Quantum Security of Cryptographic Primitives

Vom Fachbereich Informatik der
Technischen Universität Darmstadt genehmigte

Dissertation

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

Tommaso Gagliardoni, M.Sc.
geboren in Perugia

Referenten: Prof. Dr. Marc Fischlin
Prof. Dr. Christian Schaffner

Tag der Einreichung: 15. Dezember 2016
Tag der mündlichen Prüfung: 13. Februar 2017

Darmstadt, 2017
D 17

Erklärung

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

Wissenschaftlicher Werdegang

September 2002 – November 2005

Laurea Triennale in Matematica per le Applicazioni (B.Sc. Mathematics for Applications) an der Università degli Studi di Perugia, Italien

November 2005 – Mai 2011

Laurea Specialistica in Matematica (M.Sc. Mathematics) an der Università degli Studi di Perugia, Italien

seit Dezember 2011

Wissenschaftlicher Mitarbeiter in der Forschungsgruppe „Kryptoplexität“ an der Technischen Universität Darmstadt.

List of Publications

[ABF$^+$16] Gorjan Alagic, Anne Broadbent, Bill Fefferman, Tommaso Gagliardoni, Christian Schaffner, and Michael St. Jules. Computational security of quantum encryption. In *Information Theoretic Security - 9th International Conference, ICITS 2016, Tacoma, WA, USA, August 9-12, 2016, Revised Selected Papers*, pages 47–71, 2016. [**Part of this thesis**].

[AGKP14] Frederik Armknecht, Tommaso Gagliardoni, Stefan Katzenbeisser, and Andreas Peter. General impossibility of group homomorphic encryption in the quantum world. In *Public-Key Cryptography - PKC 2014 - 17th International Conference on Practice and Theory in Public-Key Cryptography, Buenos Aires, Argentina, March 26-28, 2014. Proceedings*, pages 556–573, 2014.

[DFG13a] Özgür Dagdelen, Marc Fischlin, and Tommaso Gagliardoni. The Fiat-Shamir transformation in a quantum world. In *Advances in Cryptology - ASIACRYPT 2013 - 19th International Conference on the Theory and Application of Cryptology and Information Security, Bengaluru, India, December 1-5, 2013, Proceedings, Part II*, pages 62–81, 2013. [**Part of this thesis**].

[DFG$^+$13b] Özgür Dagdelen, Marc Fischlin, Tommaso Gagliardoni, Giorgia Azzurra Marson, Arno Mittelbach, and Cristina Onete. A cryptographic analysis of OPACITY - (extended abstract). In *Computer Security - ESORICS 2013 - 18th European Symposium on Research in Computer Security, Egham, UK, September 9-13, 2013. Proceedings*, pages 345–362, 2013.

[DFF$^+$14] Jean Paul Degabriele, Victoria Fehr, Marc Fischlin, Tommaso Gagliardoni, Felix Günther, Giorgia Azzurra Marson, Arno Mittelbach, and Kenneth G. Paterson. Unpicking PLAID - a cryptographic analysis of an ISO-standards-track authentication protocol. In *Security Standardisation Research - First International Conference, SSR 2014, London, UK, December 16-17, 2014. Proceedings*, pages 1–25, 2014.

[DFF$^+$16] Jean Paul Degabriele, Victoria Fehr, Marc Fischlin, Tommaso Gagliardoni, Felix Günther, Giorgia Azzurra Marson, Arno Mittelbach, and Kenneth G. Paterson. Unpicking PLAID: a cryptographic analysis of an ISO-standards-track authentication protocol. *Int. J. Inf. Sec.*, 15(6):637–657, 2016.

[GHS16] Tommaso Gagliardoni, Andreas Hülsing, and Christian Schaffner. Semantic security and indistinguishability in the quantum world. In *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part III*, pages 60–89, 2016. [**Part of this thesis**].

[GKK17] Tommaso Gagliardoni, Nikolaos P. Karvelas, and Stefan Katzenbeisser. ORAMs in a quantum world. *IACR Cryptology ePrint Archive*, 2017. [**Preprint, Part of this thesis**].

Acknowledgments

Being a PhD student is a strange experience. I am sure that everyone who goes through this experience has their own personal stories, difficult moments to remember, and funny anecdotes to tell. I, for one, can truly say that these last five years have been exciting, funny, and productive. In short, they have been intense, and I can really say at the end that I have grown up a lot, both from an academic and from a personal perspective.

All of this I owe to my advisor, Marc Fischlin. If I could travel back in time and I were given the choice of applying as a PhD student again, at *any one* research group I possibly wished for, I would still spam ruthlessly my application to Marc. He taught me a lot of things which go well beyond academic matters, and I value his guidance immensely. When I was accepted in Marc's group in 2011, I was not aware at the time of how privileged I was. Now I am, and for this I will owe forever a debt of gratitude to Marc.

Being part of the group was a great experience, and I really would like to thank a lot my present and former colleagues for this. I am particularly grateful to Andrea for being always there to help me with the bureaucracy, to Giorgia for helping me to support the thesis that Hawaii Pizza is a mortal sin, to Özgür for taking care of me during my first months in Darmstadt, and to Paul for sharing with me a lot of good time, laugh, and hate for pigeons. I am also very grateful to Arno, Chris, Christian, Cristina, Felix, Jacqueline, Pooya, Sogol, and Victoria, for their friendship and support. Thank you all!

I would also like to thank all my coauthors for many successful collaborations and for having helped me a lot into expanding my scientific knowledge. Sometimes collaboration turned into sincere friendship as well, and therefore I would like to thank in particular Gorjan Alagic, Andreas Hülsing, Nikolaos Karvelas, and Christian Schaffner for the priceless time spent together.

Finally, I would like to thank my family for their endless love and support. I will always look at you as an example and a guidance, and I strive to make you proud of me every day of my life. Thanks.

Tommaso Gagliardoni
Darmstadt, December 2016

Abstract

We call *quantum security* the area of IT security dealing with scenarios where one or more parties have access to quantum hardware. This encompasses both the fields of *post-quantum cryptography* (that is, traditional cryptography engineered to be resistant against quantum adversaries), and *quantum cryptography* (that is, security protocols designed to be natively run on a quantum infrastructure, such as *quantum key distribution*). Moreover, there exist also *hybrid models*, where traditional cryptographic schemes are somehow 'mixed' with quantum operations in certain scenarios. Even if a fully-fledged, scalable quantum computer has yet to be built, recent results and the pace of research in its realization call for attention, lest we suddenly find ourselves one day with an obsolete security infrastructure. For this reason, in the last two decades research in quantum security has experienced an exponential growth in interest and investments.

In this work, we propose the first systematic *classification* of quantum security scenarios, and for each of them we recall the main tools and results, as well as presenting new ones. We achieve this goal by identifying four distinct *quantum security classes*, or *domains*, each of them encompassing the security notions and constructions related to a particular scenario. We start with the class **QS**0, which is 'classical cryptography' (meaning that no quantum scenario is considered), where we present some classical constructions and results as a preliminary step.

Regarding post-quantum cryptography, we introduce the class **QS**1, where we discuss in detail the problems arising when designing a classical cryptographic object meant to be resistant against adversaries with local quantum computing power, and we provide a classification of the possible quantum security reductions in this scenario when considering provable security. Moreover, we present results about the quantum security and insecurity of the *Fiat-Shamir transformation* (a useful tool used to turn interactive identification schemes into digital signatures), and ORAMs (protocols used to outsource a database in a private way).

In respect to hybrid classical-quantum models, in the security class **QS**2 we discuss in detail the possible scenarios where these scenarios arise, and what a correct formalization should be in terms of *quantum oracle access*. We also provide a novel framework for the quantum security (both in terms of indistinguishability and semantic security) of *secret-key encryption schemes*, and we give explicit secure constructions, as well as impossibility results.

Finally, in the class **QS**3 we consider all those cryptographic constructions designed to run natively on quantum hardware. We give constructions for *quantum encryption schemes* (both in the secret- and public-key scenario), and we introduce transformations for obtaining such schemes by conceptually simpler schemes from the class **QS**2. Moreover, we introduce a quantum version of ORAM, called *quantum ORAM (QORAM)*, aimed at outsourcing in a private way a database composed of quantum data. In proposing a suitable security model and an explicit construction for QORAMs, we also introduce a technique of independent interest which models a quantum adversary able to extract information from a quantum system without disturbing it 'too much'.

We believe that the framework we introduce in this work will be a valuable tool for the scientific community in addressing the challenges arising when formalizing sound constructions and notions of security in the quantum world.

Zusammenfassung

In dieser Arbeit bezeichnen wir mit der Terminologie *Quantensicherheit* den Bereich der IT-Sicherheit welcher sich mit dem Anwendungsfall beschäftigt, in dem einer oder mehrere Teilnehmer Zugriff auf Quanten-Hardware haben. Dies umfasst sowohl den Bereich der *Post-Quanten Kryptographie* (d.h. klassische Kryptographie, welche resistent gegen Quantenangreifern ist), als auch die *Quantenkryptographie* (dies sind Sicherheitsprotokolle, welche so gestaltet sind, dass sie nativ auf Quanteninfrastrukturen operieren, z.B. *Quantenschlüsselverteilung*). Weiterhin werden sogenannte *Hybridmodelle* erfasst, in welchen traditionelle kryptographische Verfahren zu einem gewissen Grade mit Quantenoperationen 'gemischt' sind. Obwohl ein voll ausgereifter, skalierbarer Quantencomputer noch gebaut werden muss, zeigen aktuelle Resultate und das rasante Bestreben nach dessen Realisierung, dass man auf den Ernstfall vorbereitet sein sollte, um zu vermeiden sich eines Tages in einer überholten Sicherheitsinfrastruktur wiederzufinden. Dies begründet auch das in den letzten zwei Jahrzehnten exponentiell gestiegene Interesse und Investment in Forschung zum Thema Quantensicherheit.

In dieser Arbeit präsentieren wir die erste systematische *Klassifikation* von Quantensicherheitsszenarien. Für jede Klasse werden die wichtigsten Resultate und Techniken studiert und durch neue Resultate und Techniken ergänzt. Die Klassifizierung erfolgt durch die Identifikation von vier unterschiedlichen *Quantensicherheitsklassen* oder *Domänen*, wobei sich jeder Klasse eigene Sicherheitsmodelle und Konstruktionen für ein bestimmtes Szenario zuordnen lassen. Die Klassifizierung beginnt mit der Klasse **QS**0, welcher klassische kryptographische Szenarien zugeordnet werden und insbesondere keine Quantenszenarien enthalten sind. Für diese Klasse werden einleitend einige klassische Konstruktionen und Resultate präsentiert.

Für Post-Quanten Kryptographie werden natürliche Probleme betrachtet die entstehen, wenn ein klassisches kryptographisches Objekt resistent gegen einen Angreifer sein soll, welcher mit der Fähigkeit eines Quantencomputers ausgestattet ist. Solche Probleme werden durch die Klasse **QS**1 klassifiziert. Weiter klassifizieren wir an dieser Stelle mögliche Quantensicherheitsreduktionen im Kontext der Sicherheitsanalyse bei Problemen aus dieser Klasse. Zudem werden auch die Quantensicherheit und -unsicherheit der Fiat-Shamir Transformation (diese ist ein hilfreiches Verfahren um interaktive Iden-

tifikationsverfahren in digitale Signaturen zu transformieren) betrachtet, sowie ORAMs (welches ein Protokoll ist um private Daten einer Datenbank auszulagern).

Die Klasse **QS**2 beinhaltet Szenarien, welche dem Hybridmodel zuzuordnen sind. Wir diskutieren im Detail sowohl welche konkreten Szenarien diesem Modell zugeordnet werden können, als auch deren Formalisierung bezüglich eines Quanten-Orakel Zugriffs. Weiter wird ein neues Rahmenmodell für Quantensicherheit von symmetrischen Verschlüsselungsverfahren (sowohl im Sinne von semantischer Sicherheit als auch Ununterscheidbarkeit) eingeführt. Es werden sowohl konkrete sichere Konstruktionen als auch Unmöglichkeitsresultate präsentiert.

Letztlich benennen wir die Klasse **QS**3, welche kryptographische Konstruktionen umfasst, die in natürlicher Weise auf Quantenhardware laufen. Einerseits werden Konstruktionen für Quantenverschlüsselungsverfahren (sowohl ein symmetrisches als auch asymmetrisches Verfahren) präsentiert, andererseits auch Transformationen um solche Verfahren aus konzeptionell einfacheren Verfahren der Hybridklasse **QS**2 zu konstruieren. Des Weiteren wird eine quantenbasierte Version von ORAM, genannt *Quanten-ORAM* (oder, *QORAM*), eingeführt, welches es erlaubt private Datensätze einer Datenbank bestehend aus Quantendaten auszulagern. Im Zuge der Formalisierung eines geeigneten Sicherheitsmodells und einer expliziten Konstruktion für QORAMs werden eigenständige Techniken entwickelt, die es erlauben einen Quantenangreifer zu formalisieren welcher Informationen aus dem Quantensystem extrahieren kann ohne es ‘zu viel’ zu stören.

Wir glauben, dass die in dieser Arbeit eingeführten Formalisierungen und Klassifikation wertvolle und brauchbare Werkzeuge für die Wissenschaftsgemeinschaft bieten, um zukünftige Konstruktionen und Quantensicherheitsmodelle zu formalisieren.

Contents

Chapter 1

Introduction

Cryptography is the subdiscipline of mathematics studying *information security*, that is, the processing of information in presence of an adversary. This includes goals such as *communication secrecy, message authentication, identity verification, multiparty computation,* and much more. In the modern era of electronic information processing, cryptography is an area of crucial importance, and its applications are ubiquitous.

Modern cryptography is based on *provable security.* This is a methodological approach to assessing the security of a cryptosystem, where rigorous mathematical models and proofs are required in order to show that the security of the cryptosystem can be formally validated. Arguably the most important branch of provable security, from a practical standpoint, is *computational security*, which aims at *reducing* the security of a cryptosystem to some basic *hardness assumptions* in a mathematically sound way. Hardness assumptions are inherent to the difficulty of solving certain mathematical problems (such as integer factorization) which, for theoretical or historical reasons, are widely considered to be very hard to solve even with the help of the most powerful supercomputers known today. If a given cryptosystem is computationally secure, this means that on one hand it is always *theoretically* possible for an adversary with enough computational resources to break the security of that cryptosystem. But on the other hand, doing so would reguire either an unreasonable amount of time (modern standards of security often refers to many times the age of the universe), or an unreasonable amount of computational resources (storage, memory, power, etc.), or both.

The advantage of having a provably secure cryptosystem is that, as long as the security model used is sound and the underlying hardness assumptions hold, one can stay assured that the cryptosystem cannot be ‘broken’. This is in stark contrast with the ‘heuristic’ approach to cryptography employed until the ’70s, where cryptosystems were designed to be secure according to the intuition of the authors, and the only guarantee of that security was given by the ‘test of time’, in the sense that nobody would find a way to attack the cryptosystem for a long enough time. This approach has turned cryptography from a mere engineering exercise to a logical-deductive discipline.

However, the effectiveness of provable security strongly relies on the hardness assumptions used, which are *not guaranteed*. Good hardness assumptions are based on the observation that algorithmical advances on solving the underlying mathematical problem would imply (unlikely) breakthrough results of scientific importance. However, all of these assumptions are also based on the *belief* that the future computing technology will never be *inherently different* from today's, save for a somewhat expected increase in performance, due to engineering improvements.

1.1 Security in a Quantum World

This is where *quantum computers* come into play. Quantum computers [Fey82] are machines, first theorized by Richard Feynmann in the early '80s, which are not based on the laws of classical physics like traditional computers are, but on the laws of *quantum mechanics* instead. Quantum mechanics is a very fundamental scientific theory, which has revolutionized physics since the early 20th century. Despite requiring a quite involved mathematical formalism and leading often to very counterintuitive consequences, it has routinely succeeded in predicting experimental results which classical physics could not explain.

From a formal point of view, a quantum computer is a mathematical model where the laws of quantum mechanics are exploited to perform some kind of computation, in a much more efficient way than traditional computers. Quantum computers promise to revolutionize the Age of Information as we know it. The ability to store, transmit, and process quantum data opens a world of new possibilities in the area of information processing. Simplified [ARTL15] or limited models [TCM$^+$16] of quantum computers have already been built, and everything from the experiments performed so far seems to confirm the validity of the underlying theory and the viability of the technology. Although a fully-fledged, scalable quantum computer has yet to be built, recent results [OBK$^+$16] and the pace of research in its realization seem to hint at the fact that quantum computing might soon become a reality.

Post-Quantum Cryptography

It turns out that, due to the effects predicted by quantum mechanics, quantum computers can perform tasks which are not possible with any classical computing device, present or future. The breakthrough result in this direction (which sparked a lot of interest for quantum computing in the area of cryptography) is the 1994 work by Peter Shor [Sho94], who showed how for a quantum computer it is possible to factor large integers efficiently, a mathematical task considered to be unreasonably difficult until then, and at the base of many modern cryptosystems such as RSA [RSA78]. Subsequent works have shown how to harness the power of quantum computing in order to speed up the search of solutions to problems like the *discrete logarithm* [Wat01] on finite fields and elliptic curves, search on unstructured database [Gro96], collision finding [BHT98], and many others. Given that these are all hardness assumptions at the base of the security of cryptosystems [DH76, Gam84, JMV01] widely adopted in the industrial, banking, and military sectors amongst others, it is clear how the realization of a scalable quantum computer would pose a threat to modern IT infrastructures.

A sound notion of security should be *proactive*, i.e., trying to take countermeasures against a reasonable future threat before the threat manifests itself. For this reason, cryptography has tried to address the looming danger of quantum computing since the early '90s. The idea is to find new mathematical problems which are supposed to be 'hard' *even* for quantum computers, so that new, 'quantum-immune' cryptosystems can be constructed by relying on such new quantum computational hardness assumptions. These are problems such as finding short vectors on lattices (which are geometric structures of a certain form), inverting hash functions, decoding certain types of linear codes, and a few others. The branch of cryptography dealing with the mathematical analysis of these assumptions and the construction of new cryptosystems based on such assumptions is called *post-quantum cryptography* [BBD09]. Post-quantum cryptography is today a thriving branch of information security, and so far it has been quite successful at designing cryptosystems which are at the same time reasonably efficient on today's hardware, and based on problems which are believed to be quantum-hard.

However, post-quantum cryptography has two fundamental issues.

The first problem is that security proof techniques that have been developed for traditional cryptosystems might *fail* when 'translated' to the quantum scenario. A typical example is *rewinding*, a technique used in the security proofs of many cryptosystems, which roughly consists in modeling a scenario where the adversary is first run once, then rewound, partially reset, and then re-run again, in order to extract two different but related 'adversarial transcripts' that are then used somehow in the security proof. The problem is that rewinding often *does not work* with quantum adversaries, because the nature of quantum mechanics does not guarantee that a 'partial reset' of a quantum computer is always possible.

Proof failures of this kind have often been ignored in the post-quantum community in the past, and there are examples of attempts to 'patching' non–post-quantum cryptosystems into post-quantum ones, by merely replacing the underlying hardness assumption with a quantum-hard one, and ignoring the fact that in so doing the security proof might become invalid.

The second problem of post-quantum cryptography is the often incomplete understanding of sound security models in the quantum world. One thing is to say that *"the cryptosystem should be secure against a quantum adversary"*, another thing is to formalize mathematically what this exactly means. Models that are used for classically secure schemes are sometimes not adequate to model quantum security, and this can lead to confusion.

A typical example is the case of the *random oracle model (ROM)*, which is a formal paradigm widely used in security proofs. A random oracle is a purely mathematical construct which is completely independent from the type of adversary considered, and there are hence no exotic technical difficulties in adopting such paradigm in security proofs for post-quantum cryptosystems. In fact, such approach has been taken before, and there exist in literature cryptosystems advertised as 'post-quantum' just because they are based on quantum-hard problems and provably secure in the ROM.

A random oracle, however, is just an abstraction describing an idealized model of hash function, which is an algorithmic object eventually run on a computing device. As the code for such a hash function is usually public, it is reasonable to assume that an adversary equipped with a quantum computer could run the code on his quantum machine, and therefore would be able to access the hash function in a way which is not modeled anymore by the ROM. For this reason, in a sound post-quantum security analysis, the random oracle model should always be avoided, and replaced by a different, more involved model called *quantum random oracle model (QROM)*. It can happen that schemes proven secure in the ROM become insecure in the QROM [BDF+11].

All the above considerations are not intended to mean that the whole idea of post-quantum cryptography relies on a flawed model. In fact, there are plenty of cryptographically sound security analyses, where such problems are carefully taken care of. However, it is often the case that 'secure against quantum adversaries' is confused with 'relying on quantum-hard assumptions'.

Quantum Cryptography

On one hand, quantum computing poses new challenges for modern cryptography, as many of the currently used cryptographic schemes and protocols base their security on the hardness of certain mathematical problems which are known to be easily solvable by a quantum machine. On the other hand, quantum computers open up new possibilities in secure information processing, as they can also be used 'defensively' in order to reach unprecedented levels of privacy, integrity, and trusted authentication. Importantly, it is often the case that such applications do not even require a fully-fledged scalable quantum computer, but only quantum hardware of modest technological engineering difficulty, which is already commercially available and deployed in many applications worldwide.

A typical example is *quantum key distribution (QKD)* [BB14], where two remote parties aim at establishing a secure communication channel by exchanging a secret key, employing the exchange of elementary quantum information packets (*qubits*) through a quantum channel. This can be technologically done, for example, by transmitting polarized photons through an optic fiber channel. QKD is already largely developed [SLB$^+$11], and it provides levels of security that classical cryptography cannot reach.

Looking into the future, with the advent of more and more advanced quantum hardware, it is easy to envision a world where a large part (if not most or all) of our global IT infrastructure will rely on quantum information processing. Under this scenario, it is important to think how to manage security related to quantum data. Not only it is required to re-model in a quantum way tasks usually performed by classical cryptography, for example *encryption of quantum data* [ABF$^+$16] or *quantum authentication* [BCG$^+$02a]. But it also means to consider tasks which are *inherently impossible* without quantum data, and which only make sense when considering a 'fully quantum infrastructure', such as *quantum money* [Aar09] or *delegated quantum computation* [DFPR14].

In general, quantum computers promise to revolutionize the Age of Information as we know it. The ability to store, transmit, and process quantum data opens a world of new possibilities in the area of information processing. *Quantum cryptography* is the branch of cryptography which deals with designing secure cryptographic solutions which are natively meant to be run on a quantum hardware - this includes QKD and all of the other examples above, and still others. Quantum cryptography is a relatively recent area of study of modern cryptography, and there is still much to be done in terms of inventing new cryptosystems, creating correct security models, and figuring out the relations between classical and quantum cryptographic constructions.

1.2 Contribution and Structure of this Work

We define *'quantum security'* to be the discipline dealing with *all* the scenarios where one or more parties have access to quantum hardware. This encompasses both the fields of post-quantum cryptography, quantum cryptography, and also *hybrid models*, where traditional cryptographic schemes are somehow 'mixed' with quantum operations in certain scenarios. The term 'quantum security', although having appeared in the scientific literature before, has often been used used inconsistently from one work to another (see, for example, [Zha12a, Unr13, KM12, BCD$^+$16]), at times denoting 'post-quantum' notions of security, and at times denoting something else.

In this work, we provide the first systematic classification of quantum security scenarios, and a new framework for modeling quantum security notions in a sound way. We achieve this by identifying four distinct *quantum security classes*, or *domains*, each of them encompassing the security notions and constructions related to a particular scenario. We denote these classes by **QS** (standing for 'quantum security'), followed by a number identifying the class. For each of these classes we recall known notions and results, as well as providing some results which are new or appearing in one or more of the author's publications. We start with a preliminary section in **Chapter 2** where we recall some basic concepts and notation, and then we proceed by presenting the four quantum security classes in the following chapters.

As it often happens in academic research, many of the results presented in the various chapters of this thesis stem from collaborative projects, where each individual achievements can be contributed by several, and most often all, researchers participating in that project. This makes it hard, if not impossible sometimes, to pinpoint who contributed to which specific part of the overall work. At the beginning of chapters 3, 4, 5, and 6, we will give an account of the results presented in that chapter which are novel or appearing in some of the author's publications, and we will give, when possible, an account of which part of these results are the author's specific contribution.

QS0

We start in **Chapter 3** with the class **QS**0, which is 'classical cryptography' (meaning that no quantum scenario is considered), where we present some results about traditional cryptography as a preliminary step. In this chapter we introduce security models for different classical cryptographic primitives, and we also introduce other building blocks and transformations from one primitive to another. More in detail, first we define and analyze in Section 3.1 some of the building blocks used in modern cryptography: *pseudorandom number generators, functions, and permutations.*

Then we look at the security models (and some example of constructions) for *secret-key and public-key encryption schemes*, in sections 3.2 and 3.3 respectively. We do it by looking at both the security models of *semantic security* and *indistinguishability of ciphertexts.*

In Section 3.4, we discuss *digital signature schemes*, both in the standard model and in the ROM, and we show how to obtain secure signature schemes through the *Fiat-Shamir transformation* in Section 3.5.

Finally, in Section 3.6, we introduce *oblivious random access machines (ORAMs)*, which are interactive protocols used to privately outsource a large database. We look at PathORAM, one of the most famous of such protocols, by using the formalism introduced in [GKK17].

QS1

In **Chapter 3**, we look at post-quantum security, and we call **QS**1 the related quantum security domain. We start in Section 4.1 with a detailed discussion of all the issues arising when modeling quantum provable security for classical cryptographic objects, including some examples of how classical proofs can fail when 'translated' to the quantum world, and the meaning of *quantum access to classical oracles.* We conclude this section with a *classification* of possible quantum security reductions which, to the best of the author's knowledge, does not explicitly appear in existing literature.

Then, in Section 4.2 we introduce the quantum random oracle model, and we give some technical tools to deal with quantum random oracles.

In Section 4.3, we see how the security models for the building blocks defined in Section 3.1 change when considering post-quantum scenarios. We also have a look at cryptographic objects which are minimal *post-quantum hardness assumptions*, such as *post-quantum one-way functions* and *post-quantum one-way trapdoor permutations*.

In Section 4.4 we discuss post-quantum security notions for encryption schemes, both in the secret-key and public-key scenario, and we show some basic constructions. Then we discuss post-quantum digital signatures in Section 4.5. We do this both for the standard post-quantum model and for the quantum random oracle model.

We proceed in Section 4.6 to the analysis of the Fiat-Shamir transformation in the quantum random oracle model. We provide here both a positive and a negative result: if the underlying identification scheme has certain properties, then the Fiat-Shamir transform of that scheme yields a secure signature scheme in the quantum random oracle model. However, if the underlying identification scheme has different properties, it is possible to find an argument (using the technique of *meta-reduction*) which shows that security proofs of a certain form cannot be found at all. The surprising result here is that identification schemes having the latter type of properties are usually *less desirable* (in terms of security) than the former ones. We exploit this fact by showing a counterintuitive but efficient technique to 'strengthen' the quantum security of a signature scheme obtained through the Fiat-Shamir transformation by 'weakening' the security of the underlying identification scheme.

Finally, in Section 4.7 we look at post-quantum ORAMs, and at sufficient and necessary conditions to obtain a post-quantum version of PathORAM.

QS2

In **Chapter 5**, we look at *superposition-based quantum security*, and we call **QS**2 the related quantum security domain. This security class deals with special scenarios, where the cryptosystems studied are still classical (and can hence be run on a classical computer), but *extra* security guarantees against quantum adversaries are required in respect to the 'post-quantum' definition of security. We model these new scenarios in terms of *quantum oracle access capabilities* of the adversaries, explaining when such access is already implied in **QS**1 and when instead it leads to new security scenarios covered by **QS**2. Such scenarios arise in certain contexts, such as *obfuscation* and *fault attacks*, as explained in Section 5.1. But very often they also stem from ambiguous interpretations of the 'post-quantum' setting (as defined in **QS**1) sometimes present in the literature. From this point of view, one of the most important contributions of this thesis is to formally clarify the distinction between these two security classes.

In Section 5.2 we look at what happens when considering cryptographic building blocks in the new scenarios. It turns out that, in respect to the post-quantum scenarios, nothing changes for most of them, with two notable exceptions: quantum secure pseudorandom functions and permutations.

Finally we discuss quantum-resistant encryption schemes in Section 5.3, with a special emphasis on the secret-key case. For such schemes, we provide new notions of indistinguishability and semantic security, as well as secure constructions and impossibility results.

QS3

Finally, in **Chapter 6**, we leave the realm of classical cryptosystems, and we look at *quantum cryptosystems*, that is, cryptosystems meant to be natively run on quantum hardware.

First we look at *quantum encryption* (that is, quantum algorithms for the encryption of quantum data) both in the secret-key (Section 6.1) and public-key (Section 6.2) scenarios. For both cases we provide security notions, as well as new constructions. We also show a novel technique for building encryption schemes secure in the **QS**3 sense starting from encryption schemes secure in the **QS**2 sense.

Finally, we introduce *quantum ORAMs (QORAMs)* in Section 6.3. This is a new primitive (basically a quantum version of ORAM) which is aimed at outsourcing in a private way a database composed of quantum data. In proposing a new security model and an explicit construction for QORAMs, we also introduce a novel technique of independent interest which models a quantum adversary able to extract information from a quantum system without disturbing its state 'too much'.

1.3 Related Work

The idea of *quantum security* as defined in this work is to encompass different types of scenarios which have in common the secure management of information in presence of quantum devices. Therefore, the existing related literature in this respect is vast, and we only cite a few key works here.

The term 'post-quantum cryptography', as meant in the **QS**1 sense, was popularized by Bernstein, Buchmann, and Dahmen in [BBD09]. The QROM was introduced in [BDF$^+$11]. Regarding the problems inherent to quantum rewinding, see Watrous [Wat06], Unruh [Unr12], and Ambainis et al. [ARU14]. Song [Son14] discussed relations between classical and quantum reductions, and Hallgren et al. [HSS11] discussed classical cryptographic protocols in the quantum world. Post-quantum building blocks and encryption schemes can be constructed from mathematical problems on lattices [GGH97, Mic11, LPR13], linear codes [McE78], multivariate equations [KPG99], and supersingular isogenies [FJP14]. In addition to the problems just mentioned, post-quantum signature schemes can be constructed from hash functions [BHH^{+}15].

Superposition-based attacks have been first proposed in [DFNS13] in respect to multiparty computation. Quantum-secure pseudorandom functions and pseudorandom permutations have been investigated by Zhandry [Zha12a, Zha16], Kuwakado and Morii [KM10, KM12], and Alagic and Russell [AR16], while secret- and public-key encryption schemes falling in the **QS**2 cathegory have been proposed by Boneh and Zhandry in [BZ13b], where superposition-resistant signature schemes also appear. Signature schemes secure against superposition attacks have also been studied in [ES15]. Anand et al. [ATTU16], Kaplan et al. [KLLN16], and Santoli and Schaffner [SS17] extended some attacks against pseudorandom permutations to other block ciphers, modes of operation, and compression functions.

Quantum key distribution was introduced in the seminal works by Wiesner [Wie83], and Bennet and Brassard [BB14]. Quantum money was introduced by Aaronson [Aar09]. Computationally secure quantum encryption was formalized by Broadbent and Jeffery [BJ15], while [AM16, BCG^{+}02b, GYZ16] deal with authentication of quantum information. See [BS16] for an overview of quantum cryptographic schemes belonging to the **QS**3 class, and Vidick and Watrous [VW16] for an overview of quantum complexity theory and reductions in the quantum world.

Chapter **2**

Preliminaries

In this chapter we discuss the notation and provide basic definitions used in the rest of this work.

2.1 Basic Notions

We start with a few basic concepts, mathematical notation and terminology. In the rest of this work, 'w.l.o.g.' stands for 'without loss of generality', 'iff' stands for 'if and only if', and 'classical' means 'non-quantum'.

Numbers, strings, and generic atomic objects are denoted by default as lowercase letters, e.g., a, b, x, y. In particular, indices for sequences or families will be often denoted by n, m, i, j, k. Sometimes inputs and outputs of an algorithm will be denoted by lowercase Sans Serif script, e.g., $\mathsf{com}, \mathsf{state}, \mathsf{sig}$. The security parameter is n, or 1^n when expressed in unary notation.

Special symbols are $\perp$ (usually denoting 'error', or 'lack of meaning') and the lowercase Roman i (denoting the imaginary unity, $\sqrt{-1}$). The symbol $\|$ denotes concatenation of *bit strings*, and the symbol 0^k (resp. 1^k) denotes a k-bit string of zeroes (resp., ones). For a bit string (or natural number) x we denote its bit size (or bit length) as $|x|$. If x is a non-integer number, $|x|$ denotes its absolute value. If x is a complex number, $|x|$ denotes its complex modulus, and $\overline{x}$ its complex conjugate.

Families or collections of objects (sets, functions, probability distributions) are of the form $(\mathcal{A}_n)_n, (\mathcal{X}_{j,k})_{j,k}$, where individual elements of the family are indexed, e.g., $\mathcal{A}_n, \mathcal{X}_{j,k}$. However, if there is no ambiguity in the choice of the index (usually this is the security parameter), such families are labeled in short just as $\mathcal{A}$,$\mathcal{X}$, etc.

Sets are usually denoted by uppercase letters, e.g., T, X, Y, except for special sets such as $\varnothing, \mathbb{N}, \mathbb{R}, \mathbb{C}$, and the set of all permutations on a set X, denoted by $S(X)$. The set of all finite bit strings or words is $\{0,1\}^*$. However, sets of bit strings will often be presented as families, where each member of the family contains bit strings of the same length. In this case, sets will be denoted by $\mathcal{T}$,$\mathcal{X}$,$\mathcal{Y}$ instead, being understood that, e.g., $\mathcal{X} = (\mathcal{X}_n)_n$, where $\mathcal{X}_n$ only contains bit strings of length $f(n)$ for some positive (usually polynomial) function f. The cardinality (number of elements) of a set X is denoted by $|X|$. Set operations are $\cup$ (union), $\cap$ (intersection), $\setminus$ (set difference), and $\times$ (Cartesian product). If a tuple $(x, y, z) \in \mathcal{X} \times \mathcal{Y} \times \mathcal{Z}$, then single entries of the tuple are isolated by writing, e.g., $(x, y, z)_{\mathcal{X}\mathcal{Y}\mathcal{Z}}\Big|_{\mathcal{Y}} = y$.

Functions (from sets to sets) are denoted by lowercase calligraphic letters, e.g., $f, g, \ell : X \to Y$. Borrowing a commonly used notation when defining 'small' quantities (relative to some parameter), exceptions to this notation are special functions ε and δ.

However, when a function is actually a family (indexed, for example, in terms of the bit size of the input) then it is denoted by uppercase calligraphic letters, e.g., $\mathcal{F}, \mathcal{G}, \mathcal{L}$. Commonly, in this case, domain and target space of these functions are also indexed as families, in relation to the bit size of the function's input. For example, $\mathcal{F} : \mathcal{X} \to \mathcal{Y}$ represents a function $\mathcal{F}$ from set $\mathcal{X}$ to set $\mathcal{Y}$, which can be seen as a family of functions $(\mathcal{F}_n)_n$, where $\mathcal{F}_n : \mathcal{X}_n \to \mathcal{Y}_n$.

A (real-valued) function f is *polynomially bounded* iff there exists a polynomial function p and an element $\bar{x}$ such that $|f(x)| \leq p(x)$, $\forall x$ with $|x| > |\bar{x}|$. In this case we write $f = \mathsf{poly}$. A (real-valued) function ε is *negligible* iff, for any polynomial function p, there exists an element $\bar{x}$ such that $|\varepsilon(x)| < \frac{1}{p(x)}$, $\forall\, x$ with $|x| > |\bar{x}|$. In this case we write $\varepsilon = \mathsf{negl}$.

Lowercase Greek letters denote quantum states, either pure ones when written in bra-ket notation (e.g., $|\varphi\rangle, |\psi\rangle$) or mixed ones when written without (e.g., σ, ρ). Exceptions are the symbols δ and ε, as already discussed, and λ (used for eigenvalues). Uppercase Greek letters (Σ, Γ, Θ) are reserved for special purposes, usually to denote quantum channels.

Data structures (trees, blocks) are labeled with Typewriter script, e.g., `tree, block, node`.

Probability

Distributions are denoted by uppercase calligraphic letters, e.g., $\mathcal{D}, \mathcal{P}, \mathcal{U}$. Distribution ensembles, or families, are denoted by $(\mathcal{D}_n)_n, (\mathcal{P}_n)_n$, etc. As usual, if there is no ambiguity in the choice of the index (usually this is the security parameter), such families are labeled in short just as $\mathcal{D}$,$\mathcal{P}$, etc., with individual member distributions being $\mathcal{D}_n, \mathcal{P}_n$, etc.

If $\mathcal{D}$ is a distribution over a set $\mathcal{X}$, then sampling an element x from the distribution is written as $x \xleftarrow{\mathcal{D}} \mathcal{X}$ (or, a shorthand notation when the domain is clear, just $x \leftarrow \mathcal{D}$). Sampling an element uniformly at random from a set $\mathcal{R}$ is written as $r \overset{\$}{\leftarrow} \mathcal{R}$.

The *support* of a distribution $\mathcal{D}$ over a set $\mathcal{X}$ is the subset of elements with non-zero probability, i.e., $\{x \in \mathcal{X} : \Pr[x \leftarrow \mathcal{D}] > 0\}$. The *cardinality of a distribution* is the cardinality of its support.

If $\mathcal{D}$ is a distribution over $\mathcal{X} \times \mathcal{Y}$, then we denote the distribution on $\mathcal{X}$ induced by $\mathcal{D}$ as $\mathcal{D}_\mathcal{X} := \mathcal{D}\Big|_\mathcal{X}$, and the sampling as $\mathcal{D}_\mathcal{X} \to x$ where $x := (x, y)\Big|_\mathcal{X}$.

The *total variation distance* (or, *statistical distance*) of two distributions $\mathcal{D}_0, \mathcal{D}_1$ is defined as:

$$|\mathcal{D}_0 - \mathcal{D}_1| := \sum_x \left|\Pr[x \leftarrow \mathcal{D}_0] - \Pr[x \leftarrow \mathcal{D}_1]\right|.$$

Linear Algebra

Vectors are denoted either as tuples (e.g., $(x_1, \ldots, x_n)$) or as boldface characters for the notation of the corresponding components (e.g., $\mathbf{x}$). The zero vector is denoted as $\mathbf{0}$. Matrices (linear operators between two vector spaces) are denoted by uppercase letters, e.g., A, B, M. (unless families, in that case $\mathcal{A}, \mathcal{B}, \mathcal{M}$ etc., as previously explained), except for the special symbols *zero matrix (or null operator) over n elements* (denoted by $\mathbb{O}_n$), and the *identity matrix (or identity operator) over n elements* (denoted by $\mathbb{I}_n$). If M is an $n \times m$ matrix (which includes the case of vectors or scalars if n or m equals 1), then M^T denotes its $m \times n$ transpose, $\overline{M}$ denotes its $n \times m$ complex conjugate, and $M^\dagger$ denotes its $m \times n$ Hermitian conjugate (or adjoint) $\overline{M^T} = \overline{M}^T$. If M is an $n \times n$ matrix with non-zero determinant, its unique inverse is denoted by M^{-1}. An $n \times n$ matrix (or linear operator) M is *Hermitian* if $M = M^\dagger$, and *unitary* if $M^\dagger = M^{-1}$. The *trace* of a square matrix M is denoted by $\operatorname{tr}(M)$, and it is the sum of the elements on the diagonal.

A *complex Hilbert space* is a complex vector space $\mathfrak{H}$, together with an inner product operation $\langle ., . \rangle : \mathfrak{H} \times \mathfrak{H} \to \mathbb{C}$ such that $\mathfrak{H}$ (seen as a metric space) is complete in respect to the metric $\|\mathbf{x}\| := \sqrt{|\langle x, x \rangle|}$ induced by the inner product. Unless otherwise specified, the inner product adopted here is always the scalar product:

$$\langle \mathbf{x}, \mathbf{y} \rangle := \mathbf{x}\mathbf{y}^\dagger = (x_1, \ldots, x_n) \begin{pmatrix} \overline{y_1} \\ \vdots \\ \overline{y_n} \end{pmatrix} = \sum_i x_i \overline{y_i}$$

The norm induced by the above product is the *Euclidean norm*, and it is denoted by $\|\mathbf{x}\|_2$. The Euclidean distance between two vectors $\mathbf{x}$ and $\mathbf{y}$ is hence $\|\mathbf{x} - \mathbf{y}\|_2$. The *dimension* of a Hilbert space is the cardinality of a minimal set of orthonormal elements spanning the whole space. Such a set is called a *basis* for the complex Hilbert space, and it is not unique. In this work we only consider finite-dimensional complex Hilbert spaces.

2.2 Classical Computation

In this section we recall the basic concepts and notation related to classical computation and complexity theory. The topic is of course vast and here we do not cover in depth every aspect of it. For a more complete treatment of the aspects of computation and complexity theory we refer to [AB09].

Circuits and Algorithms

The fundamental objects of study of computation theory are *algorithms*, which are sequences of elementary operations applied to some input data; the goal is to perform some procedure on those input data to produce some output. The *complexity* of an algorithm can refer to the number of elementary steps performed, the running time, the memory consumption, or any other resource used during its execution. Such complexity is expressed in relation to the *instance size* of the computation, which is a positive integer expressing the 'size' of the computational problem which the algorithm has to solve in order to perform the desired computation; this parameter is usually (related to) the bit size of the input. The complexity of an algorithm is then expressed as a function of the instance size: for example, if an algorithm $\mathcal{A}$ has complexity at most $O(n^2)$ for instance size n, we say that $\mathcal{A}$ has 'quadratic complexity'. An algorithm is *deterministic* if it produces always the same output for the same input, while it is *probabilistic* if it also takes an additional input (of size at most polynomial in the instance size) drawn from uniform random bits; its output is hence expressed as a distribution over these 'internal random coins'.

In this work we only deal with *time complexity*, i.e., we count as complexity the execution time of the algorithm. Time complexity is expressed in terms of the number of elementary operations performed by the algorithm, regardless of their nature, i.e., we assume for simplicity that any elementary operation (be it an addition, logical AND, division, etc.) takes one unit of time to execute. Moreover, as common in cryptography, we call the instance size the *security parameter*, denoted by n. DPT stands for '(Boolean) deterministic polynomial time', while PPT stands for '(Boolean) probabilistic polynomial time', where 'Boolean' refers to the fact that the algorithm operates on bit strings and performs elementary Boolean (bit) operations.

Traditionally, the two most commonly used models used to describe a classical algorithm are *Turing machines* and *Boolean circuits*.

- A Turing machine is a mathematical model describing an abstract machine with an internal state, acting on a data tape and performing operations according to a pre-specified set of rules.

- Boolean circuits are acyclic directed graphs where the nodes are either input bits, output bits, or elementary (Boolean) operations. Complexity in this case is given by the total number of gates in the circuit.

In this work, by 'algorithm' we mean 'a uniform family of circuits', i.e., there exists a Turing machine which, given the security parameter expressed in unary 1^n as input, runs in time at most polynomial in n, and outputs a description of the n-th member of the circuit family. So, for example, a PPT algorithm $\mathcal{A}$ is a family of Boolean circuits $\mathcal{A} := (\mathcal{A}_n)_n$ such that:

1. there exists a Turing machine $\mathcal{M}$ such that, on input 1^n, $\mathcal{M}$ runs in time $O(\mathsf{poly}(n))$ and outputs a description of $\mathcal{A}_n$; and

2. $\mathcal{A}_n$ is a Boolean circuit of size $O(\mathsf{poly}(n))$, taking as input a $\mathsf{poly}(n)$-bit value and a $\mathsf{poly}(n)$ many uniformly random bits, and producing a $O(\mathsf{poly}(n))$-bit output.

Algorithms, being families of circuits, are denoted by, e.g., $\mathcal{A} := (\mathcal{A}_n)_n$. When studying an algorithm which is a subroutine of another algorithm, or where we do not want to stress that it is a family, or anyway for clarity of notation, we use math Sans Serif script (e.g., Access, KGen, Enc). Every algorithm *always* gets as input at least the security parameter, so we will ignore it in the notation, being understood that such input is always present. In order to express that a deterministic algorithm $\mathcal{A}$, on input a value x, produces an output y, we write: $y := \mathcal{A}(x)$ or, equivalently, $\mathcal{A}(x) =: y$. For a probabilistic algorithm instead, the notation becomes $y \leftarrow \mathcal{A}(x)$ (or, equivalently, $\mathcal{A}(x) \rightarrow y$). However, if the output of a probabilistic algorithm $\mathcal{A}$ is written as $\mathcal{A}(x) = y$, that is a shorthand notation for: $\Pr[\mathcal{A}(x) \rightarrow y] = 1$, where the probability is taken over the internal randomness of $\mathcal{A}$. If an algorithm's only input is the security parameter (which we omit from the notation, as said), we only write, e.g., $\mathcal{A} =: y$, or $\mathcal{A} \rightarrow y$ if probabilistic.

The random coins of a probabilistic algorithm are almost always omitted from its input, so we write simply, e.g., $\mathcal{A}(x) \rightarrow y$; however, if for some reason it is necessary to 'de-randomize' the algorithm (that is, to consider the deterministic algorithm obtained by fixing a particular choice of randomness r), we write this as $\mathcal{A}(x; r) =: y$. If $\mathcal{A}$ is probabilistic, then the notation $\Pr[\mathcal{A}(x) \rightarrow y]$ is meant as 'probability over the randomness of $\mathcal{A}$, for that particular value x', while $\Pr_{x \in \mathcal{X}}[\mathcal{A}(x) \rightarrow y]$ (or $\Pr_{x \leftarrow \mathcal{X}}[\mathcal{A}(x) \rightarrow y]$) means 'over the randomness of $\mathcal{A}$, averaged over the uniform distribution on $\mathcal{X}$'. However, if $\mathcal{A}$ is deterministic, then $\Pr_{x \in \mathcal{X}}[\mathcal{A}(x) \rightarrow y]$ (or $\Pr_{x \leftarrow \mathcal{X}}[\mathcal{A}(x) \rightarrow y]$) is given by the fraction $\frac{|\{x \in \mathcal{X} : \mathcal{A}(x) =: y\}|}{|\mathcal{X}|}$.

Abusing notation, we express sometimes algorithms as *(families of) functions* from (families of) sets of inputs to (families of) sets of outputs. So, for example, $\mathcal{A} := (\mathcal{A}_n)_n : X \times \mathcal{Y} \rightarrow \mathcal{Z} \times \{0,1\}^*$ means that, for every $n \in \mathbb{N}, \mathcal{A}_n$ is a Boolean circuit taking as input one element of X and one element of $\mathcal{Y}_n$, and outputting one element of $\mathcal{Z}_n$ and one extra bit string of unspecified length. If an algorithm only takes as input the security parameter and outputs elements in $\mathcal{X}$ we write just: $\mathcal{A} :\rightarrow \mathcal{X}$.

Algorithms can be *interactive*, and communicate with each other. A special case is given by *stateful* algorithms, which have an internal state variable which can be updated and stored across different executions of the same algorithm (in this sense, the algorithm 'communicates with his future self'). In order to represent this communication, three different notations can be used.

1. Explicit state transport. For example, if $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ and one wants the first stage algorithm $\mathcal{A}_1$ to communicate some information to the second stage $\mathcal{A}_2$, we write something like:

   ```
   1: $\mathcal{A}_1(x) \rightarrow (y, \mathsf{state})$
   2: $\mathcal{A}_2(z, \mathsf{state}) \rightarrow w$
   ```

   where state, when left unspecified, is a bit string of size polynomial in the security parameter, carrying the information to be transmitted.

2. Circuit self-output, used in particular for stateful algorithms. For example, if $\mathcal{A}_0$ is the algorithm in the initial state, then $\mathcal{A}_0$ 'outputs y and a description of its own updated state' as: $\mathcal{A}_0(x) \rightarrow (y, \mathcal{A}_1)$. If using this notation, from now on $\mathcal{A}_0$ cannot be invoked again anymore. Instead, $\mathcal{A}_1$ is run on some other input a and updates itself as: $\mathcal{A}_1(a) \rightarrow (w, \mathcal{A}_2)$. From now on, $\mathcal{A}_1$ cannot be invoked anymore. Instead a fresh invocation can be written as: $\mathcal{A}_2(w, y, b) \rightarrow (u, r, \mathcal{A}_3)$, and so on.

3. Communication transcript. In this case, two or more algorithms communicate back and forth through a *communication channel* (which is a shared register between the two circuits). The 'history' of the content of such register during the execution of two algorithms $\mathcal{A}$ and $\mathcal{B}$ is called *communication transcript* com, and it is usually denoted as: $\mathsf{com} \leftarrow \langle \mathcal{A}(x), \mathcal{B}(y) \rangle$.

If an algorithm $\mathcal{A}$ has *oracle access* to another algorithm (or family of functions) $\mathcal{O}$, this is written as $\mathcal{A}^{\mathcal{O}}$. In this case, it is understood that $\mathcal{A}$ can communicate with $\mathcal{O}$ through $\mathcal{O}$'s input and output registers solely, while $\mathcal{A}$ does not know anything else about $\mathcal{O}$'s structure, code, or working details. Such communication is called *query*: $\mathcal{A}$ queries $\mathcal{O}$ on input value x, then $\mathcal{O}$ computes the answer $y \leftarrow \mathcal{O}(x)$ and finally y is sent back to $\mathcal{A}$. In this case, $\mathcal{O}$'s running time is ignored: it is always assumed that one oracle invocation takes one unit time to execute, regardless of $\mathcal{O}$'s running time. Giving $\mathcal{A}$ oracle access to another resource $\mathcal{O}$ models the case where $\mathcal{A}$ is given 'extra power' in performing a certain task, without having to deal with the exact way this task is performed.

Computational Complexity Theory

Complexity classes are families of problems with related asymptotic difficulty. Their definition is often given in terms of *language verifiers*: a *language* is a subset of $\{0,1\}^*$, and a *verifier* for a language is an algorithm which checks if a given input bit string belongs to that language (outputs 1) or not (outputs 0). In this work we only consider the following three classical complexity classes.

- P is the set of all languages $\mathcal{L}$ for which there exists a DPT algorithm $\mathcal{M}$ such that:

  1. $\forall\, x \in \mathcal{L} \implies \mathcal{M}(x) = 1$; and
  2. $\forall\, x \notin \mathcal{L} \implies \mathcal{M}(x) = 0$.

  Informally, P is the set of all problems which are 'easy to solve', in the sense that a solution for a given problem instance of size n can be found deterministically in time at most polynomial in n.

- BPP is the set of all languages $\mathcal{L}$ for which there exists a PPT algorithm $\mathcal{M}$ and a positive constant c such that:

  1. $\forall\, x \in \mathcal{L} \implies \Pr[\mathcal{M}(x) \to 1] \geq \frac{1}{2} + c$; and
  2. $\forall\, x \notin \mathcal{L} \implies \Pr[\mathcal{M}(x) \to 0] \geq \frac{1}{2} + c$.

  Informally, BPP is the set of all problems which are 'easy to solve with high probability', in the sense that a solution for a given problem instance of size n can be found with high probability in time at most polynomial in n. It is currently unknown whether $\mathsf{P} \neq \mathsf{BPP}$ or not [Gol11].

- NP is the set of all languages $\mathcal{L}$ for which there exists a DPT algorithm $\mathcal{M}$ and a polynomial p such that:

  1. $\forall\, x \in \mathcal{L}\ \exists\, y \in \{0,1\}^*$ with $|y| \leq p(n)$ such that $\mathcal{M}(x,y) = 1$; and
  2. $\forall\, x \notin \mathcal{L}, \forall\, y \in \{0,1\}^*$ with $|y| \leq p(n) \implies \mathcal{M}(x,y) = 0$.

  Informally, NP is the set of all problems which admit a 'solution easy to check'. in the sense that a candidate solution for a given problem instance of size n can be tested deterministically in time at most polynomial in n. It is currently unknown whether $\mathsf{P} \neq \mathsf{NP}$ or not [AB09].

Let $\mathcal{L} \in \mathsf{NP}$ be a language with a (polynomially computable) relation $\mathcal{R}$, i.e., there exists a DPT algorithm Rel and a polynomial p such that $x \in \mathcal{L}$ iff there exists some $w \in \mathcal{W} \subset \{0,1\}^*$ such that $(x,w) \in \mathcal{R}$ and $|w| \leq p(|x|) \forall x$, where $(x,w) \in \mathcal{R} \Leftrightarrow \mathsf{Rel}(x,w) = 1$. We say that w is a *witness* for $x \in \mathcal{L}$ (and x is called a *theorem* or *statement*). We sometimes use the notation $\mathcal{R}_n$ to denote the set of pairs (x,w) in $\mathcal{R}$ of complexity measured in relation to the security parameter, e.g., if $|x| = n$. In this case, with abuse of notation we identify the relation $\mathcal{R}$ with the algorithm testing its membership Rel.

2.3 Classical Cryptography

In this section we briefly recall the basic concepts and terminology used in modern cryptography.

Provable Security

Traditionally, cryptography has been seen for a long time as a 'cat-and-mouse' game, in the sense that the only way to validate the quality of a proposed cryptographic object was to perform some sort of cryptanalysis on it (i.e., 'trying to break it'), and then trying to fix the vulnerabilities potentially found, until new flaws were found, and so on. Under this perspective, the criterion to decide whether a cryptographic object should be trusted or not is just 'the test of time', in the sense that no new vulnerabilities are being found 'for a long time'.

However, this paradigm has shifted radically in the last ~30 years. The modern approach to defining good practice in cryptography is *provable security*, which is a paradigm involving a rigorous mathematical analysis of the cryptographic object, adversarial model, and security assumptions. In provable security, when analyzing a cryptographic scheme, one needs to provide rigorous definitions and models for the following aspects:

1. the *functionality* of the cryptographic object, i.e., what exactly is the goal that the object wants to achieve;

2. the *adversary model*, i.e., what does a 'reasonable' adversary against the object look like? What does the adversary want to achieve? When can we say that he is 'successful'?

