Existential unforgeability under chosen-message attack of a message authentication scheme $\mathcal{M} = (\mathsf{KGen}, \mathsf{MAC}, \mathsf{Vfy})$ is defined analogously to Definition 5.20.

5.5 A Generic Compiler for Hybrid AKE

We are now ready to answer the question of how to achieve hybrid authenticated key exchange from hybrid key encapsulation mechanisms in the two-stage adversary setting. There exists a vast body of literature on compilers for authenticated key exchange [BCK98, KY07, BCGP08, JKSS12, LSY$^+$14, dSGSW17]. We opt for the approach of building hybrid AKE from a hybrid key encapsulation mechanism combined with SigMA-style authentication [Kra03]. The compiled protocol, denoted by $\mathcal{C}_{\mathsf{SigMA}}$ is depicted in Figure 5.18. It takes as input an $\mathsf{IND\text{-}CPA}$-secure (hybrid) KEM $\mathcal{K}$, a signature scheme $\mathcal{S}$, a message authentication scheme $\mathcal{M}$ —both existentially unforgeable under chosen-message attacks— and a secure key derivation function KDF modeled as a PRF.

Compiler description. Informally, the compiled protocol proceeds as follows:
First, Alice and Bob execute the hybrid KEM to achieve key agreement on a shared key value K. They each retain their local view of the previous communication, the *transcript* t, and use the shared value K to derive the final session key as well as a MAC key K_{mac}.

In the next phase, Alice and Bob authenticate each other. Alice goes first in sampling a nonce r_A and sending it to Bob. Bob also samples his nonce r_B and uses his long-term signing key sk_B to issue a signature σ_B over the transcript t of the key exchange, as well as the nonces r_A and r_B. To indicate, that Bob has the role of $\mathsf{responder}$ in this key exchange, Bob will also include the designated label "0" within the signature.

Furthermore, Bob authenticates his identity B by computing a message authentication tag τ_b as $\mathsf{MAC}(K_{\mathsf{mac}}, \text{"0"}||B)$. He then sends his response B, r_B, σ_B, τ_b to Alice. Alice verifies both

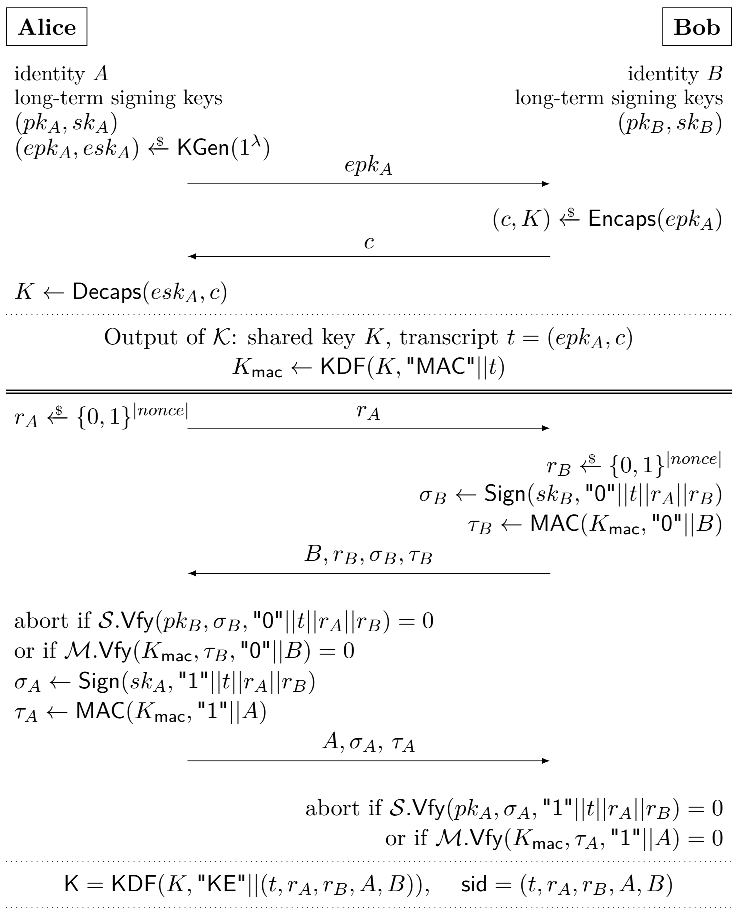
Alice
Bob
identity A
long-term signing keys
(pk_A, sk_A)
$(epk_A, esk_A) \xleftarrow{\$} \mathsf{KGen}(1^\lambda)$
identity B
long-term signing keys
(pk_B, sk_B)
epk_A
$(c, K) \xleftarrow{\$} \mathsf{Encaps}(epk_A)$
c
$K \leftarrow \mathsf{Decaps}(esk_A, c)$
Output of $\mathcal{K}$: shared key K, transcript $t = (epk_A, c)$
$K_{\mathsf{mac}} \leftarrow \mathsf{KDF}(K, \text{"MAC"}||t)$
$r_A \xleftarrow{\$} \{0,1\}^{|nonce|}$
r_A
$r_B \xleftarrow{\$} \{0,1\}^{|nonce|}$
$\sigma_B \leftarrow \mathsf{Sign}(sk_B, \text{"0"}||t||r_A||r_B)$
$\tau_B \leftarrow \mathsf{MAC}(K_{\mathsf{mac}}, \text{"0"}||B)$
B, r_B, σ_B, τ_B
abort if $\mathcal{S}.\mathsf{Vfy}(pk_B, \sigma_B, \text{"0"}||t||r_A||r_B) = 0$
or if $\mathcal{M}.\mathsf{Vfy}(K_{\mathsf{mac}}, \tau_B, \text{"0"}||B) = 0$
$\sigma_A \leftarrow \mathsf{Sign}(sk_A, \text{"1"}||t||r_A||r_B)$
$\tau_A \leftarrow \mathsf{MAC}(K_{\mathsf{mac}}, \text{"1"}||A)$
A, σ_A, τ_A
abort if $\mathcal{S}.\mathsf{Vfy}(pk_A, \sigma_A, \text{"1"}||t||r_A||r_B) = 0$
or if $\mathcal{M}.\mathsf{Vfy}(K_{\mathsf{mac}}, \tau_A, \text{"1"}||A) = 0$
$\mathsf{K} = \mathsf{KDF}(K, \text{"KE"}||(t, r_A, r_B, A, B)), \quad \mathsf{sid} = (t, r_A, r_B, A, B)$

Figure 5.18: Compiled protocol $\mathcal{C}_{\mathsf{SigMA}}$ building AKE from KEMs, signatures, and MACs.

the signature and the MAC she received from Bob and aborts if either is invalid. Analogously to Bob, she then computes her signature σ_A and tag τ_A and sends them, along with her identity A, to Bob. Alice then accepts after deriving the final session key.

Finally, Bob verifies the received signature and MAC from Alice and aborts if either is invalid. If the verifications are successful, Bob derives the final session key and accepts.

Security analysis. We now show that the compiled protocol $\mathcal{C}_{\mathsf{SigMA}}$ achieves two-stage BR security (cf. Definition 4.3).

One would generally assume that the "weakest" primitive determines the overall security of the compiled protocol. However, as it turns out this intuition is not quite correct. Naturally, in case either the unauthenticated key agreement $\mathcal{K}$ or the key derivation function KDF are only classically secure, we cannot expect more than classical $\mathsf{C^cC\text{-}BR}$ security of the compiled protocol. Similarly, post-quantum $\mathsf{Q^cQ\text{-}BR}$ security can only be achieved if *all* components of the protocol provide this level of security. Interestingly though, for the compiled protocol to guarantee security against future-quantum adversaries ($\mathsf{C^cQ\text{-}BR}$ security) it suffices for the signature and MAC scheme to be classically secure when combined with $\mathsf{Q\text{-}IND\text{-}CPA}$-secure key encapsulation and at least $\mathsf{C^cQ}$-secure key derivation. This is due to the fact that, in the proof of the main theorem (Thm. 5.21) the signatures and message authentication codes of the test session $\pi^\star$ and its potential partner $\pi^\star_{\mathsf{a}}$ must be received while the first stage adversary, which is classical, is present. As soon as the stage change occurs, the adversary loses the power to interfere with still ongoing sessions and message transmissions via the then withdrawn Send

oracle.

Theorem 5.21. *Let $\mathcal{K}$ be an* R-IND-CPA*-secure key encapsulation mechanism, $\mathcal{S}$ be an* $\mathsf{S^cT}$*-unforgeable signature scheme, $\mathcal{M}$ be a* $\mathsf{U^cV}$*-unforgeable message authentication scheme, and* KDF *be a* $\mathsf{W^cX}$*-secure key derivation function with output key space $\mathcal{D}$ modeled as a PRF. Then the compiled protocol $\mathcal{C}_{\mathsf{SigMA}}$ is* $\mathsf{Y^cZ}$-BR *secure with forward secrecy, where*

- $\mathsf{Y^cZ} = \mathsf{C^cC}$*, if either the key encapsulation mechanism $\mathcal{K}$ or the key derivation function* KDF *are only classically secure, i.e., if either* $\mathsf{R} = \mathsf{C}$ *or* $\mathsf{W^cX} = \mathsf{C^cC}$*.*
- $\mathsf{Y^cZ} = \mathsf{C^cQ}$*, if the employed signature and MAC scheme are at least classically secure, i.e., if* $\mathsf{S^cT}, \mathsf{U^cV} \in \{\mathsf{C^cC}, \mathsf{C^cQ}\}$ *(and* $\mathsf{R} = \mathsf{Q}$*,* $\mathsf{W^cX} \geq \mathsf{C^cQ}$*).*
- $\mathsf{Y^cZ} = \mathsf{Q^cQ}$*, if all components are resistant against post-quantum adversaries, i.e.,* $\mathsf{S^cT} = \mathsf{U^cV} = \mathsf{W^cX} = \mathsf{Q^cQ}$ *(and* $\mathsf{R} = \mathsf{Q}$*).*

More precisely, for any efficient QPT adversary $\mathcal{A}$ of type $\mathsf{Y^cZ}$ *there exist efficient adversaries $\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3$, and $\mathcal{B}_4$ such that*

$$\begin{aligned}\mathsf{Adv}^{\mathsf{Y^cZ\text{-}BR}}_{\mathcal{C}_{\mathsf{SigMA}},\mathcal{A}}(\lambda) \leq\ & n_s^2 \cdot 2^{|nonce|} + n_s \cdot \Big(n_u \cdot \mathsf{Adv}^{\mathsf{S^cT\text{-}euf\text{-}cma}}_{\mathcal{S},\mathcal{B}_1}(\lambda) \\ & + n_s \cdot \Big(2 \cdot \mathsf{Adv}^{\mathsf{R\text{-}ind\text{-}cpa}}_{\mathcal{K},\mathcal{B}_2}(\lambda) + 2 \cdot \mathsf{Adv}^{\mathsf{W^cX\text{-}prf\text{-}sec}}_{\mathsf{KDF},\mathcal{B}_3}(\lambda) + \mathsf{Adv}^{\mathsf{U^cV\text{-}euf\text{-}cma}}_{\mathcal{M},\mathcal{B}_4}(\lambda)\Big)\Big),\end{aligned}$$

where n_s denotes the maximum number of sessions, $|nonce|$ is the bit-length of the nonces r_A and r_B, and n_u the maximum number of participants.

To prove Theorem 5.21, we show the required properties, BR-Match security and $\mathsf{Y^cZ}$-BR key secrecy, separately.

Proof of Match security. Let t be the transcript of the key encapsulation mechanism $\mathcal{K}$ between parties A and B. Recall that the session identifier sid is set to be $\mathsf{sid} \leftarrow (t, r_A, r_B, A, B)$ which consists of public information only. We argue that $\mathcal{A}$ cannot achieve any of the three winning conditions defined in Def. 5.15 with non-negligible probability:

Ad 1. Partnered sessions agree on the session identifier sid, which fixes the transcript t and hence also the input value K to the key derivation function. Consequently, partnered sessions derive the same session keys.

Ad 2. The session identifiers contain the partner identities, thus agreement on the session identifiers implies agreement on the partner identity, excluding the possibility of different intended partners.

Ad 3. For more than two honest sessions to share a session identifier, a third honest session must share a colliding transcript t with the initial two sessions and must have a collision in either of the (randomly chosen) nonces r_A or r_B (depending on whether the third session is initiator or responder). It is easy to see that this occurs only with negligible probability.

□

Proof of key secrecy. For the proof we again apply the game hopping technique where we bound the respective difference in the adversary's advantages until the adversary cannot win anymore.

Game$_0(\lambda)$**:** The original two-stage BR key secrecy game $\mathsf{G}^{\mathsf{Y^cZ\text{-}BR},\mathcal{D}}_{\mathcal{C}_{\mathsf{SigMA}},\mathcal{A}}(\lambda)$.

Game$_1(\lambda)$**:** We start by aborting the game if two sessions of honest parties generate the same nonce r_A or r_B. Let n_s denote the maximum number of sessions and $|nonce|$ the length of the nonces. Since there are n_s possible pairs of sessions randomly selecting nonces, the probability of an abort for this reason is upper-bounded by $n_s^2 \cdot 2^{-|nonce|}$. Hence we have

$$\mathsf{Adv}^{\mathsf{G}_0}_{\mathcal{C}_{\mathsf{SigMA}},\mathcal{A}}(\lambda) \leq n_s^2 \cdot 2^{-|nonce|} + \mathsf{Adv}^{\mathsf{G}_1}_{\mathcal{C}_{\mathsf{SigMA}},\mathcal{A}}(\lambda).$$

Game$_2(\lambda)$**:** For simplicity of the argument it is beneficial to restrict the adversary to a single Test query only. This can be done via a standard hybrid argument, guessing the tested session in the beginning, and using the Reveal queries resp. random keys to answer other Test queries. Since session partnering can be checked publicly due to the public session identifiers, we can also answer consistently in such simulated Test queries. Note also that the adversary always has access to the Reveal oracle in the second stage, even if the other queries are prohibited. This strategy reduces the adversary $\mathcal{A}$'s advantage by a factor of at most $\frac{1}{n_s}$. We thus have that

$$\mathsf{Adv}^{\mathsf{G}_1}_{\mathcal{C}_{\mathsf{SigMA}},\mathcal{A}}(\lambda) \leq n_s \cdot \mathsf{Adv}^{\mathsf{G}_2}_{\mathcal{C}_{\mathsf{SigMA}},\mathcal{A}}(\lambda).$$

From now on, the test session is known in advance and we denote it by $\pi^\star$. Notice that, in order for the adversary to win, $\pi^\star$ has received all incoming messages and must have accepted before the stage change of $\mathcal{A}$ occurred.

Game$_3(\lambda)$**:** We abort the game if the test session $\pi^\star$ run by party $U \in \{A, B\}$ receives a signature σ_V on $(\text{"}\mathsf{b}\text{"}||t||r_A||r_B)$ that verifies correctly but has not been signed by some honest party V at this point. Note that this signature must have been received prior to any stage change of the adversary since the second stage of the adversary does not have access to Test. Furthermore, recall that the concerned participants, and thus their long-term secrets, may not be corrupted before the tested session has accepted. In case of a corruption of one of the involved parties after acceptance, forward secrecy is achieved since this does not interfere with the honest generation of the signature σ_V received by session $\pi^\star$ in this game hop.

The probability of an abort happening for this reason can be upper-bounded by the success probability of a reduction $\mathcal{B}_1$ against the $\mathsf{S^cT\text{-}euf\text{-}cma}$ security of the signature scheme $\mathcal{S}$. The reduction $\mathcal{B}_1$ obtains some public key $pk^\star$ as its challenge and proceeds by guessing the party V under whose identity the forgery received by $\pi^\star$ is issued. $\mathcal{B}_1$ generates all key exchange parameters as specified, except for setting $pk_V \leftarrow pk^\star$. The signing operations by V are performed by relaying the queries to the signature oracle, any other action can be carried out by $\mathcal{B}_1$ itself. If at some point the tested session $\pi^\star$ accepts a signature for a previously unsigned message, then $\mathcal{B}_1$ outputs this message-signature pair as a forgery.

Since the correctly validated signature has not been created by an honest party before, party V cannot have signed $(\text{"}\mathsf{b}\text{"}||t||r_A||r_B)$ with $\text{"}\mathsf{b}\text{"} \in \{0, 1\}$ in the past. At most it could have signed $(\text{"}\mathsf{b}'\text{"}||t||r_A||r_B)$ for $\mathsf{b}' = 1 - \mathsf{b}$. With probability $\frac{1}{n_u}$, where n_u is the total number of users, our reduction correctly anticipates the party V, and thus

$$\mathsf{Adv}^{\mathsf{G}_2}_{\mathcal{C}_{\mathsf{SigMA}},\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{G}_3}_{\mathcal{C}_{\mathsf{SigMA}},\mathcal{A}}(\lambda) + n_u \cdot \mathsf{Adv}^{\mathsf{S^cT\text{-}euf\text{-}cma}}_{\mathcal{S},\mathcal{B}_1}(\lambda).$$

Game$_4(\lambda)$**:** Next, we guess the honest session $\pi^\star_{\mathsf{a}}$ of party V that has issued the valid signature σ_V obtained by $\pi^\star$ in *Game*$_3(\lambda)$ and abort if our guess was wrong. Due to the previous game hop such a session must exist. Furthermore it is unique since there is no collision among the nonces due to $Game_1(\lambda)$. Note that the session $\pi^\star_{\mathsf{a}}$ is not necessarily partnered with the test session $\pi^\star$, since it need not have accepted yet. We therefore refer to this session as *associated.*

This game hop reduces the adversary's advantage by a factor of at most $\frac{1}{n_s}$. Thus, we have

$$\mathsf{Adv}^{\mathsf{G}_3}_{\mathcal{C}_{\mathsf{SigMA}},\mathcal{A}}(\lambda) \leq n_s \cdot \mathsf{Adv}^{\mathsf{G}_4}_{\mathcal{C}_{\mathsf{SigMA}},\mathcal{A}}(\lambda).$$

$Game_5(\lambda)$**:** We now replace the encapsulated value K in both the test session and its associated session by a uniformly random value $\widetilde{K}$ of the same length, after the KEM phase and before the parties compute the MAC key. Both parties use the key $\widetilde{K}$ instead for the subsequent computations.

If $\mathcal{A}$ were able to efficiently distinguish $Game_4(\lambda)$ and $Game_5(\lambda)$, then this implies an efficient adversary $\mathcal{B}_2$ against the $\mathsf{R\text{-}ind\text{-}cpa}$ security of $\mathcal{K}$. The reduction $\mathcal{B}_2$ simulates the environment for $\mathcal{A}$ as follows:

Initially, $\mathcal{B}_2$ receives as challenge a public key $epk^\star$ and a corresponding challenge ciphertext $c^\star$ along with the challenge key $K^\star_{b'}$.

In order to initialize $\mathcal{A}$, the reduction $\mathcal{B}_2$ generates all key exchange parameters as specified. $\mathcal{B}_2$ can initiate any new sessions that $\mathcal{A}$ establishes via $\mathsf{NewSession}$ and can answers all $\mathsf{Corrupt}$ queries with the appropriate long-term secret key.

$\mathcal{B}_2$ further simulates all Send queries. For Send queries on $\pi^\star$ and $\pi^\star_{\mathsf{a}}$, the adversary $\mathcal{B}_2$ uses $(epk^\star, c^\star)$ as transcript t for the key encapsulation, creates signatures with the correct signing key of the parties, and computes tags with $K_{\mathsf{mac}} \leftarrow \mathsf{KDF}(K^\star_{b'}, \text{"MAC"}||t)$.

For all but the predicted sessions $\pi^\star$ and $\pi^\star_{\mathsf{a}}$, algorithm $\mathcal{B}_2$ simulates any Reveal query straightforwardly by itself. For the session $\pi^\star_{\mathsf{a}}$, if it has already accepted, algorithm $\mathcal{B}_2$ derives the session key by using $K^\star_{b'}$ as the allegedly encapsulated key from the KEM phase. Note that, at this point, we have not yet shown that the associated session $\pi^\star_{\mathsf{a}}$ is partnered with the test session, such that the adversary could Reveal that session without violating freshness.

Once $\mathcal{A}$ queries Test query on $\pi^\star$, algorithm $\mathcal{B}_2$ simulates the Test by providing $K_{\mathsf{test}} \leftarrow \mathsf{KDF}(K^\star_{b'}, \text{"KE"}||(t, r_A, r_B, A, B))$ to $\mathcal{A}$. Note that depending on the nature of $K^\star_{b'}$ the environment for $\mathcal{A}$ then corresponds to either $Game_4(\lambda)$ if $b' = 0$ or $Game_5(\lambda)$ if $b' = 1$.

At some point, $\mathcal{A}$ terminates and outputs some b_{guess}. The reduction $\mathcal{B}_2$ outputs the same b_{guess}. It is easy to see that if $\mathcal{A}$ can win its game with non-negligible probability, so can $\mathcal{B}_2$. Hence,

$$\mathsf{Adv}^{\mathsf{G}_4}_{\mathcal{C}_{\mathsf{SigMA}},\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{G}_5}_{\mathcal{C}_{\mathsf{SigMA}},\mathcal{A}}(\lambda) + 2 \cdot \mathsf{Adv}^{\mathsf{R\text{-}ind\text{-}cpa}}_{\mathcal{K},\mathcal{B}_2}(\lambda).$$

$Game_6(\lambda)$**:** We now replace the session key K and the MAC key K_{mac} by uniformly random values $\widetilde{\mathsf{K}}$ and $\widetilde{K_{\mathsf{mac}}}$ from $\mathcal{D}$ in the test session $\pi^\star$ as well as the associated session $\pi^\star_{\mathsf{a}}$.

We argue that an adversary $\mathcal{A}$ that can efficiently distinguish $Game_5(\lambda)$ from $Game_6(\lambda)$ yields an efficient adversary $\mathcal{B}_3$ against the pseudorandomness of KDF.

The reduction $\mathcal{B}_3$ generates all key exchange parameters and long-term keys itself and initializes $\mathcal{A}$. This means in particular, that $\mathcal{B}_3$ can answer all $\mathsf{NewSession}$, $\mathsf{Corrupt}$ and

Reveal queries (assuming $\mathcal{A}$ does not query Reveal on $\pi^\star$ and $\pi^\star_{\mathsf{a}}$ wlog as this would result in a trivial loss for $\mathcal{A}$.)

Similarly, $\mathcal{B}_3$ can faithfully execute all Send queries. For $\pi^\star$ and $\pi^\star_{\mathsf{a}}$ when the MAC tags are computed, $\mathcal{B}_3$ queries its oracle $\mathcal{O}_{\mathsf{PRF}}$ in the pseudorandomness game on $\text{"MAC"}||t$ where t is the appropriate transcript. The response $K_{b'}$ by the oracle is then either $\mathsf{KDF}(K, \text{"MAC"}||t)$ if $b' = 0$ for some key K that was chosen at the beginning of the pseudorandomness game or $g(\text{"MAC"}||t)$ for some random function g if $b' = 1$. The reduction then uses $K_{b'}$ as MAC key in $\pi^\star$ and $\pi^\star_{\mathsf{a}}$. Note again that signatures can be faithfully simulated by the reduction since it has generated the long-term keys of the participants.

Once $\mathcal{A}$ asks a Test query, $\mathcal{B}_3$ queries $\text{"KE"}||(t, r_A, r_B, A, B)$ to $\mathcal{O}_{\mathsf{PRF}}$ to receive $\mathsf{K}_{b'}$ which is either $\mathsf{KDF}(K, \text{"KE"}||(t, r_A, r_B, A, B))$ if $b' = 0$ or $g(\text{"KE"}||(t, r_A, r_B, A, B))$ if $b' = 1$. The reduction returns $\mathsf{K}_{b'}$ as K_{test} to $\mathcal{A}$.

At some point, $\mathcal{A}$ terminates with output b_{guess} which $\mathcal{B}_3$ also uses as its output.

Observe that $\mathcal{B}_3$ perfectly simulates $\mathit{Game}_5(\lambda)$ if $b' = 0$ and $\mathit{Game}_6(\lambda)$ if $b' = 1$. Thus, if $\mathcal{A}$ can distinguish the two games, $\mathcal{B}_3$ can win the pseudorandomness game against KDF and we have

$$\mathsf{Adv}^{\mathsf{G}_5}_{\mathcal{C}_{\mathsf{SigMA}},\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{G}_6}_{\mathcal{C}_{\mathsf{SigMA}},\mathcal{A}}(\lambda) + 2 \cdot \mathsf{Adv}^{\mathsf{W^cX\text{-}prf\text{-}sec}}_{\mathsf{KDF},\mathcal{B}_3}(\lambda).$$

$\mathit{Game}_7(\lambda)$**:** Next, we abort the game if the associated session $\pi^\star_{\mathsf{a}}$ accepts with a different session identifier than the test session, i.e., if $\pi^\star_{\mathsf{a}}.\mathsf{sid} \neq \pi^\star.\mathsf{sid} \neq \bot$. Since, at this point, all entries in the session identifier are set except for the partner identity, this can only happen if the adversary can cause $\pi^\star_{\mathsf{a}}$ to accept a signature σ_W and MAC τ_W for some identity $W \neq U$. The adversary may indeed sign under an identifier of a previously corrupted party W. However, to succeed $\mathcal{A}$ still needs to forge the corresponding tag τ_W. This tag depends on the MAC key which is derived from the secret value K shared between parties U and V. Similar to Game_3, the probability of an abort happening in this game can be upper-bounded by the success probability of an adversary $\mathcal{B}_4$ against the $\mathsf{U^cV\text{-}euf\text{-}cma}$ security of the MAC scheme $\mathcal{M}$. Hence, we have

$$\mathsf{Adv}^{\mathsf{G}_6}_{\mathcal{C}_{\mathsf{SigMA}},\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{G}_7}_{\mathcal{C}_{\mathsf{SigMA}},\mathcal{A}}(\lambda) + \mathsf{Adv}^{\mathsf{U^cV\text{-}euf\text{-}cma}}_{\mathcal{M},\mathcal{B}_4}(\lambda).$$

To conclude the proof, observe that the adversary expects the challenge value K_{test} to be a uniformly random string for $b_{\mathsf{test}} = 0$ or to be the output of the key derivation function applied to the output of the key encapsulation mechanism $\mathcal{K}$. These two cases cannot be distinguished by $\mathcal{A}$ since *both* keys are drawn independently and uniformly at random from the key space. Hence, the adversary cannot gain any information about the test bit b_{test} and can do no better than to guess. We thus arrive at the final bound

$$\mathsf{Adv}^{\mathsf{G}_7}_{\mathcal{C}_{\mathsf{SigMA}},\mathcal{A}}(\lambda) \leq 0.$$

□

We saw that hybrid key exchange protocols are well-suited to safely transition to post-quantum algorithms, accounting for the case that either of the deployed algorithms may fail. Security guarantees are established for past, ongoing, and future session. In the next chapter, we answer the question which security guarantees can be achieved for standard key exchanges when a cryptographic primitive or hardness assumption fails. As far as intuition goes, if no hybrid techniques have been employed, we will only be able to make statements about sessions that have been established *before* such a breakdown occurred. However, we will see that the model can be extended to also capture the case of hybrid key exchanges as treated in this chapter.

Chapter 6

Breakdown Resilience of Key Exchange Protocols

Modern designs of cryptographic protocols are amenable to security analyses which reduce the security of the protocol to the security of the employed cryptographic primitives. The security guarantees for the protocol are thus ultimately tied to the security of each individual primitive: with only one of the primitives broken, all bets are usually off. However, the actual security guarantees that remain vary with the protocol under consideration.

Key exchange protocols in particular often rely on a significant number of cryptographic primitives and hardness assumptions. For example, the simple AKE construction presented in Section 5.5 relies on the security of the key encapsulation mechanism, unforgeability of both the MAC and signature scheme, and the security of the key derivation function. Further common assumptions for key exchange protocols are the collision resistance of hash functions or cryptographic assumptions such as decisional or computational Diffie-Hellman problems ($\mathsf{DDH}, \mathsf{CDH}$) or the interactive $\mathsf{PRF\text{-}ODH}$ assumption.

Yet, not all of the primitives and hardness assumptions contribute equally to the protocol's overall security at every point in time. While in general it is indeed expected that future sessions are vulnerable once the security of a component in a key exchange is broken, the question is: what can we say about the secrecy of sessions established *prior* to that breakdown? For the special case of hybrid protocols that were introduced in the last chapter, we even expect positive answers for ongoing and future sessions in the presence of certain breakdowns. We will see shortly that the notions of forward secrecy [Gün90, DVOW92, CK01] and post-compromise security [CCG16] answer these questions only partially. A comprehensive notion of security against breakdowns of arbitrary (keyed and unkeyed) primitives and cryptographic hardness assumptions has been lacking so far.

This is despite the fact that examples for failures of actively deployed primitives and hardness assumptions abound: they range from weak ciphers like RC4 [GMPS14, ABP+13] and OCB2 [IIMP19], over poor Diffie–Hellman parameter choices [ABD+15], to advances in breaking (still) widely deployed hash functions like MD5 [dB94, WY05, SLdW07] or SHA-1 [WYY05, Ste13, SKP16, SBK+17]. These vulnerabilities can in particular enable key-exchange-level attacks. For example, Bhargavan and Leurent [BL16] showed that in many key exchanges, the employed hash functions must actually be collision resistant and not—as often argued by practitioners—only second-preimage resistant. Their *SLOTH*[5] attacks on TLS showed that if there exist efficient collision-finding algorithms, then credential-forwarding, impersonation, and downgrade attacks are possible.

[5]Short for: security losses from obsolete and truncated transcript hashes

Our Contributions

There is hence a need for a generic formal tool to identify the security assurances of key exchange protocols in case some arbitrary underlying primitives or hardness assumptions break. In this chapter, we introduce a novel security notion called *breakdown resilience*, that models Bellare–Rogaway-style key exchange security under the breakdown of cryptographic primitives and hardness assumptions. A stronger version of our model captures the impact of cryptographic breakdowns on ongoing and future sessions. While key exchange protocols in general guarantee no security in this setting, for special designs such as hybrid protocols, this notion becomes meaningful. We finally exercise our models by studying the breakdown resilience of practice-inspired protocol designs. In more detail our contributions are as follows:

- In Section 6.2, we propose a formal security model for authenticated key exchange that is able to provide a formal ground for analyses capturing primitive breakdowns. The resulting notions are termed *(strong) breakdown resilience.* We formalize breakdowns via an additional Break oracle provided to the adversary beyond the classical oracles given in a Bellare–Rogaway-style key exchange model.

  We further concretely describe the behavior of the Break oracle for a number of cryptographic primitives and assumptions that are commonly employed in key exchange protocols. In our presentation we opt for the conservative choice of considering strong break capabilities, thus making the adversary more powerful and providing stronger security guarantees of resistant protocols. We note however, that the model can generically handle other choices for consequences of breakdowns.

- In Section 6.3, we then exercise our model on an authenticated variant of the NewHope protocol. NewHope is a post-quantum key exchange protocol proposed by Alkim et al. [ADPS16b] which has also been submitted (in a different variant [AAB$^+$19]) to the NIST Post-Quantum Cryptography standardization process. Using our new formalism, we confirm the intuition that, in particular, signature and MAC breakdowns do not compromise the security of prior completed sessions if the key agreement and derivation itself remains secure.

- In Section 6.4, we consider a hybrid AKE scheme based on the dualPRF KEM combiner proposed in Section 5.3. We leverage the formalism of strong breakdown resilience to directly show the hybrid property of this AKE construction. We argue that the formalism of (strong) breakdown resilience for AKE yields results comparable to those in the two-stage hybrid model presented in Section 5.5 and thus constitutes a viable alternative approach, especially when considering component failures that were not caused by additional quantum computing capabilities.

Personal scientific contribution in this chapter. All the material from this chapter, with the exception of Section 6.4, appeared in the joint work with Marc Fischlin and Felix Günther called *Breakdown Resilience of Key Exchange Protocols: NewHope, TLS 1.3, and Hybrids* [BFG19]. The full version of the paper can be found on ePrint [BFG17]. My main contributions lie in the development of the security model for (strong) breakdown resilience, as well as the proof of breakdown resilience for the NewHope protocol variants. The TLS 1.3 analysis in [BFG17, BFG19] was entirely conducted by Felix Günther and will thus be omitted from this thesis. The result for strong breakdown resilience of the AKE construction based on KEM combiners in Section 6.4 is a novel contribution and has not appeared elsewhere.

6.1 Related Work

The idea of breakdown resilience extends, and is inspired by, conceptual ideas of prior work on the security of both key exchange specifically and cryptographic protocols more broadly. Yet, our notion of breakdown resilience is novel and unmet by any (combination of) previously defined security goals, as we discuss in the following.

Forward secrecy. *Forward secrecy* [Gün90, DVOW92, CK01] as a security property of session keys derived in a key exchange protocol demands that even if an involved party's long-term secret is compromised, any previously derived key remains secure. While this property is closely related to our scenario, breakdown resilience takes a conceptually distinct approach to forward secrecy (and also stronger security models allowing ephemeral key reveal such as [CK01, LLM07]): its focus is on the breakdown of complete primitives or hardness assumptions rather than on the exposure of specific protocol values in selected sessions. Furthermore, the breakdown-resilience scenario also covers breaks of both unkeyed cryptographic building blocks (e.g., breaking collision resistance of hash functions) and complexity-theoretic hardness assumptions.

Post-compromise security. With their notion of *post-compromise security*, Cohn-Gordon, Cremers, and Garratt [CCG16] establish security guarantees for communication *after* participants have been compromised to various degrees. (Strong) breakdown resilience differs from this notion in that it does not consider the compromise of single parties but the global breakdown of cryptographic building blocks on a protocol level. Strong breakdown resilience may be seen as a generalization of the concept of post-compromise security while our standard notion of breakdown resilience is concerned with the security of sessions that were completed *before* a breakdown occurred.

Bitcoin security in the presence of broken primitives. Giechaskiel, Cremers, and Rasmussen [GCR16, GCR18] were the first to systematically explore how broken or weakened hash functions and/or signature schemes affect the security of Bitcoin. While their study was focused on Bitcoin, we present a general framework that can be applied to analyze a whole class of cryptographic protocols, namely AKE protocols, and may very well be transferable in spirit to other kinds of protocols as well.

Downgrade resilience. A breakdown of a primitive or hardness assumption willingly employed by both parties conducting a key exchange is conceptually different from a downgrade of a connection to an insecure cipher suite during the negotiation phase. In the breakdown-resilience setting we are concerned with the security of past sessions after a breakdown has occurred, while the notion of *downgrade resilience*, formally treated by Bhargavan et al. [BBF$^+$16] and Dowling and Stebila [DS15], assures that weak cipher suites will never be successfully negotiated in case matching stronger cipher suites are preferred by both participants.

Hybrid key exchange. The proposed model for hybrid key exchange by Bindel et al. [BBF$^+$19] (see also Chapter 5) is tailored to breakdowns of key encapsulation mechanisms that were caused specifically by quantum adversaries. Our model for (strong) breakdown resilience offers a general, alternative approach which captures the breakdown of multiple, arbitrary primitives or hardness assumptions, irrespective of the cause, thereby gaining more flexibility, while losing the directly visible implications of varying quantum adversaries on the scheme in question. However, we will see later in Section 6.4, that the notion of strong breakdown resilience enables an analysis of hybrid protocols that is comparable to the AKE version in [BBF$^+$19] presented in Chapter 5.5.

6.2 Modeling Breakdown Resilience

For integrating *breakdown resilience* into the Bellare–Rogaway security model for authenticated key exchange, we are interested in the security of completed sessions in the case that cryptographic primitives or hardness assumptions used in the key exchange protocol break.

To capture resilience against breakdowns, we augment our model with a Break query that allows the adversary to break the security of cryptographic primitives or hardness assumptions contained in a dedicated, specified set $\mathcal{F}_{\mathsf{BDR}}$. More precisely, this set has the form $\mathcal{F}_{\mathsf{BDR}} = \{(f_1, \mathsf{sec\text{-}prop}_1), (f_2, \mathsf{sec\text{-}prop}_2), \dots\}$, i.e., $\mathcal{F}_{\mathsf{BDR}}$ contains tuples $(f, \mathsf{sec\text{-}prop})$, determining all primitives/hardness assumptions f for which some security property $\mathsf{sec\text{-}prop}$ may break. As a result of the Break query, the adversary may—depending on the broken security property of the primitive or assumption—be given certain key material or access to additional oracles in the model. To capture that we expect only sessions to remain secure that completed before the breakdown occurred, we introduce a flag $\mathsf{breakdown}$ which is set when Break is called and checked within the (accordingly modified) Send query.

A note on considered sessions. As mentioned before, for classical key exchange designs one cannot expect any security guarantees to remain for ongoing and future sessions, as their security usually depends on the broken primitive. In Section 6.2.3, we will however discuss a stronger variant of breakdown resilience that is able to capture the security of hybrid designs that aim to withstand the breakdown of certain building blocks (cf. Chapter 5).

For now, however, we are interested in the question of whether the expected security level is still achieved in *past* sessions (Scenarios 1 to 3 in Figure 6.1) and thus exclude sessions that are still active at the time of breakdown or start after it (Scenarios 4 and 5 in Figure 6.1).

It turns out that not only the status of the tested session is crucial for the security guarantees, but also that of a potential (unfinished) communication partner, which we previously have referred to as the associated session. A breakdown of a primitive in the middle of the communication may enable the adversary to interfere with the correct partnering of sessions, leading to trivial attacks on the session key in question.

Consider, for example, a test session that has accepted and has output its last message to the intended partner session, say, to authenticate itself (final-message authentication is very common in key exchange protocols). An adversary with breakdown capabilities for the authentication mechanism, e.g., the signature and MAC scheme in the $\mathcal{C}_{\mathsf{SigMA}}$ compiler can modify this last message by, e.g., forging a signature and MAC tag for a different identity. This causes the intended partner to accept with a different session identifier.

Since partnering of sessions was defined via session identifiers, the two sessions are then not partnered. However, the relevant key material has already been established at this point and so the unpartnered session may still derive the same final shared key as our test session. The adversary could hence learn the session key through a Reveal query on the unpartnered session and trivially distinguish the tested key from random.

We thus need to exclude sessions from being tested that accepted prior to the breakdown but have a "semi-completed" partner session that, at the time, already holds all the relevant cryptographic material for the final key derivation (Scenario 3 in Figure 6.1). We use a notion of *contributive identifier* (cid) to identify such almost-partnered sessions. Intuitively, contributive identifiers relate two sessions which exchanged the messages establishing the key material (e.g., values g^x and g^y in a Diffie–Hellman-style protocol), but are not yet partnered (e.g., because the authenticating signatures have not been sent yet). Identical contributive identifiers thus indicate that sessions may eventually derive the same key, despite not being partnered yet.

An alternative to using contributive identifiers would be to demand that only sessions that fully completed before breakdown with an honest partner would be considered valid test sessions

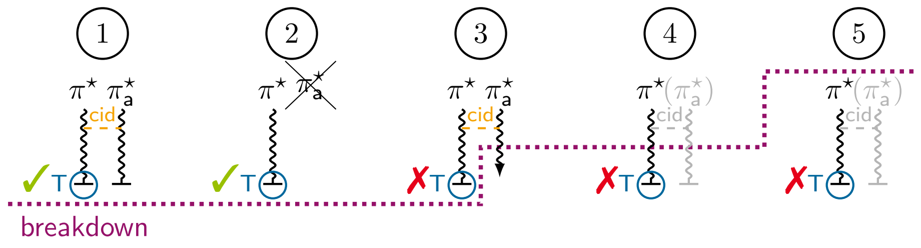

Figure 6.1: Illustration of (non-)permissible Test queries wrt. a breakdown. The dotted purple line indicates the point in time of a breakdown. T denotes a test query on session $\pi^\star$, $\pi^\star_\mathsf{a}$ denotes a (potential, if gray) associated session (semi-)partnered with $\pi^\star$ holding the same contributive identifier (cid). A checkmark ✓ (resp. a cross ✗) indicates whether the test query is admissible or not.

(as in Scenario 1). This, however, would limit the adversary to purely passive attacks in the phase before the breakdown. In contrast, our approach with contributive identifiers is less restrictive, as we still allow the adversary to test completed sessions without an honest partner (Scenario 2), e.g., where the adversary communicated with that party.

6.2.1 Extensions to the Bellare–Rogaway Model

In the following, we specify the formal extensions made to the basic Bellare–Rogaway-style security model from Section 4.2. These changes enable us to formalize a model for breakdown resilience in a generic way. As we will see later, our notion of a Break query is versatile and can capture a wide variety of breakdowns.

Breakdown flag. We introduce a global flag breakdown (initialized to false) in the security game, indicating whether the adversary has issued a Break query. This establishes the timing of the breakdown (cf. the dotted purple line in Figure 6.1).

Contributive identifiers. As briefly touched upon, we augment the model with the concept of contributive identifiers. These identifiers enable us to specify that we do not expect security of sessions that, at time of breakdown, had a “semi-partnered” session that shares the same key material. We thus demand that the tested session accepted prior to the breakdown and does not share a contributive identifier with another session that was still running at the time of breakdown. To capture this formally, we add the following variables to those associated with each session $\pi^k_{U,V}$:

- $\mathsf{cid} \in \{0,1\}^\star \cup \{\bot\}$ indicates the contributive identifier. The default value is $\bot$.
- $\mathsf{st}^{\mathsf{bd}}_{\mathsf{exec}} \in \{\mathsf{running}, \mathsf{accepted}, \mathsf{rejected}, \bot\}$ denotes the state of execution at the time of breakdown (i.e., when the Break query was issued the first time). The default value prior to a breakdown is $\bot$.

Remark 6.1. We opt to use the formalization of contributive identifiers introduced by Dowling et al. [DFGS15a] in their analysis of TLS 1.3 candidate handshakes in the multi-stage key exchange setting. There, contributive identifiers constitute “unfinished” session identifiers, where not all values have been set yet. The concept of contributive identifiers is related to the notion of *origin-sessions* for partnering based on matching conversations introduced by Cremers and Feltz [CF12] and furthermore the notion of (peer-)exchange variables used by Bhargavan et al. [BFK$^+$14]. Since we define partnering based on session identifiers, the approach by [DFGS15a] is however the most natural fit for our setting.

To avoid trivial choices and to relate the contributive identifiers to session identifiers we add two requirements for Match security:

- First, as in [DFGS15a], same session identifiers must imply same contributive identifiers, capturing the intuition that partnered sessions should in particular be contributively partnered.
- Second, since we restrict the Test query based on common contributive identifiers, we demand that at most two sessions share the same contributive identifier. This prevents that Test queries are excluded by trivial choices of colliding contributive identifiers.

Break query. We add a Break query that complements the adversarial queries described in Section 4.2.1 and allows the adversary to schedule the timing of breakdowns. The query will set the flag $\mathsf{breakdown}$ to true, record the current execution state of sessions, and provide the adversary with the capability to break the security of any $(f, \mathsf{sec\text{-}prop}) \in \mathcal{F}_{\mathsf{BDR}}$, where $\mathcal{F}_{\mathsf{BDR}}$ is a fixed parameter of the security game.

$\mathsf{Break}()$**:** Causes for all $(f, \mathsf{sec\text{-}prop}) \in \mathcal{F}_{\mathsf{BDR}}$ the breakdown of the security property $\mathsf{sec\text{-}prop}$ of the cryptographic primitive or hardness assumption f.

If $\mathsf{breakdown} = \mathsf{false}$, for all sessions π record the current state of execution in the variable $\mathsf{st}_{\mathsf{exec}}^{\mathsf{bd}}$, i.e., $\pi.\mathsf{st}_{\mathsf{exec}}^{\mathsf{bd}} \leftarrow \pi.\mathsf{st}_{\mathsf{exec}}$. Set $\mathsf{breakdown} \leftarrow \mathsf{true}$. Depending on the entries in the set $\mathcal{F}_{\mathsf{BDR}}$, provide the adversary with the specified responses and/or oracle accesses (cf., e.g. Table 6.1). The Break oracle may be queried repeatedly, which enables the adversary to obtain an updated response in order to, e.g., receive further key material used in an encryption scheme since the last call of Break.

Which capability the adversary $\mathcal{A}$ is given when breaking the security $\mathsf{sec\text{-}prop}$ of a primitive or assumption f depends on the latter's type and may, e.g., be exposing all key material used within f to $\mathcal{A}$ or granting access to additional oracles. We discuss options for common primitives and the corresponding behavior of Break later in Section 6.2.4.

Modified Send query. Once the $\mathsf{breakdown}$ flag is set to true, ongoing sessions and sessions that are initiated after the breakdown must be considered revealed as we expect their keys to be afflicted by the breakdown. To enforce this, we replace the Send query from Section 4.2.1 by the following slightly modified version that sets the session key state to $\mathsf{revealed}$ if $\mathsf{breakdown} = \mathsf{true}$; the change is underlined in the following description.

$\mathsf{Send}_{\mathsf{BDR}}(\pi_{U,V}^{k}, m)$**:** Causes the message m to be sent to the session $\pi_{U,V}^{k}$. If there exists no session $\pi_{U,V}^{k}$, the query outputs $\perp$. Else the response of the session owner U upon receipt of message m is returned, and the state of execution $\mathsf{st}_{\mathsf{exec}}$ is updated. If $\mathsf{st}_{\mathsf{exec}}$ changes to $\mathsf{accepted}$ with an intended communication partner V that was previously corrupted $\underline{\text{or if } \mathsf{breakdown} = \mathsf{true}}$, then set $\mathsf{st}_{\mathsf{key}} \leftarrow \mathsf{revealed}$.

6.2.2 Breakdown-resilient Bellare–Rogaway Security Definitions

We are now ready to define the security notion of *breakdown resilience* (BDR) for an authenticated key exchange protocol. Extending the Bellare–Rogaway-style model from Section 4.2, we again divide the security properties into $\mathsf{BDR\text{-}Match}$ security and BDR key secrecy. Both security notions differ from the original Bellare–Rogaway-like notions by including the set of primitive breakdowns $\mathcal{F}_{\mathsf{BDR}}$ under consideration and the novel Break oracle as well as replacing the original Send oracle by the modified $\mathsf{Send}_{\mathsf{BDR}}$ version. The $\mathsf{BDR\text{-}Match}$ definition furthermore reflects that contributive identifiers must coincide in matching sessions but be distinct otherwise; BDR key secrecy leverages the introduced contributive identifiers to exclude test sessions with semi-completed partners at the time of breakdown.

Definition 6.2 (BDR-Match security)**.** *Let* KE *be a key exchange protocol, and let* $\mathcal{F}_{\mathsf{BDR}}$ *be the set of cryptographic primitives and hardness assumptions the adversary can break in the model.*

Let $\mathcal{A}$ *be a PPT adversary interacting with* KE *via the queries* $\mathsf{NewSession}$, $\mathsf{Send}_{\mathsf{BDR}}$, Reveal, $\mathsf{Corrupt}$, Test, *and* Break *in the following game* $\mathsf{G}^{\mathsf{BDR\text{-}Match}(\mathcal{F}_{\mathsf{BDR}})}_{\mathsf{KE},\mathcal{A}}(\lambda)$*:*

Setup. *The challenger generates long-term public/secret-key pairs for each participant* $U \in \mathcal{U}$*.*

Query. *The adversary* $\mathcal{A}$ *receives the generated public keys and has access to the queries* $\mathsf{NewSession}$, $\mathsf{Send}_{\mathsf{BDR}}$, Reveal, $\mathsf{Corrupt}$, Test, *and* Break*.*

Stop. *At some point, the adversary stops with no output.*

We say that $\mathcal{A}$ *wins the game* $\mathsf{G}^{\mathsf{BDR\text{-}Match}(\mathcal{F}_{\mathsf{BDR}})}_{\mathsf{KE},\mathcal{A}}(\lambda)$ *if at least one of the following conditions holds:*

1. Different session keys in partnered sessions:

   There exist two distinct sessions π *and* π' *with* $\pi.\mathsf{sid} = \pi'.\mathsf{sid} \neq \bot$, *and* $\pi.\mathsf{st}_{\mathsf{exec}}, \pi'.\mathsf{st}_{\mathsf{exec}} \neq \mathsf{rejected}$, *but* $\pi.\mathsf{K} \neq \pi'.\mathsf{K}$.

2. Different or unset contributive identifiers in partnered sessions:

   There exist two distinct sessions π *and* π' *such that* $\pi.\mathsf{sid} = \pi'.\mathsf{sid} \neq \bot$, *but* $\pi.\mathsf{cid} \neq \pi'.\mathsf{cid}$ *or* $\pi.\mathsf{cid} = \pi'.\mathsf{cid} = \bot$.

3. Different intended partner in partnered sessions:

   There exist two sessions $\pi := \pi^{k}_{U,V}$ *and* $\pi' := \pi^{k'}_{V',U'}$ *such that* $\pi.\mathsf{sid} = \pi'.\mathsf{sid} \neq \bot$, $\pi.\mathsf{role} = \mathsf{initiator}$, *and* $\pi'.\mathsf{role} = \mathsf{responder}$, *but* $U \neq U'$ *or* $V \neq V'$.

4. More than two sessions share the same session identifier or contributive identifier:

   There exist at least three sessions π, π', *and* π'' *such that* π, π', π'' *are pairwise distinct, but* $\pi.\mathsf{sid} = \pi'.\mathsf{sid}' = \pi''.\mathsf{sid} \neq \bot$ *or* $\pi.\mathsf{cid} = \pi'.\mathsf{cid}' = \pi''.\mathsf{cid} \neq \bot$.

We say KE *is* $\mathsf{BDR\text{-}Match}$ secure *for* $\mathcal{F}_{\mathsf{BDR}}$ *if for all PPT adversaries* $\mathcal{A}$ *the advantage function*

$$\mathsf{Adv}^{\mathsf{BDR\text{-}Match}(\mathcal{F}_{\mathsf{BDR}})}_{\mathsf{KE},\mathcal{A}}(\lambda) := \Pr\left[\mathsf{G}^{\mathsf{BDR\text{-}Match}(\mathcal{F}_{\mathsf{BDR}})}_{\mathsf{KE},\mathcal{A}}(\lambda) = 1\right]$$

is negligible in the security parameter λ*.*

Definition 6.3 (BDR key secrecy)**.** *Let* KE *be a key exchange protocol with key distribution* $\mathcal{D}$*, and let* $\mathcal{F}_{\mathsf{BDR}}$ *be the set of cryptographic primitives and hardness assumptions the adversary can break in the model.*

Let $\mathcal{A}$ *be a PPT adversary interacting with* KE *via the queries* $\mathsf{NewSession}$, $\mathsf{Send}_{\mathsf{BDR}}$, Reveal, $\mathsf{Corrupt}$, Break, *and* Test *in the following game* $\mathsf{G}^{\mathsf{BDR}(\mathcal{F}_{\mathsf{BDR}}),\mathcal{D}}_{\mathsf{KE},\mathcal{A}}(\lambda)$*:*

Setup. *The challenger generates long-term public/secret-key pairs for each participant* $U \in \mathcal{U}$*, chooses the test bit* $b_{\mathsf{test}} \xleftarrow{\$} \{0,1\}$ *at random and sets* $\mathsf{lost} \leftarrow \mathsf{false}$*.*

Query. *The adversary* $\mathcal{A}$ *receives the generated public keys and has access to the queries* $\mathsf{NewSession}$, $\mathsf{Send}_{\mathsf{BDR}}$, Reveal, $\mathsf{Corrupt}$, Test, *and* Break*.*

Guess. *At some point, $\mathcal{A}$ stops and outputs a guess b_{guess}.*

Finalize. *The challenger sets the lost flag to $\mathsf{lost} \leftarrow \mathsf{true}$ if at least one of the following conditions hold:*

1. Adversary has tested and revealed the key in a single session or in two partnered sessions:
There exist two (not necessarily distinct) sessions π, π' such that $\pi.\mathsf{sid} = \pi'.\mathsf{sid}$, $\pi.\mathsf{st}_{\mathsf{key}} = \mathsf{revealed}$, and $\pi'.\mathsf{tested} = \mathsf{true}$.

2. Adversary has tested a session whose contributive partner session was running at the time of breakdown:
There exist two distinct sessions π, π' such that $\pi.\mathsf{tested} = \mathsf{true}$, $\pi.\mathsf{cid} = \pi'.\mathsf{cid}$, and $\pi'.\mathsf{st}^{\mathsf{bd}}_{\mathsf{exec}} = \mathsf{running}$.

The adversary $\mathcal{A}$ wins the game $\mathsf{G}^{\mathsf{BDR}(\mathcal{F}_{\mathsf{BDR}}),\mathcal{D}}_{\mathsf{KE},\mathcal{A}}(\lambda)$ if $b_{\mathsf{guess}} = b_{\mathsf{test}}$ and $\mathsf{lost} = \mathsf{false}$.

We say that KE provides BDR key secrecy *for $\mathcal{F}_{\mathsf{BDR}}$ with/without forward secrecy if for all PPT adversaries $\mathcal{A}$ the advantage function*

$$\mathsf{Adv}^{\mathsf{BDR}(\mathcal{F}_{\mathsf{BDR}}),\mathcal{D}}_{\mathsf{KE},\mathcal{A}}(\lambda) := \Pr\left[\mathsf{G}^{\mathsf{BDR}(\mathcal{F}_{\mathsf{BDR}}),\mathcal{D}}_{\mathsf{KE},\mathcal{A}}(\lambda) = 1\right] - \frac{1}{2}$$

is negligible in the security parameter λ.

Definition 6.4 (Breakdown resilience)**.** *We say a key exchange protocol KE is* breakdown resilient for $\mathcal{F}_{\mathsf{BDR}}$ *(with/without forward secrecy) if KE provides BDR-Match security and BDR key secrecy for $\mathcal{F}_{\mathsf{BDR}}$ (with/without forward secrecy), according to Definitions 6.2 and 6.3.*

Fundamental Properties

The model for breakdown resilience is a proper extension of the Bellare–Rogaway model for AKEs given in Section 4.2, thus breakdown resilience implies standard BR security:

Proposition 6.5. *If a key exchange protocol KE achieves breakdown resilience for some $\mathcal{F}_{\mathsf{BDR}}$ (incl. $\mathcal{F}_{\mathsf{BDR}} = \emptyset$) with/without forward secrecy according to Definition 6.4, then KE is also BR-secure with/without forward secrecy according to Definition 4.3.*

Proof. If the Break query is not asked by the adversary, the flag $\mathsf{breakdown}$ and the modification to the original Send query are essentially not touched and may thus be omitted. Likewise, the second **Finalize** condition in Definition 6.3 becomes void as $\mathsf{st}^{\mathsf{bd}}_{\mathsf{exec}} = \bot$ for all sessions. But then the models and in particular the Match security definition (modulo contributive identifiers) and the key secrecy definition for breakdown resilience and original BR security coincide. □

As mentioned earlier, it is often convenient to consider breakdown resilience for a stronger cryptographic hardness assumption than the one employed in a (non-breakdown-resilient) security proof, with the discrete logarithm problem DLP vs. decisional and computational DH DDH and CDH being a specific example. We hence make this relation more precise via the following proposition, which may prove useful when considering the breakdown of a cryptographic hardness assumption X whose breakdown implies the ability to break some other assumption Y. In our setting this means that one can provide the reply of the Break oracle for Y by the answer for X. We say that solving X implies solving Y.

Proposition 6.6. *Let Π be some protocol and let X and Y be some cryptographic hardness assumptions with $X \in \mathcal{F}_{\mathsf{BDR}}$, but $Y \notin \mathcal{F}_{\mathsf{BDR}}$. Assume that solving X implies solving Y. Then, if Π is breakdown resilient for $\mathcal{F}_{\mathsf{BDR}}$, then Π is also breakdown resilient for $\mathcal{F}'_{\mathsf{BDR}} = \mathcal{F}_{\mathsf{BDR}} \cup \{Y\}$.*

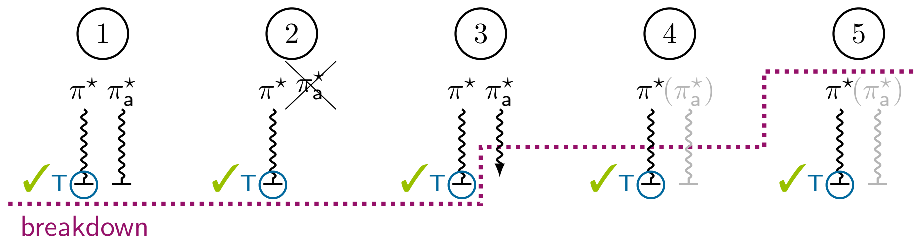

Figure 6.2: Illustration of permissible Test queries for strong breakdown resilience. Scenarios 1 and 2 are as in Figure 6.1; Scenarios 3, 4, and 5 are now permissible.

Proof. We can directly simulate the Break query for $\mathcal{F}'_{\mathsf{BDR}}$ via a Break query for $\mathcal{F}_{\mathsf{BDR}}$, since the Break response for X allows to provide the response for Y. □

6.2.3 Strong Breakdown Resilience

In Chapter 5 we discussed hybrid protocols that have built-in redundancy to withstand the breakdowns of single cryptographic components. Security of these protocols is then achievable not only for past sessions, as is the case for breakdown resilience as defined now, but also for ongoing and future sessions.

We observe, that the breakdown-resilience model can be readily adjusted to also capture these scenarios, when restricting $\mathcal{F}_{\mathsf{BDR}}$ to those cryptographic components for which the hybrid protocol ensures redundancy. We term the resulting notion *strong* breakdown resilience.

Figure 6.2 depicts the now admissible Test scenarios for *strong* breakdown resilience. In order to extend the basic model of breakdown resilience to encompass security for future and ongoing sessions, a couple of minor changes are necessary:

Send query. The previously introduced modified $\mathsf{Send}_{\mathsf{BDR}}$ ensured that ongoing and future sessions at the time of breakdown were set to revealed and could thus not be tested by the adversary. This is no longer wished for in the strong breakdown-resilience scenario, so here the original, unmodified Send query of the Bellare–Rogaway-model (cf. Section 4.2.1) is employed.

Contributive identifiers and state of execution at breakdown. Similarly, contributive identifiers (cid) that were needed to identify cases that are not testable (cf. Figure 6.1) become superfluous and any mention of them in the security definitions of BDR-Match security and BDR key secrecy (Definitions 6.2 and 6.3) can be omitted. Finally, we no longer need to record the execution state at breakdown $\mathsf{st}^{\mathsf{bd}}_{\mathsf{exec}}$.

6.2.4 Defining the Break Oracle

We next give an exemplary specification of the behavior of the Break oracle and capabilities the adversary is provided with for a number of common cryptographic primitives and hardness assumptions. Table 6.1 covers a wide range of standard primitives and assumptions underlying the security of most key exchange protocols (and in particular the NewHope protocols [ADPS16b, AAB+19] we analyze in Section 6.3).

Keyed primitives and unkeyed primitives with secret input. For keyed primitives (both public-key and secret-key ones), the basic idea for the Break oracle is to hand the adversary all secret keys which have been created in protocol executions so far. Since the adversary in our model can call the Break oracle multiple times it may also access subsequently generated keys.

Primitive or Cryptographic Hardness Assumption (f)	**Algorithms**	**Security Assumption** (sec-prop)	Break **Response**
Asymmetric or Symmetric Encryption Scheme $\mathcal{E}$	$\mathcal{E} = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$	IND-CCA (indistinguishability under adaptive chosen ciphertext attack)	return all previous outputs (pk, sk) or sk for which $(pk, sk) \leftarrow \mathsf{KGen}$ or $sk \leftarrow \mathsf{KGen}$
Key Encapsulation Mechanism $\mathcal{K}$	$\mathcal{K} = (\mathsf{KGen}, \mathsf{Encaps}, \mathsf{Decaps})$	IND-CCA (indistinguishability under adaptive chosen ciphertext attack)	return all previous outputs (pk, sk) for which $(pk, sk) \leftarrow \mathsf{KGen}$
Signature Scheme $\mathcal{S}$	$\mathcal{S} = (\mathsf{KGen}, \mathcal{S}, \mathsf{Vfy})$	EUF-CMA (existential unforgeability under chosen message attack)	return all previous pairs (pk, sk) for which $(pk, sk) \leftarrow \mathsf{KGen}$
MAC Scheme $\mathcal{M}$	$\mathcal{M} = (\mathsf{KGen}, \mathsf{MAC}, \mathsf{Vfy})$	EUF-CMA (existential unforgeability under chosen message attack)	return all previous values sk for which $sk \leftarrow \mathsf{KGen}$
Hash Function Family $\mathcal{H}$	$\mathcal{H} = (\mathsf{KGen}, \mathsf{H})$	STD-Coll-Res (standard-model collision resistance)	programmable access to H: After breakdown, $\mathcal{A}$ sets output of H queries on previously unseen values
	$\mathcal{H} = (\mathsf{KGen}, \mathsf{RO})$	RO-Coll-Res (random-oracle collision resistance)	programmable access to RO: After breakdown, $\mathcal{A}$ sets output of RO queries on previously unseen values
	$\mathcal{H} = (\mathsf{KGen}, \mathsf{H})$	SecPre (second preimage resistance)	programmable access to H: After breakdown, $\mathcal{A}$ can set output of H query on previously unseen value x' to y, where $y = \mathsf{H}(x)$ for some previously seen value x
	$\mathcal{H} = (\mathsf{KGen}, \mathsf{RO})$	RO-Rand (random-oracle randomness)	return all previous s for which $s \leftarrow \mathsf{RO}(\cdot)$
	$\mathcal{H} = (\mathsf{KGen}, \mathsf{RO})$	RO-One-Way (random-oracle one-wayness)	return all previous pairs (x, s) for which $s \leftarrow \mathsf{RO}(x)$
Pseudorandom Function Family PRF	$\mathsf{PRF} = (\mathsf{KGen}, \mathsf{F})$	prf-sec (output pseudorandomness)	return all previous values k for which $k \leftarrow \mathsf{KGen}$
	$\mathsf{PRF} = (\mathsf{KGen}, \mathsf{RO})$	RO-Rand (random-oracle randomness)	return all previous s for which $s \leftarrow \mathsf{RO}(\cdot)$
	$\mathsf{PRF} = (\mathsf{KGen}, \mathsf{RO})$	RO-One-Way (random-oracle one-wayness)	return all previous pairs (x, s) s.t. $s \leftarrow \mathsf{RO}(x)$
Discrete Log Assumption	$\mathsf{GroupExp}(h, x) = h^x$ in multiplicative cyclic group $\mathbb{G} = \langle g \rangle$, $h \in \mathbb{G}$	DLP discrete logarithm problem	return all previous pairs (x, h^x) for which $h^x \leftarrow \mathsf{GroupExp}(h, x)$
Factoring Assumption	$\mathsf{GenModulus}(1^n) = (N, p, q)$ s.t. $N = p \cdot q$ where p, q are n-bit primes	Prime-Fact prime factorization	return all previous tuples (N, p, q) for which $(N, p, q) \leftarrow \mathsf{GenModulus}(\cdot)$

Table 6.1: Potential Break oracle specifications.

In order for Break to provide the necessary information, we make the key generation algorithm of a primitive explicit and have all honest parties invoke it when generating key material for this primitive. For example, any keys used for an encryption scheme $\mathcal{E} = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ in honest sessions will be generated via the key generation algorithm KGen, with the challenger in the security game storing the output. This approach enables the challenger to return an exhaustive list of all secret keys of a primitive up to the point of breakdown when a Break query is asked.

In key exchange protocols it is common that keys for keyed primitives are not derived via an explicit key generation algorithm but, e.g., sampled at random or generated through a key derivation function. We implicitly treat such key derivations as a trivial (identity function) key generation algorithm in our model, hence recording also such keys for exposure through a Break query. This means for example that the function is no longer unpredictable or pseudorandom. To capture this formally, we again assume that the challenger keeps a list of all function outputs generated by honest sessions, in order to provide the according list to the adversary in case of a Break query.

Public primitives. For public primitives like a hash function H and security properties like collision resistance we have to capture the increased capabilities of the adversary $\mathcal{A}$ after the breakdown differently. Here, regardless of whether H is modeled as a random oracle RO or considered in the standard model, the adversary $\mathcal{A}$ must be able to generate collisions after the break. To this end, we allow $\mathcal{A}$ to program H globally on previously unseen input values after the breakdown occurred. In particular, after the break $\mathcal{A}$ answers all queries by honest sessions to the hash function H itself (but consistently with previous replies). If, on the other hand, we aim at modeling breakdown of the one-wayness of a random oracle, we instead hand the adversary all input-output pairs which honest parties have evaluated.

Cryptographic hardness assumptions. Finally, we can also treat the breakdown of interesting cryptographic assumptions for key exchange via the Break oracle. We illustrate this here by the discrete logarithm problem (DLP) and the factoring problem ($\mathsf{Prime\text{-}Fact}$), which we treat similarly to public-key primitives. For the example of DLP, we mandate that honest sessions invoke a given algorithm $\mathsf{GroupExp}$ for group exponentiations, which then allows the challenger in the security game to provide the adversary with all secret exponents employed in honest sessions on a Break query. Note that for related cryptographic assumptions, the breakdown of one assumption can imply the breakdown of the other. For example, we can restrict our attention to DLP for Diffie–Hellman-style protocols, as (resilience against) a breakdown of DLP in particular implies (resilience against) the breakdown of other commonly used assumption like DDH and CDH.

Remark 6.7. We stress that Table 6.1 only gives (conservative) recommendations on how the Break oracle can be implemented for the most common primitives and hardness assumptions in the area of key exchange. Depending on the security properties required from the primitives in a specific key exchange setting, one may wish to specify different responses for the Break query. This is easily possible in our model as the Break query itself is generic.

6.3 Breakdown Resilience of Auth-NewHope

In this section, we show a first application of our new security model. For our analysis, we consider a (classically) authenticated variant of the post-quantum secure key exchange scheme NewHope. NewHope is a lattice-based key exchange protocol that was originally introduced in 2016 by Alkim et al. [ADPS16b] as an improvement over previous work by Bos et al. [BCNS15] with respect to efficiency and parameter sizes. The protocol soon gained widespread attention, not least because of its deployment as the post-quantum component in the hybrid key exchange experiment CECPQ1 in Google Chrome Canary [Bra16].

We will see that, unlike in the Diffie-Hellman-setting, Alice and Bob only agree on *approximately* the same value after they have exchanged their public keys, due to the nature of ring elements. In order for them to agree on a common value that will be the input to the final key derivation, they have to execute a so-called *error-reconciliation mechanism*, that was first introduced by Ding, Xie, and Lin [DXL12] and is based on the idea of *fuzzy extractors* [DRS04], to establish shared keys from noisy data. This is achieved by Bob providing additional *reconciliation information* to Alice alongside his public key, that Alice can use to derive the same secret as Bob with high probability.

Later that same year, a simpler, encryption-based version, NewHope-Simple [ADPS16a], was introduced. Contrary to the first design, this variant uses encryption to establish a shared cryptographic key between the communicating parties, thereby forgoing the need for reconciliation. NewHope-Simple then served as the basis for the key encapsulation mechanisms in [AAB+19] that were submitted to the NIST Post-Quantum Cryptography standardization process [Nat15] in 2017 and that have made it into the second round of the process. In the following, for the purpose of a clear distinction we will refer to the different schemes as NewHope-Usenix, NewHope-Simple, and NewHope-Nist, respectively.

For now, we focus on the analysis of the reconciliation-based NewHope-Usenix. At the end of this section, we will investigate the breakdown resilience of generic KEM-based constructions, thereby also capturing the breakdown resilience of the variants NewHope-Simple and NewHope-Nist that were formulated as KEMs.

6.3.1 Protocol Description

In its originally proposed form, NewHope-Usenix provides unauthenticated key agreement. For our analysis, we consider an authenticated version of NewHope-Usenix, in the following referred to as Auth-NewHope. The authenticated version of the protocol is depicted in Figure 6.3, where the original protocol (above the double line) is followed by SigMA-authentication [Kra03]. In the following description, we focus on the unauthenticated NewHope-Usenix protocol as SigMA-authentication has already been discussed in Section 5.5.

Alice first generates the public parameter a by choosing a random seed seed and setting $a \leftarrow \mathsf{XOF}(\mathsf{seed})$. She then generates her public key $b \leftarrow as + e$, where $s, e \xleftarrow{\$} \psi_{16}^n$ and sends seed, b to Bob.

Bob computes the parameter a from seed and generates his public key in the same manner as Alice, i.e., $b' \leftarrow as' + e'$ with $s', e' \xleftarrow{\$} \psi_{16}^n$. He then computes the approximate shared secret v as $v \leftarrow bs' + e''$, where $e'' \xleftarrow{\$} \psi_{16}^n$. As Alice will not be able to derive the exact same value, Bob additionally computes reconciliation information r as $r \xleftarrow{\$} \mathsf{HelpRec}(v)$. This value will allow Alice and Bob to agree on an exact shared value with high probability. Finally, he sends his public key b' and the reconciliation information r to Alice.

Alice computes the approximate shared secret $v' \leftarrow b's = ass' + e's$. Note that this is indeed not the same value v that Bob holds, since $v = ass' + es' + e''$. The reconciled common shared secret w is then given as $\mathsf{Rec}(v, r)$ on Bob's side and $\mathsf{Rec}(v', r)$ on Alice's side. This can then be used to derive the final session key K (at the end of the protocol run) as well as a MAC key K_{mac}

Next, Alice and Bob authenticate each other via SigMA-authentication (cf. compiler description in Section 5.5 for details).

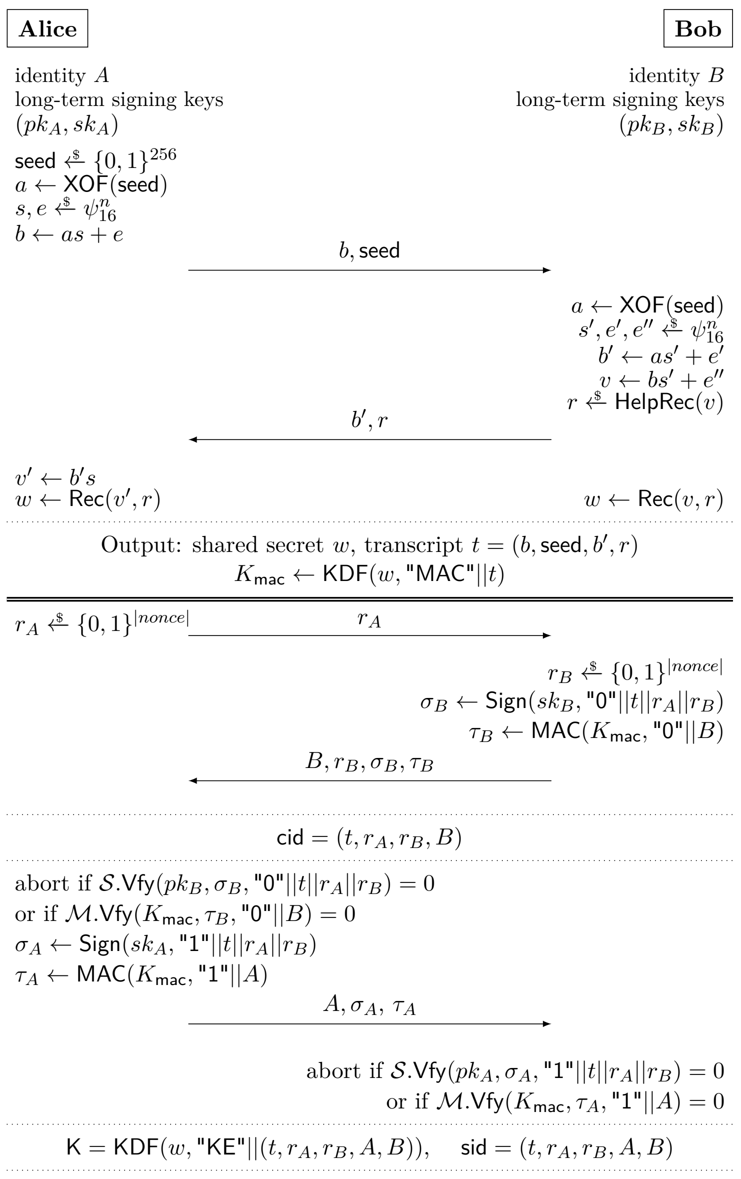
Alice
Bob
identity A
long-term signing keys
(pk_A, sk_A)
identity B
long-term signing keys
(pk_B, sk_B)
$\mathsf{seed} \xleftarrow{\$} \{0,1\}^{256}$
$a \leftarrow \mathsf{XOF}(\mathsf{seed})$
$s, e \xleftarrow{\$} \psi_{16}^n$
$b \leftarrow as + e$
b, seed
$a \leftarrow \mathsf{XOF}(\mathsf{seed})$
$s', e', e'' \xleftarrow{\$} \psi_{16}^n$
$b' \leftarrow as' + e'$
$v \leftarrow bs' + e''$
$r \xleftarrow{\$} \mathsf{HelpRec}(v)$
b', r
$v' \leftarrow b's$
$w \leftarrow \mathsf{Rec}(v', r)$
$w \leftarrow \mathsf{Rec}(v, r)$
Output: shared secret w, transcript $t = (b, \mathsf{seed}, b', r)$
$K_{\mathsf{mac}} \leftarrow \mathsf{KDF}(w, \text{"MAC"}||t)$
$r_A \xleftarrow{\$} \{0,1\}^{|nonce|}$
r_A
$r_B \xleftarrow{\$} \{0,1\}^{|nonce|}$
$\sigma_B \leftarrow \mathsf{Sign}(sk_B, \text{"0"}||t||r_A||r_B)$
$\tau_B \leftarrow \mathsf{MAC}(K_{\mathsf{mac}}, \text{"0"}||B)$
B, r_B, σ_B, τ_B
$\mathsf{cid} = (t, r_A, r_B, B)$
abort if $\mathcal{S}.\mathsf{Vfy}(pk_B, \sigma_B, \text{"0"}||t||r_A||r_B) = 0$
or if $\mathcal{M}.\mathsf{Vfy}(K_{\mathsf{mac}}, \tau_B, \text{"0"}||B) = 0$
$\sigma_A \leftarrow \mathsf{Sign}(sk_A, \text{"1"}||t||r_A||r_B)$
$\tau_A \leftarrow \mathsf{MAC}(K_{\mathsf{mac}}, \text{"1"}||A)$
A, σ_A, τ_A
abort if $\mathcal{S}.\mathsf{Vfy}(pk_A, \sigma_A, \text{"1"}||t||r_A||r_B) = 0$
or if $\mathcal{M}.\mathsf{Vfy}(K_{\mathsf{mac}}, \tau_A, \text{"1"}||A) = 0$
$\mathsf{K} = \mathsf{KDF}(w, \text{"KE"}||(t, r_A, r_B, A, B)), \quad \mathsf{sid} = (t, r_A, r_B, A, B)$

Figure 6.3: The AUTH-NEWHOPE protocol with original unauthenticated key exchange NEWHOPE-USENIX above the double line and SigMA-style authentication below.

Cryptographic assumptions. AUTH-NEWHOPE relies on the following cryptographic primitives and hardness assumptions: random-oracle randomness of the extendable-output function XOF, pseudorandomness of the key derivation function KDF, and existential unforgeability of the signature scheme $\mathcal{S}$ and MAC scheme $\mathcal{M}$.

The post-quantum security of all NEWHOPE schemes is further based on the (decisional) Ring-Learning-with-Errors ((D)RLWE), which informally states that $as + e$ for public a, secret s, and small error e, is indistinguishable from a random ring element.

More formally, let $\mathcal{R} = \mathbb{Z}[X]/X^n + 1$ for $n = 2^m$ with $m \geq 0$ be the ring of integers of the $2n$-th cyclotomic number field. For q an integer, define $\mathcal{R}_q$ to be the ring $\mathcal{R}/q\mathcal{R} \cong \mathbb{Z}_q[X]/(X^n + 1)$. Then the decisional RLWE problem is defined as follows:

Definition 6.8 (DRLWE problem)**.** *Let $\mathcal{R}_q$ be the ring $\mathcal{R}/q\mathcal{R} \cong \mathbb{Z}_q[X]/(X^n + 1)$, χ some error distribution, and $\mathcal{A}$ be a PPT algorithm. Let $\mathsf{G}^{\mathsf{drlwe}}_{\mathcal{R}_q,\chi,\mathcal{A}}$ be the game defined in Figure 6.4. We say that the* decisional Ring-LWE problem (DRLWE) is hard *for $\mathcal{R}_q, \chi$, if for any PPT adversary $\mathcal{A}$*

the advantage function defined as

$$\mathsf{Adv}^{\mathsf{drlwe}}_{\mathcal{R}_q,\chi,\mathcal{A}}(\lambda) := \left|\Pr\left[\mathsf{G}^{\mathsf{drlwe}}_{\mathcal{R}_q,\chi,\mathcal{A}}(\lambda) = 1\right] - \frac{1}{2}\right|$$

is negligible in the security parameter λ.

Hardness of DRLWE implies hardness of the decision Diffie–Hellman-like problem DDHℓ, which is particularly convenient for analyzing DH-like key exchange protocols based on DRLWE such as NewHope:

Definition 6.9 (DDHℓ problem)**.** *Let $\mathcal{R}_q$ be the ring $\mathcal{R}/q\mathcal{R} \cong \mathbb{Z}_q[X]/(X^n + 1)$, χ some error distribution, and $\mathcal{A}$ be a PPT algorithm. Let $\mathsf{G}^{\mathsf{ddh}\ell}_{\mathcal{R}_q,\chi,\mathcal{A}}$ be the game defined in Figure 6.4. We say that the* decisional Diffie–Hellman-like problem (DDHℓ) is hard *for $\mathcal{R}_q$ and χ, if for any PPT adversary $\mathcal{A}$ the advantage function defined as*

$$\mathsf{Adv}^{\mathsf{ddh}\ell}_{\mathcal{R}_q,\chi,\mathcal{A}}(\lambda) := \left|\Pr\left[\mathsf{G}^{\mathsf{ddh}\ell}_{\mathcal{R}_q,\chi,\mathcal{A}}(\lambda) = 1\right] - \frac{1}{2}\right|$$

is negligible in the security parameter λ.

```
$\mathsf{G}^{\mathsf{drlwe}}_{\mathcal{R}_q,\chi,\mathcal{A}}(\lambda)$:
$a \xleftarrow{\$} \mathcal{R}_q$
$s, e \xleftarrow{\$} \chi$
$b_0^\star \leftarrow as + e$
$b_1^\star \xleftarrow{\$} \mathcal{R}_q$
$b_{\mathsf{test}} \xleftarrow{\$} \{0,1\}$
$b_{\mathsf{guess}} \xleftarrow{\$} \mathcal{A}(a, b^\star_{b_{\mathsf{test}}})$
return $[\![b_{\mathsf{test}} = b_{\mathsf{guess}}]\!]$
```

```
$\mathsf{G}^{\mathsf{ddh}\ell}_{\mathcal{R}_q,\chi,\mathcal{A}}(\lambda)$:
$a \xleftarrow{\$} \mathcal{R}_q$
$s, s', e, e'' \xleftarrow{\$} \chi$
$b \leftarrow as + e$
$b' \leftarrow as' + e'$
$v \leftarrow bs' + e''$
$r \xleftarrow{\$} \mathsf{HelpRec}(v)$
$w_0 \xleftarrow{\$} \mathsf{Rec}(v, r)$
$w_1 \xleftarrow{\$} \{0,1\}^n$
$b_{\mathsf{test}} \xleftarrow{\$} \{0,1\}$
$b_{\mathsf{guess}} \xleftarrow{\$} \mathcal{A}(a, b, b', r, w_{b_{\mathsf{test}}})$
return $[\![b_{\mathsf{test}} = b_{\mathsf{guess}}]\!]$
```

Figure 6.4: Decisional ring LWE problem DRLWE (left) and decisional Diffie-Hellman-like problem DDHℓ (right).

Theorem 6.10 (DRLWE $\Longrightarrow$ DDHℓ)**.** *Let q be an odd integer, and χ be a distribution on $\mathcal{R}_q$. It holds: if the decision ring LWE problem for $\mathcal{R}_q, \chi$ is hard, then the DDH-like problem for $\mathcal{R}_q, \chi$ is hard, i.e., there exist efficient adversaries $\mathcal{B}_1, \mathcal{B}_2$ such that*

$$\mathsf{Adv}^{\mathsf{ddh}\ell}_{\mathcal{R}_q,\chi,\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{drlwe}}_{\mathcal{R}_q,\chi,\mathcal{B}_1}(\lambda) + \mathsf{Adv}^{\mathsf{drlwe}}_{\mathcal{R}_q,\chi,\mathcal{B}_2}(\lambda).$$

The proof of Theorem 6.10 can for example be found in [BCNS15, Theorem 1] and is based on Peikert's proof of IND-CPA security of the corresponding key encapsulation mechanism [Pei14, Lemma 4.1].

6.3.2 Security analysis

In the following, we show that Auth-NewHope achieves breakdown resilience with forward secrecy for $\mathcal{F}_{\mathsf{BDR}} = \{(\mathsf{XOF}, \mathsf{RO\text{-}Rand}), (\mathcal{S}, \mathsf{EUF\text{-}CMA}), (\mathcal{M}, \mathsf{EUF\text{-}CMA})\}$ by establishing the corresponding BDR-Match security and BDR key secrecy properties. Note that $\mathcal{F}_{\mathsf{BDR}}$ neither contains the hardness assumptions DRLWE (or DDHℓ), nor the key derivation function KDF, as a break of any of these makes key secrecy impossible to achieve. Without the hardness of DRLWE (and hence $\mathsf{DDH}\ell$) we cannot replace the input to the key derivation function by a uniform random value in order to later argue indistinguishability. Furthermore, the break of KDF causes the adversary to see all previous outputs of the key derivation function, thus trivially enabling it to distinguish real from random keys.