3. The *security proof*, i.e., a mathematical proof showing that, under the specified model and some basic, commonly accepted assumptions, it is possible to rule out *any* successful adversary against the cryptographic object in exam.

It is important to distinguish between two different concepts of security.

- *Information-theoretical (or, statistical) security.* In this case, the proof of security aims at showing that the behavior of the cryptographic object is statistically equivalent (in the sense that it produces a distribution of outputs at most negligibly different) to the behavior of an *idealized object*, against which no successful attacker can exist by definition. For example, an information-theoretical secure encryption scheme produces a distribution of ciphertexts which is at most negligibly different from the uniform distribution over all ciphertexts, regardless of the input plaintext. Clearly, information-theoretical security is very strong, because it gives security guarantees *regardless of the adversarial model.* However, being so strong, it is also limited in use, as very few cryptographic objects can be shown to be statistically secure.

- *Computational security,* on the other hand, aims at showing that a cryptographic object is secure by relying on the intrinsic computational limitations of a 'reasonable' adversary. For example, in a computationally secure (but not statistically secure) encryption scheme, an adversary might be able to break security by testing ('brute-forcing') all the possible encryption keys one after one. However if such an adversary, in so doing, takes an amount of time which exceeds by many orders of magnitude the age of the universe, we would not consider him a threat for the security of the cryptographic scheme. A commonly accepted definition of 'computationally bounded adversary' is 'polynomial-time bounded' (in the security parameter).

In this work we only focus on computational security, but sometimes we refer to statistical security when needed for comparison. The adversary model we consider in classical security is thus some form of PPT algorithm, possibly with oracle access to additional resources.

The 'winning condition' for a given adversary $\mathcal{A}$ is expressed in terms of the outcome of an *experiment* (or *game*), which is a mathematical model describing the intuitive behavior of an adversary trying to compromise the security of a cryptographic scheme $\mathcal{S}$. Formally, an experiment is an algorithm (taking as input the security parameter n and, optionally, other parameters) with oracle access to $\mathcal{A}$ and (the components of) $\mathcal{S}$, and outputting some value (typically a bit) telling whether the experiment was successful (i.e., $\mathcal{A}$ won) or not. The notation used is of the form $\mathsf{Game}^{\mathsf{LABEL}}_{\mathcal{S},\mathcal{A}}$, where LABEL identifies the particular experiment. The *advantage* of an adversary $\mathcal{A}$ running such experiment (denoted by $\mathsf{Adv}^{\mathsf{LABEL}}_{\mathcal{S},\mathcal{A}}$) is the difference between $\mathcal{A}$'s success probability, and the success probability of a 'naif' adversary who just guesses at random a possible solution to the problem of breaking $\mathcal{S}$'s security. Then, in order to define $\mathcal{S}$ secure, two possible approaches are considered:

1. *game-based security.* In this case, it is required that the advantage of *any* (computationally bounded) adversary is 'small' (meaning, negligible in the security parameter); or

2. *simulation-based security.* In this case, the success probability of an arbitrary adversary $\mathcal{A}$ in the original experiment is compared to the success probability of the same $\mathcal{A}$ in a *different* experiment, describing an idealized, or 'simulated' situation where there is basically no possibility that $\mathcal{A}$ can break the security of the underlying scheme. In this case, security requires that for *any* (computationally bounded) adversary, the difference between the success probabilities in the 'real' and the 'ideal' world are roughly the same (meaning, at most negligibly distinct).

Both approaches are widely used in provable security. Usually, simulation-based security better captures the idea of transforming in a rigorous mathematical model what intuitively we want a cryptographic object to achieve; game-based security, however, is often of more immediate formulation and simpler use in security proofs. A common technique in provable security is in fact to show equivalence between an intuitive, rigorous simulation-based security definition, and a simpler, easier-to-use game-based one.

Regarding *security proofs*, it must be noticed that such proofs are intuitively very hard to come up with. In fact, it is in theory easy to show that a particular, formally well-described adversary is unable to successfully attack a certain cryptographical scheme. However, the security proofs we need require to rule out *every possible adversary*, even those which we do not know yet, or are unable to formalize. Therefore, directly showing security against one adversary does not work, and different techniques are used instead.

A very common technique to show the security of a cryptographic scheme $\mathcal{S}$ is the concept of *reduction* to another problem, or primitive $\mathcal{P}$. Let us assume that $\mathcal{P}$ is hard to solve, or anyway widely believed to be hard. Then one could 'show' the security of $\mathcal{S}$ by proving that the problem of breaking $\mathcal{S}$'s security is 'at least as hard' as solving $\mathcal{P}$. This is accomplished by proving that, given an hypothetical, successful adversary $\mathcal{A}$ against $\mathcal{S}$, such adversary can be turned, *constructively and in an efficient way*, into an efficient solver for $\mathcal{P}$. In this case we say that the security of $\mathcal{S}$*reduces* to the hardness of $\mathcal{P}$, and the formal proof itself is called *reduction.* A typical example of reduction is giving an explicit description of an efficient algorithm $\mathcal{B}$ which solves $\mathcal{P}$, and which has oracle access to $\mathcal{A}$ (in that case $\mathcal{B}$ is also said to be the reduction itself). We say that a reduction is *'black-box'* if such oracle access is the *only* interaction between $\mathcal{A}$ and $\mathcal{B}$, and $\mathcal{B}$ does not have any other clue about $\mathcal{A}$, such as insights about $\mathcal{A}$'s code or access to oracles which, according to the security model, should be only accessible by $\mathcal{A}$. However, as it is common practice in provable security, $\mathcal{B}$ is allowed to know a priori an upper bound on $\mathcal{A}$'s running time or number of queries to his oracles.

Finally, another common topic in provable security are *impossibility results*, that is, general theorems stating that a certain class of cryptographic object having certain properties *cannot* be secure. The most direct way to do it is by providing an explicit attack, i.e., an efficient adversary working against every member of that class. However, this can be hard sometimes, and there are countless examples of cryptographic schemes where a direct attack is *not known*, but at the same time *no reduction can be found.*

A possible technique to show impossibility results is that of *meta-reductions.* Intuitively, a meta-reduction is 'a reduction on reductions': the idea is to show that, if a scheme $\mathcal{S}$ admits an efficient reduction $\mathcal{B}$ to some problem $\mathcal{P}$, then another reduction $\mathcal{M}$ exists, which uses $\mathcal{B}$ to attack another, possibly different hard problem $\mathcal{P}'$. This rules out the existence of $\mathcal{B}$.

In the case of meta-reductions, since $\mathcal{B}$ needs an efficient adversary $\mathcal{A}$

against $\mathcal{P}$ in order to work, and reductions must always be *constructive and efficient*, it should be $\mathcal{M}$'s duty to provide such adversary $\mathcal{A}$ for $\mathcal{B}$ to work with. However, since $\mathcal{M}$ cannot break $\mathcal{P}$ directly (or else this would be a contradiction), the meta-reduction *simulates* a 'fake' adversary, in such a way that the simulation cannot be used directly to break $\mathcal{P}$, but at the same time such simulation is undetectable from $\mathcal{B}$'s perspective. So, a meta-reduction technique works like this:

1. assume the existence of a reduction $\mathcal{B}$ from scheme $\mathcal{S}$ to problem $\mathcal{P}$.
2. Give an explicit description of *any* adversary $\mathcal{A}$ against $\mathcal{S}$. This adversary does not necessarily need to exist, because $\mathcal{B}$ works regardless of $\mathcal{A}$'s nature. In practice though, it is usually required that $\mathcal{B}$ is a black-box reduction.
3. Give an explicit description of an efficient algorithm $\mathcal{M}$ which can simulate $\mathcal{A}$ (from $\mathcal{B}$'s point of view) and any other resource or oracle that $\mathcal{B}$ needs to access.
4. Execute the reduction $\mathcal{B}$, and use $\mathcal{B}$'s output to break $\mathcal{P}'$.

Hardness Assumptions

Hardness assumptions relate to mathematical problems which are at the same time easy to formalize (and it is clear what a solver for these problems should accomplice), and such that to date no known general method for solving these problems has been found (and there is evidence that finding such a method is arguably very hard). These assumptions are important, because they identify problems which are very attractive reduce to during security proofs.

Since we are dealing with computational security, a very minimal assumption is that $\mathsf{P} \neq \mathsf{NP}$. This is widely believed to be the case [AB09]; however, finding cryptographic reductions to such a minimal assumption is very hard. In this section, we recall some commonly used hardness assumptions used in cryptography. In what follows, we assume w.l.o.g. that the message space is $\mathcal{X} = (\mathcal{X}_n)_n := (\{0,1\}^n)_n$.

One very well studied assumption that we will explicitly use later in this work is the *computational hardness of the discrete logarithm problem (DLP).*

Definition 2.1 (Discrete Logarithm Problem)**.** *For a security parameter n, let $(\mathcal{G}, \star)$ be a cyclic group of order exponential in n, with generator g, and such that $\star$ is efficiently computable. The* discrete logarithm problem (DLP) on $\mathcal{G}$ *is, given $h \overset{\$}{\leftarrow} \mathcal{G}$, to find $x \in \mathbb{N}$ such that $h = g^x$.*

The *DLP hardness assumption* (for a given group $(\mathcal{G}, \star)$) states that no PPT algorithm exists, which is able to solve the DLP problem with probability better than $\frac{1}{2} + c$ for any positive constant c (i.e., the DLP problem is *not* in BPP for many known groups). There exist many different variants of the DLP problem, such as the *decisional Diffie-Hellman (DDH) problem* and many others, see [Bon98] for a survey. There exist also many other number-theoretic hardness assumptions, both quantum-insecure (RSA [RSA78] and factorization, elliptic-curve DLP [JMV01], etc.) and (presumably) quantum-resistant (lattice problems [GGH97], code-based [McE78], isogenies [FJP14], etc.) but we will not address them specifically in this work

Another very minimal hardness assumption that we make heavy use of is the existence of *one-way functions*. Intuitively, these are (families of) functions that are 'easy' to evaluate on any input, but 'hard' to invert on a random output, meaning that no efficient algorithm can find a pre-image for a randomly generated image.

Definition 2.2 (One-Way Functions (OWF) and Permutations (OWP))**.** *Let* $\mathcal{F} = (\mathcal{F}_n)_n$ *be a* DPT *algorithm, with* $\mathcal{F}_n : \mathcal{X}_n \to \{0,1\}^*$*.* $\mathcal{F}$ *is a (family of)* one-way functions (OWF) *iff for any* PPT *algorithm* $\mathcal{A}$ *it holds:*

$$\Pr_{x \xleftarrow{\$} \mathcal{X}} \left[\mathcal{A}(\mathcal{F}(x)) \to x' : \mathcal{F}(x) = \mathcal{F}(x') \right] \leq \mathsf{negl}.$$

Moreover, in the special case where $\mathcal{F}_n : \mathcal{X}_n \to \mathcal{X}_n$ *are permutations on* $\mathcal{X}_n$ *for every* n*,* $\mathcal{F}$ *is a (family of)* one-way permutations (OWP)*.*

The existence of one-way functions would imply $\mathsf{P} \neq \mathsf{NP}$, but the converse is not believed to hold [AB09]. However, one-way functions are considered to be a very minimal assumption for the existence of computationally secure cryptography. In general, reducing the security of a cryptographic object to the existence of one-way functions is a strong indicator of the scheme's security.

Notice the following: Definition 2.2 does not say anything about individual members of the family being pseudorandom. For example, there might be one-way functions which always fixes certain bits of their output, which can hence be trivially inverted. However, these 'easily predictable' bits cannot be 'too many', otherwise an adversary $\mathcal{A}$ could invert the whole function by guessing the other bits, against the assumption of one-wayness. Those (Boolean functions of) bits which are *not* easily predictable are called *hard-core bits* (or *hard-core predicates*).

Definition 2.3 (Hard-Core Predicate)**.** *Let* $\mathcal{F} : \mathcal{X} \to \mathcal{Y}$ *be a OWF. A polynomial-time computable function* $\mathsf{hc}_\mathcal{F} : \mathcal{X} \to \{0,1\}$ *is a* hard-core predicate (or bit) of $\mathcal{F}$ *iff, for any* PPT *algorithm* $\mathcal{A}$ *it holds:*

$$\Pr_{x \xleftarrow{\$} \mathcal{X}} \left[\mathcal{A}(\mathcal{F}(x)) \to \mathsf{hc}_\mathcal{F}(x) \right] \leq \frac{1}{2} + \mathsf{negl}.$$

Whether *every* OWF admits hard-core predicates or not is an open problem [KL07]. But it can be shown that, given any OWF $\mathcal{F}$, it is always possible to construct another OWF $\mathcal{H}$ such that $\mathsf{hc}_\mathcal{H}$ exists. Moreover, if $\mathcal{F}$ is a OWP, then also $\mathcal{H}$ is.

Proposition 2.4 ([HILL99])**.** *Let $\mathcal{F}$ be a OWF (resp., OWP). Then it is possible to efficiently transform $\mathcal{F}$ into a OWF (resp., OWP) $\mathcal{H}$ such that at least one hard-core predicate $\mathsf{hc}_{\mathcal{H}}$ exists.*

Given the above, from now on we assume for simplicity that every OWF admits hard-core predicates. In the case that $\mathcal{F} : \mathcal{X} \to \mathcal{X}$ (in particular, if $\mathcal{F}$ is a OWP), the construction of hard-core bits can be iterated to $\mathsf{hc}_{\mathcal{H}^2}, \mathsf{hc}_{\mathcal{H}^3}, \ldots$.

Proposition 2.5 ([HILL99])**.** *Let $\mathcal{F} : \mathcal{X} \to \mathcal{X}$ be a OWF (resp., OWP) with hard-core predicate $\mathsf{hc}_{\mathcal{F}}$. Then $\mathcal{F}^2$ is a OWF (resp., OWP) with hard-core predicate $\mathsf{hc}_{\mathcal{F}^2}$.*

Another very important cryptographic assumption is the existence of *one-way trapdoor permutations (OWTP).* A OWTP is a (family of) permutations which are easy to evaluate but hard to invert, *unless* an extra piece of secret information is known (the *trapdoor*) which is specific to a certain permutation.

For our scope, it is convenient to express a family of OWTPs as indexed through an *index family*, which is efficiently sampleable together with the related trapdoor. We will denote by $\mathcal{I} := (\mathcal{I}_n)_n$ and $\mathcal{T} := (\mathcal{T}_n)_n$ the index and trapdoor spaces, respectively. W.l.o.g., we assume $\mathcal{I}_n \subseteq \{0,1\}^{\mathscr{d}(n)}$, and $\mathcal{T}_n \subseteq \{0,1\}^{t(n)}$ for security parameter $n \in \mathbb{N}$, where $\mathscr{d}$ and t are polynomial functions determined by the OWTP family.

Definition 2.6 (One-Way Trapdoor Permutation Family (OWTP))**.** *A (family of)* one-way trapdoor permutations (OWTP) *is a tuple of* PPT *algorithms* $\mathcal{P} := (\mathsf{Gen}, \mathsf{Eval}, \mathsf{Invert})$*:*

1. $\mathsf{Gen} :\to \mathcal{I} \times \mathcal{T}$*;*

2. $\mathsf{Eval} : \mathcal{I} \times \mathcal{X} \to \mathcal{X}$*;*

3. $\mathsf{Invert} : \mathcal{I} \times \mathcal{T} \times \mathcal{X} \to \mathcal{X} \cup \{\perp\}$,

and such that:

1. *for any* PPT *algorithm $\mathcal{A}$ it holds:*

$$\Pr_{\substack{x \overset{\$}{\leftarrow} \mathcal{X} \\ (i,t) \leftarrow \mathsf{Gen}}} \left[\mathcal{A}(i, \mathsf{Eval}(i,x)) \to x\right] \leq \mathsf{negl}; \ \ \textit{and}$$

2. $\mathsf{Invert}(i,t,y) = \mathsf{Eval}(i,x)$, $\forall\, x \in \mathcal{X}$, $\forall\, (i,t) \leftarrow \mathsf{Gen}$, $\forall\, y \leftarrow \mathsf{Eval}(i,x)$.

The existence of OWTP is an assumption, like in the case of OWF. It is a stronger assumption, because the existence of OWTP in particular implies the existence of OWF, but the converse is not believed to hold.

Proposition 2.7 (OWTP $\Longrightarrow$ OWP $\Longrightarrow$ OWF)**.** *Let $\mathcal{P} := (\mathsf{Gen}, \mathsf{Eval}, \mathsf{Invert})$ be a OWTP on $\mathcal{X}$. Then, for all but a negligible fraction of possible sequences $(i_n, t_n)_n \leftarrow \mathsf{Gen}(n) \Rightarrow \mathsf{Eval}(i_n, .)$ is a OWP (and thus a OWF) on $\mathcal{X} = (\mathcal{X}_n)_n$.*

Candidates OWTP can be constructed from some hard problems such as factorization, DLP, and many others. As an example, it is well known that if factoring large integers is hard, then one can build OWTP using, e.g., the *RSA cryptosystem [RSA78]*.

Theorem 2.8 (RSA $\Longrightarrow$ OWTP)**.** *If factorization of large integers is computationally hard, then OWTPs exist.*

The Random Oracle Model

In this section we briefly recall the *random oracle model (ROM)* methodology. The subject is quite involved and here we do not discuss it in detail, see [Bel98] for an overview. A random oracle (RO) is an abstract mathematical model representing an idealized version of a publicly accessible source of randomness. In practice, a RO is used in security proofs to replace pseudorandom objects, such as hash functions, which would be otherwise too difficult to analyze. The idea is that such objects approximate very well the mathematical model described by the random oracle, so that a security proof given in the ROM is ‘almost as good’ as a security proof given for the real-world implementation. However, it is important to keep in mind that there are cases of *ROM uninstantiability* [CGH98, BFM15]. That is, there exist (artificial) examples of cryptographic schemes which are provably secure in the ROM, but which become insecure whenever the random oracle is replaced by *any* hash function.

Formally, a random oracle from a bit string set $\mathcal{X}$ to a bit string set $\mathcal{Y}$ is a function $\mathcal{O} : \mathcal{X} \to \mathcal{Y}$ drawn uniformly at random from the set $\mathcal{Y}^{\mathcal{X}}$. The description of $\mathcal{O}$ is not explicitly given; instead, $\mathcal{O}$ can only be queried in a black-box way. At the beginning of the security analysis, the oracle is *initialized* by drawing a function uniformly at random from the set $\mathcal{Y}^{\mathcal{X}}$. The function so chosen remains unknown to all the parties involved in the protocol, but all those parties gain oracle access to it.

It is important to notice that, with high probability, a randomly chosen function from $\mathcal{X}$ to $\mathcal{Y}$ does not have a compact representation, so that the mere act of selecting a random function in $\mathcal{Y}^{\mathcal{X}}$ is not algorithmically defined. Because the security proofs we are interested in must be constructive and efficient, different approaches should be taken when constructing a random oracle. One possibility is *lazy sampling*: because the value distribution of a completely random function on a certain point x is independent from the value the function takes on any other point, then the following procedure defines a random function, by adaptively filling a lookup table of values as soon as they are queried for the first time. In terms of pseudocode, a lazy sampling procedure would look as follows:

```
set $\mathsf{LookupTable} = \varnothing$
for all query received on element $x$ do
    if $(x, y') \in \mathsf{LookupTable}$ for some element $y'$ then
        Return: $y'$
    else
        sample $y \xleftarrow{\$} \mathcal{Y}$
        set $\mathsf{LookupTable} := \mathsf{LookupTable} \cup \{(x, y)\}$
        Return: $y$
```

Another possible method is to instantiate the RO with an efficiently computable *pseudorandom function family*, which will be described in Section 3.1.

Finally, it is important to mention that a RO can be *reprogrammed*, that is, the underlying function can be changed 'on the fly' during the security proof. The intuition for this is that, since the RO replaces a hash function, the proof should still hold if we use a certain hash function instead of another one, as long as it does not have exploitable 'structures' which are not supposed to be found (with high probability) on a completely random function.

2.4 Quantum Computation

In this section we recall the basic concepts of quantum information theory and quantum computation. We only give here a brief overview, and refer to [NC00] for a more detailed exposition.

Quantum Mechanics

In quantum mechanics, an isolated physical system (which we denote usually by an uppercase letter, e.g., A) is represented by a complex Hilbert space, denoted by $\mathfrak{H}_A$ (or just $\mathfrak{H}$ when the physical system is implied), of dimension suitable to represent all the independent possible physical states of A. Using the *bra-ket notation*, a completely defined state φ of the system (also called a *pure state*) is represented by (a class of) unitary vectors denoted by $|\varphi\rangle$. A set of orthonormal generators for $\mathfrak{H}$ is a *basis* for $\mathfrak{H}$; a *computational basis* for $\mathfrak{H}$ is a conventionally defined basis where elements are labeled as bit strings (or integers) $\{|x\rangle : x = 1, \dots, d\}$, where $d := \dim \mathfrak{H}$. Every pure state $|\varphi\rangle$ can thus be written as:

$$|\varphi\rangle = \sum_x a_x |x\rangle ,$$

with $\sum_x |a_x|^2 = 1$. The complex coefficients a_x are the *amplitudes of* $|x\rangle$, and we say that $|\varphi\rangle$ is a *quantum superposition* of states $|x\rangle$. Sometimes, if $\mathcal{X}$ is a set, we use the notation $\mathfrak{H}_{\mathcal{X}}$ to denote a complex Hilbert space for some physical system such that the computational basis for that space is labeled with elements of $\mathcal{X}$. That is, $\mathfrak{H}_{\mathcal{X}}$ is the space generated by $\{|x\rangle : x \in \mathcal{X}\}$. For two pure quantum states $|\varphi\rangle = \sum_x a_x |x\rangle$ and $|\psi\rangle = \sum_x b_x |x\rangle$ in superposition in the basis states $|x\rangle$, the Euclidean distance is given by $\left(\sum_x |a_x - b_x|^2\right)^{\frac{1}{2}}$.

We denote by $\langle\psi|$ the *dual* of a state $|\psi\rangle$, i.e., $\langle\psi| := |\psi\rangle^\dagger$. By Riesz's Representation Theorem, for every linear functional $a : \mathfrak{H} \to \mathbb{C}$ there exists a unique $|\alpha\rangle$ such that $a(|\varphi\rangle) = \langle\alpha|\varphi\rangle, \forall\, \varphi \in \mathfrak{H}$. Notice that, since pure states are represented by classes of unitary vectors, then $|\langle\psi|\varphi\rangle|^2 \in [0,1]$, with $|\langle\psi|\varphi\rangle|^2 = 1$ iff $|\varphi\rangle = |\psi\rangle$, and $|\langle\psi|\varphi\rangle|^2 = 0$ iff $|\psi\rangle$ and $|\varphi\rangle$ are orthogonal. In particular, $\langle x_i|x_j\rangle = 0\, \forall\, i \neq j$.

According to the laws of quantum mechanics, two different types of physically valid transformations can be applied to pure states:

- *reversible transformations,* or *evolutions*, which are modeled by unitary operators of the form $U : \mathfrak{H} \to \mathfrak{H}$; and
- *measurements*, which allow an observer to extract information from the physical system.

In this work, for pure states we only consider *measurement in the computational basis*, which works in the following way: let $|\varphi\rangle = \sum_x a_x |x\rangle$. Then, measuring such state yields a single real-valued outcome x with probability $|a_x|^2$, and after such measurement the state *collapses* to the basis state $|x\rangle$.

The *composition* (joint system) of two physical systems A and B is represented by the *tensor product* of the respective Hilbert spaces, $\mathfrak{H}_{AB} := \mathfrak{H}_A \otimes \mathfrak{H}_B$. So, for example, if $\{|x\rangle\}_{x\in\mathcal{X}}$ is a basis for $\mathfrak{H}_A$ and $\{|y\rangle\}_{y\in\mathcal{Y}}$ is a basis for $\mathfrak{H}_B$ (for two sets $\mathcal{X}$ and $\mathcal{Y}$), then $\{|x\rangle \otimes |y\rangle\}_{(x,y)\in\mathcal{X}\times\mathcal{Y}}$ is a basis for $\mathfrak{H}_{AB} = \mathfrak{H}_{\mathcal{X}\times\mathcal{Y}}$. We write equivalently $|x\rangle \otimes |y\rangle = |x\rangle |y\rangle = |x, y\rangle$.

Two fundamental theorems in quantum information theory, which we only mention here informally, are the following.

Theorem 2.9 (No-Cloning Theorem)**.** *There does not exist any valid physical process which, given as input an arbitrary state $|\varphi\rangle$, produces the state $|\varphi\rangle \otimes |\varphi\rangle$.*

Theorem 2.10 (No-Signaling Theorem)**.** *There does not exist any valid physical process which allows two parties to transmit information faster-than-light, even though these parties are allowed to perform instantaneous physical action on remote and possibly entangled quantum systems.*

Entanglement

Notice that not all states of $\mathfrak{H}_{AB}$ are of the form $|\varphi\rangle \otimes |\psi\rangle$ for some $|\varphi\rangle \in \mathfrak{H}_A$ and $|\psi\rangle \in \mathfrak{H}_B$ - actually, very few of them are. For example, for 2-dimensional Hilbert spaces $\mathfrak{H}_A$ and $\mathfrak{H}_B$, the following state:

$$|\rho\rangle_{AB} = \sqrt{\frac{1}{2}}\,|00\rangle + \sqrt{\frac{1}{2}}\,|11\rangle \tag{2.1}$$

cannot be expressed as a tensor product of two pure states, even if it is a pure states itself. In fact, if we consider the state associated to the system A alone (which we denote by $|\rho\rangle_A$) we find that it is impossible to write this state as a superposition of $|x\rangle$ elements, and the same applies to $|\rho\rangle_B$. When this happens we say that $|\rho\rangle_A$ and $|\rho\rangle_B$ are *entangled* states, otherwise we say that $|\rho\rangle_{AB}$ is *separable*. It turns out that, in a composite system, the vast majority of possible quantum states are entangled, and only a small subclass of them are separable. Entangled states cannot be pure states, so a different formalism is required to express them.

The *density matrix formalism* is used to represent all those states (including entangled states) which cannot be represented as pure states. We call such states *mixed states*, and we drop the bra-ket notation to represent them, in order to highlight the fact that they are not vectors, but matrices. Mixed states can be represented as probability distributions over sets of pure states. If a mixed state ρ is defined as a distribution over elements $|\varphi_i\rangle$, each of them occurring with probability p_i, then we define:

$$\rho := \sum_i p_i \, |\varphi_i\rangle\!\langle\varphi_i| \, .$$

We call the resulting matrix representation of ρ the *density matrix (or density operator) representation* of ρ. Formally, density matrices are operators $\rho : \mathfrak{H} \to \mathfrak{H}$ such that:

1. (trace condition) $\mathrm{tr}(\rho) = 1$
2. (positivity condition) $\langle\varphi|\rho|\varphi\rangle \geq 0 \, \forall \, \varphi \in \mathfrak{H}$.

As a consequence, every density operators has diagonal elements in $[0, 1]$. We denote the set of all admissible quantum states on a system A (that is, the set of all positive, unitary-trace linear operators on $\mathfrak{H}_A$) as $\mathfrak{D}\left(\mathfrak{H}_A\right)$.

All the formalism defined for pure states can be reformulated in terms of mixed states, because mixed states describe a statistics on pure states. If $|\varphi\rangle$ is a pure state, its density matrix is defined just as $|\varphi\rangle\!\langle\varphi|$. If $\rho \in \mathfrak{D}\left(\mathfrak{H}_A\right)$ and $\sigma \in \mathfrak{D}\left(\mathfrak{H}_B\right)$, then $\rho \otimes \sigma \in \mathfrak{D}\left(\mathfrak{H}_{AB}\right)$ is the state of the joint system. A unitary evolution U applied to a mixed state ρ produces another mixed state $U\rho U^\dagger$. Measuring a state ρ in the computational basis yields outcome x_i with probability p_i, where p_i is the i-th diagonal element of ρ; in this case, the system is left in the state $|x\rangle\!\langle x|$.

If we have two (or more) physical systems A, B, and they are jointly in the state ρ_{AB}, then the state describing the system A (resp., B) alone is denoted by ρ_A (resp, ρ_B), which has density operator:

$$\rho_A := \mathrm{tr}_B(\rho_{AB}),$$

where tr_B is the *partial trace over* B, defined by:

$$\mathrm{tr}_B(|x_1\rangle\!\langle x_2|_A \otimes |y_1\rangle\!\langle y_2|_B) := |x_1\rangle\!\langle x_2|_A \cdot \mathrm{tr}(|y_1\rangle\!\langle y_2|_B).$$

The act of taking a state in a joint system and considering only the state in one of its subsystems, 'forgetting' about the rest of the system is called *tracing out (or, reducing)* to a certain subsystem. W.l.o.g this can be seen as: first measuring the state in the computational basis *only* on the subsystem to be 'forgotten' (thereby collapsing part of the state and hence obtaining a separable state between the two systems), and then discarding the collapsed state and only consider the state of the subsystem left.

Any physically allowable process in nature, according to quantum mechanics, has to obey the constraints that density operators must be mapped to other density operators. That is, the mathematical transformation describing a physical process must preserve the unitarity of the trace, and the positivity of the operators. We call such 'admissible transformations' *CPTP maps* (completely positive, trace-preserving maps), or *quantum channels.*

Quantum Circuits

The most widely used model for quantum computation is that of *quantum circuits.* A quantum circuit is the analogue of a Boolean circuit, with a few differences. For the purpose of this work, we consider the following:

- instead of acting on register of bits, a quantum circuits operates on *quantum registers*, which are physical systems composed of subsystems (called *qubits*) described by 2-dimensional complex Hilbert spaces.

- Instead of being composed of Boolean gates, quantum circuits are composed of *elementary quantum gates*, which are either measurement operators, or transformations on (some subsets of) qubits, described by unitary operators.

A quantum circuit takes as input a quantum register in a certain state and produces a quantum output, but we can always consider additional classical inputs and outputs (which can be 'embedded' into quantum registers as basis states). The outcome of the quantum computation, however, is usually recovered through a measurement. It turns out that, w.l.o.g., measurements during a quantum computation can always be postponed to the very end of the quantum circuit, without changing the distribution of outcomes.

The number of input and output qubits of a quantum circuit can be different from each other. In fact, even if unitary operators act on the same subspace, a quantum circuit can have additional constant, 'hidden input registers' (called *ancilla qubits*, usually initialized to $|0\rangle$), and can 'delete' or 'forget' some register (by tracing them out). However, *any* CPTP map can be modeled as a quantum circuit.

For the purpose of this work, we only consider measurement operators in the computational basis. If we have a single qubit in a state $|\varphi\rangle$ and we apply

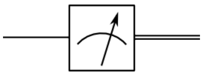


Figure 2.1: Quantum measurement gate.

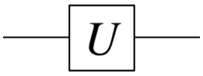


Figure 2.2: Single-qubit unitary gate.

a measurement on that qubit in order to obtain a single bit as outcome, we denote this as in Figure 2.1 (the double line denotes a classical output).

If U is a single-qubit unitary acting on the i-th qubit of an n-qubit system, it is denoted by U_i as shown in Figure 2.2.

The most basic single-qubit gate is the identity $\mathbb{I}$:

$$\mathbb{I} := \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}.$$

A very important single-qubit gate is the *Hadamard gate*, denoted by H, and defined by the unitary matrix:

$$H := \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}.$$

Other useful single-qubit operators are the *Pauli matrices* X, Y, Z defined by:

$$X := \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \quad Y := \begin{pmatrix} 0 & -\mathrm{i} \\ \mathrm{i} & 0 \end{pmatrix} \quad Z := \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

Notice how the Pauli matrices are also Hermitian. Moreover they satisfy: $XZ = \mathrm{i}Y$. We define the *Pauli group on* 1 *qubit* $\mathfrak{P}_1$ as the matrix multiplicative subgroup generated by $\{\mathrm{i}\mathbb{I}, X, Y, Z\}$. This extends to the *Pauli group on n qubits* $\mathfrak{P}_n$ as the subgroup generated by $\{\mathrm{i}\mathbb{I}_i, X_i, Y_i, Z_i : i = 1, \ldots, n\}$.

Finally, two very important 2-qubit gates are the *controlled-NOT (CNOT)* and the *SWAP* gates:

$$\mathsf{CNOT} := \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \quad \mathsf{SWAP} := \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

A *quantum algorithm* is a uniform family of quantum circuits, i.e., there exists a (classical) Turing machine which, given the security parameter expressed

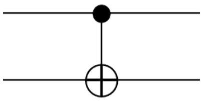


Figure 2.3: CNOT gate.

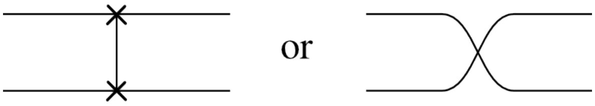


Figure 2.4: SWAP gate.

in unary 1^n as input, runs in time at most polynomial in n, and outputs a (classical) description of the n-th member of the quantum circuit family. QPT stands for 'quantum polynomial time', so a QPT algorithm is a uniform family of quantum circuits of size polynomial in the security parameter.

As quantum algorithms are probabilistic by nature, there is no quantum analogue of the classical complexity security class P. However, there is an analogue for BPP: the complexity BQP is the set of all languages $\mathcal{L}$ for which there exists a QPT algorithm $\mathcal{M}$ and a positive constant c such that:

1. $\forall\, x \in \mathcal{L} \implies \Pr[\mathcal{M}(x) \to 1] \geq \frac{1}{2} + c$; and
2. $\forall\, x \notin \mathcal{L} \implies \Pr[\mathcal{M}(x) \to 0] \geq \frac{1}{2} + c$.

Informally, BQP is the set of all problems which are 'easy to solve with high probability' on a quantum computer.

'Famous' quantum algorithms include Shor's algorithm [Sho94] for factoring integers and solving DLP in polynomial time, Simon's algorithm [Sim97] for recognizing in polynomial time black-box functions of a certain form, and Grover's algorithm [Gro96] for polynomially speeding up search on unsorted databases, inversion of functions, and general brute-force attacks.

Quantum Oracles

As in the classical case, the computational capabilities of a quantum algorithm $\mathcal{A}$ can be expanded by giving to the algorithm access to an oracle $\mathcal{O}$, which we denote by $\mathcal{A}^{\mathcal{O}}$. The oracle can be classical (with the same meaning as in the classical case), or it can be quantum. In the latter case, we have to distinguish between:

- *(standard) quantum oracle access.* In this case the oracle is a unitary operation U which $\mathcal{A}$ can query on a quantum state ρ at unit time cost in order to receive the response state $U\rho U^\dagger$. Whenever not specified, by 'oracle access' we always mean the standard one.

- *Quantum gate access.* In this case the oracle is also a unitary operator, like in the standard oracle access, the only difference is that $\mathcal{A}$ automatically gains access to the inverse operator $U^\dagger$ as well.

- *Quantum circuit access.* In this case the oracle is not necessarily unitary, but an arbitrary CPTP map. This means that the oracle could, e.g., perform measurements, or tracing out qubits, or act on additional quantum registers outside of $\mathcal{A}$'s control.

We use the following technical tool in the proof of Theorem 4.39. Let $\mathcal{A}$ be a quantum algorithm performing quantum queries to an oracle $\mathcal{O}$, and let $q_x(|\varphi_j\rangle)$ be the magnitude squared of basis element x in the j-th query, which we call the *query probability of x in query j*. If we sum over all queries, we get an upper bound on the total query probability of x.

Lemma 2.11 ([BBBV97, Theorem 3.3])**.** *Let $\mathcal{A}$ be a quantum algorithm running in time t with quantum oracle access to $\mathcal{O} : \mathcal{X} \to \mathcal{Y}$. Let $\varepsilon > 0$ and let $S \subseteq \{1, \ldots, t\} \times \mathcal{X}$ be a set of time-string pairs such that $\sum_{(j,x)\in S} q_x(|\varphi_j\rangle) \leq \varepsilon$. If we modify $\mathcal{O}$ into an oracle $\mathcal{O}'$ which answers each query x at time j by providing the same string $\bar{x}$ (which has been sampled independently from $\mathcal{O}$) whenever $(j, x) \in S$, then the Euclidean distance between the final states of $\mathcal{A}$ when invoking $\mathcal{O}$ and $\mathcal{O}'$ is at most $\sqrt{t\varepsilon}$.*

Distinguishing Quantum States

A crucial problem in quantum information theory is *distinguishing quantum states*. Because quantum states form a continuum, and because the only way we have to extract information from them is by performing measurements, distinguishing different quantum states with certainty is not always possible. In fact, for any practical purpose two quantum states are 'the same state' if *there is no physically admissible process extracting measurement outcomes with different distributions from those states*. In other words, it is only possible to distinguish different quantum states if we can perform operations on them leading to measurement outcome distributions which are themselves distinguishable. Because in this work we only deal with computationally bounded processes, it is clear that the minimal requirement for two states to be distinguishable is that they are (or can be efficiently transformed to) states which yield computationally distinguishable outcome distributions when measured.

The following lemma from [BV97] upperbounds the statistical distance between the distributions of measurements on two quantum states in terms of their Euclidean distance.

Lemma 2.12 ([BV97, Lemma 3.6])**.** *Let $|\varphi\rangle, |\psi\rangle$ be pure quantum states with Euclidean distance at most ε. Then, performing the same measurement on $|\varphi\rangle, |\psi\rangle$ yields distributions with statistical distance at most 4ε.*

For mixed states on isolated systems, the *trace distance* is a useful mathematical tool which gives directly an upper bound on the probability of distinguishing two states *for any physical process.*

Definition 2.13 (Trace Distance)**.** *Let $\rho, \sigma \in \mathfrak{D}(\mathfrak{H})$. The* trace distance *between ρ and σ is defined by:*

$$\|\rho - \sigma\|_{\mathrm{tr}} := \frac{1}{2}\sum_i |\lambda_i|,$$

where λ_i are the eigenvalues of $\rho - \sigma$.

We call the *totally, or maximally mixed (or entangled) state* over a physical system A the mixed state $\tau_A := \frac{\mathbb{I}}{\dim \mathfrak{H}_A}$; it has the property that a measurement over *any* possible orthonormal basis on this state always yields the uniform distribution of possible outcomes. This state represents somehow 'a state of maximal uncertainty', and a common technique to show that no information can be extracted from a quantum state is to show that such state has 'low' trace distance from the maximally mixed state.

However, when trying to distinguish between two CPTP maps, or two possible states on a *non-isolated system*, the trace distance is not enough. The reason is that, because of entanglement, two states which are *different* on a joint system AB might yield *the same* reduced state on a subsystem A. In that case, the trace distance on A would be 0, but a distinguisher with access to B might still be able to tell them apart. In these cases, the *diamond norm* is used, which induces a distance between CPTP maps.

Definition 2.14 (Diamond Norm)**.** *If Φ is a CPTP map (quantum channel) from operator spaces $\mathfrak{D}(\mathfrak{H}_A)$ to $\mathfrak{D}(\mathfrak{H}_B)$, then its* diamond norm *is defined by:*

$$\|\Phi\|_\diamond := \sup_{\rho \in \mathfrak{D}(\mathfrak{H}_{AK})} \|(\Phi \otimes \mathbb{I}_K)(\rho)\|_{\mathrm{tr}},$$

where $\mathfrak{H}_K$ is any *Hilbert space such that* $\dim \mathfrak{H}_K \geq \dim \mathfrak{H}_A$.

It can be shown that an upper bound to the probability of distinguishing two quantum channels Φ and Ψ is given by $\|\Phi - \Psi\|_\diamond$.

Chapter **3**

QS0: Classical Security

The first class of cryptographic security notions that we are going to analyze encompasses the weakest notions in the quantum world. Namely: no quantum at all. In our new labeling system, the security class **QS**0 refers to all the security notions and concepts which make *no mention* of quantum information theory. That is, **QS**0 is just classical cryptography, in a sense 'pre-(post-quantum)'. Studying this security class is essential in order to understand how the results change when we introduce quantum adversaries.

In this chapter we introduce security models for different classical cryptographic primitives, starting from the very basic ones (such as secret-key encryption schemes) to more elaborated ones. We also introduce other building blocks and transformations from one primitive to another.

A key feature of this part of our work is to perform all this analysis using a formalism which sometimes deviates from the one conventionally used in the existing literature, but which has the advantage of being easier to translate to the quantum world.

My Scientific Contribution in this Chapter

Most of the material in this chapter can be found in the existing literature (see for example [KL07, Gol01, Gol04], and is part of the preliminary technical results needed to understand the challenges arising when modeling security scenarios in a quantum world. However, to the best my knowledge, the proof of Theorem 3.32 has never been made explicit before. In fact, separation examples between CPA and CCA scenarios in the scientific literature usually refer either to the public-key scenario (where one can exploit group homomorphic properties) or to the separation between CPA and CCA2. Moreover, all the material from Section 3.6 first appeared in [GKK17], which is a joint work with Nikolaos P. Karvelas and Stefan Katzenbeisser. In that work, I focused on the quantum and post-quantum results (corresponding to Section 6.3 and 4.7 of this thesis, respectively).

3.1 Building Blocks

We start our analysis of classical cryptographic primitives by recalling some basic building blocks which we will use throughout the rest of this work. In what follows, $\mathcal{X}$ and $\mathcal{Y}$ are (sub)sets of binary strings. W.l.o.g., we assume that $\mathcal{X} = (\mathcal{X}_n)_n := (\{0,1\}^n)_n$. The *key space* $\mathcal{K}$ instead, is identified with $(\mathcal{K}_n)_n \subseteq \{0,1\}^{\mathscr{s}(n)}$ for security parameter $n \in \mathbb{N}$, where $\mathscr{s}$ is a polynomial function determined by the scheme considered. W.l.o.g. we assume that, for security parameter n, keys are of bit size n.

Pseudorandom Number Generators

A *pseudorandom number generator (PRNG)* is a DPT stateful algorithm which outputs bit strings with a distribution computationally indistinguishable from the uniform distribution over some set. There is no secret key involved, but a secret internal state of the algorithm determines the value to be output next. As the algorithm is deterministic, the same internal state produces the same output value, so the state must be updated after every execution, according to a procedure specified by the algorithm itself. The initial value of the PRNG's state is called the *seed*.

Formally, we give a slightly different definition.

Definition 3.1 (Pseudorandom Number Generator (PRNG))**.** *Let p be a polynomial such that $p(n) \geq n+1, \forall n \in \mathbb{N}$. A* pseudorandom number generator (PRNG) *with expansion factor p is a* DPT *algorithm $\mathcal{G}$ such that:*

1. *given as input a bit string $s \in \{0,1\}^n$, (the* seed*), outputs a bit string $\mathcal{G}(s) \in \{0,1\}^{p(n)}$; and*
2. *for any* PPT *algorithm $\mathcal{D}$:*

$$\left|\Pr\left[\mathcal{D}(r) \to 1\right] - \Pr\left[\mathcal{D}(\mathcal{G}(s)) \to 1\right]\right| \leq \mathsf{negl},$$

where $r \xleftarrow{\$} \{0,1\}^{p(n)}, s \xleftarrow{\$} \{0,1\}^n$, and the probabilities are taken over the choice of r and s, and the randomness of $\mathcal{D}$.

However, it is possible to show that with the above definition one can actually also define a procedure to output a stream of polynomially many values of bit size polynomial in n. The idea is to define a bit stream, where some of the $p(n)$ output bits are used to form the stream, and the others are used to generate a new, updated seed for the $\mathcal{G}$. Therefore, one usually speaks of *PRNG with n-bit output.* Analogously, it is easy to see that bits truncated by $\mathcal{G}$'s output are also pseudorandom.

Moreover, it is possible to prove that the condition of indistinguishability from random is equivalent to the condition of *non-predictability*, that is, no PPT algorithm can reliably guess the next bit output by $\mathcal{G}$, even by observing polynomially many bits output in the past. Clearly, if one could predict such bits then the pseudorandomness property would be violated. The other direction of the equivalence is known as Yao's Test [Yao82].

PRNGs have many useful applications. Given the existence of a PRNG, it is always possible (see, e.g., [Gol01]) to build a OWF (by encoding the input to the OWF as a seed for the PRNG), but it is also possible to show the converse. Namely, given a OWF $\mathcal{F}$, one can define a PRNG which outputs a hard-core bit of $\mathcal{F}$, computed on the seed. This construction can be iterated producing a PRNG which we denote by $\mathcal{G}_{\mathcal{F}}$, and which outputs polynomially many hard-core bits of $\mathcal{F}, \mathcal{F}^2, \mathcal{F}^3, \ldots$.

Construction 3.2 (Goldreich-Levin PRNG [GL89])**.** *Let $\mathcal{F} : \mathcal{X} \to \mathcal{Y}$ be a OWF with hard-core predicate $\mathsf{hc}_{\mathcal{F}}$. Define a stateful* DPT *algorithm $\mathcal{G}_{\mathcal{F}} : \mathcal{X} \to \mathcal{X}$ which, given as input an n-bit seed $x \in \mathcal{X}$, outputs the n-bit string:*

$$\mathsf{hc}_{\mathcal{F}}(x) \| \mathsf{hc}_{\mathcal{F}^2}(x) \| \ldots \| \mathsf{hc}_{\mathcal{F}^n}(x).$$

We call $\mathcal{G}_{\mathcal{F}}$ the Goldreich-Levin construction for OWF *$\mathcal{F}$.*

Theorem 3.3 ([GL89])**.** *Construction 3.2 is a PRNG.*

It must be noticed that the proof for the above theorem does not make any assumption on the adversary in terms of queries to the OWF. This fact will be important in the next chapter. It follows from Proposition 2.4 that a PRNG can be constructed by any OWF.

Corollary 3.4 (OWF ⇔ PRNG)**.** *OWFs exist iff PRNGs exist.*

Pseudorandom Functions

A (family of) *pseudorandom functions (PRF)* from $\mathcal{X}$ to $\mathcal{Y}$ with key space $\mathcal{K}$ is a family of efficiently computable functions $\mathcal{F} : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ which, without knowledge of the secret key $k \in \mathcal{K}$ indexing the particular member of the family, is computationally indistinguishable from the collection of all functions from $\mathcal{X}$ to $\mathcal{Y}$ (denoted by $\mathcal{Y}^{\mathcal{X}}$). We identify $\mathcal{F}$ as a DPT algorithm computing $\mathcal{F}$ for a specific security parameter n. As a shorthand notation, we write $\mathcal{F}_k : \mathcal{X} \to \mathcal{Y}$ meaning the member of the family indexed by $k \in \mathcal{K}$.

Definition 3.5 (Pseudorandom Function (PRF))**.** *A (family of)* pseudorandom functions (PRF) *from $\mathcal{X}$ to $\mathcal{Y}$ with key space $\mathcal{K}$ is a* DPT *algorithm $\mathcal{F} : (k \in \mathcal{K}_n, x \in \mathcal{X}_n) \mapsto y \in \mathcal{Y}_n$ such that for any* PPT *algorithm $\mathcal{D}$ it holds:*

$$\left| \Pr_{k \xleftarrow{\$} \mathcal{K}} \left[\mathcal{D}^{\mathcal{F}_k} \to 1 \right] - \Pr_{\hbar \xleftarrow{\$} \mathcal{Y}^{\mathcal{X}}} \left[\mathcal{D}^{\mathcal{O}_\hbar} \to 1 \right] \right| \le \mathsf{negl},$$

where $\mathcal{O}_\hbar$ is an oracle computing $\hbar$ (i.e., a random oracle), and the probabilities are over the choice of k and $\hbar$, and the randomness of $\mathcal{D}$.

In security reductions, PRFs are usually modeled as random oracles. However, unlike PRNGs, their security depends on the secrecy of the key used, because any party with knowledge of such key can trivially distinguish the PRF from a completely random function.

PRFs, being indistinguishable from random functions, can be used as PRNGs (for a vast majority of the possible keys).

Theorem 3.6 (PRF $\implies$ PRNG)**.** *If a PRFs exist, then PRNGs exist.*

Still, one can show that PRFs can be built by using PRNGs, and therefore their existence is equivalent to the existence of OWFs.

Theorem 3.7 ([GGM84])**.** *If a PRNGs exist, then PRFs exist.*

However, unlike in the case of Theorem 3.3, the proof does make assumptions on the query capabilities of the adversary.

Corollary 3.8. *OWF exist iff PRF exist.*

Pseudorandom Permutations

Pseudorandom permutations (PRP) are just PRFs which also happen to be (invertible) permutations on some space $\mathcal{X}$, for any choice of key. That is, a PRP $\mathcal{P}$ is a family of permutations (and their inverses) which is computationally indistinguishable from the family $S(\mathcal{X})$ of all the permutations on $\mathcal{X}$. As in the PRF case, we identify a PRP $\mathcal{P}$ with the DPT algorithm evaluating it, and as a shorthand notation, we write $\mathcal{P}_k : \mathcal{X} \to \mathcal{X}$ meaning the member of the (circuit or function) family indexed by $k \in \mathcal{K}$.

We start by defining a *weak* PRP, that is, indistinguishable from random to any adversary who does not have oracle access to the inverse permutation.

Definition 3.9 (Weak Pseudorandom Permutation (WPRP))**.** *A (family of)* weak pseudorandom permutations (WPRP) *on $\mathcal{X}$ with key space $\mathcal{K}$ is a pair of* DPT *algorithm $(\mathcal{P}, \mathcal{P}^{-1}) : (k \in \mathcal{K}, x \in \mathcal{X}) \mapsto x' \in \mathcal{X}$ such that:*

1. $\forall k \in \mathcal{K} \implies \mathcal{P}_k, \mathcal{P}_k^{-1}$ *are permutations on $\mathcal{X}$;*
2. $\forall k \in \mathcal{K} \implies (\mathcal{P}_k)^{-1} = \mathcal{P}_k^{-1}$*; and*
3. *for any* PPT *algorithm $\mathcal{D}$ it holds:*

$$\left| \Pr_{k \xleftarrow{\$} \mathcal{K}} \left[\mathcal{D}^{\mathcal{P}_k} \to 1 \right] - \Pr_{p \xleftarrow{\$} S(\mathcal{X})} \left[\mathcal{D}^{\mathcal{O}_p} \to 1 \right] \right| \leq \mathsf{negl},$$

where $\mathcal{O}_p$ is an oracle for p, and the probabilities are over the choice of k and p, and the randomness of $\mathcal{D}$.

In many applications though, we also need the possibility of inverting the permutations, hence we we also require the existence of another DPT algorithm $\mathcal{P}^{-1}$ computing the inverse permutation. A PRP is called *strong* if it maintains pseudorandomness also in this setting.

Definition 3.10 (Strong Pseudorandom Permutation (SPRP))**.** *A (family of)* strong pseudorandom permutations (SPRP) *on $\mathcal{X}$ with key space $\mathcal{K}$ is a pair of* DPT *algorithms* $(\mathcal{P}, \mathcal{P}^{-1}) : (k \in \mathcal{K}, x \in \mathcal{X}) \mapsto x' \in \mathcal{X}$ *such that:*

1. $\forall k \in \mathcal{K} \implies \mathcal{P}_k, \mathcal{P}_k^{-1}$ *are permutations on* $\mathcal{X}$*;*

2. $\forall k \in \mathcal{K} \implies (\mathcal{P}_k)^{-1} = \mathcal{P}_k^{-1}$*; and*

3. *for any* PPT *algorithm* $\mathcal{D}$ *it holds:*

$$\left| \Pr_{k \xleftarrow{\$} \mathcal{K}} \left[\mathcal{D}^{\mathcal{P}_k, \mathcal{P}_k^{-1}} \to 1 \right] - \Pr_{p \xleftarrow{\$} S(\mathcal{X})} \left[\mathcal{D}^{\mathcal{O}_p, \mathcal{O}_{p^{-1}}} \to 1 \right] \right| \leq \mathsf{negl},$$

where $\mathcal{O}_p$ is an oracle for p, $\mathcal{O}_{p^{-1}}$ is an oracle for p^{-1}, and the probabilities are over the choice of k and p, and the randomness of $\mathcal{D}$.

When left unspecified, by 'PRP' we mean the strong version. A PRP is clearly also a PRF, but not necessarily the other way around. However, there exist constructions of PRPs from PRFs, such as the *Feistel* construction. Therefore, the existence of PRPs is also equivalent to the existence of OWFs.

Theorem 3.11 (PRF $\Leftrightarrow$ PRP)**.** *PRFs exist iff PRPs exist.*

3.2 Secret-Key Encryption Schemes

A very fundamental object in cryptography is *secret-key (or, symmetric-key) encryption schemes (SKES)*. In what follows, $\mathcal{X}$ and $\mathcal{Y}$ represent the *plaintext and ciphertext message spaces* respectively, while $\mathcal{K}$ is the *key space*.

Definition 3.12 (Secret-Key Encryption Scheme (SKES))**.** *A* secret-key encryption scheme (SKES) *with plaintext space $\mathcal{X}$, ciphertext space $\mathcal{Y}$, and key space $\mathcal{K}$ is a tuple of* PPT *algorithms* $\mathcal{E} := \mathcal{E}_{\mathcal{K},\mathcal{X},\mathcal{Y}} := (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$*:*

1. $\mathsf{KGen} :\to \mathcal{K}$*;*

2. $\mathsf{Enc} : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$*;*

3. $\mathsf{Dec} : \mathcal{K} \times \mathcal{Y} \to \mathcal{X} \cup \{\bot\}$*;*

such that $\forall\ n \in \mathbb{N}, x \in \mathcal{X}, k \leftarrow \mathsf{KGen} \implies \mathsf{Dec}(k, \mathsf{Enc}(k, x)) = x$.

Notice the following:

- KGen only gets as input a security parameter, but $\mathsf{Enc}, \mathsf{Dec}$ also need as input the correct security parameter related to the second input (the secret key) they receive. In order to lighten notation and w.l.o.g. we just assume that n is also appended to every $k \leftarrow \mathsf{KGen}$, so that every key also implicitly contains the security parameter.

- Strictly speaking, it is not necessary to define $\perp$ as a possible output for Dec. However, this is useful when defining schemes which can also *reject* certain ciphertexts (such as *CCA2 secure encryption schemes*).

- As a shorthand notation, we will write Enc_k meaning the Enc algorithm with $k \in \mathcal{K}$ fixed as a first input; analogously for Dec_k.

- KGen is always assumed to be a nondeterministic algorithm, otherwise the encryption scheme would be trivial.

- Enc can be a probabilistic algorithm, so it is certainly possible that two different executions of $\mathsf{Enc}_k(x)$ for fixed k and x yield two different ciphertexts. However, those ciphertexts would still decrypt to the same x through Dec_k.

- As an immediate consequence of the previous point, it is clear that, for a given $k \in \mathcal{K}$, the image sets of different plaintexts are disjoint. That is: $x \neq x' \implies \mathrm{Supp}\left(\mathsf{Enc}_k(x)\right) \cap \mathrm{Supp}\left(\mathsf{Enc}_k(x')\right) = \varnothing$.

- The behavior of Dec_k is unspecified (and dependent on the SKES considered) if given as input an element of $\mathcal{Y}$ which is not a valid encryption, i.e., of the form $\mathsf{Enc}_k(x)$ for some $x \in \mathcal{X}$.

- If Enc_k is nondeterministic for all $k \in \mathcal{K}$, then we say that $\mathcal{E}$ is *randomized*, otherwise we say that $\mathcal{E}$ is *deterministic.*

Finally, notice that Definition 3.12 does not say anything about the *security* of a SKES. We will study this aspect in the next section. In particular, for a SKES to be considered 'secure', the size of $\mathrm{Supp}\left(\mathsf{KGen}(n)\right)$ must be superpolynomial in n. One of the most basic examples of SKES is the well known *one-time pad (OTP).*

Construction 3.13 (One-Time Pad (OTP))**.** *Let* $\mathcal{X} = \mathcal{K} = \mathcal{Y} = \{0,1\}^n$*. Define the* one-time pad (OTP) on n bits $\mathcal{E} = \mathcal{E}_{\mathcal{K},\mathcal{X},\mathcal{Y}} := (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ *as the SKES with key space* $\mathcal{K}$*, plaintext space* $\mathcal{X}$*, and ciphertext space* $\mathcal{Y}$*, defined as follows:*

1. $\mathsf{KGen} \to k$*, with* $k \xleftarrow{\$} \mathcal{K}$*;*

2. $\mathsf{Enc}_k(x) := x \oplus k$*;*

3. $\mathsf{Dec}_k(y) := y \oplus k$*.*

It is well known [Sha01] that the OTP is *information-theoretically secure*, as long as the key is completely random and only used once.

Semantic Security

In order to analyze the security of a SKES, we first have to define what it means for a SKES to be 'secure'. That is, we have to define a 'meaning', i.e., a *semantics* of the term 'security'. Intuitively, we want to formalize the fact that no reasonable adversary should be able, given a ciphertext, to find out any 'interesting' information about the underlying plaintext. There are three aspects to consider here.

First of all, we should define what a 'reasonable' adversary is. In our case we will consider computationally bounded adversaries, that is, adversaries as PPT algorithms, because we consider computational security. However, adversaries could be given additional power in the form of oracles. We will see a few examples in the next sections, while in this part we will start with the basic scenario (without oracles).

Secondly, we should define what constitutes 'interesting information' about the underlying plaintext. We do not consider 'interesting' all that information which is already publicly available, leaked, or manifest. For example, the length (bit size) of the plaintext is usually identifiable by only looking at the length of the ciphertext. Moreover, if some information about the plaintext is known a priori, e.g.: 'the message starts with a vowel', we do not consider an adversary succesful if he is only able to tell that the message starts with a vowel, because that fact is already known. We want security to protect the encryption scheme only against those adversaries who can extract 'interesting' information from the ciphertexts.

Finally, we should define the 'winning conditions' for our adversaries, so that we can define our schemes 'secure' if they prevent the adversaries from reaching those conditions. In theory, we could define a scheme to be 'secure' if every adversary fails consistently in his goals, regardless of the choice of keys and plaintexts he intends to attack. However, this is not reasonable to expect, for three reasons:

- the choice of some particular key might influence the adversary's winning probability. For example, what if the message is encrypted with a key that the adversary happens to know as well?

- The choice of the plaintexts is important as well. On one hand, we need the scheme to be secure even in the worst case scenario (that is, the best case scenario from the adversary's perspective.) On the other hand we cannot leave arbitrary freedom to the adversary in choosing the underlying plaintext - otherwise he could just break the encryption of a message he already knows, but that would not be 'interesting' information.

- The adversary might just get lucky. For example, when trying to decrypt a single bit of the message, he might just guess randomly, and still be succesful 50% of the times.

In literature, *semantic security* is the well-established golden standard in defining the security of an encryption scheme. Semantic security is a simulation-based security notion, where the success probability of an adversary trying to guess meaningful information about a ciphertext is compared to that of a *simulator*, which has the same goal as the adversary but is not allowed to see the ciphertext at all. The probability is taken over the internal randomness of the algorithms (and, hence, over all the keys), and 'interesting' and 'non-interesting' information is defined in terms of a *target function* f and an *auxiliary information function* h, respectively (these are functions of the possible plaintexts.) The goal of the adversary/simulator is to guess $f(x)$ when having access to $h(x)$, for a certain plaintext x drawn from a chosen distribution. The scheme is considered secure if the adversary and the simulator have roughly the same probability of guessing $f(x)$.

There are many, different but equivalent ways to define semantic security for SKES. In this work, we follow the approach from [Gol04].

Definition 3.14 (SEM Adversary, SEM Simulator)**.** *Let* $\mathcal{E} := \mathcal{E}_{\mathcal{K},\mathcal{X},\mathcal{Y}}$ *be a SKES, and* $f, h : \{0,1\}^* \to \{0,1\}^*$ *two functions efficiently computable and polynomially bounded in the input bit size. A* SEM adversary $\mathcal{A}$ for $\mathcal{E}$ *is a* PPT *algorithm* $\mathcal{A} : \mathcal{Y} \times Supp(h) \to Supp(f)$. *A* SEM simulator $\mathcal{S}$ for $\mathcal{E}$ *is a* PPT *algorithm* $\mathcal{S} : Supp(h) \to Supp(f)$.

Notice that, w.l.o.g., we can assume that $h(x)$ always includes the bit size of the plaintext x. We assume that h and f are efficiently computable, but actually, as shown in [Gol04], this is redundant.

Experiment 3.15 ($\mathsf{Game}^{\mathsf{SEM}}_{\mathcal{E},\mathcal{A}}$)**.** *Let* $\mathcal{E}$ *be a SKES, and* $\mathcal{A}$ *a SEM adversary. The* SEM experiment *proceeds as follows:*

```
Input: $n \in \mathbb{N}$, $f, h : \{0,1\}^* \to \{0,1\}^*$ efficiently computable and polynomially bounded in the input bit size, $\mathcal{M} := (\mathcal{M}_n)_n$, where $\mathcal{M}_n$ are probability distributions over $\mathcal{X}_n$ with $|\mathcal{M}_n| = \mathsf{poly}(n)$
$k \leftarrow \mathsf{KGen}$
$m \leftarrow \mathcal{M}_n$
$c \leftarrow \mathsf{Enc}_k(m)$                ▷ this is called 'SEM challenge query'
$f \leftarrow \mathcal{A}(c, h(m))$
if $f = f(m)$ then
    Output: 1
else
    Output: 0
```

Experiment 3.16 ($\mathsf{Game}^{\mathsf{SEM}*}_{\mathcal{E},\mathcal{S}}$)**.** *Let $\mathcal{E}$ be a SKES, and $\mathcal{S}$ a SEM simulator. The* simulated SEM experiment *proceeds as follows:*

```
Input: $n \in \mathbb{N}$, $\mathscr{f}, \mathscr{h} : \{0,1\}^* \to \{0,1\}^*$ efficiently computable and polynomially bounded in the input bit size, $\mathcal{M} := (\mathcal{M}_n)_n$, where $\mathcal{M}_n$ are probability distributions over $\mathcal{X}_1^n$ with $|\mathcal{M}_n| = \mathsf{poly}(n)$
$k \leftarrow \mathsf{KGen}$
$m \leftarrow \mathcal{M}_n$
$f \leftarrow \mathcal{S}(\mathscr{h}(m))$
if $f = \mathscr{f}(m)$ then
    Output: 1
else
    Output: 0
```

Definition 3.17 (Semantic Security (SEM))**.** *A SKES $\mathcal{E}$ is* semantically secure (SEM) *iff, for any SEM adversary $\mathcal{A}$ there exists a SEM simulator $\mathcal{S}$ such that, for every efficiently computable $\mathscr{f}, \mathscr{h} : \{0,1\}^* \to \{0,1\}^*$ polynomially bounded in the input bit size, for every probability ensemble $\mathcal{M} := (\mathcal{M}_n)_n$, where $\mathcal{M}_n$ are probability distributions over $\mathcal{X}_n$ with $|\mathcal{M}_n| = \mathsf{poly}(n)$, it holds:*

$$\left|\Pr\left[\mathsf{Game}^{\mathsf{SEM}}_{\mathcal{E},\mathcal{A}}(\mathcal{M}, \mathscr{f}, \mathscr{h}) \to 1\right] - \Pr\left[\mathsf{Game}^{\mathsf{SEM}*}_{\mathcal{E},\mathcal{S}}(\mathcal{M}, \mathscr{f}, \mathscr{h}) \to 1\right]\right| \leq \mathsf{negl},$$

where the probabilities are taken over the randomness of $\mathcal{A}, \mathcal{E}, \mathcal{M}, \mathcal{S}$.

Intuitively, the notion of SEM tells us the following: any information about the plaintext the adversary could guess from the ciphertext, could also be guessed by only looking at publicly available information. That means, the ciphertext does not leak any meaningful information about the plaintext. This security notion captures in a very complete way what we want from an encryption scheme, but it has the drawback of being quite involved formally, and cumbersome to use in security proofs. Because of this, different notions of security are often used, which are equivalent to SEM but easier to formalize.

Ciphertext Indistinguishability

Another notion of security for encryption schemes is *indistinguishability of ciphertexts (IND)*. Unlike SEM, this notion is game-based instead of simulation-based: there is no simulator at all, and security requires that no reasonable adversary can win a certain security game with probability substantially better than merely guessing. The IND security game consists in distinguishing the encryption of two different plaintexts (chosen by the adversary). Although, unlike in the case of SEM, it is unclear at a first glance that IND captures in a complete way exactly what we require from a 'secure' encryption scheme, we will see that the two notions are actually equivalent.

As in SEM, we model IND adversaries as PPT algorithms, as we are interested in computational security. However, in the IND game it is usually convenient to separate the adversary in two *stages*, each one with a specific function. The first stage, the *message generator* $\mathcal{M}$, chooses two messages from the plaintext space – the idea being that, in order to achieve the strongest security notion, the adversary is allowed to choose the most favourable scenario when playing this game. Then, one of these two messages is selected at random and encrypted with a key unknown to the adversary. Finally, the second stage of the adversary, the *distinguisher* $\mathcal{D}$, receives the resulting ciphertext, and his goal is to guess which one of the two plaintexts was encrypted. Formally, the adversary outputs a bit, and he wins the game if that bit is equal to the secret bit used to select one of the two plaintexts.

More formally, we define an *IND adversary* as follows.

Definition 3.18 (IND Adversary)**.** *Let $\mathcal{E}$ be a SKES. An* IND adversary $\mathcal{A}$ for $\mathcal{E}$ *is a pair of* PPT *algorithms $\mathcal{A} := (\mathcal{M}, \mathcal{D})$, where:*

1. $\mathcal{M} :\to \mathcal{X} \times \mathcal{X} \times \{0,1\}^*$ *is the* IND message generator*;*

2. $\mathcal{D} : \mathcal{Y} \times \{0,1\}^* \to \{0,1\}$ *is the* IND distinguisher

The security experiment related to the IND notion is as follows.

Experiment 3.19 ($\mathsf{Game}^{\mathsf{IND}}_{\mathcal{E},\mathcal{A}}$)**.** *Let $\mathcal{E}$ be a SKES, and $\mathcal{A} := (\mathcal{M}, \mathcal{D})$ an IND adversary. The* IND experiment *proceeds as follows:*

```
Input: $n \in \mathbb{N}$
$k \leftarrow \mathsf{KGen}$
$(m^0, m^1, \mathsf{state}) \leftarrow \mathcal{M}$
$b \xleftarrow{\$} \{0,1\}$
$c \leftarrow \mathsf{Enc}_k(m^b)$                ▷ this is called ‘IND challenge query’
$b' \leftarrow \mathcal{D}(c, \mathsf{state})$
if $b = b'$ then
    Output: 1
else
    Output: 0
```

The advantage of $\mathcal{A}$ *is defined as:*

$$\mathsf{Adv}^{\mathsf{IND}}_{\mathcal{E},\mathcal{A}} := \Pr\left[\mathsf{Game}^{\mathsf{IND}}_{\mathcal{E},\mathcal{A}} \to 1\right] - \frac{1}{2}.$$

Notice the following:

- $\mathcal{D}$ and $\mathcal{M}$ are part of the same ‘entity’ (the IND adversary $\mathcal{A}$), so that they should be allowed to exchange information. In particular, $\mathcal{D}$ should know which are the two original messages generated by $\mathcal{M}$. In the security game, this is modeled by exchanging a state string state from $\mathcal{M}$ to $\mathcal{D}$ (obviously this string has bit size at most polynomial in the security parameter since $\mathcal{M}$ is PPT.)

- There is no need to impose the condition that the two plaintexts generated by $\mathcal{M}$ must be distinct, as the security notion requires that *all* adversaries (including those who choose distinct messages) fail at winning the game.

- Since there are only two messages to choose from, the adversary can always win with 50% probability by guessing randomly. Therefore, the advantage of the adversary is measured in terms of doing better than merely guessing.

- The probability is over b and the internal randomness of $\mathcal{A}$ and KGen.

Definition 3.20 (Indistinguishability of Ciphertexts (IND))**.** *A SKES $\mathcal{E}$ has* indistinguishable encryptions (or, it is IND secure) *iff, for any IND adversary $\mathcal{A}$ it holds that:* $\mathsf{Adv}^{\mathsf{IND}}_{\mathcal{E},\mathcal{A}} \leq \mathsf{negl}$.

The advantage of the IND notion is that, being game-based, it is easier to use in cryptographic reductions. At the same time, one can show that it is equivalent to IND.

Theorem 3.21 ([Gol04])**.** *A SKES is IND secure iff it is SEM secure.*

Moreover, it has to be mentioned that the choice of defining the IND game in terms of two different messages is not compulsory: there are alternative definitions of the game where $\mathcal{M}$ only generates a message, and the other is either chosen randomly or set to 0, or where $\mathcal{M}$ generates polynomially many messages, and one of them is selected for the encryption. All these notions turn out to be equivalent, with small modifications.

An example of (unconditionally) IND secure SKES is the OTP.

The notions of IND can be augmented, i.e., made *stronger*, by granting extra power to the IND adversary in the form of *oracles.* Since the adversary acquires additional computational power in so doing, it might be the case that IND secure schemes now become insecure because of this extra power. Therefore, the resulting security notions are (potentially) stronger, and encryption schemes which are resistant against the new, augmented adversaries are automatically resistant to the weaker adversaries as well. The more power is given to the adversaries, the potentially stronger the security notion.

Traditionally, oracles have been used to model attack scenarios not covered by the IND notion alone. Of course, one could simply give the adversary unlimited access to a decryption oracle and make him super powerful. But that would make the security notion so strong to be unachievable – after all, SKES are not meant to protect by adversaries in possession of the secret key. Instead, other scenarios are considered.

Chosen Plaintext Attacks

In the *chosen plaintext attack (CPA)* scenario, the adversary is able to see encryptions of additional messages, in addition to the ones used in the IND game. He is allowed to choose the plaintexts to be encrypted by querying the encryption oracle Enc_k during the execution of the IND game. Moreover, he can perform the oracle queries in an *adaptive* way, i.e., reacting adaptively to the oracle's answers, for a polynomial number of queries, both before and after the IND challenge query. The resulting security game is as follows.

Experiment 3.22 ($\mathsf{Game}^{\mathsf{IND-CPA}}_{\mathcal{E},\mathcal{A}}$)**.** *Let $\mathcal{E}$ be a SKES, and $\mathcal{A} := (\mathcal{M}, \mathcal{D})$ an IND adversary. The* IND-CPA experiment *proceeds as follows:*

```
Input: $n \in \mathbb{N}$
$k \leftarrow \mathsf{KGen}$
$(m^0, m^1, \mathsf{state}) \leftarrow \mathcal{M}^{\mathsf{Enc}_k}$
$b \xleftarrow{\$} \{0,1\}$
$c \leftarrow \mathsf{Enc}_k(m^b)$
$b' \leftarrow \mathcal{D}^{\mathsf{Enc}_k}(c, \mathsf{state})$
if $b = b'$ then
    Output: 1
else
    Output: 0
```

The advantage of $\mathcal{A}$ *is defined as:*

$$\mathsf{Adv}^{\mathsf{IND-CPA}}_{\mathcal{E},\mathcal{A}} := \Pr\left[\mathsf{Game}^{\mathsf{IND-CPA}}_{\mathcal{E},\mathcal{A}} \to 1\right] - \frac{1}{2}.$$

Definition 3.23 (Indistinguishability of Ciphertexts under Chosen Plaintext Attack (IND-CPA))**.** *A SKES $\mathcal{E}$ has* indistinguishable encryptions under chosen plaintext attack (or, it is IND-CPA secure) *iff, for any IND adversary $\mathcal{A}$ it holds that:* $\mathsf{Adv}^{\mathsf{IND-CPA}}_{\mathcal{E},\mathcal{A}} \leq \mathsf{negl}$.

As discussed above, IND-CPA is clearly at least as strong as IND.

Theorem 3.24 (IND-CPA $\implies$ IND)**.** *If a SKES is IND-CPA secure, then it is also IND secure.*

But the converse is not true. In particular, all the encryption schemes that are not randomized cannot be IND-CPA secure, because then the adversary could always win the security game by first encrypting two messages of his choice, then performing the IND challenge, and then compare the resulting ciphertext with the encryption previously obtained. As an example, the OTP is not IND-CPA secure, despite being IND secure.

Theorem 3.25 (IND $\not\Longrightarrow$ IND-CPA)**.** *There exist SKES which are IND secure, but not IND-CPA secure.*

IND-CPA secure SKES can be constructed in a block-box way using PRFs.

Construction 3.26 ([Gol04, Construction 5.3.9])**.** *Let $\mathcal{F} : \mathcal{X} \to \mathcal{Y}$ be a PRF with key space $\mathcal{K}$. Define $\mathcal{E} = \mathcal{E}_{\mathcal{K},\mathcal{Y},\mathcal{Y}\times\mathcal{X}} := (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ as a SKES with key space $\mathcal{K}$, plaintext space $\mathcal{Y}$, and ciphertext space $\mathcal{Y} \times \mathcal{X}$, as follows:*

1. $\mathsf{KGen} \to k$, *with* $k \overset{\$}{\leftarrow} \mathcal{K}$;
2. $\mathsf{Enc}_k(x) \to (y, r)$, *with* $y := x \oplus \mathcal{F}_k(r)$, *where* $r \overset{\$}{\leftarrow} \mathcal{X}$;
3. $\mathsf{Dec}_{\mathsf{sk}}(y, r) := y \oplus \mathcal{F}_k(r)$.

Theorem 3.27. *Construction 3.26 is an IND-CPA SKES.*

Proof (sketch). The one-time pad is perfectly (statistically) secure if used with random, independent keys. This means that the only way to break the security of $\mathcal{E}$ is to break the security of $\mathcal{F}$. Since a fresh randomness r is chosen for every encryption, and since the image $\mathcal{F}_k(r)$ can be recovered by the related plaintext/ciphertext pairs, giving oracle access to Enc_k for the adversary is equivalent to giving oracle access to $\mathcal{F}_k$. However, by Definition 3.5, this is indistinguishable from a random oracle for any PPT adversary, so that the security of the one-time pad carries over, although only computationally. □

Then, recalling Corollary 3.4 and Theorem 3.7, we can state the following.

Corollary 3.28 (IND-CPA SKES from OWF)**.** *If OWFs exist, then IND-CPA SKES exist.*

Non-Adaptive Chosen Ciphertext Attacks

In the *non-adaptive chosen ciphertext attack (CCA1)* scenario, in addition to the IND-CPA capabilities, the adversary is able to also see decryptions of certain ciphertexts. As in the CPA case, he is allowed to choose the ciphertexts to be decrypted by querying the decryption oracle Dec_k during the execution of the IND game. However, unlike in the CPA case, he is only able to interact with this oracle *before* the IND challenge query, and not afterward. The adversary is allowed to perform the decryption oracle queries in an adaptive way, for a polynomial number of queries, but only before the IND challenge query, hence the term 'non-adaptive'[1]. Notice, in fact, that if the adversary were able to perform arbitrary decryption queries *after* the challenge query as well, this would allow him to decrypt the challenge ciphertext, and therefore it would render the security notion unachievable.

The resulting security game for the CCA1 scenario is as follows.

[1] Admittedly, this well-established term in the scientific literature is somewhat misleading, because this 'non-adaptivity' refers to 'in respect to the challenge ciphertext', while the queries to the decryption oracle can actually be performed adaptively.

Experiment 3.29 ($\mathsf{Game}^{\mathsf{IND-CCA1}}_{\mathcal{E},\mathcal{A}}$)**.** *Let $\mathcal{E}$ be a SKES, and $\mathcal{A} := (\mathcal{M}, \mathcal{D})$ an IND adversary. The* IND-CCA1 experiment *proceeds as follows:*

```
Input: $n \in \mathbb{N}$
$k \leftarrow \mathsf{KGen}$
$(m^0, m^1, \mathsf{state}) \leftarrow \mathcal{M}^{\mathsf{Enc}_k, \mathsf{Dec}_k}$
$b \xleftarrow{\$} \{0, 1\}$
$c \leftarrow \mathsf{Enc}_k(m^b)$
$b' \leftarrow \mathcal{D}^{\mathsf{Enc}_k}(c, \mathsf{state})$
if $b = b'$ then
    Output: $1$
else
    Output: $0$
```

The advantage of $\mathcal{A}$ *is defined as:*

$$\mathsf{Adv}^{\mathsf{IND-CCA1}}_{\mathcal{E},\mathcal{A}} := \Pr\left[\mathsf{Game}^{\mathsf{IND-CCA1}}_{\mathcal{E},\mathcal{A}} \to 1\right] - \frac{1}{2}.$$

Definition 3.30 (Indistinguishability of Ciphertexts under Non-Adaptive Chosen Ciphertext Attack (IND-CCA1))**.** *A SKES $\mathcal{E}$ has* indistinguishable encryptions under non-adaptive chosen ciphertext attack (or, it is IND-CCA1 secure) *iff, for any IND adversary $\mathcal{A}$ it holds that:* $\mathsf{Adv}^{\mathsf{IND-CCA1}}_{\mathcal{E},\mathcal{A}} \leq \mathsf{negl}$.

IND-CCA1 is clearly at least as strong as IND-CPA.

Theorem 3.31 (IND-CCA1 $\implies$ IND-CPA)**.** *If a SKES is IND-CCA1 secure, then it is also IND-CPA secure.*

But the converse is not true. There are IND-CPA secure SKES where, being able to decrypt different but related ciphertexts, can leak information about the secret key used.

Theorem 3.32 (IND-CPA $\centernot\implies$ IND-CCA1)**.** *There exists a SKES which is IND-CPA secure, but not IND-CCA1 secure.*

Proof (sketch). Consider a SKES $\mathcal{E}' = (\mathsf{KGen}', \mathsf{Enc}', \mathsf{Dec}')$ obtained by modifying another, IND-CPA secure SKES $\mathcal{E} = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ as follows:

1. $\mathsf{KGen}' \to (k, \overline{m})$,
   where $k \leftarrow \mathsf{KGen}$, and $\overline{m}$ is a special message, unknown to the adversary;

2. $\mathsf{Enc}'_k(m) \to \begin{cases} (\mathsf{Enc}_k(m), \mathsf{Enc}_k(\overline{m})) & \text{if } m \neq \overline{m}, \\ (\mathsf{Enc}_k(\overline{m}), k) & \text{otherwise;} \end{cases}$

3. $\mathsf{Dec}'_k(y, z) = \mathsf{Dec}_k(y)$.

The new SKES $\mathcal{E}'$ is still IND-CPA secure, because the probability for any adversary of guessing the plaintext $\overline{m}$ is negligible. However, in the CCA1 scenario it is trivial to break such modified scheme, by first performing a CPA query to obtain a valid ciphertext, then performing a CCA1 decryption query on the ciphertext obtained by swapping the two ciphertext halves, therefore recovering $\overline{m}$, and then performing another CPA query on $\overline{m}$, hence recovering the secret key. □

However, Construction 3.26 is also IND-CCA1.

Theorem 3.33. *Let $\mathcal{E}$ be the SKES from Construction 3.26. Then $\mathcal{E}$ is an IND-CCA1 SKES.*

Proof (sketch). Being able to perform decryption queries (before the challenge phase) gives to the adversary the possibility to forge new ciphertexts different (but related in a known way) to some other ciphertext of his choice. However, before the challenge phase, this does not provide any extra power, except the possibility of performing (polynomially many) extra queries to the PRF. □

Then, recalling Corollary 3.4 and Theorem 3.7, we can state the following.

Corollary 3.34 (IND-CCA1 SKES from OWF)**.** *If OWFs exist, then IND-CCA1 SKES exist.*

Adaptive Chosen Ciphertext Attacks

Finally, in the *adaptive chosen ciphertext attack* scenario, in addition to the IND-CCA1 capabilities, the adversary is able to query the decryption oracle also after the challenge query, *with an important exception*: he is not allowed to query Dec_k on the challenge ciphertext received. This restriction is necessary, as we have already discussed in the CCA1 case, otherwise the adversary could simply decrypt the challenge ciphertext and trivially win the game, and this would make the security notion unachievable. Formally, we have therefore to define a 'modified' decryption oracle, which is able to *reject* certain 'forbidden' decryption queries (those trying to decrypt the challenge ciphertext), by replying with a special symbol $\perp$ to those queries.

Definition 3.35 (CCA2 Oracle)**.** *Let $\mathcal{E} := (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ be a SKES, and $c \in Supp(\mathsf{Enc})$. The* CCA2 decryption oracle rejecting *c is defined by:*

$$\mathsf{Dec}_k^c(c') \longrightarrow \begin{cases} \mathsf{Dec}_k(c') & \text{if } c' \neq c, \\ \perp & \text{otherwise.} \end{cases}$$

The new security game is defined as follows.

Experiment 3.36 ($\mathsf{Game}_{\mathcal{E},\mathcal{A}}^{\mathsf{IND-CCA2}}$)**.** *Let $\mathcal{E}$ be a SKES, and $\mathcal{A} := (\mathcal{M}, \mathcal{D})$ an IND adversary. The* IND-CCA2 experiment *proceeds as follows:*

```
Input: $n \in \mathbb{N}$
$k \leftarrow \mathsf{KGen}$
$(m^0, m^1, \mathsf{state}) \leftarrow \mathcal{M}^{\mathsf{Enc}_k, \mathsf{Dec}_k}$
$b \xleftarrow{\$} \{0, 1\}$
$c \leftarrow \mathsf{Enc}_k(m^b)$
$b' \leftarrow \mathcal{D}^{\mathsf{Enc}_k, \mathsf{Dec}^c_k}(c, \mathsf{state})$
if $b = b'$ then
    Output: 1
else
    Output: 0
```

The advantage of $\mathcal{A}$ *is defined as:*

$$\mathsf{Adv}^{\mathsf{IND-CCA2}}_{\mathcal{E},\mathcal{A}} := \Pr\left[\mathsf{Game}^{\mathsf{IND-CCA2}}_{\mathcal{E},\mathcal{A}} \to 1\right] - \frac{1}{2}.$$

Definition 3.37 (Indistinguishability of Ciphertexts under Adaptive Chosen Ciphertext Attack (IND-CCA2))**.** *A SKES $\mathcal{E}$ has* indistinguishable encryptions under adaptive chosen ciphertext attack (or, it is IND-CCA2 secure) *iff, for any IND adversary $\mathcal{A}$ it holds that:* $\mathsf{Adv}^{\mathsf{IND-CCA2}}_{\mathcal{E},\mathcal{A}} \leq \mathsf{negl}$.

IND-CCA2 is clearly at least as strong as IND-CCA1.

Theorem 3.38 (IND-CCA2 $\implies$ IND-CCA1)**.** *If a SKES is IND-CCA2 secure, then it is also IND-CCA1 secure.*

But the converse is not true. There exist IND-CCA1 secure SKESs where an adversary able to decrypt ciphertexts which are different, but related, to the challenge ciphertext, can find out information about the underlying plaintext.

Theorem 3.39 (IND-CCA1 $\not\Longrightarrow$ IND-CCA2)**.** *There exist SKES which are IND-CCA1 secure, but not IND-CCA2 secure.*

Proof (sketch). The counterexample is given by Construction 3.26, as already hinted in the proof of Theorem 3.33. Being able to forge a valid ciphertext related in a controlled way to a target challenge ciphertext c allows the adversary to ask for decryptions of such ciphertexts without violating the CCA2 limitation that the ciphertext must be different from the challenge one. For example, the adversary might be able to ask for a decryption of $c \oplus 1 \dots 1$, therefore recovering $m \oplus 1 \dots 1$, where m was the original plaintext. □

Finally, although we are not going to write it down formally, it is possible to extend the SEM security notion to CPA, CCA1, and CCA2 scenarios as well, obtaining the security notions SEM-CPA, SEM-CCA1, and SEM-CCA2 respectively. Each of them can be shown to be equivalent to their IND counterpart. The situation is summarized in Figure 3.1.

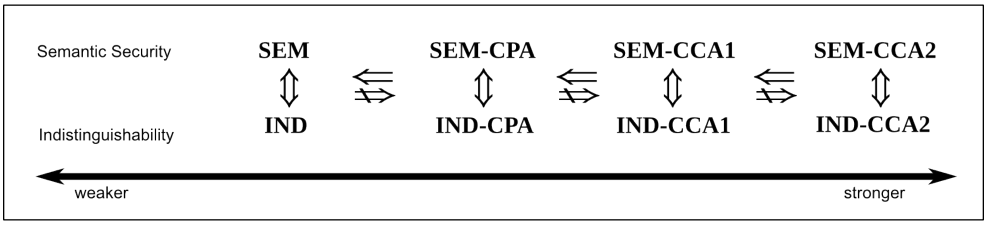


Figure 3.1: Relations for SKES security notions in **QS**0.

3.3 Public-Key Encryption Schemes

Another important cryptographic primitive are *public-key encryption schemes (PKES)*. Analogously to SKES, PKES work by encrypting messages from a plaintext space $\mathcal{X}$ to a ciphertext space $\mathcal{Y}$, and decrypting ciphertexts the other way around. The difference this time is that the key generation algorithm generates *pairs* of keys: a *public-key* pk which is only used to encrypt, and a *secret key* sk which is only used to decrypt. W.l.o.g. we assume that, for security parameter n, public keys are of bit size $\mathscr{p}(n)$, while secret keys are of bit size $\mathscr{s}(n)$, where $\mathscr{p}$,$\mathscr{s}$ are polynomial functions determined by the scheme considered. Under this notation, we identify the (public, private) keyspace $\mathcal{K}$ as $(\mathcal{K}_n)_n = (\mathcal{K}^{\mathscr{p}}{}_n)_n \times (\mathcal{K}^{\mathscr{s}}{}_n)_n =: \mathcal{K}^{\mathscr{p}} \times \mathcal{K}^{\mathscr{s}} \subset \{0,1\}^{\mathscr{p}(n)} \times \{0,1\}^{\mathscr{s}(n)}$.

Definition 3.40 (Public-Key Encryption Scheme (PKES))**.** *A* public-key encryption scheme (PKES) *with plaintext space $\mathcal{X}$, ciphertext space $\mathcal{Y}$, and key space $\mathcal{K} := \mathcal{K}^{\mathscr{p}} \times \mathcal{K}^{\mathscr{s}}$ is a tuple of* PPT *algorithms $\mathcal{E} := \mathcal{E}_{\mathcal{K},\mathcal{X},\mathcal{Y}} := (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$:*

1. $\mathsf{KGen} :\to \mathcal{K}$;
2. $\mathsf{Enc} : \mathcal{K}^{\mathscr{p}} \times \mathcal{X} \to \mathcal{Y}$;
3. $\mathsf{Dec} : \mathcal{K}^{\mathscr{s}} \times \mathcal{Y} \to \mathcal{X} \cup \{\perp\}$;

such that $\forall\, n \in \mathbb{N}$, $\forall\, x \in \mathcal{X}$, $\forall\, (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen} \implies \mathsf{Dec}(\mathsf{sk}, \mathsf{Enc}(\mathsf{pk}, x)) = x$.

As in the case of SKES, the following hold:

- we assume w.l.o.g. that n is also appended to every pk and every sk such that $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}$, so that every key also implicitly contains the security parameter.
- As a shorthand notation, we will write $\mathsf{Enc}_{\mathsf{pk}}$ meaning the Enc algorithm with $\mathsf{pk} \in \mathcal{K}^{\mathscr{p}}$ fixed as a first input; analogously for $\mathsf{Dec}_{\mathsf{sk}}$.
- If $\mathsf{Enc}_{\mathsf{pk}}$ is probabilistic for all $\mathsf{pk} \in \mathcal{K}^{\mathscr{p}}$, then we say that $\mathcal{E}$ is *randomized*, otherwise we say that $\mathcal{E}$ is *deterministic*.

The notions of security for PKES are basically the same as the ones for SKES, with two important differences:

1. because the public keys are, in fact, public, *all* the parties (including every stage of any adversary) can perform encryptions in polynomial time. Therefore, all parties have oracle access to $\mathsf{Enc}_{\mathsf{pk}}$. In particular, giving $\mathcal{M}$ and $\mathcal{D}$ the public key pk as input also implies access to $\mathsf{Enc}_{\mathsf{pk}}$.

2. As an immediate consequence, notice that for PKES, IND-CPA is the *minimal* meaningful security notion. In fact, if $\mathcal{E}$ is a PKES and $\mathcal{A}$ an IND adversary for $\mathcal{E}$, notice that $\mathsf{Game}^{\mathsf{IND-CPA}}_{\mathcal{E},\mathcal{A}} = \mathsf{Game}^{\mathsf{IND}}_{\mathcal{E},\mathcal{A}^{\mathsf{Enc}_{\mathsf{pk}}}}$.

Finally, as in the SKES case, it is clear that for a PKES to be IND-CPA secure, Supp $(\mathsf{KGen}(n))$ must be superpolynomial in n – actually, both $|\{\mathsf{pk} \in \mathcal{K}^p_n\}|$ and $|\{\mathsf{sk} \in \mathcal{K}^s_n\}|$ must be superpolynomial in n.

IND-CPA secure PKES can be built from OWTPs. Assume for simplicity that $\mathcal{X} = \{0,1\}^n$. Then we define the following.

Construction 3.41 (PKES from OWTP)**.** *Let* $\mathcal{P} := (\mathsf{Gen}, \mathsf{Eval}, \mathsf{Invert})$ *be a OWTP on* $\mathcal{X}$*, with index and trapdoor spaces* $\mathcal{I}$ *and* $\mathcal{T}$ *respectively, and let* $\mathcal{G}_{\mathcal{P}} : \mathcal{X} \to \mathcal{X}$ *be the Goldreich-Levin PRNG for* $\mathcal{P}$ *(seen as a OWF with hard-core predicates). Define* $\mathcal{E} = \mathcal{E}_{\mathcal{K},\mathcal{X},\mathcal{X}^2} := (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ *as a PKES with (public,private) key space* $\mathcal{K} = \mathcal{K}^p \times \mathcal{K}^s$ *(where* $\mathcal{K}^p := \mathcal{I}$ *and* $\mathcal{K}^s := \mathcal{T}$*, plaintext space* $\mathcal{X}$*, and ciphertext space* $\mathcal{X}^2$*, in the following way:*

1. $\mathsf{KGen} \to (\mathsf{pk}, \mathsf{sk})$*, with* $(\mathsf{pk}, \mathsf{sk}) := (i, t) \leftarrow \mathsf{Gen}$*;*

2. $\mathsf{Enc}_{\mathsf{pk}}(x) \to (y, z)$*,*
   with $y := x \oplus \mathcal{G}_{\mathcal{P}}(r)$ *and* $z \leftarrow \mathsf{Eval}(\mathsf{pk}, r)$*, where* $r \xleftarrow{\$} \mathcal{X}$*;*

3. $\mathsf{Dec}_{\mathsf{sk}}(y, z) := y \oplus \mathcal{G}_{\mathcal{P}}(s)$*, where* $s \leftarrow \mathsf{Invert}(\mathsf{pk}, \mathsf{sk}, z)$*.*

Theorem 3.42 (IND-CPA PKES from OWTP)**.** *Construction 3.41 is an IND-CPA secure PKES.*

Proof (sketch). If we omit the second half z of the ciphertext, then the indistinguishability of the encryptions immediately follows from the information-theoretical security of the OTP, as the key r of the OTP is always sampled indipendently and uniformly at random, and the output from the PRNG is computationally indistinguishable from random. So the only way to attack the scheme would be to extract information about the seed r of the PRNG, by looking at the OWTP image z obtained through Eval. However, since $\mathcal{G}_{\mathcal{P}}$ only outputs a sequence built from hard-core bits of $\mathcal{P}$, this would violate the one-wayness of the OWTP. □

3.4 Digital Signature Schemes

Digital signature schemes (DSS) are another fundamental cryptographic building block for many other advanced constructions. In a DSS, each user has a unique private/public key pair, as in PKES. However, the goal is not to protect the secrecy of the message, but its *authenticity*, intended as assurance about the identity of the originator of the message, and *integrity*, intended as a guarantee that the original message sent by the originator has not been altered prior to being received. This is achieved by computing a piece of information (the *signature*) to attach to a message, in such a way that everyone can verify that such signature could not be computed without possession of a specific secret key. More in detail, the signature is computed by the *sender* of a message using the sender's private key, and it is attached to the message. The *verifier*, upon receiving the message, checks the validity of the signature by using the sender's public-key. The signature is a (short) message– and secret-key– specific bit string, with the following properties:

1. for any message and any secret-key, it is efficiently computable; and
2. for any message, it is hard to generate a valid signature for any public-key without having the corresponding secret-key.

More formally, and borrowing the notation used in Section 3.3, we define a DSS as follows.

Definition 3.43 (Digital Signature Scheme (DSS))**.** *A* digital signature scheme (DSS) *with message space $\mathcal{X}$, signature space $\mathcal{T}$, and key space $\mathcal{K} := \mathcal{K}^{p} \times \mathcal{K}^{s}$ is a tuple of* PPT *algorithms $\mathcal{S}ig := \mathcal{S}ig_{\mathcal{K},\mathcal{X},\mathcal{T}} := (\mathsf{KGen}, \mathsf{Sign}, \mathsf{SigVerify})$:*

1. $\mathsf{KGen} :\to \mathcal{K}$;
2. $\mathsf{Sign} : \mathcal{K}^{s} \times \mathcal{X} \to \mathcal{T}$;
3. $\mathsf{SigVerify} : \mathcal{K}^{p} \times \mathcal{X} \times \mathcal{T} \to \{0,1\}$;

such that the following correctness condition holds:

$$\forall\, n \in \mathbb{N},\ \forall\, x \in \mathcal{X},\ \forall\, (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen},\ \forall\, \mathsf{sig} \leftarrow \mathsf{Sign}(\mathsf{sk}, x)$$
$$\Longrightarrow \mathsf{SigVerify}(\mathsf{pk}, x, \mathsf{sig}) = 1.$$

As in the case of SKES, the following hold:

- we assume w.l.o.g. that n is also appended to every pk and every sk such that $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}$, so that every key also implicitly contains the security parameter.
- As a shorthand notation, we will write $\mathsf{Sign}_{\mathsf{sk}}$ meaning the Sign algorithm with $\mathsf{sk} \in \mathcal{K}^{s}$ fixed as a first input; analogously for $\mathsf{SigVerify}_{\mathsf{pk}}$.

Existential Unforgeability

The notions of security for DSS is given in terms of *(strong) existential unforgeability under chosen message attack* (there are also weaker notions, but we will not use them here). An adversary is successful if he manages to create a valid signature for a message and public-key without having the corresponding secret key, even after observing a polynomial number of valid message/signature pairs.

Experiment 3.44 ($\mathsf{Game}^{\mathsf{EUF-CMA}}_{\mathcal{Sig},\mathcal{A}}$)**.** *Let $\mathcal{Sig}$ be a DSS, and $\mathcal{A}$ a* PPT *algorithm. The* EUF-CMA experiment *proceeds as follows:*

```
1: Input: $n, q_s \in \mathbb{N}$
2: $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}$
3: $(x, \mathsf{sig}) \leftarrow \mathcal{A}^{\mathsf{Sign}_{\mathsf{sk}}}(\mathsf{pk})$ after making at most $q_s$ queries to $\mathsf{Sign}_{\mathsf{sk}}$, receiving
   signatures $(x_1, \mathsf{sig}_1), \dots (x_{q_s}, \mathsf{sig}_{q_s})$
4: if $\mathsf{SigVerify}(\mathsf{pk}, x, \mathsf{sig}) = 1$ and $x \neq x_i \, \forall \, i = 1, \dots, q_s$ then
5:     Output: 1
6: else
7:     Output: 0
```

The advantage of $\mathcal{A}$ *is defined as:*

$$\mathsf{Adv}^{\mathsf{EUF-CMA}}_{\mathcal{Sig},\mathcal{A}}(n, q_s) := \Pr\left[\mathsf{Game}^{\mathsf{EUF-CMA}}_{\mathcal{Sig},\mathcal{A}}(n, q_s) \to 1\right].$$

Sometimes we also consider a slightly different version of Experiment 3.44, where the public/private key pair is given as an input to the game instead of being generated randomly. This is useful if we want to target a specific public key during some security reduction.

Definition 3.45 (Existential Unforgeability under Chosen Message Attack (EUF-CMA))**.** *A DSS $\mathcal{Sig}$ is* existentially unforgeable under chosen message attack (or, it is EUF-CMA secure) *iff, for any* PPT *algorithm $\mathcal{A}$ it holds that:*

$$\mathsf{Adv}^{\mathsf{EUF-CMA}}_{\mathcal{Sig},\mathcal{A}} \leq \mathsf{negl}.$$

Signatures in the Random Oracle Model

For certain applications it makes sense to investigate the security properties of signature schemes in the random oracle model. Recall that, in the ROM, all the parties involved gain access to an oracle $\mathcal{O}_{\hbar}$, where $\hbar$ is a function chosen uniformly at random from the set of all functions on certain spaces. This also means, in particular, that Definition 3.43 changes by allowing $\mathsf{KGen}, \mathsf{Sign}, \mathsf{SigVerify}$ oracle access to $\mathcal{O}_{\hbar}$. The resulting security model changes as follows.

Experiment 3.46 ($\mathsf{Game}^{\mathsf{EUF-CMA-RO}}_{\mathcal{Sig},\mathcal{A}}$)**.** *Let $\mathcal{Sig}$ be a DSS, $\mathcal{O}_{\hbar}$ a random oracle (computing a function $\hbar$ selected uniformly at random), and $\mathcal{A}$ a* PPT *algorithm. The* EUF-CMA-RO experiment *proceeds as follows:*

```
Input: $n, q_s, q_h \in \mathbb{N}$
$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}^{\mathcal{O}_{\mathscr{h}}}$
$(x, \mathsf{sig}) \leftarrow \mathcal{A}^{\mathsf{Sign}_{\mathsf{sk}}, \mathcal{O}_{\mathscr{h}}}(\mathsf{pk})$ after making at most $q_h$ queries to $\mathcal{O}_{\mathscr{h}}$, and $q_s$
    queries to $\mathsf{Sign}_{\mathsf{sk}}$ receiving signatures $(x_1, \mathsf{sig}_1), \dots (x_{q_s}, \mathsf{sig}_{q_s})$
if $\mathsf{SigVerify}(\mathsf{pk}, x, \mathsf{sig}) = 1$ and $x \neq x_i \, \forall \, i = 1, \dots, q_s$ then
    Output: 1
else
    Output: 0
```

The advantage of $\mathcal{A}$ *is defined as:*

$$\mathsf{Adv}^{\mathsf{EUF-CMA-RO}}_{\mathcal{S}ig,\mathcal{A}}(n, q_s, q_h) := \Pr\left[\mathsf{Game}^{\mathsf{EUF-CMA-RO}}_{\mathcal{S}ig,\mathcal{A}}(n, q_s, q_h) \rightarrow 1\right].$$

Definition 3.47 (Existential Unforgeability under Chosen Message Attack in the Random Oracle Model (EUF-CMA-RO))**.** *A DSS $\mathcal{S}ig$ is* existentially unforgeable under chosen message attack in the random oracle model (or, it is EUF-CMA-RO secure) *iff, for any* PPT *algorithm $\mathcal{A}$ it holds that:*

$$\mathsf{Adv}^{\mathsf{EUF-CMA-RO}}_{\mathcal{S}ig,\mathcal{A}} \leq \mathsf{negl}.$$

3.5 The Fiat-Shamir Transformation

The Fiat-Shamir (FS) transformation [FS86] is a well known method to remove interaction in three-move identification schemes between a prover and verifier (also called Σ-*protocol*), by letting the verifier's challenge ch be determined via a hash function $\mathscr{h}$ applied to the prover's first message com. Currently, the only generic, provably secure instantiation is by modeling the hash function $\mathscr{h}$ as a random oracle [BR93, PS00]. In this section, we will investigate the security of the FS transformation when applied to a Σ-protocol $(\mathcal{P}, \mathcal{V})$ in order to obtain a DSS $\mathcal{S}ig$, which we call *the FS transform of* $(\mathcal{P}, \mathcal{V})$.

Hard Languages

Let $\mathcal{L} \in \mathsf{NP}$ be a language with a (polynomially computable) relation $\mathcal{R}$, i.e., $\forall \, x : x \in \mathcal{L} \Leftrightarrow \exists \, w \in \mathcal{W} \subset \{0,1\}^{\mathsf{poly}(|x|)} : (x, w) \in \mathcal{R}$. In this case we also write that $x \in \mathcal{L}_n$ and $(x, w) \in \mathcal{R}_n$, for $n = |x|$. For using $\mathcal{L}$ in cryptographic applications, we need to discuss the following two issues:

1. given a statement $x \in \mathcal{L}$, how hard is to find a valid witness for x? And,

2. is it possible at all to find valid pairs $(x, w) \in \mathcal{R}$ in an efficient way?

For an interesting security notion, finding a witness from x alone should be infeasible for computationally bounded adversaries. On the other hand, it is useful to have a way to efficiently sample elements from the relation.

To this end we assume the existence of an efficient *hard instance generator* Inst, which on input the security parameter n outputs a pair $(x, w) \in \mathcal{R}_n$ such that no PPT algorithm can find valid witnesses for the overwhelming majority of statements contained in any of Inst's output. If $\mathcal{L}$ admits a hard instance generator, we say that $\mathcal{L}$ is a *hard language*.

Definition 3.48 (Hard Language and Instance Generator)**.** *Let $\mathcal{R}$ be an NP relation between language $\mathcal{L}$ and witness space $\mathcal{W}$. A PPT algorithm Inst is a* hard instance generator for $\mathcal{R}$ *iff the following hold:*

1. $(x, w) \in \mathcal{R}_n$*, for any* $(x, w) \leftarrow \mathsf{Inst}$*; and*
2. *for any* PPT *algorithm* $\mathcal{A}$ *it holds:*

$$\Pr_{(x,w)\leftarrow\mathsf{Inst}} [(x, \mathcal{A}(x)) \in \mathcal{R}] \leq \mathsf{negl}.$$

If $\mathcal{L}$ admits a hard instance generator, we say that $\mathcal{L}$ is a hard language*, and we denote it by $\mathcal{L}_{\mathcal{W},\mathcal{R},\mathsf{Inst}}$.*

Notice that the existence of a hard instance generator does not mean that it is hard to find a valid witness for *any* statement in $\mathcal{L}$. But this certainly holds for the vast majority of those statements in the subclass output by Inst. Moreover, the cardinality of this subclass is at least superpolynomial in n (otherwise PPT algorithms with oracle access to Inst could find valid witnesses by exhaustive search). This fact is used in the following paragraphs about the *Fiat-Shamir transformation* in order to show that large enough commitment spaces can be built from hard languages. Candidates for hard languages are at the base of many cryptographic constructions, and stem from NP problems such as *graph isomorphism* [GMW86], *decisional Diffie-Hellman for finite groups* [Bon98], and many others.

Identification Schemes

An *identification scheme (IS)* between a *prover* $\mathcal{P}$ and a *verifier* $\mathcal{V}$ is an interactive protocol which allows $\mathcal{P}$ to prove his *identity* to $\mathcal{V}$. By 'proving identity' we mean 'proving a statement about one's identity'. This is usually done with the help of a hard language $\mathcal{L}_{\mathcal{W},\mathcal{R},\mathsf{Inst}}$ where every user identity is bound to a certain statement; in practice, identities are usually linked to some public key, and for the prover to succeed he must prove ownership of the corresponding private key. We write $d \leftarrow (\mathcal{P}(x, w), \mathcal{V}(x))$ for the final outcome of the protocol, where $d \in \{0, 1\}$ is a bit denoting the final decision (acceptance or rejection) of the verifier.

ISs are related to a class of cryptographic objects known as *interactive proofs of knowledge*. Traditionally, the security notion for an IS is based on *impersonation security*, which intuitively states that no efficient adversary should be able to make $\mathcal{V}$ accept a statement x without knowing a valid witness

w. However, for the scope of this work, a weaker notion of security (which we call *weak impersonation security*) suffices. In this notion, additional effort is required for an adversary to be successful. Namely, given a statement x, the adversary should be able, after interacting with $(\mathcal{P}, \mathcal{V})$, to output a valid witness for x, breaking the security of the hard language. This, in particular, would allow the adversary to make $\mathcal{V}$ accept the execution of the scheme (but the converse is not necessarily true, that is why this notion is called 'weak'). Moreover, weak security comes in two variants, depending on the level of interaction that the adversary is allowed to have with $(\mathcal{P}, \mathcal{V})$. For *passive* adversaries, the only allowed interaction is given by observing and recording the executions of (at most polynomially many) sequential instances of the IS for a given statement. Therefore, passive weak security only relies on the hardness of the language $\mathcal{L}_{\mathcal{W},\mathcal{R},\mathsf{Inst}}$.

Definition 3.49 (Passively and Weakly Secure Identification Scheme (PWSIS))**.** *A* passively and weakly secure identification scheme (PWSIS)*, $(\mathcal{P}, \mathcal{V})$ for a hard language $\mathcal{L}_{\mathcal{W},\mathcal{R},\mathsf{Inst}}$ is an interactive protocol between two* PPT *algorithms $\mathcal{P}$ and $\mathcal{V}$ satisfying:*

$$\forall\, n, \forall\, (x, w) \leftarrow \mathsf{Inst} \implies (\mathcal{P}(x, w), \mathcal{V}(x)) \to 1.$$

An *active* adversary, instead, is also allowed to interact directly with $\mathcal{P}(x, w)$ by impersonating $\mathcal{V}$, and its goal is to output a valid witness for x given this interaction. That is, an active adversary $\mathcal{A} := (\mathcal{A}_1, \mathcal{A}_2)$ is a passive adversary who has also access to $\mathcal{P}(x, w)$ (seen as oracles). However, in order to avoid trivial breaks of the identification scheme (e.g., by man-in-the-middle attacks), during the security game the adversary can only be active before actually seeing x, and becomes passive afterwards. We express this as $\mathcal{A}_1^{\mathcal{P}(x,w)}(x)$. Obviously, if an IS is weakly secure against active attacks, it is also secure against passive attacks, but the converse does not necessarily hold. More formally, we define the following.

Definition 3.50 (Actively Weakly Secure Identification Scheme (AWSIS))**.** *An* actively and weakly secure identification scheme (AWSIS)*, $(\mathcal{P}, \mathcal{V})$ for a hard language $\mathcal{L}_{\mathcal{W},\mathcal{R},\mathsf{Inst}}$ is a PWSIS (according to Definition 3.49) such that, for every* PPT *algorithms $\mathcal{A}_1, \mathcal{A}_2$, the following holds:*

$$\Pr_{(x,w)\leftarrow \mathsf{Inst}}\left[(x, \mathcal{A}_2(x, \mathcal{A}_1^{\mathcal{P}(x,w)}(x))) \in \mathcal{R}\right] \leq \mathsf{negl}.$$

Σ-Protocols

A Σ-protocol for a hard language $\mathcal{L}_{\mathcal{W},\mathcal{R},\mathsf{Inst}}$ between a *prover* $\mathcal{P}$ and a *verifier* $\mathcal{V}$ is a 3-step interactive protocol which allows $\mathcal{P}$ to convince $\mathcal{V}$ that he knows a witness w for a public theorem $x \in \mathcal{L}$, without giving to $\mathcal{V}$ non-trivial information beyond this fact. Informally, a Σ-protocol $(\mathcal{P}, \mathcal{V})$ consists of an interactive exchange of three messages $(\mathsf{com}, \mathsf{ch}, \mathsf{resp})$ where the first message com (the *commitment*) is sent by $\mathcal{P}$, the second message ch (the *challenge*) is sampled uniformly from a challenge space by $\mathcal{V}$, and the last message resp (the *response*) is computed by $\mathcal{P}$ by using the witness. We write $(\mathsf{com}, \mathsf{ch}, \mathsf{resp}) \leftarrow (\mathcal{P}(x, w), \mathcal{V}(x))$ for the randomized output (the *communication transcript*) of an interaction between $\mathcal{P}$ and $\mathcal{V}$. We denote individual messages of the (stateful) prover in such an execution by $\mathsf{com} \leftarrow \mathcal{P}(x, w)$ and $\mathsf{resp} \leftarrow \mathcal{P}(x, w, \mathsf{com}, \mathsf{ch})$, respectively. Analogously, we denote the verifier's steps by $\mathsf{ch} \leftarrow \mathcal{V}(x, \mathsf{com})$ for the challenge step, and $d \leftarrow \mathcal{V}(x, \mathsf{com}, \mathsf{ch}, \mathsf{resp})$ for the final decision, where $d \in \{0, 1\}$ is a bit denoting acceptance or rejection.