Theorem 6.11. *Let $\mathcal{F}_{\mathsf{BDR}} = \{(\mathsf{XOF}, \mathsf{RO\text{-}Rand}), (\mathcal{S}, \mathsf{EUF\text{-}CMA}), (\mathcal{M}, \mathsf{EUF\text{-}CMA})\}$. Then* Auth-NewHope *is breakdown-resilient for $\mathcal{F}_{\mathsf{BDR}}$ with forward secrecy. More precisely, for any efficient PPT adversary $\mathcal{A}$ there exist efficient adversaries $\mathcal{B}_1$, $\mathcal{B}_2$, $\mathcal{B}_3$, and $\mathcal{B}_4$ such that*

$$\begin{aligned}\mathsf{Adv}^{\mathsf{BDR}(\mathcal{F}_{\mathsf{BDR}}),\mathcal{D}}_{\text{A-NH},\mathcal{A}}(\lambda) \leq\ & n_s^2 \cdot 2^{-|nonce|} + n_s \cdot \Big(n_u \cdot \mathsf{Adv}^{\mathsf{euf\text{-}cma}}_{\mathcal{S},\mathcal{B}_1}(\lambda) \\ +\ & n_s \cdot \Big(2 \cdot \mathsf{Adv}^{\mathsf{ddh}\ell}_{\mathcal{B}_2}(\lambda) + 2 \cdot \mathsf{Adv}^{\mathsf{prf\text{-}sec}}_{\mathsf{KDF},\mathcal{B}_3}(\lambda) + \mathsf{Adv}^{\mathsf{euf\text{-}cma}}_{\mathcal{M},\mathcal{B}_4}(\lambda)\Big)\Big),\end{aligned}$$

where n_s is the maximum number of sessions, n_u is the maximum number of users, and $|nonce|$ is the bit-length of the nonces r_A and r_B.

Proof of $\mathsf{BDR\text{-}Match}$ ***security.*** In order to achieve $\mathsf{BDR\text{-}Match}$ Security, we need to show that the four conditions from Definition 6.2 are satisfied. Recall that the session identifiers are defined as $\mathsf{sid} = (pk, c, r_A, r_B, A, B)$, containing public information only, and that the contributive identifiers are set as $\mathsf{cid} = (pk, c, r_A, r_B, B)$.

Ad 1. Since the session identifier already determines all inputs to the key derivation function KDF, partnered sessions necessarily also agree on the session key.

Ad 2. Since cid contains all entries in sid except for A's identity, it trivially holds that same session identifiers imply identical contributive identifiers.

Ad 3. Both identifiers A and B are comprised in the session identifier. Thus, agreement on the session identifier implies agreement on the intended partner's identity.

Ad 4. In order for three sessions to share the same session or contributive identifier, with respect to two honest sessions a third honest session must pick, as responder, its random values s', e', e'', r, and r_B such that they collide or, as initiator, pick colliding random values seed, s, e, and r_A. This will only happen with negligible probability.

□

Proof of BDR ***key secrecy.***

Game$_0(\lambda)$**:** The original BDR key secrecy game $\mathsf{G}^{\mathsf{BDR}(\mathcal{F}_{\mathsf{BDR}}),\mathcal{D}}_{\text{A-NH},\mathcal{A}}(\lambda)$.

Game$_1(\lambda)$**:** We abort the game if there are two sessions of honest parties which generate the same nonce r_A resp. r_B. The probability of this happening is at most $n_s \cdot 2^{-|nonce|}$, where n_s denotes the maximum number of sessions, since nonces in any n_s^2 possible pairs of sessions are both chosen at random.

We thus have

$$\mathsf{Adv}^{\mathsf{G}_0}_{\text{A-NH},\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{G}_1}_{\text{A-NH},\mathcal{A}}(\lambda) + n_s^2 \cdot 2^{-|nonce|}.$$

Game$_2(\lambda)$**:** We proceed by guessing the tested session, thus reducing our adversary's advantage by a factor of at most $\frac{1}{n_s}$:

$$\mathsf{Adv}^{\mathsf{G}_1}_{\text{A-NH},\mathcal{A}}(\lambda) \leq n_s \cdot \mathsf{Adv}^{\mathsf{G}_2}_{\text{A-NH},\mathcal{A}}(\lambda).$$

In the following, this allows us to know the tested session, denoted by $\pi^\star$, in advance. Observe that $\pi^\star$ must have accepted (and received all incoming messages) prior to the first Break query issued by $\mathcal{A}$ in order for the latter to win, as otherwise its session key would be considered revealed.

Game$_3(\lambda)$**:** Next, we abort the game if the tested session $\pi^\star$, run by some party U (where $U \in \{A, B\}$), obtains a valid signature σ_V on ("b"$||t||r_A||r_B$) which has not been signed by an honest party V at this point. Recall that this message must have been received prior to any Break query, in particular before a breakdown of $\mathcal{S}$, as otherwise $\pi^\star$ would be considered revealed and could not be tested.

Furthermore, long-term secrets of the involved parties may not be corrupted before the test session has accepted. Forward secrecy is achieved since a subsequent $\mathsf{Corrupt}$ query on the owner of the test session $\pi^\star$ (or its intended partner) does not contradict the fact that $\pi^\star$ receives an honestly generated signature according to this game hop.

We now show that the probability of an abort happening for this reason can be bounded by the success probability of the following reduction $\mathcal{B}_1$ against the unforgeability of the signature scheme $\mathcal{S}$.

The reduction $\mathcal{B}_1$ receives a public key $pk^\star$ as challenge and guesses the party V under whose name the forgery obtained in $\pi^\star$ is issued. It creates all parameters for the key exchange as specified, except for setting $pk_V \leftarrow pk^\star$. Any signature creation of V is performed through a query to the signature oracle, all other steps can be carried out by $\mathcal{B}_1$ itself.

If at some point the tested session $\pi^\star$ accepts a signature for a previously unsigned message, then $\mathcal{B}_1$ outputs this message-signature pair as a forgery. In this case, since the nonces are unique and the valid signature has not been created by an honest party before, party V cannot have signed ("b"$||t||r_A||r_B$) earlier, only ("b'"$||t||r_A||r_B$) for $\mathsf{b}' = 1 - \mathsf{b}$ (if at all). With probability $\frac{1}{n_u}$, where n_u is the total number of users, our reduction predicts the party V correctly, such that we have

$$\mathsf{Adv}^{\mathsf{G}_2}_{\text{A-NH},\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{G}_3}_{\text{A-NH},\mathcal{A}}(\lambda) + n_u \cdot \mathsf{Adv}^{\mathsf{euf\text{-}cma}}_{\mathcal{S},\mathcal{B}_1}(\lambda).$$

Game$_4(\lambda)$**:** In the next step, we guess the honest session $\pi^\star_{\mathsf{a}}$ of party V which has sent the valid signature σ_V received by $\pi^\star$ in *Game*$_3$ and abort if we guessed incorrectly. This session is unique because the nonces are unique and there must be such a session which creates the signature according to the previous game. Still, the session may not necessarily be partnered with the test session, but must (at least) have the same contributive identifier, such that we call this session *associated.*

Changing the game like this reduces the adversary's advantage by a factor of at most $\frac{1}{n_s}$, with n_s again being the maximum number of sessions. Hence, we have

$$\mathsf{Adv}^{\mathsf{G}_3}_{\text{A-NH},\mathcal{A}}(\lambda) \leq n_s \cdot \mathsf{Adv}^{\mathsf{G}_4}_{\text{A-NH},\mathcal{A}}(\lambda).$$

Game$_5(\lambda)$**:** As the next step, we replace the value w in the test session (and its associated session $\pi^\star_{\mathsf{a}}$) by a uniformly random value $\widetilde{w}$. If the adversary $\mathcal{A}$ can distinguish *Game*$_5$ from *Game*$_4$, then there exists an adversary $\mathcal{B}_2$ that can solve the $\mathsf{DDH}\ell$ problem as follows.

Algorithm $\mathcal{B}_2$ obtains a $\mathsf{DDH}\ell$ challenge $(\hat{a}, \hat{b}, \hat{b}', \hat{r}, \hat{w})$ and simulates the environment for $\mathcal{A}$ picking the according seed seed' upfront and programming the random oracle modeling XOF such that $\mathsf{XOF}(\mathsf{seed}') = \hat{a}$ before any execution starts.

In the predicted sessions $\pi^\star$ and $\pi^\star_{\mathsf{a}}$ algorithm $\mathcal{B}_2$ then sets $a \leftarrow \hat{a}$, $b \leftarrow \hat{b}$, $b' \leftarrow \hat{b}'$, and $r \leftarrow \hat{r}$. Note that it is irrelevant for the argument if another honest session accidentally picks the same seed seed' and thus derives the same $\hat{a}$ since the non-uniqueness of the parameter does not affect the security of the protocol in terms of key secrecy. In fact, in many RLWE-based key exchange schemes, this parameter is globally fixed upfront for all executions.

Furthermore, a breakdown of XOF does not imply any advantage for the adversary in detecting the simulation (as the value $\hat{a}$ will appear to have been validly generated in an honest execution) or disturbing the programming of the random oracle (since the breakdown can only happen after the test session has been completed). Thus, when computing the keys K and K_{mac} in the two sessions, the given value $\hat{w}$ is used instead as input to the key derivation function KDF.

At some point, $\mathcal{A}$ terminates and outputs a guess bit b_{guess}. Upon this, $\mathcal{B}_2$ also terminates and outputs the same b_{guess}.

If $\hat{w}$ is genuine, , i.e., the internal bit b' of the $\mathsf{DDH}\ell$ challenge is 1, then the simulation above is as in $Game_4$. If $\hat{w}$ is random, i.e., $b' = 0$, $\mathcal{B}_2$ faithfully simulates $Game_5$. Hence, if the efficient adversary $\mathcal{A}$ can distinguish the two games with non-negligible advantage, then $\mathcal{B}_2$ can solve $\mathsf{DDH}\ell$ efficiently with non-negligible advantage. It follows that

$$\mathsf{Adv}^{\mathsf{G}_4}_{\text{A-NH},\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{G}_5}_{\text{A-NH},\mathcal{A}}(\lambda) + 2 \cdot \mathsf{Adv}^{\mathsf{ddh}\ell}_{\mathcal{B}_2}(\lambda).$$

Since $\mathsf{DDH}\ell$ is not part of $\mathcal{F}_{\mathsf{BDR}}$, this bound especially holds in the BDR scenario.

$Game_6(\lambda)$**:** Next, we replace the session key K and the MAC key K_{mac} by uniformly random values $\widetilde{\mathsf{K}}$ and $\widetilde{K_{\mathsf{mac}}}$ from the appropriate key space $\mathcal{D}$ in $\pi^\star$ and $\pi^\star_{\mathsf{a}}$. Distinguishing $Game_6$ and $Game_5$ by $\mathcal{A}$ would immediately imply the existence of an efficient adversary $\mathcal{B}_3$ that breaks the pseudorandomness of KDF with non-negligible advantage.

The reduction $\mathcal{B}_3$ generates all key exchange parameters and long-term keys itself and initializes $\mathcal{A}$. This means in particular, that $\mathcal{B}_3$ can answer all NewSession, Corrupt and Reveal queries (assuming $\mathcal{A}$ does not query Reveal on $\pi^\star$ and $\pi^\star_{\mathsf{a}}$ wlog as this would result in a trivial loss for $\mathcal{A}$.)

Similarly, $\mathcal{B}_3$ can faithfully execute all Send queries. For $\pi^\star$ and $\pi^\star_{\mathsf{a}}$ when the MAC tags are computed, $\mathcal{B}_3$ queries its oracle $\mathcal{O}_{\mathsf{PRF}}$ in the pseudorandomness game on $\texttt{"MAC"}||t$ where t is the appropriate transcript. The response $K_{b'}$ of the oracle is then either $\mathsf{KDF}(K, \texttt{"MAC"}||t)$ if $b' = 0$ for some key K that was chosen at the beginning of the pseudorandomness game or $g(\texttt{"MAC"}||t)$ for some random function g if $b' = 1$. The reduction then uses $K_{b'}$ as MAC key in $\pi^\star$ and $\pi^\star_{\mathsf{a}}$. Note again that signatures can be faithfully simulated by the reduction since it has generated the long-term keys of the participants.

Once $\mathcal{A}$ asks a Test query, $\mathcal{B}_3$ queries $\texttt{"KE"}||(t, r_A, r_B, A, B)$ to $\mathcal{O}_{\mathsf{PRF}}$ to receive $\mathsf{K}_{b'}$ which is either $\mathsf{KDF}(K, \texttt{"KE"}||(t, r_A, r_B, A, B))$ if $b' = 0$ or $g(\texttt{"KE"}||(t, r_A, r_B, A, B))$ if $b' = 1$. The reduction returns $\mathsf{K}_{b'}$ as K_{test} to $\mathcal{A}$.

At some point, $\mathcal{A}$ terminates with output b_{guess} which $\mathcal{B}_3$ also uses as its output.

Observe that $\mathcal{B}_3$ perfectly simulates $Game_5(\lambda)$ if $b' = 0$ and $Game_6(\lambda)$ if $b' = 1$. Thus, if $\mathcal{A}$ can distinguish the two games, $\mathcal{B}_3$ can win the pseudorandomness game against KDF and we have

$$\mathsf{Adv}^{\mathsf{G}_5}_{\text{A-NH},\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{G}_6}_{\text{A-NH},\mathcal{A}}(\lambda) + 2 \cdot \mathsf{Adv}^{\mathsf{prf\text{-}sec}}_{\mathsf{KDF},\mathcal{B}_3}(\lambda).$$

As $(\mathsf{KDF}, \cdot) \notin \mathcal{F}_{\mathsf{BDR}}$, this bound again particularly holds in the BDR scenario.

There are now four possibilities for the status of the associated session $\pi^\star_{\mathsf{a}}$.

1. First, at the point of the breakdown query, the associated session had **not accepted yet**, i.e., $\pi^\star_{\mathsf{a}}$ owned by V has the role $\mathsf{responder}$ and waited for the final authentication message. But then $\pi^\star_{\mathsf{a}}$'s state was $\mathsf{running}$ and its contributive identifier was, and is, identical to the one in U's session $\pi^\star$, since the signature is over the entries in cid and V knows resp. sent its identifier. This however means that the adversary is not allowed to test the session it has actually tested, by definition of a successful attack.
2. The associated session $\pi^\star_{\mathsf{a}}$ had already **finished** upon the breakdown query with status $\mathsf{rejected}$. This means that it does not hold a session key and is therefore of no relevance to the authentication.
3. The associated session $\pi^\star_{\mathsf{a}}$ had already **finished** upon the breakdown query and it is **partnered** with the test session. But then it cannot be revealed by the adversary.
4. Lastly, the associated session $\pi^\star_{\mathsf{a}}$ had already **finished** upon the breakdown query but it is **not partnered** with the test session. But this case would allow the adversary to safely reveal the session key of the associated (but unpartnered) session, which could break key secrecy. Yet, this would lead to a contradiction of the unforgeability of the MAC, as we discuss next.

Game$_7(\lambda)$**:** As the final change, we abort the game if the associated session $\pi^\star_{\mathsf{a}}$ of party V accepts before the breakdown query with a session identifier $\pi^\star_{\mathsf{a}}.\mathsf{sid} \neq \perp$ which does not equal $\pi^\star.\mathsf{sid}$. This can only happen if the adversary is able to make $\pi^\star_{\mathsf{a}}$ obtain a valid signature σ_W and MAC tag τ_W for some identity $W \neq U$ since all entries except for the peer identity of $\pi^\star_{\mathsf{a}}.\mathsf{sid}$ are already fixed at this point.

We assume that the associated session has already accepted and that no Break query has occurred yet. In particular, while the adversary may be able to sign under a corrupt party's identifier W, the MAC scheme, on the other hand, must still be secure. Furthermore, the MAC tag depends on the key K_{mac} shared between the honest parties U and V and includes the sender's identity.

Similarly to *Game*$_3$, the probability of an abort happening for this reason can be bounded by the success probability of an adversary $\mathcal{B}_4$ against the unforgeability of the MAC scheme $\mathcal{M}$. That is, since we have already replaced the key K_{mac} by an independent random value, we can use an external $\mathcal{O}_{\mathsf{MAC}}$ oracle for an unknown key in a simulation instead, and use oracle queries to create the MACs for $\texttt{"b"}||U$ and $\texttt{"b'"}||V$ for $\mathsf{b}' = 1 - \mathsf{b}$ as required in the test session and its associated session. It follows that a valid MAC τ_W for $\texttt{"b"}||W$ created by the adversary for identity $W \neq U$ in the associated session constitutes a successful forgery for a fresh message. We have

$$\mathsf{Adv}^{\mathsf{G}_6}_{\text{A-NH},\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{G}_7}_{\text{A-NH},\mathcal{A}}(\lambda) + \mathsf{Adv}^{\mathsf{euf\text{-}cma}}_{\mathcal{M},\mathcal{B}_4}(\lambda).$$

To complete the proof we note that the adversary expects the challenge value K_{test} to be either the output of $\mathsf{KDF}(w, \text{"}\mathsf{KE}\text{"}||(t, r_A, r_B, A, B))$ ($b_{\mathsf{test}} = 0$) or a uniformly random string ($b_{\mathsf{test}} = 1$). At this point, both cases $b_{\mathsf{test}} = 0$ and $b_{\mathsf{test}} = 1$ are indistinguishable for $\mathcal{A}$ since both keys are of equal length and are drawn independently and uniformly at random. Furthermore, the session key in the associated session (which coincides with the now random key K_{test} in case of $b_{\mathsf{test}} = 0$ and is independent of K_{test} for $b_{\mathsf{test}} = 1$) cannot be revealed, because that session is either partnered or held the same contributive identifier upon breakdown. Thus $\mathcal{A}$ cannot learn any information about the bit b_{test}. The only strategy for $\mathcal{A}$ is to guess and thus we have the final bound:

$$\mathsf{Adv}^{\mathsf{G}_7}_{\text{A-NH},\mathcal{A}}(\lambda) \leq 0.$$

□

Remark 6.12. By modeling the breakdown of the signatures and MACs, while the unauthenticated key agreement and the key derivation function remain secure, we, in particular, re-confirm the result from the previous chapter that for $\mathsf{C^cQ}$ security, i.e. security against future-quantum adversaries, it is not necessary for the authentication mechanisms to remain secure in order to protect sessions before the breakdown.

A note on the security bound. It may be surprising at first that the unforgeability of the signature and MAC scheme enter into the security bound of Theorem 6.11 although the signature scheme $\mathcal{S}$ as well as the MAC scheme $\mathcal{M}$ are affected by the breakdown. However, both the valid signature obtained by $\pi^\star$ in *Game*$_3$ as well as the MAC tag in *Game*$_7$ must necessarily have been created before a breakdown had occurred. Thus, both unforgeability assumptions still hold at the respective points in time.

The security of the extendable-output function XOF (modeled as a random oracle) does not enter into the security bound for key secrecy. We recall that in NewHope-Usenix the function XOF is applied to a uniformly random chosen seed to generate the public parameter a freshly for each protocol execution. This is done to avoid backdoors and all-for-the-price-of-one-attacks. However, the security of RLWE-based protocols does not rely on the parameter a being indistinguishable from random as it is in general public and fixed for all executions (cf. for example [BCNS15]).

Furthermore, note that the key derivation function KDF and the extendable-output function XOF proposed in [ADPS16b] both rely on the (pseudo-)randomness of SHA-3. In our analysis we treat these two primitives as independent and generic cryptographic building blocks such that a break of XOF does not imply a break in the key derivation function KDF (and vice versa). This result shows, that the protocol can withstand breakdowns of XOF if the key derivation function KDF is based on a different primitive.

6.3.3 Encryption-based Auth-NewHope

As mentioned before, the encryption-based variants of NewHope are formalized as key encapsulation mechanisms. In Figure 6.5, we see the key exchange flow Auth-Kem between Alice and Bob in this setting, with only the parts shown that differ from Auth-NewHope in Figure 6.3. Note, that there is nothing peculiar that distinguishes this variant from any other KEM flow.

Therefore, we can show the breakdown resilience of any authenticated key exchange protocol Auth-Kem that combines a KEM $\mathcal{K}$ with SigMA-authentication. For the precise specifications of the algorithms $\mathsf{KGen}, \mathsf{Encaps}$ and Decaps for the case of NewHope, we refer the interested reader to [AAB+19].

Theorem 6.13 (BDR security of Auth-Kem)**.** *Let* $\mathcal{F}_{\mathsf{BDR}} = \{(\mathcal{S}, \mathsf{EUF\text{-}CMA}), (\mathcal{M}, \mathsf{EUF\text{-}CMA})\}$. *Then* Auth-Kem *is breakdown resilient for* $\mathcal{F}_{\mathsf{BDR}}$ *with forward secrecy. More precisely, for any efficient, adversary* $\mathcal{A}$ *there exist efficient adversaries* $\mathcal{B}_1$, $\mathcal{B}_2$, $\mathcal{B}_3$ *and* $\mathcal{B}_4$ *such that*

$$\begin{aligned}\mathsf{Adv}^{\mathsf{BDR}(\mathcal{F}_{\mathsf{BDR}}),\mathcal{D}}_{\text{A-KEM},\mathcal{A}} \leq\ & n_s^2 \cdot 2^{-|nonce|} + n_s \cdot \Big(n_u \cdot \mathsf{Adv}^{\mathsf{euf\text{-}cma}}_{\mathcal{S},\mathcal{B}_1} \\ & + n_s \cdot \Big(2 \cdot \mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathcal{K},\mathcal{B}_2} + 2 \cdot \mathsf{Adv}^{\mathsf{prf\text{-}sec}}_{\mathsf{KDF},\mathcal{B}_3} + \mathsf{Adv}^{\mathsf{euf\text{-}cma}}_{\mathcal{M},\mathcal{B}_4}\Big)\Big),\end{aligned}$$

where n_s is the maximum number of sessions, n_u is the maximum number of users, and $|nonce|$ is the bit-length of the nonces.

Proof. The proof is mostly analogous to the proof of Theorem 6.11. In the proof of key secrecy $Game_5(\lambda)$ is replaced by the following:

$Game'_5(\lambda)$: As the next step, we replace the value K in the test session (and its associated session $\pi^\star_{\mathsf{a}}$) by a uniformly random value $\widetilde{K}$ of equal length. If the adversary $\mathcal{A}$ can distinguish $Game'_5(\lambda)$ from $Game_4(\lambda)$, then there exists an adversary $\mathcal{B}_2$ that can break the IND-CPA security of the key encapsulation mechanism $\mathcal{K}$ as follows.

Algorithm $\mathcal{B}_2$ obtains its challenge public key, ciphertext and key $epk, c^\star, K^\star_{b'}$, where either $(c^\star, K^\star_{b'}) \xleftarrow{\$} \mathsf{Encaps}(epk)$ $(b' = 0)$ or $K^\star_{b'}$ is a random element from the key space $(b' = 1)$.

$\mathcal{B}_2$ simulates the environment for $\mathcal{A}$ by creating all long-term keys of participants as specified and initializing $\mathcal{A}$ with the resulting public keys of participants. This ensures in particular, that $\mathcal{B}_2$ can answer all NewSession and Corrupt queries of the adversary.

Furthermore, $\mathcal{B}_2$ can execute all Send requests by $\mathcal{A}$ for sessions other than the test session and the associated session $\pi \neq \pi^\star, \pi^\star_{\mathsf{a}}$. For $\pi^\star$ and $\pi^\star_{\mathsf{a}}$, $\mathcal{B}_2$ uses its challenge epk and $c^\star$ for the first two message flows. The session key K and the MAC key K_{mac} are computed as the KDF keyed with $K^\star_{b'}$ and the respective label. $\mathcal{B}_2$ can also answer all Reveal for sessions that are not the test session or its associated session. For $\pi^\star$ and $\pi^\star_{\mathsf{a}}$, $\mathcal{A}$ will not query Reveal since this would cause it to trivially lose the game.

Once $\mathcal{A}$ queries Test on $\pi^\star$, the reduction $\mathcal{B}_2$ computes the challenge key for $\mathcal{A}$ as $\mathsf{KDF}(K^\star_{b'}, \text{"KE"}||(t, r_A, r_B, A, B))$, i.e., when computing the keys K and K_{mac} in the two sessions, the given value $K^\star_{b'}$ is used instead as input to the key derivation function KDF.

At some point, $\mathcal{A}$ terminates and outputs a guess bit b_{guess}. Upon this, $\mathcal{B}_2$ also terminates and outputs the same b_{guess}.

If $K^\star_{b'}$ is genuine (i.e., $b' = 0$), then the simulation above is as in $Game_4(\lambda)$. If $K^\star_{b'}$ is random (i.e., $b' = 1$), $\mathcal{B}_2$ simulates $Game'_5(\lambda)$. Hence, if the efficient adversary $\mathcal{A}$ can distinguish the two games with non-negligible advantage, then $\mathcal{B}_2$ can distinguish real from random keys in key encapsulation mechanisms efficiently with non-negligible advantage.

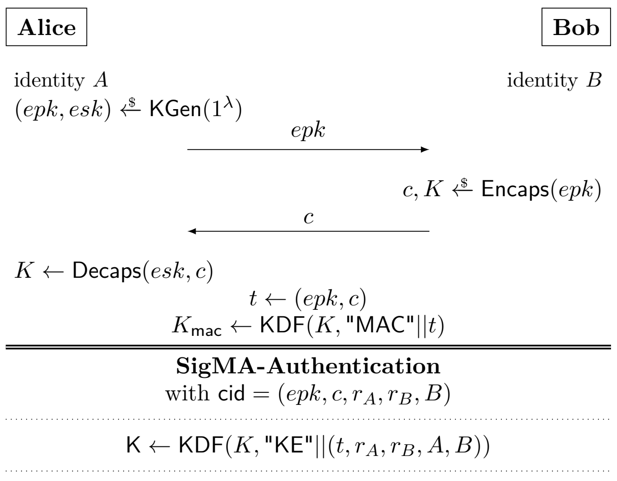
Alice
Bob
identity A
identity B
$(epk, esk) \xleftarrow{\$} \mathsf{KGen}(1^\lambda)$
epk
$c, K \xleftarrow{\$} \mathsf{Encaps}(epk)$
c
$K \leftarrow \mathsf{Decaps}(esk, c)$
$t \leftarrow (epk, c)$
$K_{\mathsf{mac}} \leftarrow \mathsf{KDF}(K, \text{"MAC"}||t)$
SigMA-Authentication
with $\mathsf{cid} = (epk, c, r_A, r_B, B)$
$\mathsf{K} \leftarrow \mathsf{KDF}(K, \text{"KE"}||(t, r_A, r_B, A, B))$

Figure 6.5: The NewHope-Nist key encapsulation mechanism $\mathcal{K} = (\mathsf{KGen}, \mathsf{Encaps}, \mathsf{Decaps})$.

It follows that

$$\mathsf{Adv}^{\mathsf{G}_4}_{\textsc{Auth-Kem},\mathcal{A}} \leq \mathsf{Adv}^{\mathsf{G}'_5}_{\textsc{Auth-Kem},\mathcal{A}} + 2 \cdot \mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathcal{K},\mathcal{B}_2}.$$

Since $(\mathcal{K}, \mathsf{IND\text{-}CPA})$ is not part of $\mathcal{F}_{\mathsf{BDR}}$, this bound especially holds in the BDR scenario.

□

6.4 Strong Breakdown Resilience of Hybrid AKE

In this section, we show how our model of (strong) breakdown resilience from Section 6.2.3 can be leveraged to achieve results about hybrid AKE constructions similar to the two-stage model in Section 5.5.

Comparison with two-stage hybrid AKE. Chapter 5 was specifically concerned with the question of how to design hybrid key exchanges that ease the transition from the classical to the post-quantum setting. The (strong) breakdown-resilience framework on the other hand is not concerned with *what* causes the breakdowns and is thus meaningful in generic scenarios. However, we can make statements that are comparable to the setting of hybrid key exchanges with quantum adversaries.

For an authenticated key exchange protocol KE and appropriate definitions of the set $\mathcal{F}_{\mathsf{BDR}}$, the notion of strong breakdown resilience coincides with the notion of two-stage $\mathsf{X^cZ\text{-}BR}$ security (Def. 5.17) for $\mathsf{X^cZ} \in \{\mathsf{C^cC}, \mathsf{Q^cQ}\}$, i.e., for either classical or post-quantum adversaries. Two-stage $\mathsf{C^cQ\text{-}BR}$ security, i.e., BR security against future-quantum adversaries, is captured by the basic breakdown-resilience model for AKE (Def. 6.4). This is visible in the security analyses of the NewHope AKE protocols in Section 6.3, where we employed a post-quantum secure key exchange scheme with classical authentication, generating the same result for future-quantum adversaries as in the generic hybrid AKE compiler analysis in Section 5.5.

We furthermore note that we do not see any benefits in a definition of (strong) breakdown-resilient $\mathsf{IND\text{-}ATK}$ security of KEMs analogous to the two-stage $\mathsf{IND\text{-}ATK}$ definitions (cf. Def. 5.2) that we used to establish the security of our hybrid KEM constructions. Our aim when considering (strong) breakdown resilience is to show security of entire AKE protocols when underlying primitives or hardness assumptions break. On a primitive level, if one is not specifically interested in the impacts of quantum adversaries, the existing proof techniques for combiners are sufficient (cf., e.g., [GHP18] for examples of such proofs).

6.4.1 The AKE Compiler

As an example for an AKE compiler, we again consider the combination of a key encapsulation mechanism with SigMA-authentication as in the previous chapters. In particular, this facilitates direct comparison with the results on hybrid AKE security from Section 5.5.

We choose to analyse the compiler $\mathcal{C}_{\mathsf{SigMA}}[\mathsf{dualPRF}, \mathcal{S}, \mathcal{M}, \mathsf{KDF}]$ consisting of the dual-PRF combiner $\mathsf{dualPRF}[\mathcal{K}_1, \mathcal{K}_2, \mathsf{dPRF}, \mathsf{F}]$ (cf. Figure 5.11), a signature scheme $\mathcal{S}$, MAC scheme $\mathcal{M}$, and key derivation function KDF modeled as a PRF. The compiler is essentially depicted in Figure 6.5, where instead of the NewHope-Nist KEM we use the algorithmic descriptions of $\mathsf{KGen}, \mathsf{Encaps}$, and Decaps of the $\mathsf{dualPRF}$ combiner. As mentioned before, in the strong breakdown resilience setting we have no needs for contributive identifiers and thus their consideration can be omitted for this analysis.

Since we only have defined dual-PRF security in the two-stage setting so far, we want to take this opportunity to define it also in the (standard) adversarial setting:

Definition 6.14 (Dual-PRF security)**.** *Let* $\mathsf{F} : \{0,1\}^{\kappa(\lambda)} \times \{0,1\}^{\iota(\lambda)} \to \{0,1\}^{\omega(\lambda)}$ *be an efficient keyed function key length* $\kappa(\lambda)$*, input length* $\iota(\lambda)$ *and output length* $\omega(\lambda)$*. Define* $\mathsf{F}' : \{0,1\}^{\iota(\lambda)} \times \{0,1\}^{\kappa(\lambda)} \to \{0,1\}^{\omega(\lambda)}$ *such that* $\mathsf{F}'(x, K) := \mathsf{F}(K, x)$*. Let* $\mathcal{A}$ *be a PPT adversary interacting with* F, F' *in the Game* $\mathsf{G}^{\mathsf{dprf\text{-}sec}}_{\mathsf{F},\mathcal{A}}(\lambda)$ *given in Figure 6.6.*

We say that F *is a* dual pseudorandom function *if both* F *and* F' *are pseudorandom functions according to Definition 3.13. In particular, for all PPT adversaries* $\mathcal{A}$ *the advantage function defined as*

$$\begin{aligned}\mathsf{Adv}^{\mathsf{dprf\text{-}sec}}_{\mathsf{F},\mathcal{A}}(\lambda) &:= \left| \Pr\left[\mathsf{G}^{\mathsf{dprf\text{-}sec}}_{\mathsf{F},\mathcal{A}}(\lambda) = 1\right] - \frac{1}{2} \right| \\ &= \max\left\{\mathsf{Adv}^{\mathsf{prf\text{-}sec}}_{\mathsf{F},\mathcal{A}}(\lambda), \mathsf{Adv}^{\mathsf{prf\text{-}sec}}_{\mathsf{F}',\mathcal{A}}(\lambda)\right\}\end{aligned}$$

is negligible in the security parameter λ*.*

```
$\mathsf{G}^{\mathsf{dprf\text{-}sec}}_{\mathsf{F},\mathcal{A}}(\lambda)$:
1  $K \xleftarrow{\$} \{0,1\}^{\kappa(\lambda)}$
2  $x \xleftarrow{\$} \{0,1\}^{\iota(\lambda)}$
3  $g \xleftarrow{\$} \{\text{functions } f : \{0,1\}^{\iota(\lambda)} \to \{0,1\}^{\omega(\lambda)}\}$
4  $g' \xleftarrow{\$} \{\text{functions } f : \{0,1\}^{\kappa(\lambda)} \to \{0,1\}^{\omega(\lambda)}\}$
5  $b \xleftarrow{\$} \{0,1\}$
6  $b' \xleftarrow{\$} \mathcal{A}^{\mathcal{O}^{\mathsf{F}}_{\mathsf{PRF}}, \mathcal{O}^{\mathsf{F}'}_{\mathsf{PRF}}}$
7  return $[\![b' = b]\!]$

$\mathcal{O}^{\mathsf{F}}_{\mathsf{PRF}}(x)$:
8  if $b = 0$
9      return $\mathsf{F}(K, x)$
10 else
11     return $g(x)$

$\mathcal{O}^{\mathsf{F}'}_{\mathsf{PRF}}(K)$:
12 if $b = 0$
13     return $\mathsf{F}'(x, K)$
14 else
15     return $g'(K)$
```

Figure 6.6: Definition of a dual pseudorandom function F.

6.4.2 Security Analysis

We now conduct the security analysis of the $\mathcal{C}_{\mathsf{SigMA}}$ compiler with the $\mathsf{dualPRF}$ combiner in the strong breakdown-resilience model. The aim of this analysis is to show that this model is a viable alternative for analysis of post-quantum hybrid AKE. The previous chapter considered AKE from KEM that had already been proven to be hybrid. Now we implicitly prove the hybrid property of the KEM simultaneously with the proof for the hybrid AKE.

Choice of $\mathcal{F}_{\mathsf{BDR}}$ **and specification of oracle** Break**.** The choice for $\mathcal{F}_{\mathsf{BDR}}$ in the hybrid AKE setting is straightforward, as this can only contain components of “combiner primitives”. Since $\mathsf{dualPRF}$ is built from two KEMs $\mathcal{K}_1$ and $\mathcal{K}_2$, and $\mathcal{C}_{\mathsf{SigMA}}$ requires the key exchanges to be secure at least against chosen-plaintext attacks, we have that $\mathcal{F}_{\mathsf{BDR}} \in \{\{(\mathcal{K}_1, \mathsf{IND\text{-}ATK})\}, \{(\mathcal{K}_2, \mathsf{IND\text{-}ATK})\}\}$.

As mentioned earlier, security analyses aim to be conservative when it comes to the power of the adversary since stronger adversaries lead to more meaningful results. Thus when $\mathcal{A}$ calls Break, it will get access to an unrestricted decapsulation oracle $\mathcal{O}^{\mathcal{F}_{\mathsf{BDR}}}_{\mathsf{Decaps}}$ for the KEM specified by $\mathcal{F}_{\mathsf{BDR}}$. On input a public key pk and ciphertext c, the oracle then returns the decapsulation $\mathsf{Decaps}_1(sk, c)$ for $\mathcal{F}_{\mathsf{BDR}} = \{(\mathcal{K}_1, \mathsf{IND\text{-}ATK})\}$ and analogously $\mathsf{Decaps}_2(sk, c)$ for $\mathcal{F}_{\mathsf{BDR}} = \{(\mathcal{K}_2, \mathsf{IND\text{-}ATK})\}$, where sk is the corresponding secret key to pk.

Alternative specifications for Break could be to hand either all previously generated key pairs or all previous outputs of the encapsulation algorithm of the respective KEM to the adversary. The solution with the decapsulation oracle is slightly more elegant as it does not require the adversary to call Break multiple times to get the latest information since the last call of Break. This is however a minor consideration and does not make any difference in the proof.

Theorem 6.15 ($\mathcal{C}_{\mathsf{SigMA}}[\mathsf{dualPRF}, \mathcal{S}, \mathcal{M}, \mathsf{KDF}]$ is strongly breakdown resilient)**.**
Let $\mathcal{F}_{\mathsf{BDR}} \in \{\{(\mathcal{K}_1, \mathsf{IND\text{-}ATK})\}, \{(\mathcal{K}_2, \mathsf{IND\text{-}ATK})\}\}$. *Let* $\mathsf{KE} := \mathcal{C}_{\mathsf{SigMA}}[\mathsf{dualPRF}, \mathcal{S}, \mathcal{M}, \mathsf{KDF}]$ *with* $\mathsf{dualPRF}[\mathcal{K}_1, \mathcal{K}_2, \mathsf{dPRF}, \mathsf{F}]$ *the KEM combiner defined as in Figure 5.11. Then* KE *achieves strong breakdown resilience for* $\mathcal{F}_{\mathsf{BDR}}$ *with forward secrecy. More precisely, for any efficient adversary* $\mathcal{A}$ *there exist efficient adversaries* $\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_7$ *such that*

$$\mathsf{Adv}^{\mathsf{sBDR}(\mathcal{F}_{\mathsf{BDR}}),\mathcal{D}}_{\mathsf{KE},\mathcal{A}} \leq n_s^2 \cdot 2^{-|nonce|} + n_s \cdot \Big(n_u \cdot \mathsf{Adv}^{\mathsf{euf\text{-}cma}}_{\mathcal{S},\mathcal{B}_1} + n_s \cdot \Big(2 \cdot \min\Big\{\mathsf{Adv}^{\mathsf{ind\text{-}atk}}_{\mathcal{K}_2,\mathcal{B}_2}(\lambda), \mathsf{Adv}^{\mathsf{ind\text{-}atk}}_{\mathcal{K}_1,\mathcal{B}_3}(\lambda)\Big\}$$
$$+ 2 \cdot \mathsf{Adv}^{\mathsf{dprf\text{-}sec}}_{\mathsf{dPRF},\mathcal{B}_4}(\lambda) + 2 \cdot \mathsf{Adv}^{\mathsf{prf\text{-}sec}}_{\mathsf{F},\mathcal{B}_5}(\lambda) + 2 \cdot \mathsf{Adv}^{\mathsf{prf\text{-}sec}}_{\mathsf{KDF},\mathcal{B}_6}(\lambda) + \mathsf{Adv}^{\mathsf{euf\text{-}cma}}_{\mathcal{M},\mathcal{B}_7}\Big)\Big),$$

where n_s *is the maximum number of sessions,* n_u *is the maximum number of users, and* $|nonce|$ *is the bit-length of the nonces.*

Proof. For the proof we focus on the case where $\mathcal{F}_{\mathsf{BDR}} = \{(\mathcal{K}_1, \mathsf{IND\text{-}CCA})\}$. The case where $\mathcal{K}_2$ becomes insecure works analogously. The $\mathsf{IND\text{-}CPA}$ case follows from the $\mathsf{IND\text{-}CCA}$ case.

Note that $\mathsf{sBDR\text{-}Match}$ security is independent of the adversarial model and the employed KEM and thus follows from the Match security established for $\mathcal{C}_{\mathsf{SigMA}}$ in Section 5.5. Next we show sBDR key secrecy.

Game$_0(\lambda)$**:** The original sBDR key secrecy game $\mathsf{G}^{\mathsf{sBDR}(\mathcal{F}_{\mathsf{BDR}}),\mathcal{D}}_{\mathsf{KE},\mathcal{A}}(\lambda)$.

Game$_1(\lambda)$**:** We abort the game if there are two sessions of honest parties which generate the same nonce r_A resp. r_B. The probability of this happening is at most $n_s \cdot 2^{-|nonce|}$, where n_s denotes the maximum number of sessions, since nonces in any n_s^2 possible pairs of sessions are both chosen at random.

We thus have

$$\mathsf{Adv}^{\mathsf{G}_0}_{\mathsf{KE},\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{G}_1}_{\mathsf{KE},\mathcal{A}}(\lambda) + n_s^2 \cdot 2^{-|nonce|}.$$

Game$_2(\lambda)$**:** We proceed by guessing the tested session, thus reducing our reduction's advantage by a factor of at most $\frac{1}{n_s}$:

$$\mathsf{Adv}^{\mathsf{G}_1}_{\mathsf{KE},\mathcal{A}}(\lambda) \leq n_s \cdot \mathsf{Adv}^{\mathsf{G}_2}_{\mathsf{KE},\mathcal{A}}(\lambda).$$

In the following, this allows us to know the tested session, denoted by $\pi^\star$, in advance. Observe that, since we are in the strong breakdown-resilience setting, $\pi^\star$ must not have accepted prior to the first Break query issued by the adversary.

Game$_3(\lambda)$**:** Next, we abort the game if the tested session $\pi^\star$, run by some party U (where $U \in \{A, B\}$), obtains a valid signature σ_V on ("b"$||t||r_A||r_B$) which has not been signed by an honest party V at this point.

Note that long-term secrets of the involved parties may not be corrupted before the test session has accepted. Forward secrecy is achieved since a subsequent $\mathsf{Corrupt}$ query on the owner of the test session $\pi^\star$ (or its intended partner) does not contradict the fact that $\pi^\star$ receives an honestly generated signature according to this game hop.