More formally, we define the following.

Definition 3.51 (Σ-Protocol)**.** *A Σ-protocol (*'sigma-protocol'*) $(\mathcal{P}, \mathcal{V})$ for a hard language $\mathcal{L}_{\mathcal{W},\mathcal{R},\mathsf{Inst}}$ is a 3-move interactive protocol with exchange of messages* $\mathsf{com}, \mathsf{ch}, \mathsf{resp}$ *between two* PPT *algorithms $\mathcal{P}$ and $\mathcal{V}$ satisfying the following properties:*

1. ***Completeness:*** $\forall n \in \mathbb{N}, (x, w) \in \mathcal{R}_n, (\mathsf{com}, \mathsf{ch}, \mathsf{resp}) \leftarrow (\mathcal{P}(x, w), \mathcal{V}(x))$ *it holds that:* $\mathcal{V}(x, \mathsf{com}, \mathsf{ch}, \mathsf{resp}) = 1$*.*

2. ***Public-Coin:*** $\forall n \in \mathbb{N}, (x, w) \in \mathcal{R}_n, \mathsf{com} \leftarrow \mathcal{P}(x, w)$*, the challenge distribution* $\mathsf{ch} \leftarrow \mathcal{V}(x, \mathsf{com})$ *is uniform on* $\{0, 1\}^{\mathsf{poly}(n)}$*.*

3. ***Special Soundness:*** *there exists a* PPT *algorithm $\mathcal{J}$ (the* extractor*) such that, given two valid transcripts* $(\mathsf{com}, \mathsf{ch}, \mathsf{resp})$ *and* $(\mathsf{com}, \mathsf{ch}', \mathsf{resp}')$ *for $x \in \mathcal{L}$ (with* $\mathsf{ch} \neq \mathsf{ch}'$*) and* $\mathcal{V}(x, \mathsf{com}, \mathsf{ch}, \mathsf{resp}) = \mathcal{V}(x, \mathsf{com}, \mathsf{ch}', \mathsf{resp}') = 1$*, the extractor outputs a witness* $w \leftarrow \mathcal{J}(x, \mathsf{com}, \mathsf{ch}, \mathsf{resp}, \mathsf{ch}', \mathsf{resp}')$ *for x, satisfying $(x, w) \in \mathcal{R}$.*

4. ***Honest-Verifier Zero-Knowledge (HVZK):*** *there exists a* PPT *algorithm $\mathcal{S}$ (the* zero-knowledge simulator*) which, on input $x \in \mathcal{L}$, outputs a transcript* $(\mathsf{com}, \mathsf{ch}, \mathsf{resp})$ *that is computationally indistinguishable from a valid transcript derived in a $(\mathcal{P}, \mathcal{V})$ interaction. That is, for any* PPT *algorithm $\mathcal{V} = (\mathcal{V}_1, \mathcal{V}_2)$, the following two distributions are statistically indistinguishable:*

```
Input: $n \in \mathbb{N}$
$(x, w, \mathsf{state}) \leftarrow \mathcal{V}_1^*$
if $(x, w) \in \mathcal{R}$ then
    $(\mathsf{com},\mathsf{ch},\mathsf{resp}) \leftarrow (\mathcal{P}(x, w),\mathcal{V}(x))$
else
    $(\mathsf{com}, \mathsf{ch}, \mathsf{resp}) := (\bot, \bot, \bot)$
$b \leftarrow \mathcal{V}_2^*(\mathsf{com}, \mathsf{ch}, \mathsf{resp}, \mathsf{state})$
Output: $b$
```

```
Input: $n \in \mathbb{N}$
$(x, w, \mathsf{state}) \leftarrow \mathcal{V}_1^*$
if $(x, w) \in \mathcal{R}$ then
    $(\mathsf{com}, \mathsf{ch}, \mathsf{resp}) \leftarrow \mathcal{S}(x)$
else
    $(\mathsf{com}, \mathsf{ch}, \mathsf{resp}) := (\bot, \bot, \bot)$
$b \leftarrow \mathcal{V}_2^*(\mathsf{com}, \mathsf{ch}, \mathsf{resp}, \mathsf{state})$
Output: $b$
```

It turns out that Σ-protocols are also (passively, weakly-secure) identification schemes.

Theorem 3.52 (Σ-Protocols as IS)**.** *Let* $(\mathcal{P}, \mathcal{V})$ *be a* Σ*-protocol. Then* $(\mathcal{P}, \mathcal{V})$ *is a PWSIS.*

It is important to notice two things in the above theorem:

- HVZK is not necessary for Theorem 3.52 to hold; and
- a Σ-protocol may or may not be also an AWSIS.

The FS Transformation applied to Σ-Protocols

The Fiat-Shamir transformation of a Σ-protocol $(\mathcal{P}, \mathcal{V})$ is a modification of the protocol where the computation of ch is done as $\mathsf{ch} \leftarrow \hbar(x, \mathsf{com})$ instead of $\leftarrow \mathcal{V}(x, \mathsf{com})$. Here, $\hbar$ is a public hash function which is usually modeled as a random oracle $\mathcal{O}_\hbar$; in this case we speak of the *Fiat-Shamir (FS) transformation of* $(\mathcal{P}, \mathcal{V})$ *in the random-oracle model.* Note that we include x in the hash computation, but all of our results remain valid if x is omitted from the input. If applying the FS transformation to a Σ-protocol, one obtains a signature scheme, if the hash computation also includes the message m to be signed. We call the resulting signature scheme *FS transform of* $(\mathcal{P}, \mathcal{V})$ *in the ROM*, and we denote it by $\mathcal{S}ig_{\mathsf{FS}}^{\mathcal{O}_\hbar}(\mathcal{P}, \mathcal{V})$.

Definition 3.53 (FS Transform of a Σ-Protocol)**.** *Let* $(\mathcal{P}, \mathcal{V})$ *be a* Σ*-protocol for a hard language* $\mathcal{L}_{\mathcal{W},\mathcal{R},\mathsf{Inst}}$*, with commitment space* $\mathcal{X}$*, challenge space* $\mathcal{Y}$*, and response space* $\mathcal{Z}$*. Let* $\mathcal{O}_\hbar$ *be a random oracle for a random function* $\hbar : \mathcal{L} \times \mathcal{X} \times \mathcal{M} \to \mathcal{Y}$*. The* FS transform of $(\mathcal{P}, \mathcal{V})$ in the ROM, $\mathcal{S}ig_{\mathsf{FS}}^{\mathcal{O}_\hbar}(\mathcal{P}, \mathcal{V})$, *is a DSS with message space* $\mathcal{M}$*, signature space* $\mathcal{T} := \mathcal{X} \times \mathcal{Y} \times \mathcal{Z}$*, and key space* $\mathcal{K} := \mathcal{L} \times \mathcal{W}$*, defined as follows:*

1. $\mathsf{KGen} \to (\mathsf{pk}, \mathsf{sk})$*, where* $(\mathsf{pk}, \mathsf{sk}) := (x, w) \leftarrow \mathsf{Inst}$
2. $\mathsf{Sign}^{\mathcal{O}_\hbar}(\mathsf{sk}, m) \to \mathsf{sig} := (\mathsf{com}, \mathsf{ch}, \mathsf{resp})$*,*
   where $\mathsf{com} \leftarrow \mathcal{P}(\mathsf{pk}, \mathsf{sk})$*,* $\mathsf{ch} := \hbar(\mathsf{pk}, \mathsf{com}, m)$*,*
   and $\mathsf{resp} \leftarrow \mathcal{P}(\mathsf{pk}, \mathsf{sk}, \mathsf{com}, \mathsf{ch})$
3. $\mathsf{SigVerify}^{\mathcal{O}_\hbar}(\mathsf{pk}, m, \mathsf{sig}) \to b$*,*
   where $\mathsf{sig} := (\mathsf{com}, \mathsf{ch}, \mathsf{resp})$*,* $b \leftarrow \mathcal{V}(\mathsf{pk}, \mathsf{com}, \hbar(\mathsf{pk}, \mathsf{com}, m), \mathsf{resp})$

Notice that in the above signature the challenge ch can always be omitted (and it is infact ignored in the verification step), because it is recovered by computing $\hbar$ on the message, the commitment, and the public key. In this case we define the signature space of the DSS as $\mathcal{T} := \mathcal{X} \times \mathcal{Z}$. The following theorem states that the above construction yields secure DSSs in the ROM.

Theorem 3.54 (Security of a Fiat-Shamir Transform [PS00])**.** *Let* $(\mathcal{P}, \mathcal{V})$ *be a* Σ*-protocol. Then* $\mathcal{S}ig_{\mathsf{FS}}^{\mathcal{O}_{\mathfrak{H}}}(\mathcal{P}, \mathcal{V})$ *is an EUF-CMA-RO secure DSS.*

Sketch. The proof of this theorem uses rewinding. Intuitively, given a statement x and an adversary forging a signature for the DSS, this is used to extract a transcript $(\mathsf{com}, \mathsf{ch}, \mathsf{resp})$ for the underlying Σ-protocol. After that, the adversary is rewound, and the random oracle reprogrammed, in such a way that letting the adversary run again with the new oracle yields a related transcript $(\mathsf{com}, \mathsf{ch}', \mathsf{resp}')$ for the same com but $\mathsf{ch}' \neq \mathsf{ch}$. This, in turn, allows to use the special soundness property to extract a valid witness for x, therefore breaking the weak security of the underlying Σ-protocol, in contrast to Theorem 3.52. □

3.6 ORAMs

In this chapter we have presented many different cryptographic objects, in order of growing technical complexity. As a last example, we conclude with the concept of *Oblivious Random Access Machine (ORAM)*, and we define and analyze security models against classical adversaries.

Defining ORAMs in a fully formal way is a long, delicate, and strenuous task [GO96]. Therefore, in the following we will use a simplified model (introduced in [GKK17]) which covers most of the existing ORAM constructions without delving too much into the fine print - but still retaining a reasonable level of formalism - and which has the advantage of being much easier to treat.

Informally, an ORAM is an interactive protocol between two parties: a *client* $\mathcal{C}$ and a *server* $\mathcal{S}$, which we model as two PPT Turing machines (or, in our case, uniform families of circuits) sharing a communication tape (circuit register) Ξ to exchange data. In this scenario, a computationally limited $\mathcal{C}$ wants to outsource a *database (DB)* to the more powerful $\mathcal{S}$. Moreover, $\mathcal{C}$ wants to perform *operations* on the DB (by interactively communicating with $\mathcal{S}$) in such a way that $\mathcal{S}$, or any other honest-but-curious adversary $\mathcal{A}$ having read-only access to Ξ and $\mathcal{S}$'s internal memory, cannot determine the nature of such operations. The security notion for ORAM schemes is therefore a particular notion of *privacy*.

More formally: we define *blocks*, the basic storage units used in an ORAM construction. A block is an area of memory (circuit register) storing a n_{blk}-bit value, for a fixed parameter $n_{\mathsf{blk}} \in \mathbb{N}$ which depends on $\mathcal{C}$'s and $\mathcal{S}$'s architectures. A *database* (DB) of size $n_{\mathsf{db}} \in \mathbb{N}$ is an area of $\mathcal{S}$'s memory which stores an array $(\mathtt{block}_1, \ldots, \mathtt{block}_{n_{\mathsf{db}}})$ of such blocks. As we assume this database to reside on the server's side, we will denote it as $\mathcal{S}.\mathtt{DB}$. Notice that the precise way this array of blocks is represented in the database is unspecified, and left to the exact implementation of the ORAM scheme taken into account. For example, in the ORAM construction we are going to analyze in detail, the server's database $\mathcal{S}.\mathtt{DB}$ stores blocks in a binary tree structure. We will abuse notation and write that $\mathcal{S}.\mathtt{DB}(i) = \mathtt{block}$ if $\mathtt{block}$ is the i-th component of $\mathcal{S}.\mathtt{DB}$, and that $\mathtt{block} \in \mathcal{S}.\mathtt{DB}$ if $\mathtt{block}$ is stored at some position in the database $\mathcal{S}.\mathtt{DB}$.

Next we define *data units* as the basic units of data that the client wants to access, read, or write. Formally, a data unit is an n_{dat}-bit value for a fixed parameter $n_{\mathsf{dat}} \leq n_{\mathsf{blk}}$ which depends on $\mathcal{C}$'s and $\mathcal{S}$'s architectures. Every block encodes (usually in an encrypted form) a data unit, plus possibly auxiliary information such as a block identifier, checksum, or hash value. Since every block can encode a single data unit, at any given time t it is defined a function $\mathsf{Data}_t : \mathcal{S}.\mathtt{DB} \rightarrow \{0,1\}^{n_{\mathsf{dat}}}$. With abuse of notation, we will denote by $\mathsf{Data}(\mathtt{block})$ the data unit encoded in the block $\mathtt{block}$ at a certain time t. The client $\mathcal{C}$ can operate on the database through *data requests*.

Definition 3.55 (Data Request). *A* data request *to a database $\mathcal{S}$.DB of size n_{db} is a tuple* $\mathsf{dr} = (\mathsf{op}, i, \mathsf{data})$*, where* $\mathsf{op} \in \{\mathit{read}, \mathit{write}\}, i \in \{1, \ldots, n_{\mathsf{db}}\}$*, and* $\mathsf{data} \in \{0,1\}^{n_{\mathsf{dat}}}$ *is a data unit (*data *can also be* $\perp$ *if* $\mathsf{op} = \mathit{read}$*).*

Finally, we define the meaning of a *communication transcript* during an execution of an ORAM protocol. Since this also depends on the exact implementation of the ORAM scheme, we will use the following definition.

Definition 3.56 (Communication Transcript). *A* communication transcript com_t *at time t is the content of the communication channel* Ξ *at time t of the protocol's execution.*

Notice that the above defines the communication transcript as a function of time, but since an ORAM is a multi-round interactive protocol we will just consider com as a discrete function of the round $1, 2, \ldots$ of the protocol.

We are now ready to give a definition of ORAM. We assume that a server's database is always initialized empty (usually with randomized encryptions of 0 elements as blocks), and it is left up to the client the task of 'populating' the database with appropriate *write* operations.

Definition 3.57 (ORAM). *Let* $n_{\mathsf{Max}} \in \mathbb{N}, n_{\mathsf{msg}} \geq n_{\mathsf{dat}} \in \mathbb{N}$ *be fixed parameters, and* $\mathcal{E} = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ *be a SKES mapping* n_{msg}*-bit plaintexts to* n_{blk}*-bit ciphertexts. An* ORAM $\mathsf{ORAM}_{\mathcal{E}}$ *with parameters* $(n_{\mathsf{Max}}, n_{\mathsf{dat}}, \mathcal{E})$ *is a pair of two-party interactive randomized algorithms,* $(\mathsf{Init}, \mathsf{Access})$*, such that:*

- $\mathsf{Init}(n, n_{\mathsf{db}}) \rightarrow (\mathcal{C}, \mathcal{S})$ *in the following way:*

  1. *n is the security parameter,* $n_{\mathsf{db}} \leq n_{\mathsf{Max}}$*;*
  2. $k \leftarrow \mathsf{KGen}(n)$ *is generated by* $\mathcal{C}$*;*
  3. *$\mathcal{S}$ includes a database* $\mathcal{S}.DB = (\mathtt{block}_1, \ldots, \mathtt{block}_{n_{\mathsf{db}}})$*,*
     where $\forall\, i \implies \mathtt{block}_i \leftarrow \mathsf{Enc}_k(0)$*;*

- $\mathsf{Access}(\mathcal{C}, \mathcal{S}, \mathsf{dr}) \to (\mathcal{C}', \mathcal{S}', \mathsf{com})$ *in the following way:*

  1. *$\mathcal{C}$ issues a data request* dr*;*
  2. *$\mathcal{C}$ and $\mathcal{S}$ communicate through Ξ and produce the communication's transcript* com*;*

One might wonder why it is necessary to explicitly condition the definition of an ORAM in respect to a symmetric-key encryption scheme $\mathcal{E}$. It is actually possible to use different primitives, such as PKES, but most of the known ORAM constructions work well with just a simple primitive such as SKES. One might also wonder why the definition does not depend on other cryptographic primitives, such as PRNGs or PRFs. The reason is that not all ORAM constructions use such primitives, for example the 'trivial' ORAM scheme [GO96] (which consists in just transferring the whole encrypted database from $\mathcal{S}$ to $\mathcal{C}$ and back at every data request) does not use anything else than a SKES $\mathcal{E}$ as a building block. On the other hand, notice that encryption of the database is a minimal requirement for security, as we will see, therefore it makes sense to explicitly specify the scheme $\mathcal{E}$ in the notation.

An ORAM must satisfy *soundness* and *security*. We are going to define security in Section 4.7. Regarding soundness, the exact specification depends on the particular ORAM construction considered. A simplified, game-based definition of soundness (*'correctness'*) can be found in [GMP16], but it is difficult to adapt to the model from [GKK17] which we consider here, and which is more aimed at studying ORAM security, while a general definition (that can be found in [GO96]) is rather involved, and goes outside the scope of this work. The meaning of the soundness property is that the ORAM protocol 'should work', i.e., after any execution of Init or Access the two parties $\mathcal{C}$ and $\mathcal{S}$ must be left in such a state that allows them to continue the protocol in the next round. Despite the generality of this statement, in the model we consider here minimal soundness conditions can be identified, which must hold for *any* ORAM construction.

Definition 3.58 (Minimal ORAM Soundness Conditions)**.** *An ORAM construction* $\mathsf{ORAM}_\mathcal{E}$ *has* minimal soundness *if the following hold:*

1. *for any (n, n_{db}), if $(\mathcal{C}, \mathcal{S}) \leftarrow \mathsf{Init}(n, n_{\mathsf{db}})$, then $\mathcal{C}$ stores the secret key k from Def. 3.57;*
2. *for any* $\mathsf{dr} = (\mathsf{op}, i, \mathsf{data})$*, if* $(\mathcal{C}', \mathcal{S}', \mathsf{com}) \leftarrow \mathsf{Access}(\mathcal{C}, \mathcal{S}, \mathsf{dr})$*, then:*

   a) *if $\mathcal{C}$ stores the secret key k, then also $\mathcal{C}'$ stores k;*
   b) *if* $\mathsf{op} =$ *read and* $\mathcal{S}.DB(i) = \mathtt{block}$*, then $\mathcal{C}'$ stores* $\mathsf{Data}(\mathtt{block})$*;*
   c) *if* $\mathsf{op} =$ *write and* $\mathcal{S}'.DB(i) = \mathtt{block}$*, then* $\mathsf{Data}(\mathtt{block}) = \mathsf{data}$.

Notice that conditions 1 and 2a do not say anything about $\mathcal{S}$ having access to the key k or not: This is a property of *security*, not soundness, as we will see in Section 4.7. An ORAM scheme ORAM can have additional soundness conditions, depending on the particular construction. We assume that whenever $\mathcal{C}$ (resp., $\mathcal{S}$) is modified during the execution of the protocol to $\mathcal{C}$' (resp., $\mathcal{S}$') after Access calls, all these soundness conditions (the minimal ones above as well as the special ones) are always satisfied. In this case, we also say that $\mathcal{C}$' is a *sound evolution* of $\mathcal{C}$ and that $\mathcal{S}$' is a *sound evolution* of $\mathcal{S}$.

Classical Security of ORAMs

We now look at the security model for ORAMs against classical adversaries introduced in [GKK17]. Traditionally, the threat model in this case is defined by an *honest-but-curious adversary* $\mathcal{A}$. This means that $\mathcal{A}$ is some entity who wants to compromise $\mathcal{C}$'s privacy by having access to the communication channel Ξ and $\mathcal{S}$'s internal memory, but who is not allowed to modify the content of the channel or the database against the protocol, i.e., soundness must be preserved. In general, one does not lose generality by assuming that $\mathcal{S}$ itself is the adversary: $\mathcal{S}$ must behave 'honestly' (in the sense that he follows the protocol, in particular related to the protocol's soundness), but at the same time he will use all the information he can get through the interaction with $\mathcal{C}$ in order to compromise $\mathcal{C}$'s privacy. In particular, this also implies that $\mathcal{S}$ cannot know the key k generated during ORAM.Init, as noted above.

Formally, this model is defined in terms of *access patterns*, which are the adversarial views during an execution of data requests in ORAM.Access. Security requires that the adversary's view over a certain run of the protocol does not leak any information about the data requests executed by $\mathcal{C}$, except the sequences' length. This formulation reminds of the definition of semantic security for encryption schemes. As in that case, equivalent but easier-to-deal-with formulations can be given in terms of *computational indistinguishability of access patterns*. Following the security model introduced in [GKK17], we will consider an adaptive, game-based indistinguishability notion stating that for any two data requests, no computationally bounded adversary with knowledge of the access pattern of the client executing one of the two can distinguish which one was executed. This definition is equivalent [GKK17] to the simulation-based notion given in [GMP16], which states that no computationally bounded adversary can distinguish between the interaction with a real client or with a simulator that produces bogus transcripts.

More formally: when a data request is executed, we assume that the honest-but-curious adversary $\mathcal{A}$ records all the communication between $\mathcal{C}$ and $\mathcal{S}$, plus the changes in $\mathcal{S}$'s internal status. Without loss of generality, as we assume that $\mathcal{A}$ and $\mathcal{S}$ coincide, we assume that the only meaningful changes in the database area $\mathcal{S}$.DB only happen between the beginning and the end of an Access execution. The communications are polynomially bounded and, for simplicity, we assume that the channel Ξ does not erase symbols, i.e., it is write-once. Hence, the adversarial view is composed of the communication transcript, and the server's database before and after the execution of the data request. We call this adversarial view, the *access pattern* of the execution.

Definition 3.59 (Access Pattern)**.** *Given ORAM client and server $\mathcal{C}$ and $\mathcal{S}$, and a data request* dr*, the* access pattern $\mathsf{ap}(\mathsf{dr})$ *is the tuple* $(\mathcal{S}.DB, \mathsf{com}, \mathcal{S}'.DB)$*, where* $(\mathcal{C}', \mathcal{S}', \mathsf{com}) \leftarrow \mathsf{Access}(\mathcal{C}, \mathcal{S}, \mathsf{dr})$*.*

Next, we define formally a *classical ORAM adversary.*

Definition 3.60 (Classical ORAM Adversary)**.** *A* classical ORAM adversary $\mathcal{A}$ *is a* PPT *algorithm which is computationally indistinguishable from an honest server* $\mathcal{S}$ *for every ORAM client* $\mathcal{C}$ *(in particular, soundness is preserved).*

Notice the following fact: this definition of adversary can generally be *stronger* than in the usual 'honest-but-curious' meaning. In fact, such adversary could still manipulate the channel and the database in a malicious way, as long as the client $\mathcal{C}$ cannot detect such manipulation – in particular, the soundness of the protocol must be preserved. We define the security of an ORAM through the following indistinguishability game.

Experiment 3.61 ($\mathsf{Game}^{\mathsf{AP-IND-CQA}}_{\mathsf{ORAM},\mathcal{A}}$)**.** *Let* $\mathsf{ORAM} = (\mathsf{Init}, \mathsf{Access})$ *be an ORAM construction with parameters* $(n_{\mathsf{Max}}, n_{\mathsf{dat}}, \mathcal{E})$*,* n *a security parameter and* $\mathcal{A}$ *a classical ORAM adverary. The* computational indistinguishability of access patterns game under adaptive chosen query attack $\mathsf{Game}^{\mathsf{AP-IND-CQA}}_{\mathsf{ORAM},\mathcal{A}}$ *proceeds as follows:*

```
Input: $n \in \mathbb{N}$
$\mathcal{A} \to (\mathcal{A}_0, \mathsf{dr}_1, n_{\mathsf{db}} \leq n_{\mathsf{Max}})$
$(\mathcal{C}_0, \mathcal{S}_0) \leftarrow \mathsf{Init}(n, n_{\mathsf{db}})$
loop for $i = 1, \ldots, q_1 \in \mathbb{N}$:                    ▷ first CQA learning phase
    $\mathsf{Access}(\mathcal{C}_{i-1}, \mathcal{S}_{i-1}, \mathsf{dr}_i) \to (\mathcal{C}_i, \mathcal{S}_i, \mathsf{ap}_i)$
    $\mathcal{A}_{i-1}(\mathsf{ap}_i, \mathcal{S}_i) \to (\mathcal{A}_i, \mathsf{dr}_{i+1})$
$\mathcal{A}_{q_1}(\mathsf{dr}_{q_1+1}) \to (\mathcal{A}', \mathsf{dr}^0, \mathsf{dr}^1)$
$b \xleftarrow{\$} \{0,1\}$
$\mathsf{Access}(\mathcal{C}_{q_1}, \mathcal{S}_{q_1}, \mathsf{dr}^b) \to (\mathcal{C}_{q_1+1}, \mathcal{S}_{q_1+1}, \mathsf{ap}_{q_1+1})$    ▷ AP-IND challenge query
$\mathcal{A}'(\mathsf{ap}_{q_1+1}, \mathcal{S}_{q_1+1}) \to (\mathcal{A}_{q_1+1}, \mathsf{dr}_{q_1+2})$
loop for $i = q_1 + 2, \ldots, q_2 \geq q_1 + 2$:              ▷ second CQA learning phase
    $\mathsf{Access}(\mathcal{C}_{i-1}, \mathcal{S}_{i-1}, \mathsf{dr}_i) \to (\mathcal{C}_i, \mathcal{S}_i, \mathsf{ap}_i)$
    $\mathcal{A}_{i-1}(\mathsf{ap}_i, \mathcal{S}_i) \to (\mathcal{A}_i, \mathsf{dr}_{i+1})$
$\mathcal{A}_{q_2}(\mathsf{dr}_{q_2+1}) \to b' \in \{0,1\}$
if $b = b'$ then
    Output: 1
else
    Output: 0
```

The advantage of $\mathcal{A}$ *is defined as:*

$$\mathsf{Adv}^{\mathsf{AP-IND-CQA}}_{\mathsf{ORAM},\mathcal{A}} := \Pr\left[\mathsf{Game}^{\mathsf{AP-IND-CQA}}_{\mathsf{ORAM},\mathcal{A}} \to 1\right] - \frac{1}{2}.$$

In this game the adversary, after selecting suitable ORAM parameters of his choice, is first allowed to see the access patterns originated by executions of Access for data requests of his choice, chosen adaptively one after the other (this is called 'first CQA learning phase'.) At some point, the adversary issues a challenge query composed of two (w.l.o.g. different) data requests. One of the two is selected at random and executed through Access, and the adversary, after being allowed a second CQA learning phase, must guess which one of the two was executed. Notice that, since $\mathcal{A}$ is polynomially bounded, q_1 and q_2 are at most polynomials in n. We are now ready to define the classical security notion for ORAMs.

Definition 3.62 (Access Pattern Indistinguishability Under Adaptive Chosen Query Attack)**.** *An ORAM construction* ORAM *has* computationally indistinguishable access patterns under adaptive chosen query attack *(or, it is AP-IND-CQA-secure) iff for any classical ORAM adversary $\mathcal{A}$ it holds that* $\mathsf{Adv}^{\mathsf{AP-IND-CQA}}_{\mathsf{ORAM},\mathcal{A}} \leq \mathsf{negl}$.

PathORAM

As an example of ORAM construction, we recall here PathORAM, one of the most efficient ORAM constructions proposed to date, introduced by Stefanov et al. in [SvDS+13]. We only give a high-level explanation of PathORAM, and for a thorough description of the construction, as well as a detailed proof of its functionality, we refer to [SvDS+13].

In PathORAM a client stores n_{db} blocks of bit size n_{blk} on a server, in a binary tree structure of height $n_{\mathsf{tree}} = \lceil \log_2 n_{\mathsf{db}} \rceil$. Each node of the tree can store a constant amount n_{bkt} of blocks. Every block encodes (in an encrypted form, using an IND-CPA SKES) a data unit of bit size n_{dat}, and optionally additional information which is used to label the block for efficient retrieval. There are many different ways one can implement this labeling of the blocks. In our case we will use the simple approach of concatenating to the data unit data an n_{tag}-bit string encoding the block identifier $i \in \{1, \ldots, n_{\mathsf{db}}\}$, that is, blocks are of the form $\mathtt{block}_i \leftarrow \mathsf{Enc}_k(i \| \mathsf{data}_i)$. This system is very general, and as we will see it has the advantage that it easily translates to the quantum setting, unlike other approaches such as identifying blocks by using a hash table. At the beginning, all the blocks in the tree are initialized in an 'empty' state, which is defined by setting to 0 the identifying label – recall in fact that valid block identifiers are $1, \ldots, n_{\mathsf{db}}$ only. Every block is mapped to a leaf of the tree, and this mapping is recorded in a correspondence table, called *position map*, by the client[2].

A read (or write) operation for a block $\texttt{block}_i$ is performed by the client, by downloading the path (tree branch) from the root of the tree to the leaf indicated in the client's position map, and randomly remapping $\texttt{block}_i$ to another leaf in the position map. Then the client decrypts and re-encrypts (re-randomizing) all the blocks in the downloaded path, and for every valid (non-empty) block $\texttt{block}_j$ found, the client checks its corresponding leaf in the position map, and moves $\texttt{block}_j$ (if there is enough available space) to the node in the path which is closest to the leaf level and that belongs both to the downloaded path and the path to the leaf of $\texttt{block}_j$ given by the position map. If a block does not fit anywhere in the downloaded path, then an extra storage, called *'stash'* is used by the client to store this overflowing block locally. The blocks found in the stash are also examined during every read (or write) operation and checked if they can be evicted from the stash and placed in the tree. Since the stash must be stored locally by the client, the stash's size should be reasonably small; in fact, in [SvDS+13], the authors show that the probability that the stash exceeds a size of $\mathcal{O}(\log n_{\mathsf{db}})$ is negligible in the number of queries. The intuition is to notice that the stash is only used if the tree root is full, but the average action of a data request is to push only $\texttt{block}_i$ toward the tree root, and push many other blocks $\texttt{block}_j$ toward the leaf level. In the following we will mostly ignore the use of the stash for simplicity.

More concretely, we give here a full description of PathORAM (which we denote as $\texttt{PathORAM}$) according to the formalism introduced.

Construction 3.63 ($\texttt{PathORAM}$ [GKK17, Definition 18])**.** *For fixed parameters* $n_{\mathsf{dat}}, n_{\mathsf{Max}} \in \mathbb{N}$*, let* $n_{\mathsf{tag}} = \lceil \log_2 n_{\mathsf{Max}} \rceil, n_{\mathsf{bkt}} \in \mathbb{N}, n_{\mathsf{msg}} = n_{\mathsf{dat}} + n_{\mathsf{tag}}, n_{\mathsf{blk}} \geq n_{\mathsf{msg}}$*. Let* $\mathcal{G}$ *be a PRNG outputting* n_{tag}*-bit values, and* $\mathcal{E} = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ *be a SKES with* n_{msg}*-bit plaintexts and* n_{blk}*-bit ciphertexts. We define an ORAM construction called* $\texttt{PathORAM} = \texttt{PathORAM}_{\mathcal{E},\mathcal{G}}$ *as follows:*

- $\mathsf{Init}(n, n_{\mathsf{db}}) \to (\mathcal{C}, \mathcal{S})$ *in the following way:*

```
$\mathcal{C}$ generates a secret key $k \leftarrow \mathsf{KGen}$
set $n_{\mathsf{tree}} := \lceil \log_2 n_{\mathsf{db}} \rceil$                ▷ notice $n_{\mathsf{tree}} \leq n_{\mathsf{tag}}$
$\mathcal{C}$ initializes a position map of the form $((1, r_1), \ldots, (n_{\mathsf{db}}, r_{n_{\mathsf{db}}}))$, where
    $r_i$ are $n_{\mathsf{tree}}$-bit values generated by truncating bits from $\mathcal{G}$'s output
$\mathcal{S}$.DB is stored in a binary tree of height $n_{\mathsf{tree}}$, with root $\texttt{Root}$ and
    leaves $\texttt{Leaf}_0, \ldots, \texttt{Leaf}_{2^{n_{\mathsf{tree}}-1}}$, and such that:
    1. each node of the tree stores up to $n_{\mathsf{bkt}}$ blocks;
    2. every block of every node is initialized to $\mathsf{Enc}_k(0^{n_{\mathsf{tag}}} \| 0^{n_{\mathsf{dat}}})$.
```

[2] Note that the size of the position map is linear in the number of blocks that the client has, and thus cannot be stored locally by the client. The authors of [SvDS+13] propose storing the position map recursively to smaller PathORAMs following an idea from [SSS12]. For ease of exposition however, we will assume here that the position map is stored locally.

- *If* $\mathsf{dr} = (\mathsf{op}, i, \mathsf{data})$, *then* $\mathsf{Access}(\mathcal{C}, \mathcal{S}, \mathsf{dr}) \to (\mathcal{C}', \mathcal{S}', \mathsf{com})$ *as follows:*

```
1: $\mathcal{C}$ reads $r_i$ from his position map and sends it to $\mathcal{S}$
2: $\mathcal{S}$ sends to $\mathcal{C}$ the path Branch from Root to $\mathtt{Leaf}_{r_i}$
3: remap $(i, r_i)$ to $(i, r'_i)$ in the position map of $\mathcal{C}$, where $r'_i$ is a fresh
   pseudorandom $n_{\mathsf{tree}}$-bit value (generated by truncating the first $n_{\mathsf{tag}} - n_{\mathsf{tree}}$ bits of $\mathcal{G}$'s output), obtaining $\mathcal{C}'$
4: for all block contained in Branch do
5:     $\mathcal{C}'$ decrypts $\mathsf{Dec}_k(\mathtt{block}) \to (j \| \mathsf{data}_j) \in \{0,1\}^{n_{\mathsf{msg}}}$,
         where $j \in \{0,1\}^{n_{\mathsf{tag}}}$, $\mathsf{data}_j \in \{0,1\}^{n_{\mathsf{dat}}}$
6:     if $j = i$ then
7:         if $\mathsf{op} =$ 'read' then
8:             $\mathcal{C}'$ reads $\mathsf{data}_j$                    ▷ $\mathcal{C}'$ now has access to $\mathsf{data}_j$
9:         else if $\mathsf{op} =$ 'write' then
10:            $\mathcal{C}'$ sets $\mathsf{data}_j = \mathsf{data}$                      ▷ block is updated
11:    $\mathcal{C}'$ re-encrypts (re-randomizing) block
12:    find in Branch the common parent node Node between $\mathtt{Leaf}_{r_i}$
         and $\mathtt{Leaf}_{r_j}$, closer to the leaf level
13:    set $b_{\mathsf{swap}}$ := 'false'
14:    for all block' in Node do
15:        $\mathcal{C}'$ decrypts $\mathsf{Dec}_k(\mathtt{block}') \to (j' \| \mathsf{data}'_j) \in \{0,1\}^{n_{\mathsf{msg}}}$
16:        $\mathcal{C}'$ re-encrypts (re-randomizing) $\mathtt{block}'' \leftarrow \mathsf{Enc}_k(j' \| \mathsf{data}'_j)$
17:        if $j' = 0 \ldots 0$ then                 ▷ block'' is empty, can be used
18:            swap block and block''
19:            set $b_{\mathsf{swap}}$ := 'true'
20:    if $b_{\mathsf{swap}} =$ 'false' then            ▷ no empty blocks in current Node
21:        if Node $\neq$ Root then
22:            set Node to be one level up in the tree (i.e., Node's parent)
23:            go to step 14
24:        else
25:            store block in the Stash              ▷ no empty blocks found
26: $\mathcal{C}'$ sends back the updated tree branch, NewBranch, to $\mathcal{S}$
27: update $\mathcal{S}$.DB with NewBranch, obtaining $\mathcal{S}'$
28: produce com, which contains $r_i$, Branch, NewBranch
```

In the above, we recap the meaning of the parameters as follows:

- n is the security parameter, used by the encryption scheme $\mathcal{E}$.

- n_{Max} is the maximum number of blocks that the server's architechture can support (an upper bound to $\mathcal{S}$'s tree storage).

- n_{db} is the maximum number of 'real' blocks that the client $\mathcal{C}$ wants to store (so, $n_{\mathsf{db}} \leq n_{\mathsf{Max}}$). Unlike n_{Max} thus, n_{db} can be chosen by the adversary in the security game.

- n_{tag} is the minimum number of bits that are needed to index all the 'real' blocks in the limit scenario where $n_{\mathsf{db}} = n_{\mathsf{Max}}$. Hence, n_{tag} is also architecture-dependant, and not chosen by $\mathcal{A}$.

- n_{bkt} is the maximum number of blocks that can be stored in every tree node. Lower values reduce the amount of memory used by $\mathcal{S}$ to store the tree (for a fixed n_{db}), but increase the risk of using large amounts of memory by the client for the stash. This is a parameter of the particular `PathORAM` implementation: as we do not care about performance analysis here, we will leave n_{bkt} undefined, as any nonzero value works for us.

- n_{tree} is the minimum number of bits that are needed to index all the 'real' n_{db} blocks (hence, $n_{\mathsf{tree}} \leq n_{\mathsf{tag}}$). n_{tree} also represents the minimum height of the tree necessary to store all blocks in the limit case $n_{\mathsf{bkt}} = 1$.

- n_{dat} is the bit size of the data units used in the `PathORAM` implementation, and it is hence architecture-dependant.

- n_{msg} is the total bit size of a data unit, plus the number of bits necessary to address the block where this data unit is encoded, so also this value is architecture-dependant. The encryption scheme $\mathcal{E}$ must be able to work with n_{msg}-bit plaintexts.

- n_{blk} is the size of a ciphertext produced by the encryption scheme $\mathcal{E}$, and hence the total size of a block. The size of $\mathcal{S}$'s tree storage memory is thus at most $n_{\mathsf{blk}} n_{\mathsf{Max}}$ bits.

We now show the (classical) security of `PathORAM`.

Theorem 3.64 (AP-IND-CQA Security of `PathORAM`)**.** *Let $\mathcal{E} = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ be an IND-CPA SKES, and let $\mathcal{G}$ be a PRNG. Then, `PathORAM` instantiated using $\mathcal{E}$ and $\mathcal{G}$ is an AP-IND-CPA secure ORAM.*

Proof. By assumption, the outputs of $\mathcal{G}$ are indistinguishable from random. Therefore, in the following analysis, we can w.l.o.g. replace $\mathcal{G}$ with a real source of randomness.

Suppose that there exists an adversary $\mathcal{A}$ and a non-negligible ℓ, such that:

$$\Pr\left[\mathsf{Game}^{\mathsf{AP-IND-CQA}}_{\mathtt{PathORAM},\mathcal{A}} = 1\right] = \frac{1}{2} + \ell.$$

We will use $\mathcal{A}$ in a black-box way to construct a PPT algorithm able to break the IND-CPA security of $\mathcal{E}$, against the assumption. The idea is to build an algorithm $\mathcal{D}$ which simulates a `PathORAM` client $\mathcal{C}$, playing the AP-IND-CQA game against $\mathcal{A}$ (w.l.o.g., we assume that $\mathcal{A}$ itself simulates the server $\mathcal{S}$, otherwise $\mathcal{S}$ can be also simulated by $\mathcal{D}$). Throughout the game, $\mathcal{D}$ also stores a copy of the server's database $\mathcal{S}.\mathtt{DB}$, in plaintext. This is allowed, as

$\mathcal{S}.\mathtt{DB}$ is of size linear in n_{db}, and $\mathcal{D}$ is only simulating $\mathcal{C}$, so he is not limited by the storage constraints usually assumed in a normal ORAM client. Then $\mathcal{D}$ will use the interaction with $\mathcal{A}$ to win the IND-CPA game for scheme $\mathcal{E}$.

More in detail: first, $\mathcal{D}$ executes $\mathcal{A}$. Then $\mathcal{A}$ starts $\mathsf{Game}^{\mathsf{AP-IND-CQA}}_{\mathtt{PathORAM},\mathcal{A}}$ by choosing n and n_{db}, and $\mathcal{D}$ simulates a $\mathtt{PathORAM}$ client $\mathcal{C}$ created during Init, by initializing his own position map (populated with random values), but *without* generating a secret encryption key. Furthermore, $\mathcal{D}$ creates a tree memory structure of height n_{tree}, with leaves indexed $0, \dots, 2^{n_{\mathsf{tree}}}-1$, where every node stores n_{bkt} plaintexts of bit size n_{msg}, which are initialized to $(0^{n_{\mathsf{tag}}} \| 0^{n_{\mathsf{dat}}})$ (the parameters are the same as in Construction 3.63). This structure will be used by $\mathcal{D}$ to 'mirror' $\mathcal{S}.\mathtt{DB}$ in cleartext throughout the execution of $\mathtt{PathORAM}$.

$\mathcal{D}$ now starts $\mathsf{Game}^{\mathsf{IND-CPA}}_{\mathcal{E},\mathcal{D}}$, obtaining oracle access to Enc_k for an unknown secret key k, and choosing as security parameter the same n chosen by $\mathcal{A}$. At this point, notice that $\mathcal{D}$ is able to perfectly simulate a valid client $\mathcal{C}$ having access to the key k, in the following way:

- whenever $\mathcal{C}$ downloads a branch of $\mathcal{S}.\mathtt{DB}$ identified by leaf r by calling Access, $\mathcal{D}$ does the same (although the blocks in such downloaded branch will be ignored, as we will see);
- whenever $\mathcal{C}$ decrypts a certain block in a downloaded branch, $\mathcal{D}$ simulates the decryption oracle Dec_k by fetching the plaintext $(i \| \mathsf{data})$ found at the corresponding position in the 'mirrored' tree;
- whenever $\mathcal{C}$ swaps two blocks in a downloaded branch, $\mathcal{D}$ swaps the two plaintexts found at the corresponding positions in the 'mirrored' tree;
- whenever $\mathcal{C}$ encrypts a plaintext $(i \| \mathsf{data})$ to obtain a new encrypted block, $\mathcal{D}$ does so by using the encryption oracle Enc_k obtained from the IND-CPA game;
- whenever $\mathcal{C}$ updates his position map, or uploads an updated branch to $\mathcal{S}.\mathtt{DB}$, $\mathcal{D}$ does the same.

Given the above, it is clear that now whenever $\mathcal{A}$ asks for the execution of a data request dr, $\mathcal{D}$ is able to simulate the correct communication transcript com and a correctly formed updated branch $\mathtt{NewBranch}$. Therefore, for every data request performed during the first CQA phase, $\mathcal{A}$ always receives the correct access pattern.

Eventually, at the challenge step $\mathcal{A}$ produces two data requests $\mathsf{dr}^0, \mathsf{dr}^1$, and requests the execution of one of them. For $a \in \{0,1\}$, let $\mathsf{dr}^a = (\mathsf{op}^a, i^a, \mathsf{data}^a)$ be the two data requests and let $m^a \in \{0,1\}^{n_{\mathsf{msg}}}$ be formed as follows:

- if $\mathsf{op}^a =$ 'write', then set $m^a = (i^a \| \mathsf{data}^a)$;
- else, set $m^a = (i^a \| \mathsf{data}_{i^a})$, where data_{i^a} is retrieved by looking for identifier i^a in the mirrored tree.

Now, it could happen that $m^0 = m^1$. For example, it might be that the two data requests are of the form (‘write’, i, data) and (‘read’, i, data') respectively, but $\texttt{block}_i$ already encodes data. If this happens we say that the challenge query is *non-meaningful*. It is easy to see that two data requests from a non-meaningful challenge query will produce the same statistical distributions of communication transcripts[3] and updated paths, because their effect on the database is equivalent. Therefore, since $\mathcal{A}$ distinguishes the two resulting access patterns with non-negligible probability by assumption, it is clear that the challenge query must be *meaningful*, i.e., $m^0 \neq m^1$.

At this point $\mathcal{D}$ executes the challenge IND query using m^0, m^1 as plaintexts, and receiving back an encryption $c \leftarrow \mathsf{Enc}_k(m^b)$ for a secret bit b. $\mathcal{D}$ will also generate a random bit $b^* \xleftarrow{\$} \{0,1\}$ (a ‘guess’), and will answer $\mathcal{A}$’s challenge query by simulating the execution of dr^{b^*} as in the CQA phase, but injecting c as an updated block with identifier i^{b^*} during the execution of dr^{b^*}. Then $\mathcal{D}$ keeps simulating $\mathcal{C}$ during the second CQA phase as before, and waits until $\mathcal{A}$ outputs a bit $\hat{b}$. Finally: if $\hat{b} = b^*$, then $\mathcal{D}$ outputs b^* in the IND-CPA game, otherwise $\mathcal{D}$ outputs a new random bit.

Now, notice the following. In the case that $\mathcal{D}$’s guess was correct, i.e., $b = b^*$, it means that c was the right ciphertext at the right place, so that $\mathcal{A}$ has received a correctly formed access pattern. This means that $\mathcal{A}$ correctly guesses $\hat{b} = b^*$ with probability at least $\frac{1}{2} + \ell$, by assumption. In that case, also $\mathcal{D}$ wins, so:

$$\Pr\left[\mathsf{Game}^{\mathsf{IND-CPA}}_{\mathcal{E},\mathcal{D}} = 1 \middle| b = b^*\right] \geq \frac{1}{2} + \ell. \tag{3.1}$$

On the other hand, if $b \neq b^*$ we cannot say anything on $\mathcal{A}$’s success probability, because now $\mathcal{A}$ has a malformed access pattern. But we can say that, even if $\mathcal{A}$ fails, $\mathcal{D}$ still succeeds with probability $\frac{1}{2}$.

$$\Pr\left[\mathsf{Game}^{\mathsf{IND-CPA}}_{\mathcal{E},\mathcal{D}} = 1 \middle| b \neq b^*\right] \geq \frac{1}{2}. \tag{3.2}$$

Thus, combining 3.1 and 3.2, the reduction’s overall success probability is:

$$\Pr\left[\mathsf{Game}^{\mathsf{IND-CPA}}_{\mathcal{E},\mathcal{D}} = 1\right] \geq \frac{1}{2} + \frac{\ell}{2},$$

which concludes the proof. □

[3] Notice how this is not true anymore if the values in the position map are not totally random. Therefore, this step fails if the PRNG used is not secure.

Chapter 4

QS1: Post-Quantum Security

The next step in our analysis of quantum security notions is to consider what happens to classical encryption primitives when the adversaries have access to a quantum computing device. In this scenario, the cryptographic objects we are studying are still classical, as in the security class **QS**0. However, since many constructions in **QS**0 rely on computational hardness assumptions which do not hold anymore against quantum computers, new security models and constructions have to be considered in order to retain security in the new scenario. The branch of cryptography which aims at this goal has traditionally been called *post-quantum cryptography*. That is, post-quantum cryptography is about the security of *classical* primitives *after* (hence 'post-') quantum computing becomes available[1]. The security class which we denote by **QS**1 in our new labeling system covers this scenario.

But *how do we model post-quantum security exactly?* In the scientific community there has not always been mutual agreement on this. For example, one of the questions which most often cryptographers ask is: *"When should one consider classical access to a function for a quantum adversary, and when should one consider quantum access instead?"*. As we will see, the answer to this question is: *"Whenever the security model implies that the adversary computes the function on his local device, then quantum access should be used."* We call this principle *the* **QS***1 principle*.

In this chapter we will discuss in detail the **QS**1 principle and all the issues arising toward properly defining post-quantum security. Next, we introduce security models and definitions for post-quantum cryptographic primitives, starting from the very basic ones to more elaborated ones. We also discuss post-quantum assumptions, building blocks, and transformations from one primitive to another.

[1] Admittedly, this naming is a bit misleading, because it might be meant as *'cryptography resistant against the more advanced model of computation which will conceivably come after quantum computing'*. We do not want to argue here about the term 'post-quantum', which has become commonly accepted in the literature.

My Scientific Contribution in this Chapter

Theorem 4.10 is commonly considered folklore, but to the best of my knowledge the first fully formal proof (which I developed together with Gorjan Alagic and Bill Fefferman) appears in [ABF$^+$16].

All the material from Section 4.6 appeared first in [DFG13], which is a joint work with Özgür Dagdelen and Marc Fischlin. In that work, Özgür focused on the patching transformation for Σ-protocols using trapdoor commitments, and gave an explicit instantiation of such transformation applied to the lattice-based signature scheme by Lyubashevsky [Lyu12], which does not appear in this work. Marc focused on formalizing some necessary tools (Definition 4.33 and 4.34) and assessing the properties of our meta-reduction, while Theorem 4.36 is joint work of all of the authors. My contribution there is instead the positive result, i.e., Section 4.6. Moreover, the definition of Λ-protocol first appears in this work as a useful tool to bridge some formalization issues when defining the FS transform of oblivious-commitment Σ-protocols.

Regarding post-quantum ORAMs, all of Section 4.7 is my work. These results first appeared in [GKK17], where Nikolaos took care of the classical (**QS**0) ORAM scenario, while I developed the post-quantum (**QS**1) and fully quantum (**QS**3) scenarios.

Finally, to best of my knowledge, the classification of quantum security reductions appearing in Section 4.1 has never been made explicit before, and it appears in this work for the first time, although single examples of any of those kind of reductions have appeared in the literature before.

4.1 Issues in Post-Quantum Security

Post-quantum security constructions are usually obtained by replacing some underlying hardness assumption with a different, quantum-hard assumption, and then repeating the construction process (i.e., the security proof) leading to the realization of a secure primitive as in **QS**0. For example, when designing a post-quantum signature scheme, a natural option would be to consider a signature scheme in **QS**0 based on, e.g., the DLP problem, and see if it is possible to obtain a new scheme by replacing the DLP problem with some other quantum hardness assumption, e.g., *learning with errors (LWE)* or *shortest vector problem (SVP)*. Alternatively, one could simply try to design a signature scheme from scratch by relying on a new security proof reducing the security of the scheme to the quantum hardness of one of the aforementioned mathematical problems. Traditionally, schemes produced by such approaches are labeled 'post-quantum'. However, this labeling is sometimes inappropriate. The goal of this section is to give an overview of the many things that *could go wrong* when adopting too blindly the procedure described above, and to explain why one should take a more careful approach when defining meaningful notions of post-quantum security.

Proof Failures

The general issue when designing post-quantum primitives is that the classical security proofs might fail quantumly, even when only relying on quantum-hard assumptions. Common reasons for this are (but not limited to) the following.

- **No-Cloning:** when the security proof works by using the same value or element for two different purposes, care must be taken in making sure that this does not contradict the No-Cloning Theorem. If the element in question is a classical element, there is no problem. However, for quantum states, it is usually not possible to re-use the state for computing more than a single operation. Sometimes this can be solved by defining the operation in such a way that it does not destroy the input state.

- **Memory Snapshots:** as a consequence of the previous point, problems may arise when the security proof requires recording a ‘snapshot’ of an algorithm, or adversary, in order to execute it on different instances, or to analyze some internal area of memory. As the adversary is now a quantum machine, this cannot usually be done.

- **Rewinding:** analogously, proofs that use rewinding are notoriously hard to translate to the quantum setting. Limited positive results have been achieved in this respect in the existing literature [ARU14, Wat06].

- **Quantum Queries:** if the security proof requires ‘counting the number of queries’ to a certain oracle, it will probably fail when the oracle is replaced by a quantum oracle. The reason is that a quantum oracle can, in some sense, be queried over *all* the domain elements at once.

- **Lookup Tables:** analogously, if the proof requires storing a transcript of a protocol execution, including the query calls to some oracle, and if the oracle is quantum, problems may arise.

- **Measurements:** conditional procedures such as “if the value of x is y, then do...” are often an issue in the context of analyzing quantum states, because the information in the state is usually destroyed in the measurement process. This is particularly problematic when analyzing the values of queries to quantum oracles, or when comparing those values to those contained in some set.

Unfortunately, there is no general recipe to solve all of the above problems, and much of the existing literature erroneously advertises cryptographic constructions as ‘post-quantum’ just because they are based on quantum-hard problems, without addressing the previous issues. We strongly argue against the use of the term ‘post-quantum’ when describing the security of such constructions. Regardless, over the last few years many important tools have been developed in order to deal with these problems.

Quantum-Classical Oracles

The first important concept to define is what happens when an oracle $\mathcal{O}_f$ computing a *classical function* $f : \mathcal{X} \to \mathcal{Y}$ is invoked by a quantum algorithm. Two possible scenarios arise, depending on the *interaction*, or *access mode*, of the algorithm to the oracle:

1. the interaction is classical; in this case, the oracle is still a classical object which can be queried on classical inputs $x \in \mathcal{X}$ and returning outputs $y \in \mathcal{Y}$; or
2. the interaction is quantum; in this case the classical oracle $\mathcal{O}_f$ must be replaced by a *quantum-classical oracle* (which we denote by $|\mathcal{O}_f\rangle$).

A quantum-classical oracle can be queried on a *quantum superposition of classical input values*, usually of the form:

$$\sum_{x \in \mathcal{X}, y \in \mathcal{Y}} a_{x,y} \left| x, y \right\rangle, \text{ where } \sum_{x,y} |a_{x,y}|^2 = 1,$$

and it returns a quantum state encoding somehow the evaluation of f on the inputs in the superposition query. The exact form of the input and output states can vary, and it depends on the type of quantum access considered, as mentioned in Section 2.4. However, for most applications, and unless differently specified, we will denote by $|\mathcal{O}_f\rangle$ the unitary operator acting as follows.

Definition 4.1 (Canonical Quantum-Classical Oracle)**.** *Let $\mathcal{X}$,$\mathcal{Y}$ be sets, and $f : \mathcal{X} \to \mathcal{Y}$. The* (canonical) quantum-classical oracle for f*, denoted by $|\mathcal{O}_f\rangle$, is a unitary operator on $\mathfrak{H}_{\mathcal{X} \otimes \mathcal{Y}}$, defined by:*

$$|\mathcal{O}_f\rangle : |x, y\rangle \mapsto |x, y \oplus f(x)\rangle \, .$$

When not necessary to specify otherwise, in order to simplify notation we assume the ancilla register to be initialized with $|0\rangle$, so that:

$$|\mathcal{O}_f\rangle : \sum_{x \in \mathcal{X}} a_x \left| x, 0 \right\rangle \mapsto \sum_{x \in \mathcal{X}} a_x \left| x, f(x) \right\rangle, \text{ where } \sum_{x} |a_x|^2 = 1.$$

One important question regards quantum-classical oracles for randomized functions. For instance, if f is a randomized function, we can explicit the dependence from the randomness r (sampled from some appropriate distribution $\mathcal{R}$) by writing: $y := f(x; r)$. Then the question is: when considering $|\mathcal{O}_f\rangle$, should we consider superpositions of evaluations using the same, fixed randomness r, or should we consider evaluations where a fresh new randomness r is sampled for every element in the superposition? In other words, should we consider:

$$|\mathcal{O}_f\rangle : \sum_{x \in \mathcal{X}} a_x \left| x, 0 \right\rangle \mapsto \sum_{x \in \mathcal{X}} a_x \left| x, f(x; r) \right\rangle, \text{ where } r \leftarrow \mathcal{R},$$

or should we consider the following instead?

$$|\mathcal{O}_f\rangle : \sum_{x\in\mathcal{X}, r\leftarrow\mathcal{R}} a_{x,r} |x, 0\rangle \mapsto \sum_{x\in\mathcal{X}, r\leftarrow\mathcal{R}} a_{x,r} |x, f(x; r)\rangle .$$

As observed in [BZ13b], it turns out that the two cases are equivalent. The reason is that, using the first case, we can simulate the second case by first sampling a single r from $\mathcal{R}$, and then applying a quantum-secure PRF (described in Section 5.2) to generate independent pseudorandom values for every component of the superposition query. Because of the security properties of such PRF, the result would look the same to any QPT adversary.

Quantum Reductions

Another thing to discuss is the meaning of *quantum reductions*. As in the classical case, a quantum reduction $\mathcal{B}$ from (the security of) a scheme Σ to (the security of) a primitive, or (the hardness of) a problem Π, is an efficient algorithmic procedure which uses an hypothetical adversary $\mathcal{A}$ against Σ to attack Π. The existence of a reduction shows that: if an efficient adversary against Σ exists, then an efficient algorithm breaking Π's security must also exist. In this work we only consider black-box reductions, that is, reductions which do not have access to $\mathcal{A}$'s or Σ's internal code/circuit, but are only allowed to use the interactions between these components to attack Π.

Let us consider different possible scenarios in the quantum world. The following is a *classification* of possible (post-)quantum security reductions.

1. $\mathcal{A}$ is classical but $\mathcal{B}$ is quantum. In this case, $\mathcal{B}$ is a QPT algorithm using $\mathcal{A}$ as a (classical) subroutine. These kind of reductions offer the weakest form of security guarantees because they basically say: "if a *classical* adversary against Σ exists, then a *quantum* algorithm breaking Π's security exists". They do not say anything about the possibility that a quantum adversary against Σ might exist, so they are not really useful in our **QS**1 setting. We call these *weak quantum reductions*.

2. $\mathcal{A}$ is quantum and $\mathcal{B}$ is quantum. This is the most common scenario. These reductions say: "if a *quantum* adversary against Σ exists, then a *quantum* algorithm breaking Π's security exists". In particular, this rules out classical adversaries against Σ, but the existence of any of these adversaries would not necessarily imply a *classical* algorithm against Π, only a quantum one. We call these *(standard) quantum reductions*.

3. $\mathcal{A}$ is quantum but $\mathcal{B}$ is classical. These reductions offer the strongest security guarantees, because they say: "if a *quantum* adversary against Σ exists, then a *classical* algorithm breaking Π's security exists, with only black-box access to the adversary". Not only this rules out quantum and classical adversaries alike, but it also implies that the post-quantum

security of Σ can rely just on the post-quantum security of Π, so that in particular one does need to worry about oracle access modes. We call these *semi-classical reductions.*

Finally, it should be discussed what 'black-box' in the quantum setting means. Classically, this means that $\mathcal{B}$ is allowed to interact with $\mathcal{A}$ without accessing $\mathcal{A}$'s internal code or state. In other words, $\mathcal{B}$ can only act on $\mathcal{A}$'s inputs, outputs, and oracle queries. Furthermore, in cryptographic reductions, one usually has to make sure that $\mathcal{B}$'s action is computationally undetectable for $\mathcal{A}$, which means that the probability that $\mathcal{A}$'s output is affected by this action is negligible. This is important, for example, in the case that $\mathcal{B}$ injects or reads values inside $\mathcal{A}$'s queries to an oracle.

In the quantum setting, we adopt the same principle: $\mathcal{B}$ can tamper with $\mathcal{A}$'s inputs, outputs, and queries, as long as $\mathcal{A}$'s behaviour is only negligibly affected. So, for example, $\mathcal{B}$ could measure (fully or partially) $\mathcal{A}$'s queries to some quantum oracle, and even modify the queries and reprogram the oracle, as long as it can be proven that this action does not disturb $\mathcal{A}$'s working behaviour too much.

However, one could also take a stricter approach. Since measuring unknown quantum states might destroy the information therein, we could also consider quantum reductions that *do not measure external quantum states at all*, and only rely on the classical interactions with $\mathcal{A}$ (or other oracles) instead. For example, in the case of quantum oracle queries, such reductions would ignore those queries, and only interact classically with the (quantum) adversary. Clearly, these 'careful' reductions are quite powerful, because they work even when ignoring some potential source of information (the quantum queries). They basically say: "if a *quantum* adversary against Σ exists, then a *quantum* algorithm breaking Π's security exists, by using *only classical access* to some external quantum resources". These kind of reductions are placed somewhere between points 2 and 3 of the above hierarchy, and we call them *strong quantum reductions.*

4.2 The Quantum Random Oracle Model

One archetypical example of where the **QS**1 principle comes into play is the *Quantum Random Oracle Model (QROM).* Recall that, in **QS**0, the Random Oracle Model (ROM) is a computation model where all parties have access to an oracle $\mathcal{O}_{\mathcal{h}}$ computing a function $\mathcal{h}$ picked uniformly at random from the set of all functions from some domain $\mathcal{X}$ to some range $\mathcal{Y}$. This model is useful when analyzing the security of schemes employing PRFs or hash functions. In other words, the (truly) random function $\mathcal{h}$ is just an abstraction, or a model, for a real-world function g which we assume *behaving* like a random function.

But this also means that the random oracle $\mathcal{O}_{\mathcal{h}}$ itself is an abstract model for the computation of the real-world, algorithmic function g, *performed on* *some computer.* And since the code for g is public, and can be run by anyone (after all, in the ROM the access to $\mathcal{O}_{\hbar}$ is given to every participant in the scheme because of this reason), it is necessary to assume that a quantum adversary could implement the circuit computing g on his quantum computer, therefore being able to query g quantumly. Therefore, in the *Quantum Random Oracle Model (QROM)*, the random oracle $\mathcal{O}_{\hbar}$ must be replaced by a *quantum random oracle* $|\mathcal{O}_{\hbar}\rangle$. It is important to stress the fact that there exist models where security is proven in the random oracle model against quantum adversaries. We strongly argue against the use of the term 'post-quantum' when referring to those models.

So, in other words, in **QS**1 the ROM *must* be replaced by the QROM, where every QPT algorithm has access to a quantum oracle:

$$|\mathcal{O}_{\hbar}\rangle : |x, y\rangle \mapsto |x, y \oplus \hbar(x)\rangle .$$

and where $\hbar$ is chosen uniformly at random from the set of all functions from $\mathcal{X}$ to $\mathcal{Y}$, as in the random oracle model.

QROM Emulation

Notice the following difficulty when defining the QROM operationally. Classically, as explained in Section 2.3, during a cryptographic reduction a random oracle is *emulated* by a PPT algorithm, for example through lazy sampling. But lazy sampling cannot work for quantum random oracles, for two reasons.

First of all, a single quantum query to $|\mathcal{O}_{\hbar}\rangle$ could require the emulator to lazy-sample too many elements. E.g., a query of the form:

$$\sum_{x \in \{0,1\}^n} \frac{1}{\sqrt{2^n}} |x, 0\rangle$$

would query all the exponentially-many input values at once, and so it would 'force' the emulator to 'fix' all those values at the same time. This is not compatible with what we require from an efficient cryptographic reduction.

The second problem is that the concept of lookup table, used in the classical ROM to answer consistently with the previous queries, becomes meaningless. Firstly because such table could quickly reach exponential size, as the previous query example shows; and secondly because, as discussed in Section 4.1, there might be no way to check whether the values of some query are in the table or not without destroying the query.

Luckily, there exist a few other techniques to solve the above issues and to make the QROM a meaningful tool in **QS**1. If the number of queries performed by the adversary to the QRO is known a priori, then the QRO can be efficiently emulated by *d-wise independent functions.* These are families of functions that are statistically indistinguishable from random functions if queried (classically) no more than d times. An example are polynomial functions of degree $d-1$. It is known [Zha12b] that no quantum algorithm performing at most q queries can distinguish between random oracles and distributions of $2q$-wise independent functions.

Another common technique is to emulate a RO with a PRF, which is useful if one does not know a priori an upper bound to the number of adversarial queries. In the QROM we need something analogous, but classical PRFs alone cannot work. One idea might be to use *post-quantum PRFs* (we will define them in the next section), but actually for emulating a QRO, classical access to the PRF is not enough, so we need something more: *quantum-secure (superposition-secure) PRFs* will be defined in the next chapter.

QROM Reprogramming

It is important to analyze what happens when reprogramming a quantun random oracle $|\mathcal{O}_{\hbar}\rangle$. In particular, a useful technique often consists in *injecting* some fixed value y for a subset $\mathcal{S} \subset \mathcal{X}$ of possible input query values, so that $\hbar(x) := y$ for all $x \in \mathcal{S}$. Intuitively, if the set S is 'very small', it is going to be very hard for a quantum algorithm to distinguish the modified oracle from a true QRO. However, some proofs might rely explicitly on the probability of the adversary querying one of those values, so it is important to have a detailed quantitative analysis for these probabilities.

We start by recalling [Zha12a] a tool known as *semi-constant distributions.*

Definition 4.2 (Semi-Constant Distributions)**.** *Let $\mathcal{H} := \{\hbar : \mathcal{X} \to \mathcal{Y}\}$ be the family of functions between sets $\mathcal{X}$ and $\mathcal{Y}$, and let $\delta \in [0,1]$. We define the* δ-fraction semi-constant distribution $\mathcal{U}^\delta$ *as the distribution over $\mathcal{H}$ resulting from the following procedure:*

```
sample $y \xleftarrow{\$} \mathcal{Y}$
for all $x \in \mathcal{X}$ do
    $p \xleftarrow{\$} [0,1]$
    if $p \le \delta$ then
        define $\hbar(x) := y$
    else
        sample $y' \xleftarrow{\$} \mathcal{Y}$
        define $\hbar(x) := y'$
Return: $\hbar$
```

Notice that $\mathcal{U}^0$ is the uniform distribution, while $\mathcal{U}^1$ is a constant distribution. Also note that the distribution, when used within an oracle, is consistent in the sense that the settings are chosen once at the outset. We will use this definition to describe a QRO which has been 'reprogrammed' on a fraction δ of its possible inputs. The following lemma [Zha12b] gives an upper bound on the probability that a quantum algorithm's behavior changes when switching from a truly QRO to a quantum oracle for a function drawn from $\mathcal{U}^\delta$ in terms of statistical distance.

Lemma 4.3 ([Zha12b, Corollary 4.3])**.** *Let $\mathcal{A}^{|\mathcal{O}_\hbar\rangle}$ be a* QPT *algorithm making at most q_h queries to the quantum random oracle $|\mathcal{O}_\hbar\rangle$. Let $\delta \in (0,1)$ and let $|\mathcal{O}_\hbar^\delta\rangle$ be the classical-quantum oracle obtained by reprogramming $\mathcal{O}_\hbar$ on a fraction δ of its possible inputs, i.e., $|\mathcal{O}_\hbar^\delta\rangle$ is described by the semi-constant distribution $\mathcal{U}^\delta$. Then, the following holds:*

$$\left| \mathcal{A}^{|\mathcal{O}_\hbar\rangle} - \mathcal{A}^{|\mathcal{O}_\hbar^\delta\rangle} \right| \leq \frac{8}{3} \cdot q_h^4 \cdot \delta^2.$$

The above lemma is quite general, because it does not take into account the specific values where the reprogramming happens, but just a generic fraction δ of all possible values. Therefore, it is especially useful in those cases where the quantum random oracle is reprogrammed randomly, i.e., by just replacing some of its values with a certain probability δ. However, in all those cases where it is possible to track the specific amplitudes (across the oracle queries) of the elements to be reprogrammed, then one can usually find better bounds, for example by using Lemma 2.11.

4.3 Post-Quantum Assumptions, Building Blocks

In this section we redefine the basic assumptions and building blocks for the post-quantum setting.

Post-Quantum OWFs

As in the **QS**0 case, the existence of *post-quantum one-way functions (pqOWF)* is a basic security assumptions. Because a OWF's code is public, and recalling the **QS**1 principle, we expect quantum adversaries to be able to query a OWF on a superposition of values. However, for the same reason, since in the definition of OWF the quantifier is *'for all'* PPT algorithms, without mentioning oracle access, it is enough to define post-quantum OWFs by just replacing PPT adversaries with QPT adversaries.

Definition 4.4 (Post-Quantum One-Way Functions (pqOWF) and Permutations (pqOWP))**.** *Let $\mathcal{F} = \left(\mathcal{F}_n\right)_n$ be a* DPT *algorithm, with $\mathcal{F}_n : \mathcal{X}_n \to \{0,1\}^*$. $\mathcal{F}$ is a (family of)* post-quantum one-way functions (pqOWF) *iff for any* QPT *algorithm $\mathcal{A}$ it holds:*

$$\Pr_{x \xleftarrow{\$} \mathcal{X}} \left[\mathcal{A}(\mathcal{F}(x)) \to x' : \mathcal{F}(x) = \mathcal{F}(x') \right] \leq \mathsf{negl}.$$

Moreover, in the special case where $\mathcal{F}_n : \mathcal{X}_n \to \mathcal{X}_n$ are permutations on $\mathcal{X}_n$ for every n, $\mathcal{F}$ is a (family of) post-quantum one-way permutations (OWP)*.*

The definition of *post-quantum hard-core predicates* is as in the **QS**0 case.

Definition 4.5 (Post-Quantum Hard-Core Predicate)**.** *Let $\mathcal{F} : \mathcal{X} \to \mathcal{Y}$ be a OWF. A polynomial-time computable function $\mathsf{hc}_{\mathcal{F}} : \mathcal{X} \to \{0, 1\}$ is a* post-quantum hard-core predicate of $\mathcal{F}$ *iff, for any* QPT *algorithm $\mathcal{A}$ it holds:*

$$\Pr_{x \xleftarrow{\$} \mathcal{X}} \left[\mathcal{A}(\mathcal{F}(x)) \to \mathsf{hc}_{\mathcal{F}}(x) \right] \leq \frac{1}{2} + \mathsf{negl}.$$

Proposition 4.6. *Let $\mathcal{F}$ be a pqOWF (resp., pqOWP). Then it is possible to efficiently transform $\mathcal{F}$ into a pqOWF (resp., pqOWP) $\mathcal{H}$ such that at least one post-quantum hard-core predicate $\mathsf{hc}_{\mathcal{H}}$ exists.*

Given the above, from now on for simplicity we assume that every pqOWF admits post-quantum hard-core predicates. In the case that $\mathcal{F} : \mathcal{X} \to \mathcal{X}$ (in particular, if $\mathcal{F}$ is a pqOWP), the construction of hard-core bits can be iterated as in Proposition 2.5.

Post-Quantum OWTPs

The same discussion in the case of post-quantum OWFs applies for the assumption of the existence of *post-quantum one-way trapdoor permutations (pqOWTP)*. As usual, we express a family of pqOWTPs as indexed through efficiently sampleable index family $\mathcal{I}$ and associated trapdoor space $\mathcal{T}$.

Definition 4.7 (Post-Quantum One-Way Trapdoor Permutation (pqOWTP))**.** *A (family of)* post-quantum one-way trapdoor permutations (pqOWTP) *is a tuple* $(\mathsf{Gen}, \mathsf{Eval}, \mathsf{Invert})$ *of* PPT *algorithms:*

1. $\mathsf{Gen} : \to \mathcal{I} \times \mathcal{T}$;
2. $\mathsf{Eval} : \mathcal{I} \times \mathcal{X} \to \mathcal{X}$;
3. $\mathsf{Invert} : \mathcal{I} \times \mathcal{T} \times \mathcal{X} \to \mathcal{X} \cup \{\perp\}$,

and such that:

1. *for any* QPT *algorithm $\mathcal{A}$ it holds:*
$$\Pr_{\substack{x \xleftarrow{\$} \mathcal{X} \\ (i,t) \leftarrow \mathsf{Gen}}} \left[\mathcal{A}(i, \mathsf{Eval}(i, x)) \to x \right] \leq \mathsf{negl}; \text{ and}$$
2. $\mathsf{Invert}(i, t, y) = \mathsf{Eval}(i, x), \forall x \in \mathcal{X}, \forall\, (i, t) \leftarrow \mathsf{Gen}, \forall\, y \leftarrow \mathsf{Eval}(i, x)$.

As in the **QS**0 case, the existence of pqOWTP implies the existence of pqOWP and pqOWF.

Proposition 4.8 (pqOWTP $\implies$ pqOWP $\implies$ pqOWF)**.** *Let $\mathcal{P} := (\mathsf{Gen}, \mathsf{Eval}, \mathsf{Invert})$ be a pqOWTP on $\mathcal{X}$. Then, for all but a negligible fraction of possible sequences $((i_n, t_n))_n$ of outputs of* $\mathsf{Gen}(n) \implies \mathsf{Eval}(i_n, .)$ *is a pqOWP (and hence a pqOWF) on $\mathcal{X}$.*

Post-Quantum PRNGs

Again, the same principle from OWF and OWTP applies when translating PRNGs to the post-quantum setting. Remember that the security property for PRNGs does not mention any kind of oracle access or code emulation, but it just says that no efficient adversaries, by looking at the stream of (classical) values output by the PRNG, can distinguish such stream from a random stream. So, the interaction is still classical, and the only change is that the adversary is now a quantum algorithm.

Definition 4.9 (Post-Quantum PRNG (pqPRNG))**.** *Let p be a polynomial such that $p(n) \geq n+1, \forall n \in \mathbb{N}$. A* post-quantum pseudorandom number generator (pqPRNG) *with expansion factor p is a* DPT *algorithm $\mathcal{G}$ such that:*

1. *given as input a bit string $s \in \{0,1\}^n$, (the* seed*), outputs a bit string $\mathcal{G}(s) \in p(n)$; and*
2. *for any* QPT *algorithm $\mathcal{D}$:*

$$\left|\Pr\left[\mathcal{D}(r) \to 1\right] - \Pr\left[\mathcal{D}(\mathcal{G}(s)) \to 1\right]\right| \leq \mathsf{negl},$$

where $r \overset{\$}{\leftarrow} \{0,1\}^{p(n)}, s \overset{\$}{\leftarrow} \{0,1\}^n$, and the probabilities are taken over the choice of r and s, and the randomness of $\mathcal{D}$.

Moreover, as noticed in Section 3.1, the proof of Theorem 3.3 still goes through in the post-quantum scenario, because it does not make any assumption on the query capabilities of the adversary.

Theorem 4.10 ([ABF+16, Lemma 19])**.** *If $\mathcal{F}$ is a pqOWF, then $\mathcal{G}_\mathcal{F}$ (defined as in Construction 3.2) is a pqPRNG.*

Corollary 4.11 (pqOWF ⇔ pqPRNG)**.** *pqOWFs exist iff pqPRNGs exist.*

Clearly, a pqPRNG it is also a PRNG. However, the opposite is not believed to hold, as the following example shows.

Lemma 4.12. *Under the DLP hardness assumption, there exists a PRNG $\mathcal{G}_{BM}$ which is* quantumly predictable*. I.e., there exists a non-negligible function δ and a* QPT *algorithm $\mathcal{D}$ which, on input n sequential values output by $\mathcal{G}_{BM}$ on any random seed, predicts the $(n+1)$-th value output by $\mathcal{G}_{BM}$ with probability at least $\delta(n)$.*

Proof. A counterexample $\mathcal{G}_{BM}$ is the modular exponentiation Blum-Micali generator [KL07], but many other similar variants work as well [GdAJ13]. This construction is based on exponentiation of a public generator g modulo a public large prime p, and it is a classically secure PRNG under the assumption that computing discrete logarithms is computationally hard. More specifically, if s_i is the current state of the generator, one output bit is computed as a hardcore predicate of the value $s_{i+1} = g^{s_i} \mod p$ (where s_{i+1} becomes the next state of the generator). Thus, starting from a secret seed s_0, a pseudorandom bit string can be generated by applying iteratively the procedure.

However, there exists a quantum attack [GdAJ13] (based on variants of both Shor's and Grover's algorithms) which, given p, g and a sequence $(r_1, \ldots, r_n)$ of values output by $\mathcal{G}_{BM}$, can recover the initial state s_0 with probability δ non-negligible in n. This, in turns, allows to predict the whole sequence of $\mathcal{G}_{BM}$. □

Post-Quantum PRFs

The case of pseudorandom functions, instead, is a bit different. Definition 3.5 specifically conditions the existence of (classical) PRFs to the query capabilities of the adversary, so we should make a distinction whether, in the post-quantum case, these queries should still be classical or not.

The **QS**1 principle comes handy here. In a reasonable security model, should the adversary be able to implement the code of the PRF on his local computing device? The answer is: *"normally, no, because he does not know the secret key"*. After all, the whole point of a PRF is that the adversary should not be able to distinguish the output of the PRF from the output of an (abstractly defined) completely random function, which in particular means that the adversary should not be able to see the PRF's code, because there might be *no code at all.* This is in striking contrast with the QROM, and the reason is that a QRO models a *public hash function*, which everybody can compute, while a PRF exists *as long as the key remains secret.*

In other words, *post-quantum pseudorandom functions (pqPRFs)* are defined by merely replacing the PPT adversary with a QPT adversary, and keeping the oracle access classical. *Quantum-secure PRFs* instead, as defined in [BDF$^+$11, Zha12a], are a different object, and they will be presented in the next chapter in the context of the domain **QS**2.

Definition 4.13 (Post-Quantum Pseudorandom Function (pqPRF))**.** *A (family of)* post-quantum pseudorandom functions (pqPRF) *from $\mathcal{X}$ to $\mathcal{Y}$ with key space $\mathcal{K}$ is a* DPT *algorithm $\mathcal{F} : (k \in \mathcal{K}, x \in \mathcal{X}) \mapsto y \in \mathcal{Y}$ such that for any* QPT *algorithm $\mathcal{D}$ it holds:*

$$\left| \Pr_{k \xleftarrow{\$} \mathcal{K}} \left[\mathcal{D}^{\mathcal{F}_k} \to 1 \right] - \Pr_{\hbar \xleftarrow{\$} \mathcal{Y}^{\mathcal{X}}} \left[\mathcal{D}^{\mathcal{O}_\hbar} \to 1 \right] \right| \leq \mathsf{negl},$$

where $\mathcal{O}_\hbar$ is an oracle for $\hbar$ (i.e., a random oracle), and the probabilities are over the choice of k and $\hbar$, and the randomness of $\mathcal{D}$.

Moreover, the same proofs for Theorems 3.6 and 3.7 go through unchanged, because we are not modifying the oracle access mode, but just the adversary computation model. As a consequence, we can state the following.

Theorem 4.14 (pqPRF ⇔ pqPRNG)**.** *pqPRFs exist iff pqPRNGs exist.*

Corollary 4.15. *pqOWF exist iff pqPRF exist.*

Post-Quantum PRPs

The case of post-quantum PRPs is analogous to the one for pqPRFs.

Definition 4.16 (Post-Quantum Weak PRP (pqWPRP))**.** *A (family of)* post-quantum weak pseudorandom permutations (pqWPRP) *on $\mathcal{X}$ with key space $\mathcal{K}$ is a pair of* DPT *algorithms* $(\mathcal{P}, \mathcal{P}^{-1}) : (k \in \mathcal{K}, x \in \mathcal{X}) \mapsto x' \in \mathcal{X}$ *such that:*

1. $\forall k \in \mathcal{K} \implies \mathcal{P}_k, \mathcal{P}_k^{-1}$ *are permutations on $\mathcal{X}$;*
2. $\forall k \in \mathcal{K} \implies (\mathcal{P}_k)^{-1} = \mathcal{P}_k^{-1}$*; and*
3. *for any* QPT *algorithm $\mathcal{D}$ it holds:*

$$\left| \Pr_{k \xleftarrow{\$} \mathcal{K}} \left[\mathcal{D}^{\mathcal{P}_k} \to 1 \right] - \Pr_{p \xleftarrow{\$} S(\mathcal{X})} \left[\mathcal{D}^{\mathcal{O}_p} \to 1 \right] \right| \leq \mathsf{negl},$$

where $\mathcal{O}_p$ is an oracle for p, and the probabilities are over the choice of k and p, and the randomness of $\mathcal{D}$.

Definition 4.17 (Post-Quantum Strong PRP (pqSPRP))**.** *A (family of)* post-quantum strong pseudorandom permutations (pqSPRP) *on $\mathcal{X}$ with key space $\mathcal{K}$ is a pair of* DPT *algorithms* $(\mathcal{P}, \mathcal{P}^{-1}) : (k \in \mathcal{K}, x \in \mathcal{X}) \mapsto x' \in \mathcal{X}$ *such that:*

1. $\forall k \in \mathcal{K} \implies \mathcal{P}_k, \mathcal{P}_k^{-1}$ *are permutations on $\mathcal{X}$;*
2. $\forall k \in \mathcal{K} \implies (\mathcal{P}_k)^{-1} = \mathcal{P}_k^{-1}$*; and*
3. *for any* QPT *algorithm $\mathcal{D}$ it holds:*

$$\left| \Pr_{k \xleftarrow{\$} \mathcal{K}} \left[\mathcal{D}^{\mathcal{P}_k, \mathcal{P}_k^{-1}} \to 1 \right] - \Pr_{p \xleftarrow{\$} S(\mathcal{X})} \left[\mathcal{D}^{\mathcal{O}_p, \mathcal{O}_{p^{-1}}} \to 1 \right] \right| \leq \mathsf{negl},$$

where $\mathcal{O}_p$ is an oracle for p, $\mathcal{O}_{p^{-1}}$ is an oracle for p^{-1}, and the probabilities are over the choice of k and p, and the randomness of $\mathcal{D}$.

When left unspecified, by 'pqPRP' we mean the strong version. A pqPRP is clearly also a pqPRF, but the converse does not necessarily hold. Again, as we are not modifying the oracle access mode, the classical constructions of PRPs from PRFs go through unchanged in the post-quantum setting. Therefore, the existence of pqPRPs is also equivalent to the existence of pqOWFs.

Theorem 4.18 (pqPRF ⇔ pqPRP)**.** *pqPRFs exist iff pqPRPs exist.*

4.4 Post-Quantum Encryption

Post-quantum encryption schemes are classical encryption schemes meant to retain their security also against quantum adversaries. It is common for this scenario to just assume the same definitions and security notions we saw in Chapter 3, and just replacing PPT adversaries with QPT ones. However, in the case of public-key encryption, one must be a bit careful in doing so.

Post-Quantum Secret-Key Encryption

Following the **QS**1 principle, in *post-quantum secret-key encryption* one can just 'blindly' replace classical adversaries with quantum ones, because the adversary itself is never supposed to run encryption or decryption procedures locally (after all, he does not have the secret key). So we discuss here the modified security definitions as follows (we do it just for the IND and IND-CPA notions, but the same procedures yields equivalent post-quantum security notions for SEM, IND-CCA1, and IND-CCA2). As usual, $\mathcal{E} := \mathcal{E}_{\mathcal{K},\mathcal{X},\mathcal{Y}} := (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ denotes a SKES with plaintext space $\mathcal{X}$, ciphertext space $\mathcal{Y}$, and key space $\mathcal{K}$.

Definition 4.19 (Post-Quantum IND Adversary)**.** *Let $\mathcal{E}$ be a SKES. A* post-quantum IND (pq-IND) adversary $\mathcal{A}$ for $\mathcal{E}$ *is a pair of* QPT *algorithms* $\mathcal{A} := (\mathcal{M}, \mathcal{D})$*, where:*

1. $\mathcal{M} :\to \mathcal{X} \times \mathcal{X} \times \mathfrak{H}$ *is the* pq-IND message generator*;*

2. $\mathcal{D} : \mathcal{Y} \times \mathfrak{H} \to \{0,1\}$ *is the* pq-IND distinguisher*,*

where $\mathfrak{H}$ is a Hilbert space of appropriate dimension, modeling the state communication register between $\mathcal{M}$ and $\mathcal{D}$.

Experiment 4.20 ($\mathsf{Game}^{\mathsf{pq-IND}}_{\mathcal{E},\mathcal{A}}$)**.** *Let $\mathcal{E}$ be a SKES, and $\mathcal{A} := (\mathcal{M}, \mathcal{D})$ a pq-IND adversary. The* pq-IND experiment *proceeds as follows:*

```
Input: $n \in \mathbb{N}$
$k \leftarrow \mathsf{KGen}$
$(m^0, m^1, |\mathsf{state}\rangle) \leftarrow \mathcal{M}$
$b \overset{\$}{\leftarrow} \{0,1\}$
$c \leftarrow \mathsf{Enc}_k(m^b)$
$b' \leftarrow \mathcal{D}(c, |\mathsf{state}\rangle)$
if $b = b'$ then
    Output: 1
else
    Output: 0
```

The advantage of $\mathcal{A}$ *is defined as:*

$$\mathsf{Adv}^{\mathsf{pq-IND}}_{\mathcal{E},\mathcal{A}} := \Pr\left[\mathsf{Game}^{\mathsf{pq-IND}}_{\mathcal{E},\mathcal{A}} \to 1\right] - \frac{1}{2}.$$

Definition 4.21 (Post-Quantum Indistinguishability (pq-IND))**.** *A SKES $\mathcal{E}$ has* post-quantum indistinguishable encryptions (or, it is pq-IND secure) *iff, for any pq-IND adversary $\mathcal{A}$ it holds that:* $\mathsf{Adv}_{\mathcal{E},\mathcal{A}}^{\mathsf{pq-IND}} \leq \mathsf{negl}$.

Experiment 4.22 ($\mathsf{Game}_{\mathcal{E},\mathcal{A}}^{\mathsf{pq-IND-CPA}}$)**.** *Let $\mathcal{E}$ be a SKES, and $\mathcal{A} := (\mathcal{M}, \mathcal{D})$ a pq-IND adversary. The* pq-IND-CPA experiment *proceeds as follows:*

```
1: Input: $n \in \mathbb{N}$
2: $k \leftarrow \mathsf{KGen}$
3: $(m^0, m^1, |\mathsf{state}\rangle) \leftarrow \mathcal{M}^{\mathsf{Enc}_k}$
4: $b \overset{\$}{\leftarrow} \{0,1\}$
5: $c \leftarrow \mathsf{Enc}_k(m^b)$
6: $b' \leftarrow \mathcal{D}^{\mathsf{Enc}_k}(c, |\mathsf{state}\rangle)$
7: if $b = b'$ then
8:     Output: 1
9: else
10:     Output: 0
```

The advantage of $\mathcal{A}$ *is defined as:*

$$\mathsf{Adv}_{\mathcal{E},\mathcal{A}}^{\mathsf{pq-IND-CPA}} := \Pr\left[\mathsf{Game}_{\mathcal{E},\mathcal{A}}^{\mathsf{pq-IND-CPA}} \rightarrow 1\right] - \frac{1}{2}.$$

Definition 4.23 (Post-Quantum Indistinguishability of Ciphertexts under Chosen Plaintext Attack (pq-IND-CPA))**.** *A SKES $\mathcal{E}$ has* post-quantum indistinguishable encryptions under chosen plaintext attack (or, it is pq-IND-CPA secure) *iff, for any pq-IND adversary $\mathcal{A}$ it holds that:* $\mathsf{Adv}_{\mathcal{E},\mathcal{A}}^{\mathsf{pq-IND-CPA}} \leq \mathsf{negl}$.

Clearly, pq-IND-CPA is at least as strong as IND-CPA.

Theorem 4.24 (pq-IND-CPA $\Longrightarrow$ IND-CPA)**.** *If a SKES is pq-IND-CPA secure, then it is also IND-CPA secure.*

It is common folklore that, unlike some PKES, the most widely used constructions for SKES are actually also post-quantum secure. However, the converse of Theorem 4.24 does not hold, and it is important to remember that post-quantum notions for SKES are actually *strictly stronger* than the classical ones in **QS**0.

Theorem 4.25 (IND-CPA SKES $\not\Longrightarrow$ pq-IND-CPA SKES)**.** *Under standard hardness assumptions, there exist SKES which are IND-CPA secure, but not pq-IND-CPA secure.*

Proof (sketch). It is sufficient to consider an IND-CPA SKES which appends to every ciphertext the secret key used, encrypted with another, IND-CPA but non–post-quantum secure PKES (e.g., some RSA variant) under a fixed, known public key. With the knowledge of the public key, a quantum adversary can emulate a quantum oracle for the encryption of the PKES, which can then be broken by, e.g., Shor's algorithm, thus revealing the SKES's secret key. □

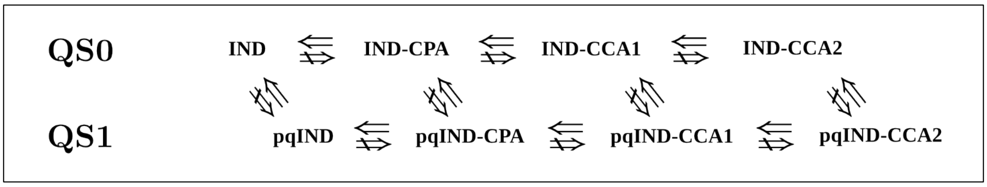


Figure 4.1: Relations for SKES security notions in **QS**0 and **QS**1.

The Goldreich scheme from Construction 3.26 is pq-IND-CPA when instantiated with a pqPRF, because the same arguments used in Theorem 3.27 go through as long as the adversary is unable to distinguish the PRF from a real source of randomness.

Theorem 4.26. *Let $\mathcal{E}_\mathcal{F}$ be the SKES from Construction 3.26 implemented through a pqPRF $\mathcal{F}$. Then $\mathcal{E}_\mathcal{F}$ in a pq-IND-CPA SKES.*

The same relations and separations examples between pq-IND, pq-IND-CPA, pq-IND-CCA1, and pq-IND-CCA2, hold as from Section 3.2, and with analogous separation examples from their classical counterparts as in Theorem 4.25. Therefore, the relations between security notions for SKES in **QS**0 and **QS**1 are as summarized in Figure 4.1.

Post-Quantum Public-Key Encryption

In *post-quantum public-key encryption schemes* the situation is quite different. The reason is that, in this case, the presence of a public-key allows the adversary to compute encryptions autonomously. In this scenario, following the **QS**1 principle, the encryption oracle Enc_pk should be replaced by the quantum counterpart $|\mathsf{Enc}_\mathsf{pk}\rangle$. However, this is only true for the *learning phases* during the security game (recall that, for PKES, IND security alone does not constitute a meaningful notion). The IND phase, on the other hand, models the attack of the adversary against the encryption of some unknown message, encryption that, therefore, is performed by some *classical* third party (the *IND challenger*). Moreover, as $\mathcal{M}$ and $\mathcal{D}$ are QPT algorithms, giving them the public key pk as input automatically implies access to $|\mathsf{Enc}_\mathsf{pk}\rangle$.

The resulting post-quantum IND-CPA security game is modified as follows.

Experiment 4.27 ($\mathsf{Game}^{\mathsf{pq-IND-CPA}}_{\mathcal{E},\mathcal{A}}$ for PKES)**.** *Let $\mathcal{E}$ be a PKES, and $\mathcal{A} := (\mathcal{M}, \mathcal{D})$ a pq-IND adversary. The* pq-IND-CPA experiment *(in the post-quantum public-key setting) proceeds as follows:*

```
Input: $n \in \mathbb{N}$
$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}$
$(m^0, m^1, |\mathsf{state}\rangle) \leftarrow \mathcal{M}(\mathsf{pk})$
$b \xleftarrow{\$} \{0, 1\}$
```

```
5: $c \leftarrow \mathsf{Enc}_{\mathsf{pk}}(m^b)$
6: $b' \leftarrow \mathcal{D}(c, |\mathsf{state}\rangle, \mathsf{pk})$
7: if $b = b'$ then
8:     Output: 1
9: else
10:    Output: 0
```

The advantage of $\mathcal{A}$ *is defined as:*

$$\mathsf{Adv}^{\mathsf{pq-IND-CPA}}_{\mathcal{E},\mathcal{A}} := \Pr\left[\mathsf{Game}^{\mathsf{pq-IND-CPA}}_{\mathcal{E},\mathcal{A}} \to 1\right] - \frac{1}{2}.$$

Notice how only the encryption oracle during the learning phases is replaced by a quantum oracle, but it is still classical during the IND phase. This notion was introduced in [BZ13b], but we will discuss more the implications of this important difference in Section 5.3. Also notice how $\mathsf{Game}^{\mathsf{pq-IND-CPA}}_{\mathcal{E},\mathcal{A}} = \mathsf{Game}^{\mathsf{pq-IND}}_{\mathcal{E},\mathcal{A}^{|\mathsf{Enc}_{\mathsf{pk}}\rangle}}$ only holds for the public-key setting.

The security notions pq-IND-CCA1 and pq-IND-CCA2 in the public-key setting are a straightforward modification of the ones for the SKES case, by giving to the adversary quantum oracle access to $|\mathsf{Enc}_{\mathsf{pk}}\rangle$ – but the oracle $\mathsf{Dec}_{\mathsf{sk}}$ remains classical. It is well-known that certain PKES which are IND-CPA secure under standard assumptions are *not* pq-IND-CPA secure (examples are RSA, ElGamal EC-based schemes, etc.) Instead, pq-IND-CPA (or stronger) PKESs can be constructed under other quantum-hardness assumptions, as discussed in Section 2.3.

4.5 Post-Quantum Signatures

In the case of *post-quantum signature schemes*, as the oracle access to $\mathsf{Sign}_{\mathsf{sk}}$ is kept classical according to the **QS**1 principle, the definition of existential unforgeability is modified in the standard post-quantum way, e.g., by merely replacing PPT adversaries with QPT ones.

Experiment 4.28 ($\mathsf{Game}^{\mathsf{pq-EUF-CMA}}_{\mathcal{S}ig,\mathcal{A}}$)**.** *Let* $\mathcal{S}ig$ *be a DSS, and* $\mathcal{A}$ *a* QPT *algorithm. The* pq-EUF-CMA experiment *proceeds as follows:*

```
1: Input: $n, q_s \in \mathbb{N}$
2: $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}$
3: $(x, \mathsf{sig}) \leftarrow \mathcal{A}^{\mathsf{Sign}_{\mathsf{sk}}}(\mathsf{pk})$ after making at most $q_s$ queries to $\mathsf{Sign}_{\mathsf{sk}}$, receiving
   signatures $(x_1, \mathsf{sig}_1), \ldots (x_{q_s}, \mathsf{sig}_{q_s})$
4: if $\mathsf{SigVerify}(\mathsf{pk}, x, \mathsf{sig}) = 1$ and $x \neq x_i \; \forall \, i = 1, \ldots, q_s$ then
5:     Output: 1
6: else
7:     Output: 0
```

The advantage of $\mathcal{A}$ *is defined as:*

$$\mathsf{Adv}^{\mathsf{pq-EUF-CMA}}_{\mathcal{S}ig,\mathcal{A}}(n, q_s) := \Pr\left[\mathsf{Game}^{\mathsf{pq-EUF-CMA}}_{\mathcal{S}ig,\mathcal{A}}(n, q_s) \to 1\right].$$

Definition 4.29 (Post-Quantum Existential Unforgeability under Chosen Message Attack (pq-EUF-CMA))**.** *A DSS $\mathcal{S}ig$ is* post-quantum existentially unforgeable under chosen message attack (or, it is pq-EUF-CMA secure) *iff, for any* QPT *algorithm $\mathcal{A}$ it holds that:*

$$\mathsf{Adv}^{\mathsf{pq-EUF-CMA}}_{\mathcal{S}ig,\mathcal{A}} \leq \mathsf{negl}.$$

However, the situation changes in the case of signatures in the random oracle model: in this case, it would not make sense to define a notion of post-quantum security without switching to the quantum random oracle model. The resulting security notion should be called, for consistency with our naming conventions, pq-EUF-CMA-QRO. However, it is clear that the presence of QRO automatically implies QPT adversaries, which in turn implies a post-quantum security notion *at least*. Therefore, for simplicity, we will call this new security notion just EUF-CMA-QRO.

Experiment 4.30 ($\mathsf{Game}^{\mathsf{EUF-CMA-QRO}}_{\mathcal{S}ig,\mathcal{A}}$)**.** *Let $\mathcal{S}ig$ be a DSS, $\mathcal{O}_{\hbar}$ a random oracle with corresponding quantum random oracle $|\mathcal{O}_{\hbar}\rangle$, and $\mathcal{A}$ a* QPT *algorithm. The* EUF-CMA-QRO experiment *proceeds as follows:*

```
1: Input: $n, q_s, q_h \in \mathbb{N}$
2: $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}^{\mathcal{O}_{\hbar}}$
3: $(x, \mathsf{sig}) \leftarrow \mathcal{A}^{\mathsf{Sign}_{\mathsf{sk}}, |\mathcal{O}_{\hbar}\rangle}(\mathsf{pk})$ after making at most $q_h$ queries to $|\mathcal{O}_{\hbar}\rangle$, and $q_s$
   queries to $\mathsf{Sign}_{\mathsf{sk}}$ receiving signatures $(x_1, \mathsf{sig}_1), \dots (x_{q_s}, \mathsf{sig}_{q_s})$
4: if $\mathsf{SigVerify}(\mathsf{pk}, x, \mathsf{sig}) = 1$ and $x \neq x_i \, \forall \, i = 1, \dots, q_s$ then
5:     Output: 1
6: else
7:     Output: 0
```

The advantage of $\mathcal{A}$ *is defined as:*

$$\mathsf{Adv}^{\mathsf{EUF-CMA-QRO}}_{\mathcal{S}ig,\mathcal{A}}(n, q_s, q_h) := \Pr\left[\mathsf{Game}^{\mathsf{EUF-CMA-QRO}}_{\mathcal{S}ig,\mathcal{A}}(n, q_s, q_h) \to 1\right].$$

Notice how, in the above experiment, only the adversary has access to $|\mathcal{O}_{\hbar}\rangle$, while honest parties have only access to $\mathcal{O}_{\hbar}$.

Definition 4.31 ((Post-Quantum) Existential Unforgeability under Chosen Message Attack in the Quantum Random Oracle Model (EUF-CMA-QRO))**.** *A DSS $\mathcal{S}ig$ is* (post-quantum) existentially unforgeable under chosen message attack in the quantum random oracle model (or, it is EUF-CMA-QRO secure) *iff, for any* QPT *algorithm $\mathcal{A}$ it holds that:*

$$\mathsf{Adv}^{\mathsf{EUF-CMA-QRO}}_{\mathcal{S}ig,\mathcal{A}} \leq \mathsf{negl}.$$

4.6 Fiat-Shamir in the QROM

The Fiat-Shamir transformation is a fascinating example of how things can go wrong when blindly switching to QPT adversaries in defining post-quantum security notions. The presence of a random oracle and, especially, of rewinding in the security proof makes this a case to be treated carefully.

In the last few years, a few works have been presented dealing with the FS transformation in a quantum world. Here, we only discuss the results from Dagdelen et al. [DFG13], which was hystorically the first work in the direction of assessing the security of FS in the quantum world.

Preliminaries

We start by defining *quantum-hard languages* as the 'post-quantum analogue' of hard languages.

Definition 4.32 (Quantum-Hard Language)**.** *A hard language $\mathcal{L}_{\mathcal{W},\mathcal{R},\mathsf{Inst}}$ is a* quantum-hard language *iff for any* QPT *algorithm $\mathcal{A}$ it holds:*

$$\Pr_{(x,w)\leftarrow\mathsf{Inst}}[(x,\mathcal{A}(x))\in\mathcal{R}]\leq\mathsf{negl}.$$

Next, we identify a special class of Σ-protocols, where the prover's commitment com does not depend on the witness w.

Definition 4.33 (Σ-Protocol with Witness-Independent Commitments)**.** *A Σ-protocol $(\mathcal{P},\mathcal{V})$ for a hard language $\mathcal{L}_{\mathcal{W},\mathcal{R},\mathsf{Inst}}$ has* witness-independent commitments *iff there exists a* PPT *algorithm* Com *which, on input a statement $x\in\mathcal{L}$, produces the same distribution as the prover's first message* $\mathsf{com}(x,w)$ *for input $(x,w)\leftarrow\mathsf{Inst}$. In this case, we also write the first message as* $\mathsf{com}\leftarrow\mathsf{Com}(x)$*.*

Many Σ-protocols are actually of this type. Examples are the well known graph-isomorphism proof [GMW86], the Schnorr proof of knowledge [Sch91], or the protocol for lattices used in an anonymous credential system [CNR12]. A typical example of non–witness-independent commitment Σ-protocol is the graph 3-coloring ZKPoK scheme [GMW86], where the prover commits to a random permutation of the coloring.

Finally, we define a class of Σ-protocols, where the prover's commitment com can be actually generated obliviously by the verifier instead.

Definition 4.34 (Σ-Protocol with Oblivious Commitments)**.** *A Σ-protocol $(\mathcal{P},\mathcal{V})$ for a hard language $\mathcal{L}_{\mathcal{W},\mathcal{R},\mathsf{Inst}}$ has* oblivious commitments *iff there exist* PPT *algorithms* Com *and* $\mathsf{SmplRnd}$ *such that the following distributions are statistically indistinguishable:*

```
Input: $n \in \mathbb{N}, (x, w) \in \mathcal{R}$
$r \xleftarrow{\$} \{0,1\}^{\mathsf{poly}(n)}$
$\mathsf{com} \leftarrow \mathsf{Com}(x; r)$
$\mathsf{ch} \leftarrow \mathcal{V}(x, \mathsf{com})$
$\mathsf{resp} \leftarrow \mathcal{P}(x, w, \mathsf{com}, \mathsf{ch})$
Output: $(x, w, r, \mathsf{com}, \mathsf{ch}, \mathsf{resp})$
```

```
Input: $n \in \mathbb{N}, (x, w) \in \mathcal{R}$
$(\mathsf{com}, \mathsf{ch}, \mathsf{resp}) \leftarrow (\mathcal{P}(x, w), \mathcal{V}(x))$
$r \leftarrow \mathsf{SmplRnd}(x, \mathsf{com})$
Output: $(x, w, r, \mathsf{com}, \mathsf{ch}, \mathsf{resp})$
```

Notice that a Σ-protocol with oblivious commitments has, in particular, witness-independent commitments. With oblivious commitments, the prover is able to compute a response from the given commitment com without knowing the randomness used to compute the commitment. This is usually achieved by placing some extra trapdoor into the witness w. For example, for the Guillou-Quisquater RSA based proof of knowledge [GQ88] where the prover shows knowledge of $w \in \mathbb{Z}_n^*$ with $w^e = y \bmod n$ for $x = (e, n, y)$, the prover would need to compute an e-th root for a given commitment $r \in \mathbb{Z}_n^*$. If the witness would contain the prime factorization of n, instead of the e-th root of y, this would indeed be possible.

Σ-protocols with oblivious commitments allow to move the generation of the commitment from the prover to the honest verifier. For most schemes this infringes on active security, because a malicious verifier could generate the commitment 'non-obliviously'. However, the scheme remains honest-verifier zero-knowledge, and this suffices for deriving secure signature schemes through the FS transformation. We call such modified scheme a Λ-*protocol*[2].

Definition 4.35 (Λ-Protocol)**.** *Let* $(\mathcal{P}, \mathcal{V})$ *be a* Σ-*protocol for a hard language* $\mathcal{L}_{\mathcal{W},\mathcal{R},\mathsf{Inst}}$ *with oblivious commitments. The* Λ-protocol $(\mathcal{P}_\Lambda, \mathcal{V}_\Lambda)$ associated to $(\mathcal{P}, \mathcal{V})$ *is a* 3*-move interactive protocol with exchange of messages* $r, (\mathsf{com}, \mathsf{ch}), \mathsf{resp}$ *between two* PPT *algorithms* $\mathcal{P}_\Lambda$ *and* $\mathcal{V}_\Lambda$ *such that:*

1. $\mathcal{P}_\Lambda(x, w) \to r$*, where* $r \xleftarrow{\$} \{0,1\}^{\mathsf{poly}(n)}$

2. $\mathcal{V}_\Lambda(x) \to (\mathsf{com}, \mathsf{ch})$*, where* $\mathsf{com} \leftarrow \mathsf{Com}(x; r)$*, and* $\mathsf{ch} \leftarrow \mathcal{V}(x, \mathsf{com})$

3. $\mathcal{P}_\Lambda(x, w, \mathsf{com}, \mathsf{ch}) \to \mathsf{resp}$*, where* $\mathsf{resp} \leftarrow \mathcal{P}(x, w, \mathsf{com}, \mathsf{ch}; r')$*, and* $r' \leftarrow \mathsf{SmplRnd}(x, \mathsf{com})$

4. $\mathcal{V}_\Lambda(x, \mathsf{com}, \mathsf{ch}, \mathsf{resp}) := \mathcal{V}(x, \mathsf{com}, \mathsf{ch}, \mathsf{resp})$

The generation of the initial randomness r can be performed by $\mathcal{V}_\Lambda$ himself, so that a Λ-protocol can generally be seen as a 2-move interactive protocol.

[2] The choice of the symbol 'Λ', in analogy to the choice of 'Σ' in 'Σ-protocol', is meant as a mnemonic graphical representation of the protocol flow. For Σ-protocols, in fact, the Σ recalls a stylization of the left-to-right (and viceversa) arrows denoting exchange of messages between one 'prover side' to the left and one 'verifier side' to the right when representing the protocol as a workflow, with the direction of time going down. Analogously, Λ-protocols can be seen as Σ-protocols where part of the interaction (i.e., some 'arrows') are removed. This is stylized by rotating the Λ by 90 degrees.

Impossibility Result for Post-Quantum Fiat-Shamir

In this section, we use a meta-reduction technique to rule out the existence of strongly black-box reductions for the Fiat-Shamir transformation of actively secure Σ-protocols under certain conditions. That is: *it is not possible to find reductions with strong security guarantees for the Fiat-Shamir transformation in the QRO, by only relying on the active security of certain Σ-protocols.* Before assessing more in detail the strength of this result, we outline here the proof. Recall that, classically, if $(\mathcal{P}, \mathcal{V})$ is a Σ-protocol, then its FS transform in the ROM, $\mathcal{S}ig^{\mathcal{O}_{\mathfrak{h}}}_{\mathsf{FS}}(\mathcal{P}, \mathcal{V})$, is an EUF-CMA-RO secure digital signature scheme (Theorem 3.54).

1. First we describe a hypothetical, all-powerful adversary $\mathcal{A}^{|\mathcal{O}_{\mathfrak{h}}\rangle}$ with quantum access to the random oracle (and no oracle access to the signing algorithm Sign at all), able to break the EUF-CMA-RO security (generate forgeries) for $\mathcal{S}ig^{\mathcal{O}_{\mathfrak{h}}}_{\mathsf{FS}}(\mathcal{P}, \mathcal{V})$ for any input public key. This adversary does not need to exist in practice – it is sufficient for our meta-reduction to successfully emulate it. The adversary $\mathcal{A}^{|\mathcal{O}_{\mathfrak{h}}\rangle}$ uses his unbounded power to find a secret key sk to its input pk, and then uses a (single) query to the random oracle to generate a forgery. Moreover, such adversary uses the quantum access to the random oracle to 'hide' his query in a superposition (this prevents any strong quantum reduction to apply the rewinding techniques of Pointcheval and Stern [PS00] as in the classical setting). Finally, this hypothetical adversary uses the secret key and the random oracle query to output a valid forgery.

2. Then we describe the behavior of a strongly black-box reduction $\mathcal{B}$ reducing the EUF-CMA-RO security of $\mathcal{S}ig^{\mathcal{O}_{\mathfrak{h}}}_{\mathsf{FS}}(\mathcal{P}, \mathcal{V})$ to the weak security of an identification scheme $(\mathcal{P}, \mathcal{V})$. We show how this is equivalent to finding valid witnesses for statements in a quantum-hard language $\mathcal{L}_{\mathcal{W},\mathcal{R},\mathsf{Inst}}$ by having only classical access to an efficient adversary for $\mathcal{S}ig^{\mathcal{O}_{\mathfrak{h}}}_{\mathsf{FS}}(\mathcal{P}, \mathcal{V})$. We call these very powerful reductions *strong quantum extractors* (or, in short, just 'extractors').

3. Then we build a reduction $\mathcal{M}$ which breaks the active security of $(\mathcal{P}, \mathcal{V})$ by having classical access to an extractor $\mathcal{B}$.

4. Finally, we show how $\mathcal{M}$ can successfully emulate the all-powerful adversary $\mathcal{A}$ for $\mathcal{B}$ by interacting with the honest prover $\mathcal{P}$ and with the same random oracle $\mathcal{O}_{\mathfrak{h}}$ generated by $\mathcal{B}$. That is, $\mathcal{M}$ is actually a *meta-reduction* which breaks the active security of $(\mathcal{P}, \mathcal{V})$ by using $\mathcal{B}$.