We now show that the probability of an abort happening for this reason can be bounded by the success probability of the following reduction $\mathcal{B}_1$ against the unforgeability of the signature scheme $\mathcal{S}$. The reduction $\mathcal{B}_1$ receives a public key $pk^\star$ as challenge and guesses the party V under whose name the forgery obtained by $\pi^\star$ is issued. It creates all parameters for the key exchange as specified, except for setting $pk_V \leftarrow pk^\star$. Any signature creation of V is performed through a query to the signature oracle, all other steps can be carried out by $\mathcal{B}_1$ itself.

If at some point the tested session $\pi^\star$ accepts a signature for a previously unsigned message, then $\mathcal{B}_1$ outputs this message-signature pair as a forgery. In this case, since the nonces are unique and the valid signature has not been created by an honest party before, party V cannot have signed ("b"$||t||r_A||r_B$) earlier, only ("b′"$||t||r_A||r_B$) for $\mathsf{b}' = 1 - \mathsf{b}$ (if at all). With probability $\frac{1}{n_u}$, where n_u is the total number of users, our reduction predicts the party V correctly, such that we have

$$\mathsf{Adv}^{\mathsf{G}_2}_{\mathsf{KE},\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{G}_3}_{\mathsf{KE},\mathcal{A}}(\lambda) + n_u \cdot \mathsf{Adv}^{\mathsf{euf\text{-}cma}}_{\mathcal{S},\mathcal{B}_1}(\lambda).$$

As $(\mathcal{S}, \cdot) \notin \mathcal{F}_{\mathsf{BDR}}$, this bound again particularly holds in the sBDR scenario.

A call of Break at any point in time, results in the adversary gaining access to an unrestricted decapsulation oracle $\mathcal{O}^{\mathcal{F}_{\mathsf{BDR}}}_{\mathsf{Decaps}}$ of $\mathcal{K}_1$. Since a break of the primitive implies that there exists an efficient algorithm to solve the underlying problem, the reduction can answer all decapsulation requests, even those that involve public keys and ciphertexts that have been adversarially generated. This is the same for all reductions in this proof and we will not mention it further.

Game$_4(\lambda)$**:** In the next step, we guess the honest session $\pi^\star_{\mathsf{a}}$ of party V which has sent the valid signature σ_V received by $\pi^\star$ in *Game*$_3(\lambda)$ and abort if we guessed incorrectly. This session is unique because the nonces are unique and there must be such a session which creates the signature according to the previous game.

Changing the game like this reduces the adversary's advantage by a factor of at most $\frac{1}{n_s}$, with n_s again being the maximum number of sessions. Hence, we have

$$\mathsf{Adv}^{\mathsf{G}_3}_{\mathsf{KE},\mathcal{A}}(\lambda) \leq n_s \cdot \mathsf{Adv}^{\mathsf{G}_4}_{\mathsf{KE},\mathcal{A}}(\lambda).$$

Game$_5(\lambda)$**:** We now replace the value K_2 in the computations of $\pi^\star$ and $\pi^\star_{\mathsf{a}}$ by a uniformly random element $\widetilde{K_2}$ from the same key space $\mathscr{K}_2$. The real session key $(b = 0)$ is then computed from the value $K \leftarrow \mathsf{F}(\mathsf{dPRF}(K_1, \widetilde{K_2}), c)$.

We argue that if there exists an efficient adversary $\mathcal{A}$ that can distinguish *Game*$_4(\lambda)$ from *Game*$_5(\lambda)$, this implies the existence of an efficient adversary $\mathcal{B}_2$ against the IND-CCA security of $\mathcal{K}_2$. The reduction works as follows:

Algorithm $\mathcal{B}_2$ receives as input $(pk^\star_2, c^\star_2, K^\star_{b'})$, where $pk^\star_2$ is the public key, $c^\star_2$ the challenge ciphertext, and $K^\star_{b'}$ the real or random challenge key in the IND-CCA game for $\mathcal{K}_2$.

To initialize the environment for $\mathcal{A}$, the reduction $\mathcal{B}_2$ generates all long-term key pairs of participants. This enables $\mathcal{B}_2$ to correctly establish all new sessions via NewSession requests of $\mathcal{A}$ and return sound answers to any Corrupt queries.

Send queries that involve sessions other than $\pi^\star$ and $\pi^\star_{\mathsf{a}}$, can be executed by $\mathcal{B}_2$. For the predicted sessions $\pi^\star$ and $\pi^\star_{\mathsf{a}}$, $\mathcal{B}_2$ chooses $(pk_1, sk_1) \xleftarrow{\$} \mathsf{KGen}_1$ itself and computes the respective encapsulation $(c_1, K_1) \xleftarrow{\$} \mathsf{Encaps}_1(pk_1)$. In the second component it injects its challenge, i.e., $\mathcal{B}_2$ sets $pk \leftarrow (pk_1, pk^\star_2)$ and $c \leftarrow (c_1, c^\star_2)$. The MAC key K_{mac} and the session key K are computed from the value $K \leftarrow \mathsf{F}(\mathsf{dPRF}(K_1, \widetilde{K_2}), c)$.

All Reveal queries on sessions $\pi \neq \pi^\star, \pi^\star_{\mathsf{a}}$ can be answered faithfully by $\mathcal{B}_2$. Note that we can assume that $\mathcal{A}$ will not query Reveal on $\pi^\star$ or $\pi^\star_{\mathsf{a}}$ as this would result in trivial loss of the game for $\mathcal{A}$.

When $\mathcal{A}$ queries Test on $\pi^\star$, the reduction computes the challenge key as $\mathsf{KDF}(K, \text{"KE"}||(t, r_A, r_B, A, B))$, where $K \leftarrow \mathsf{F}(\mathsf{dPRF}(K_1, K_{b'}^\star), c)$.

At some point, $\mathcal{A}$ terminates and outputs a guess bit b_{guess}. The reduction $\mathcal{B}_2$ then outputs the same bit b_{guess}.

Clearly, $\mathcal{B}_2$ simulates the environment for $\mathcal{A}$ corresponding to $Game_0(\lambda)$ if its challenge key $K_{b'}^\star$ is the actual key (i.e., if $b' = 0$), and corresponding to $Game_1(\lambda)$ if $K_{b'}^\star$ is random (i.e., $b' = 1$).

Hence, we have

$$\mathsf{Adv}^{\mathsf{G}_4}_{\mathsf{KE},\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{G}_5}_{\mathsf{KE},\mathcal{A}}(\lambda) + 2 \cdot \mathsf{Adv}^{\mathsf{ind\text{-}cca}}_{\mathcal{K}_2,\mathcal{B}_2}(\lambda).$$

The game hop works completely analogous for $\mathcal{K}_1$ being secure and $\mathcal{K}_2$ insecure, yielding an adversary $\mathcal{B}_3$ against the $\mathsf{IND\text{-}CCA}$ security of $\mathcal{K}_1$.

$Game_6(\lambda)$**:** Next, we replace the value K' in the computations of $\pi^\star$ and $\pi_{\mathsf{a}}^\star$ by a uniformly random value $\widetilde{K'} \xleftarrow{\$} \mathscr{K}'$.

We argue that if an adversary can efficiently distinguish $Game_5(\lambda)$ from $Game_6(\lambda)$, this implies an efficient adversary $\mathcal{B}_4$ against the PRF security of dPRF' and thus contradict the dual-PRF security of dPRF. The reduction works as follows:

To initialize the environment for $\mathcal{A}$, the reduction $\mathcal{B}_4$ generates all long-term key pairs of participants. This enables $\mathcal{B}_4$ to correctly establish all new sessions via $\mathsf{NewSession}$ requests of $\mathcal{A}$ and return sound answers to any $\mathsf{Corrupt}$ queries.

All Send queries can be executed by $\mathcal{B}_4$. Note again that we can assume that $\mathcal{A}$ will not ask a Reveal query on $\pi^\star$ or $\pi_{\mathsf{a}}^\star$ as this would result in a trivial loss.

Once $\mathcal{A}$ asks Test, $\mathcal{B}_4$ queries its PRF oracle $\mathcal{O}_{\mathsf{PRF}}$ on K_1 from $\pi^\star$ and $\pi_{\mathsf{a}}^\star$ to receive a value $\kappa_{b'}^\star$ which is either $\mathsf{dPRF}'(K, K_1) = \mathsf{dPRF}(K_1, K)$ with $K \xleftarrow{\$} \mathscr{K}_2$ chosen by its challenger ($b' = 0$), or a uniformly random element computed as $g(K_1)$ from $\mathscr{K}_2$ ($b' = 1$) in the PRF-security game for dPRF'. $\mathcal{B}_4$ returns the session key K computed as $\mathsf{KDF}(\mathsf{F}(\kappa_{b'}^\star, c), \text{"KE"}||(t, r_A, r_B, A, B)))$ as challenge to $\mathcal{A}$.

At some point, $\mathcal{A}$ terminates with output bit b_{guess}. $\mathcal{B}_4$ outputs the same guess.

Observe that if $\mathcal{B}_4$ receives the real values, i.e., $b' = 0$, then $K_0^\star \leftarrow \mathsf{F}(\mathsf{dPRF}(\widetilde{K_1}, K), c)$ for random K and the situation is as in $Game_5(\lambda)$ since K is distributed as $\widetilde{K_2}$. If on the other hand $\mathcal{B}_4$ receives $\kappa_{b'}^\star \leftarrow g(K_1)$, i.e., if $b' = 1$, then $K_0^\star \leftarrow \mathsf{F}(\kappa_{b'}^\star, c)$ is distributed as in $Game_6(\lambda)$.

Hence, if $\mathcal{A}$ is able to efficiently distinguish the two games, $\mathcal{B}_4$ can win the PRF game against dPRF' and thus the dual-PRF game against dPRF with non-negligible advantage as well.

For $\mathcal{K}_1$ being secure and $\mathcal{K}_2$ insecure, the argument is analogous, as dPRF is a dual PRF, i.e., $\mathsf{dPRF}(K, \cdot)$ is also a pseudorandom function.

We have:

$$\mathsf{Adv}^{\mathsf{G}_5}_{\mathsf{KE},\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{G}_6}_{\mathsf{KE},\mathcal{A}}(\lambda) + \mathsf{Adv}^{\mathsf{dprf\text{-}sec}}_{\mathsf{dPRF},\mathcal{B}_4}(\lambda).$$

As $(\mathsf{dPRF}, \cdot) \notin \mathcal{F}_{\mathsf{BDR}}$, this bound again particularly holds in the sBDR scenario.

Game$_7(\lambda)$**:** We now replace the value $\mathsf{F}(\widetilde{K'}, c^\star)$ by a uniformly random value $\widetilde{K} \xleftarrow{\$} \mathscr{K}$.

We argue that any efficient distinguisher $\mathcal{A}$ between *Game*$_6(\lambda)$ and *Game*$_7(\lambda)$ immediately yields an efficient adversary $\mathcal{B}_5$ against the pseudorandomness of F:

The reduction $\mathcal{B}_5$ works analogously to the reduction $\mathcal{B}_4$. It can faithfully simulate all $\mathsf{NewSession}, \mathsf{Corrupt}, \mathsf{Reveal}$, and $\mathcal{O}^{\mathcal{F}_{\mathsf{BDR}}}_{\mathsf{Decaps}}$ queries. When $\mathcal{A}$ queries Test, $\mathcal{B}_5$ uses its own $\mathcal{O}_{\mathsf{PRF}}$ oracle on the ciphertext of $\pi^\star$ and uses the response $\kappa^\star_{b'}$ to derive the final challenge key for $\mathcal{A}$.

At some point, $\mathcal{A}$ terminates and outputs a guess bit b_{guess}. $\mathcal{B}_4$ then outputs the same guess bit.

We see again that if $b' = 0$, $\mathcal{B}_5$ faithfully simulates *Game*$_6(\lambda)$ and if $b' = 1$ we are in *Game*$_7(\lambda)$. Thus, if $\mathcal{A}$ can distinguish between the two games, $\mathcal{B}_5$ also wins its game with non-negligible advantage.

We have:

$$\mathsf{Adv}^{\mathsf{G}_6}_{\mathsf{KE},\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{G}_7}_{\mathsf{KE},\mathcal{A}}(\lambda) + 2 \cdot \mathsf{Adv}^{\mathsf{prf\text{-}sec}}_{\mathsf{F},\mathcal{B}_5}(\lambda).$$

The analogous argument applies when $\mathcal{K}_1$ is secure. As $(\mathsf{F}, \cdot) \notin \mathcal{F}_{\mathsf{BDR}}$, this bound again particularly holds in the sBDR scenario.

Game$_8(\lambda)$**:** Next, we finally replace the session key K and the MAC key K_{mac} by uniformly random values $\widetilde{\mathsf{K}}$ and $\widetilde{K_{\mathsf{mac}}}$ in $\pi^\star$ and $\pi^\star_{\mathsf{a}}$. Distinguishing *Game*$_7$ and *Game*$_8$ by $\mathcal{A}$ would immediately imply the existence of an efficient adversary $\mathcal{B}_6$ that breaks the pseudorandomness of KDF with non-negligible advantage.

For this, as elaborated in previous games, $\mathcal{B}_6$ simply replaces KDF executions keyed with K by an oracle call in its pseudorandomness game, again simulating one of the two games depending on the oracle response.

Thus, we have

$$\mathsf{Adv}^{\mathsf{G}_7}_{\mathsf{KE},\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{G}_8}_{\mathsf{KE},\mathcal{A}}(\lambda) + 2 \cdot \mathsf{Adv}^{\mathsf{prf\text{-}sec}}_{\mathsf{KDF},\mathcal{B}_6}(\lambda).$$

As $(\mathsf{KDF}, \cdot) \notin \mathcal{F}_{\mathsf{BDR}}$, this bound again particularly holds in the sBDR scenario.

Game$_9(\lambda)$**:** Finally, we abort the game if the associated session $\pi^\star_{\mathsf{a}}$ of party V accepts with a session identifier $\pi^\star_{\mathsf{a}}.\mathsf{sid} \neq \bot$ which does not equal $\pi^\star.\mathsf{sid}$. This can only happen if the adversary is able to make $\pi^\star_{\mathsf{a}}$ obtain a valid signature σ_W and MAC τ_W for some identity $W \neq U$ since all entries except for the peer identity of $\pi^\star_{\mathsf{a}}.\mathsf{sid}$ are already fixed at this point. In particular, while the adversary may be able to sign under a corrupt party's identifier W for which the adversary may know the signing key due to a $\mathsf{Corrupt}$ query, the MAC scheme, on the other hand, must still be secure. Furthermore, the MAC tag depends on the key K_{mac} shared between the honest parties U and V and includes the sender's identity.

Again, the probability of an abort happening for this reason can be bounded by the success probability of an adversary $\mathcal{B}_7$ against the unforgeability of the MAC scheme $\mathcal{M}$. That is, since we have already replaced the key K_{mac} by an independent random value, we can use an external $\mathcal{O}_{\mathsf{MAC}}$ oracle for an unknown key in a simulation instead, and use oracle queries to create the MACs for $\texttt{"b"}||U$ and $\texttt{"b'"}||V$ for $\mathsf{b}' = 1 - \mathsf{b}$ as required in the test session and its associated session. It follows that a valid MAC τ_W for $\texttt{"b"}||W$ created by the adversary for identity $W \neq U$ in the associated session constitutes a successful forgery for a fresh message. We have

$$\mathsf{Adv}^{\mathsf{G}_8}_{\mathsf{KE},\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{G}_9}_{\mathsf{KE},\mathcal{A}}(\lambda) + \mathsf{Adv}^{\mathsf{euf\text{-}cma}}_{\mathcal{M},\mathcal{B}_7}(\lambda).$$

To complete the proof we note that the adversary expects the challenge value K_{test} to be either the output of $\mathsf{KDF}(K, \text{"}\mathsf{KE}\text{"}||(t, r_A, r_B, A, B))$ ($b_{\mathsf{test}} = 0$) or a uniformly random value ($b_{\mathsf{test}} = 1$). At this point, both cases $b_{\mathsf{test}} = 0$ and $b_{\mathsf{test}} = 1$ are indistinguishable for $\mathcal{A}$ since both keys are of equal length and are drawn independently and uniformly at random. Furthermore, the session key in the associated session (which coincides with the now random key K_{test} in case of $b_{\mathsf{test}} = 0$ and is independent of K_{test} for $b_{\mathsf{test}} = 1$) cannot be revealed, because that session is either partnered or held the same contributive identifier upon breakdown. Thus $\mathcal{A}$ cannot learn any information about the bit b_{test}. The only strategy for $\mathcal{A}$ is to guess and thus we have the final bound:

$$\mathsf{Adv}^{\mathsf{G}_9}_{\mathsf{KE},\mathcal{A}}(\lambda) \leq 0.$$

□

In the last two chapters we have seen how key exchanges can be adjusted to handle future failures of cryptographic primitives and hardness assumptions. Key exchanges can be actively built with these failures in mind by employing combiner techniques (cf. Chapter 5) and in this chapter we showed that if a breakdown occurs, we can retrospectively gain assurances about the security of communications that had been established before the fact.

In the next part of the thesis we turn towards a relatively new foundational assumption in key exchange called PRF-ODH. We give a systematic study of the PRF-ODH assumption which underlies today's key exchanges based on Diffie–Hellman. While not actively "future-proofing" the key exchange itself, the rigorous study of new assumptions contributes to the confidence that we place in the analysis of current and future protocols that employ this assumption.

Chapter **7**

PRF-ODH
Relations, Instantiations, and Impossibility Results

Proposing new cryptographic assumptions is a valid strategy to analyze or design protocols which have escaped formal treatment so far. However, the security analysis—usually carried out via a reduction to the new assumption—is only the first step. Only the systematic categorization of the new assumption completes the analysis and yields a meaningful security claim.

In the context of key exchange protocols, a fairly new assumption called the pseudorandom-function oracle Diffie–Hellman ($\mathsf{PRF\text{-}ODH}$) has been put forward by Jager et al. [JKSS12] for the analysis of TLS 1.2. Informally, the $\mathsf{PRF\text{-}ODH}$ assumption says that the function value $\mathsf{PRF}(g^{uv}, x^\star)$ for a DH key g^{uv} looks random, even if given the DH key shares g^u and g^v, and if seeing related values $\mathsf{PRF}(S^u, x)$ and/or $\mathsf{PRF}(T^v, x)$ for chosen values S, T, and x. It is a variant of the oracle Diffie–Hellman assumption introduced by Abdalla et al. [ABR01] in the context of the encryption scheme DHIES, where the hash function is replaced by a pseudorandom function.

$\mathsf{PRF\text{-}ODH}$ appears to be a natural assumption for any DH-based key exchange protocol, aiming at security against man-in-the-middle attacks (see Figure 7.1). In Diffie–Hellman-based protocols both parties, Alice and Bob, exchange key shares g^u and g^v and locally compute the session key by keying a pseudorandom key derivation function with the shared secret g^{uv}, applied to some label x which usually contains (parts of) the transcript. The man-in-the-middle adversary Eve can now try to attack Bob's session key $\mathsf{PRF}(g^{uv}, \cdot)$ by submitting a modified value S instead of g^v to Alice, yielding a related key $\mathsf{PRF}(S^u, \cdot)$ on Alice's side. The $\mathsf{PRF\text{-}ODH}$ assumption now guarantees that Bob's key can still be considered indistinguishable from random for the adversary.

Note that simple authentication of transmissions does not provide a remedy against the just described problem. The adversary could impersonate a different (corrupt) party towards Alice, and only re-use the Diffie–Hellman data, authenticated under the corrupt party's key. Then the Diffie–Hellman keys in the executions would still be non-trivially related. This happens especially if keys are used across multiple sessions. Another problem is that some protocols may derive keys "early", e.g., before applying signatures. Examples for this are deriving keys for encryption of the handshake or the post-handshake authentication mechanism in TLS 1.3 [Res18].

After its introduction in [JKSS12] (and prior to the publication of [BFGJ17a] on which this chapter is based) the $\mathsf{PRF\text{-}ODH}$ assumption has been used to analyze many further key exchange protocols such as [KPW13, BFK+14, DFGS15a, DFGS15b, DFGS16, FG17, BF17].

Notably, these analyses employ different versions of the $\mathsf{PRF\text{-}ODH}$ assumption, due to the different usages of the key shares g^u and g^v. These key shares can be *ephemeral* (used in a single session), *semi-static* (used in a small number of sessions), or *static* (used in multiple sessions). Therefore, the man-in-the middle adversary may ask to see no related key for either key share,

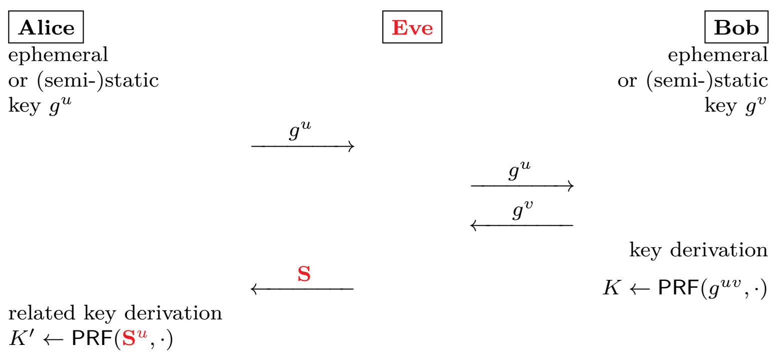

Figure 7.1: Origin of the PRF-ODH assumption: Man-in-the-middle attack on DH-based key exchange protocol.

a single related key, or multiple related keys.

For instance, while Jager et al. [JKSS12] required only security against a single query for one of the two key shares, Krawczyk et al. [KPW13] had to modify the originally proposed PRF-ODH assumption because they require security wrt. multiple oracle queries against one key share. In [FG17] an extra query to the other key share has been added, and [BF17] must ensure security in presence of multiple queries to both key shares.

A unifying definition and a comprehensive treatment of the various PRF-ODH assumptions had however been lacking. Especially the question whether the PRF-ODH assumption (or, rather, which *variant*) can be instantiated in the standard model is of utmost interest. Some of the aforementioned works refer to PRF-ODH as a standard-model assumption, since there is no immediate reference to a random oracle. We will see later that this conclusion may be misguided.

Our Contributions

Our contributions are as follows:

- As our first contribution, in Section 7.2, we give a comprehensive, unified definition of the various PRF-ODH assumptions used in prior works on key exchange analyses. We generally speak of the lrPRF-ODH assumption, where $\mathsf{l}, \mathsf{r} \in \{\mathsf{n}, \mathsf{s}, \mathsf{m}\}$ allowing the adversary no (n), a single (s), or multiple (m) PRF oracle queries under keys related to the "left" key share g^u or the "right" key share g^v. Such queries are handled by oracles ODH_u and ODH_v, returning the corresponding pseudorandom function value.

- In order to support a better comparison between the various notions, in Section 7.3, we relate the different flavors of lrPRF-ODH in terms of strength of the underlying assumption.

- In Section 7.4, we discuss how the different flavors of PRF-ODH can be instantiated from various Diffie–Hellman assumptions and thus provide a systematic categorization into the realm of well-understood hardness assumptions based on Diffie–Hellman.

- Since PRF-ODH has been used in connection with applied protocols like TLS, we finally address the question which security guarantees we get for practical key derivation functions used in such protocols. We are especially interested in the security of HMAC [KBC97] which is the basis of the key derivation function HKDF [Kra10, KE10] that is, for example, used in TLS 1.3 to derive keys.

We show in Section 7.5 that HMAC achieves the strongest notion of mmPRF-ODH security under the strong Diffie–Hellman assumption StDH and assuming that the compression function is a random oracle.

- In Section 7.6, we briefly sketch our impossibility result which argues that it is implausible to instantiate even the mildest one-sided PRF-ODH assumptions in the standard model.

Personal contributions in this section. All the material from this chapter appeared in the joint work with Marc Fischlin, Felix Günther, and Christian Janson with the title *PRF-ODH: Relations, Instantiations, and Impossibility Results* [BFGJ17a]. The full version of the paper can be found on ePrint [BFGJ17b]. My main contributions lie in the development of the generalized notion of lrPRF-ODH, the instantiation proofs, as well as the proof of the PRF-ODH security of HMAC.

7.1 Related Work

Here, we briefly discuss the origins of PRF-ODH and show where PRF-ODH has been employed in previous works and where it has been adopted since the publication of [BFGJ17a] in 2017. Furthermore, we briefly discuss the relationship between PRF-ODH and *related-key security* for pseudorandom functions as introduced by Bellare and Kohno [BK03], where the adversary can ask to see PRF values for transformed keys $\phi(K)$. While similar in spirit at first glance, it seems to us that the notions differ in technical details which makes it hard to relate them.

Oracle Diffie–Hellman assumption. In [ABR01], Abdalla, Bellare, and Rogaway introduced the oracle Diffie–Hellman (ODH) assumption to show security of their public key encryption scheme DHIES. Let H be a cryptographic hash function. Informally, the ODH assumption asks an adversary $\mathcal{A}$ to distinguish $\mathsf{H}(g^{uv})$ from a random string of the same length when given g^u and g^v as well as access to an oracle that, on input X, returns $\mathsf{H}(X^v)$.

Usages of (lr)PRF-ODH. As mentioned before, PRF-ODH is a natural assumption for DH-based key exchanges and has thus, unsurprisingly, been used in analyses of different protocols. These include the analysis of the TLS 1.2 [DR08] ephemeral and static Diffie–Hellman handshake modes [JKSS12, KPW13, BFK+14], the Diffie–Hellman-based and resumption handshake candidates [DFGS15a, DFGS15b, DFGS16] of TLS 1.3 [Res18], as well as a former TLS 1.3 0-RTT handshake candidate [FG17], and a 0-RTT extension of the Extended Access Control (EAC) protocol [BF17]. Since the systematic introduction of lrPRF-ODH in [BFGJ17a], the unified assumption has further proven useful in the realm of secure messaging. The assumption has been employed in the (full version) analysis of the Signal messaging protocol [CCD+16, CCD+17] as well as the Asynchronous Ratcheting Tree (ART) protocol for secure group messaging [CCG+18]. Further real-world key exchange protocol analyses using lrPRF-ODH include the security analysis of Wireguard [DP18] and the Noise framework [DRS19].

Related-key attacks. PRF-ODH requires the output of a PRF keyed with a DH value to be indistinguishable from random even when given access to PRF evaluations keyed with related group elements that share (at least) one exponent with the challenge key. On a high level, this setting resembles the concept of *related-key attack (RKA) security* for pseudorandom functions as introduced by Bellare and Kohno [BK03]. This resemblance raises the question whether the PRF-ODH assumption can be instantiated from RKA-secure PRFs (or vice versa).

Related-key attack security of a PRF $f\colon \mathcal{K} \times \mathcal{D} \to \mathcal{R}$ with respect to a set Φ of related-key-deriving (RKD) functions is defined as the indistinguishability of two oracles $F_{(\cdot,K)}(\cdot)$ and $G_{(\cdot,K)}(\cdot)$ for a randomly chosen key $K \xleftarrow{\$} \mathcal{K}$. The distinguishing adversary $\mathcal{A}$ may query the oracles on inputs $(\phi, x) \in \Phi \times \mathcal{D}$ to which the oracles respond with $F_{(\phi,K)}(x) := f(\phi(K), x)$ and $G_{(\phi,K)}(x) := g(\phi(K), x)$ for a function g drawn uniformly at random from the set $\mathcal{FF}(\mathcal{K}, \mathcal{D}, \mathcal{R})$ of all functions $\mathcal{K} \times \mathcal{D} \to \mathcal{R}$. Formally, the advantage of $\mathcal{A}$ against the RKA-PRF security of f with respect to the set Φ is defined as

$$\mathsf{Adv}^{\mathsf{RKA\text{-}PRF},\Phi}_{f,\mathcal{A}}(\lambda) := \left| \Pr\left[\mathcal{A}^{F_{(\cdot,K)}} = 1 \mid K \xleftarrow{\$} \mathcal{K}\right] - \Pr\left[\mathcal{A}^{G_{(\cdot,K)}} = 1 \mid K \xleftarrow{\$} \mathcal{K}, g \xleftarrow{\$} \mathcal{FF}(\mathcal{K}, \mathcal{D}, \mathcal{R})\right] \right|.$$

Intuitively, one should now be able to relate $\mathsf{RKA\text{-}PRF}$ security to $\mathsf{PRF\text{-}ODH}$ security by considering two correlated sets of RKD functions corresponding to the $\mathsf{PRF\text{-}ODH}$ oracles ODH_u and ODH_v with respect to a group $\mathbb{G}$ with generator g and two exponents $u, v \in \mathbb{Z}_q$:

$$\Phi_{\mathsf{ODH}_u} := \{\phi_{\mathsf{ODH}_u,S} \mid S \in \mathbb{G} \setminus \{g^v\}\} \qquad \text{where } \phi_{\mathsf{ODH}_u,S}(K) := (K^{1/v})^{\log_g(S)},$$
$$\Phi_{\mathsf{ODH}_v} := \{\phi_{\mathsf{ODH}_v,T} \mid T \in \mathbb{G} \setminus \{g^u\}\} \qquad \text{where } \phi_{\mathsf{ODH}_v,T}(K) := (K^{1/u})^{\log_g(T)}.$$

Insurmountable hurdles however seem to remain when trying to relate $\mathsf{PRF\text{-}ODH}$ notions and RKA-PRF security (for according sets Φ) via implications.

In the one direction, the adversary in the $\mathsf{PRF\text{-}ODH}$ setting is provided with the DH shares g^u and g^v forming the (challenge) PRF key while such side information on the key is not given in the RKA-PRF setting. Hence, in a reduction of $\mathsf{PRF\text{-}ODH}$ security to some RKA-PRF notion, even for an appropriate RKD function set a simulation always lacks the means to provide the $\mathsf{PRF\text{-}ODH}$ adversary with these shares.

In the other direction, the RKA-PRF challenge can be issued on any related key $\phi(K)$ for an admissible RKD function ϕ while the $\mathsf{PRF\text{-}ODH}$ challenge is, for the case of the real PRF response, always computed on the key g^{uv}. A reduction would hence need to map the RKA-PRF challenge for an arbitrary, related key onto the fixed $\mathsf{PRF\text{-}ODH}$ challenge key.

Though on a high level capturing a relatively similar idea, the exact relation between $\mathsf{PRF\text{-}ODH}$ and RKA-PRF security hence remains an open question.

7.2 Modeling PRF Security under Related DH Keys

In this section we provide a unifying definition of the $\mathsf{PRF\text{-}ODH}$ assumption that captures variants from the literature and discuss its relation to these previous occurrences [JKSS12, KPW13, DFGS15b, DFGS16, BF17, FG17]. Furthermore, we provide a *symmetric* variant of this generic definition that has been employed in key exchange analyses following [BFGJ17a].

Definition 7.1 (Generic $\mathsf{PRF\text{-}ODH}$ assumption)**.** *Let $\mathbb{G} = \mathbb{G}_\lambda$ be a cyclic group of order q with generator g whose choice depends on the security parameter λ. Let $\mathsf{F}\colon \mathbb{G} \times \{0,1\}^\star \to \{0,1\}^\lambda$ be a pseudorandom function keyed with $K \in \mathbb{G}$ and with an input label $x \in \{0,1\}^\star$ that outputs a value $y \in \{0,1\}^\lambda$, i.e., $y \leftarrow \mathsf{F}(K, x)$.*

The generic security game for $\mathsf{lrPRF\text{-}ODH}$ *(cf. Figure 7.2) is parametrized by* $\mathsf{l}, \mathsf{r} \in \{\mathsf{n}, \mathsf{s}, \mathsf{m}\}$ *indicating how often the adversary is allowed to query a certain "left" resp. "right" oracle (*ODH_u *resp.* ODH_v*), where* n *means that no query is allowed,* s *that a single query is allowed, and* m *that multiple, i.e., polynomially many, queries are allowed to the respective oracle.*

We say that a pseudorandom function F *with keys from* $\mathbb{G} = \mathbb{G}_\lambda$ *provides* $\mathsf{lrPRF\text{-}ODH}$ *security (for* $\mathsf{l}, \mathsf{r} \in \{\mathsf{n}, \mathsf{s}, \mathsf{m}\}$*) if for any PPT* $\mathcal{A}$ *the advantage defined as*

$$\mathsf{Adv}^{\mathsf{lrPRF\text{-}ODH}}_{\mathbb{G},\mathsf{F},\mathcal{A}}(\lambda) := \left| \Pr\left[\mathsf{G}^{\mathsf{lrPRF\text{-}ODH}}_{\mathbb{G},\mathsf{F},\mathcal{A}}(\lambda) = 1\right] - \frac{1}{2} \right|$$

is negligible in the security parameter λ*.*

```
$\mathsf{G}^{\mathsf{lrPRF\text{-}ODH}}_{\mathbb{G},\mathsf{F},\mathcal{A}}(\lambda)$:
1  $u \xleftarrow{\$} \mathbb{Z}_q$
2  $n_l, n_r \leftarrow 0$
3  $Q \leftarrow \emptyset$
4  $\mathsf{afterchall} \leftarrow \mathsf{false}$
5  $(x^\star, st) \xleftarrow{\$} \mathcal{A}^{\mathsf{ODH}_u}(\mathbb{G}, g, g^u)$
6  $v \xleftarrow{\$} \mathbb{Z}_q$
7  $b \xleftarrow{\$} \{0,1\}$
8  $y_0^\star \leftarrow \mathsf{F}(g^{uv}, x^\star)$
9  $y_1^\star \xleftarrow{\$} \{0,1\}^\lambda$
10 $\mathsf{afterchall} \leftarrow \mathsf{true}$
11 $b' \xleftarrow{\$} \mathcal{A}^{\mathsf{ODH}_u,\mathsf{ODH}_v}(g^v, y_b^\star, st)$
12 if $x^\star \in Q$
13     $b' \xleftarrow{\$} \{0,1\}$
14 return $[\![b' = b]\!]$

$\mathsf{ODH}_u(S, x)$:
15 if $n_l \geq \mathsf{l}$ or $S \notin \mathbb{G}$
16     return $\bot$
17 if $\mathsf{l} = \mathsf{s}$ and $\mathsf{afterchall} = \mathsf{false}$
18     return $\bot$
19 else
20     $n_l \leftarrow n_l + 1$
21     $y \leftarrow \mathsf{F}(S^u, x)$
22 if $S = g^v$
23     $Q \leftarrow Q \cup \{x\}$
24 return $y$

$\mathsf{ODH}_v(S, x)$:
25 if $n_r \geq \mathsf{r}$ or $S \notin \mathbb{G}$
26     return $\bot$
27 else
28     $n_r \leftarrow n_r + 1$
29     $y \leftarrow \mathsf{F}(S^v, x)$
30 if $S = g^u$
31     $Q \leftarrow Q \cup \{x\}$
32 return $y$
```

Figure 7.2: PRF-ODH security of pseudorandom function F. In numerical evaluations of l and r we naturally define $\mathsf{n} = 0$, $\mathsf{s} = 1$, and $\mathsf{m} = \infty$.

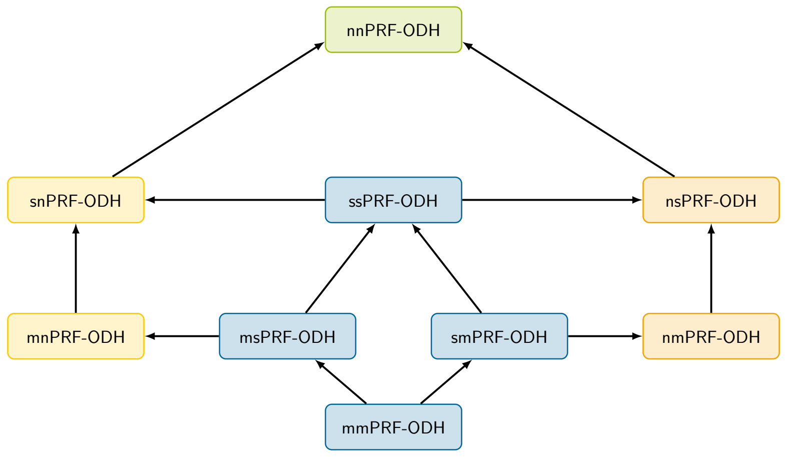

Figure 7.3: Different PRF-ODH variants from Definition 7.1. Solid arrows indicate the trivial implications between PRF-ODH variants.

In the following, if clear from the context, we will omit the group $\mathbb{G}$ and its generator g as explicit inputs to the adversary.

Relations to PRF-ODH assumptions in the literature. This generic and parametrized lrPRF-ODH definition gives rise to nine different notions (depicted in Figure 7.3) and captures different variants of the PRF-ODH assumption present in the literature. The PRF-ODH formulation put forward by Jager et al. [JKSS12] is captured by ours when setting the parameters to $\mathsf{l} = \mathsf{s}$ and $\mathsf{r} = \mathsf{n}$, meaning that the “left” oracle (i.e., the DH share g^u) can be queried only once. Note that the access to ODH_u in Line 5 is only given if $\mathsf{l} = \mathsf{m}$, capturing that Jager et al. first request their challenge before issuing an oracle query. The administrative flag afterchall that is set to true once the challenge has been generated ensures this. The same variant, snPRF-ODH, was also used by Dowling et al. [DFGS16].

Krawczyk et al. [KPW13] modified the PRF-ODH formulation of Jager et al. since they require security wrt. multiple “left” oracle queries against the respective DH key share. Thus, their variant is captured by ours through setting the parameters to $\mathsf{l} = \mathsf{m}$ and $\mathsf{r} = \mathsf{n}$, and thus making use of Line 5.

Other works further introduced an additional query to the other DH key share, due to the fact that these key shares are static or semi-static, i.e., used across multiple sessions. In more detail, Fischlin and Günther [FG17] require an extra single "right" oracle query while still requesting polynomially many queries to the "left" oracle. This is captured by our definition when setting the parameters to $\mathsf{l} = \mathsf{m}$ and $\mathsf{r} = \mathsf{s}$. Lastly, Brendel and Fischlin [BF17] for their 0-RTT extension of the Extended Access Control (EAC) protocol require to query both key shares multiple times, which our definition captures as well by choosing the parameters as $\mathsf{l} = \mathsf{m}$ and $\mathsf{r} = \mathsf{m}$. Due to the different ways in which ephemeral, semi-static, and long-term keys are combined, both the analysis of the Signal protocol by Cohn-Gordon et al. [CCD+16], as well as the analysis of the Noise framework by Dowling, Rösler, and Schwenk [DRS19] require a multitude of the above mentioned assumptions. These analyses, as well as the analysis of earlier TLS 1.3 draft handshakes by Dowling et al. [DFGS15b] and the Wireguard analysis by Dowling and Paterson [DP18] require notions of PRF-ODH in which the challenger provides the value g^v to the adversary at the outset in Line 5.

Such change is accompanied by giving the adversary in Line 5 of Figure 7.2 also access to the ODH_v oracle in case $\mathsf{r} = \mathsf{m}$. The respective game $\mathsf{G}^{\mathsf{sym\text{-}lrPRF\text{-}ODH}}_{\mathbb{G},\mathsf{F},\mathcal{A}}(\lambda)$ is depicted in Figure 7.4. Note that this definition does not capture Dowling et al.'s interpretation of PRF-ODH [DFGS15b], as they only allow oracle accesses after the challenge has been generated.

```
$\mathsf{G}^{\mathsf{sym\text{-}lrPRF\text{-}ODH}}_{\mathbb{G},\mathsf{F},\mathcal{A}}(\lambda)$:
1  $u, v \xleftarrow{\$} \mathbb{Z}_q$
2  $n_l, n_r \leftarrow 0,\ Q \leftarrow \emptyset$
3  $\mathsf{afterchall} \leftarrow \mathsf{false}$
4  $(x^\star, st) \xleftarrow{\$} \mathcal{A}^{\mathsf{ODH}_u, \mathsf{ODH}_v}(\mathbb{G}, g, g^u, g^v)$
5  $b \xleftarrow{\$} \{0,1\}$
6  $y_0^\star \leftarrow \mathsf{F}(g^{uv}, x^\star)$
7  $y_1^\star \xleftarrow{\$} \{0,1\}^\lambda$
8  $\mathsf{afterchall} \leftarrow \mathsf{true}$
9  $b' \xleftarrow{\$} \mathcal{A}^{\mathsf{ODH}_u, \mathsf{ODH}_v}(y_b^\star, st)$
10 if $x^\star \in Q$
11     $b' \xleftarrow{\$} \{0,1\}$
12 return $[\![b' = b]\!]$

$\mathsf{ODH}_u(S, x)$:
13 if $n_l \geq \mathsf{l}$ or $S \notin \mathbb{G}$
14     return $\perp$
15 if $\mathsf{l} = \mathsf{s}$ and $\mathsf{afterchall} = \mathsf{false}$
16     return $\perp$
17 else
18     $n_l \leftarrow n_l + 1$
19     $y \leftarrow \mathsf{F}(S^u, x)$
20 if $S = g^v$
21     $Q \leftarrow Q \cup \{x\}$
22 return $y$

$\mathsf{ODH}_v(S, x)$:
23 if $n_r \geq \mathsf{r}$ or $S \notin \mathbb{G}$
24     return $\perp$
25 if $\mathsf{r} = \mathsf{s}$ and $\mathsf{afterchall} = \mathsf{false}$
26     return $\perp$
27 else
28     $n_r \leftarrow n_r + 1$
29     $y \leftarrow \mathsf{F}(S^v, x)$
30 if $S = g^u$
31     $Q \leftarrow Q \cup \{x\}$
32 return $y$
```

Figure 7.4: Symmetric PRF-ODH security of pseudorandom function F. In numerical evaluations of l and r we naturally define $\mathsf{n} = 0$, $\mathsf{s} = 1$, and $\mathsf{m} = \infty$.