We give such impossibility result in respect to the subclass of witness-independent Σ-protocols, while leaving open the other cases. Moreover, we assume that the strong quantum extractor is *input-preserving* (i.e., it forwards

x faithfully to the adversary). In this case we can easily derandomize the adversary (with respect to classical randomness) by ‘hardwiring’ a key of a random function into it, which he initially applies to its input x to recover the same classical randomness for each run. Since the strong extractor has to work for all adversaries, it in particular needs to succeed for those where we pick the function randomly but fix it from thereon.

Theorem 4.36 (Impossibility Result for Fiat-Shamir)**.** *If $(\mathcal{P}, \mathcal{V})$ is an actively and weakly secure Σ-protocol with witness-independent commitments, then it does not admit any input-preserving strong quantum extractor.*

Proof. We follow the proof sketch above by giving explicit descriptions of the adversary $\mathcal{A}$, the extractor $\mathcal{B}$, and the meta-reduction $\mathcal{M}$. At the beginning of the game, the honest prover $\mathcal{P}$ generates a public/secret key pair $(\mathsf{pk}, \mathsf{sk}) \leftarrow$ KGen for the DSS $\mathcal{S}ig^{\mathcal{O}_{\hbar}}_{\mathsf{FS}}(\mathcal{P}, \mathcal{V})$ (which is actually a valid statement/witness pair $(x, w) \leftarrow \mathsf{Inst}$ for the quantum hard language $\mathcal{L}_{\mathcal{W},\mathcal{R},\mathsf{Inst}}$). The public key pk is also given to the honest verifier $\mathcal{V}$.

The Adversary. Our hypothetical, all-powerful adversary $\mathcal{A}$ works as follows (see Figure 4.2). He receives as input the public key $\mathsf{pk} = x$ and first uses its unbounded computational power to compute a random witness w' (according to uniform distributions of coin tosses $\mathcal{D}$ subject to $\mathsf{Inst}(n; \mathcal{D}) \rightarrow (x, w')$, but where $\mathcal{D}$ is a random function of x). Then $\mathcal{A}$ prepares all possible random strings $r \in \{0,1\}^{\mathfrak{r}(n)}$ (for some appropriate polynomial function $\mathfrak{r}$) for the prover's algorithm in superposition, i.e., $\mathcal{A}$ prepares the state:

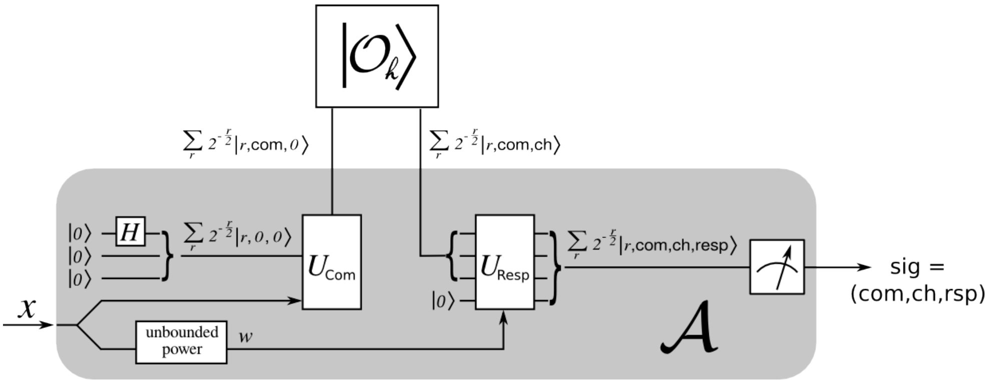
$$\sum_{r=0}^{2^{\mathfrak{r}}-1} \frac{1}{\sqrt{2^{\mathfrak{r}}}} \left| r \right\rangle$$

(this can be done efficiently by using Hadamard gates). In the next step, $\mathcal{A}$ evaluates (a unitary version of) the classical witness-independent algorithm Com for (deterministically) computing the prover's commitment com on this superposition (and on x) in order to obtain a superposition of all $\left| r, \mathsf{com} := \mathsf{Com}(x; r) \right\rangle$ plus an extra $\left| 0 \right\rangle$ ancilla register, i.e., the state:

$$\left| \varphi \right\rangle := \sum_{r=0}^{2^{\mathfrak{r}}-1} \frac{1}{\sqrt{2^{\mathfrak{r}}}} \left| r, \mathsf{com}, 0 \right\rangle .$$

At this point, $\mathcal{A}$ evaluates the QRO $\left| \mathcal{O}_{\hbar} \right\rangle$ in superposition on the com component of the above state (and using the public-key pk and a chosen message m), thereby obtaining the state:

$$\left| \psi \right\rangle := \sum_{r=0}^{2^{\mathfrak{r}}-1} \frac{1}{\sqrt{2^{\mathfrak{r}}}} \left| r, \mathsf{com}, \mathsf{ch} := \hbar(\mathsf{pk}, \mathsf{com}, m) \right\rangle .$$



$|\mathcal{O}_\hbar\rangle$
$\sum_r 2^{-\frac{r}{2}}|r,\mathsf{com},0\rangle$
$\sum_r 2^{-\frac{r}{2}}|r,\mathsf{com},\mathsf{ch}\rangle$
$|0\rangle$
H
$|0\rangle$
$|0\rangle$
$\sum_r 2^{-\frac{r}{2}}|r,0,0\rangle$
U_{Com}
U_{Resp}
$|0\rangle$
$\sum_r 2^{-\frac{r}{2}}|r,\mathsf{com},\mathsf{ch},\mathsf{resp}\rangle$
sig = (com,ch,rsp)
x
unbounded power
w
$\mathcal{A}$

Figure 4.2: the all-powerful adversary.

Then $\mathcal{A}$ computes, in superposition, responses $\mathsf{resp} \leftarrow \mathcal{P}(x, w', \mathsf{com}, \mathsf{ch}, r)$ for all values in the superposition, by using w' to emulate a valid prover, obtaining the state:

$$\sum_{r=0}^{2^{\mathfrak{r}}-1} \frac{1}{\sqrt{2^{\mathfrak{r}}}} |r, \mathsf{com}, \mathsf{ch}, \mathsf{resp}\rangle ,$$

Finally, $\mathcal{A}$ measures such state, obtaining a valid transcript $(\mathsf{com}, \mathsf{ch}, \mathsf{resp})$, and hence a valid forgery sig for $\mathcal{S}ig_{\mathsf{FS}}^{\mathcal{O}_\hbar}(\mathcal{P}, \mathcal{V})$.

The Extractor. An extractor $\mathcal{B}$ for $(\mathcal{P}, \mathcal{V})$ is a strong (black-box) quantum reduction which uses an adversary against $\mathcal{S}ig_{\mathsf{FS}}^{\mathcal{O}_\hbar}(\mathcal{P}, \mathcal{V})$ in order to break the weak security of $(\mathcal{P}, \mathcal{V})$. Therefore, it has the following characteristics.

- $\mathcal{B}$ is a QPT algorithm, taking as input a public-key pk for $\mathcal{S}ig_{\mathsf{FS}}^{\mathcal{O}_\hbar}(\mathcal{P}, \mathcal{V})$ (i.e., a statement x in $\mathcal{L}_{\mathcal{W},\mathcal{R},\mathsf{Inst}}$).

- Because he wants to break the *weak* security of $(\mathcal{P}, \mathcal{V})$, the goal of $\mathcal{B}$ is eventually to output a valid witness w'' for x.

- $\mathcal{B}$ is a black-box reduction, so it works by interacting with *any* successful adversary against the EUF-CMA-RO security of $\mathcal{S}ig_{\mathsf{FS}}^{\mathcal{O}_\hbar}(\mathcal{P}, \mathcal{V})$, but without having any information about the internal workings of the adversary. In particular, it must work for the all-powerful adversary $\mathcal{A}$.

- Because $\mathcal{A}$ eventually wants to interact with a quantum random oracle, $\mathcal{B}$ must also emulate a valid $|\mathcal{O}_\hbar\rangle$ for $\mathcal{A}$. In particular, $\mathcal{B}$ *must* be a quantum reduction.

- However, since $\mathcal{B}$ is a *strong* extractor, he is not allowed to tamper with $\mathcal{A}$'s queries to $|\mathcal{O}_\hbar\rangle$. That is, $\mathcal{B}$ cannot perform measurements or other quantum operations on those queries, except the evaluation through $|\mathcal{O}_\hbar\rangle$ (but $\mathcal{B}$ could, for example, reprogram the oracle, or rewind $\mathcal{A}$).

For example, such extractors might work by running $\mathcal{A}$ twice, obtaining two distinct signature forgeries for the same messages, and then applying the special soundness property of $(\mathcal{P}, \mathcal{V})$ to extract a valid witness w''. These extractors can be passive or active (i.e., interacting with $\mathcal{P}$), there is no restriction on that as long as they output a valid w''.

On the other hand, we restrict our impossibility result to extractors with the two following additional properties:

1. they are *input-preserving*, that is, the same statement x (public-key pk) input to $\mathcal{B}$ is relayed as input to the black-box adversary; and

2. they are *RO-broadcasting*, that is, they provide a public interface for evaluating $\mathcal{O}_{\hbar}$ to be used by other external parties, not only exclusively by the black-box adversary.

It is important to notice that this last condition is perfectly natural: recall that the ROM idealizes a publicly known hash function, so that it is reasonable to postulate that, once $\mathcal{B}$ has set up the emulated $|\mathcal{O}_{\hbar}\rangle$, everyone can have access to it. Actually, for this reason, one could also assume that the extractor is *QRO-broadcasting* (i.e., providing a public quantum interface for evaluating $|\mathcal{O}_{\hbar}\rangle$), but for our result it is sufficient for the meta-reduction to query $\mathcal{O}_{\hbar}$ classically, and a single query is enough.

The Meta-Reduction. We illustrate the meta-reduction $\mathcal{M}$ in Figure 4.3. Assume that there exists an extractor $\mathcal{B}$ with black-box access to an underlying quantum adversary $\mathcal{A}$, and which on input a statement (public-key) x sampled according to Inst, is able to extract a witness w'' to x by running several resetting executions of $\mathcal{A}$, each time answering $\mathcal{A}$'s QRO queries $|\varphi\rangle$ by emulating a QRO $|\mathcal{O}_{\hbar}\rangle$ for a classical, possibly probabilistic function $\hbar$ for which $\mathcal{B}$ also provides a public interface to be (at least classically) accessed by $\mathcal{M}$. Then $\mathcal{M}$ can use w'' to break the (weak and strong) security of the underlying Σ-protocol $(\mathcal{P}, \mathcal{V})$ by impersonating a valid prover for x against $\mathcal{V}$, against the assumption, and thereby concluding the proof.

It is left to show how $\mathcal{M}$ can succesfully simulate a quantum adversary for $\mathcal{B}$. In particular, we describe here how $\mathcal{M}$ can simulate the all-powerful adversary $\mathcal{A}$. Clearly, $\mathcal{M}$ can produce the same query $|\varphi\rangle$ that $\mathcal{A}$ produces, because of the witness-independence of $(\mathcal{P}, \mathcal{V})$. However, upon receiving back the reply $|\psi\rangle$ from $|\mathcal{O}_{\hbar}\rangle$, this state is discarded and ignored, and a valid forgery is instead generated in a different way. Namely, $\mathcal{M}$ initiates a $(\mathcal{P}, \mathcal{V})$ execution with the valid prover $\mathcal{P}$ for x, receiving a commitment com. $\mathcal{M}$ can now compute a valid challenge $\mathsf{ch} := \hbar(\mathsf{com})$ by using the public interface provided by $\mathcal{B}$ for evaluating $\hbar$, that is, $\mathcal{M}$ is simulating a valid verifier $\mathcal{V}$ for $\mathcal{P}$. At this point, a valid response resp is computed by $\mathcal{P}$, and $\mathcal{M}$ can use the transcript $(\mathsf{com}, \mathsf{ch}, \mathsf{resp})$ to output a valid forgery for $\mathcal{B}$. □

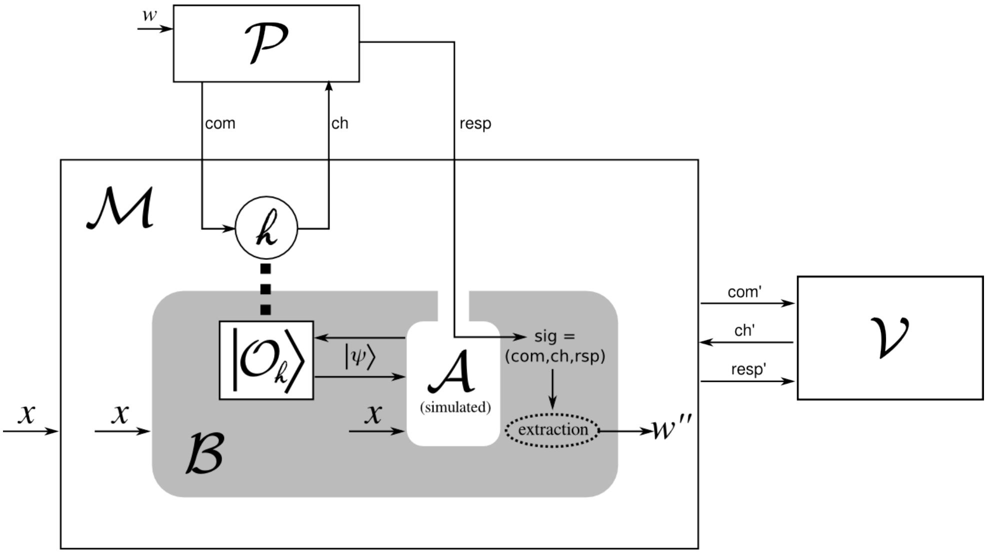


Figure 4.3: An overview of our meta-reduction

The above theorem is a special case of [DFG13, Theorem 3.3] with DSS in mind, but the Fiat-Shamir transform can also be cast in the scenario of non-interactive zero-knowledge proofs. It is important to notice that the above impossibility result has the following limitations:

- it only holds for witness-independent commitment Σ-protocols.
- It only holds for *strong black-box quantum extractors*. I.e., the extractor is not allowed to tamper with the adversary's queries to the QRO.
- The extractors must be input-preserving, i.e., they use their underlying black-box adversary by giving as input the same public-key used to break the Σ-protocol.
- It only holds for extractors breaking weak security, that is, *witness-extracting* – they are stronger than extractors who just win the impersonation game in the Σ-protocol.
- It is necessary that the extractor allows the meta-reduction to evaluate $\mathcal{O}_\hbar$ at least once.

Before discussing more in detail some of the above limitations, it is important to put this result in hystorical perspective: this was the *first* impossibility result for Fiat-Shamir in the quantum world, and following works [ARU14, Unr15] rely on more advanced tools. As already discussed, the witness-independence of the commitments is not a strong limitation, as most Σ-protocols have this property. Finally, notice that the existence of strong black-box extractors is *not* an unreasonable assumption – and therefore the above impossibility result is not unreasonably weak. In fact, Theorem 4.39 in the next section shows that certain Σ-protocols *do* indeed admit such extractors.

As we have already noticed, the extractor has to choose and provide public classical access to a classical function $\hbar$ for answering random oracle queries. While this may be considered a 'gray-box' restriction in general interactive quantum proofs, it seems to be inevitable in the QROM; it is rather a consequence of the approach where a quantum adversary mounts attacks in a classical setting. After all, both the honest parties as well as the adversary expect a classical hash function. The adversary is able to check this property easily, even if it treats the hash function otherwise as a black box (and may thus not be able to spot that the hash function uses (pseudo)randomness). We remark that this approach also complies with previous efforts [BDF$^+$11, BZ13a, Zha12b, Zha12a] and the positive result in the next section to answer such hash queries. Moreover, notice that in the above proof technically $\mathcal{M}$ only needs to evaluate $\hbar$ *once*, i.e., it must not necessarily require unlimited access to $\mathcal{O}_{\hbar}$. For these reasons, the meta-reduction still qualifies as *black-box*.

Furthermore, the extractor *can* rewind the quantum adversary to any point before the final measurement. Recall that for this impossibility result it is assumed, to the advantage of the extractor, that the adversary does not perform any measurement until the very end. Since the extractor can re-run the adversary from scratch for the same classical randomness, and the 'no-cloning restriction' does not apply to our adversary with classical input, the extractor can therefore easily put the adversary in the same (quantum) state as in a previous execution, up to the final measurement. However, because we consider *strong black-box* extractors, the extractor can only influence the adversary's behavior via the answers it provides to $\mathcal{A}$'s external communication. In this sense, the extractor may always rewind the adversary to such communication points. The extractor is also allowed to measure and abort at such communication points.

The extraction strategy by Pointcheval and Stern [PS00] in the purely classical case *can* be cast in the strong black-box extractor framework. For this the extractor would run the adversary for the same classical randomness twice, providing a lazy-sampling–based hash function description, with different replies in the i-th answers in the two runs. The extractor then extracts the witness from two valid signatures. This shows that a different approach than in the classical setting is necessary for extractors in the QROM.

One might ask why the meta-reduction does not apply to the Fiat-Shamir transform when adversaries have only classical access to the random oracle. The reason is the following: if the adversary made a classical query about a single commitment (and so would the meta-reduction), then one could apply the rewinding technique of Pointcheval and Stern changing the random oracle answers, and extract the underlying witness via special soundness of the identification scheme. The quantum adversary here, however, queries the random oracle in a superposition. In this scenario, as we explained above, the extractor is not allowed to 'read' the query of the adversary unless it makes the adversary stop. In other words, the extractor cannot measure the query and then keep running the adversary until a valid witness is output. This intrinsic property of strong black-box quantum extractors, hence, makes 'quantum' rewinding impossible. Note that rewinding in the classical sense – as described by Pointcheval and Stern – is still possible, as this essentially means to start the adversary with the same random coins. This does not cover the case where $\mathcal{B}$ measures (at least partially) the query state without disturbing $\mathcal{A}$'s behavior significantly (i.e., *non-strong* extractors), but subsequent works [Unr15] have also ruled out this possibility.

Finally, we briefly discuss that active security is basically necessary for an impossibility result as above. That is, we outline a three-move protocol for any quantum-hard language which, when applying the FS transformation, supports a straight-line extractor, and is honest-verifier zero-knowledge, but not actively secure. This holds as long as there are *post-quantum dense encryption schemes*, and *post-quantum non-interactive zero-knowledge proofs*. The latter are classical non-interactive zero-knowledge proofs (in the common random string model) for which simulated and genuine proofs are indistinguishable, even for *quantum* distinguishers. The former are pq-IND-CPA encryption schemes where honestly generated public keys are quantum-indistinguishable from random strings. The construction is based on the (classical) non-interactive zero-knowledge proofs of knowledge of De Santis and Persiano [SP92] and works as follows. The first message is irrelevant, e.g., we let the prover simply send the constant 0 (potentially padded with redundant randomness). In the second message the verifier sends a random string which the prover interprets as a public key pk of the dense encryption scheme and a common random string crs for the NIZK. The prover encrypts the witness under pk and gives a NIZK that the encrypted value forms a valid witness for the public value x. The verifier only checks the NIZK proof. The protocol is clearly not secure against active (classical) adversaries because such an adversary can create a public key pk via the key generation algorithm, thus, knowing the secret key and allowing the adversary to recover the witness from a proof by the prover. It is, however, honest-verifier zero-knowledge against quantum distinguishers, because of the pq-IND-CPA security and the simulatability of the NIZK hide the witness and allow for a simulation.

Security Result for Post-Quantum Fiat-Shamir

In this section, we show how it is possible to actually resurrect the security of the FS transformation for a certain class of Σ-protocols able to overcome the previous impossibility result. The intuition is the following: as such impossibility result works by exploiting the active security of the Σ-protocol, and since such property is not needed for the FS transformation to yield secure signature schemes, we can 'patch' the Σ-protocol by removing its active security. That is, by *weakening* the security guarantees of a Σ-protocol (seen as an identification scheme) we work toward *strengthening* the properties of its FS transform (seen as a DSS).

We achieve this goal by considering the FS transform of Λ-protocols obtained by Σ-protocols with oblivious commitments. In particular, using random oracles one can hash directly into pairs $(\mathsf{com}, \mathsf{ch})$ by first computing the output of the hash function obtaining a (public-coin) challenge ch and some randomness r', and then running $\mathsf{Com}(x; r')$ to sample a commitment com obliviously. The existence of $\mathsf{SmplRnd}$ guarantees that we could 'bend' this value back to an actual pre-image r for com. In the sequel we therefore often identify r' with $\mathsf{Com}(x; r')$ in the sense that we assume that the hash function maps to $\mathsf{Com}(x; r')$ directly, and for a (randomized) hash function $\hbar$ and message m we write $(\mathsf{com}, \mathsf{ch}) \leftarrow \hbar(x, m, r)$. The modified FS transformation then looks as follows.

Definition 4.37 (FS Transform of a Λ-Protocol)**.** *Let $(\mathcal{P}_\Lambda, \mathcal{V}_\Lambda)$ be a Λ-protocol for a hard language $\mathcal{L}_{\mathcal{W},\mathcal{R},\mathsf{Inst}}$, with commitment space $\mathcal{X}$ (with associated randomness space $\mathcal{X}^{-1} = \{r : r \leftarrow \mathsf{SmplRnd}\} := \{0,1\}^{\mathsf{poly}(n)}$), challenge space $\mathcal{Y}$, and response space $\mathcal{Z}$. Let $\mathcal{O}_\hbar$ be a random oracle for a random function $\hbar : \mathcal{L} \times \mathcal{M} \times \mathcal{X}^{-1} \to \mathcal{Y}$. The* FS transform of $(\mathcal{P}_\Lambda, \mathcal{V}_\Lambda)$ in the ROM, *$\mathcal{S}ig_{\mathsf{FS}}^{\mathcal{O}_\hbar}(\mathcal{P}_\Lambda, \mathcal{V}_\Lambda)$, is a DSS with message space $\mathcal{M}$, signature space $\mathcal{T} := \mathcal{Y} \times \mathcal{Z}$, and key space $\mathcal{K} := \mathcal{L} \times \mathcal{W}$, defined as follows:*

1. $\mathsf{KGen} \to (\mathsf{pk}, \mathsf{sk})$, *where* $(\mathsf{pk}, \mathsf{sk}) := (x, w) \leftarrow \mathsf{Inst}$

2. $\mathsf{Sign}^{\mathcal{O}_\hbar}(\mathsf{sk}, m) \to \mathsf{sig} := (r, \mathsf{resp})$,
   where $r \overset{\$}{\leftarrow} \mathcal{X}^{-1}$, $(\mathsf{com}, \mathsf{ch}) \leftarrow \hbar(\mathsf{pk}, m, r)$,
   and $\mathsf{resp} \leftarrow \mathcal{P}_\Lambda(\mathsf{pk}, \mathsf{sk}, \mathsf{com}, \mathsf{ch}, r)$

3. $\mathsf{SigVerify}^{\mathcal{O}_\hbar}(\mathsf{pk}, m, \mathsf{sig}) \to b$,
   where $\mathsf{sig} := (r, \mathsf{resp})$, $b \leftarrow \mathcal{V}(\mathsf{pk}, \hbar(\mathsf{pk}, m; r), \mathsf{resp})$

As we have already discussed, this modified FS transformation eludes the impossibility result from the previous section. In order to show its security, we exploit the special soundness of the Λ-protocol: by reprogramming the QRO $|\mathcal{O}_\hbar\rangle$ for a forgery-generating adversary $\mathcal{A}$, eventually we obtain two related transcripts $(\mathsf{com}^\star, \mathsf{ch}^\star, \mathsf{resp}^\star)$ and $(\mathsf{com}^\star, \mathsf{ch}', \mathsf{resp}')$ for $\mathsf{ch}^\star \neq \mathsf{ch}'$, and thus extracting a valid witness for x and breaking the weak security of $(\mathcal{P}_\Lambda, \mathcal{V}_\Lambda)$. The idea of the proof is as follows.

1. First, we run the HVZK simulator $\mathcal{S}$ of the Λ-protocol to obtain a valid transcript $(\mathsf{com}^\star, \mathsf{ch}^\star, \mathsf{resp}^\star)$.

2. We reprogram the QRO $|\mathcal{O}_\hbar\rangle$ by 'injecting' the value $(\mathsf{com}^\star, \mathsf{ch}')$ (for $\mathsf{ch}^\star \neq \mathsf{ch}'$) on a fraction δ of the possible oracle answers. That is, we replace $\mathcal{O}_\hbar$ with a semi-constant distribution $\mathcal{U}^\delta$.

3. Then, we run the adversary $\mathcal{A}$ against the modified quantum oracle, obtaining a forgery for $\mathcal{S}ig_{\mathsf{FS}}^{\mathcal{O}_\hbar}(\mathcal{P}_\Lambda, \mathcal{V}_\Lambda)$ for some message m, and hence a valid transcript $(\mathsf{com}, \mathsf{ch}, \mathsf{resp})$ for $(\mathcal{P}_\Lambda, \mathcal{V}_\Lambda)$.

4. Finally, if it happens that $\mathsf{com} = \mathsf{com}^\star$ and $\mathsf{ch} \neq \mathsf{ch}^\star$, we can use the special soundness extractor $\mathcal{J}$ to obtain a valid witness for x and breaking the weak security of $(\mathcal{P}_\Lambda, \mathcal{V}_\Lambda)$, concluding the proof.

In order for this proof strategy to work, the following two (seemingly contradictory) conditions have to be fulfilled:

- we need to ensure that $\mathcal{A}$ eventually outputs a valid signature yielding a transcript for the commitment $\mathsf{com}^\star$ of our choice (the one we obtained from the zero-knowledge simulator of the underlying Σ-protocol). This requires that $\mathsf{com}^\star$ appears with sufficiently large probability in the responses for oracle queries.

- On the other hand, we still require that $\mathcal{A}$ has a small probability of distinguishing a true QRO $|\mathcal{O}_\hbar\rangle$ from the reprogrammed one. Otherwise, the adversary may refuse to give a valid signature at all.

The following technical lemma shows that both conditions can be satisfied simultaneously by choosing δ carefully.

Lemma 4.38. *Let $(\mathcal{P}_\Lambda, \mathcal{V}_\Lambda)$ be a Λ-protocol for a quantum-hard language $\mathcal{L}_{\mathcal{W},\mathcal{R},\mathsf{Inst}}$, and let $\mathcal{O}'$ be the oracle obtained by reprogramming $\mathcal{O}_\hbar$ on a fraction δ of its possible inputs (pk, m, r) such that $\mathcal{O}'(\mathsf{pk}, m, r) = (\mathsf{com}^\star, \mathsf{ch}')$ with probability $\delta \in (0,1)$ for fixed values $\mathsf{com}^\star$ and ch'. Let $\mathcal{A}$ be a QPT algorithm such that $\mathcal{A}^{|\mathcal{O}_\hbar\rangle}(\mathsf{pk})$ outputs a valid forgery for $\mathcal{S}ig_{\mathsf{FS}}^{\mathcal{O}_\hbar}(\mathcal{P}_\Lambda, \mathcal{V}_\Lambda)$ for a public key pk with probability at least ε after performing q_h queries to $|\mathcal{O}_\hbar\rangle$, and let $(\mathsf{com}, \mathsf{ch}, \mathsf{resp})$ the transcript obtained by the output of the same algorithm $\mathcal{A}^{|\mathcal{O}'\rangle}(\mathsf{pk})$ running against the reprogrammed quantum oracle. Then:*

$$\Pr\left[\mathcal{V}_\Lambda^{\mathcal{O}'}(x, \mathsf{com}, \mathsf{ch}, \mathsf{resp}) \to 1 \wedge (\mathsf{com}, \mathsf{ch}) = (\mathsf{com}^\star, \mathsf{ch}')\right] \geq \delta \cdot \varepsilon - \frac{8}{3} \cdot q_h^4 \delta^2.$$

Proof. Consider the probability that we first run $\mathcal{A}$ on the original oracle $|\mathcal{O}_{\mathscr{h}}\rangle$ and check if it successfully forges a signature (r, resp) for pk and some message m (leading to a transcript $(\mathsf{com}, \mathsf{ch}, \mathsf{resp})$), and then, independently, we also verify that (pk, m, r) is mapped to $(\mathsf{com}^\star, \mathsf{ch}')$ under $\mathcal{O}'$. Then:

$$\Pr\left[\mathcal{A}^{|\mathcal{O}_{\mathscr{h}}\rangle}(\mathsf{pk}) \text{ succeeds } \wedge \mathcal{O}'(\mathsf{pk}, m, r) = (\mathsf{com}^\star, \mathsf{ch}')\right] \geq \delta \cdot \varepsilon.$$

This follows from the independence of the events: the oracle $\mathcal{O}'$ reprograms the output with probability δ, independently of $\mathcal{A}$'s behavior, but at the same time we know that $\mathcal{A}^{|\mathcal{O}_{\mathscr{h}}\rangle}$ succeeds with probability at least ε by assumption. Next, we replace $|\mathcal{O}_{\mathscr{h}}\rangle$ with $|\mathcal{O}'\rangle$ for $\mathcal{A}$, and we consider the new output (m, r, resp), arguing that:

$$\Pr\left[\mathcal{A}^{|\mathcal{O}'\rangle}(\mathsf{pk}) \text{ succeeds } \wedge \mathcal{O}'(\mathsf{pk}, m; r) = (\mathsf{com}^\star, \mathsf{ch}')\right] \geq \delta \cdot \varepsilon - \frac{8}{3} \cdot q_h^4 \delta^2.$$

This follows from Lemma 4.3: switching to the new oracle can change the distance of the output distribution of $\mathcal{A}$ by at most $\frac{8}{3} \cdot q_h^4 \delta^2$, and adding the verification step $\mathcal{V}_\Lambda^{\mathcal{O}'}(x, \mathsf{com}, \mathsf{ch}, \mathsf{resp}) \to 1$ cannot increase this distance. Therefore, we conclude that the probability for the event

$$\mathcal{V}_\Lambda^{\mathcal{O}'}(x, \mathsf{com}, \mathsf{ch}, \mathsf{resp}) \to 1 \wedge (\mathsf{com}, \mathsf{ch}) = (\mathsf{com}^\star, \mathsf{ch}')$$

cannot be smaller than the claimed bound, because $(\mathsf{com}, \mathsf{ch}) := \mathcal{O}'(\mathsf{pk}, m, r)$ by construction. □

The previous lemma informally tell us that, in order to succeed, we have to balance between a large δ to increase the chances of the adversary outputting a signature containing our desired $\mathsf{com}^\star$, and a small δ to avoid that the adversary detects the reprogrammed oracle. We are now ready to prove the main theorem.

Theorem 4.39 (Security of a Fiat-Shamir Transform for Λ-Protocols)**.** *Let* $(\mathcal{P}_\Lambda, \mathcal{V}_\Lambda)$ *be a* Λ*-protocol for a quantum-hard language. Then* $\mathcal{S}ig_{\mathsf{FS}}^{\mathcal{O}_{\mathscr{h}}}(\mathcal{P}_\Lambda, \mathcal{V}_\Lambda)$ *is an EUF-CMA-QRO secure DSS.*

Proof. We assume towards contradiction the existence of an efficient quantum adversary $\mathcal{A}$ which, on input a public key pk, outputs a valid forgery (m, sig) under pk with non-negligible probability ε, hence breaking the existential unforgeability of $\mathcal{S}ig_{\mathsf{FS}}^{\mathcal{O}_{\mathscr{h}}}(\mathcal{P}_\Lambda, \mathcal{V}_\Lambda)$. This adversary has access to a quantum-accessible random oracle $|\mathcal{O}_{\mathscr{h}}\rangle$ with $\mathscr{h}(\mathsf{pk}, m_i, r_j) = (\mathsf{com}_{i,j}, \mathsf{ch}_{i,j})$, and to a signing oracle $\mathsf{Sign}_{\mathsf{sk}}$ for the secret key sk (where $(\mathsf{pk}, \mathsf{sk}) := (x, w) \in \mathcal{R}$) producing, on input a message m, a (classical) signature $\mathsf{sig} = (r, \mathsf{resp}) \leftarrow \mathsf{Sign}^{\mathcal{O}_{\mathscr{h}}}(\mathsf{sk}, m)$.

The adversary $\mathcal{A}$ gets pk as an input, and is then allowed to perform up to $q_h = \mathsf{poly}(n)$ quantum queries to $|\mathcal{O}_{\mathscr{h}}\rangle$, and up to $q_s = \mathsf{poly}(n)$ classical queries to $\mathsf{Sign}_{\mathsf{sk}}$. Then, after running for $\mathsf{poly}(n)$ time, $\mathcal{A}$ produces (with non-negligible probability ε) a forgery (m, sig) such that m has never been asked to the signing oracle $\mathsf{Sign}_{\mathsf{sk}}$ throughout $\mathcal{A}$'s execution (i.e., m is a fresh message). We assume that q_h also covers a classical query of the verifier to check the signature.

Under these assumptions we show how to build a strong black-box quantum extractor $\mathcal{B}$, with access to $\mathcal{A}$ as a subroutine, and which is able to break the hardness of $\mathcal{L}_{\mathcal{W},\mathcal{R},\mathsf{Inst}}$ with non-negligible probability. That is, $\mathcal{B}$ on input $x \in \mathcal{L}$ generated according to Inst, is able to output a valid witness w' such that $(x, w') \in \mathcal{R}$ by only interacting classically with $\mathcal{A}$. The quantum extractor $\mathcal{B}$ works as follows:

- on input statement x, it first runs the simulator $\mathcal{S}$ of the underlying Λ-protocol to obtain a valid transcript $(\mathsf{com}^\star, \mathsf{ch}^\star, \mathsf{resp}^\star)$. This is possible because of the honest-verifier zero-knowledge property. Note also that this does not require access to the random oracle. As already explained, we assume for simplicity that the oblivious commitment is a random string; else we would need to run $\mathsf{SmplRnd}$ on $(\mathsf{pk}, \mathsf{com}^\star)$ to derive a preimage randomness r, and then use r in the hash reply (and argue that this is indistinguishable).

- Then, $\mathcal{B}$ simulates a quantum-classical oracle $|\mathcal{O}_0\rangle := |\mathcal{O}_{\hbar}^{\delta}\rangle$ which is obtained by reprogramming a (simulated) quantum random oracle $|\mathcal{O}_{\hbar}\rangle$ over a fraction δ of its possible inputs (pk, m, r) with the value $(\mathsf{com}^\star, \mathsf{ch}')$. Here, δ is some non-negligible probability in the security parameter (whose optimal value will be computed later), and ch' is an arbitrarily chosen challenge different from $\mathsf{ch}^\star$. That is, $\mathcal{O}_0(\mathsf{pk}, m, r) = (\mathsf{com}^\star, \mathsf{ch}')$ with probability δ, and random elsewhere.

- Next, $\mathcal{B}$ invokes $\mathcal{A}$ on input $\mathsf{pk} = x$.

- Whenever $\mathcal{A}$ performs the i-th (classical) query to $\mathsf{Sign}_{\mathsf{sk}}$ for signing a message m_i, $\mathcal{B}$ does the following:
  - choose a random value $r_i \xleftarrow{\$} \mathcal{X}^{-1}$;
  - execute the honest-verifier zero-knowledge simulator $\mathcal{S}$ of the Λ-protocol, obtaining a valid (simulated) transcript$(\mathsf{com}_i, \mathsf{ch}_i, \mathsf{resp}_i)$;
  - reprogram $\mathcal{O}_{i-1}$ with value $(\mathsf{com}_i, \mathsf{ch}_i)$ for the input (pk, m_i, r_i). We denote by $\mathcal{O}_i$ the reprogrammed oracle after the i-th query to the signing oracle;
  - then output $\mathsf{sig}_i := (r_i, \mathsf{com}_i, \mathsf{ch}_i, \mathsf{resp}_i)$ as $\mathsf{Sign}_{\mathsf{sk}}$'s reply to $\mathcal{A}$.

- Finally, when $\mathcal{A}$ outputs a (hopefully valid) fresh forgery (m, sig), where $\mathsf{sig} = (r, \mathsf{resp})$ and $\mathcal{O}_{q_s}(\mathsf{pk}, m; r) = (\mathsf{com}, \mathsf{ch})$, the extractor $\mathcal{B}$ aborts if $\mathsf{com} \neq \mathsf{com}^\star$ or $\mathsf{ch} = \mathsf{ch}^\star$. Otherwise, it uses the special soundness extractor $\mathcal{J}$ of the underlying Λ-protocol on input $(\mathsf{com}^\star, \mathsf{ch}^\star, \mathsf{resp}^\star)$ and $(\mathsf{com}, \mathsf{ch}, \mathsf{resp})$ to obtain a valid witness w' for x, concluding the attack.

Note that we can formally let $\mathcal{B}$ implement the dynamic reprogramming of the quantum-classical oracle, basically hardwiring all changes due to reprogramming into the code of the underlying classical algorithm. In a second step we can emulate the quantum oracle as explained in Section 4.2.

We next show that the success probability of our extraction procedure $\mathcal{B}$ is non-negligible given a successful $\mathcal{A}$. The proof follows the common game-hopping technique where we gradually deprive the adversary of (a negligible amount of) its success probability.

Game₁ : this is $\mathsf{Game}^{\mathsf{EUF-CMA-QRO}}_{\mathcal{S}ig^{\mathcal{O}_\hbar}_{\mathsf{FS}}(\mathcal{P}_\Lambda,\mathcal{V}_\Lambda),\mathcal{A}}$ describing $\mathcal{A}$'s original attack against $\mathcal{S}ig^{\mathcal{O}_\hbar}_{\mathsf{FS}}(\mathcal{P}_\Lambda, \mathcal{V}_\Lambda)$ constructed according to Definition 4.37, played against a public key pk. By assumption we have:

$$\Pr[\mathcal{A} \text{ wins } \mathsf{Game}_1] \geq \varepsilon$$

for some non-negligible value ε.

Game₂ : this game is identical to Game_1, except that we abort if $\mathcal{A}$ outputs a valid fresh forgery (m, sig) where sig *does not* contain a randomness leading to the pre-selected commitment $\mathsf{com}^\star$ and challenge ch'. Furthermore, we replace the random oracle $\mathcal{O}_\hbar$ with the oracle $\mathcal{O}_0$. Recall that $\mathcal{O}_0$ is obtained by reprogramming $\mathcal{O}_\hbar$ on a fraction δ of its entries with the value $(\mathsf{com}^\star, \mathsf{ch}')$. By Lemma 4.38 we have:

$$\Pr[\mathcal{A} \text{ wins } \mathsf{Game}_2] \geq \delta\varepsilon - \frac{8}{3} q_\hbar^4 \delta^2.$$

Game₃ is actually a sub-sequence of q_s different experiments denoted by $\mathsf{Game}_3^{(i)}$ for $i = 1, \ldots, q_s$.

$\mathbf{Game_3^{(1)}}$: this is as Game_2, but this time $\mathcal{O}_0$ is reprogrammed to $\mathcal{O}_1$ (i.e., $\mathcal{O}_1(\mathsf{pk}, m_1, r_1) := (\mathsf{com}_1, \mathsf{ch}_1)$) as soon as $\mathcal{A}$ performs its 1^{st} classical query m_1 to $\mathsf{Sign}_{\mathsf{sk}}$. From then on, the oracle $\mathcal{O}_1$ always answers consistently with this value. We need to show that this switching does not change the winning probability significantly. For this we basically need to show that, so far, the amplitudes of this value (pk, m_1, r_1) in the queries to the quantum oracle are small, or else the adversary may be able to spot some inconsistency.

Let $\mathcal{X}^{-1}$ the randomness space from $\mathsf{SmplRnd}$ as from Definition 4.37, and let $|\mathcal{X}^{-1}| = 2^{\mathscr{z}}$ for some function $\mathscr{z}$ polynomial in the security parameter. We define the value $(\mathsf{pk}, m'_i, r'_j)$ to have *high amplitude* if there exists at least one of the quantum queries $|\varphi_1\rangle, |\varphi_2\rangle, \ldots$ to the quantum oracle $|\mathcal{O}_0\rangle$ *before the current (1^{st}) signing query*, where the amplitude $a_{i,j}$ associated to the corresponding basis element of $(\mathsf{pk}, m'_i, r'_j)$ is such that $|a_{i,j}|^2 \geq 2^{\frac{-\mathscr{z}}{2}}$. Otherwise, the tuple is said to have *low amplitude*. Note that each query to the quantum oracle can have at most $2^{\frac{\mathfrak{r}}{2}}$ tuples with high amplitude, because the (square of the) amplitudes need to sum up to 1.

When $\mathcal{O}_0$ is reprogrammed to $\mathcal{O}_1$, the choice of m_1 is fixed (i.e., determined by the 1^{st} query of $\mathcal{A}$ to $\mathsf{Sign}_{\mathsf{sk}}$), but r_1 is still chosen uniformly at random in $\mathcal{X}^{-1}$. Since $\mathcal{A}$ performs at most q_h queries to $|\mathcal{O}_0\rangle$ before the signing query, we have thus at most $q_h \cdot 2^{\frac{\mathfrak{r}}{2}}$ tuples with high amplitude before this query. The probability of hitting such a tuple is then given by:

$$\Pr\left[(\mathsf{pk}, m_1, r_1) \text{ has high amplitude}\right] \leq q_h \cdot 2^{\frac{-\mathfrak{r}}{2}}. \tag{4.1}$$

Moreover, provided (pk, m_1, r_1) has *low* amplitude, and since there are at most $q_h + q_s$ query steps, using Lemma 2.12 and Lemma 2.11 we obtain:

$$\left|\mathcal{A}^{|\mathcal{O}_0\rangle} - \mathcal{A}^{|\mathcal{O}_1\rangle}\right| \leq 4\sqrt{(q_h + q_h) \cdot 2^{\frac{-\mathfrak{r}}{2}}}. \tag{4.2}$$

Let us assume, on behalf of the adversary, that $\mathcal{A}$ fails whenever (pk, m_1, r_1) has high amplitude. Still, from equations (4.1) and (4.2), we have:

$$\begin{aligned}\Pr\left[\mathcal{A} \text{ wins } \mathsf{Game}_3^{(1)}\right] &\geq \Pr\left[\mathcal{A} \text{ wins } \mathsf{Game}_2\right] - 4\sqrt{(q_h + q_s) \cdot 2^{\frac{-\mathfrak{r}}{2}}} - q_H \cdot 2^{\frac{-\mathfrak{r}}{2}} \\ &= \delta\varepsilon - \frac{8}{3} q_h^4 \delta^2 - \mathsf{negl}.\end{aligned}$$

Here, we use the fact that reprogramming the oracle for (pk, m_1, r_1) does not change the adversary's success probability for a forgery *for a fresh message m*. That is, since the adversary's forgery is for $m \neq m_1, m_2, \ldots$ it cannot simply copy a signature query as a forgery, but must still forge on the original oracle $\mathcal{O}_0$. So the argument about the winning probability applies as it did for $\mathcal{O}_0$.

We now repeat at most q_s times the game hopping, from $\mathsf{Game}_3^{(1)}$ to $\mathsf{Game}_3^{(q_s)}$, every time repeating the previous game but switching from $\mathcal{O}_{i-1}$ to $\mathcal{O}_i$ during the i^{th} query to $\mathsf{Sign}_{\mathsf{sk}}$, each time losing at most a negligible factor in the winning probability. Note that the probability of hitting a high amplitude with the signature generation in each hop increases to at most $q_h \cdot 2^{\frac{-\mathfrak{r}}{2}} + q_s \cdot 2^{-\mathfrak{r}}$ when taking into account the at most q_s hash queries in the previous signature requests, but this remains negligible.

After q_s steps we reach the following game.

$\mathbf{Game_3^{(q_s)}}$: as Game_2, but now $\mathcal{O}_0$ is dynamically reprogrammed as a sequence $\mathcal{O}_1, \ldots, \mathcal{O}_{q_s}$ throughout all of the $\mathcal{A}$'s queries to $\mathsf{Sign}_{\mathsf{sk}}$. We have:

$$\Pr\left[\mathcal{A} \text{ wins } \mathsf{Game}_3^{(q_s)}\right] \geq \delta\varepsilon - \frac{8}{3} q_h^4 \delta^2 - \mathsf{negl}.$$

$\mathbf{Game_4}$: as before, but now $\mathsf{Sign}_{\mathsf{sk}}$ is just simulated through the zero-knowledge simulator $\mathcal{S}$ of the underlying Λ-protocol. If, by contradiction, $\mathcal{A}$'s winning probability is affected by more than a negligible amount in so doing, then we could use $\mathcal{A}$ to build an efficient distinguisher between 'real' and 'simulated' transcripts of the Λ-protocol. This would require a distinguisher with access to a random oracle, in order to simulate the game. According to [Zha12b, Theorem 6.1], however, we can simulate the oracle via q-wise independent functions (which exists without requiring cryptographic assumptions). Furthermore, a hybrid argument can be used to reduce the case of q_s proofs to a single proof. Therefore:

$$\Pr\left[\mathcal{A} \text{ wins } \mathsf{Game}_4\right] \geq \delta\varepsilon - \frac{8}{3} q_h^4 \delta^2 - \mathsf{negl}.$$

Game$_5$: finally, in this game the special soundness extractor $\mathcal{J}$ is run on the transcript obtained from $\mathcal{A}$'s output from the previous game. Change the winning condition of $\mathcal{A}$ such that the adversary wins if this extraction yields a valid witness w' for x. If the winning probability in this game is more than negligibly far from the winning probability of $\mathcal{A}$ in the previous game then this can only be due to the fact that the simulated proof with $(\mathsf{com}^\star, \mathsf{ch}^\star, \mathsf{resp}^\star)$ cannot be accepted by the verifier; else the extractor would be guaranteed to work for this proof and the (accepted) signature. But this would allow an easy distinguisher against the zero-knowledge property, similar to the previous games. Hence:

$$\Pr\left[\mathcal{A} \text{ wins } \mathsf{Game}_5\right] \geq \delta\varepsilon - \frac{8}{3} q_h^4 \delta^2 - \mathsf{negl}.$$

Note that $\mathcal{A}$'s winning condition in the final game corresponds exactly to the probability of $\mathcal{B}$ successfully deriving a witness w' for its input x. This winning probability can be maximized (by zeroing the first derivative in δ) by choosing:

$$\delta := \frac{3\varepsilon}{16 q_h^4}.$$

This yields:

$$\Pr\left[\mathcal{A} \text{ wins } \mathsf{Game}_5\right] \geq \frac{3\varepsilon^2}{16 q_H^4} - \mathsf{negl},$$

which is non-negligible. This concludes the proof of the theorem. □

The results from this section regarding the security and impossibility results for the Fiat-Shamir transform of witness-independent commitments in the QROM is summarized in Figure 4.4: a security proof can be found for Σ-protocols with oblivious commitments (that is, Λ-protocols), while strong extractors can be ruled out whenever the FS transformation is applied to Σ-protocols which are actively secure (seen as identification schemes). However, some of these schemes can be 'patched' by using commitment trapdoors in order to make them oblivious commitment and remove their active

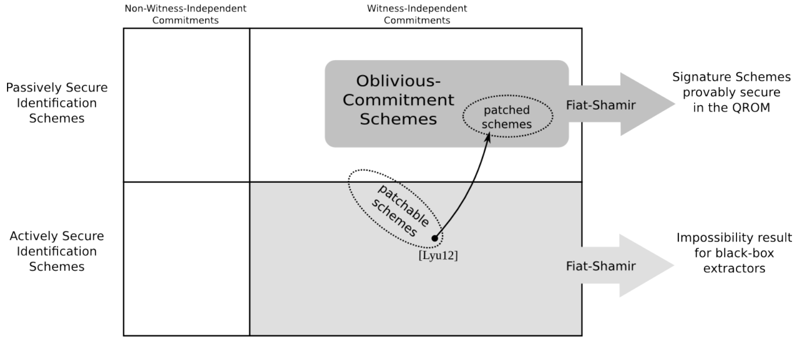

Non-Witness-Independent Commitments
Witness-Independent Commitments
Passively Secure Identification Schemes
Actively Secure Identification Schemes
Oblivious-Commitment Schemes
patched schemes
patchable schemes
[Lyu12]
Fiat-Shamir
Fiat-Shamir
Signature Schemes provably secure in the QROM
Impossibility result for black-box extractors

Figure 4.4: Security results for the Fiat-Shamir transformation in the QROM.

security, yielding signature schemes in a way similar to the hash-and-sign paradigm [GPV08]. This is for example the situation in the lattice-based signature scheme by Lyubashevsky [Lyu12], which can be patched in such a way to be rendered EUF-CMA-QRO secure according to Theorem 4.39, as explained in [DFG13].

4.7 Post-Quantum ORAMs

In this section we look at the post-quantum security of ORAMs. First of all, we define a suitable security model. Then we show that the extension of a classically secure ORAM to its post-quantum secure counterpart is not necessarily trivial. To this end, we examine `PathORAM` and we show that merely substituting the underlying encryption scheme with a post-quantum one does not generally yield a post-quantum ORAM. The idea is to exploit the weakness of other components of the ORAM construction under examination (in this case, the PRNG used). This is not surprising, because it has to be somewhat expected that post-quantum security can only be achieved by hardening *all* the underlying components of a cryptographic scheme, not only the encryption. However, it is important to keep this possibility in mind.

Then, we show that building post-quantum secure ORAMs is possible. We do it by showing that `PathORAM`, instantiated with a post-quantum secure SKES *and* a post-quantum PRNG, achieves post-quantum security. This is important from an application perspective, because it shows that efficient and post-quantum secure ORAMs can indeed be obtained in a straightforward way. Moreover, the proof of this fact is a straightforward adaptation from Theorem 3.64, and the resulting security reduction is semi-classical, therefore offering very strong security guarantees, as discussed in Section 4.1.

Post-Quantum Security of ORAMs

Since the security model for ORAM only involves a classical communication channel and there is no oracle access involved, we can simply switch to a post-quantum model of security for ORAMs in the usual way: we keep the AP-IND-CQA game as from Experiment 3.61, but we switch to QPT adversaries.

Definition 4.40 (Quantum ORAM Adversary)**.** *A quantum ORAM adversary $\mathcal{A}$ is a* QPT *algorithm which is computationally indistinguishable from an honest server $\mathcal{S}$ for every ORAM client $\mathcal{C}$. In particular, the ORAM's soundness is preserved.*

Definition 4.41 (Post-Quantum Access Pattern Indistinguishability Under Adaptive Chosen Query Attack)**.** *An ORAM construction* ORAM *has* post-quantum computationally indistinguishable access patterns under adaptive chosen query attack *(or, it is pq-AP-IND-CQA-secure) iff for any quantum ORAM adversary $\mathcal{A}$ it holds that* $\mathsf{Adv}^{\mathsf{AP-IND-CQA}}_{\mathsf{ORAM},\mathcal{A}} \leq \mathsf{negl}$.

Clearly, if an ORAM is pq-AP-IND-CQA-secure, then it is also AP-IND-CQA-secure. The converse does not hold (under standard hardness assumptions) as we will see.

The Impossibility Result

In order to show that one cannot in general obtain post-quantum ORAMs by just using a post-quantum SKES in a black-box way, we provide the following counterexample.

Theorem 4.42. *Let $\mathcal{E} = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ be a pq-IND-CPA SKES according to Definition 4.23, and let $\mathcal{G}_{BM}$ be the Blum-Micali PRNG from Lemma 4.12. Let $\mathtt{PathORAM}_{BM}$ be the ORAM obtained by instantiating the $\mathtt{PathORAM}$ construction from Definition 3.63 using $\mathcal{E}$ and $\mathcal{G}_{BM}$. Then, under the DLP hardness assumption, $\mathtt{PathORAM}_{BM}$ is an AP-IND-CQA secure ORAM, but not pq-AP-IND-CQA secure.*

At the light of Theorem 3.64 and Definition 4.41, in order to prove Theorem 4.42 we only need to show the following lemma.

Lemma 4.43. *There exists a* QPT *algorithm $\mathcal{A}$ winning* $\mathsf{Game}^{\mathit{AP\text{-}IND\text{-}CQA}}_{\mathcal{A},\mathtt{PathORAM}_{BM}}$ *with non-negligible advantage over guessing.*

Proof. We start by making a key observation concerning the access patterns produced in $\mathtt{PathORAM}$. Let $\mathsf{dr} = (\mathsf{op}, i, \mathsf{data})$ be a data request sent by $\mathcal{C}$. By only examining the communication transcript com resulting from the execution of this data request, one can see which path (branch of the tree) $\mathcal{S}$ sent to $\mathcal{C}$, thus learning the leaf r_i to which i was mapped to, even without knowing i itself. In normal circumstances, this is of no use to an adversary, because this value r_i becomes immediately obsolete, being replaced by a new fresh value output by the PRNG in the position map. But it will be important in our attack as we will see.

Let $\mathcal{D}$ be the BQP algorithm (the 'PRNG predictor') of Lemma 4.12. We build the adversary $\mathcal{A}$ with oracle access to $\mathcal{D}$. First of all $\mathcal{A}$ chooses $n, n_{\mathsf{db}} \leq n_{\mathsf{Max}}$ and starts the AP-IND-CQA game by calling $\mathsf{Init}(n, n_{\mathsf{db}})$. For his attack, $\mathcal{A}$ fixes an arbitrary identifier $i \in \{1, \ldots, n_{\mathsf{db}}\}$, and an arbitrary data unit $\mathsf{data} \in \{0,1\}^{n_{\mathsf{dat}}}$.

During the first CQA learning phase, $\mathcal{A}$ asks $\mathcal{C}$ to execute $\hbar = \mathsf{poly}(n)$ consecutive data requests of the form $(\text{'write'}, i, \mathsf{data})$. $\mathcal{A}$ records the resulting access patterns from all these queries, $\mathsf{ap}_1, \ldots, \mathsf{ap}_\hbar$, which include the communication transcripts $\mathsf{com}_1, \ldots, \mathsf{com}_\hbar$ and then, by the observation made before, a 'history' $(r_i^{(0)}, \ldots, r_i^{(\hbar-1)})$ of the past mappings of block i at the beginning of the execution of every data request from 1 to $\hbar$. These mappings, in turn, are $\hbar$ outputs of $\mathcal{G}_{BM}$, and they are given as input to the algorithm $\mathcal{D}$, which then outputs a candidate prediction r^* for the current secret leaf value $r_i^{(\hbar)}$.