In this work, we focus on the common structure of studied PRF-ODH notions [JKSS12, KPW13, DFGS16, BF17, FG17] that lead to the unifying notion from [BFGJ17a] and are captured by Definition 7.1 above. However, in Section 7.3 we briefly discuss the impact of the above mentioned changes regarding the analysis of the relations between the different variants of the assumption.

Remark 7.2. If we provide the adversary additionally with the share g^v in the initialization phase, then Figure 7.6 symmetrically "collapses" along the central vertical axis, resulting in Figure 7.5. In other words, this results in equivalences of the notions snPRF-ODH and nsPRF-ODH, mnPRF-ODH and nmPRF-ODH, as well as msPRF-ODH and smPRF-ODH. We already note that this is not a contradiction to our separation results among those notions (given in the next Section 7.3), as they rely on the fact that g^v is not given in advance.

Further design options. Another reasonable change could encompass enabling the adversary in multi-query variants (i.e., where $\mathsf{l} = \mathsf{m}$ or $\mathsf{r} = \mathsf{m}$) to also issue multiple challenge queries, for the same value g^v or even freshly chosen values g^{v_i} in each call. However, one can show via a

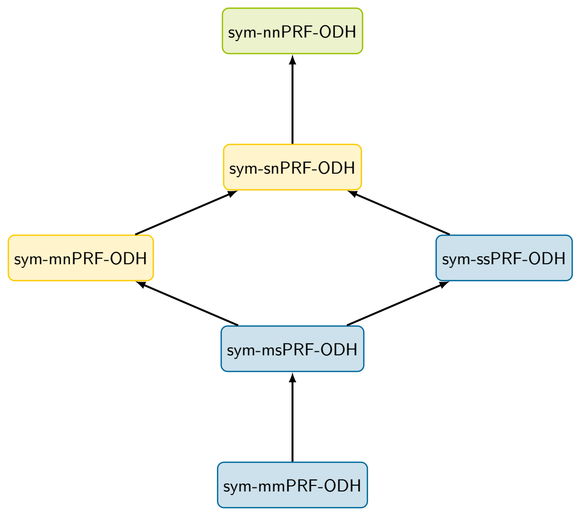

Figure 7.5: Notions of the symmetric variant sym-lrPRF-ODH with trivial implications indicated by solid arrows.

standard hybrid argument that both notions (i.e., single challenge query and multiple challenge query) are polynomially equivalent.

7.3 Relations

In this section we study the relations of different PRF-ODH variants spanned by our generic Definition 7.1. An overview of the results is illustrated in Figure 7.6.

Let us start with observing the trivial implications (indicated by solid arrows in Figure 7.6) which are induced by restricting the adversary's capabilities in our definition. That is, by restricting the access to one of the oracles ODH_u or ODH_v (from multiple queries to a single query or from a single query to no query) for a notion from Definition 7.1, we obtain a trivially weaker variant. The more interesting question is which of these implications are strict, i.e., which of two given PRF-ODH notions is *strictly* stronger than the other. For most cases we can give separations which only require that the underlying primitive exists at all; for some separations we rely on the random oracle model and a plausible number-theoretic DH-type assumption. We begin with the standard model separations.

7.3.1 Separations in the Standard Model

Our standard model separations rely on the following family of functions $\mathcal{F}$:

Definition 7.3 (Separating function family $\mathcal{F}$)**.** *Let* $\mathsf{G}\colon \mathbb{G} \times \{0,1\}^\star \to \{0,1\}^\lambda$*. We define the family of functions* $\mathcal{F} = \{\mathsf{F}_n\}_{n\in\mathbb{N}}$ *with* $\mathsf{F}_n\colon \mathbb{G} \times \{0,1\}^\star \to \{0,1\}^\lambda$ *as follows:*

$$\mathsf{F}_n(K,x) := \begin{cases} \mathsf{G}(K,1) \oplus \ldots \oplus \mathsf{G}(K,n) & \textit{if } x = 0 \\ \mathsf{G}(K,x) & \textit{otherwise.} \end{cases}$$

To capture the main idea behind the standard model separations, let us first consider the (in)security of functions $\mathsf{F}_n \in \mathcal{F}$ in the standard PRF setting. It is easy to see that no function $\mathsf{F}_n \in \mathcal{F}$ can satisfy the usual security notion for pseudorandom functions as given in Definition 3.13: For any function F_n, querying the PRF oracle on $x_0 = 0, \ldots, x_n = n$ yields responses $y_0, \ldots, y_n$ for which the combined XOR value $y = y_0 \oplus \ldots \oplus y_n$ is always 0 in case

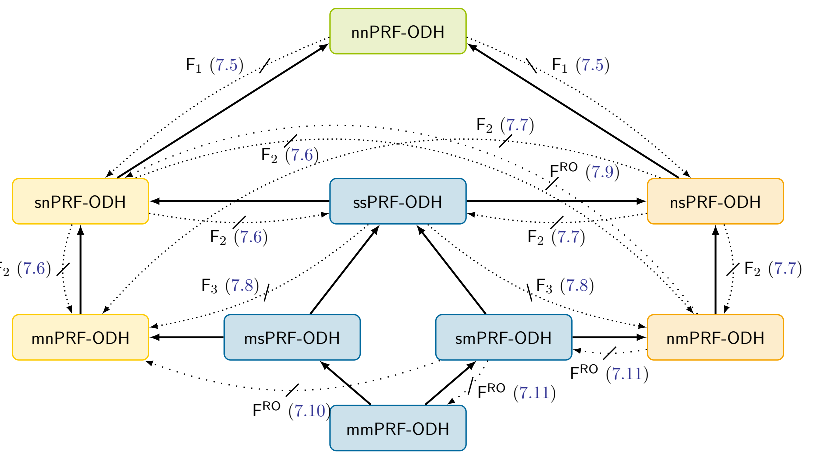

Figure 7.6: Relations between the different $\mathsf{PRF\text{-}ODH}$ variants. Solid arrows indicate the trivial implications between $\mathsf{PRF\text{-}ODH}$ variants, dashed arrows indicate implications we establish. Struck-out, densely dotted arrows indicate separations in the standard model via the indicated function $\mathsf{F}_n \in \mathcal{F}$ (cf. Definition 7.3). Struck-out, sparsely dotted arrows indicated separations in the random-oracle model. Numbers in parentheses indicate the respective propositions and theorems.

the oracle computes the real function F_n, whereas otherwise it is 0 only with probability $2^{-\lambda}$. However, in a setting where the PRF adversary $\mathcal{A}$ is allowed to query the oracle only a limited number of times (at most n queries for function F_n), we can indeed establish the following, restricted PRF security for functions $\mathsf{F}_n \in \mathcal{F}$.

Proposition 7.4 ($\mathcal{F}$ is restricted-PRF secure)**.** *If* G *is a pseudorandom function, then each* $\mathsf{F}_n \in \mathcal{F}$ *from Definition 7.3 is an* n-restricted pseudorandom function *in the sense that it provides PRF security wrt. Definition 3.13 against any adversary that is allowed to query the PRF oracle* $\mathcal{O}_{\mathsf{PRF}}$ *at most* n *times.*

Proof (informal). Let $\mathsf{F}_n \in \mathcal{F}$ be an arbitrary but fixed function from $\mathcal{F}$.

Game$_0(\lambda)$**:** The original $\mathsf{G}^{\mathsf{r\text{-}prf\text{-}sec}}_{\mathsf{F}_n,\mathcal{A}}(\lambda)$ restricted-PRF game, which is the PRF game where the adversary trivially loses the game when making more than n queries to the oracle $\mathcal{O}_{\mathsf{PRF}}$.

Game$_1(\lambda)$**:** We replace the pseudorandom function G in the definition of F_n by a truly random function G'. We argue that any adversary $\mathcal{A}$ that can efficiently distinguish *Game*$_0(\lambda)$ from *Game*$_1(\lambda)$ yields an efficient adversary $\mathcal{B}$ against the pseudorandomness of G.

The reduction $\mathcal{B}$ against the PRF security of G simply initiates $\mathcal{A}$ and faithfully simulates the (restricted) PRF game for $\mathcal{A}$ by relaying all of $\mathcal{A}$'s queries to its own PRF oracle for G, also relaying the oracle responses.

After this change, the output values of F_n on inputs $x > 0$ are independent random values and the output on $x = 0$ is the XOR of the outputs on $x = 1, \dots, n$. In contrast, for a truly random function, the outputs on all inputs (incl. $x = 0$) are independent and random. However, any adversary $\mathcal{A}$ that is allowed to query the PRF oracle on at most n inputs cannot distinguish these two cases, thus its advantage at this point is bounded by 0. □

Let us now turn to the more involved $\mathsf{PRF\text{-}ODH}$ setting. Equipped with the function family $\mathcal{F}$, we can establish separations between various $\mathsf{PRF\text{-}ODH}$ variants, as illustrated in Figure 7.6. The key insight for these separations is similar to the one in the standard PRF setting:

An adversary with a limited number of n queries (including the challenge query in the $\mathsf{PRF\text{-}ODH}$ setting) cannot distinguish F_n from a truly random function.

This allows us to establish the following separations:

- $\mathsf{nnPRF\text{-}ODH}$ (with single challenge query) from $\mathsf{snPRF\text{-}ODH}$ and $\mathsf{nsPRF\text{-}ODH}$ (with two queries: one challenge query and one to one of the ODH oracles) via function F_1 (Proposition 7.5).
- $\mathsf{snPRF\text{-}ODH}$ and $\mathsf{nsPRF\text{-}ODH}$ (with two queries) from $\mathsf{mnPRF\text{-}ODH}$, $\mathsf{ssPRF\text{-}ODH}$, and $\mathsf{nmPRF\text{-}ODH}$ (with three or polynomially many queries) via F_2 (Proposition 7.6).
- $\mathsf{ssPRF\text{-}ODH}$ (three queries) from $\mathsf{mnPRF\text{-}ODH}$ and $\mathsf{nmPRF\text{-}ODH}$ (with multiple queries to one of the ODH oracles) using function F_3 (Proposition 7.8).

Note that functions $\mathsf{F}_n \in \mathcal{F}$ cannot provide separations between two notions that both allow polynomially many queries (for example between $\mathsf{mnPRF\text{-}ODH}$ and $\mathsf{msPRF\text{-}ODH}$). To keep the propositions compact, the given separations constitute the minimal spanning set; recall that if a notion A implies another notion B, separating a notion C from B also separates C from A.

Proposition 7.5 ($\mathsf{nnPRF\text{-}ODH} \not\Longrightarrow \mathsf{snPRF\text{-}ODH}, \mathsf{nsPRF\text{-}ODH}$)**.**
If G *from Definition 7.3 is* $\mathsf{nnPRF\text{-}ODH}$ *secure, then* $\mathsf{F}_1 \in \mathcal{F}$ *is* $\mathsf{nnPRF\text{-}ODH}$ *secure, but neither* $\mathsf{snPRF\text{-}ODH}$ *nor* $\mathsf{snPRF\text{-}ODH}$ *secure. More precisely, for any efficient adversary* $\mathcal{A}$ *against the* $\mathsf{nnPRF\text{-}ODH}$ *security of* F_1*, there exists an efficient algorithm* $\mathcal{B}$ *such that*

$$\mathsf{Adv}^{\mathsf{nnPRF\text{-}ODH}}_{\mathbb{G},\mathsf{F}_1,\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{nnPRF\text{-}ODH}}_{\mathbb{G},\mathsf{G},\mathcal{B}}(\lambda),$$

but there exist efficient algorithms $\mathcal{A}_1$ *and* $\mathcal{A}_2$ *with non-negligible advantage*

$$\mathsf{Adv}^{\mathsf{snPRF\text{-}ODH}}_{\mathbb{G},\mathsf{F}_1,\mathcal{A}_1}(\lambda) = \mathsf{Adv}^{\mathsf{nsPRF\text{-}ODH}}_{\mathbb{G},\mathsf{F}_1,\mathcal{A}_2}(\lambda) = 1 - 2^{-\lambda}.$$

Proof. We first show that the function F_1 does not provide security in the $\mathsf{PRF\text{-}ODH}$ sense if a single query to either oracle is allowed. However, as we will show then, if the underlying function G is $\mathsf{nnPRF\text{-}ODH}$ secure, then so is F_1.

<u>F_1 is not $\mathsf{snPRF\text{-}ODH}$ or $\mathsf{nsPRF\text{-}ODH}$ secure.</u> First, we argue that the $\mathsf{snPRF\text{-}ODH}$ adversary $\mathcal{A}_1$ and $\mathsf{nsPRF\text{-}ODH}$ adversary $\mathcal{A}_2$ are successful (except with negligible probability): Both first challenge F_1 on $x^\star = 0$ (obtaining as $y_b^\star$ either $y_0^\star \leftarrow \mathsf{G}(g^{uv}, 1)$ or $y_1^\star \xleftarrow{\$} \{0,1\}^\lambda$). Next, they query $(g^v, 1)$ to their ODH_u or $(g^u, 1)$ to their ODH_v oracle respectively, obtaining the value $y \leftarrow \mathsf{G}(g^{uv}, 1)$. They distinguish the challenge by outputting 0 if $y_b^\star = y$ and 1 otherwise and win except if coincidentally $y_1^\star = y$, which happens with probability at most $2^{-\lambda}$.

<u>F_1 is $\mathsf{nnPRF\text{-}ODH}$ if G is $\mathsf{nnPRF\text{-}ODH}$ secure.</u> To see that F_1 is $\mathsf{nnPRF\text{-}ODH}$ secure if G is, consider a reduction $\mathcal{B}$ that simply relays its obtained value g^u to $\mathcal{A}$. Furthermore, it relays the unmodified challenge query of $\mathcal{A}$ to its challenger, if $x^\star \neq 0$. If $x^\star = 0$, $\mathcal{B}$ asks its challenger on 1. Forwarding the response and outputting the same bit b' as $\mathcal{A}$ outputs, $\mathcal{B}$ provides a correct simulation for $\mathcal{A}$ and, moreover, wins if $\mathcal{A}$ does.

□

We continue with the separation of $\mathsf{snPRF\text{-}ODH}$ from $\mathsf{mnPRF\text{-}ODH}$, $\mathsf{ssPRF\text{-}ODH}$, and $\mathsf{nmPRF\text{-}ODH}$ security.

Proposition 7.6 ($\mathsf{snPRF\text{-}ODH} \not\Longrightarrow \mathsf{mnPRF\text{-}ODH}, \mathsf{ssPRF\text{-}ODH}, \mathsf{nmPRF\text{-}ODH}$)**.**
If G *from Definition 7.3 is* mnPRF-ODH *secure, then* $\mathsf{F}_2 \in \mathcal{F}$ *is* snPRF-ODH *secure, but neither* mnPRF-ODH*, nor* ssPRF-ODH*, nor* nmPRF-ODH *secure. More precisely, for any efficient adversary* $\mathcal{A}$ *against the* snPRF-ODH *security of* F_2*, there exist efficient algorithms* $\mathcal{B}_1$*,* $\mathcal{B}_2$*,* $\mathcal{B}_3$*, and* $\mathcal{B}_4$ *such that*

$$\begin{aligned}\mathsf{Adv}^{\mathsf{snPRF\text{-}ODH}}_{\mathbb{G},\mathsf{F}_2,\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{mnPRF\text{-}ODH}}_{\mathbb{G},\mathsf{G},\mathcal{B}_1}(\lambda) &+ 4 \cdot \mathsf{Adv}^{\mathsf{mnPRF\text{-}ODH}}_{\mathbb{G},\mathsf{G},\mathcal{B}_2}(\lambda) \\ &+ 4 \cdot \mathsf{Adv}^{\mathsf{mnPRF\text{-}ODH}}_{\mathbb{G},\mathsf{G},\mathcal{B}_3}(\lambda) + \mathsf{Adv}^{\mathsf{mnPRF\text{-}ODH}}_{\mathbb{G},\mathsf{G},\mathcal{B}_4}(\lambda),\end{aligned}$$

but there exist efficient algorithms $\mathcal{A}_1$*,* $\mathcal{A}_2$*, and* $\mathcal{A}_3$ *with non-negligible advantage*

$$\mathsf{Adv}^{\mathsf{mnPRF\text{-}ODH}}_{\mathbb{G},\mathsf{F}_2,\mathcal{A}_1}(\lambda) = \mathsf{Adv}^{\mathsf{ssPRF\text{-}ODH}}_{\mathbb{G},\mathsf{F}_2,\mathcal{A}_2}(\lambda) = \mathsf{Adv}^{\mathsf{nmPRF\text{-}ODH}}_{\mathbb{G},\mathsf{F}_2,\mathcal{A}_3}(\lambda) = 1 - 2^{-\lambda}.$$

Proof. We first show that F_2 is not mnPRF-ODH, ssPRF-ODH, or nmPRF-ODH secure. Afterwards we show that F_2 is however snPRF-ODH secure if G is mnPRF-ODH secure.

F_2 is not mnPRF-ODH, ssPRF-ODH, or nmPRF-ODH secure. For this, consider the respective adversaries $\mathcal{A}_1$, $\mathcal{A}_2$, $\mathcal{A}_3$ which first query the challenge $x^\star = 0$, obtaining, besides, g, g^v, a value $y_b^\star$ which equals either $y_0^\star \leftarrow \mathsf{F}_2(g^{uv}, 0) = \mathsf{G}(g^{uv}, 1) \oplus \mathsf{G}(g^{uv}, 2)$ if $b = 0$ or $y_1^\star \xleftarrow{\$} \{0,1\}^\star$ if $b = 1$.

Then the adversaries make the following queries to their oracles, thereby obtaining values $y' \leftarrow \mathsf{G}(g^{uv}, 1)$ and $y'' \leftarrow \mathsf{G}(g^{uv}, 2)$.

- $\mathcal{A}_1$ issues two queries $(g^v, 1)$ and $(g^v, 2)$ to its ODH_u oracle.
- $\mathcal{A}_2$ issues $(g^v, 1)$ to its ODH_u oracle and $(g^u, 2)$ to its ODH_v oracle.
- $\mathcal{A}_3$ issues queries $(g^u, 1)$ and $(g^u, 2)$ to its ODH_v oracle.

The adversaries then check whether $y_b^\star = y' \oplus y''$. If so, they output 0, else 1. Hence, $\mathcal{A}_1$, $\mathcal{A}_2$, and $\mathcal{A}_3$ always win the mnPRF-ODH, ssPRF-ODH, resp. nmPRF-ODH game for F_2, except when $y_1^\star = y' \oplus y''$, which happens with negligible probability $2^{-\lambda}$.

F_2 is snPRF-ODH secure if G is mnPRF-ODH secure. For this, we separately consider the two distinct cases that $\mathcal{A}$ issues a challenge query for $x^\star > n$ and the case that $x^\star \leq n$.

In case $x^\star > n$, we can bound $\mathcal{A}$'s advantage, denoted as $\mathsf{Adv}^{\mathsf{snPRF\text{-}ODH}, x^\star > n}_{\mathsf{F}_2,\mathcal{A}}(\lambda)$, by the advantage of an adversary $\mathcal{B}_1$ against the mnPRF-ODH security of G as follows.

Algorithm $\mathcal{B}_1$ provides its initially obtained values g, g^u as the initial input to $\mathcal{A}$. When $\mathcal{A}$ queries $x^\star$, $\mathcal{B}_1$ relays the challenge to its own challenge oracle, and forwards the obtained response $(g^v, y_b^\star)$ back to $\mathcal{A}$. When $\mathcal{A}$ issues its (sole) ODH_u query (S, x), $\mathcal{B}_1$ simply relays query and response in case $x \neq 0$. In case $x = 0$, $\mathcal{B}_1$ instead queries its ODH_u oracle on $(S, 1)$ and $(S, 2)$ and returns the combined XOR to $\mathcal{A}$. Finally, when $\mathcal{A}$ outputs its guess b', $\mathcal{B}_1$ stops and outputs the same guess b'.

Observe that $\mathcal{B}_1$ is efficient (issuing three queries, where $\mathcal{A}$ issues two queries) and provides a sound simulation for $\mathcal{A}$.

Note in particular that, as $x^\star > n$, all of $\mathcal{B}_1$'s ODH_u queries (in particular on $x \leq n$) are permissible. As the challenge bit b coincides for $\mathcal{B}_1$'s mnPRF-ODH game and the simulated snPRF-ODH game for $\mathcal{A}$, $\mathcal{B}_1$ wins if $\mathcal{A}$ does, hence

$$\mathsf{Adv}^{\mathsf{snPRF\text{-}ODH}, x^\star > n}_{\mathbb{G},\mathsf{F}_2,\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{mnPRF\text{-}ODH}}_{\mathbb{G},\mathsf{G},\mathcal{B}_1}(\lambda). \tag{7.1}$$

In case $x^\star \leq n$, we apply the game hopping technique on the snPRF-ODH game for $\mathcal{A}$.

Game$_0(\lambda)$**:** The original $\mathsf{snPRF\text{-}ODH}$ game for $\mathcal{A}$ (with challenge queries $x^\star \leq n$).

Game$_1(\lambda)$**:** First, we change the function F_2 to a function F'_2 which is defined as follows:

$$\mathsf{F}'_2(K, x) := \begin{cases} \mathsf{F}_2(K, x) & \text{if } K \neq g^{uv} \\ \mathsf{R}(i) \oplus \mathsf{G}(K, j) & \text{if } K = g^{uv} \text{ and } x = 0 \\ \mathsf{R}(i) & \text{if } K = g^{uv} \text{ and } x = i \\ \mathsf{G}(K, x) & \text{if } K = g^{uv} \text{ and } x \notin \{0, i\} \end{cases}$$

for a truly random function R, the group elements g^u, g^v from the $\mathsf{snPRF\text{-}ODH}$ challenge, and some random, but fixed $i \xleftarrow{\$} \{1, 2\}$ and $j \in \{1, 2\}$ such that $j \neq i$.

We show that if there exists an efficient adversary that can distinguish *Game*$_0(\lambda)$ from *Game*$_1(\lambda)$, then there exists an efficient adversary $\mathcal{B}_2$ against the $\mathsf{mnPRF\text{-}ODH}$ security of G.

Algorithm $\mathcal{B}_2$ obtains g^u and begins by sampling $b \xleftarrow{\$} \{0, 1\}$ and $i \xleftarrow{\$} \{1, 2\}$ at random. It asks $x^{\star\star} = i$ as its challenge and obtains $(g^v, y^{\star\star}_{b'})$ where $y^{\star\star}_{b'}$ is either $y^{\star\star}_0 \leftarrow \mathsf{G}(g^{uv}, i)$ or $y^{\star\star}_1 \xleftarrow{\$} \{0, 1\}^\lambda$, depending on the challenge bit b'. It then provides $\mathcal{A}$ with g^u and responds to $\mathcal{A}$'s queries as follows, dependent on $\mathcal{A}$'s challenge query:

$\mathcal{A}$ asks the challenge $\mathbf{x}^\star = \mathbf{0}$: In this case, $\mathcal{B}_2$ queries (g^v, j), obtains y as response and computes $y^\star_b$, where $y^\star_0 \leftarrow y^{\star\star}_{b'} \oplus y$ and $y^\star_1 \xleftarrow{\$} \{0, 1\}^\lambda$. It returns $y^\star_b$ to $\mathcal{A}$. Observe that including $y^{\star\star}_{b'}$ makes the response in case $b = 0$ represent either F_2 (if $b' = 0$) or F'_2 (if $b' = 1$).

For $\mathcal{A}$'s ODH_u query, we let $\mathcal{B}_2$ respond with $y^{\star\star}_{b'}$ if $\mathcal{A}$ queries (g^v, i) and have $\mathcal{B}_2$ relay the query to its own ODH_u oracle otherwise.

$\mathcal{A}$ asks a challenge $\mathbf{x}^\star \neq \mathbf{0}$: We let $\mathcal{B}_2$ abort if $x^\star = i$ (which happens with probability at most $\frac{1}{2}$). Otherwise we know that $x^\star = j$ and let $\mathcal{B}_2$ query $(g^v, x^\star)$ to its own ODH_u oracle, obtaining y. It sets $y^\star_0 = y$ and $y^\star_1 \xleftarrow{\$} \{0, 1\}^\lambda$ and returns $y^\star_b$ to $\mathcal{A}$.

For $\mathcal{A}$'s ODH_u query, $\mathcal{B}_2$ simply relays queries (S, x) with $S \neq g^v$ or $x \notin \{0, i\}$. In case $\mathcal{A}$ queries $(g^v, 0)$, $\mathcal{B}_2$ asks its own ODH_u query on (g^v, j), obtaining y', and returns $y^{\star\star}_{b'} \oplus y'$ to $\mathcal{A}$. In case $\mathcal{A}$ queries (g^v, i), $\mathcal{B}_2$ responds with $y^{\star\star}_{b'}$.

When $\mathcal{A}$ stops and outputs its guess b_{guess}, $\mathcal{B}_2$ outputs 0 if $b = b_{\mathsf{guess}}$ and 1 otherwise.

$\mathcal{B}_2$ correctly simulates the $\mathsf{snPRF\text{-}ODH}$ game either for function F_2 (in case $b' = 0$ in $\mathcal{B}_2$'s $\mathsf{mnPRF\text{-}ODH}$ game) or for function F'_2 (in case $b' = 1$). By using $y^{\star\star}_{b'} = \mathsf{G}(g^{uv}, i)$ if $b' = 0$, $\mathcal{B}_2$ computes $\mathsf{F}_2(g^{uv}, 0) \leftarrow \mathsf{G}(g^{uv}, i) \oplus \mathsf{G}(g^{uv}, j)$ in the challenge and ODH_u responses for $x^\star = 0$ resp. $x = 0$, otherwise it computes $\mathsf{F}'_2(g^{uv}, 0) \leftarrow \mathsf{R}(i) \oplus \mathsf{G}(g^{uv}, j)$.

Similarly, $\mathcal{B}_2$ responds to ODH_u queries on (g^v, i) with either $\mathsf{G}(g^{uv}, i)$ or $\mathsf{R}(i)$, depending on $y^{\star\star}_{b'}$. Furthermore, by picking $b \xleftarrow{\$} \{0, 1\}$ on its own and aborting on $x^\star = i$, algorithm $\mathcal{B}_2$ also correctly provides $\mathcal{A}$ with a real-or-random challenge response in both cases.

We can determine the advantage of $\mathcal{B}_2$ against the $\mathsf{mnPRF\text{-}ODH}$ security of G as follows (informally denoting by, e.g., "$\mathcal{B}_2 = 0$" the event that $\mathcal{B}_2$ outputs 0).

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{mnPRF\text{-}ODH}}_{\mathbb{G},\mathsf{G},\mathcal{B}_2}(\lambda) &= \left|\Pr[\mathcal{B}_2 = b'] - \tfrac{1}{2}\right| \\
&= \left|\tfrac{1}{2} \cdot \left(\Pr[\mathcal{B}_2 = 0 \mid b' = 0] + \Pr[\mathcal{B}_2 = 1 \mid b' = 1]\right) - \tfrac{1}{2}\right| \\
&= \tfrac{1}{2} \cdot \left|\Pr[\mathcal{B}_2 = 0 \mid b' = 0] - \Pr[\mathcal{B}_2 = 0 \mid b' = 1]\right| \\
&= \tfrac{1}{4} \cdot \left|\Pr[\mathcal{B}_2 = 0 \mid b' = 0 \wedge \neg abort] - \Pr[\mathcal{B}_2 = 0 \mid b' = 1 \wedge \neg abort]\right| \\
&\geq \tfrac{1}{4} \cdot \left|\mathsf{Adv}^{\mathsf{snPRF\text{-}ODH},x^\star \leq n}_{\mathbb{G},\mathsf{F}_2,\mathcal{A}}(\lambda) - \tfrac{1}{2} - \mathsf{Adv}^{\mathsf{snPRF\text{-}ODH},x^\star \leq n}_{\mathbb{G},\mathsf{F}'_2,\mathcal{A}}(\lambda) + \tfrac{1}{2}\right| \\
&= \tfrac{1}{4} \cdot \left|\mathsf{Adv}^{\mathsf{snPRF\text{-}ODH},x^\star \leq n}_{\mathbb{G},\mathsf{F}_2,\mathcal{A}}(\lambda) - \mathsf{Adv}^{\mathsf{snPRF\text{-}ODH},x^\star \leq n}_{\mathbb{G},\mathsf{F}'_2,\mathcal{A}}(\lambda)\right|
\end{aligned}$$

Hence, by switching from F_2 to F'_2 we get

$$\mathsf{Adv}^{\mathsf{snPRF\text{-}ODH},x^\star \leq n}_{\mathbb{G},\mathsf{F}_2,\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{snPRF\text{-}ODH},x^\star \leq n}_{\mathbb{G},\mathsf{F}'_2,\mathcal{A}}(\lambda) + 4 \cdot \mathsf{Adv}^{\mathsf{mnPRF\text{-}ODH}}_{\mathbb{G},\mathsf{G},\mathcal{B}_2}(\lambda). \tag{7.2}$$

Game$_2(\lambda)$**:** In a second step, we similarly modify the $\mathsf{snPRF\text{-}ODH}$ game for $\mathcal{A}$ to the game $\mathsf{G}^{\mathsf{snPRF\text{-}ODH},x^\star \leq n}_{\mathbb{G},\mathsf{F}''_2,\mathcal{A}}(\lambda)$ by switching from F'_2 to a function F''_2 which now also replaces values $\mathsf{G}(K, j)$ by the output of a random function:

$$\mathsf{F}''_2(K, x) := \begin{cases} \mathsf{F}_2(K, x) & \text{if } K \neq g^{uv} \\ \mathsf{R}(1) \oplus \mathsf{R}(2) & \text{if } K = g^{uv} \text{ and } x = 0 \\ \mathsf{R}(x) & \text{if } K = g^{uv} \text{ and } x \in \{1, 2\} \\ \mathsf{G}(K, x) & \text{if } K = g^{uv} \text{ and } x > 2 \end{cases}$$

As before, this change can be bounded by the advantage of an algorithm $\mathcal{B}_3$ against the $\mathsf{mnPRF\text{-}ODH}$ security of G. Much like the reduction $\mathcal{B}_2$, the reduction $\mathcal{B}_3$ encodes its challenge on $x^{\star\star} = i \xleftarrow{\$} \{1, 2\}$ as the value representing $\mathsf{G}(g^{uv}, i)$ in F'_2 resp. $\mathsf{R}(i)$ in F''_2. For inputs $x = j$, $i \neq j \in \{1, 2\}$, $\mathcal{B}_3$ samples a random value $\mathsf{R}(j) \xleftarrow{\$} \{0, 1\}^\lambda$ on its own.

Following the same analysis as for the switch from F_2 to F'_2, we can bound the advantage difference introduced by this change. Hence we have

$$\mathsf{Adv}^{\mathsf{snPRF\text{-}ODH},x^\star \leq n}_{\mathbb{G},\mathsf{F}'_2,\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{snPRF\text{-}ODH},x^\star \leq n}_{\mathbb{G},\mathsf{F}''_2,\mathcal{A}}(\lambda) + 4 \cdot \mathsf{Adv}^{\mathsf{mnPRF\text{-}ODH}}_{\mathbb{G},\mathsf{G},\mathcal{B}_3}(\lambda). \tag{7.3}$$

Finally, in the $\mathsf{snPRF\text{-}ODH}$ game for F''_2, the values $\mathsf{F}''_2(g^{uv}, 1)$ and $\mathsf{F}''_2(g^{uv}, 2)$ are independent, uniformly random values and $\mathsf{F}''_2(g^{uv}, 0)$ their XOR combination. The advantage of $\mathcal{A}$ in this game can hence immediately be reduced to the advantage $\mathcal{B}_4$ in a $\mathsf{mnPRF\text{-}ODH}$ game against G.

For this, $\mathcal{B}_4$ replies to all challenge and ODH_u queries on (g^{uv}, x) for $x \in \{0, 1, 2\}$ on its own with $\mathsf{R}(x)$ (for $x \in \{1, 2\}$) resp. $\mathsf{R}(1) \oplus \mathsf{R}(2)$ (for $x = 0$), picking random values $\mathsf{R}(1)$, $\mathsf{R}(2) \xleftarrow{\$} \{0, 1\}^\lambda$ itself. Note that, since $\mathcal{A}$ can pose at most two queries (one challenge and one ODH_u query), these responses are consistent even though $\mathcal{B}_4$ does respond with distinct real ($\mathsf{R}(x)$) or random values.

Beyond that, $\mathcal{B}_4$ can simply relay all other queries to its own challenge and ODH_u oracles. This finally determines the advantage of $\mathcal{A}$ against F''_2 as

$$\mathsf{Adv}^{\mathsf{snPRF\text{-}ODH},x^\star \leq n}_{\mathbb{G},\mathsf{F}''_2,\mathcal{A}}(\lambda) = \mathsf{Adv}^{\mathsf{mnPRF\text{-}ODH}}_{\mathbb{G},\mathsf{G},\mathcal{B}_4}(\lambda). \tag{7.4}$$

Combining the above advantage bounds, i.e., Equations (7.1) – (7.4), yields the overall bound. □

In a similar way as for Proposition 7.6 we now also separate nsPRF-ODH security from mnPRF-ODH, ssPRF-ODH, and nmPRF-ODH security.

Proposition 7.7 (nsPRF-ODH $\not\Longrightarrow$ mnPRF-ODH, ssPRF-ODH, nmPRF-ODH)**.**
If G *from Definition 7.3 is* nmPRF-ODH *secure, then* $\mathsf{F}_2 \in \mathcal{F}$ *is* nsPRF-ODH *secure, but neither* mnPRF-ODH*, nor* ssPRF-ODH*, nor* nmPRF-ODH *secure. More precisely, for any efficient adversary* $\mathcal{A}$ *against the* nsPRF-ODH *security of* F_2*, there exist efficient algorithms* $\mathcal{B}_1$*,* $\mathcal{B}_2$*,* $\mathcal{B}_3$*, and* $\mathcal{B}_4$ *such that*

$$\begin{aligned}\mathsf{Adv}^{\mathsf{nsPRF\text{-}ODH}}_{\mathbb{G},\mathsf{F}_2,\mathcal{A}}(\lambda) \leq\ & \mathsf{Adv}^{\mathsf{nmPRF\text{-}ODH}}_{\mathbb{G},\mathsf{G},\mathcal{B}_1}(\lambda) + 4 \cdot \mathsf{Adv}^{\mathsf{nmPRF\text{-}ODH}}_{\mathbb{G},\mathsf{G},\mathcal{B}_2}(\lambda) \\ &+\ 4 \cdot \mathsf{Adv}^{\mathsf{nmPRF\text{-}ODH}}_{\mathbb{G},\mathsf{G},\mathcal{B}_3}(\lambda) + \mathsf{Adv}^{\mathsf{nmPRF\text{-}ODH}}_{\mathbb{G},\mathsf{G},\mathcal{B}_4},\end{aligned}$$

but there exist efficient algorithms $\mathcal{A}_1$*,* $\mathcal{A}_2$*,* $\mathcal{A}_3$ *with non-negligible advantage*

$$\mathsf{Adv}^{\mathsf{mnPRF\text{-}ODH}}_{\mathbb{G},\mathsf{F}_2,\mathcal{A}_1}(\lambda) = \mathsf{Adv}^{\mathsf{ssPRF\text{-}ODH}}_{\mathbb{G},\mathsf{F}_2,\mathcal{A}_2}(\lambda) = \mathsf{Adv}^{\mathsf{nmPRF\text{-}ODH}}_{\mathbb{G},\mathsf{F}_2,\mathcal{A}_3}(\lambda) = 1 - 2^{-\lambda}.$$

Proof. In Proposition 7.6 we have already established that F_2 provides no nmPRF-ODH, ssPRF-ODH and mnPRF-ODH security given the respective algorithms $\mathcal{A}_1$, $\mathcal{A}_2$, and $\mathcal{A}_3$. For the proof that F_2 is nsPRF-ODH secure, observe that applying the mirrored proof that F_2 is snPRF-ODH secure from Proposition 7.6 establishes the desired bound. In particular, via algorithms $\mathcal{B}_1$, $\mathcal{B}_2$, $\mathcal{B}_3$, $\mathcal{B}_4$ with multiple queries to the ODH_v oracle we can similarly to before reduce the security to the nmPRF-ODH security of G. □

We now consider the separation of ssPRF-ODH from mnPRF-ODH and nmPRF-ODH.

Proposition 7.8 (ssPRF-ODH $\not\Longrightarrow$ mnPRF-ODH, nmPRF-ODH)**.**
If G *from Definition 7.3 is* msPRF-ODH *secure, then* $\mathsf{F}_3 \in \mathcal{F}$ *is* ssPRF-ODH *secure, but neither* mnPRF-ODH *nor* nmPRF-ODH *secure. More precisely, for any efficient adversary* $\mathcal{A}$ *against the* ssPRF-ODH *security of* F_3*, there exist efficient algorithms* $\mathcal{B}_1$*,* $\mathcal{B}_2$*, ...,* $\mathcal{B}_5$ *such that*

$$\begin{aligned}\mathsf{Adv}^{\mathsf{ssPRF\text{-}ODH}}_{\mathbb{G},\mathsf{F}_3,\mathcal{A}}(\lambda) \leq\ & \mathsf{Adv}^{\mathsf{msPRF\text{-}ODH}}_{\mathbb{G},\mathsf{G},\mathcal{B}_1}(\lambda) + 3 \cdot \mathsf{Adv}^{\mathsf{msPRF\text{-}ODH}}_{\mathbb{G},\mathsf{G},\mathcal{B}_2}(\lambda) \\ &+ 3 \cdot \mathsf{Adv}^{\mathsf{msPRF\text{-}ODH}}_{\mathbb{G},\mathsf{G},\mathcal{B}_3}(\lambda) + 3 \cdot \mathsf{Adv}^{\mathsf{msPRF\text{-}ODH}}_{\mathbb{G},\mathsf{G},\mathcal{B}_4}(\lambda) + \mathsf{Adv}^{\mathsf{msPRF\text{-}ODH}}_{\mathbb{G},\mathsf{G},\mathcal{B}_5}(\lambda),\end{aligned}$$

but there exist algorithms $\mathcal{A}_1$ *and* $\mathcal{A}_2$ *with non-negligible advantage*

$$\mathsf{Adv}^{\mathsf{mnPRF\text{-}ODH}}_{\mathbb{G},\mathsf{F}_3,\mathcal{A}_1}(\lambda) = \mathsf{Adv}^{\mathsf{nmPRF\text{-}ODH}}_{\mathbb{G},\mathsf{F}_3,\mathcal{A}_2}(\lambda) = 1 - 2^{-\lambda}.$$

Proof. As for the previous separations, it is apparent that F_3 cannot provide security in the sense of mnPRF-ODH or nmPRF-ODH. An adversary querying the three values $\mathsf{F}_3(g^{uv}, i)$, $i \in \{1, 2, 3\}$ can use its ODH_u resp. ODH_v oracle to distinguish the real value $\mathsf{F}_3(g^{uv}, 0)$ from a random value, except with negligible probability $2^{-\lambda}$.

For proving that F_3 is ssPRF-ODH secure if G is msPRF-ODH secure, we apply the same proof strategy applied in the proof of Proposition 7.6 for showing snPRF-ODH security of F_2 based on the mnPRF-ODH security of G, but with two modifications:

Modification 1: The first difference we have to take into account is that an ssPRF-ODH adversary $\mathcal{A}$ may also issue a (single) ODH_v query for a value $T \neq g^u$ (beyond its challenge and one ODH_u query). While such a query does not interfere with our proof steps, we cannot relay such a query to an ODH_u oracle. Thus we need that G is msPRF-ODH secure, providing the reductions with a (single) ODH_v oracle call.

Modification 2: The other change is that we replace values $\mathsf{G}(g^{uv}, i)$ by the outputs $\mathsf{R}(i)$ of a random function R for three ($i \in \{1, 2, 3\}$) instead of two labels. We abort in each step if $\mathcal{A}$ asks a challenge $x^\star = i$ for our guessed i, which now happens with probability at most $\frac{1}{3}$. Accounting for the modified factor, this yields the three intermediary advantage bounds of $3 \cdot \mathsf{Adv}^{\mathsf{msPRF\text{-}ODH}}_{\mathbb{G},\mathsf{G},\mathcal{B}_j}(\lambda)$ for $j \in \{1, 2, 3\}$.

□

7.3.2 Separations in the Random Oracle Model

We now turn towards the separations in the random oracle model. Here, we will use a version of the following interactive-inversion DH problem of computing non-trivial v-th roots in $\mathbb{G}$ for implicitly given exponent v:

Interactive-inversion DH. Consider an algorithm $\mathcal{A}$ which outputs some group element $X \in \mathbb{G}$ with $X \neq 1$ (and some state information) and then receives g^v for random $v \xleftarrow{\$} \mathbb{Z}_q$. At some point, $\mathcal{A}$ terminates with some output Y, such that $Y^v = X$ (cf. Figure 7.7).

```
$\mathsf{G}^{\mathsf{iDH}}_{\mathbb{G},\mathcal{A}}(\lambda)$:
$v \xleftarrow{\$} \mathbb{Z}_q$
$Y \xleftarrow{\$} \mathcal{A}(g^v)$
return $[\![Y^v = g]\!]$
```

```
$\mathsf{G}^{\mathsf{iiDH}}_{\mathbb{G},\mathcal{A}}(\lambda)$:
$X \xleftarrow{\$} \mathcal{A}$
$v \xleftarrow{\$} \mathbb{Z}_q$
$Y \xleftarrow{\$} \mathcal{A}(g^v)$
if $x = 1$
    return $\perp$
else
    return $[\![Y^v = X]\!]$
```

```
$\mathsf{G}^{\mathsf{siiDH}}_{\mathbb{G},\mathcal{A}}(\lambda)$:
$X \xleftarrow{\$} \mathcal{A}$
$v \xleftarrow{\$} \mathbb{Z}_q$
$Y \xleftarrow{\$} \mathcal{A}^{\mathsf{DDH}(g^v,\cdot,\cdot)}(g^v)$
if $X = 1$
    return $\perp$
else
    return $[\![Y^v = X]\!]$

$\mathsf{DDH}(g^v, A, B)$:
return $[\![A^v = B]\!]$
```

Figure 7.7: Inversion DH problem iDH, interactive-inversion DH problem iiDH and strong interactive-inversion DH problem siiDH.

Note that the problem would be trivial if $X = 1$ was allowed (in which case $Y = 1$ would provide a solution), or if X can be chosen after having seen g^v (in which case $X = g^v$ and $Y = g$ would trivially work). Excluding these trivial cases, in terms of generic or algebraic hardness the problem is equivalent to the CDH problem (given g^u, g^v compute g^{uv}; cf. Section 3.5):

Assume $\mathcal{A}$ "knows" $\alpha \in \mathbb{Z}_q$ such that $X = g^\alpha$. Since X is chosen before seeing g^v, the adversary can only compute it as a power of g and, in addition, $X \neq 1$ implies $\alpha \neq 0$. Therefore, for any valid solution Y the value $Y^{1/\alpha}$ would be a v-th root of g, because $(Y^{1/\alpha})^v = X^{1/\alpha} = g$. This problem of computing $g^{1/v}$ from g, g^v, however, is known as the *inversion DH* (iDH) problem (cf. Figure 7.7) and has been shown to be equivalent to the CDH problem with a loose reduction by Bao, Deng, and Zhu [BDZ03].

Strong interactive-inversion DH. For our separation result we need a slightly stronger version, where, in the second phase, the adversary also gets access to a decision oracle which, on input two group elements $A, B \in \mathbb{G}$ outputs 1 if and only if $A^v = B$.[6] We call this the *strong interactive-inversion DH* problem and denote it by siiDH (cf. Figure 7.7). Let us further denote

[6]Note that for example in a pairing-based group such an oracle is given for free, while computing a v-th root of g (or, equivalently, solving the computational DH problem) may still be hard.

by $\mathsf{Adv}^{\mathsf{siiDH}}_{\mathbb{G},\mathcal{A}}(\lambda)$ the probability that $\mathcal{A}$ succeeds in the strong interactive-inversion DH game, i.e.,

$$\mathsf{Adv}^{\mathsf{siiDH}}_{\mathbb{G},\mathcal{A}}(\lambda) = \Pr\left[\mathsf{G}^{\mathsf{siiDH}}_{\mathbb{G},\mathcal{A}}(\lambda) = 1\right]. \tag{7.5}$$

With the help of siiDH and the strong Diffie–Hellman assumption StDH, we are now ready to separate $\mathsf{nmPRF\text{-}ODH}$ from $\mathsf{snPRF\text{-}ODH}$ and $\mathsf{smPRF\text{-}ODH}$ from $\mathsf{mnPRF\text{-}ODH}$.

Proposition 7.9 ($\mathsf{nmPRF\text{-}ODH} \not\Rightarrow \mathsf{snPRF\text{-}ODH}$)**.**
In the random oracle model, and assuming the intractability of StDH *and* siiDH*, there exists a function* F^{RO} *which is* $\mathsf{nmPRF\text{-}ODH}$ *secure but not* $\mathsf{snPRF\text{-}ODH}$ *secure. More precisely, for any efficient adversary* $\mathcal{A}^{\mathsf{RO}}$ *against the* $\mathsf{nmPRF\text{-}ODH}$ *security of* F^{RO}*, there exist efficient algorithms* $\mathcal{B}_1$ *and* $\mathcal{B}_2$ *such that*

$$\mathsf{Adv}^{\mathsf{nmPRF\text{-}ODH}}_{\mathbb{G},\mathsf{F}^{\mathsf{RO}},\mathcal{A}^{\mathsf{RO}}}(\lambda) \leq \mathsf{Adv}^{\mathsf{stDH}}_{\mathbb{G},\mathcal{B}_1}(\lambda) + q_{\mathsf{ODH}_v} \cdot \mathsf{Adv}^{\mathsf{siiDH}}_{\mathbb{G},\mathcal{B}_2}(\lambda) + \frac{q_{\mathsf{RO}}}{q} + 2^{-\lambda},$$

where q_{RO} *and* q_{ODH_v} *are the maximum number of queries to the random oracle and the* ODH_v *oracle, respectively, and* q *is the order of the group* $\mathbb{G}$*. However there exists an efficient algorithm* $\mathcal{A}^{\mathsf{RO}}$ *with non-negligible advantage in the* $\mathsf{snPRF\text{-}ODH}$ *game, i.e., such that*

$$\mathsf{Adv}^{\mathsf{snPRF\text{-}ODH}}_{\mathbb{G},\mathsf{F}^{\mathsf{RO}},\mathcal{A}^{\mathsf{RO}}}(\lambda) \geq 1 - 2^{-\lambda+1}.$$

Proof. For the proof consider the function $\mathsf{F}^{\mathsf{RO}} : \mathbb{G} \times (\mathbb{G} \times \{0,1\}^\lambda) \to \{0,1\}^\lambda$ defined as

$$\mathsf{F}^{\mathsf{RO}}(K,(x,y)) := \begin{cases} y & \text{if } \mathsf{RO}(Kx^{-1},(x,0^\lambda)) = y \text{ and } x \neq 1 \\ \mathsf{RO}(K,(x,y)) & \text{else} \end{cases}$$

We first show that this function is not secure in a $\mathsf{PRF\text{-}ODH}$ sense if a single query to the right ODH_u oracle is allowed. However, as we will argue then, it can withstand polynomially many queries to the ODH_v oracle and thus achieves $\mathsf{nmPRF\text{-}ODH}$ security.

$\underline{\mathsf{F}^{\mathsf{RO}}\text{ is not }\mathsf{snPRF\text{-}ODH}\text{ secure.}}$ We first argue that this function is easy to attack in the $\mathsf{snPRF\text{-}ODH}$ sense, when given an ODH_u oracle which can be queried once after having learned the challenge values. To this end let $\mathcal{A}^{\mathsf{RO}}$, on input g and g^u use $x^\star = (g^u, 0^\lambda)$ in the challenge query, resulting in the response $(g^v, y^\star_b)$. If $g^u = 1$ then our adversary can compute the key $g^{uv} = g^v$ and verify $y^\star_b$ directly via RO, outputting $b_{\mathsf{guess}} = 0$ if the challenge value matches the computed value. Otherwise, in the single query to the ODH_u oracle let the adversary forward $(g^{v+1}, (g^u, y^\star_b))$ to get the value y. The adversary then outputs $b_{\mathsf{guess}} = 0$ if and only if this answer y matches $y^\star_b$.

Note that for $b = 0$, i.e., $y^\star_b$ is computed as $\mathsf{F}^{\mathsf{RO}}(g^{uv}, (g^u, 0^\lambda))$ and our adversary always returns $b_{\mathsf{guess}} = 0$. For $g^u = 1$ this is straightforward to see. Otherwise, the key in the query to ODH_u equals g^{uv+v} such that the exception is satisfied.

If $y^\star_b$ is random, i.e. $b = 1$, then we have a match with probability at most $2^{-\lambda+1}$: either the random $y^\star$ accidentally matches the actual pseudorandom value, or the random oracle output $\mathsf{RO}(g^{uv+u}, (g^u, y^\star))$ coincides with $y^\star$. Both events happen with probability at most $2^{-\lambda}$.

$\underline{\mathsf{F}^{\mathsf{RO}}\text{ is }\mathsf{nmPRF\text{-}ODH}\text{ secure via }\mathsf{siiDH}.}$

For the security in the $\mathsf{nmPRF\text{-}ODH}$ sense recall that now the adversary has "only" access to an ODH_v oracle for multiple queries, after having received the challenge value $y^\star_b$ for challenge query $x^\star$.

Game$_0(\lambda)$: The original $\mathsf{nmPRF\text{-}ODH}$ game.

Game$_1(\lambda)$: We first argue that the adversary can never ask the random oracle about the DH key g^{uv} used during the attack. This step will be discussed more thoroughly for the more general case in the instantiation proof of $\mathsf{nmPRF\text{-}ODH}$ under the StDH assumption (cf. Theorem 7.15), so we skip it here. We emphasize that this also provides a sound simulation of the random oracle answers without explicit knowledge of u, v in the attack.

We have

$$\mathsf{Adv}^{\mathsf{nmPRF\text{-}ODH}}_{\mathbb{G},\mathsf{F},\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{G}_1}_{\mathbb{G},\mathsf{F},\mathcal{A}}(\lambda) + \mathsf{Adv}^{\mathsf{stDH}}_{\mathbb{G},\mathcal{B}_1}(\lambda). \tag{7.6}$$

Game$_2(\lambda)$: Next, note that we can assume that the adversary with its challenge query $x^\star = (x', y')$ does not trigger the exceptional event of having the function output $y' = \mathsf{RO}(g^{uv}(x')^{-1}, (x', 0^\lambda))$.

This is since v is only picked after the adversary has submitted $x^\star$. Hence, the probability that any previous random oracle query was about this key, is at most $\frac{q_{\mathsf{RO}}}{q}$. Else, y' matches this previously not queried random oracle value only with probability $2^{-\lambda}$. Hence, the adversary receives the value $y^\star$ as either $\mathsf{RO}(g^{uv}, (x', y'))$ or as a random value.

We have

$$\mathsf{Adv}^{\mathsf{G}_1}_{\mathbb{G},\mathsf{F},\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{G}_2}_{\mathbb{G},\mathsf{F},\mathcal{A}}(\lambda) + \frac{q_{\mathsf{RO}}}{q} + 2^{-\lambda}.$$

Since queries $(g^u, x^\star)$ to the ODH_v oracle are prohibited and the adversary never asks the random oracle about the key g^{uv}, the only possibility to distinguish a random challenge $y^\star$ from an actual random oracle answer, is to trigger the exceptional case in the function evaluation for the challenge data. As we will show next, this however would immediately imply an efficient solver for the siiDH problem.

Game$_3(\lambda)$: So assume that the adversary queries its ODH_v oracle about $(T, (x, y))$ such that $T^v x^{-1} = g^{uv}$, $x^\star = (x, 0^\lambda)$ for $x \neq 1$, and y matches $\mathsf{RO}(T^v x^{-1}, (x, 0^\lambda))$. In this case we have $x = (Tg^{-u})^v$ and therefore Tg^{-u} is a v-th root of $x \neq 1$. This can now be straightforwardly turned into an efficient solver $\mathcal{B}_2$ of the (strong) interactive inversion assumption siiDH, as follows:

$\mathcal{B}_2$ initiates the environment for $\mathcal{A}$ by picking $u \xleftarrow{\$} \mathbb{Z}_q$ itself. Random oracle queries of $\mathcal{A}$ before it has output its challenge value $x^\star$, are answered by lazy sampling. Once $\mathcal{A}$ outputs $x^\star$, $\mathcal{B}_2$ forwards $x^\star$ to its own challenger and receives a value g^v which it forwards to $\mathcal{A}$ along with a random challenge element $y^\star$.

The adversary $\mathcal{A}$ may now query the ODH_v oracle in addition to the random oracle. $\mathcal{B}_2$ uses its decisional oracle $\mathsf{DDH}(g^v, \cdot, \cdot)$ to answer $\mathcal{A}$'s ODH_v and RO queries consistently (as in the reduction to the StDH problem).

Note that, since $\mathcal{B}_2$ had no knowledge of g^v prior to $\mathcal{A}$ outputting $x^\star$, random oracle responses of $\mathcal{A}$ before the challenge request might not have been soundly simulated. However, the probability that the adversary has made such a query earlier (and thus could detect the soundness error) is at most $\frac{q_{\mathsf{RO}}}{q}$, since v is chosen at random in $\mathbb{Z}_q$, and we have already accounted for such cases for the event that the adversary has not triggered the exceptional event in the challenge query in *Game*$_2(\lambda)$.

The reduction then guesses the "correct" ODH_v query $(T, (x, y))$ of the adversary and outputs Tg^{-u} as the v-th root in its strong interactive-inversion DH game.

Clearly, if $\mathcal{A}$ is able to trigger the exceptional case and $\mathcal{B}_2$ guesses the correct query, then $\mathcal{B}_2$ wins its siiDH game with non-negligible advantage.

We have

$$\mathsf{Adv}^{\mathsf{G}_2}_{\mathbb{G},\mathsf{F},\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{G}_3}_{\mathbb{G},\mathsf{F},\mathcal{A}}(\lambda) + q_{\mathsf{ODH}_v} \cdot \mathsf{Adv}^{\mathsf{siiDH}}_{\mathbb{G},\mathcal{B}_2}(\lambda)$$

where q_{ODH_v} is the number of oracle queries $\mathcal{A}$ makes to the ODH_v oracle.

Having excluded all of the above scenarios, we have excluded adversary from making a "bad" query to the ODH_v oracle. Thus, $\mathcal{A}$ cannot distinguish the two challenge cases at all, i.e., $\mathsf{Adv}^{\mathsf{G}_3}_{\mathbb{G},\mathsf{F},\mathcal{A}}(\lambda) \leq 0$, which concludes the proof.

□

The idea can now be transferred to the case that we further allow one oracle query to ODH_u, basically by "secret sharing" the reply in the exceptional case among two queries:

Proposition 7.10 ($\mathsf{smPRF\text{-}ODH} \not\Rightarrow \mathsf{mnPRF\text{-}ODH}$)**.**
In the random oracle model, and assuming StDH *and* siiDH*, there exists a function* F^{RO} *which is* $\mathsf{smPRF\text{-}ODH}$ *secure but not* $\mathsf{mnPRF\text{-}ODH}$ *secure. More precisely, for any efficient adversary* $\mathcal{A}^{\mathsf{RO}}$ *against the* $\mathsf{smPRF\text{-}ODH}$ *security of* F^{RO}*, there exist efficient algorithms* $\mathcal{B}_1, \mathcal{B}_2$*, such that*

$$\mathsf{Adv}^{\mathsf{smPRF\text{-}ODH}}_{\mathbb{G},\mathsf{F}^{\mathsf{RO}},\mathcal{A}^{\mathsf{RO}}}(\lambda) \leq \sqrt{\mathsf{Adv}^{\mathsf{stDH}}_{\mathbb{G},\mathcal{B}_1}(\lambda)} + q_{\mathsf{ODH}_v} \cdot \mathsf{Adv}^{\mathsf{siiDH}}_{\mathbb{G},\mathcal{B}_2}(\lambda) + \frac{q_{\mathsf{RO}}}{q} + 2^{-\lambda},$$

for the at most q_{RO} *and* q_{ODH_v} *queries to the random oracle and* ODH_v *oracle, respectively, but there exists an efficient algorithm* $\mathcal{A}^{\mathsf{RO}}$ *with non-negligible advantage*

$$\mathsf{Adv}^{\mathsf{mnPRF\text{-}ODH}}_{\mathbb{G},\mathsf{F}^{\mathsf{RO}},\mathcal{A}^{\mathsf{RO}}}(\lambda) \geq 1 - 2^{-\lambda+1}.$$

Proof. Consider the function $\mathsf{F}^{\mathsf{RO}} : \mathbb{G} \times (\mathbb{G} \times \{0,1\}^\lambda) \to \{0,1\}^\lambda$ defined as:

$$\mathsf{F}^{\mathsf{RO}}(K,(x,y)) = \begin{cases} y \oplus \mathsf{RO}(K,(K,0^\lambda)) & \text{if } \mathsf{RO}(Kx^{-1},(x,0^\lambda)) = y \text{ and } x \neq 1 \\ \mathsf{RO}(K,(K,0^\lambda)) & \text{if } \mathsf{RO}(Kx^{-1},(x,0^\lambda)) = 1^\lambda \oplus y \text{ and } x \neq 1 \\ \mathsf{RO}(K,(x,y)) & \text{else} \end{cases}$$

$\underline{\mathsf{F}^{\mathsf{RO}}\text{ is not }\mathsf{mnPRF\text{-}ODH}\text{ secure.}}$ We first show again that this function is easy to attack in the $\mathsf{mnPRF\text{-}ODH}$ case. The adversary is very similar to the one in Proposition 7.9. Namely, our adversary asks the challenge query $x^\star = (g^u, 0^\lambda)$ to receive $y^\star$, then asks its ODH_u oracle about $(g^{v+1},(g^u,y^\star))$ and $(g^{v+1},(g^u,y^\star \oplus 1^\lambda))$, and adds the answers and compares them to $y^\star$. If and only if the values match it outputs 0. In case $g^u = 1$ it can again compute the reply directly. Overall, it always outputs 0 if $y^\star$ is the function value, and only with probability at most $2^{-\lambda+1}$ if it is random.

$\underline{\mathsf{F}^{\mathsf{RO}}\text{ is }\mathsf{smPRF\text{-}ODH}\text{ secure via }\mathsf{siiDH}.}$ Arguing again that an algorithm with multi-query access to ODH_v and single-query access to ODH_u cannot succeed with non-negligible probability is very close to the previous case in Proposition 7.9, where the single query to the ODH_u oracle is simulated by returning an independent, uniformly random value. Ensuring consistency between the two ODH oracles yields a loose reduction to the StDH problem.

□

In fact, in the negative result for mnPRF-ODH the adversary only needs to ask two queries to the ODH_u oracle *after* receiving the challenge query. Since the function is still secure for a single ODH_u query, this is optimal in this regard.

Corollary 7.11 (nmPRF-ODH $\not\Rightarrow$ smPRF-ODH)**.** *Since it holds that* smPRF-ODH *implies* ssPRF-ODH*, which in turn implies* snPRF-ODH*, the separation between* nmPRF-ODH *and* snPRF-ODH *from Proposition 7.9 immediately also establishes* nmPRF-ODH $\not\Rightarrow$ smPRF-ODH*.*

Corollary 7.12 (smPRF-ODH $\not\Rightarrow$ mmPRF-ODH)**.** *As* mmPRF-ODH *implies* msPRF-ODH*, which in turn implies* mnPRF-ODH*, the separation between* smPRF-ODH *and* mnPRF-ODH *from Proposition 7.10 immediately also establishes* smPRF-ODH $\not\Rightarrow$ mmPRF-ODH*.*

7.3.3 Discussion

Let us close this section with some remarks about the separations.

Remark 7.13. Our separating function family (cf. Definition 7.3) establishes a number of separations, but cannot be used in order to separate the remaining variants. This is due to the fact that our function family cannot separate between notions that both allow polynomially many queries as for example nmPRF-ODH and smPRF-ODH. Thus, we have turned to the random oracle model to establish further separations. We already note that using this idealized model is alleviated by the implausibility result of instantiating PRF-ODH in the standard model which we will briefly discuss later.

In the random oracle model we have shown that it is crucial whether the adversary has access to the ODH_u oracle or not (or how many times). This uses some asymmetry in the two oracles, namely, that g^u is given before the challenge query, and g^v only after. Our separations take advantage of this difference, visualized via the interactive-inversion DH problem which is only hard if $x^\star$ is chosen before receiving g^v.

It is currently open if the other notions are separable. Beyond the asymmetry that g^u is already available before the challenge by $\mathcal{A}$, it is unclear how to "encode" other distinctive information into the input to the "memoryless" PRF which one oracle can exploit but the other one cannot.

Relations between sym-lrPRF-ODH notions. As mentioned already in the remark above, the random oracle separations given in this section do not carry over to the symmetric setting as they crucially rely on the asymmetry in the oracles. The results from this section that do apply in the symmetric setting are depicted in Figure 7.8. The separation between sym-msPRF-ODH and sym-mmPRF-ODH remains open.

7.4 Instantiations

We next turn to the question how to instantiate the PRF-ODH assumption. Concretely, we provide instantiations of the two notions that mark both ends of the strength spectrum of the PRF-ODH variants.

We first show that the weakest PRF-ODH variant, nnPRF-ODH, can be instantiated in the standard model under well-established assumptions, namely from a pseudorandom function and assuming intractability of the decisional Diffie–Hellman (DDH) problem.

On the other end of the spectrum, we establish that both the strongest *one-sided* PRF-ODH variants, mnPRF-ODH and nmPRF-ODH, as well as the most general mmPRF-ODH assumption can be instantiated from the strong Diffie–Hellman assumption (StDH) in the (programmable) random oracle model.

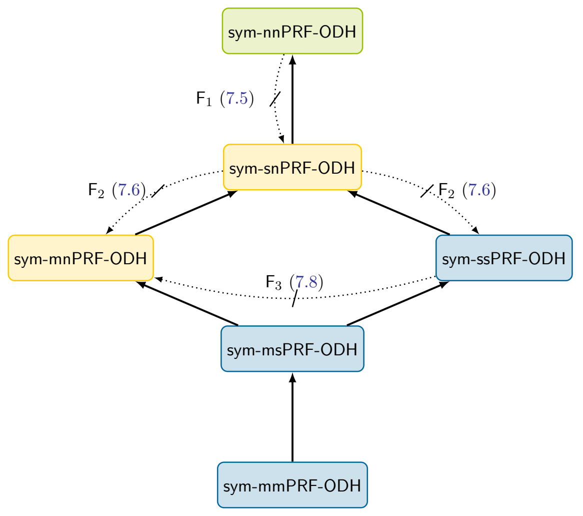

Figure 7.8: Relations result in the symmetric PRF-ODH setting.

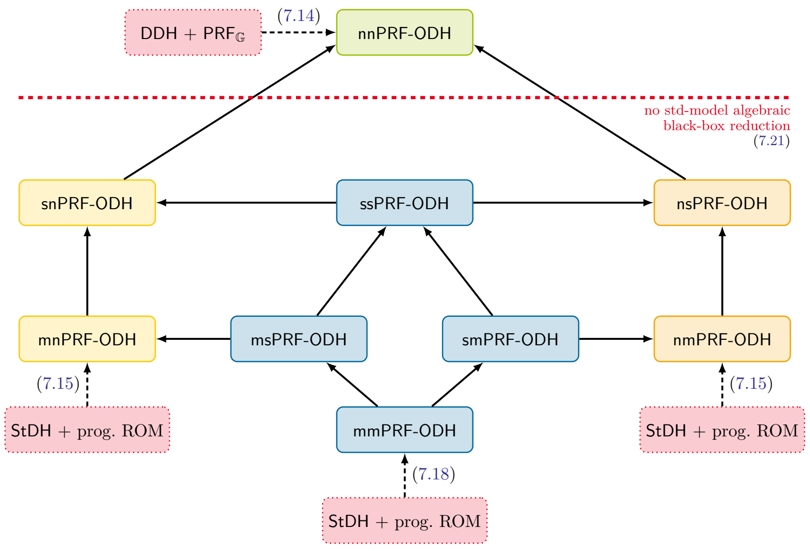

Figure 7.9: Assumptions (in dotted-line rounded rectangles). The dashed horizontal line demarcates the boundary below which our impossibility result for standard-model algebraic black-box reductions holds. Numbers in parentheses indicate the respective propositions and theorems.

The proofs for some of the instantiation results have appeared already implicitly in previous work about key exchange, e.g., in [Kra05, Ust08, DF11, FG14, LJBN15, KW16, LXZ$^+$16]. There the reduction to the respective DH problem in the random oracle model has been carried out by dragging along all the steps of the key exchange protocol itself. This is both cumbersome and, due to the increased complexity, error-prone. Our instantiation results thus, in particular, yield cleaner, simpler reductions in key exchange protocol security analyses by allowing for a straightforward reduction to the corresponding PRF-ODH notion.

7.4.1 Standard-Model Instantiation of nnPRF-ODH

We begin with instantiating the $\mathsf{nnPRF\text{-}ODH}$ assumption in the standard model. We next show however, that the $\mathsf{nnPRF\text{-}ODH}$ assumption can be instantiated from a pseudorandom function F keyed with group elements from $\mathbb{G}$, and assuming that the DDH assumption holds in $\mathbb{G}$:

Theorem 7.14 ($\mathsf{DDH} + \mathsf{PRF}_{\mathbb{G}} \implies \mathsf{nnPRF\text{-}ODH}$)**.**
If $\mathsf{F}\colon \mathbb{G} \times \{0,1\}^\star \to \{0,1\}^\lambda$ *is a pseudorandom function and the* DDH *assumption holds in* $\mathbb{G}$*, then* F *is also* $\mathsf{nnPRF\text{-}ODH}$*-secure. More precisely, for any efficient adversary* $\mathcal{A}$ *against the* $\mathsf{nnPRF\text{-}ODH}$ *security of* F*, there exist efficient algorithms* $\mathcal{B}_1$ *and* $\mathcal{B}_2$ *such that*

$$\mathsf{Adv}^{\mathsf{nnPRF\text{-}ODH}}_{\mathbb{G},\mathsf{F},\mathcal{A}}(\lambda) \leq 2 \cdot \left(\mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G},\mathcal{B}_1}(\lambda) + \mathsf{Adv}^{\mathsf{prf\text{-}sec}}_{\mathsf{F},\mathcal{B}_2}(\lambda)\right).$$

Proof. The game proceeds in a sequence of game. The respective game hops are depicted in Figure 7.10.

```
$\mathit{Game}_0(\lambda)$:
$u \xleftarrow{\$} \mathbb{Z}_q$
$(x^\star, st) \xleftarrow{\$} \mathcal{A}(\mathbb{G}, g, g^u)$
$v \xleftarrow{\$} \mathbb{Z}_q$
$b \xleftarrow{\$} \{0,1\}$
$y_0^\star \leftarrow \mathsf{F}(g^{uv}, x^\star)$
$y_1^\star \xleftarrow{\$} \{0,1\}^\lambda$
$b' \xleftarrow{\$} \mathcal{A}(g^v, y_b^\star, st)$
return $[\![b' = b]\!]$
```

```
$\mathit{Game}_1(\lambda)$:
$u \xleftarrow{\$} \mathbb{Z}_q$
$(x^\star, st) \xleftarrow{\$} \mathcal{A}(\mathbb{G}, g, g^u)$
$v \xleftarrow{\$} \mathbb{Z}_q$
$b \xleftarrow{\$} \{0,1\}$
$z \xleftarrow{\$} \mathbb{Z}_q$
$y_0^\star \leftarrow \mathsf{F}(g^z, x^\star)$
$y_1^\star \xleftarrow{\$} \{0,1\}^\lambda$
$b' \xleftarrow{\$} \mathcal{A}(g^v, y_b^\star, st)$
return $[\![b' = b]\!]$
```

```
$\mathit{Game}_2(\lambda)$:
$u \xleftarrow{\$} \mathbb{Z}_q$
$(x^\star, st) \xleftarrow{\$} \mathcal{A}(\mathbb{G}, g, g^u)$
$v \xleftarrow{\$} \mathbb{Z}_q$
$b \xleftarrow{\$} \{0,1\}$
$z \xleftarrow{\$} \mathbb{Z}_q$
$y_0^\star \xleftarrow{\$} \{0,1\}^\lambda$
$y_1^\star \xleftarrow{\$} \{0,1\}^\lambda$
$b' \xleftarrow{\$} \mathcal{A}(g^v, y_b^\star, st)$
return $[\![b' = b]\!]$
```

Figure 7.10: Game hops for proof of Theorem 7.14.

Let $\mathcal{A}$ be an efficient adversary against the $\mathsf{nnPRF\text{-}ODH}$ security of F. We show that $\mathcal{A}$'s advantage $\mathsf{Adv}^{\mathsf{nnPRF\text{-}ODH}}_{\mathsf{F},\mathcal{A}}(\lambda)$ in winning the $\mathsf{nnPRF\text{-}ODH}$ game is bounded by $\mathcal{B}_1$'s advantage $\mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G},\mathcal{B}_1}(\lambda)$ against DDH, and $\mathcal{B}_2$'s advantage $\mathsf{Adv}^{\mathsf{prf\text{-}sec}}_{\mathsf{F},\mathcal{B}_2}(\lambda)$ against the pseudorandomness of F. This is again done via the game-hopping technique.

Game$_0(\lambda)$**:** The original $\mathsf{nnPRF\text{-}ODH}$ game.

Game$_1(\lambda)$**:** As the original game, but we replace the key g^{uv} used in computing the challenge value $y_0^\star$ by an independent random group element $g^z \in \mathbb{G}$. We claim that $\mathcal{A}$ cannot distinguish *Game*$_0(\lambda)$ from *Game*$_1(\lambda)$ efficiently with non-negligible advantage, since otherwise there exists an adversary $\mathcal{B}_1$ that can efficiently solve DDH. Assume that $\mathcal{A}$ is able to distinguish the two games. Then $\mathcal{B}_1$ is constructed as follows:

In the DDH game, $\mathcal{B}_1$ receives its challenge (g, g^u, g^v, g^z), where g^z is either g^{uv} or a uniformly random group element. In order to decide whether the internal bit b' of the DDH challenge is 0 (i.e., $g^z = g^{uv}$) or 1 (i.e., $g^z \xleftarrow{\$} \mathbb{G}$), $\mathcal{B}_1$ runs $\mathcal{A}$ as a subroutine on input g, g^u. $\mathcal{B}_1$ answers $\mathcal{A}$'s challenge query $x^\star$ with $(g^v, y^\star)$ where $y^\star \leftarrow \mathsf{F}(g^z, x^\star)$. Eventually, $\mathcal{A}$ outputs a bit b_{guess}, and $\mathcal{B}$ outputs the same guess.

Note, that if $\mathcal{A}$ can efficiently distinguish the games, i.e., detect whether $g^z = g^{uv}$ (*Game*$_0(\lambda)$) or not (*Game*$_1(\lambda)$), $\mathcal{B}_1$ can also efficiently solve its DDH challenge with the same advantage. Thus, we can bound the advantage by

$$\mathsf{Adv}^{\mathsf{nnPRF\text{-}ODH}}_{\mathbb{G},\mathsf{F},\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{G_1}}_{\mathbb{G},\mathsf{F},\mathcal{A}}(\lambda) + 2 \cdot \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G},\mathcal{B}_1}(\lambda).$$

Game$_2(\lambda)$**:** As the previous game, but this time we replace the challenge value $y_0^\star$ itself by a uniform random value, i.e., $y_0^\star \xleftarrow{\$} \{0,1\}^\lambda$. We show that if there exists an efficient adversary $\mathcal{A}$ that can distinguish *Game*$_1(\lambda)$ from *Game*$_2(\lambda)$, then there necessarily exists an efficient algorithm $\mathcal{B}_2$ that can break the pseudorandomness of F. We construct $\mathcal{B}_2$ as follows:

To initiate the environment for $\mathcal{A}$, algorithm $\mathcal{B}_2$ chooses some arbitrary group element g^u and forwards it to $\mathcal{A}$. At some point, $\mathcal{A}$ asks the challenge query $x^\star$, which $\mathcal{B}_2$ relays to its own $\mathcal{O}_{\mathsf{PRF}}$ oracle to receive a value $y_{b'}^\star$, which is either $F(K, x^\star)$ for some randomly chosen key $K \xleftarrow{\$} \mathbb{G}$ (if $b' = 0$) or $g(x^\star) \in \{0,1\}^\lambda$ for some random function g (if $b' = 1$). Algorithm $\mathcal{B}_2$ forwards $y_{b'}^\star$ along with an arbitrarily chosen group element g^v to $\mathcal{A}$. Eventually, $\mathcal{A}$ stops and outputs a bit b_{guess}. Algorithm $\mathcal{B}_2$ outputs the same bit as $\mathcal{A}$.

Note that $b' = 0$ corresponds to *Game*$_1(\lambda)$, whereas $b' = 1$ corresponds to the execution of *Game*$_2(\lambda)$. Therefore, if $\mathcal{A}$ can efficiently distinguish between the two games, $\mathcal{B}_2$ can distinguish between F values and independent random values with the same advantage. Hence, we can bound $\mathcal{A}$'s advantage by

$$\mathsf{Adv}^{\mathsf{G}_1}_{\mathbb{G},\mathsf{F},\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{G}_2}_{\mathbb{G},\mathsf{F},\mathcal{A}}(\lambda) + 2 \cdot \mathsf{Adv}^{\mathsf{prf\text{-}sec}}_{\mathsf{F},\mathcal{B}_2}(\lambda).$$

Since both $y_0^\star$ and $y_1^\star$ are now drawn independently and at random from $\{0,1\}^\lambda$, $\mathcal{A}$ cannot do better than guessing, i.e., $\mathsf{Adv}^{\mathsf{G}_2}_{\mathbb{G},\mathsf{F},\mathcal{A}}(\lambda) \leq 0$, which completes the proof. □

7.4.2 Random-Oracle Instantiation of mnPRF-ODH and nmPRF-ODH

For the original ODH assumption, Abdalla et al. [ABR01] proved that it is implied by the strong Diffie–Hellman assumption StDH in the random oracle model. In the following, we show that our strongest *one-sided* $\mathsf{PRF\text{-}ODH}$ variants, i.e., $\mathsf{mnPRF\text{-}ODH}$ and $\mathsf{nmPRF\text{-}ODH}$, can also be instantiated under the strong Diffie–Hellman assumption. Many proofs that implicitly used $\mathsf{PRF\text{-}ODH}$, have employed reductions to GapDH. Note that StDH is implied by the GapDH assumption, where the adversary can choose the first group element of the decision oracle freely.

Theorem 7.15 ($\mathsf{StDH} \implies \mathsf{mnPRF\text{-}ODH}, \mathsf{nmPRF\text{-}ODH}$)**.**
In the random oracle model, the strong DH assumption StDH *implies* $\mathsf{mnPRF\text{-}ODH}$ *security and* $\mathsf{nmPRF\text{-}ODH}$ *security of* $\mathsf{F}(K, x) = \mathsf{RO}(K, x)$ *for programmable random oracle* RO*. More precisely, for any efficient adversary* $\mathcal{A}$ *against the* $\mathsf{mnPRF\text{-}ODH}$ *or* $\mathsf{nmPRF\text{-}ODH}$ *security of* F*, there exists an efficient algorithm* $\mathcal{B}$ *each such that*

$$\mathsf{Adv}^{\mathsf{mnPRF\text{-}ODH}}_{\mathbb{G},\mathsf{F},\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{stDH}}_{\mathbb{G},\mathcal{B}}(\lambda) \qquad \textit{and} \qquad \mathsf{Adv}^{\mathsf{nmPRF\text{-}ODH}}_{\mathbb{G},\mathsf{F},\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{stDH}}_{\mathbb{G},\mathcal{B}}(\lambda).$$

We prove the following theorem for the case of $\mathsf{mnPRF\text{-}ODH}$ only; the case of $\mathsf{nmPRF\text{-}ODH}$ follows analogously, noting that by symmetry of the inputs g^u, g^v we can assume that the adversary $\mathcal{B}$ gets access to a $\mathsf{DDH}(g^v, \cdot, \cdot)$ decision oracle.

Proof. The proof is by straightforward reduction $\mathcal{B}$ from $\mathsf{mnPRF\text{-}ODH}$ to StDH. First, $\mathcal{B}$ obtains group elements g, g^u, g^v as challenge in the StDH game. To initiate the $\mathsf{mnPRF\text{-}ODH}$ game-environment for $\mathcal{A}$, $\mathcal{B}$ forwards g and g^u to $\mathcal{A}$ as input. $\mathcal{A}$ now has access to the oracles RO and ODH_u, i.e., $\mathcal{A}$ may send queries of the form (S, x) to the ODH_u oracle, with $S \in \mathbb{G}$ and x being some bit string. To provide an appropriate simulation, it must be ensured that, if $\mathcal{A}$ first queries some (S, x) to ODH_u and then (S^u, x) to the random oracle RO, the answer of RO is consistent with the simulation of ODH_u, and vice versa. This can be achieved if $\mathcal{B}$ can program the RO and has access to a decisional $\mathsf{DDH}(g^u, \cdot, \cdot)$ oracle.[7] $\mathcal{B}$ simulates the two oracles as follows:

[7]We remark that this is the reason why, in the given scenario, the computational DH assumption is not sufficient.

Simulation of RO. Repeating queries to RO return the same answer. This can be ensured by standard bookkeeping techniques. If a previously unseen query (K, x) is received, $\mathcal{B}$ must consider the case that $\mathcal{A}$ has already queried (S, x) with $K = S^u$ to ODH_u. Thus, when receiving a call (K, x) to RO, algorithm $\mathcal{B}$ queries its $\mathsf{DDH}(g^u, \cdot, \cdot)$ oracle on (S, K) for any group element S that has been queried with x to ODH_u. If the DDH oracle returns 1 on any such input then $\mathcal{B}$ answers consistently with the corresponding answer from earlier. Else, $\mathcal{B}$ assigns a fresh value y to (K, x) and returns y to $\mathcal{A}$.

Simulation of ODH_u. Analogously to the simulation of the random oracle, $\mathcal{B}$ checks each newly received request by $\mathcal{A}$ against all previous query-response pairs of ODH_u and answers consistently in case of repetition. If a previously unseen query (S, x) is received by ODH_u, $\mathcal{B}$ must further check whether the related value (S^u, x) has been queried to RO before. Similar to the reverse case, $\mathcal{B}$ uses its $\mathsf{DDH}(g^u, \cdot, \cdot)$ oracle on (S, K) on all previous RO queries (K, x) to detect this. If $\mathsf{DDH}(g^u, S, K) = 1$ for some K, the simulation of ODH_u answers with the respective output of RO. Otherwise, a response y is drawn uniformly at random from $\{0,1\}^\lambda$ and returned to $\mathcal{A}$ (and the tuple (S, x, y) stored for future reference).

At some point, $\mathcal{A}$ issues a challenge query $x^\star$ to its challenger. Algorithm $\mathcal{B}$ answers this query with g^v and some value $y^\star$, drawn uniformly at random from $\{0,1\}^\lambda$. Adversary $\mathcal{A}$ can now query ODH_u and RO further, with the limitation that it may not query the pair $(g^v, x^\star)$ to ODH_u. These queries are simulated as before.

Eventually, $\mathcal{A}$ stops and outputs a guess bit b_{guess}. $\mathcal{B}$ queries $\mathsf{DDH}(g^u, g^v, K)$ on all queries $(K, x^\star)$ of $\mathcal{A}$ to RO. If $\mathsf{DDH}(g^u, g^v, K) = 1$ for some RO-query $(K, x^\star)$, then $\mathcal{B}$ outputs K in the StDH game. Otherwise $\mathcal{B}$ aborts, admitting failure.

If the efficient adversary $\mathcal{A}$ wins the $\mathsf{mnPRF\text{-}ODH}$ game with non-negligible advantage, then $\mathcal{B}$ also outputs the correct value g^{uv} with non-negligible probability. To see this, we note that $\mathcal{A}$ can only win the $\mathsf{mnPRF\text{-}ODH}$ game in the random oracle model with non-negligible advantage if g^{uv} appears in one of its RO queries.

Assume that this is not the case. Then $\mathcal{A}$ expects $y^\star$ to be either $y_0^\star = \mathsf{RO}(g^{uv}, x^\star)$ or $y_1^\star \xleftarrow{\$} \{0,1\}^\lambda$. By the nature of random oracles (cf. Section 2.6), the two cases are indistinguishable for $\mathcal{A}$ since both are drawn uniformly at random from $\{0,1\}^\lambda$. This holds even if $\mathcal{A}$ could correctly determine the value g^{uv}, since it cannot compute $y_0^\star$ without querying the random oracle to compare with the received challenge. Thus, $\mathcal{A}$ *must* query g^{uv} to the random oracle in order to distinguish $y_0^\star$ from $y_1^\star$. But if $\mathcal{A}$ makes such a query (in the simulated game), then $\mathcal{B}$ also efficiently finds the DH value in the list of queries and correctly outputs it. ☐

7.4.3 Random-Oracle Instantiation of mmPRF-ODH

We next look at the case that the adversary can make multiple queries to both oracles, ODH_u and ODH_v. Interestingly, this does not follow straightforwardly from the StDH assumption as above. The reason is that, there, we have used the DDH oracle with fixed element g^u to check for consistency of ODH_u queries with random oracle queries.