Then $\mathcal{A}$ executes his challenge query by using data requests $(\mathsf{dr}^0, \mathsf{dr}^1)$ with $\mathsf{dr}^0 = (\text{'write'}, i, \mathsf{data})$, and $\mathsf{dr}^1 = (\text{'write'}, j, \mathsf{data})$ for $j \neq i$, and records the resulting access pattern $\mathsf{ap}_{\hbar+1} = \mathsf{ap}(\mathsf{dr}^b)$ (where b is the secret bit to be guessed). At this point, the adversary looks at this last communication transcript $\mathsf{com}_{\hbar+1}$ and, by the observation made at the beginning of the proof, checks the leaf index r related to the tree branch exchanged during the execution of the challenge query. If $r = r^*$, then $\mathcal{A}$ sets $b' = 0$ (where b' is $\mathcal{A}$'s current 'guess' at b), otherwise $\mathcal{A}$ sets $b' = 1$.

However, before outputting his guess b' in order to win the AP-IND-CQA game, $\mathcal{A}$ has to perform an additional check (during the second CQA challenge phase) in order to verify whether $\mathcal{D}$ had correctly guessed the right value $r_i^{(\hbar)}$ or not. The problem here is that, if $\mathcal{D}$ is unsuccessful (which happens with probability as high as $1 - \delta$), we cannot say anything about the predicted value r^*. In fact, in that case $\mathcal{D}$ could potentially act maliciously against $\mathcal{A}$, and output a value r^* which maximizes the probability of b' being wrong in the above strategy: for example, $r^* = r_j^{(0)}$. For this reason $\mathcal{A}$ performs the following 'sanity check' after the challenge query:

- if $b' = 1$, then $\mathcal{A}$ demands the execution of an additional query of the form $(\text{'write'}, i, \mathsf{data})$, and verifies that the resulting path leads to leaf r^*. This guarantees that r^* was actually correct, and it was not observed during the challenge query just because dr^1 was chosen, as guessed.

- Otherwise, if $b' = 0$, then $\mathcal{A}$ demands the execution of an additional query of the form $(\text{'write'}, j, \mathsf{data})$, and verifies that the resulting tree branch *does not* lead to leaf r^*. This guarantees with high probability that $\mathcal{D}$ did not maliciously output the secret leaf state for element j instead of i.

It is easy to see that in the case of misbehavior of $\mathcal{D}$, both of the above tests fail with high probability. In fact, in the case $b' = 1$, the current mapping of element i leads to leaf $r_i^{(\bar{k})}$, which was *not* correctly predicted by $\mathcal{D}$ by assumption. In the latter case instead, recall that $\mathcal{A}$ had guessed $b' = 0$ because during the execution of the challenge query he observed the leaf r^*; this could only lead to a fail in the case that $r_j^{(0)} = r_i^{(\bar{k})}$, which only happens with negligible probability at most ε, or if $r_j^{(0)} = r^*$, which is detected by the sanity check.

Finally, if the above sanity check is passed, $\mathcal{A}$ outputs b', otherwise he outputs a random bit.

Notice that (provided $\mathcal{D}$ was successful) this strategy is always correct, *except* in the case that: dr_1 was chosen (probability $\frac{1}{2}$) *and* the initial mapping of $\texttt{block}_j$ (which is $r_j^{(0)}$), coincides with $r_i^{(\bar{k})}$. As already mentioned, the latter event can only happen at most with probability ε negligible in the bit size of $\mathcal{G}_{BM}$'s output, and hence in the security parameter n (it is easy to see that this is a minimum requirement for any classically secure PRNG, as $\mathcal{G}_{BM}$ is). Thus:

$$\Pr\left[\mathsf{Game}^{\texttt{AP-IND-CQA}}_{\mathcal{A},\texttt{PathORAM}_{BM}} \to 1 \middle| \mathcal{D} \text{ succeeds}\right] \geq 1 - \frac{\varepsilon}{2}. \tag{4.3}$$

On the other hand, if $\mathcal{D}$ fails (which happens with probability $(1-\delta)$ at most) and predicts a wrong value $r^* \neq r_i^{(\bar{k})}$, the above strategy still succeeds with probability at least $\frac{1}{2} - \frac{\varepsilon}{2}$ (again, because of the remote possibility that $r_j^{(0)} = r_i^{(\bar{k})}$). Hence:

$$\Pr\left[\mathsf{Game}^{\texttt{AP-IND-CQA}}_{\mathcal{A},\texttt{PathORAM}_{BM}} \to 0 \middle| \mathcal{D} \text{ fails}\right] \leq \frac{1}{2}\left(1+\varepsilon\right). \tag{4.4}$$

Thus, combining 4.3 and 4.4, the adversary's overall success probability is:

$$\Pr\left[\mathsf{Game}^{\texttt{AP-IND-CQA}}_{\mathcal{A},\texttt{PathORAM}_{BM}} \to 1\right]$$

$$= \Pr\left[\mathcal{A} \text{ wins}\right] \cdot \Pr\left[\mathcal{D} \text{ succeeds}\right] + \left(1 - \Pr\left[\mathcal{A} \text{ loses}\right] \cdot \Pr\left[\mathcal{D} \text{ fails}\right]\right)$$
$$\geq \delta\left(1 - \frac{\varepsilon}{2}\right) + \left(1 - (1-\delta)\frac{1}{2}\left(1+\varepsilon\right)\right) \geq \frac{1}{2} + \frac{1}{2}\delta - \frac{1}{2}\varepsilon,$$

which concludes the proof, because ε is negligible, while δ is not. □

Construction of a Post-Quantum ORAM

A careful examination of $\texttt{PathORAM}$'s construction details reveals that an important role in the security is played by the pseudorandom number generator used to map a block to a leaf during every access. As we have just shown, a PRNG which is not post-quantum secure is enough to break $\texttt{PathORAM}$'s security in a quantum setting. It is natural then to wonder whether the attack on $\texttt{PathORAM}$ can be avoided by using a post-quantum PRNG, *in addition* to a post-quantum secure encryption scheme, when instantiating `PathORAM`. Here, we give a positive answer to such question.

Theorem 4.44. *Let $\mathcal{E}$ be a pq-IND-CPA SKE according to Definition 4.23, and let $\mathcal{G}$ be a pq-PRNG as from Definition 4.9. Then,* `PathORAM` *instantiated using $\mathcal{E}$ and $\mathcal{G}$ is a pq-AP-IND-CPA secure ORAM.*

Proof. The proof follows step-by-step the proof of Theorem 3.64. In fact this time, since $\mathcal{G}$ is a pq-PRNG by assumption, the new output values used to update the position map in `PathORAM` are indistinguishable from random (and therefore, in particular, unpredictable) even for QPT adversaries. As $\mathcal{G}$ has an internal state which is completely unrelated to $\mathcal{E}$'s internal randomness, and because there is no quantum oracle access involved, the security arguments at every step in the proof of Theorem 3.64 remain unchanged. Therefore, any QPT adversary who can distinguish the execution of two data request sequences with probability non-negligibly better than guessing, can be turned into a successful adversary against the pq-IND-CPA security of $\mathcal{E}$, or against the pqPRNG, against the security assumptions. □

Chapter 5

QS2: Quantum (Superposition-Based) Security

In this chapter we conclude our study of quantum security notions for classical cryptographic objects by presenting the quantum security class **QS**2. In this domain, the schemes are classical and the adversaries are quantum, as in **QS**1. However, unlike in **QS**1, the adversaries are *always* given quantum access to classical oracles, not only when the 'realistic' model requires it. So, for example, encryption schemes in **QS**2 must provide security against adversaries with quantum access to the encryption oracle, even in the secret-key case, and digital signature schemes must be unforgeable toward adversaries with quantum access to the signing oracle, even if such schemes are still classical. What we call here the **QS***2 principle* states: *"Whenever an adversary has access to a classical oracle, then such oracle should be accessible by the adversary in a quantum way."*

As we will see, the resulting security notions can be strictly stronger than 'post-quantum' notions as defined in the previous chapter. Constructions which are secure in **QS**2 retain in particular their security in **QS**1, but the converse does not always hold. **QS**2 is, in a sense, quantum security *beyond* post-quantum security. When a cryptographic construction is secure in the **QS**2 sense, we will just call it *quantum-secure.*

In the following sections first we discuss the motivations for considering this scenario, and then we introduce security models and definitions for quantum-secure cryptographic building blocks and secret-key encryption schemes.

My Scientific Contribution in this Chapter

Most of the new contributions in this chapter first appeared in [GHS16], which is a joint work with Andreas Hülsing and Christian Schaffner. The impossibility result in Section 5.3 is my contribution, but Andreas formalized the helpful tool of core function. The idea of superposition-based quantum indistinguishability, including the type-(2) oracles, the 'security tree' mentioned in Section 5.3, as well as the definitions of qIND and qIND-qCPA, and their relations to other notions, are my contribution. The idea of Constructions 5.18 and 5.19 and the intuition behind their security came from discussions between Andreas and me. However, Christian developed Lemma 5.20, and Andreas helped me with the proof of Theorem 5.22. Andreas also had the idea of extending quantum indistinguishability to the weak case, i.e., classical representations of quantum states. The resulting semantic security notion wqSEM is a joint work of all of the authors, as well as Theorem 5.36. However, the intermediate notions qaSEM and iqSEM are my contribution.

Quantum-secure PRPs (Definitions 5.3 and 5.4) were first formally defined in [GHS16]. Finally, Theorem 5.9 is considered folklore but, to the best of my knowledge, the first formal proof appears in this thesis.

5.1 Why Superposition Access?

The obvious question one might ask is: *"why considering quantum access to classical primitives, in the case where the adversary does not implement the primitive's code himself? Doesn't this clash with the* **QS***1 principle?"* Actually, it does not: the **QS**1 principle only states that whenever a quantum adversary can implement some code locally, this should be modeled as a quantum access, *but it does not say anything about the converse.* In fact, classical access to an oracle can be seen just as a special case of quantum access, where the adversary is limited to queries in the form of basis states. So, the first 'trivial' reason why one should consider quantum access is the following.

Reason #1: it is a more general model. Nothing is lost, in terms of security, by considering adversaries able to execute superposition queries. The resulting security notions will be at least *as strong* as the corresponding post-quantum security notions, and sometimes strictly so, as we will see. Of course this does not make post-quantum notions obsolete: for example it might be impossible (or much harder, or worse in performance) to achieve certain **QS**2 notions in contrast with the analogous **QS**1 notions. It will be the model and the circumstances to dictate whether post-quantum security is enough, or something more should be requested. But for sure, all other factors being equal, one does not lose anything by requesting security in the more challenging scenario considered in **QS**2.

There are, of course, less 'trivial' reasons. We have already met one in Section 4.2 about the emulation of a quantum random oracle: since the QROM describes an object with quantum superposition access by definition, emulating it using post-quantum PRFs would not be enough, because post-quantum PRFs are only accessed classically, and their security model says nothing about what happens when the access is quantum. For this reason, if we want to emulate a quantum oracle with PRFs, we need a security model which covers the quantum superposition access, *even* if we are using the quantum random oracle 'only' in a post-quantum security proof.

Another example is the case of post-quantum obfuscation, in particular *indistinguishability obfuscation (iO)*. This is a relatively recent branch of cryptographic techniques which, roughly speaking, achieves certain functionalities by 'obfuscating' the code of some algorithm in a secure way. One typical example (which has also received interest [CEJvO02] from an application perspective) is how to build PKES from SKES. The idea is to hardcode the secret key of the SKES in the code of the encryption routine, and then obfuscate the code and distribute it as a public key. In the standard model, it is known [IR88] that it is impossible to achieve key-exchange and public-key encryption in a black-box way just from one-way functions. However, Corollary 4.15 and Theorem 4.26 tell us that, using iO, it might be possible to build post-quantum PKES from pqOWF. Regardless whether iO is a reasonable assumption or not, it is clear that for this to work, the post-quantum security of the underlying SKES would *not* be enough because, as discussed in Section 4.4, post-quantum PKES can be queried in superposition. Therefore, for this application we also need a superposition-based security notion for SKES.

Summing up, we can say the following.

Reason #2: it is useful for post-quantum security proofs. If a security reduction for an object in **QS**0 fails when 'translating' it to **QS**1, one of the reasons (in addition to the ones described in Section 4.1) might be that the security of some of the underlying building blocks should be 'lifted' to **QS**2, not just **QS**1.

A less obvious reason regards the physical interaction between the adversary and the device where the cryptographic code is running. An adversary able to 'trick' a classical computation device into quantum behavior might exploit such behavior to gain superposition access to the function computed by the device. In order to fix the ideas on what this actually means we give a motivating example. In this mind experiment, we consider a not-so-distant future where the target of an attack is a tiny encryption chip, e.g., integrated into an RFID tag or smart-card. It is reasonable to assume that it will include elements of technology currently researched but undeployed (i.e., extreme miniaturization, optical electronics, etc.) Regardless, the chip we consider is a purely classical device, performing classical encryption (e.g., AES) on classical inputs, and outputting classical outputs. Consider an adversary equipped with some future technology which subjects the device to a fault-injection environment, by varying the physical parameters (temperature, power, speed, etc.) under which the device usually operates. As a figurative example, our ‘quantum hacker’ could place the chip into an isolation pod, which keeps the device at a very low temperature and shields it from any external electromagnetic or thermal interference. This situation would be analogous to what happens when security researchers perform side channel analysis on cryptographic hardware in nowaday’s labs, using techniques such as thermal or electromagnetic manipulation which were previously considered futuristic. There is no guarantee that, under these conditions, the chip does not start to show full or partial quantum behaviour. At this point, the adversary could query the device on a superposition of plaintexts by using, e.g., a laser and an array of beam splitters when feeding signals into the chip via optic fiber. It is unclear today what a future attacker might be able to achieve using such an attack. As traditionally done in cryptography, we assume the worst-case scenario where the attacker can actually query the target device in superposition. Classical and post-quantum security notions such as IND-CPA do not cover this scenario. This setting is an example of what we mean by ‘tricking classical parties into quantum behaviour’.

Another example of a sort of ‘quantum fault attack’ occurs in a situation where one party using a quantum computer encrypts messages for another party that uses a classical computer, and the adversary is able to observe the outcome of the quantum computation before measurement.

Reason #3: it covers quantum fault attack scenarios. Also notice that the threat deriving from these kind of attacks is potentially high considering that, unlike for the post-quantum scenario, they do not necessarily require the adversary to build a fully-fledged quantum computer.

Finally, it is important to consider superposition-based quantum security in all those cases where a classical cryptographic object is used as a building block for more complex quantum protocols (meant to run natively on quantum computing devices). Post-quantum guarantees alone are usually not enough to ensure secure composition in these scenarios.

Reason #4: it might be necessary for securely composing fully quantum constructions. For instance, we will see an example in the next chapter where schemes for securely encrypting quantum data can be built by adapting classical encryption schemes, but only if such schemes are (superposition-based) quantum-secure.

5.2 Quantum-Secure Building Blocks

We look first at the basic (superposition-based) quantum-secure building blocks. As already discussed in Section 4.3, there is nothing to say about quantum-secure OWF, OWTP, and PRNG. In the first two cases, the superposition access is already implied by the post-quantum definition, so that the post-quantum and the superposition-based quantum security notions coincide. We will use the two terms interchangeabily, as the meaning is the same. In the latter case instead, a superposition-based security notion for PRNG makes no sense, because PRNG security, by definition, is based on a stream of classical data, and there is no oracle access involved. As we mentioned already, the situation is instead quite different in the case of PRF and PRP.

Quantum-Secure PRF

In the case of pseudorandom functions, an adversary might be able to distinguish the PRF $\mathcal{F}$ from a random function by gaining quantum access to the oracle for $\mathcal{F}$, which we denote by $|\mathcal{F}\rangle$. Since a PRF is a keyed family of functions, we write sometimes $|\mathcal{F}_k\rangle$ to denote the quantum-classical oracle for $\mathcal{F}$ keyed by k.

Definition 5.1 (Quantum-Secure Pseudorandom Function (qPRF))**.** *A (family of)* quantum-secure pseudorandom functions (qPRF) *from $\mathcal{X}$ to $\mathcal{Y}$ with key space $\mathcal{K}$ is a* DPT *algorithm $\mathcal{F} : (k \in \mathcal{K}, x \in \mathcal{X}) \mapsto y \in \mathcal{Y}$ such that for any* QPT *algorithm $\mathcal{D}$ it holds:*

$$\left| \Pr_{k \xleftarrow{\$} \mathcal{K}} \left[\mathcal{D}^{|\mathcal{F}_k\rangle} \to 1 \right] - \Pr_{\hbar \xleftarrow{\$} \mathcal{Y}^{\mathcal{X}}} \left[\mathcal{D}^{|\mathcal{O}_\hbar\rangle} \to 1 \right] \right| \leq \mathsf{negl},$$

where $|\mathcal{O}_\hbar\rangle$ is a quantum-classical oracle for $\hbar$ (i.e., a quantum random oracle), and the probabilities are over the choice of k and $\hbar$, and the randomness of $\mathcal{D}$.

Obviously, a qPRF is also a pqPRF and, in particular, a PRF. As discussed in Section 3.1, and unlike in the case of pqPRFs in Section 4.3, the security proof of Theorem 3.7 does *not* go through, because of the impossibility of dealing with the quantum oracle access in the standard way required for such proof. However, [Zha12a] shows that qPRFs *can* indeed be built from post-quantum OWF using standard constructions, so the analogue of Corollary 4.15 still holds. The following is a corollary of [Zha12a, Theorem 4.5].

Theorem 5.2. *pqOWF exist iff qPRF exist.*

Quantum-Secure PRP

Quantum-secure PRPs are defined in a similar way as qPRFs, denoting by $|\mathcal{P}_k\rangle$ the quantum-classical oracle evaluating $\mathcal{P}$ with secret key k.

Definition 5.3 (Quantum-Secure Weak PRP (qWPRP))**.** *A (family of)* quantum-secure weak pseudorandom permutations (qWPRP) *on $\mathcal{X}$ with key space $\mathcal{K}$ is a pair of* DPT *algorithms* $(\mathcal{P},\mathcal{P}^{-1}):(k\in\mathcal{K},x\in\mathcal{X})\mapsto x'\in\mathcal{X}$ *such that:*

1. $\forall k \in \mathcal{K} \implies \mathcal{P}_k, \mathcal{P}_k^{-1}$ *are permutations on $\mathcal{X}$;*
2. $\forall k \in \mathcal{K} \implies (\mathcal{P}_k)^{-1} = \mathcal{P}_k^{-1}$*; and*
3. *for any* QPT *algorithm $\mathcal{D}$ it holds:*

$$\left| \Pr_{k \xleftarrow{\$} \mathcal{K}} \left[\mathcal{D}^{|\mathcal{P}_k\rangle} \to 1\right] - \Pr_{p \xleftarrow{\$} S(\mathcal{X})} \left[\mathcal{D}^{|\mathcal{O}_p\rangle} \to 1\right] \right| \leq \mathsf{negl},$$

where $|\mathcal{O}_p\rangle$ is a quantum-classical oracle for p, and the probabilities are over the choice of k and p, and the randomness of $\mathcal{D}$.

Definition 5.4 (Quantum-Secure Strong PRP (qSPRP))**.** *A (family of)* quantum-secure strong pseudorandom permutations (qSPRP) *on $\mathcal{X}$ with key space $\mathcal{K}$ is a pair of* DPT *algorithms* $(\mathcal{P}, \mathcal{P}^{-1}) : (k \in \mathcal{K}, x \in \mathcal{X}) \mapsto x' \in \mathcal{X}$ *such that:*

1. $\forall k \in \mathcal{K} \implies \mathcal{P}_k, \mathcal{P}_k^{-1}$ *are permutations on $\mathcal{X}$;*
2. $\forall k \in \mathcal{K} \implies (\mathcal{P}_k)^{-1} = \mathcal{P}_k^{-1}$*; and*
3. *for any* QPT *algorithm $\mathcal{D}$ it holds:*

$$\left| \Pr_{k \xleftarrow{\$} \mathcal{K}} \left[\mathcal{D}^{|\mathcal{P}_k\rangle,|\mathcal{P}_k^{-1}\rangle} \to 1\right] - \Pr_{p \xleftarrow{\$} S(\mathcal{X})} \left[\mathcal{D}^{|\mathcal{O}_p\rangle,|\mathcal{O}_{p^{-1}}\rangle} \to 1\right] \right| \leq \mathsf{negl},$$

where $|\mathcal{O}_p\rangle$ is a quantum-classical oracle for p, $|\mathcal{O}_{p^{-1}}\rangle$ is a quantum oracle for p^{-1}, and the probabilities are over the choice of k and p, and the randomness of $\mathcal{D}$.

It is important to notice that building provably secure qPRPs is not trivial. Kuwakado and Morii showed [KM10, KM12] that the two most commonly used constructions for building PRPs are actually *quantum-insecure*, in the sense that there exist specific quantum attacks (using a modified version of Simon's algorithm) able to distinguish such constructions from random. Their attacks are limited to the (3-round) Feistel construction (for building WPRPs from PRFs) and the (1-round) Even-Mansour construction (for building SPRPs from public random permutations). However, Zhandry [Zha16] shows that qSPRPs *can* indeed be built from qPRFs (and hence by post-quantum OWF) using constructions based on *format-preserving encryption*, so the analogue of the result from Theorem 4.18 still holds.

Theorem 5.5 (qPRF ⇔ qPRP)**.** *qPRFs exist iff qPRPs exist.*

5.3 Quantum-Secure Secret-Key Encryption

In Section 4.4, we have seen how security notions for public-key encryption in the post-quantum setting should allow for an adversary to query the encryption oracle in superposition. Following the **QS**2 principle, in this section we extend such a possibility to the secret-key scenario (we limit our analysis here to the CPA case). We start by considering indistinguishability notions for SKESs where the IND phase is still classical, but the adversary has oracle access to the encryption oracle (this would be the analogue, for SKESs, of the pq-IND-CPA notion for PKESs).

Then we look at what happens when also the IND query becomes quantum. We also discuss a modification of such scenario, which can be useful in certain situations, where the adversary is restricted to working with quantum messages having efficient classical representations.

We conclude with a brief discussion on the extension of the above models to the CCA1 and CCA2 scenarios.

Classical IND, Quantum CPA

The first indistinguishability notion with quantum CPA query phase, called IND-qCPA, was proposed in [BZ13b]. Formally, the base adversarial model is the same pq-IND adversary from Definition 4.19.

Experiment 5.6 ($\mathsf{Game}^{\mathsf{IND-qCPA}}_{\mathcal{E},\mathcal{A}}$)**.** *Let $\mathcal{E}$ be a SKES, and $\mathcal{A} := (\mathcal{M}, \mathcal{D})$ a pq-IND adversary. The* IND-qCPA experiment *proceeds as follows:*

```
Input: $n \in \mathbb{N}$
$k \leftarrow \mathsf{KGen}$
$(m^0, m^1, |\mathsf{state}\rangle) \leftarrow \mathcal{M}^{|\mathsf{Enc}_k\rangle}$
$b \xleftarrow{\$} \{0,1\}$
$c \leftarrow \mathsf{Enc}_k(m^b)$
$b' \leftarrow \mathcal{D}^{|\mathsf{Enc}_k\rangle}(c, \mathsf{state})$
if $b = b'$ then
    Output: $1$
else
    Output: $0$
```

The advantage of $\mathcal{A}$ *is defined as:*

$$\mathsf{Adv}^{\mathsf{IND-qCPA}}_{\mathcal{E},\mathcal{A}} := \Pr\left[\mathsf{Game}^{\mathsf{IND-qCPA}}_{\mathcal{E},\mathcal{A}} \to 1\right] - \frac{1}{2}.$$

Notice how, as in the **QS**0 case, we have: $\mathsf{Game}^{\mathsf{IND-qCPA}}_{\mathcal{E},\mathcal{A}} = \mathsf{Game}^{\mathsf{IND}}_{\mathcal{E},\mathcal{A}^{|\mathsf{Enc}_k\rangle}}$.

Definition 5.7 (Indistinguishability of Ciphertexts under Quantum Chosen Plaintext Attack (IND-qCPA))**.** *A SKES $\mathcal{E}$ has* indistinguishable encryptions under quantum chosen plaintext attack (or, it is IND-qCPA secure) *iff, for any pq-IND adversary $\mathcal{A}$ it holds that:* $\mathsf{Adv}^{\mathsf{IND-qCPA}}_{\mathcal{E},\mathcal{A}} \leq \mathsf{negl}$.

Clearly, IND-qCPA is at least as strong as pq-IND-CPA (and it is actually equivalent for PKES). But the converse is not true.

Theorem 5.8 (IND-qCPA $\implies$ pq-IND-CPA)**.** *If a SKES is IND-qCPA secure, then it is also pq-IND-CPA secure.*

Theorem 5.9 (pq-IND-CPA SKES $\centernot\implies$ IND-qCPA SKES)**.** *Under standard hardness assumptions, there exist SKES which are pq-IND-CPA secure, but not IND-qCPA secure.*

Proof (sketch). Consider the same counterexample described in the proof of Theorem 4.25, but where this time the public key used for the (IND-CPA but non–post-quantum secure) PKES is generated by KGen and kept secret. This way, in the post-quantum setting the adversary would lose access to the quantum encryption oracle for the PKES, and hence the pq-IND-CPA security notion coincides with the IND-CPA notion, which the resulting scheme achieves by construction. However, an adversary for the IND-qCPA security notion would still have access to such encryption oracle, thereby being able to break the security of the PKES, and thus recovering the SKES key. □

A simple modification from [BZ13b, Theorem 4.10] shows that Construction 3.26 is IND-qCPA when instantiated with a *quantum-secure PRF*.

Theorem 5.10. *Let $\mathcal{E}_\mathcal{F}$ be the SKES from Construction 3.26 implemented through a qPRF $\mathcal{F}$. Then $\mathcal{E}_\mathcal{F}$ in an IND-qCPA SKES.*

Type-(2) Oracles

Before discussing other quantum security notions, we must provide a technical tool arising from the following consideration. In quantum computing, the 'canonical' way of evaluating an oracle for a classical function f in superposition is, as discussed in Section 4.1, by using an auxiliary register and then the canonical quantum-classical oracle:

$$|\mathcal{O}_f\rangle : \sum_{x,y} a_{x,y} |x, y\rangle \mapsto \sum_{x,y} a_{x,y} |x, y \oplus f(x)\rangle .$$

This way ensures that the resulting operator is invertible, even if f itself is not. We call these *type-*(1) *transformations*, and we denote them by $|\mathcal{O}_f\rangle_{(1)}$ when necessary to specify (by default, we assume $|\mathcal{O}_f\rangle = |\mathcal{O}_f\rangle_{(1)}$). For SKES, if Enc_k is an encryption mapping m-bit plaintexts to c-bit ciphertexts, the resulting operator in this case will act on $m + c$ qubits in the following way:

$$|\mathsf{Enc}_k\rangle_{(1)} : \sum_{x,y} a_{x,y} |x, y\rangle \mapsto \sum_{x,y} a_{x,y} |x, y \oplus \mathsf{Enc}_k(x)\rangle ,$$

where the y's are ancillary values.

In our case, though, we do not consider arbitrary functions, but encryptions, which act as *bijections* on some bit-string spaces (assuming that the randomness, if in presence of a randomized SKES, is treated as an input, although never chosen by the adversary.) Therefore, provided that the encryption does not change the size of a message, the following transformation is also invertible:

$$\sum_x a_x \left|x\right\rangle \mapsto \sum_x a_x \left|\mathsf{Enc}_k(x)\right\rangle . \tag{5.1}$$

For the more general case of arbitrary message expansion factors, we will consider transformations of the form:

$$\sum_{x,y} a_{x,y} \left|x, y\right\rangle \mapsto \sum_{x,y} a_{x,y} \left|\varphi_{x,y}\right\rangle ,$$

where the length of the ancilla register is $|y| = |\mathsf{Enc}_k(x)| - |x|$ and $\varphi_{x,0} = \mathsf{Enc}_k(x)$ for every x – i.e., initializing the ancilla register in the $\left|0\right\rangle$ state produces a correct encryption, which is what we expect from an honest execution of the encryption. As in the SKES case we always assume that encryption and decryption oracles are actually provided by an honest third party (usually the *challenger*), we will not consider cases where $y \neq 0$. We call the resulting operator *type-*(2) *transformations*[1], and we denote them by $\left|\mathcal{O}_f\right\rangle_{(2)}$ when necessary to specify), that is:

$$\left|\mathsf{Enc}_k\right\rangle_{(2)} : \sum_x a_x \left|x, 0\right\rangle \mapsto \sum_x a_x \left|\mathsf{Enc}_k(x)\right\rangle ,$$

where the ancillary $\left|0\right\rangle$ is of the necessary qubit-size.

Notice that, in general, type-(1) and type-(2) transformations are very different: having quantum gate access to a type-(2) unitary encryption oracle (that is, quantum oracle access to $\left|\mathsf{Enc}_k\right\rangle_{(2)}$ and its adjoint $\left|\mathsf{Enc}_k\right\rangle^{\dagger}_{(2)}$) also gives access to the related type-(2) *decryption oracle* $\left|\mathsf{Dec}_k\right\rangle_{(2)} : \sum_x a_x \left|\mathsf{Enc}_k(x)\right\rangle \mapsto \sum_x a_x \left|x\right\rangle$. In fact, notice that $\left|\mathsf{Enc}_k\right\rangle^{\dagger}_{(2)} = \left|\mathsf{Dec}_k\right\rangle_{(2)}$, while the adjoint of a type-(1) encryption operator, $\left|\mathsf{Enc}_k\right\rangle^{\dagger}_{(1)}$, is generally *not* a type-(1) decryption operator. In particular, type-(2) operators are 'more powerful' in the sense that knowledge of the secret key is required in order to build any efficient quantum circuit implementing them. However, we stress the fact that whenever access to a decryption oracle is allowed, the two models are completely equivalent, because then we can simulate a type-(2) operator by using ancilla qubits and 'uncomputing' the resulting garbage lines (see Figure 5.1). This is in fact the case in our security model for SKES, as it is not the adversary himself who computes the encryptions, but they are instead provided by a challenger who, in particular, already knows the secret key.

[1] These are called *minimal quantum oracles* in [KKVB02].

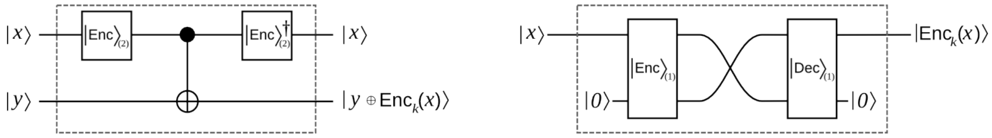

$|x\rangle$
$|y\rangle$
$|\mathsf{Enc}\rangle_{(2)}$
$|\mathsf{Enc}\rangle^{\dagger}_{(2)}$
$|x\rangle$
$|y \oplus \mathsf{Enc}_k(x)\rangle$
$|x\rangle$
$|0\rangle$
$|\mathsf{Enc}\rangle_{(1)}$
$|\mathsf{Dec}\rangle_{(1)}$
$|\mathsf{Enc}_k(x)\rangle$
$|0\rangle$

Figure 5.1: Equivalence between type-(1) and type-(2) in the case of 1-qubit messages. Left: building a type-(1) encryption oracle by using a type-(2) encryption oracle (and its inverse) as a black-box. Right: building a type-(2) encryption oracle by using type-(1) encryption and decryption oracles as black-boxes.

Quantum Indistinguishability

When trying to apply the **QS**2 principle at its fullest in the context of security notions for SKESs, the main difficulty is how to properly define a quantum version of the IND notion. As shown in [BZ13b], trying to define a new notion where all the communication and interaction of IND is blindly moved into quantum registers does not work, because the resulting notion would be trivially unachievable. The work [GHS16] presents an in-depth discussion about other possible strategies spanning a *'security tree'* of definitions. Most of these strategies lead to quantum indistinguishability notions that are either unachievable, or equivalent to IND-qCPA. However, some of them lead to more meaningful notions for the **QS**2 setting. These notions, in [GHS16], are called *quantum indistinguishability (qIND)* and *general quantum indistinguishability (gqIND)*. However, for the purpose of this work, we rename them as *weak quantum indistinguishability (wqIND)* and *quantum indistinguishability (qIND)* respectively, because the latter is of more direct interest to our framework. That is, what we call 'qIND' in this work was originally called 'gqIND' in [GHS16], and what we call 'wqIND' was originally called 'qIND' in [GHS16]. We will use such denomination from now on, and we will discuss qIND in this section, while presenting wqIND at a later point.

We give the qIND model for the most general case of adversaries able to query oracles on mixed states. This can happen if, for example, the adversary queries the oracle on a state which is entangled with another state kept by the adversary. Basically, what happens in the qIND experiment is the following:

1. first, the adversary outputs two quantum states φ^0, φ^1 representing the challenge plaintexts of his choice. These states can be thought as superpositions of classical plaintexts, but in general can also be mixed states, possibly entangled together or with some other state kept by the adversary.

2. Then, these two states are sent over a quantum channel to some abstract challenger algorithm. This challenger selects at random one of the two states and traces out the other one. The selected state is encrypted

according to $|\mathsf{Enc}_k\rangle_{(2)}$ with a secret key k generated by the challenger, and sent back to the adversary.

3. Finally the adversary, upon receiving such encrypted state, has to guess which of the two states was selected.

More formally, we define the following.

Definition 5.11 (Quantum IND Adversary)**.** *Let $\mathcal{E}$ be a SKES with plaintext space $\mathcal{X}$ and ciphertext space $\mathcal{Y}$. A* quantum IND (qIND, or QIND) adversary $\mathcal{A}$ for *$\mathcal{E}$ is a pair of* QPT *algorithms $\mathcal{A} := (\mathcal{M}, \mathcal{D})$, where:*

1. *$\mathcal{M} :\rightarrow \mathfrak{D}(\mathfrak{H}_{\mathcal{X}}) \times \mathfrak{D}(\mathfrak{H}_{\mathcal{X}}) \times \mathfrak{D}(\mathfrak{H}_{\mathsf{Env}})$ is the* qIND (or QIND) message generator*;*

2. *$\mathcal{D} : \mathfrak{D}(\mathfrak{H}_{\mathcal{Y}}) \times \mathfrak{D}(\mathfrak{H}_{\mathsf{Env}}) \rightarrow \{0, 1\}$ is the* qIND (or QIND) distinguisher,

where $\mathfrak{H}_{\mathsf{com}}$ is a Hilbert space of appropriate dimension, modeling the state communication register (or, environment*) between $\mathcal{M}$ and $\mathcal{D}$.*

Experiment 5.12 ($\mathsf{Game}^{\mathsf{qIND}}_{\mathcal{E},\mathcal{A}}$)**.** *Let $\mathcal{E}$ be a SKES, and $\mathcal{A} := (\mathcal{M}, \mathcal{D})$ a qIND adversary. The* qIND experiment *proceeds as follows:*

```
Input: $n \in \mathbb{N}$
$k \leftarrow \mathsf{KGen}$
$(\varphi^0, \varphi^1, \sigma) \leftarrow \mathcal{M}$
$b \overset{\$}{\leftarrow} \{0, 1\}$
$\psi \leftarrow |\mathsf{Enc}_k\rangle_{(2)}(\varphi^b)$
trace out $\varphi^{1-b}$
$b' \leftarrow \mathcal{D}(\psi, \sigma)$
if $b = b'$ then
    Output: 1
else
    Output: 0
```

The advantage of $\mathcal{A}$ *is defined as:*

$$\mathsf{Adv}^{\mathsf{qIND}}_{\mathcal{E},\mathcal{A}} := \Pr\left[\mathsf{Game}^{\mathsf{qIND}}_{\mathcal{E},\mathcal{A}} \rightarrow 1\right] - \frac{1}{2}.$$

Definition 5.13 (Quantum Indistinguishability of Ciphertexts (qIND))**.** *A SKES $\mathcal{E}$ has* quantum indistinguishable encryptions (or, it is qIND secure) *iff, for any qIND adversary $\mathcal{A}$ it holds that: $\mathsf{Adv}^{\mathsf{qIND}}_{\mathcal{E},\mathcal{A}} \leq \mathsf{negl}$.*

We can strengthen this security notion by adding quantum CPA capabilities to the adversary.

Experiment 5.14 ($\mathsf{Game}^{\mathsf{qIND-qCPA}}_{\mathcal{E},\mathcal{A}}$)**.** *Let $\mathcal{E}$ be a SKES, and $\mathcal{A} := (\mathcal{M}, \mathcal{D})$ a qIND adversary. The* qIND-qCPA experiment *proceeds as follows:*

```
Input: $n \in \mathbb{N}$
$k \leftarrow \mathsf{KGen}$
$(\varphi^0, \varphi^1, \sigma) \leftarrow \mathcal{M}^{|\mathsf{Enc}_k\rangle_{(2)}}$
$b \xleftarrow{\$} \{0,1\}$
$\psi \leftarrow |\mathsf{Enc}_k\rangle_{(2)}(\varphi^b)$
trace out $\varphi^{1-b}$
$b' \leftarrow \mathcal{D}^{|\mathsf{Enc}_k\rangle_{(2)}}(\psi, \sigma)$
if $b = b'$ then
    Output: $1$
else
    Output: $0$
```

The advantage of $\mathcal{A}$ *is defined as:*

$$\mathsf{Adv}_{\mathcal{E},\mathcal{A}}^{\mathsf{qIND-qCPA}} := \Pr\left[\mathsf{Game}_{\mathcal{E},\mathcal{A}}^{\mathsf{qIND-qCPA}} \to 1\right] - \frac{1}{2}.$$

Definition 5.15 (Quantum Indistinguishability of Ciphertexts Under Quantum Chosen Plaintext Attack (qIND-qCPA))**.** *A SKES $\mathcal{E}$ has* quantum indistinguishable encryptions under quantum chosen plaintext attack (or, it is qIND-qCPA secure) *iff, for any qIND adversary $\mathcal{A} \Rightarrow \mathsf{Adv}_{\mathcal{E},\mathcal{A}}^{\mathsf{qIND-qCPA}} \leq \mathsf{negl}$.*

Clearly, qIND-qCPA is at least as strong as IND-qCPA, because a classical IND query is a special case of a quantum IND query.

Theorem 5.16 (qIND-qCPA $\implies$ IND-qCPA)**.** *If a SKES is qIND-qCPA secure, then it is also IND-qCPA secure.*

However, as we will show later, the converse is not necessarily true.

Corollary 5.17 (of Theorem 5.39 and Corollary 5.28)**.** *There exist SKES which are IND-qCPA secure, but not qIND-qCPA secure.*

In particular, Construction 3.26 (which is IND-qCPA secure according to Theorem 5.10) is not qIND-qCPA secure, because it is covered in the impossibility result from Section 5.3. However, [GHS16] shows how to build qIND-qCPA secure SKES from qPRPs.

Construction 5.18 ([GHS16, Construction 6.4])**.** *Let $(\mathcal{P}, \mathcal{P}^{-1})$ be a qPRP over $\mathcal{X} \times \mathcal{R}$ with key space $\mathcal{K}$, where $\mathcal{X}$ and $\mathcal{R}$ are both of size superpolynomial in n. Define $\mathcal{E} = \mathcal{E}_{\mathcal{K},\mathcal{X},\mathcal{X}\times\mathcal{R}} := (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ as a SKES with key space $\mathcal{K}$, plaintext space $\mathcal{X}$, and ciphertext space $\mathcal{X} \times \mathcal{R}$, in the following way:*

1. $\mathsf{KGen} \to k$*, with* $k \xleftarrow{\$} \mathcal{K}$*;*
2. $\mathsf{Enc}_k(x) \to \mathcal{P}_k(x \| r)$*, where* $r \xleftarrow{\$} \mathcal{R}$*;*
3. $\mathsf{Dec}_k(y) := \mathcal{P}^{-1}(y)\Big|_{\mathcal{X}}$.

Instead of proving the qIND-qCPA security of this construction directly, we prove it instead for another construction which *generalizes* it. Construction 5.18 has the drawback that the message length is upper bounded by the input length of the qPRP (minus the bit length of the randomness). However, like in the case of block ciphers, we can overcome this issue with a *mode of operation*. More specifically, we can handle arbitrary message lengths by splitting the message into blocks of a fixed length and applying the encryption algorithm of Construction 5.18 independently to each message block (using the same key but new randomness for each block). This procedure is akin to a 'randomized ECB mode', in the sense that each message block is processed separately, like in the ECB (Electronic Code Book) mode, but in our case the underlying cipher is inherently randomized (since we use fresh randomness for each block), so we can still achieve qCPA security. For simplicity we consider only message lengths which are multiples of the chosen blocksize. The construction can be generalized to arbitrary message lengths using standard padding techniques. Moreover, the randomness for every block can be generated efficiently using a single random seed and a pqPRNG.

Construction 5.19 ([GHS16, Construction 6.6])**.** *Let* $(\mathcal{P}, \mathcal{P}^{-1})$ *be a qPRP over* $\mathcal{X} \times \mathcal{R}$ *with key space* $\mathcal{K}$*, where* $\mathcal{X}$ *and* $\mathcal{R}$ *are both of size superpolynomial in* n*. For a polynomial function* ℓ*, let* $\mathcal{M} := \mathcal{X}^\ell$ *and* $\mathcal{C} := (\mathcal{X} \times \mathcal{R})^\ell$*. Define* $\mathcal{E} = \mathcal{E}_{\mathcal{K},\mathcal{M},\mathcal{C}} := (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ *as a SKES with key space* $\mathcal{K}$*, plaintext space* $\mathcal{M}$*, and ciphertext space* $\mathcal{C}$*, in the following way:*

1. $\mathsf{KGen} \to k$*, with* $k \overset{\$}{\leftarrow} \mathcal{K}$*;*
2. $\mathsf{Enc}_k(x_1 \| \dots \| x_\ell) \to \mathcal{P}_k(x_1 \| r_1) \| \dots \| \mathcal{P}_k(x_\ell \| r_\ell)$*,*
   where $r_i \overset{\$}{\leftarrow} \mathcal{R}$*,* $\forall\, i = 1, \dots, \ell$*;*
3. $\mathsf{Dec}_k(y_1 \| \dots y_\ell) := \mathcal{P}^{-1}(y_1)\Big|_{\mathcal{X}} \| \dots \| \mathcal{P}^{-1}(y_\ell)\Big|_{\mathcal{X}}$*.*

Before proving the security of Construction 5.19, we need a technical lemma. Let us assume w.l.o.g. that $\mathcal{X} = \{0,1\}^{m(n)}, \mathcal{R} = \{0,1\}^{\imath(n)}$ for polynomial functions m and $\imath$, so that $\mathcal{C} = \{0,1\}^{\ell \cdot (m+\imath)}$.

Lemma 5.20 ([GHS16, Lemma 6.7])**.** *Let* Φ *be the quantum channel that takes as input an arbitrary* m*-qubit state, attaches other* $\imath$ *qubits in state* $|0\rangle$*, and then applies a permutation picked uniformly at random from* $S(\{0,1\}^{m+\imath})$ *to the computational basis space. Let* Ψ *be the constant quantum channel which maps any* m*-qubit state to the totally mixed state* $\tau := \frac{\mathbb{I}}{2^{m+\imath}}$ *on* $m + \imath$ *qubits. Then,* $\|\Phi - \Psi\|_\diamond \leq 2^{-\imath+2}$*.*

Proof. In order to consider the fact that the m-qubit input state might be entangled with something else, we have to start with a purification of such a state. This is a bipartite pure $2m$-qubit state $|\varphi\rangle_{XY} = \sum_{x,y} a_{x,y} |x\rangle_X |y\rangle_Y$

whose m-qubit Y register is input into the channel and gets transformed into $\mathbb{I}_X \otimes \Phi(|\varphi\rangle\langle\varphi|) = \operatorname{tr}_Z |\psi\rangle\langle\psi|$, where:

$$|\psi\rangle := \sum_{x\in\{0,1\}^m, y\in\{0,1\}^m, p\in S(\{0,1\}^{m+\mathfrak{t}})} a_{x,y} |x\rangle_X |p(y\|0\dots0)\rangle_C |\pi\rangle_Z .$$

By definition of the diamond norm, we have to show that for any $2m$-qubit state ρ, we have that $\|(\mathbb{I}\otimes\Phi)(\rho)-(\mathbb{I}\otimes\Psi)(\rho)\|_{\mathrm{tr}} \leq 2^{-\mathfrak{t}+2}$. Due to the convexity of the trace distance, we may assume that $\rho = |\varphi\rangle\langle\varphi|$ is pure with $|\varphi\rangle_{XY} = \sum_{x,y} a_{x,y} |x\rangle_X |y\rangle_Y$. Hence, we obtain:

$$\begin{aligned}
(\mathbb{I}_X \otimes \Phi)(|\varphi\rangle\langle\varphi|) &= \operatorname{tr}_Z |\psi\rangle\langle\psi| \\
&= \frac{1}{2^{m+\mathfrak{t}}!} \sum_{x,x',y,y',p} a_{x,y}\overline{a_{x',y'}}\, |x\rangle\langle x|\, x'_X \otimes |p(y\|0\dots0)\rangle\, \langle p(y'\|0\dots0)|_C \\
&= \frac{1}{2^{m+\mathfrak{t}}!} \sum_{x,x',y} a_{x,y}\overline{a_{x',y}}\, |x\rangle\langle x|\, x'_X \otimes \sum_p |p(y\|0\dots0)\rangle\, \langle p(y\|0\dots0)|_C \\
&\quad + \frac{1}{2^{m+\mathfrak{t}}!} \sum_{x,x',y\neq y'} a_{x,y}\overline{a_{x',y'}}\, |x\rangle\langle x|\, x'_X \otimes \sum_p |p(y\|0\dots0)\rangle\, \langle p(y'\|0\dots0)|_C \\
&= \sum_{x,x',y} a_{x,y}\overline{a_{x',y}}\, |x\rangle\langle x|\, x'_X \otimes \frac{1}{2^{m+\mathfrak{t}}} \sum_z |z\rangle\langle z|\, z_C \\
&\quad + \sum_{x,x',y\neq y'} a_{x,y}\overline{a_{x',y'}}\, |x\rangle\langle x|\, x'_X \otimes \frac{1}{2^{m+\mathfrak{t}}(2^{m+\mathfrak{t}}-1)} \sum_{z\neq z'} |z\rangle\langle z|\, z'_C \\
&= \operatorname{tr}_Y |\varphi\rangle\langle\varphi| \otimes \tau_C + \chi_{XC} \\
&= (\mathbb{I}_X \otimes \Psi)(|\varphi\rangle\langle\varphi|) + \chi_{XC},
\end{aligned}$$

where we defined the 'difference state':

$$\chi_{XC} := \sum_{x,x',y\neq y'} a_{x,y}\overline{a_{x',y'}}\, |x\rangle\langle x'|_X \otimes \frac{1}{2^{m+\mathfrak{t}}(2^{m+\mathfrak{t}}-1)} \sum_{z\neq z'} |z\rangle\langle z'|_C .$$

In order to conclude, it remains to show that $\|\chi_{XC}\|_{\mathrm{tr}} \leq 2^{-\mathfrak{t}+2}$. For the C-register $\chi_C = \frac{1}{2^{m+\mathfrak{t}}(2^{m+\mathfrak{t}}-1)} \sum_{z\neq z'} |z\rangle\langle z'|_C$, one can verify that the $2^{m+\mathfrak{t}}$ eigenvalues are $(\lambda\cdot(2^{m+\mathfrak{t}}-1), -\lambda, -\lambda, \dots, -\lambda)$ where $\lambda := \frac{1}{2^{m+\mathfrak{t}}(2^{m+\mathfrak{t}}-1)}$. Hence, the trace norm (which is the sum of the absolute eigenvalues) is exactly $\lambda\cdot 2(2^{m+\mathfrak{t}}-1) = 2^{-m-\mathfrak{t}+1}$.

For the X-register, we split χ_X into two parts $\chi_X = \xi_X - \xi'_X$ where:

$$\begin{aligned}
\xi_X &:= \sum_{x,x'} |x\rangle\langle x'| \sum_{y,y'} a_{x,y}\overline{a_{x',y'}}; \\
\xi'_X &:= \sum_{x,x'} |x\rangle\langle x'| \sum_{y} a_{x,y}\overline{a_{x',y}},
\end{aligned}$$

and use the triangle inequality for the trace norm $\|\chi_X\|_{\mathrm{tr}} = \|\xi_X - \xi'_X\|_{\mathrm{tr}} \leq \|\xi_X\|_{\mathrm{tr}} + \|\xi'_X\|_{\mathrm{tr}}$. Observe that $\|\xi_X\|_{\mathrm{tr}} = \|\sum_{x,y} a_{x,y} |x\rangle \sum_{x',y'} \overline{a_{x',y'}} \langle x'|\,\|_{\mathrm{tr}} =$

$\| |s\rangle\langle s| \|_{\mathrm{tr}}$ for the (non-normalized) vector $|s\rangle := \sum_{x,y} a_{x,y} |x\rangle$. Hence, the trace norm $\|\xi_X\|_{\mathrm{tr}} = |\langle s|s\rangle| = \sum_x |\sum_y a_{x,y}|^2 \leq \sum_x \sum_y |a_{x,y}|^2 \cdot 2^m = 2^m$ by the Cauchy-Schwarz inequality and the normalization of the $a_{x,y}$'s. Furthermore, we note that ξ'_X is exactly the reduced density matrix of $|\varphi\rangle_{XY}$ after tracing out the Y register. Hence, ξ'_X is positive semi-definite and its trace norm is equal to its trace which is 1. In summary, we have shown that:

$$\begin{aligned}\|\chi_{XC}\|_{\mathrm{tr}} &= \|\chi_X\|_{\mathrm{tr}} \cdot \|\chi_C\|_{\mathrm{tr}} \leq (\|\xi_X - \xi'_X\|_{\mathrm{tr}}) \cdot 2^{-m-\imath+1} \\ &\leq (\|\xi_X\|_{\mathrm{tr}} + \|\xi'_X\|_{\mathrm{tr}}) \cdot 2^{-m-\imath+1} \leq (2^m + 1) \cdot 2^{-m-\imath+1} \leq 2^{-\imath+2}.\end{aligned}$$

□

If we consider a slightly different encryption channel Φ^T which still maps m qubits to $m + \imath$ qubits but where the permutation p is not picked uniformly from the whole set $S(\{0,1\}^{m+\imath})$, but instead we are guaranteed that a certain subset $T \subset \{0,1\}^{m+\imath}$ of outputs never occurs in these permutations, we can see such permutations as picked uniformly at random from a smaller set $S(\{0,1\}^{m+\imath} \setminus T)$. In this setting, we are interested in the distance of the channel (modeling the encryption operation) Φ^T from the slightly different constant channel Ψ^T which maps all inputs to the $(m + \imath)$-qubit state τ^T which is completely mixed on the smaller set $\{0,1\}^{m+\imath} \setminus T$ of basis elements. The set T represents 'forbidden' values that the encryption algorithm does never produce if we assume certain conditions on the randomness used. This technique will be used in the proof of the next theorem. By modifying slightly the proof of Lemma 5.20 we get the following.

Corollary 5.21 ([GHS16, Corollary 6.8])**.** *Let Φ^T, Ψ^T be quantum channels described as above. Then:*

$$\|\Phi^T - \Psi^T\|_\diamond \leq \frac{4}{2^\imath - |T|/2^m}. \tag{5.2}$$

We can now prove the qIND-qCPA security of Construction 5.19.

Theorem 5.22 ([GHS16, Theorem 6.9])**.** *Let $\mathcal{E}$ be the SKES from Construction 5.19 implemented through a (weak) qPRP family $(\mathcal{P}, \mathcal{P}^{-1})$. Then $\mathcal{E}$ in a qIND-qCPA SKES.*

Proof. We want to show that no QPT adversary can win the qIND-qCPA game with probability substantially better than guessing. We first transform the game through a short game-hopping sequence into a computationally equivalent game for which we can bound the success probability of the quantum distinguisher $\mathcal{D}$.

Game$_0$: this is the original qIND-qCPA game.

Game₁ : this is like Game_0, but instead of using a permutation drawn from the qPRP family $\mathcal{P}$, a random permutation $p \in S(\{0,1\}^{m+\tau})$ is chosen from the set of all permutations over $\{0,1\}^{m+\tau}$. The difference in the success probability of $\mathcal{D}$ winning one or the other of these two games is negligible, otherwise, we could use $\mathcal{D}$ to distinguish a random permutation drawn from $\mathcal{P}$ from one drawn from $S(\{0,1\}^{m+\tau})$. This would contradict the assumption that $\mathcal{P}$ is a qPRP.

Game₂ : this is like Game_1, but $\mathcal{D}$ is guaranteed that the randomness used for each encryption query are ℓ new random τ-bit strings that were not used before. In other words, the challenger keeps track of all random values used so far and excludes those when sampling a new randomness. Since in Game_1 the same randomness is sampled twice only with negligible probability, the probabilities of winning these two games differ at most negligibly.

Game₃ : this is like Game_2, except that the answer to each query asked by $\mathcal{D}$ also contains the randomness $r_1, \dots, r_\ell$ used by the challenger for answering that query. Clearly, $\mathcal{D}$'s probability of winning this game is at least the probability of winning Game_2.