In the most general $\mathsf{mmPRF\text{-}ODH}$ case, however, we also need to check consistency across ODH_u and ODH_v queries. In particular, a simulator needs to be able to check for queries (S, x) to ODH_u and (T, x) to ODH_v that result in the same key $S^u = K = T^v$. Yet, the simulator is only given S, T, g, g^u, and g^v. Such a test cannot be immediately performed with the $\mathsf{DDH}(g^u, \cdot, \cdot)$ oracle as in the StDH case, and not even with the more liberal $\mathsf{DDH}(\cdot, \cdot, \cdot)$ oracle as in the GapDH case. The above problem of having to identify S, T such that $S^u = T^v$, reminds of the so-called *claw finding* problem from complexity theory: given black-box access (i.e., oracles) to two functions $f_1 : A \to C$ and $f_2 : B \to C$ find $x \in A$ and $y \in B$ such that $f_1(x) = f_2(y)$.

ClawStDH. In our case, we do not need to find these elements S and T, but must be able to verify them for the functions $f_1(x) := x^u$ and $f_2(x) = x^v$. This leads us to augmenting the StDH problem with a claw-verifying oracle, i.e., an oracle which allows to check for elements S, T with $S^u = T^v$. We call this *claw-verifying oracle* Claw and the resulting DH problem the Claw-StDH problem (cf. Figure 7.11).

```
$\mathsf{G}^{\mathsf{Claw\text{-}StDH}}_{\mathbb{G},\mathcal{A}}(\lambda)$:
1 $u, v \xleftarrow{\$} \mathbb{Z}_q$
2 $S \xleftarrow{\$} \mathcal{A}^{\mathsf{DDH}(g^u,\cdot,\cdot),\mathsf{Claw}(\cdot,\cdot)}(g^u, g^v)$
3 return $[\![S = g^{uv}]\!]$

$\mathsf{DDH}(g^u, X, Y)$:
4 return $[\![X^u = Y]\!]$

$\mathsf{Claw}(X, Y)$:
5 return $[\![X^u = Y^v]\!]$

$\mathsf{G}^{\mathsf{sqDH}}_{\mathbb{G},\mathcal{A}}(\lambda)$:
1 $v \xleftarrow{\$} \mathbb{Z}_q$
2 $S \xleftarrow{\$} \mathcal{A}(g^v)$
3 return $[\![S = g^{v^2}]\!]$
```

Figure 7.11: Claw-StDH and sqDH problem.

For pairing-friendly groups $\mathbb{G}$ we get this decisional Claw oracle for free via the bilinear map e. $\mathsf{Claw}(S, T)$ then outputs 1, if $e(g^u, S) = e(g^v, T)$ and 0 otherwise. Next, we show that also for general groups $\mathbb{G}$ the claw-verifying oracle can be implemented in the StDH game, but at the cost of a loose security reduction to StDH.

The idea of representing the oracle Claw is as follows. Suppose that, in addition to g, g^u, and g^v we would also receive the value $g^{u/v}$ (where we assume here and in the following that $v \neq 0$, since the case $v = 0$ is trivial to deal with). Then we can run the check for claws via the stronger DDH oracle from the Gap DH problem by calling $\mathsf{DDH}(g^{u/v}, S, T)$, checking that $S^{u/v} = T$ and therefore $S^u = T^v$. We will see that we can relax the requirement to a $\mathsf{DDH}(g^u, \cdot, \cdot)$ oracle as given in the StDH assumption.

The crucial question remains if the computational problem of computing g^{uv} given $g^{u/v}$ (in the presence of a DDH oracle) becomes significantly easier. Switching to the square DH problem sqDH (cf. Figure 7.11) in an intermediate step, we show that this is *not* the case, although the intermediate step causes a loose security relationship.

Proposition 7.16 (SqDH $\implies$ Claw-StDH)**.** *If the square DH problem* sqDH *is hard in* $\mathbb{G}$*, then so is the claw strong DH problem* Claw-StDH*. More precisely, for any efficient adversary* $\mathcal{A}$ *against* Claw-StDH*, there exists an efficient algorithm* $\mathcal{B}$ *against* sqDH *such that*

$$\mathsf{Adv}^{\mathsf{claw\text{-}stDH}}_{\mathbb{G},\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{sqDH}}_{\mathbb{G},\mathcal{B}}(\lambda).$$

Proof. Assume that we have an algorithm $\mathcal{A}$ against Claw-StDH which on input (g, g^u, g^v) is able to compute g^{uv} with the help of oracle access to $\mathsf{DDH}(g^u, \cdot, \cdot)$ and the claw-verifying oracle Claw. Then we show that we can use this algorithm to build an algorithm $\mathcal{B}$ for the square DH problem sqDH (given g, g^v compute g^{v^2}) relative to a $\mathsf{DDH}(g^v, \cdot, \cdot)$ oracle.

For this, algorithm $\mathcal{B}$ for input g, g^v picks an $r \xleftarrow{\$} \mathbb{Z}_q^\star$ at random and sets $g^u = (g^v)^r$. With this choice, $g^{u/v} = g^r$ can be easily computed with the knowledge of r, allowing to implement the claw-verifying oracle quasi for free. Similarly, we have $\mathsf{DDH}(g^u, \cdot, \cdot) = \mathsf{DDH}(g^v, (\cdot)^r, \cdot)$, giving us the "mirrored" oracle for free. Algorithm $\mathcal{B}$ now runs $\mathcal{A}$ on input (g, g^u, g^v) and answers all oracle requests of $\mathcal{A}$ during the computation with the help of its $\mathsf{DDH}(g^v, \cdot, \cdot)$ oracle. Suppose that the adversary $\mathcal{A}$ eventually outputs K. Then, $\mathcal{B}$ returns $K^{1/r}$ which equals g^{v^2} for a correct answer $K = g^{uv} = g^{rv^2}$ of $\mathcal{A}$. □

Next, we show that from a solver for the square-DH problem (with $\mathsf{DDH}(g^v,\cdot,\cdot)$ oracle) we can build a solver for the StDH problem. Going from the square DH problem to CDH is already known. Interestingly, though, the common strategies in the literature [MW96, BDZ03, Gal12] require three calls to the square DH solver in order to compute the square $g^{(u+v)^2} = g^{u^2+2uv+v^2}$ and then to divide out g^{u^2} and g^{v^2}. Fortunately, two calls are sufficient, see for example [Kil01], yielding a tighter security bound.

Proposition 7.17 (StDH $\implies$ sqDH)**.** *If the strong DH problem* StDH *is hard in* $\mathbb{G}$*, then so is the square DH problem* sqDH*. More precisely, for any efficient adversary* $\mathcal{A}$ *against* sqDH*, there exists an efficient algorithm* $\mathcal{B}$ *against* StDH *such that*

$$\mathsf{Adv}^{\mathsf{sqDH}}_{\mathbb{G},\mathcal{A}}(\lambda) \leq \sqrt{\mathsf{Adv}^{\mathsf{stDH}}_{\mathbb{G},\mathcal{B}}(\lambda)}.$$

Proof. Suppose we have an efficient algorithm $\mathcal{A}$ against the square DH problem (with oracle $\mathsf{DDH}(g^v,\cdot,\cdot)$). A reduction $\mathcal{B}$ against the strong DH problem on input g^u and g^v can then call $\mathcal{A}$ once on g, g^{u+v} and once on $g, g^{r(u-v)}$ for some randomizer $r \xleftarrow{\$} \mathbb{Z}_q^\star$. Since both inputs are random and independent, we get two valid answers $g^{u^2+2uv+v^2}$ and $g^{r^2(u^2-2uv+v^2)}$ with the product of the square-DH algorithm's success probability. Note that these two executions at most double the number of oracle queries to the DDH oracle. Dividing out the exponent r^2 from the second term by raising it to the power $1/r^2$, and then dividing the two group elements, the reduction $\mathcal{B}$ then obtains g^{4uv} from which it can easily compute g^{uv} and thus solve its StDH challenge. $\square$

Overall, combining Proposition 7.16 and 7.17, it holds that solving the problem in presence of the decisional oracles for g^u and g^v, and an additional claw-verifying oracle, is implied by the StDH assumption, albeit with a security loss. More precisely, for any efficient adversary $\mathcal{A}$ against $\mathsf{Claw\text{-}StDH}$ we get an efficient adversary $\mathcal{B}$ (making at most twice as many calls to its StDH oracle as $\mathcal{A}$) such that

$$\mathsf{Adv}^{\mathsf{claw\text{-}stDH}}_{\mathbb{G},\mathcal{A}}(\lambda) \leq \sqrt{\mathsf{Adv}^{\mathsf{stDH}}_{\mathbb{G},\mathcal{B}}(\lambda)}.$$

We can now give our security proof for $\mathsf{mmPRF\text{-}ODH}$, which also implies security of $\mathsf{msPRF\text{-}ODH}$ and $\mathsf{smPRF\text{-}ODH}$:

Theorem 7.18. *In the random oracle model,* $\mathsf{Claw\text{-}StDH}$ *(resp.* StDH*) implies* $\mathsf{mmPRF\text{-}ODH}$ *security of* $\mathsf{F}(K,x)$ *modeled as a random oracle* RO*. More precisely, for any efficient adversary* $\mathcal{A}$ *against the* $\mathsf{mmPRF\text{-}ODH}$ *security of* F*, there exist efficient algorithms* $\mathcal{B}_1$ *and* $\mathcal{B}_2$ *such that*

$$\mathsf{Adv}^{\mathsf{mmPRF\text{-}ODH}}_{\mathbb{G},\mathsf{F},\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{claw\text{-}stDH}}_{\mathbb{G},\mathcal{B}_1}(\lambda) \leq \sqrt{\mathsf{Adv}^{\mathsf{stDH}}_{\mathbb{G},\mathcal{B}_2}(\lambda)}$$

Proof. The proof is almost identical to the one for $\mathsf{mnPRF\text{-}ODH}$, only that we here simulate the other oracle ODH_v as the oracle ODH_u, and for each query to either of the oracles also check via the help of Claw consistency between ODH_u and ODH_v evaluations. This provides a sound simulation of the random oracle. It follows as before that the adversary $\mathcal{A}$ can only distinguish genuine $y^\star$ from random ones if it queries the random oracle about g^{uv} (in the sound simulation), in which case $\mathcal{B}_1$ finds this value in the list of queries. From Propositions 7.16 and 7.17, we know that this advantage can further be bounded by an adversary $\mathcal{B}_2$ against the strong DH assumption. $\square$

7.5 The PRF-ODH Security of HMAC

HMAC is a keyed message authentication code which was introduced by Bellare, Canetti, and Krawczyk in 1996 [BCK96]. It has been standardized both by the IETF as RFC 2104 [KBC97] and NIST as FIPS Publiction 198-1 [NIS08] and is used in major internet protocols such as TLS and IPsec. The construction of HMAC is based on Merkle-Damgård hashes [Dam90, Mer90], which iterate a compression function h over arbitrary-length inputs. Given appropriate padding Merkle-Damgård-based hash functions inherit their collision-resistance from the underlying compression function. Similarly, HMAC has been shown to be a PRF, if the underlying compression function is a PRF [Bel06, Bel15].

In this section we examine the PRF-ODH security of HMAC, complementing previous results on the PRF security of HMAC [CDMP05, Kra10, BL15]. In particular, we show that $\mathsf{HMAC}(K, x)$ as well as its dual-PRF usage $\mathsf{HMAC}(x, K)$ that is keyed on the *second* input, as encountered in TLS 1.3 [Res18] (see below), is mmPRF-ODH secure, which is our strongest notion of PRF-ODH security.

Description of HMAC

The basic construction of HMAC is illustrated in Figure 7.12. Let $h : \{0,1\}^c \times \{0,1\}^b \to \{0,1\}^c$ be the underlying compression function and $\mathsf{H} : \{0,1\}^\star \to \{0,1\}^c$ the iterated Merkle-Damgård hash. The iterated compression function is denoted by $h^\star : \{0,1\}^c \times B^+ \to \{0,1\}^c$, where B^+ is the set of all bit strings of length $n \cdot b$ with $n \in \mathbb{N}^+$. On input key $K \in \{0,1\}^b$ and message $\widetilde{m} = m_1 m_2 \ldots m_n$ of n b-bit blocks, the output of $h^\star$ (the upper chain in Figure 7.12) is a_n computed as

$$\begin{aligned} a_0 &\leftarrow K \\ a_1 &\leftarrow h(a_0, m_1) \\ &\vdots \\ a_n &\leftarrow h(a_{n-1}, m_n). \end{aligned}$$

With the above convention we have that $\mathsf{H}(m) = h^\star(\mathsf{IV}, \widetilde{m})$, where $\mathsf{IV} \in \{0,1\}^c$ is the initialization vector fixed by the description of H and $\widetilde{m} \in B^+$ the message M padded to a multiple of the block size b. Finally, for key $K \in \{0,1\}^b$ and label x, HMAC is defined as $\mathsf{HMAC}(K, x) := \mathsf{H}(K \oplus \mathsf{opad} || \mathsf{H}(K \oplus \mathsf{ipad} || x))$, where opad and ipad are fixed (distinct) constants in $\{0,1\}^b$. In terms of the iterated compression function we have

$$\mathsf{HMAC}(K, x) = h^\star(\mathsf{IV}, K \oplus \mathsf{opad} || h^\star(\mathsf{IV}, K \oplus \mathsf{ipad} || x || \mathsf{padding}) || \mathsf{padding}).$$

Deviating key lengths. HMAC is in general also defined for keys whose lengths differ from the block size b. The minimal requirements from a security point of view is a length of c bits, i.e., the output length of the underlying hash function H. Keys of this minimal length are simply padded to the correct length. Shorter keys are not recommended.

Longer keys K are first hashed down to $\mathsf{H}(K) \in \{0,1\}^c$ and then padded. For ease of notation, we introduce the auxiliary function HMAC' that takes as input keys of length greater than b and is defined as $\mathsf{HMAC}'(K, x) := \mathsf{HMAC}(\widetilde{L}, x)$ where $L := \mathsf{H}(K) \in \{0,1\}^c$ and $\widetilde{L} \in \{0,1\}^b$ is the padded value of L (cf. Figure 7.13). We will see that it is beneficial to consider this more general case when lifting results about the PRF-ODH security of HMAC to statements about the key derivation function HKDF, which uses HMAC as dual PRF (cf. Section 7.5.2).

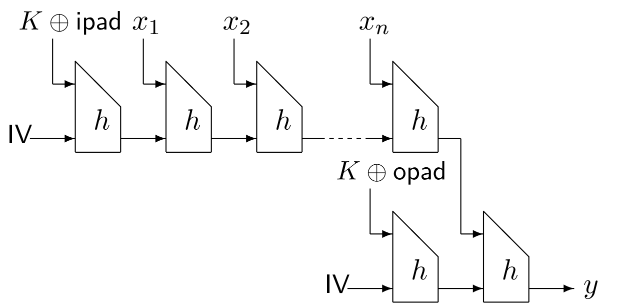

Figure 7.12: Illustration of computation of $y \leftarrow \mathsf{HMAC}(K, x)$ with key $K \in \{0,1\}^b$ and label $x = x_1x_2\ldots x_n$ of size nb.

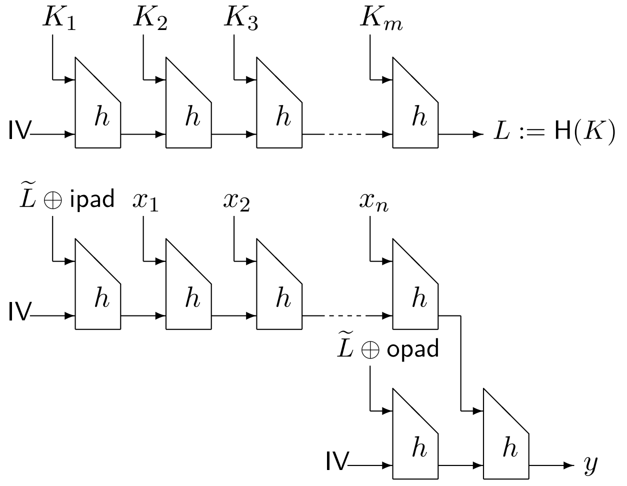
K_1
K_2
K_3
K_m
IV
h
$L := \mathsf{H}(K)$
$\widetilde{L} \oplus \mathsf{ipad}$
x_1
x_2
x_n
$\widetilde{L} \oplus \mathsf{opad}$
y

Figure 7.13: Illustration of computation of $y \leftarrow \mathsf{HMAC}(K, x)$ with key K consisting of m blocks $K_1K_2\ldots K_m$ of size b and label $x = x_1x_2\ldots x_n$ of size nb.

7.5.1 PRF-ODH Security of HMAC

Theorem 7.19. *Assume that the underlying compression function $h : \{0,1\}^c \times \{0,1\}^b \to \{0,1\}^c$ of* HMAC *is a random oracle. Then* HMAC *is* $\mathsf{mmPRF\text{-}ODH}$*-secure under the* StDH *assumption. More precisely, for any efficient adversary $\mathcal{A}$ against the* $\mathsf{mmPRF\text{-}ODH}$ *security of* HMAC*, there exists an efficient algorithm $\mathcal{B}$ such that*

$$\mathsf{Adv}^{\mathsf{mmPRF\text{-}ODH}}_{\mathbb{G},\mathsf{HMAC},\mathcal{A}}(\lambda) \leq \sqrt{\mathsf{Adv}^{\mathsf{stDH}}_{\mathbb{G},\mathcal{B}}(\lambda)} + (q_{\mathsf{RO}} + (q_{\mathsf{ODH}_u} + q_{\mathsf{ODH}_v}) \cdot \ell_{\mathsf{ODH}} + 1)^2 \cdot 2^{-c}$$

where q with the respective index denotes the maximal number of the corresponding oracle queries, and ℓ_{ODH} the maximal number of oracle calls to h in each evaluation of any ODH *oracle call.*

Proof. Let the compression function h underlying HMAC be modeled as a random oracle RO. We note that we can assume each output of the compression function (genuine or simulated, as below) to be unique and to never hit the initialization vector IV. This assumption is reflected in the loss of the factor $(q_{\mathsf{RO}} + (q_{\mathsf{ODH}_u} + q_{\mathsf{ODH}_v}) \cdot \ell_{\mathsf{ODH}} + 1)^2 \cdot 2^{-c}$ in the above security statement, applying the birthday bound to the maximal number of queries (adding 1 for IV) and noting that in the simulation we simulate the same number of random oracle evaluations as in the actual attack.

Evaluation chains. A key insight for unique outputs of the compression function is that we can determine *evaluation chains* in a list of random oracle queries and answers. That is, starting from any input/output pair of the random oracle, we can try to go backwards along the HMAC iteration via the unique pre-images in the table, to check if we end up at a value (IV, L') for the hash function's initialization vector IV. Since we assume that no random oracle evaluation yields IV we can easily identify the beginning of such a chain. This holds for both the inner and outer branch of the HMAC evaluation such that we can check if a value has been derived as a full HMAC evaluation, for the key value L. From there on we can also check if we have a full evaluation chain of the key (if the key is larger than the block size). To distinguish the two cases we call the former an HMAC evaluation chain for key value L, and the latter a *complete* HMAC evaluation chain. They coincide for keys which fit into the block size b.

For evaluation chains we can also extract the input string (and possibly the key input). In particular, at any point during the simulation we can take any entry in the simulated random oracle table and determine if there is an evaluation chain for this entry. This implies that we can determine all existing evaluation chains at any point in time. For some of such completed chains, we will only know an implicit presentation $[Q, R]$ of the key K with $\mathsf{DDH}(Q, R, K) = 1$, as in the proof of Theorem 7.18.

Reduction to StDH. We show that if there exists an adversary $\mathcal{A}$ against the $\mathsf{mmPRF\text{-}ODH}$ security of HMAC, then there necessarily also exists an adversary $\mathcal{B}$ that can break StDH with the corresponding advantage. In the following we discuss the case where the group $\mathbb{G}$ is such that its canonical bit string representation exceeds b bits and we use upstream hashing of the key. The cases concerning keys from groups with block-sized representations (or even shorter) can be proven analogously.

The StDH adversary $\mathcal{B}$ simulates the $\mathsf{mmPRF\text{-}ODH}$ environment for $\mathcal{A}$ and programs the random oracle. In addition to the $\mathsf{DDH}(g^u, \cdot, \cdot)$, oracle we assume that $\mathcal{B}$ has a claw-verifying oracle Claw which for inputs S, T checks that $S^u = T^v$, and that it can simulate the $\mathsf{DDH}(g^v, \cdot, \cdot)$ oracle. These extra oracles are accounted for as in the monolithic random oracle case by using the square root of the advantage against StDH.

Once $\mathcal{B}$ has obtained its challenge (g, g^u, g^v), it runs the $\mathsf{mmPRF\text{-}ODH}$ adversary $\mathcal{A}$ as a sub-routine on input (g, g^u). $\mathcal{A}$ then has access to the oracles RO and ODH_u (and later also ODH_v). Here, the random oracle RO corresponds to the compression function h and thus takes inputs of the form $(A, x) \in \{0,1\}^c \times \{0,1\}^b$. Oracle ODH_u takes as input $(A, x) \in \mathbb{G} \times \{0,1\}^\star$ and is supposed to return a value corresponding to $\mathsf{HMAC}(A^u, x)$. $\mathcal{B}$ must provide sound simulations of these oracles. This is done as follows:

Simulation of ODH_u. Repeated queries (S, x) to ODH_u are answered consistently by returning the same value as before. Thus, in the following we can assume that the received query of the form (S, x) has not been queried to ODH_u beforehand. Else, algorithm $\mathcal{B}$ first checks if there already exists an evaluation chain for a pair (K, x) such that $K = S^u$; this can be verified with the DDH oracle. If this is the case, $\mathcal{B}$ answers consistently with the final output of the evaluation chain. If not, it verifies if there is an implicit key $K = [Q, R]$ with $Q = g^u$ and $R = S$, or with $Q = g^v$ and $R^v = S^u$, where the former can be checked directly and the latter can be verified with the help of the oracle Claw.

If $\mathcal{B}$ finds a matching key according to one of the cases above, it looks up the corresponding key value L as before. If, on the other hand, there is no (explicitly or implicitly) matching key, then $\mathcal{B}$ sets $\mathsf{H}([g^u, S]) \leftarrow L$ to a uniformly random value $L \in \{0,1\}^c$. It stores the implicit key $[g^u, S]$ with the value L for future use.

$\mathcal{B}$ then iterates the HMAC computation to get the return value y with key L by calling its (simulated) oracle RO an all values. More precisely, $\mathcal{B}$ computes $y := \mathsf{RO}^\star(\mathsf{IV}, \widetilde{L} \oplus \mathsf{opad}||\mathsf{RO}^\star(\mathsf{IV}, \widetilde{L} \oplus \mathsf{ipad}||x||\mathsf{padding})||\mathsf{padding})$, where $\widetilde{L} = L||\mathsf{padding}$, and $\mathcal{B}$ then returns y as response. Note that, if the evaluation chain had already been computed before, the outcome of this evaluation is consistent with the previous result.

Simulation of ODH_v. Analogously to ODH_u.

Simulation of RO. Outputs of RO for equal input queries are answered consistently. If a previously unseen query, say, $(\widetilde{A}, \widetilde{x})$ is received, then $\mathcal{B}$ must consider if $\mathsf{RO}(\widetilde{A}, \widetilde{x})$ completes an $\mathsf{H}([g^u, S])$ (or $\mathsf{H}([g^v, S])$) computation for only implicitly known key $K = S^u$ (or $K = S^v$) that has been set beforehand to a uniformly random value, say, L (see the simulation of oracle $\mathsf{ODH}_u/\mathsf{ODH}_v$). This can be checked again with the DDH oracle. If this is the case, then the reduction answers the query consistently with value L. Otherwise, a response y is drawn uniformly at random from $\{0,1\}^c$ and returned.

At some point, $\mathcal{A}$ issues a challenge query $x^\star$. The reduction emulates the challenger by replying with g^v and some value $y^\star$, drawn uniformly at random from $\{0,1\}^c$. $\mathcal{A}$ can now query ODH_u and RO further, with the sole limitation that it may not query the pair $(g^v, x^\star)$ to ODH_u. Additionally, $\mathcal{A}$ acquires access to the ODH_v oracle which is simulated analogously to the ODH_u oracle and may not be queried with $(g^u, x^\star)$. Eventually, $\mathcal{A}$ stops and outputs a guess bit b'. If $\mathsf{DDH}(g^u, g^v, \widetilde{K}) = 1$ for some completed chain of RO queries with associated key $\widetilde{K}$, $\mathcal{B}$ outputs $\widetilde{K}$ in the StDH game.

The rest of the proof is as before, given that the simulation of all oracle queries is sound. That is, $\mathcal{B}$ outputs the correct value g^{uv} with high probability if $\mathcal{A}$ wins $\mathsf{mmPRF\text{-}ODH}$ with non-negligible advantage. We show this by arguing that $\mathcal{A}$ can win the $\mathsf{mmPRF\text{-}ODH}$ game in the random oracle model with non-negligible advantage if and only if g^{uv} is an input key of a completed chain of RO queries with input (padded) label $x^\star$. To this end note that $\mathcal{A}$ expects $y^\star$ to be either $y_0^\star \leftarrow \mathsf{HMAC}(g^{uv}, x^\star)$ or $y_1^\star \xleftarrow{\$} \{0,1\}^c$. By the nature of random oracles, $y_0^\star$ and $y_1^\star$ are indistinguishable for $\mathcal{A}$ since both are drawn uniformly at random from $\{0,1\}^c$. Even if $\mathcal{A}$ can correctly determine the value g^{uv}, it cannot compute $y_0^\star$ by itself to compare with the received challenge. Thus, $\mathcal{A}$ must iteratively query RO on the full $\mathsf{HMAC}(g^{uv}, x^\star)$ computation, including the key g^{uv}, in order to distinguish $y_0^\star$ and $y_1^\star$. Furthermore, $\mathcal{B}$ is efficient, since $\mathcal{A}$ is efficient and asks at most polynomially many queries to each oracle. $\square$

7.5.2 Application to HKDF

As mentioned earlier, one specific use case of the $\mathsf{PRF\text{-}ODH}$ assumption arises in the setting of TLS 1.3. Here, the HKDF scheme [Kra10, KE10] is adapted for key derivation. In particular, the function $\mathsf{HKDF.Extract}$ is used to derive an internal key K' as

$$K' \leftarrow \mathsf{HKDF.Extract}(x, K) := \mathsf{HMAC}(x, K),$$

where an adversarially known value x is used as the HMAC key while the secret randomness source in the form of a DH shared secret $K = g^{uv}$ is used as the label. At a first glance, this swapping of inputs may seem odd. However, the specified purpose of $\mathsf{HKDF.Extract}$ is to extract uniform randomness from its *second* component.

One way to prove that K' is indeed a random key (as long as g^{uv} is not revealed to the adversary) is to model $\mathsf{HKDF.Extract}(x, \cdot)$ as a random oracle. An alternative approach is pursued in [DFGS15b, DFGS16, FG17] where the authors prove the statement under the assumption that $\mathsf{HKDF.Extract}(XTS, IKM) = \mathsf{HMAC}(XTS, IKM)$ is $\mathsf{PRF\text{-}ODH}$ secure when understood as a PRF keyed with $IKM \in \mathbb{G}$ (i.e., when the key is the *second* input). In this light, it is beneficial to show that $\mathsf{HMAC}(x, K)$ remains PRF-ODH secure for key $K \in \mathbb{G}$ and $x \in \{0,1\}^{\star}$.[8] Fortunately, our general treatment of $\mathsf{HMAC}(K, x)$ in Theorem 7.19 with arbitrarily long keys allows us to conclude the analogous result for $\mathsf{HMAC}(x, K)$ with swapped key and label.

Corollary 7.20. *Let $h : \{0,1\}^c \times \{0,1\}^b \to \{0,1\}^c$ be the underlying compression function of* HMAC*. If h is modeled as a random oracle, then* $\mathsf{HMAC}'(K, x) := \mathsf{HMAC}(x, K)$ *is* mmPRF-ODH *secure under* StDH*.*

Alternative approaches. Alternatively, one may wish to prove Theorem 7.19 along the results established by Coron et al. [CDMP05], who showed that if the compression function h is modeled as a random oracle then a variant of HMAC can be shown to be *indifferentiable from a random oracle* in the sense of Maurer et al. [MRH04]. Krawczyk [Kra10] mentions that the above result can also immediately be applied to the unmodified plain HMAC design. However, we believe that providing a detailed proof of Theorem 7.19 is nevertheless beneficial since it enables us to argue in a straightforward manner that the “reversed” result presented in Corollary 7.20, i.e., $\mathsf{HMAC}(x, K)$ with swapped label and key as inputs, also holds. Else one would need to check if HMAC with a hashed key would also be a random oracle.

In the NIST hash function competition it has been established that sponge-based constructions can be used to build cryptographic hash functions. We are confident that the proof of Theorem 7.19 can be adapted to achieve the same result for HMAC if the underlying cryptographic hash function H is replaced by a sponge-based construction such as SHA3-256 with the random permutation π modeled as a random oracle.[9] This proof can also be established along the lines of Bertoni et al. [BDPV08] who provide results showing that the sponge construction is indifferentiable from a random oracle when being used with a random transformation or a random permutation.

7.6 Impossibility Result

To close the chapter, we will briefly state for completeness the impossibility result from [BFGJ17a], which basically shows that it is implausible to assume that even the mild, one-sided variants of lrPRF-ODH with only a single query can be instantiated in the standard model:

Theorem 7.21. *Assume that there is an efficient algebraic black-box reduction $\mathcal{R}$ from the* snPRF-ODH *(or* nsPRF-ODH*) assumption to a* DDH*-augmented problem. Then either the* DDH*-augmented problem is not hard, or the decisional square DH problem is not hard.*

The proof of Theorem 7.21 is conducted via a meta-reduction technique [GMR88, BV98, PV05] and can be found in the paper [BFGJ17a]. The general proof idea is as follows.

Proof sketch. Assume that we have an algebraic reduction $\mathcal{R}$ from the snPRF-ODH assumption which turns any black-box adversary into a solver for a DDH-augmented problem (a problem from a class of hard cryptographic problems). Then we can in particular consider an inefficient adversary $\mathcal{A}_\infty$ which successfully breaks the snPRF-ODH assumption with constant probability. The reduction, with black-box access to $\mathcal{A}_\infty$, must then solve the DDH-augmented problem.

For this it can then either not take any advantage of the infinite power of $\mathcal{A}_\infty$—in which case we can already break the DDH-augmented problem—or it tries to elicit some useful information

[8] Though formally defined for arbitrary length, recall that the minimal recommended length is c bits.

[9] SHA3-256 is part of the Keccak sponge function family [BDPA11]. It has been standardized in the FIPS Publication 202 [NIS15], wherein it is explicitly approved for usage in HMAC.

from $\mathcal{A}_\infty$. In the latter case we build our meta-reduction by simulating $\mathcal{A}_\infty$ *efficiently*. This is accomplished by exploiting the algebraic property of the reduction and "peeking" at the internals of the reduction's group element choices. Our meta-reduction will then solve the decisional square DH problem, which says that (g, g^a, g^{a^2}) is indistinguishable from (g, g^a, g^b) for random a, b. $\square$

Our impossibility result works for pseudorandom functions F, which take as input arbitrary bit strings $\{0,1\}^\star$ and map them to λ bits. We stick with this convention here, but remark that our negative result also holds if the input length is 1 only, and the output length is super-logarithmic in λ. Similarly, we assume that F is $\mathsf{nnPRF\text{-}ODH}$ secure, although it suffices for our negative result that the function F for a random group element (and some fixed input, say 1) is pseudorandom, i.e., that $\mathsf{F}(X, 1)$ is indistinguishable from random for a uniformly chosen group element $X \xleftarrow{\$} \mathbb{G}$ (without giving any "Diffie–Hellman decomposition" of X).

We note that the impossibility result does not only apply to the schemes analyzed with respect to the $\mathsf{PRF\text{-}ODH}$ assumption, but potentially also to other works where the general Gap Diffie–Hellman GapDH or related assumptions in the random oracle have been used for the analysis, yet where the $\mathsf{PRF\text{-}ODH}$ assumption is a promising alternative for carrying out a proof. Examples include the QUIC protocol [FG14, LJBN15] and OPTLS [KW16] which forms the base of TLS 1.3.

Chapter 8

Conclusion

The break of cryptographic hardness assumptions and primitives is probably as old of a phenomenon as cryptography itself. These breaks occur mostly as a result of more sophisticated cryptanalytic techniques and/or the steady increase in computational resources. However, more often than not, we face the situation that broken algorithms are still widely deployed due to traditionally slow adoption rates for new cryptographic algorithms and protocol versions. Answers to the following question are thus needed - both in retrospect as well as in anticipation of the break of cryptographic primitives: “Must one consider session keys (and thus the following encrypted communication) as compromised if they were established in key exchanges that employed a then unbroken, but now broken primitive X?”

With the notion of breakdown resilience we gave the first model for authenticated key exchange that could answer this type of question for arbitrary cryptographic primitives and hardness assumptions. An extension of the model also captures the security of ongoing and future sessions, and is especially useful for analyzing generic hybrid constructions that utilize built-in redundancies to ensure robustness in the case of failure of one of the algorithms.

We looked more specifically at such hybrid constructions for key exchange protocols that aim for robustness against adversaries that may gain access to quantum computing power. We presented a framework that is able to differentiate between varying levels of quantum adversaries (e.g., between future-quantum and post-quantum adversaries) to analyze the security of hybrid key encapsulation mechanisms and AKE protocols. We furthermore designed three novel, provably-secure constructions for hybrid key encapsulation mechanisms that can withstand post-quantum adversaries.

Future-proofing key exchanges does however not only encompass actively guarding against future adversaries but also basing analyses on well-understood hardness assumptions. A careful categorization of hardness assumptions minimizes the risk of security claims becoming meaningless due to untenable or uninstantiable assumptions.

An example for a fairly new assumption in key exchange protocols is the PRF-ODH assumption, which is commonly employed in security proofs of Diffie–Hellman-based key exchanges since its introduction in 2012 by Jager et al. [JKSS12]. Despite its popularity, the various variants of the PRF-ODH assumption had not received a unified formalization and in-depth analysis as to their strength. We thus gave a unified definition encompassing (most) variants present in the literature and systematically categorized it by firmly embedding it into the ecosystem of known DH-type assumptions. Additionally, there was uncertainty as to whether the employment of PRF-ODH allows to circumvent the random oracle methodology. We could show that this is unfortunately implausible, even for the mildest one-sided notions.

How to go on from here?

The need for cryptographic agility and modular approaches has become a key wish list item for many new protocol designs. This hopefully supports a swifter replacement of broken cryptographic primitives and protocol versions in the future. Especially the transition to post-quantum secure algorithms (most likely via hybrid constructions) is a fruitful area for both further “pen-and-paper” research as well as hands-on involvement by providing implementations and benchmarks. It is imperative to further map out the concrete challenges that occur in real-world deployment of cryptographic algorithms within protocols, as, e.g., the question of how to deal with interoperability of hybrid-aware and non-hybrid systems, as well as the perpetual issue of (un-updatable) legacy systems.

It is worth noting, that in order to transition to post-quantum algorithms we actually need to migrate systems *twice*, first to hybrid designs and eventually back to single-algorithm designs that are solely post-quantum. Cryptographic agility and the development of seamless migration tools are therefore key. While major internet protocols such as TLS and SSH have already begun to gain some attention with respect to these issues, eventually all of our protocols need to make the shift. In particular it will be interesting to investigate the combination of hybrid designs for key exchange and authentication.

Furthermore, one hybrid solution does not fit all. Hybrids necessarily introduce some overhead—in computation, storage, and communication. So far, there is little consideration on how we can build dedicated lightweight hybrid designs that can be supported by resource-constrained devices, for example in the IoT setting.

With respect to the more foundational nature of security models and hardness assumptions we note that the breakdown resilience framework may be added to different AKE model designs. It is conceivable that the approach may even be beneficial in other types of protocol analyses such as, e.g., for secure channel protocols. For $\mathsf{PRF\text{-}ODH}$ it would be nice to fill in the missing separation results such that the hierarchy between the various notions is established completely. For non-DH-based key exchanges it may be interesting to see whether analogues of the $\mathsf{PRF\text{-}ODH}$ assumption are necessary and how these can be defined and instantiated.

On a more general note, the paradigm of provable security is a success story. However, the story does not end there. Personally, I find it highly interesting how the different foundational approaches within the paradigm (e.g., manual game-based proofs and automated formal analysis) can be composed to achieve stronger security guarantees of the analysed protocol.

Bibliography

[AAB$^+$19] Erdem Alkim, Roberto Avanzi, Joppe Bos, Léo Ducas, Antonio de la Piedra, Thomas Pöppelmann, Peter Schwabe, and Douglas Stebila. Newhope algorithm specifications and supporting documentation 1.03. `https://newhopecrypto.org/data/NewHope_2019_07_10.pdf`, July 2019. Accessed: 2019-09-13.

[AB81] C.A. Asmuth and G.R. Blakley. An efficient algorithm for constructing a cryptosystem which is harder to break than two other cryptosystems. *Computers and Mathematics with Applications*, 7(6):447 – 450, 1981.

[ABD$^+$15] David Adrian, Karthikeyan Bhargavan, Zakir Durumeric, Pierrick Gaudry, Matthew Green, J. Alex Halderman, Nadia Heninger, Drew Springall, Emmanuel Thomé, Luke Valenta, Benjamin VanderSloot, Eric Wustrow, Santiago Zanella-Béguelin, and Paul Zimmermann. 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.

[ABP$^+$13] Nadhem J. AlFardan, Daniel J. Bernstein, Kenneth G. Paterson, Bertram Poettering, and Jacob C. N. Schuldt. On the security of RC4 in TLS. In Samuel T. King, editor, *USENIX Security 2013: 22nd USENIX Security Symposium*, pages 305–320, Washington, DC, USA, August 14–16, 2013. USENIX Association.

[ABR01] Michel Abdalla, Mihir Bellare, and Phillip Rogaway. The oracle Diffie-Hellman assumptions and an analysis of DHIES. In David Naccache, editor, *Topics in Cryptology – CT-RSA 2001*, volume 2020 of *Lecture Notes in Computer Science*, pages 143–158, San Francisco, CA, USA, April 8–12, 2001. Springer, Heidelberg, Germany.

[ACD$^+$18] Martin R. Albrecht, Benjamin R. Curtis, Amit Deo, Alex Davidson, Rachel Player, Eamonn W. Postlethwaite, Fernando Virdia, and Thomas Wunderer. Estimate all the {lwe, ntru} schemes! In Dario Catalano and Roberto De Prisco, editors, *Security and Cryptography for Networks*, pages 351–367, Cham, 2018. Springer International Publishing.

[ADPS16a] Erdem Alkim, Léo Ducas, Thomas Pöppelmann, and Peter Schwabe. NewHope without reconciliation. Cryptology ePrint Archive, Report 2016/1157, 2016. `http://eprint.iacr.org/2016/1157`.

[ADPS16b] 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.

[BBF$^+$16] Karthikeyan Bhargavan, Christina Brzuska, Cédric Fournet, Matthew Green, Markulf Kohlweiss, and Santiago Zanella-Béguelin. Downgrade resilience in key-exchange protocols. In *2016 IEEE Symposium on Security and Privacy*, pages 506–525, San Jose, CA, USA, May 22–26, 2016. IEEE Computer Society Press.

[BBF$^+$18] Nina Bindel, Jacqueline Brendel, Marc Fischlin, Brian Goncalves, and Douglas Stebila. Hybrid key encapsulation mechanisms and authenticated key exchange. Cryptology ePrint Archive, Report 2018/903, 2018. https://eprint.iacr.org/2018/903.

[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*, pages 206–226, Cham, 2019. Springer International Publishing.

[BBM$^+$19] Richard Barnes, Benjamin Beurdouche, Jon Millican, Emad Omara, Katriel Cohn-Gordon, and Raphael Robert. The Messaging Layer Security (MLS) Protocol. Internet-Draft draft-ietf-mls-protocol-07, Internet Engineering Task Force, 2019. Work in Progress, accessed 2019-09-13.

[BCGP08] Colin Boyd, Yvonne Cliff, Juan González Nieto, and Kenneth G. Paterson. Efficient one-round key exchange in the standard model. In Yi Mu, Willy Susilo, and Jennifer Seberry, editors, *ACISP 08: 13th Australasian Conference on Information Security and Privacy*, volume 5107 of *Lecture Notes in Computer Science*, pages 69–83, Wollongong, Australia, July 7–9, 2008. Springer, Heidelberg, Germany.

[BCK96] Mihir Bellare, Ran Canetti, and Hugo Krawczyk. Keying hash functions for message authentication. In Neal Koblitz, editor, *Advances in Cryptology – CRYPTO'96*, volume 1109 of *Lecture Notes in Computer Science*, pages 1–15, Santa Barbara, CA, USA, August 18–22, 1996. Springer, Heidelberg, Germany.

[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 Jeffrey Scott Vitter, editor, *30th Annual ACM Symposium on Theory of Computing*, pages 419–428, Dallas, TX, USA, May 23–26, 1998. ACM Press.

[BCNS15] Joppe W. Bos, Craig Costello, Michael Naehrig, and Douglas Stebila. Post-quantum key exchange for the TLS protocol from the ring learning with errors problem. In *2015 IEEE Symposium on Security and Privacy*, pages 553–570, San Jose, CA, USA, May 17–21, 2015. IEEE Computer Society Press.

[BD16] Jacqueline Brendel and Denise Demirel. Efficient Proactive Secret Sharing. In *2016 14th Annual Conference on Privacy, Security and Trust (PST)*, pages 543–550, Dec 2016.

[BDF$^+$11] Dan Boneh, Özgür Dagdelen, Marc Fischlin, Anja Lehmann, Christian Schaffner, and Mark Zhandry. Random oracles in a quantum world. In Dong Hoon Lee and Xiaoyun Wang, editors, *Advances in Cryptology – ASIACRYPT 2011*, volume 7073 of *Lecture Notes in Computer Science*, pages 41–69, Seoul, South Korea, December 4–8, 2011. Springer, Heidelberg, Germany.

[BDPA11] G. Bertoni, J. Daemen, M. Peeters, and G. Van Assche. The Keccak SHA-3 submission. Submission to NIST (Round 3), 2011.

[BDPV08] Guido Bertoni, Joan Daemen, Michael Peeters, and Gilles Van Assche. On the indifferentiability of the sponge construction. In Nigel P. Smart, editor, *Advances in Cryptology – EUROCRYPT 2008*, volume 4965 of *Lecture Notes in Computer Science*, pages 181–197, Istanbul, Turkey, April 13–17, 2008. Springer, Heidelberg, Germany.

[BDZ03] Feng Bao, Robert H. Deng, and Huafei Zhu. Variations of Diffie-Hellman problem. In Sihan Qing, Dieter Gollmann, and Jianying Zhou, editors, *ICICS 03: 5th International Conference on Information and Communication Security*, volume 2836 of *Lecture Notes in Computer Science*, pages 301–312, Huhehaote, China, October 10–13, 2003. Springer, Heidelberg, Germany.

[Bel06] Mihir Bellare. New proofs for NMAC and HMAC: Security without collision-resistance. In Cynthia Dwork, editor, *Advances in Cryptology – CRYPTO 2006*, volume 4117 of *Lecture Notes in Computer Science*, pages 602–619, Santa Barbara, CA, USA, August 20–24, 2006. Springer, Heidelberg, Germany.

[Bel15] Mihir Bellare. New proofs for NMAC and HMAC: Security without collision resistance. *Journal of Cryptology*, 28(4):844–878, October 2015.

[Ber19] Daniel J. Bernstein. Comparing proofs of security for lattice-based encryption. Cryptology ePrint Archive, Report 2019/691 and in *NIST 2nd Post-Quantum Cryptography Standardization Conference 2019*, August 2019, Santa Barbara, 2019. https://eprint.iacr.org/2019/691.

[BF17] Jacqueline Brendel and Marc Fischlin. Zero round-trip time for the extended access control protocol. In Simon N. Foley, Dieter Gollmann, and Einar Snekkenes, editors, *ESORICS 2017: 22nd European Symposium on Research in Computer Security, Part I*, volume 10492 of *Lecture Notes in Computer Science*, pages 297–314, Oslo, Norway, September 11–15, 2017. Springer, Heidelberg, Germany.

[BFG17] Jacqueline Brendel, Marc Fischlin, and Felix Günther. Breakdown resilience of key exchange protocols and the cases of NewHope and TLS 1.3. Cryptology ePrint Archive, Report 2017/1252, 2017. https://eprint.iacr.org/2017/1252.

[BFG19] Jacqueline Brendel, Marc Fischlin, and Felix Günther. Breakdown Resilience of Key Exchange Protocols and the Cases of NewHope and TLS 1.3. In Kazue Sako, Steve Schneider, and Peter Y.A. Ryan, editors, *Computer Security – ESORICS 2019*, pages 521–541, Cham, 2019. Springer. Also available as Cryptology ePrint Archive Report 2017/1252, https://eprint.iacr.org/2017/1252.

[BFGJ17a] Jacqueline Brendel, Marc Fischlin, Felix Günther, and Christian Janson. PRF-ODH: Relations, instantiations, and impossibility results. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology – CRYPTO 2017, Part III*, volume 10403 of *Lecture Notes in Computer Science*, pages 651–681, Santa Barbara, CA, USA, August 20–24, 2017. Springer, Heidelberg, Germany.

[BFGJ17b] Jacqueline Brendel, Marc Fischlin, Felix Günther, and Christian Janson. PRF-ODH: Relations, instantiations, and impossibility results. Cryptology ePrint Archive, Report 2017/517, 2017. http://eprint.iacr.org/2017/517.

[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, *Advances in Cryptology – CRYPTO 2014, Part II*, volume 8617 of *Lecture Notes in Computer Science*, pages 235–255, Santa Barbara, CA, USA, August 17–21, 2014. Springer, Heidelberg, Germany.

[BFWW11] Christina Brzuska, Marc Fischlin, Bogdan Warinschi, and Stephen C. Williams. Composability of Bellare-Rogaway key exchange protocols. In Yan Chen, George Danezis, and Vitaly Shmatikov, editors, *ACM CCS 2011: 18th Conference on Computer and Communications Security*, pages 51–62, Chicago, Illinois, USA, October 17–21, 2011. ACM Press.

[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, Utrecht, The Netherlands Proceedings*, pages 384–405, Cham, 2017. Springer International Publishing.

[Bin18] Nina Bindel. *On the Security of Lattice-Based Signature Schemes in a Post-Quantum World*. PhD thesis, Technische Universität, Darmstadt, 2018.

[BK03] Mihir Bellare and Tadayoshi Kohno. A theoretical treatment of related-key attacks: RKA-PRPs, RKA-PRFs, and applications. In Eli Biham, editor, *Advances in Cryptology – EUROCRYPT 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 491–506, Warsaw, Poland, May 4–8, 2003. Springer, Heidelberg, Germany.

[BL15] Mihir Bellare and Anna Lysyanskaya. Symmetric and dual PRFs from standard assumptions: A generic validation of an HMAC assumption. Cryptology ePrint Archive, Report 2015/1198, 2015. http://eprint.iacr.org/2015/1198.

[BL16] Karthikeyan Bhargavan and Gaëtan Leurent. Transcript collision attacks: Breaking authentication in TLS, IKE and SSH. In *ISOC Network and Distributed System Security Symposium – NDSS 2016*, San Diego, CA, USA, February 21–24, 2016. The Internet Society.

[BMP00] Victor Boyko, Philip D. MacKenzie, and Sarvar Patel. Provably secure password-authenticated key exchange using Diffie-Hellman. In Bart Preneel, editor, *Advances in Cryptology – EUROCRYPT 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 156–171, Bruges, Belgium, May 14–18, 2000. 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.

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In Dorothy E. Denning, Raymond Pyle, Ravi Ganesan, Ravi S. Sandhu, and Victoria Ashby, editors, *ACM CCS 93: 1st Conference on Computer and Communications Security*, pages 62–73, Fairfax, Virginia, USA, November 3–5, 1993. ACM Press.

[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.

[BR95] Mihir Bellare and Phillip Rogaway. Provably secure session key distribution: The three party case. In Frank Thomson Leighton and Allan Borodin, editors, *27th Annual ACM Symposium on Theory of Computing*, pages 57–66, Las Vegas, NV, USA, May 29 – June 1, 1995. ACM Press.

[Bra16] Matt Braithwaite. Google Security Blog: Experimenting with post-quantum cryptography. https://security.googleblog.com/2016/07/experimenting-with-post-quantum.html, July 2016. Accessed: 2019-09-13.

[Brz13] Christina Brzuska. *On the Foundations of Key Exchange*. PhD thesis, Technische Universität Darmstadt, Darmstadt, Germany, 2013. http://tuprints.ulb.tu-darmstadt.de/3414/.

[BV98] Dan Boneh and Ramarathnam Venkatesan. Breaking RSA may not be equivalent to factoring. In Kaisa Nyberg, editor, *Advances in Cryptology – EUROCRYPT'98*, volume 1403 of *Lecture Notes in Computer Science*, pages 59–71, Espoo, Finland, May 31 – June 4, 1998. Springer, Heidelberg, Germany.

[BWJM97] Simon Blake-Wilson, Don Johnson, and Alfred Menezes. Key agreement protocols and their security analysis. In Michael Darnell, editor, *6th IMA International Conference on Cryptography and Coding*, volume 1355 of *Lecture Notes in Computer Science*, pages 30–45, Cirencester, UK, December 17–19, 1997. Springer, Heidelberg, Germany.

[BWM98] Simon Blake-Wilson and Alfred Menezes. Entity authentication and authenticated key transport protocols employing asymmetric techniques. In Bruce Christianson, Bruno Crispo, Mark Lomas, and Michael Roe, editors, *Security Protocols*, pages 137–158, Berlin, Heidelberg, 1998. Springer Berlin Heidelberg.

[Can01] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *42nd Annual Symposium on Foundations of Computer Science*, pages 136–145, Las Vegas, NV, USA, October 14–17, 2001. IEEE Computer Society Press.

[CC19] Matt Campagna and Eric Crockett. BIKE and SIKE Hybrid Key Exchange Cipher Suites for Transport Layer Security (TLS) draft-campagna-tls-bike-sike-hybrid-01. https://tools.ietf.org/html/draft-campagna-tls-bike-sike-hybrid-01, May 2019. Accessed: 2019-09-13.

[CCD$^+$16] Katriel Cohn-Gordon, Cas Cremers, Benjamin Dowling, Luke Garratt, and Douglas Stebila. A formal security analysis of the signal messaging protocol. Cryptology ePrint Archive, Report 2016/1013, 2016. http://eprint.iacr.org/2016/1013.

[CCD$^+$17] Katriel Cohn-Gordon, Cas J. F. Cremers, Benjamin Dowling, Luke Garratt, and Douglas Stebila. A formal security analysis of the signal messaging protocol. In *2017 IEEE European Symposium on Security and Privacy, EuroS&P 2017, Paris, France, April 26-28, 2017*, pages 451–466, 2017.

[CCG16] Katriel Cohn-Gordon, Cas J. F. Cremers, and Luke Garratt. On Post-compromise Security. In *IEEE CSF 16*, pages 164–178, 2016.

[CCG$^+$18] Katriel Cohn-Gordon, Cas Cremers, Luke Garratt, Jon Millican, and Kevin Milner. On ends-to-ends encryption: Asynchronous group messaging with strong security guarantees. In David Lie, Mohammad Mannan, Michael Backes, and XiaoFeng Wang, editors, *ACM CCS 2018: 25th Conference on Computer and Communications Security*, pages 1802–1819, Toronto, ON, Canada, October 15–19, 2018. ACM Press.

[CDMP05] Jean-Sébastien Coron, Yevgeniy Dodis, Cécile Malinaud, and Prashant Puniya. Merkle-Damgård revisited: How to construct a hash function. In Victor Shoup, editor, *Advances in Cryptology – CRYPTO 2005*, volume 3621 of *Lecture Notes in Computer Science*, pages 430–448, Santa Barbara, CA, USA, August 14–18, 2005. Springer, Heidelberg, Germany.

[CEL$^+$16] Craig Costello, Karen Easterbrook, Brian LaMacchia, Patrick Longa, and Michael Naehrig. SIDH Library. https://www.microsoft.com/en-us/research/project/sidh-library/, April 2016. Accessed: 2019-09-13.

[CF01] Ran Canetti and Marc Fischlin. Universally composable commitments. In Joe Kilian, editor, *Advances in Cryptology – CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 19–40, Santa Barbara, CA, USA, August 19–23, 2001. Springer, Heidelberg, Germany.

[CF12] Cas J. F. Cremers and Michele Feltz. Beyond eCK: Perfect forward secrecy under actor compromise and ephemeral-key reveal. In Sara Foresti, Moti Yung, and Fabio Martinelli, editors, *ESORICS 2012: 17th European Symposium on Research in Computer Security*, volume 7459 of *Lecture Notes in Computer Science*, pages 734–751, Pisa, Italy, September 10–12, 2012. Springer, Heidelberg, Germany.

[CGH98] Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited (preliminary version). In Jeffrey Scott Vitter, editor, *30th Annual ACM Symposium on Theory of Computing*, pages 209–218, Dallas, TX, USA, May 23–26, 1998. ACM Press.

[CHK19] Cas Cremers, Britta Hale, and Konrad Kohbrok. Revisiting post-compromise security guarantees in group messaging. Cryptology ePrint Archive, Report 2019/477, 2019. https://eprint.iacr.org/2019/477.

[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.

[CK02] Ran Canetti and Hugo Krawczyk. Security analysis of IKE's signature-based key-exchange protocol. In Moti Yung, editor, *Advances in Cryptology – CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 143–161, Santa Barbara, CA, USA, August 18–22, 2002. Springer, Heidelberg, Germany. http://eprint.iacr.org/2002/120/.

[CPS19] Eric Crockett, Christian Paquin, and Douglas Stebila. Prototyping post-quantum and hybrid key exchange and authentication in tls and ssh. Cryptology ePrint Archive, Report 2019/858 and in *NIST 2nd Post-Quantum Cryptography Standardization Conference 2019*, August 2019, Santa Barbara, 2019. https://eprint.iacr.org/2019/858.

[Dam90] Ivan Damgård. A design principle for hash functions. In Gilles Brassard, editor, *Advances in Cryptology – CRYPTO'89*, volume 435 of *Lecture Notes in Computer Science*, pages 416–427, Santa Barbara, CA, USA, August 20–24, 1990. Springer, Heidelberg, Germany.

[dB94] Bert den Boer and Antoon Bosselaers. Collisions for the compressin function of MD5. In Tor Helleseth, editor, *Advances in Cryptology – EUROCRYPT'93*, volume 765 of *Lecture Notes in Computer Science*, pages 293–304, Lofthus, Norway, May 23–27, 1994. Springer, Heidelberg, Germany.

[Den06a] Alexander W Dent. Fundamental problems in provable security and cryptography. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 364(1849):3215–3230, 2006.

[Den06b] Alexander W. Dent. A note on game-hopping proofs. Cryptology ePrint Archive, Report 2006/260, 2006. http://eprint.iacr.org/2006/260.

[DF11] Özgür Dagdelen and Marc Fischlin. Security analysis of the extended access control protocol for machine readable travel documents. In Mike Burmester, Gene Tsudik, Spyros S. Magliveras, and Ivana Ilic, editors, *ISC 2010: 13th International Conference on Information Security*, volume 6531 of *Lecture Notes in Computer Science*, pages 54–68, Boca Raton, FL, USA, October 25–28, 2011. Springer, Heidelberg, Germany.

[DFGS15a] Benjamin Dowling, Marc Fischlin, Felix Günther, and Douglas Stebila. A cryptographic analysis of the TLS 1.3 handshake protocol candidates. In Indrajit Ray, Ninghui Li, and Christopher Kruegel, editors, *ACM CCS 2015: 22nd Conference on Computer and Communications Security*, pages 1197–1210, Denver, CO, USA, October 12–16, 2015. ACM Press.

[DFGS15b] Benjamin Dowling, Marc Fischlin, Felix Günther, and Douglas Stebila. A cryptographic analysis of the TLS 1.3 handshake protocol candidates. Cryptology ePrint Archive, Report 2015/914, 2015. http://eprint.iacr.org/2015/914.

[DFGS16] Benjamin Dowling, Marc Fischlin, Felix Günther, and Douglas Stebila. A cryptographic analysis of the TLS 1.3 draft-10 full and pre-shared key handshake protocol. Cryptology ePrint Archive, Report 2016/081, 2016. http://eprint.iacr.org/2016/081.

[DG19] Nir Drucker and Shay Gueron. Continuous key agreement with reduced bandwidth. In Shlomi Dolev, Danny Hendler, Sachin Lodha, and Moti Yung, editors, *Cyber Security Cryptography and Machine Learning*, pages 33–46, Cham, 2019. Springer International Publishing.

[DH76] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.

[DK05] Yevgeniy Dodis and Jonathan Katz. Chosen-ciphertext security of multiple encryption. In Joe Kilian, editor, *TCC 2005: 2nd Theory of Cryptography Conference*, volume 3378 of *Lecture Notes in Computer Science*, pages 188–209, Cambridge, MA, USA, February 10–12, 2005. Springer, Heidelberg, Germany.

[DP18] Benjamin Dowling and Kenneth G. Paterson. A cryptographic analysis of the WireGuard protocol. In Bart Preneel and Frederik Vercauteren, editors, *ACNS 18: 16th International Conference on Applied Cryptography and Network Security*, volume 10892 of *Lecture Notes in Computer Science*, pages 3–21, Leuven, Belgium, July 2–4, 2018. Springer, Heidelberg, Germany.

[DR08] Tim Dierks and Eric Rescorla. The Transport Layer Security (TLS) Protocol Version 1.2. RFC 5246 (Proposed Standard), August 2008. Updated by RFCs 5746, 5878, 6176, 7465, 7507, 7568, 7627, 7685, 7905, 7919.

[DRS04] Yevgeniy Dodis, Leonid Reyzin, and Adam Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology – EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 523–540, Interlaken, Switzerland, May 2–6, 2004. Springer, Heidelberg, Germany.

[DRS19] Benjamin Dowling, Paul Rösler, and Jörg Schwenk. Flexible authenticated and confidential channel establishment (facce): Analyzing the noise protocol framework. Cryptology ePrint Archive, Report 2019/436, 2019. https://eprint.iacr.org/2019/436.

[DS15] Benjamin Dowling and Douglas Stebila. Modelling ciphersuite and version negotiation in the TLS protocol. In Ernest Foo and Douglas Stebila, editors, *ACISP 15: 20th Australasian Conference on Information Security and Privacy*, volume 9144 of *Lecture Notes in Computer Science*, pages 270–288, Brisbane, QLD, Australia, June 29 – July 1, 2015. Springer, Heidelberg, Germany.

[dSGSW17] Cyprien de Saint Guilhem, Nigel P. Smart, and Bogdan Warinschi. Generic Forward-Secure Key Agreement Without Signatures. In *Information Security - 20th International Conference, ISC 2017, Ho Chi Minh City, Vietnam, November 22-24, 2017, Proceedings*, pages 114–133. Springer International Publishing, 2017.

[dV17] Henry de Valence. SIDH in Go for quantum-resistant TLS 1.3. The Cloudflare Blog, https://blog.cloudflare.com/sidh-go/, September 2017. Accessed: 2019-09-13.

[DVOW92] Whitfield Diffie, Paul C. Van Oorschot, and Michael J. Wiener. Authentication and authenticated key exchanges. *Designs, Codes and Cryptography*, 2(2):107–125, 1992.

[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, Report 2012/688, 2012. http://eprint.iacr.org/2012/688.

[EG85] S. Even and Oded Goldreich. On the power of cascade ciphers. *ACM Transactions on Computer Systems*, 3(2):108–116, 1985.

[ETS15] Quantum safe cryptography and security. Technical Report 8, https://www.etsi.org/images/files/ETSIWhitePapers/QuantumSafeWhitepaper.pdf, June 2015. Accessed: 2019-09-13.

[Fey82] Richard P. Feynman. Simulating physics with computers. *International Journal of Theoretical Physics*, 21(6):467–488, Jun 1982.

[FG14] Marc Fischlin and Felix Günther. Multi-stage key exchange and the case of Google's QUIC protocol. In Gail-Joon Ahn, Moti Yung, and Ninghui Li, editors, *ACM CCS 2014: 21st Conference on Computer and Communications Security*, pages 1193–1204, Scottsdale, AZ, USA, November 3–7, 2014. ACM Press.

[FG17] Marc Fischlin and Felix Günther. Replay Attacks on Zero Round-Trip Time: The Case of the TLS 1.3 Handshake Candidates. In *IEEE EuroS&P 17*, pages 60–75. IEEE Computer Society Press, April 2017.

[FO99] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In Michael J. Wiener, editor, *Advances in Cryptology – CRYPTO'99*, volume 1666 of *Lecture Notes in Computer Science*, pages 537–554, Santa Barbara, CA, USA, August 15–19, 1999. Springer, Heidelberg, Germany.

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

[Gal12] Steven D. Galbraith. *Mathematics of Public Key Cryptography.* Cambridge University Press, 2012.

[GCR16] Ilias Giechaskiel, Cas J. F. Cremers, and Kasper Bonne Rasmussen. On bitcoin security in the presence of broken cryptographic primitives. In Ioannis G. Askoxylakis, Sotiris Ioannidis, Sokratis K. Katsikas, and Catherine A. Meadows, editors, *ESORICS 2016: 21st European Symposium on Research in Computer Security, Part II*, volume 9879 of *Lecture Notes in Computer Science*, pages 201–222, Heraklion, Greece, September 26–30, 2016. Springer, Heidelberg, Germany.

[GCR18] I. Giechaskiel, C. Cremers, and K. B. Rasmussen. When the crypto in cryptocurrencies breaks: Bitcoin security under broken primitives. *IEEE Security Privacy*, 16(4):46–56, July 2018.

[GHP18] Federico Giacon, Felix Heuer, and Bertram Poettering. KEM combiners. In Michel Abdalla and Ricardo Dahab, editors, *PKC 2018: 21st International Conference on Theory and Practice of Public Key Cryptography, Part I*, volume 10769 of *Lecture Notes in Computer Science*, pages 190–218, Rio de Janeiro, Brazil, March 25–29, 2018. Springer, Heidelberg, Germany.

[GM82] Shafi Goldwasser and Silvio Micali. Probabilistic encryption and how to play mental poker keeping secret all partial information. In *14th Annual ACM Symposium on Theory of Computing*, pages 365–377, San Francisco, CA, USA, May 5–7, 1982. ACM Press.

[GM84] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984.

[GMPS14] Sourav Sen Gupta, Subhamoy Maitra, Goutam Paul, and Santanu Sarkar. (Non-)random sequences from (non-)random permutations - analysis of RC4 stream cipher. *Journal of Cryptology*, 27(1):67–108, January 2014.

[GMR88] Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal on Computing*, 17(2):281–308, 1988.

[Gol06] Oded Goldreich. On post-modern cryptography. Cryptology ePrint Archive, Report 2006/461, 2006. http://eprint.iacr.org/2006/461.

[Goo] Google's Quantum Artificial Intelligence Lab. https://ai.google/research/teams/applied-science/quantum-ai/. Accessed: 2019-09-13.

[Gro96] Lov K. Grover. A fast quantum mechanical algorithm for database search. In *28th Annual ACM Symposium on Theory of Computing*, pages 212–219, Philadephia, PA, USA, May 22–24, 1996. ACM Press.

[Gro01] Lov K. Grover. From Schrödinger's equation to the quantum search algorithm. *American Journal of Physics*, 69(7):769–777, 2001.

[Gün90] Christoph G. Günther. An identity-based key-exchange protocol. In Jean-Jacques Quisquater and Joos Vandewalle, editors, *Advances in Cryptology – EUROCRYPT'89*, volume 434 of *Lecture Notes in Computer Science*, pages 29–37, Houthalen, Belgium, April 10–13, 1990. Springer, Heidelberg, Germany.

[Her05] Amir Herzberg. On tolerant cryptographic constructions. In Alfred Menezes, editor, *Topics in Cryptology – CT-RSA 2005*, volume 3376 of *Lecture Notes in Computer Science*, pages 172–190, San Francisco, CA, USA, February 14–18, 2005. Springer, Heidelberg, Germany.

[HHK17] Dennis Hofheinz, Kathrin Hövelmanns, and Eike Kiltz. A modular analysis of the Fujisaki-Okamoto transformation. In Yael Kalai and Leonid Reyzin, editors, *TCC 2017: 15th Theory of Cryptography Conference, Part I*, volume 10677 of *Lecture Notes in Computer Science*, pages 341–371, Baltimore, MD, USA, November 12–15, 2017. Springer, Heidelberg, Germany.

[HKN$^+$05] Danny Harnik, Joe Kilian, Moni Naor, Omer Reingold, and Alon Rosen. On robust combiners for oblivious transfer and other primitives. In Ronald Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 96–113, Aarhus, Denmark, May 22–26, 2005. Springer, Heidelberg, Germany.

[HKSU18] Kathrin Hövelmanns, Eike Kiltz, Sven Schäge, and Dominique Unruh. Generic authenticated key exchange in the quantum random oracle model. Cryptology ePrint Archive, Report 2018/928, 2018. https://eprint.iacr.org/2018/928.

[Hof19] Paul Hoffman. The Transition from Classical to Post-Quantum Cryptography draft-hoffman-c2pq-05. https://tools.ietf.org/html/draft-hoffman-c2pq-05, May 2019. Accessed: 2019-09-13.

[IBM] IBM Q. Quantum starts here. https://www.research.ibm.com/ibm-q/. Accessed: 2019-09-13.

[IIMP19] Akiko Inoue, Tetsu Iwata, Kazuhiko Minematsu, and Bertram Poettering. Cryptanalysis of ocb2: Attacks on authenticity and confidentiality. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology – CRYPTO 2019*, pages 3–31, Cham, 2019. Springer International Publishing.

[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.

[JZC$^+$18] Haodong Jiang, Zhenfeng Zhang, Long Chen, Hong Wang, and Zhi Ma. IND-CCA-secure key encapsulation mechanism in the quantum random oracle model, revisited. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology – CRYPTO 2018, Part III*, volume 10993 of *Lecture Notes in Computer Science*, pages 96–125, Santa Barbara, CA, USA, August 19–23, 2018. Springer, Heidelberg, Germany.

[JZM19] Haodong Jiang, Zhenfeng Zhang, and Zhi Ma. Key encapsulation mechanism with explicit rejection in the quantum random oracle model. In Dongdai Lin and Kazue Sako, editors, *Public-Key Cryptography – PKC 2019*, pages 618–645, Cham, 2019. Springer International Publishing.

[KBC97] Hugo Krawczyk, Mihir Bellare, and Ran Canetti. HMAC: Keyed-Hashing for Message Authentication. RFC 2104 (Informational), February 1997. Updated by RFC 6151.

[KE10] Hugo Krawczyk and Pasi Eronen. HMAC-based Extract-and-Expand Key Derivation Function (HKDF). RFC 5869 (Informational), May 2010.

[KHN$^+$14] Charlie Kaufman, Paul Hoffmann, Yoav Nir, Pasi Eronen, and Tero Kivinen. Internet Key Exchange Protocol Version 2 (IKEv2). RFC 7296, October 2014.

[Kil01] Eike Kiltz. A tool box of cryptographic functions related to the Diffie-Hellman function. In C. Pandu Rangan and Cunsheng Ding, editors, *Progress in Cryptology - INDOCRYPT 2001: 2nd International Conference in Cryptology in India*, volume 2247 of *Lecture Notes in Computer Science*, pages 339–350, Chennai, India, December 16–20, 2001. Springer, Heidelberg, Germany.

[KK18] Frankziskus Kiefer and Krzysztof Kwiatkowski. Hybrid ECDHE-SIDH Key Exchange for TLS draft-kiefer-tls-ecdhe-sidh-00. `https://tools.ietf.org/html/draft-kiefer-tls-ecdhe-sidh-00`, Nov 2018. Accessed: 2019-09-13.

[KL08] Jonathan Katz and Andrew Y. Lindell. Aggregate message authentication codes. In Tal Malkin, editor, *Topics in Cryptology – CT-RSA 2008*, volume 4964 of *Lecture Notes in Computer Science*, pages 155–169, San Francisco, CA, USA, April 7–11, 2008. Springer, Heidelberg, Germany.

[KL14] Jonathan Katz and Yehuda Lindell. *Introduction to Modern Cryptography, Second Edition*. Chapman & Hall/CRC, 2nd edition, 2014.

[KM04] Neal Koblitz and Alfred Menezes. Another look at “provable security”. Cryptology ePrint Archive, Report 2004/152, 2004. `http://eprint.iacr.org/2004/152`.

[KM06] Neal Koblitz and Alfred Menezes. Another look at provable security. II. Cryptology ePrint Archive, Report 2006/229, 2006. http://eprint.iacr.org/2006/229.

[KM19] Neal Koblitz and Alfred Menezes. Critical perspectives on provable security: Fifteen years of "another look" papers. *Advances in Mathematics of Communications*, 13:517, 2019. http://aimsciences.org//article/id/5d2d9acd-8f7a-4e46-8e4a-dfab9701f215.

[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, *Advances in Cryptology – CRYPTO 2013, Part I*, volume 8042 of *Lecture Notes in Computer Science*, pages 429–448, Santa Barbara, CA, USA, August 18–22, 2013. Springer, Heidelberg, Germany.

[Kra03] Hugo Krawczyk. SIGMA: The "SIGn-and-MAc" approach to authenticated Diffie-Hellman and its use in the IKE protocols. In Dan Boneh, editor, *Advances in Cryptology – CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 400–425, Santa Barbara, CA, USA, August 17–21, 2003. Springer, Heidelberg, Germany.

[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.

[Kra10] Hugo Krawczyk. Cryptographic extraction and key derivation: The HKDF scheme. In Tal Rabin, editor, *Advances in Cryptology – CRYPTO 2010*, volume 6223 of *Lecture Notes in Computer Science*, pages 631–648, Santa Barbara, CA, USA, August 15–19, 2010. Springer, Heidelberg, Germany.

[KS05] Stephen Kent and Karen Seo. Security Architecture for the Internet Protocol. RFC 4301 (Proposed Standard), December 2005. Updated by RFCs 6040, 7619.

[KSL$^+$19] Krzysztof Kwiatkowski, Nick Sullivan, Adam Langley, Dave Levin, and Alan Mislove. Towards Post-Quantum Cryptography in TLS. in *NIST 2nd Post-Quantum Cryptography Standardization Conference 2019*, August 2019, Santa Barbara, available at https://csrc.nist.gov/CSRC/media/Events/Second-PQC-Standardization-Conference/documents/accepted-papers/kwiatkowski-measuring-tls.pdf, August 2019. Accessed: 2019-09-13.

[KW16] Hugo Krawczyk and Hoeteck Wee. The OPTLS protocol and TLS 1.3. In *IEEE EuroS&P 16*, pages 81–96. IEEE Computer Society Press, March 2016.

[Kwi19] Krzysztof Kwiatkowski. Towards Post-Quantum Cryptography in TLS. The Cloudflare Blog, https://blog.cloudflare.com/towards-post-quantum-cryptography-in-tls/, June 2019. Accessed: 2019-09-13.

[KY07] Jonathan Katz and Moti Yung. Scalable protocols for authenticated group key exchange. *Journal of Cryptology*, 20(1):85–113, January 2007.

[Lan16] Adam Langley. Intent to Implement and Ship: CECPQ1 for TLS. Imperial Violet, Blog, https://groups.google.com/a/chromium.org/forum/topic/security-dev/DS9pp2U0SAc, July 2016. Accessed: 2019-09-13.

[Lan18] Adam Langley. CECPQ2. Imperial Violet, Blog, https://www.imperialviolet.org/2018/12/12/cecpq2.html, December 2018. Accessed: 2019-09-13.

[LJBN15] Robert Lychev, Samuel Jero, Alexandra Boldyreva, and Cristina Nita-Rotaru. How secure and quick is QUIC? Provable security and performance analyses. In *2015 IEEE Symposium on Security and Privacy*, pages 214–231, San Jose, CA, USA, May 17–21, 2015. IEEE Computer Society Press.

[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: 1st International Conference on Provable Security*, volume 4784 of *Lecture Notes in Computer Science*, pages 1–16, Wollongong, Australia, November 1–2, 2007. Springer, Heidelberg, Germany.

[LSY$^+$14] Yong Li, Sven Schäge, Zheng Yang, Christoph Bader, and Jörg Schwenk. New modular compilers for authenticated key exchange. In Ioana Boureanu, Philippe Owesarski, and Serge Vaudenay, editors, *ACNS 14: 12th International Conference on Applied Cryptography and Network Security*, volume 8479 of *Lecture Notes in Computer Science*, pages 1–18, Lausanne, Switzerland, June 10–13, 2014. Springer, Heidelberg, Germany.

[LXZ$^+$16] Xinyu Li, Jing Xu, Zhenfeng Zhang, Dengguo Feng, and Honggang Hu. Multiple handshakes security of TLS 1.3 candidates. In *IEEE Symposium on Security and Privacy, SP 2016, San Jose, CA, USA, May 22-26, 2016*, pages 486–505. IEEE Computer Society, 2016.

[Mer90] Ralph C. Merkle. One way hash functions and DES. In Gilles Brassard, editor, *Advances in Cryptology – CRYPTO'89*, volume 435 of *Lecture Notes in Computer Science*, pages 428–446, Santa Barbara, CA, USA, August 20–24, 1990. Springer, Heidelberg, Germany.

[Moo19] Dustin Moody. Round 2 of the NIST PQC "Competition" - What was NIST Thinking? https://csrc.nist.gov/Presentations/2019/Round-2-of-the-NIST-PQC-Competition-What-was-NIST, May 2019. Accessed: 2019-09-13.

[MRH04] Ueli M. Maurer, Renato Renner, and Clemens Holenstein. Indifferentiability, impossibility results on reductions, and applications to the random oracle methodology. In Moni Naor, editor, *TCC 2004: 1st Theory of Cryptography Conference*, volume 2951 of *Lecture Notes in Computer Science*, pages 21–39, Cambridge, MA, USA, February 19–21, 2004. Springer, Heidelberg, Germany.

[MW96] Ueli M. Maurer and Stefan Wolf. Diffie-Hellman oracles. In Neal Koblitz, editor, *Advances in Cryptology – CRYPTO'96*, volume 1109 of *Lecture Notes in Computer Science*, pages 268–282, Santa Barbara, CA, USA, August 18–22, 1996. Springer, Heidelberg, Germany.

[Nat15] National Institute of Standards and Technology (NIST). Post-quantum cryptography. https://csrc.nist.gov/projects/post-quantum-cryptography, Aug 19, 2015. Accessed: 2019-09-13.

[NC00] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2000.

[NIS08] NIST. Federal Information Processing Standard 198, The Keyed-Hash Message Authentication Code (HMAC), July 2008.

[NIS15] NIST. Federal Information Processing Standard 202, SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions, Aug 2015.

[OP01] Tatsuaki Okamoto and David Pointcheval. The gap-problems: A new class of problems for the security of cryptographic schemes. In Kwangjo Kim, editor, *PKC 2001: 4th International Workshop on Theory and Practice in Public Key Cryptography*, volume 1992 of *Lecture Notes in Computer Science*, pages 104–118, Cheju Island, South Korea, February 13–15, 2001. Springer, Heidelberg, Germany.

[OQS18] OQS Open Quantum Safe Project. OQS-OpenSSL-1-1-1_stable. https://openquantumsafe.org/, November 2018. Accessed: 2019-09-13.

[Pei14] Chris Peikert. Lattice cryptography for the internet. In Michele Mosca, editor, *Post-Quantum Cryptography - 6th International Workshop, PQCrypto 2014*, pages 197–219, Waterloo, Ontario, Canada, October 1–3 2014. Springer, Heidelberg, Germany.

[PV05] Pascal Paillier and Damien Vergnaud. Discrete-log-based signatures may not be equivalent to discrete log. In Bimal K. Roy, editor, *Advances in Cryptology – ASIACRYPT 2005*, volume 3788 of *Lecture Notes in Computer Science*, pages 1–20, Chennai, India, December 4–8, 2005. Springer, Heidelberg, Germany.

[Rab79] Michael O. Rabin. Digital signatures and public key functions as intractable as factorization. Technical Report MIT/LCS/TR-212, Massachusetts Institute of Technology, January 1979.

[Res18] Eric Rescorla. The Transport Layer Security (TLS) Protocol Version 1.3. RFC 8446, August 2018.

[SBK$^+$17] Marc Stevens, Elie Bursztein, Pierre Karpman, Ange Albertini, and Yarik Markov. The first collision for full SHA-1. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology – CRYPTO 2017, Part I*, volume 10401 of *Lecture Notes in Computer Science*, pages 570–596, Santa Barbara, CA, USA, August 20–24, 2017. Springer, Heidelberg, Germany.

[SFG19] Douglas Stebila, Scott Fluhrer, and Shay Gueron. Design issues for hybrid key exchange in TLS 1.3 draft-stebila-tls-hybrid-design-01. https://tools.ietf.org/html/draft-stebila-tls-hybrid-design-01, Jul 2019. Accessed: 2019-09-13.

[Sha48] Claude E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27(3):379–423, 1948.

[Sha49] Claude E. Shannon. Communication theory of secrecy systems. *Bell Systems Technical Journal*, 28(4):656–715, 1949.

[Sho94] P. W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In *Proceedings of the 35th Annual Symposium on Foundations of Computer Science*, SFCS '94, pages 124–134, Washington, DC, USA, 1994. IEEE Computer Society.

[Sho97] Peter W. Shor. Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. *SIAM Journal on Computing*, pages 1484–1509, 1997.

[Sho99] Victor Shoup. On formal models for secure key exchange. Cryptology ePrint Archive, Report 1999/012, 1999. http://eprint.iacr.org/1999/012.

[Sho04] Victor Shoup. Sequences of games: a tool for taming complexity in security proofs. Cryptology ePrint Archive, Report 2004/332, 2004. http://eprint.iacr.org/2004/332.

[SKP16] Marc Stevens, Pierre Karpman, and Thomas Peyrin. Freestart collision for full SHA-1. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology – EUROCRYPT 2016, Part I*, volume 9665 of *Lecture Notes in Computer Science*, pages 459–483, Vienna, Austria, May 8–12, 2016. Springer, Heidelberg, Germany.

[SLdW07] Marc Stevens, Arjen K. Lenstra, and Benne de Weger. Chosen-prefix collisions for MD5 and colliding X.509 certificates for different identities. In Moni Naor, editor, *Advances in Cryptology – EUROCRYPT 2007*, volume 4515 of *Lecture Notes in Computer Science*, pages 1–22, Barcelona, Spain, May 20–24, 2007. Springer, Heidelberg, Germany.

[Soc04] American Mathematical Society. The Culture of Research and Scholarship in Mathematics: Joint Research and Its Publication. https://www.ams.org/profession/leaders/culture/CultureStatement04.pdf, 2004. Accessed: 2019-09-13.

[SS17] Jan Schank and Douglas Stebila. A Transport Layer Security (TLS) Extension For Establishing An Additional Shared Secret draft-schanck-tls-additional-keyshare-00. https://tools.ietf.org/html/draft-schanck-tls-additional-keyshare-00, Apr 2017. Accessed: 2019-09-13.

[Ste13] Marc Stevens. New collision attacks on SHA-1 based on optimal joint local-collision analysis. In Thomas Johansson and Phong Q. Nguyen, editors, *Advances in Cryptology – EUROCRYPT 2013*, volume 7881 of *Lecture Notes in Computer Science*, pages 245–261, Athens, Greece, May 26–30, 2013. Springer, Heidelberg, Germany.

[SXY18] Tsunekazu Saito, Keita Xagawa, and Takashi Yamakawa. Tightly-secure key-encapsulation mechanism in the quantum random oracle model. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2018, Part III*, volume 10822 of *Lecture Notes in Computer Science*, pages 520–551, Tel Aviv, Israel, April 29 – May 3, 2018. Springer, Heidelberg, Germany.

[TTB$^+$19] CJ Tjhai, Martin Tomlinson, Graham Bartlett, Scott Fluhrer, Daniel Van Geest, Oscar Garcia-Morchon, and Valery Smyslov. Framework to Integrate Post-quantum Key Exchanges into Internet Key Exchange Protocol Version 2 (IKEv2) draft-tjhai-ipsecme-hybrid-qske-ikev2-04. https://tools.ietf.org/html/draft-tjhai-ipsecme-hybrid-qske-ikev2-04, jul 2019. Accessed: 2019-09-13.

[Ust08] Berkant Ustaoglu. Obtaining a secure and efficient key agreement protocol from (H)MQV and NAXOS. *Designs, Codes and Cryptography*, 46(3):329–342, Mar 2008.

[WC81] Mark N. Wegman and J.Lawrence Carter. New hash functions and their use in authentication and set equality. *Journal of Computer and System Sciences*, 22(3):265 – 279, 1981.

[WFZGM17] William Whyte, Scott Fluhrer, Zhenfei Zhang, and Oscar Garcia-Morchon. Quantum-Safe Hybrid (QSH) Key Exchange for Transport Layer Security (TLS) version 1.3 draft-whyte-qsh-tls13-06. `https://tools.ietf.org/html/draft-whyte-qsh-tls13-06`, Oct 2017. Accessed: 2019-09-13.

[WY05] Xiaoyun Wang and Hongbo Yu. How to break MD5 and other hash functions. In Ronald Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 19–35, Aarhus, Denmark, May 22–26, 2005. Springer, Heidelberg, Germany.

[WYY05] Xiaoyun Wang, Yiqun Lisa Yin, and Hongbo Yu. Finding collisions in the full SHA-1. In Victor Shoup, editor, *Advances in Cryptology – CRYPTO 2005*, volume 3621 of *Lecture Notes in Computer Science*, pages 17–36, Santa Barbara, CA, USA, August 14–18, 2005. Springer, Heidelberg, Germany.

[ZHSI04] Rui Zhang, Goichiro Hanaoka, Junji Shikata, and Hideki Imai. On the security of multiple encryption or CCA-security+CCA-security=CCA-security? In Feng Bao, Robert Deng, and Jianying Zhou, editors, *PKC 2004: 7th International Workshop on Theory and Practice in Public Key Cryptography*, volume 2947 of *Lecture Notes in Computer Science*, pages 360–374, Singapore, March 1–4, 2004. Springer, Heidelberg, Germany.