When Game_3 starts, the qIND message generator $\mathcal{M}$ (where $\mathcal{A} = (\mathcal{M}, \mathcal{D})$ is the qIND adversary as in Definition 5.11) chooses two different plaintext states. One of them is chosen at random and sent back encrypted with fresh randomness values $\hat{r}_1, \dots, \hat{r}_\ell$. Let Q denote the set of $q \cdot \ell = \mathsf{poly}(n)$ query values used during the previous q queries to $|\mathsf{Enc}_k\rangle$ in the first learning qCPA-phase. We have to consider that from this phase, $\mathcal{D}$ knows a set $T \subset \{0,1\}^{m+\tau}$ of 'taken' outputs (ciphertexts), i.e., he knows that any $p(x \| \hat{r}_i)$ will not take one of these values, as $\hat{r}_i$ has not been used before. So, from the adversary's point of view, p is a permutation randomly chosen from S', the set of those permutations over $\{0,1\}^{m+\tau}$ that fix these $|T|$ values. In order to simplify the proof, we will consider a very conservative bound where $|T| = q \cdot \ell \cdot 2^m$, and the size of S' is $|S'| = (2^{m+\tau} - |T|)!$. Notice that this bound is very conservative because it assumes that the adversary learns 2^m different (classical) ciphertexts for each one of the $q \cdot \ell$ 'taken' randomness values but, as we will see, this knowledge is still insufficient to win the game.

By construction, the encryption of an $(\ell \cdot m)$-qubit (possibly mixed) state ρ is performed in ℓ separate blocks of m qubits each. We are guaranteed that fresh randomness is used in each block, hence it follows from Corollary 5.21 that $\mathsf{Enc}_k(\rho)$ is negligibly close to the ciphertext state where the first $m + \tau$ qubits are replaced with the completely mixed state (by noting that $\frac{|T|}{2^m} = m \cdot q$ is polynomial in n in our case, and hence the right-hand side of (5.2) is negligible). Another application of Corollary 5.21 gives negligible distance to the ciphertext state where the first $2(m + \tau)$ qubits are replaced with the completely mixed state, etc. After ℓ applications of Corollary 5.21, we have shown that $\mathsf{Enc}_k(\rho)$ is negligibly close to the totally mixed state on $\ell(m + \imath)$ qubits. As this argument can be made for any plaintext state ρ, we have shown that, from $\mathcal{D}$'s point of view, all encrypted states have negligible distance from the totally mixed state, and therefore cannot be distinguished. This holds regardless of any additional query during the second qCPA phase, because a polynomial number of such queries cannot change this distance by more than a negligible amount. □

Corollary 5.23 ([GHS16, Theorem 6.9])**.** *Let $\mathcal{E}$ be the SKES from Construction 5.18 implemented through a (weak) qPRP family $(\mathcal{P}, \mathcal{P}^{-1})$. Then $\mathcal{E}$ in a qIND-qCPA SKES.*

Notice how the security of Constructios 5.18 and 5.19 does not require *strong* qPRPs. The reason is that, even if we are considering type-(2) transformations (which could be used to compute p^{-1}), these transformations are never implemented directly by the adversary, but only evaluated as oracles. And since we only consider CPA quantum oracles here, and not CCA, the adversary is never granted access to the decryption oracle. Hence, p^{-1} is not needed by the reduction. However, extending the constructions to CCA1 security *would* require strong qPRPs.

Weak Quantum Indistinguishability

Before providing further results related to the qIND notion, we introduce here a slight relaxation of qIND which might be of use in certain contexts which we explain in this section. The idea is to restrict the power of the adversary in the qIND notion, by only allowing quantum states of a certain form for the qIND challenge phase. This notion was originally introduced in [GHS16] as ‘qIND’ but, as already mentioned at the beginning of this section, we relabel it as ‘wqIND’ (where ‘w’ stands for ‘weak’) for consistency with our framework.

We start by defining the ‘restricted’ quantum states which can be used by the adversary in the new security notion.

Definition 5.24 (Classical Description of Quantum States)**.** *A* classical description *of a quantum state ρ is a (classical) bit string $\mathsf{Dsc}(\rho)$ describing a quantum circuit which (takes no input but starts from a fixed initial state $|0\rangle$ and) outputs ρ.*

We deviate here from the traditional meaning of ‘classical description’ referring to individual numerical entries of the density matrix. The reason is that Definition 5.24 also covers the cases where those numerical entries are not easily computable, as long as we can give an explicit constructive procedure for that state. Clearly, every pure quantum state $|\varphi\rangle$ has a classical description given by a description of the quantum circuit which implements the unitary that maps $|0\rangle$ to $|\varphi\rangle$. The classical description of a mixed state ρ_A is given by the circuit which first creates a purification $|\varphi\rangle_{AR}$ of ρ_A and then only outputs the A register. Note that a state admitting a classical description cannot be entangled with any other system. We say that a state has an *efficient classical representation* if it has a classical representation, and such representation has a bit size at most polynomial in some security parameter n. In this case, we assume the existence of a (fixed, public, canonical) QPT algorithm Qbuild which, given as input a classical description of a quantum state, outputs that state, i.e., $\mathsf{Qbuild}(\mathsf{Dsc}(\rho)) \to \rho$ (the notation for the output is probabilistic, because ρ could be a mixed state, i.e., a distribution on pure states).

In classical models, there is no difference between sending a description of a message or the message itself. In the quantum world, there is a big difference between these two cases, as the latter allows the adversary $\mathcal{A}$ to establish entanglement of the message(s) with other registers. This is not possible when using classical descriptions. It might intuitively appear that the more general model considered for the qIND notion is more natural. However, the above scenario models the case where $\mathcal{A}$ is well aware of the message that is encrypted, but the message is not constructed by $\mathcal{A}$ himself. Giving $\mathcal{A}$ the ability to choose the challenge messages for the qIND game models the worst case that might happen: $\mathcal{A}$ knows that the ciphertext he receives is the encryption of one out of the two messages that he can distinguish best. This closely reflects the intuition behind the classical IND notion: in that game, the adversary is allowed to send the two messages not because in the real world he would be allowed to do so, but because we want to achieve security even for the best possible choice of messages from the adversary's perspective. Hence, the model using classical descriptions of quantum states is a valid alternative.

Experiment 5.25 ($\mathsf{Game}^{\mathsf{wqIND-qCPA}}_{\mathcal{E},\mathcal{A}}$)**.** *Let $\mathcal{E}$ be a SKES, and $\mathcal{A} := (\mathcal{M}, \mathcal{D})$ a qIND adversary. The* wqIND-qCPA experiment *proceeds as follows:*

```
Input: $n \in \mathbb{N}$
$k \leftarrow \mathsf{KGen}$
$(\mathsf{Dsc}(\varphi^0), \mathsf{Dsc}(\varphi^1), \sigma) \leftarrow \mathcal{M}^{|\mathsf{Enc}_k\rangle_{(2)}}$
$b \xleftarrow{\$} \{0,1\}$
$\varphi^b \leftarrow \mathsf{Qbuild}(\mathsf{Dsc}(\varphi^b))$
$\psi \leftarrow |\mathsf{Enc}_k\rangle_{(2)}(\varphi^b)$
$b' \leftarrow \mathcal{D}^{|\mathsf{Enc}_k\rangle_{(2)}}(\psi, \sigma)$
if $b = b'$ then
    Output: 1
else
    Output: 0
```

The advantage of $\mathcal{A}$ *is defined as:*

$$\mathsf{Adv}^{\mathsf{wqIND-qCPA}}_{\mathcal{E},\mathcal{A}} := \Pr\left[\mathsf{Game}^{\mathsf{wqIND-qCPA}}_{\mathcal{E},\mathcal{A}} \to 1\right] - \frac{1}{2}.$$

Definition 5.26 (Weak Quantum Indistinguishability of Ciphertexts Under Quantum Chosen Plaintext Attack (wqIND-qCPA))**.** *A SKES $\mathcal{E}$ has* weakly quantum indistinguishable encryptions under quantum chosen plaintext attack (or, it is wqIND-qCPA secure) *iff, for any qIND adversary $\mathcal{A}$ it holds:* $\mathsf{Adv}^{\mathsf{wqIND-qCPA}}_{\mathcal{E},\mathcal{A}} \leq \mathsf{negl}$.

Clearly, qIND-qCPA is at least as strong as wqIND-qCPA, because quantum states admitting an efficient classical description (used in wqIND) are just a special case of arbitrary quantum plaintext states (used in qIND).

Theorem 5.27 ([GHS16, Theorem 3.3])**.** *If a SKES is qIND-qCPA secure, then it is also wqIND-qCPA secure.*

Corollary 5.28 (qIND $\implies$ wqIND)**.** *If a SKES is qIND secure, then it is also wqIND secure.*

Finding a separation between wqIND and qIND is an open problem, as explained in [GHS16]. Morally, the notion wqIND-qCPA should lie somewhere between IND-qCPA and qIND-qCPA, because it covers indistinguishability for messages which are not necessarily classical, but not arbitrarily quantum. The reason for considering the seemingly artificial wqIND is that in the context of classical encryption schemes resistant to superposition quantum access, it is important to not lose focus of what the capabilities of a 'reasonable' adversary should be. Namely, recall the following classical IND argument: *'allowing the adversary to send plaintexts to the challenger is equivalent to the fact that indistinguishability must hold even for the most favorable case from the adversary's perspective'.* Such an argument does *not* hold anymore quantumly. In fact, the qIND model presents the following issues:

1. it allows entanglement between the adversary and the IND challenger: $\mathcal{A}$ could prepare a state of the form $\rho_{AB} = \frac{1}{\sqrt{2}} |00\rangle + \frac{1}{\sqrt{2}} |11\rangle$, sending ρ_A as a plaintext but keeping ρ_B; and

2. it allows the adversary to create certain non-reproduceable states. For example, consider the state $|\psi\rangle = \sum_{x \in \mathcal{X}} \frac{1}{\sqrt{|\mathcal{X}|}} |x, \hbar(x)\rangle$, where $\hbar$ is a collision-resistant hash function. $\mathcal{A}$ could measure the second register, obtaining a random outcome y, and knowing therefore that the remaining state is the superposition of the preimages of y, i.e.:

$$|\psi_y\rangle = \sum_{x \in \mathcal{X} : \hbar(x) = y} \frac{1}{\sqrt{|\{x \in \mathcal{X} : \hbar(x) = y\}|}} |x\rangle .$$

$\mathcal{A}$ could then use $|\psi_y\rangle$ as a plaintext in the challenge phase, but note that $\mathcal{A}$ cannot reproduce $|\psi_y\rangle$ for a given value y.

Both of the above examples highlight adversary capabilities which might be considered unreasonably strong in certain scenarios. Entanglement between $\mathcal{A}$ and the IND challenger $\mathcal{C}$ represents a sort of 'quantum watermarking' of messages, which goes beyond what a meaningful notion of indistinguishability should achieve. Knowledge of intermediate, unpredictable measurements also renders $\mathcal{A}$ too powerful, because it gives $\mathcal{A}$ access to information not available to $\mathcal{C}$ itself; e.g., in the example above $\mathcal{C}$ would not even know the value of y. As it is $\mathcal{C}$ who prepares the state to be encrypted by running Qbuild, it is reasonable to assume that it is $\mathcal{C}$ who should know these intermediate measurements, not $\mathcal{A}$. In the example above, what $\mathcal{A}$ could see instead (provided he knows the circuit generating the state, as we assume in wqIND) is that the plaintext is a mixture $\Psi = \sum_y \psi_y$ for all possible values of y.

The possibility offered by qIND of allowing the adversary to play the IND game with arbitrary states is certainly elegant from a theoretical point of view, but from the perspective of the quantum security of the kind of schemes we are considering, it is sometimes useful to consider the restricted notion wqIND, because it inherently provides guidelines and reasonable limitations on what a quantum adversary can or cannot do. Also, wqIND is often easier to deal with: notice that in such a model, unlike in the qIND model, $\mathcal{A}$ always receives back an unentangled state from a challenge query. In security reductions, this means that we can more easily simulate the challenger, and that we do not have to take care of measures of entanglement when analyzing the properties of quantum states - for example, indistinguishability of states can be shown by only resorting to the *trace norm* instead of the more general *diamond norm* as in the proof of Theorem 5.23.

Finally, it is important to notice that it is actually unclear whether a separation between qIND and wqIND can be found at all in the realm of classical encryption schemes. In fact, all the positive results present in [GHS16] hold for the more general qIND notion, while the impossibility result we present in Section 5.3 holds for both qIND and wqIND.

Quantum Semantic Security

In this section, we discuss notions of semantic security in **QS**2. All of them have been presented before in [GHS16]. We start by defining a semantic security equivalent of IND-qCPA, called *SEM-qCPA*. This is just the usual notion of SEM, augmented by giving to the adversary qCPA capabilities. In order to not overload notation, we refer to 'adversary' and 'simulator' simply as QPT versions of the PPT algorithms from Definition 3.14.

Definition 5.29 ([GHS16, Definition 4.1])**.** *A SKES $\mathcal{E}$ is* semantically secure under quantum chosen plaintext attack (or, it is SEM-qCPA secure) *iff, for any* QPT *adversary $\mathcal{A}$ there exists a* QPT *simulator $\mathcal{S}$ such that, for every efficiently computable $f, \hbar : \{0,1\}^* \to \{0,1\}^*$ polynomially bounded in the* *input bit size, for every probability ensemble* $\mathcal{M} := (\mathcal{M}_n)_n$*, where* $\mathcal{M}_n$ *are probability distributions over* $\mathcal{X}_n$ *with* $|\mathcal{M}_n| = \mathsf{poly}(n)$*, such that:*

$$\left|\Pr\left[\mathsf{Game}^{\mathsf{SEM}}_{\mathcal{E},\mathcal{A}^{|\mathsf{Enc}_k\rangle}}(\mathcal{M}, f, \hbar) \to 1\right] - \Pr\left[\mathsf{Game}^{\mathsf{SEM}*}_{\mathcal{E},\mathcal{S}^{|\mathsf{Enc}_k\rangle}}(\mathcal{M}, f, \hbar) \to 1\right]\right| \leq \mathsf{negl},$$

where $k \leftarrow \mathsf{KGen}$ *is the secret key generated during the experiments, and the probabilities are taken over the randomness of* $\mathcal{A}, \mathcal{E}, \mathcal{M}, \mathcal{S}$*.*

Unsurprisingly, the above notion is equivalent to IND-qCPA. The proof is a straightforward modification of Theorem 3.21 by also accounting for the quantum CPA queries.

Theorem 5.30 ([GHS16, Theorem 5.1])**.** *A SKES is IND-qCPA secure iff it is SEM-qCPA secure.*

We might ask what happens if the above definition is strenghtened by providing the adversary (and the simulator) *quantum* advice, instead of a classical advice $\hbar(x)$ for some plaintext x. The following two cases appear.

- We might replace the classical function $\hbar$ with a unitary operator U which, acting on a basis element $|x\rangle$ for a (classical) plaintext x, produces a quantum advice state $|\xi\rangle$. The resulting security notion is called *quantum advice semantic security under quantum chosen plaintext attack (qaSEM-qCPA)* [GHS16, Definition D.1], and it turns out to be meaningless, because trivially achievable by *any* SKES. The reason is that a unitary U can always be inverted as $U^\dagger$ by both adversary *and* simulator. Both of them are then able to recover the plaintext given the quantum advice.

- To fix the above problem, we might allow more general quantum circuits $\mathcal{U}$ that can somehow provide non-reversible information, for example by applying some partial measurement at the end, or by providing $\mathcal{A}$ (resp. $\mathcal{S}$) only with some output qubits, while tracing out the others. Towards this end let $\mathcal{U}$ be an arbitrary quantum circuit (the *advice circuit*) that takes as input a basis element $|x\rangle$ and a quantum state ρ provided by $\mathcal{A}$ (resp. $\mathcal{S}$) (that includes possibly needed auxiliary registers), and computes a (possibly mixed) quantum advice state ξ. The resulting security notion is called *ideal quantum advice semantic security under quantum chosen plaintext attack (iqSEM-qCPA)* [GHS16, Definition D.2], and it turns out to be equivalent to IND-qCPA. The reason is that the proof in Theorem 5.30 only uses the advice function to transmit classical information, and therefore iqSEM-qCPA can be reduced to IND-qCPA.

It seems therefore that introducing quantum advice states is not meaningful as long as the messages are still classical. We proceed now instead to present a quantum security notion equivalent to the wqIND-qCPA notion. First of all, we redefine the meaning of quantum SEM adversary and simulator.

Definition 5.31 (Quantum SEM Adversary, Quantum SEM Simulator)**.** *Let $\mathcal{E} := \mathcal{E}_{\mathcal{K},\mathcal{X},\mathcal{Y}}$ be a SKES, and $\mathfrak{H}_f, \mathfrak{H}_\hbar$ two Hilbert spaces of appropriate dimension (exponential in the security parameter). A* quantum SEM adversary $\mathcal{A}$ for $\mathcal{E}$ *is a* QPT *algorithm $\mathcal{A} : \mathfrak{D}(\mathfrak{H}_\mathcal{Y}) \times \mathfrak{D}(\mathfrak{H}_\hbar) \to \mathfrak{D}(\mathfrak{H}_f)$. A* quantum SEM simulator $\mathcal{S}$ for $\mathcal{E}$ *is a* QPT *algorithm $\mathcal{S} : \mathfrak{D}(\mathfrak{H}_\hbar) \to \mathfrak{D}(\mathfrak{H}_f)$.*

The wqSEM notion is given by replacing classical functions f and $\hbar$ with quantum CPTP maps Σ, Ξ, which are quantum circuits taking as input m-qubit quantum states (where m is the bit size of plaintexts, polynomial in n) and outputting $\mathsf{poly}(m)$-qubit quantum states. The idea is that, since we are using quantum states with efficient classical representations, we can sample some classical randomness once, and reuse it with Qbuild to create many copies of the same plaintext state.

Experiment 5.32 ($\mathsf{Game}^{\mathsf{wqSEM}}_{\mathcal{E},\mathcal{A}}$)**.** *Let $\mathcal{E}$ be a SKES, and $\mathcal{A}$ a quantum SEM adversary. The* wqSEM experiment *proceeds as follows:*

```
Input: $n \in \mathbb{N}$, CPTP maps $\Sigma, \Xi$ with $m$-qubit input and $\mathsf{poly}(m)$-qubit output, $\mathcal{M} := (\mathcal{M}_n)_n$, where $\mathcal{M}_n$ are probability distributions over a family of randomness spaces $(\mathcal{R}_n)_n$ ,with $|\mathcal{M}_n| = \mathsf{poly}(n)$
$k \leftarrow \mathsf{KGen}$
$r \leftarrow \mathcal{M}_n$
$\varphi \leftarrow \mathsf{Qbuild}(r)$                ▷ Qbuild is invoked with randomness $r$
$\psi \leftarrow |\mathsf{Enc}_k\rangle_{(2)}(\varphi)$
$\varphi \leftarrow \mathsf{Qbuild}(r)$                ▷ a second copy of $\varphi$ is generated, using the same $r$
$\xi \leftarrow \Xi(\varphi)$                ▷ this is the quantum advice state
$\sigma \leftarrow \mathcal{A}(\psi, \xi)$
if $\sigma$ is computationally indistinguishable from $\Sigma(\varphi)$ then
    Output: 1
else
    Output: 0
```

We use 'computationally indistinguishable' as a shorthand for: 'for every QPT algorithm $\mathcal{D}$ with outputs in $\{0,1\}$(a quantum distinguisher), the probability that the output differs on the two states given as input is negligible'. As usual, a third copy of φ (to be processed by Σ) can be generated using the same randomness r and the Qbuild algorithm.

Experiment 5.33 ($\mathsf{Game}^{\mathsf{wqSEM}*}_{\mathcal{E},\mathcal{S}}$)**.** *Let $\mathcal{E}$ be a SKES, and $\mathcal{S}$ a quantum SEM simulator. The* simulated wqSEM experiment *proceeds as follows:*

```
Input: $n \in \mathbb{N}$, CPTP maps $\Sigma, \Xi$ with $m$-qubit input and $\mathsf{poly}(m)$-qubit output, $\mathcal{M} := (\mathcal{M}_n)_n$, where $\mathcal{M}_n$ are probability distributions over a family of randomness spaces $(\mathcal{R}_n)_n$ ,with $|\mathcal{M}_n| = \mathsf{poly}(n)$
```

```
$k \leftarrow \mathsf{KGen}$
$r \leftarrow \mathcal{M}_n$
$\varphi \leftarrow \mathsf{Qbuild}(r)$
$\xi \leftarrow \Xi(\varphi)$
$\sigma \leftarrow \mathcal{S}(\xi)$        ▷ $\mathcal{S}$ only gets the quantum advice, not the ciphertext
if $\sigma$ is computationally indistinguishable from $\Sigma(\varphi)$ then
    Output: 1
else
    Output: 0
```

Definition 5.34 (Weak Quantum Semantic Security (wqSEM))**.** *A SKES $\mathcal{E}$ is* weakly quantumly semantically secure (wqSEM) *iff, for any quantum SEM adversary $\mathcal{A}$ there exists a quantum SEM simulator $\mathcal{S}$ such that, for every CPTP maps Σ, Ξ with m-qubit input and $\mathsf{poly}(m)$-qubit output, for every probability ensemble $\mathcal{M} := (\mathcal{M}_n)_n$ with polynomial-size support over some randomness space, it holds:*

$$\left|\Pr\left[\mathsf{Game}^{\mathsf{wqSEM}}_{\mathcal{E},\mathcal{A}}(\mathcal{M}, \Sigma, \Xi) \rightarrow 1\right] - \Pr\left[\mathsf{Game}^{\mathsf{wqSEM}*}_{\mathcal{E},\mathcal{S}}(\mathcal{M}, \Sigma, \Xi) \rightarrow 1\right]\right| \leq \mathsf{negl},$$

where the probabilities are taken over the randomness of $\mathcal{A}, \mathcal{E}, \mathcal{M}, \mathcal{S}$.

Definition 5.35 (Weak Quantum Semantic Security Under Quantum Chosen Plaintext Attack (wqSEM-qCPA))**.** *A SKES $\mathcal{E}$ is* weakly quantumly semantically secure under quantum chosen plaintext attack (wqSEM-qCPA) *iff, for any quantum SEM adversary $\mathcal{A}$ there exists a quantum SEM simulator $\mathcal{S}$ such that, for every CPTP maps Σ, Ξ with m-qubit input and $\mathsf{poly}(m)$-qubit output, for every probability ensemble $\mathcal{M} := (\mathcal{M}_n)_n$ with polynomial-size support over some randomness space, it holds:*

$$\left|\Pr\left[\mathsf{Game}^{\mathsf{wqSEM}}_{\mathcal{E},\mathcal{A}^{|\mathsf{Enc}_k\rangle_{(2)}}}(\mathcal{M}, \Sigma, \Xi) \rightarrow 1\right] - \Pr\left[\mathsf{Game}^{\mathsf{wqSEM}*}_{\mathcal{E},\mathcal{S}^{|\mathsf{Enc}_k\rangle_{(2)}}}(\mathcal{M}, \Sigma, \Xi) \rightarrow 1\right]\right| \leq \mathsf{negl},$$

where the probabilities are taken over the randomness of $\mathcal{A}, \mathcal{E}, \mathcal{M}, \mathcal{S}$.

The resulting wqSEM-qCPA notion is equivalent to wqIND-qCPA.

Theorem 5.36 ([GHS16, Theorem 5.4])**.** *A SKES is wqIND-qCPA secure iff it is wqSEM-qCPA secure.*

Proof. The proof closely follows the one for Theorem 3.21, with some careful modifications. We prove the theorem by splitting it in two parts.

wqIND − qCPA $\Longrightarrow$ wqSEM − qCPA. Let $\mathcal{A}$ be an efficient quantum SEM adversary. We want to show that a quantum SEM simulator $\mathcal{S}$ exists, with roughly the same success probability as $\mathcal{A}$, by exploiting the wqIND-qCPA security of the encryption scheme. The idea of the proof is to hand $\mathcal{A}$'s circuit as non-uniform advice to the simulator $\mathcal{S}$. This is allowed, because $\mathcal{A}$

is a QPT adversary against the wqSEM-qCPA game, and hence $\mathcal{A}$'s circuit has a short classical representation. $\mathcal{S}$ can then build and run $\mathcal{A}$'s circuit, and simulate a qSEM-qCPA experiment for $\mathcal{A}$ by generating a new key and answering all of $\mathcal{A}$'s queries using this key. When $\mathcal{S}$ performs his 'real' wqSEM challenge query (using the challenge query generated by $\mathcal{A}$), he does not receive back a valid ciphertext. However, $\mathcal{S}$ can generate a bogus ciphertext by encrypting (with his own key) the $|1\dots1\rangle$ basis element of the same size as the original plaintext state. It follows from the indistinguishability of encryptions that $\mathcal{A}$'s success probability in this game must be negligibly close to its success probability with a real ciphertext, otherwise $\mathcal{A}$ would be an efficient distinguisher for the scheme $\mathcal{E}$.

wqSEM − qCPA $\implies$ wqIND − qCPA. Assume there exists an efficient wqIND-qCPA distinguisher $\mathcal{D}$ for the scheme $\mathcal{E}$. Then we show how to construct a QPT algorithm $\mathcal{A}$ that has oracle access to $\mathcal{D}$ and breaks the wqSEM-qCPA security of the scheme, in the sense that no simulator $\mathcal{S}$ can do better than $\mathcal{A}$. The construction works as follows: $\mathcal{A}$ starts the $\mathsf{Game}^{\mathsf{wqSEM}}_{\mathcal{E},\mathcal{A}}$ game, and then he runs $\mathcal{D}$, emulating the quantum encryption oracle by simply forwarding all the qCPA queries performed by $\mathcal{D}$ to its own oracle (the $|\mathsf{Enc}_k\rangle_{(2)}$ oracle of the wqSEM-qCPA game). When $\mathcal{D}$ executes the wqIND challenge query by sending classical descriptions of two states φ^0 and φ^1, $\mathcal{A}$ produces the wqSEM template $(\mathcal{M}, \Xi, \Sigma)$, with $\mathcal{M}$ such that $\mathsf{Qbuild}(r)$ outputs φ^0 for half of the possible values $r \leftarrow \mathcal{M}$ and φ^1 for the other half, Ξ is the constant map outputting $|1\dots1\rangle$, and Σ is the identity map $\Sigma(\rho) = \rho$. Then $\mathcal{A}$ performs a qSEM challenge query with this template. Given challenge ciphertext state $|\mathsf{Enc}_k\rangle_{(2)}(\varphi^b)$ (for $b \in \{0,1\}$), $\mathcal{A}$ forwards it as an answer to $\mathcal{D}$'s wqIND challenge query. As $\mathcal{D}$ distinguishes $|\mathsf{Enc}_k\rangle_{(2)}(\varphi^0)$ from $|\mathsf{Enc}_k\rangle_{(2)}(\varphi^1)$ with non-negligible success probability by assumption, $\mathcal{D}$ returns the correct value of b with non-negligible advantage over guessing. Then $\mathcal{A}$, having recorded a copy of the classical descriptions of φ^0 and φ^1, is able to create another copy of φ^b through Qbuild and compute the state $\Sigma(\varphi^b)$ exactly, and consequently win the wqSEM-qCPA game with non-negligible advantage. However, as Ξ generates the same (constant, useless) advice state $|1\dots1\rangle$ independently of the encrypted message, no simulator can do better than guessing the plaintext. This concludes the proof. □

In this work, we will not explicitly define a notion of quantum semantic security related to qIND. However, we will show in the next chapter how the qIND notion is equivalent to the quantum indistinguishability notion Q-IND (introduced in [BJ15]) for quantum encryption schemes, when these are obtained by implementing a classical SKES in unitary type-(2) mode. In [ABF$^+$16], on the other hand, notions of quantum semantic security are presented, which are proven to be equivalent to Q-IND, and therefore easily adaptable to the case of quantumly-accessible SKES that we consider here.

Impossibility Result

In this section we show how the qIND security notion cannot be achieved by a large class of SKESs: namely, all those schemes which do *not* substantially expand the message during encryption. First we formally define what it means for a cipher to expand or keep constant the message size by defining the *core function* of a SKES. Intuitively, the definition splits the ciphertext into the randomness and a part carrying the message-dependent information. This definition covers most encryption schemes in the literature.

Definition 5.37 (Core Function [GHS16, Definition 6.1])**.** *Let $\mathcal{E} = \mathcal{E}_{\mathcal{K},\mathcal{X},\mathcal{Y}}$ be a SKES, and let $\mathcal{R}$ be the randomness space of* Enc*. Let $f : \mathcal{K} \times \mathcal{R} \times \mathcal{X} \to \mathcal{Y}$ be a function such that:*

- *for all $k \in \mathcal{K}$ and for all $x \in \mathcal{X}$, $\mathsf{Enc}_k(x)$ can be written as $(r, f(k, r, x))$, where $r \in \mathcal{R}$ is independent of the message; and*
- *there exists a function g such that for all $k \in \mathcal{K}, r \in \mathcal{R}, x \in \mathcal{X}$ it holds: $g(k, r, f(k, x, r)) = x$.*

Then, we call f the core function *of the encryption scheme.*

For example, in case of Construction 3.26 (where $\mathsf{Enc}_k(x)$ is defined as $(r, \mathcal{F}_k(r) \oplus x)$ for a PRF $\mathcal{F}$) the core function would be $f(k, r, x) := \mathcal{F}_k(r) \oplus x$, with associated $g(k, r, z) := z \oplus \mathcal{F}_k(r)$.

Definition 5.38 (Quasi–Length-Preserving Encryption [GHS16, Definition 6.2])**.** *We call a SKES with core function f* quasi–length-preserving *iff:*

$$\forall\, x \in \mathcal{X},\, \forall\, r \in \mathcal{R},\, \forall\, k \in \mathcal{K} \implies |f(k, x, r)| = |x|,$$

i.e., the output of the core function has the same bit length as the plaintext.

For example, Construction 3.26 is quasi–length-preserving.

The crucial observation for our impossibility result is the following: for a quasi–length-preserving encryption scheme, the space of possible input and (core function) output bit strings (with respect to plaintext and ciphertext) coincide, therefore these ciphers act as permutations on these spaces. This means that, if we start with an input state which is a superposition of *all* the possible basis states, all of them with the *same* amplitude, this state will be left unmodified by the unitary type-(2) encryption operation (because such operator will just ‘shuffle’ in the space of computational basis-states amplitudes which are exactly the same).

Theorem 5.39 ([GHS16, Theorem 6.3])**.** *If a SKES is quasi–length-preserving, then it is not wqIND secure.*

Proof. Let $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ be a quasi–length-preserving scheme. We give an explicit, efficient distinguisher attack.

1. For m-bit message strings, the distinguisher $\mathcal{D}$ sets the two plaintext states for the qIND- game to be: $|\varphi^0\rangle = H|0^m\rangle, |\varphi_1\rangle = H|1^m\rangle$, where H is the m-fold tensor Hadamard transformation. Notice that both these states admit efficient classical representations, and are thus allowed in the wqIND game.
2. A random bit b is flipped, and the challenge ciphertext state $|\psi\rangle = |\mathsf{Enc}_k\rangle_{(2)} |\varphi^b\rangle$ is returned to $\mathcal{D}$.
3. $\mathcal{D}$ applies H to the core-function part of the ciphertext $|\psi\rangle$ and measures it in the computational basis. $\mathcal{D}$ outputs 0 iff the outcome is 0^m, and outputs 1 otherwise.

Notice that applying $|\mathsf{Enc}_k\rangle_{(2)}$ to $H|0^m\rangle$ leaves the state untouched: since the encryption oracle merely performs a permutation in the basis space, and since $|\varphi_0\rangle$ is a superposition of every basis element with the same amplitude, it follows that whenever b is equal to 0, the ciphertext state will be left unchanged. In this case, after applying the self-inverse transformation H again, $\mathcal{D}$ obtains measurement outcome 0^m with probability 1.

On the other hand, if $b = 1$, then $|\varphi^1\rangle = \frac{1}{2^{m/2}} \sum_y (-1)^{y \cdot 1^m} |y\rangle$ where $a \cdot b$ denotes the bitwise inner product between a and b. Hence, $|\varphi^1\rangle$ is a superposition of every basis element where (depending on the parity of y) half of the elements have a positive amplitude and the other half have a negative one, but all of them will be equal in absolute value. Applying $|\mathsf{Enc}_k\rangle_{(2)}$ to this state results in $\frac{1}{2^{m/2}} \sum_y (-1)^{y \cdot 1^m} |\mathsf{Enc}(y)\rangle$. After re-applying H, the amplitude of the basis state $|0^m\rangle$ becomes $\sum_y (-1)^{y \cdot 1^m + \mathsf{Enc}(y) \cdot 0^m} = \sum_y (-1)^{\|y\|}$ (where $\|y\|$ is the *Hamming weight of* y) which is 0. Hence, the probability for $\mathcal{D}$ of observing 0^m after the measurement is 0. This gives $\mathcal{D}$ a way of distinguishing between encryptions of the two plaintext states. □

Notice that the above attack works also against qIND, because of Theorem 5.27. In particular, Theorem 5.39 shows that Construction 3.26, which is IND-qCPA secure if the used PRF is quantum secure, does not fulfill qIND, nor wqIND. This attack is a consequence of the well-known fact [AMTdW00, BR03] that, in order to perfectly (information-theoretically) encrypt a single quantum bit, *two* bits of classical information are needed: one to hide the basis bit, and one to hide the phase (i.e., the signs of the amplitudes). The fact that we are restricted to quantum operations of the form $|\mathsf{Enc}_k\rangle_{(2)}$ (that is, quantum instantiations of classical encryptions) means that we cannot afford to hide the phase as well, and this restriction allows for an easy distinguishing procedure in the case of a quasi–length-preserving SKES.

Summing up up, all the semantic security notions presented in this section are summarized in Figure 5.2.

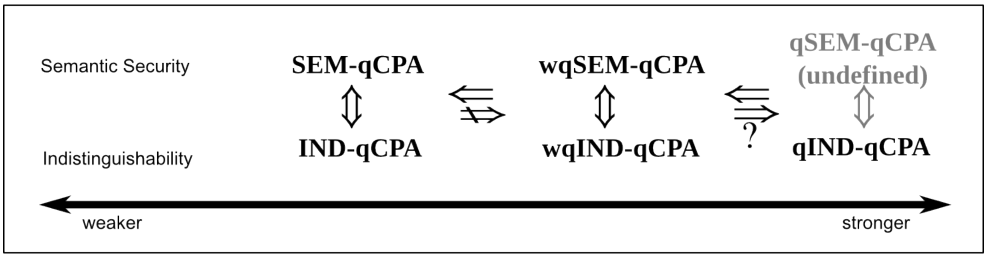


Figure 5.2: Relations for semantic security notions in **QS**2.

Quantum CCA

Finally, here we give a brief discussion about the possibility of extending the **QS**2 framework of security notions for SKES to the quantum chosen ciphertext attack (qCCA) case. The resulting notions, when applicable, are always stronger than the related qCPA notions, with counterexamples closely matching the classical ones. However, a few issues arise.

The case of quantum CCA1 is straightforward for the classical IND case. The resulting IND-qCCA1 notion is just as the IND-qCPA notion, augmented by a quantum CCA query *before* the classical IND query. This is modeled in the security game by giving to the first stage IND adversary oracle access to the quantum decryption oracle $|\mathsf{Dec}_k\rangle$.

The case of wqIND-qCCA1 and qIND-qCCA1 are also straightforward, as the decryption queries only happen before the qIND query. It is just necessary to define the type-(2) decryption oracle $|\mathsf{Dec}_k\rangle_{(2)}$, but this is trivial considered that $|\mathsf{Dec}_k\rangle_{(2)} = |\mathsf{Enc}_k\rangle^{\dagger}_{(2)}$. However, Construction 5.19 will require strong qPRPs in order to be secure under the new notion, as already discussed.

The case of qCCA2, instead, is much more delicate. For the classical IND case, [BZ13b] shows how to correctly define IND-qCCA2 (and how to achieve it), by carefully defining the decryption oracle *after* the IND query. For the 'fully quantum case' qIND-qCCA2, however, it is unclear whether such a notion is even possible to define. The problem is that in the CCA2 game it is necessary to ensure that the adversary does not ask for a decryption of the challenge ciphertext, leading to a trivial break. While this is easily demanded in the classical world, it raises several issues in the quantum world. What does it mean for a quantum ciphertext state to be different from the challenge ciphertext? And, more importantly: how can the challenger check? There might be several reasonable ways to solve the first issue but, as long as the queries are not classical, it is not known how to solve the second issue without disturbing the challenge ciphertext and the query states. Defining CCA2 security notions in the quantum world is an outstanding open problem [GHS16, ABF$^+$16].

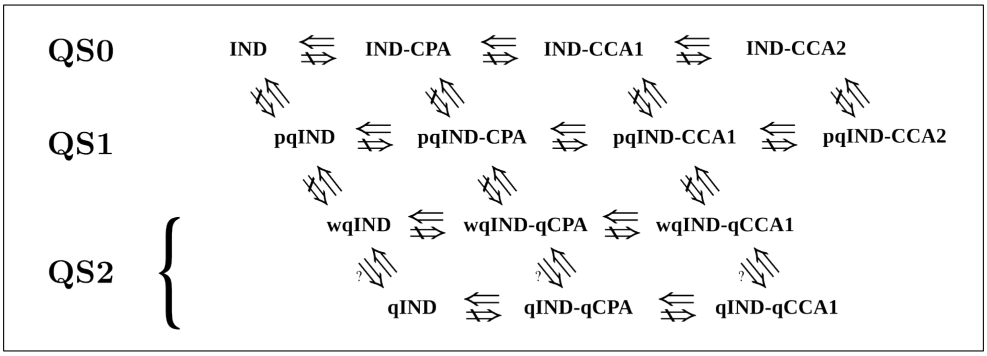


Figure 5.3: Relations for SKES security notions in the quantum world.

All the indistinguishability notions for classical SKES in the quantum world are summarized in Figure 5.3.

Chapter 6

QS3: Fully Quantum Security

In the previous chapters, we studied the security of *classical* cryptographic primitives in different quantum scenarios. In this chapter, instead, we focus on the security of *quantum* cryptographic primitives, that is, cryptographic primitives which are meant to be natively run on a quantum computing device. The quantum security class **QS**3 encompasses all those cryptographic objects which deal mainly with the manipulation and protection of *quantum data*. As such, one can see **QS**3 as a natural extension of **QS**0 to a 'fully quantum computing world', that is, a world where quantum computing has become ubiquitous, and honest users have access to quantum devices.

One could consider **QS**3 to be somehow 'the last step', from a chronological point of view, in the study of computer security, in the sense that the models therein only concern possible future scenarios, somehow far away from the contemporary era of classical devices. However, such interpretation has not to be taken too literally. **QS**3 is about *security of cryptographic primitives which natively deal with quantum information*, and this does not necessarily involve computation performed on some futuristic, fully-fledged quantum computer. As an example, *quantum key distribution (QKD)* [BB14] is a well-studied area in modern cryptography, where honest parties want to establish a shared secret by using quantum communication channels[1]. As such, QKD perfectly fits in the **QS**3 domain; however it is far from being futuristic: commercial implemementations of large-scale QKD systems have been available for a few years already [SLB$^+$11], and have been deployed in many real-world scenarios.

Despite this, in the rest of this chapter we will focus on the study of the quantum security of cryptographic primitives natively designed to run on a fully-fledged quantum computer. We will first introduce the concept of *quantum encryption* (that is, cryptographic schemes meant to protect quantum data), and then we will see an application by extending ORAMs to the case where the database to be protected is composed of quantum data.

[1] Remarkably, most often than not, the term 'quantum cryptography' is (incorrectly) used a synonym for 'QKD' in scientific literature.

My Scientific Contributions in this Chapter

Regarding quantum encryption, most of the material from sections 6.1 and 6.2 first appears in [ABF$^+$16], which is a joint work with Gorjan Alagic, Anne Broadbent, Bill Fefferman, Christian Schaffner, and Michael St. Jules. In particular, Anne, Gorjan and I devised the idea of Construction 6.13 and Construction 6.18, while I developed Theorem 6.14 and Theorem 6.19 with the help of Bill and Gorjan. Theorem 6.16 is my contribution, while Anne, Christian and Michael focused on the notion of quantum semantic security in the **QS**3 sense [Jul17], which is a topic not covered in this thesis.

The part about QORAMs in Section 6.3 is my contribution, including the safe extractor technique.

6.1 Secret-Key Quantum Encryption

In this section, we study the computational security of *quantum encryption schemes*, that is, schemes which are meant to protect quantum data. In this sense, plaintexts and ciphertexts are pure quantum states from Hilbert spaces of appropriate dimension, or mixed states of such. In fact, the schemes described in this section are meant to work on *arbitrary* quantum states, even those who might be entangled with external systems, therefore it is crucial to use the density matrix formalism. Accordingly, (families of) classical plaintext and ciphertext spaces $\mathcal{X}$ and $\mathcal{Y}$ are replaced with quantum operator spaces $\mathfrak{D}\left(\mathfrak{H}_{\mathcal{X}}\right)$ and $\mathfrak{D}\left(\mathfrak{H}_{\mathcal{Y}}\right)$ respectively, where $\mathfrak{H}_{\mathcal{X}}$ and $\mathfrak{H}_{\mathcal{Y}}$ are (families of) complex Hilbert spaces of dimension $|\mathcal{X}| = 2^m$ and $|\mathcal{Y}| = 2^c$ respectively, for functions m and c polynomial in the security parameter n.

However, the encryption keys used will still be classical. This is actually a feature, as these schemes require for honest parties to be able to encrypt and decrypt several times with the same keys, and classical keys can be stored and managed more easily.

Definitions, and the Quantum One-Time Pad

We start by defining *secret-key quantum encryption schemes (SKQES)*, as introduced in [ABF$^+$16]. We assume that the secret-key space is defined as $\mathcal{K} = \left(\mathcal{K}_n\right)_n := \{0,1\}^n$, so that the key-length is n bits. Later, we will define an additional Hilbert space $\mathfrak{H}_{\mathsf{Env}}$ (the *environment space*) in order to model auxiliary information used by some adversary. Encryption accepts a classical key and a quantum plaintext, and outputs a quantum ciphertext; decryption accepts a classical key and a quantum ciphertext, and outputs a quantum plaintext. The correctness guarantee is that plaintexts are preserved (up to negligible error) under encryption followed by decryption under the same key.

Definition 6.1 (Secret-Key Quantum Encryption Scheme (SKQES))**.** *A secret-key quantum encryption scheme* (SKQES) *with plaintext space* $\mathfrak{D}\left(\mathfrak{H}_{\mathcal{X}}\right)$*,* *ciphertext space* $\mathfrak{D}(\mathfrak{H}_{\mathcal{Y}})$*, and (classical) key space* $\mathcal{K}$ *is a tuple of* QPT *algorithms* $\mathcal{E} := \mathcal{E}_{\mathcal{K},\mathfrak{D}(\mathfrak{H}_{\mathcal{X}}),\mathfrak{D}(\mathfrak{H}_{\mathcal{Y}})} := (\mathsf{KGen}, \mathsf{QEnc}, \mathsf{QDec})$*:*

1. $\mathsf{KGen} :\to \mathcal{K}$;
2. $\mathsf{QEnc} : \mathcal{K} \times \mathfrak{D}(\mathfrak{H}_{\mathcal{X}}) \to \mathfrak{D}(\mathfrak{H}_{\mathcal{Y}})$;
3. $\mathsf{QDec} : \mathcal{K} \times \mathfrak{D}(\mathfrak{H}_{\mathcal{Y}}) \to \mathfrak{D}(\mathfrak{H}_{\mathcal{X}})$;

such that $\left|\mathsf{QDec}_k \circ \mathsf{QEnc}_k - \mathbb{I}_{\mathfrak{H}_{\mathcal{X}}}\right|_\diamond \leq \mathsf{negl}$ *for all* $k \leftarrow \mathsf{KGen}$.

As usual, we denote by QEnc_k the action of QEnc on a specific, fixed key $k \leftarrow \mathsf{KGen}$, and analogously for QDec_k. However, unlike in the case of Definition 3.12, for simplicity we will omit the possibility that the decryption algorithm answers (a quantum analogue of) $\perp$ to some decryption queries. One of the most basic examples of SKQES is the *quantum one-time pad (QOTP)*. The QOTP takes as input an n-qubit plaintext spaces and a $2n$-bit secret key. Every pair of bits from the key selects one over four possible single-qubit Pauli operators $\mathbb{I}, X, Y, Z$ as $X^{(\text{first bit})} Z^{(\text{second bit})}$. Thus, the secret key defines a sequence of n independent single-qubit Pauli operators, each of them to be applied separately to each of the n qubits of the plaintext (that is, the key defines an element of the n-qubit Pauli group), resulting in the ciphertext. Since Pauli operators are self-adjoint, decryption just applies the same procedure to the ciphertext state.

Construction 6.2 (Quantum One-Time Pad (QOTP)[AMTdW00, BR03])**.** *Let* $\mathfrak{H}_{\mathcal{X}} = \mathfrak{H}_{\mathcal{Y}}$ *of dimension* $\{0,1\}^n$*, and let* $\mathcal{K} = \{0,1\}^{2n}$*. Define the* quantum one-time pad (QOTP) on n qubits $\mathsf{QOTP}_k := (\mathsf{KGen}, \mathsf{QEnc}, \mathsf{QDec})$ *as the SKQES with key space* $\mathcal{K}$*, plaintext space* $\mathfrak{D}(\mathfrak{H}_{\mathcal{X}})$*, and ciphertext space* $\mathfrak{D}(\mathfrak{H}_{\mathcal{Y}})$*, defined as:*

1. $\mathsf{KGen} \to k$*, with* $k \overset{\$}{\leftarrow} \mathcal{K}$;
2. $\mathsf{QEnc}_k(\varphi) := P(k)\varphi P(k)^\dagger$;
3. $\mathsf{QDec}_k(\rho) := P(k)\psi P(k)^\dagger$,

where $P(k) := \prod_{j=1}^n X_j^{k_{2j-1}} Z_j^{k_{2j}} \in \mathfrak{P}_n$*, and* k_j *is the* j*-th bit of* k*.*

Notice how *two* bits of key are needed for every qubit of plaintext. The QOTP is known [AMTdW00, BR03] to be quantum information-theoretically secure, as long as the key is completely random and only used once.

Quantum Indistinguishability

We use a definition of *computational quantum indistinguishability* introduced in [BJ15], which we relabel here as QIND for our purposes (notice the capital 'Q', unlike Definition 5.13), and which is the analogue of the classical IND notion, by keeping in mind that a quantum adversary for a SKQES could try to distinguish states that he has previously entangled with the environment. Intuitively, the adversary produces a tripartite system, composed of two plaintext states and an environment state. The environment state is passed to the second stage adversary, who also receives an encryption of one of the two other states, selected at random, while the other one is traced out. As usual, the goal of the adversary is to guess which one of the two plaintext system was selcted for encryption. Formally, we define the following.

Experiment 6.3 ($\mathsf{Game}^{\mathsf{QIND}}_{\mathcal{E},\mathcal{A}}$)**.** *Let $\mathcal{E}$ be a SKQES, and $\mathcal{A} := (\mathcal{M}, \mathcal{D})$ a QIND adversary as from Definition 5.11. The* QIND experiment *proceeds as follows:*

```
1:  Input: $n \in \mathbb{N}$
2:  $k \leftarrow \mathsf{KGen}$
3:  $(\varphi^0, \varphi^1, \sigma) \leftarrow \mathcal{M}$
4:  $b \xleftarrow{\$} \{0,1\}$
5:  $\psi \leftarrow \mathsf{QEnc}(\varphi^b)$
6:  trace out $\varphi^{1-b}$
7:  $b' \leftarrow \mathcal{D}(\psi, \sigma)$
8:  if $b = b'$ then
9:      Output: 1
10: else
11:     Output: 0
```

The advantage of $\mathcal{A}$ *is defined as:*

$$\mathsf{Adv}^{\mathsf{QIND}}_{\mathcal{E},\mathcal{A}} := \Pr\left[\mathsf{Game}^{\mathsf{QIND}}_{\mathcal{E},\mathcal{A}} \to 1\right] - \frac{1}{2}.$$

Definition 6.4 (Indistinguishability of Quantum Ciphertexts (QIND))**.** *A SKQES $\mathcal{E}$ has* indistinguishable quantum encryptions (or, it is QIND secure) *iff, for any QIND adversary $\mathcal{A}$ it holds that:* $\mathsf{Adv}^{\mathsf{QIND}}_{\mathcal{E},\mathcal{A}} \leq \mathsf{negl}$.

Notice how this definition and the related experiment are exactly the same as Experiment 5.12 and Definition 5.13, even the adversarial model is the same as in the qIND case from Chapter 5. This is not incidental: historically, notions of computational indistinguishability for encrypted quantum states have been introduced in [BJ15] and [GHS16] as concurrent and independent works (although [BJ15] was published earlier), but for different purposes and with slightly different flavors. What we call here QIND was originally called q-IND-CPA-2 in [BJ15] (minus the CPA part), while qIND was originally called $(\mathcal{C}\mathcal{Q}n2e)$-IND in [GHS16]. However, the former notion was given in the context of *fully homomorphic quantum encryption* (which, according to our framework, belongs to the **QS**3 setting), while the latter was given in the context of *superposition-resistant quantum encryption* (as we mean it in the **QS**2 sense). Further developments on the topic appeared in [ABF$^+$16] and in the proceedings version of [GHS16], which led to the conclusion that this indistinguishability model for quantum encryption is virtually the same, which can be used both in the setting of classical encryption resistant to quantum queries (**QS**2) or 'fully' quantum encryption (**QS**3). In this work, in the attempt of providing a unified notation to work with, we use respectively 'qIND' and 'QIND' (with different capitalization of the first letter) in order to highlight the specific domain we are talking about, but making clear that, technically, it is the same model.

As usual, we can extend the QIND notion to CPA and non-adaptive CCA attacks. Since we are in the **QS**3 domain, it is not ambiguous to write (e.g.) QIND-CPA instead of QIND-QCPA, because the plaintexts we are considering are inherently quantum, so a CPA notion in this scenario *must* be quantum. Hence, without need of specifying further, we call the resulting notions QIND-CPA and QIND-CCA1. This is also useful in order to understand 'at first glance' that we are talking about a **QS**3 notion.

Experiment 6.5 ($\mathsf{Game}^{\mathsf{QIND-CPA}}_{\mathcal{E},\mathcal{A}}$)**.** *Let $\mathcal{E}$ be a SKQES, and $\mathcal{A} := (\mathcal{M}, \mathcal{D})$ a QIND adversary. The* QIND-CPA experiment *proceeds as follows:*

```
Input: $n \in \mathbb{N}$
$k \leftarrow \mathsf{KGen}$
$(\varphi^0, \varphi^1, \sigma) \leftarrow \mathcal{M}^{\mathsf{QEnc}}$
$b \xleftarrow{\$} \{0,1\}$
$\psi \leftarrow \mathsf{QEnc}(\varphi^b)$
trace out $\varphi^{1-b}$
$b' \leftarrow \mathcal{D}^{\mathsf{QEnc}}(\psi, \sigma)$
if $b = b'$ then
    Output: 1
else
    Output: 0
```

The advantage of $\mathcal{A}$ *is defined as:*

$$\mathsf{Adv}^{\mathsf{QIND-CPA}}_{\mathcal{E},\mathcal{A}} := \Pr\left[\mathsf{Game}^{\mathsf{QIND-CPA}}_{\mathcal{E},\mathcal{A}} \to 1\right] - \frac{1}{2}.$$

Definition 6.6 (Indistinguishability of Quantum Ciphertexts Under Chosen Plaintext Attack (QIND-CPA))**.** *A SKQES $\mathcal{E}$ has* indistinguishable quantum encryptions under chosen plaintext attack (or, it is QIND-CPA secure) *iff, for any QIND adversary $\mathcal{A}$ it holds:* $\mathsf{Adv}^{\mathsf{QIND-CPA}}_{\mathcal{E},\mathcal{A}} \leq \mathsf{negl}$.

Clearly, QIND-CPA is at least as strong as QIND.

Theorem 6.7 (QIND-CPA $\implies$ QIND)**.** *If a SKQES is QIND-CPA secure, then it is also QIND secure.*

However, the converse is not necessarily true. For example, the QOTP (Construction 6.2) is information-theoretically secure for random, unrelated keys, and thus it is also QIND. However, as in the classical OTP analogue, security is compromised if the same key is used more than once.

Theorem 6.8 (QIND $\not\Longrightarrow$ QIND-CPA)**.** *There exist SKQES which are QIND secure, but not QIND-QCPA secure.*

As usual, extending the above security notion to the QCCA1 case is straightforward.

Experiment 6.9 ($\mathsf{Game}^{\mathsf{QIND-CCA1}}_{\mathcal{E},\mathcal{A}}$)**.** *Let $\mathcal{E}$ be a SKQES, and $\mathcal{A} := (\mathcal{M}, \mathcal{D})$ a QIND adversary. The* QIND-CCA1 experiment *proceeds as follows:*

```
1:  Input: $n \in \mathbb{N}$
2:  $k \leftarrow \mathsf{KGen}$
3:  $(\varphi^0, \varphi^1, \sigma) \leftarrow \mathcal{M}^{\mathsf{QEnc},\mathsf{QDec}}$
4:  $b \xleftarrow{\$} \{0,1\}$
5:  $\psi \leftarrow \mathsf{QEnc}(\varphi^b)$
6:  trace out $\varphi^{1-b}$
7:  $b' \leftarrow \mathcal{D}^{\mathsf{QEnc}}(\psi, \sigma)$
8:  if $b = b'$ then
9:      Output: 1
10: else
11:     Output: 0
```

The advantage of $\mathcal{A}$ *is defined as:*

$$\mathsf{Adv}^{\mathsf{QIND-CCA1}}_{\mathcal{E},\mathcal{A}} := \Pr\left[\mathsf{Game}^{\mathsf{QIND-CCA1}}_{\mathcal{E},\mathcal{A}} \to 1\right] - \frac{1}{2}.$$

Definition 6.10 (Indistinguishability of Quantum Ciphertexts Under Non-Adaptive Chosen Ciphertext Attack (QIND-CCA1))**.** *A SKQES $\mathcal{E}$ has* indistinguishable quantum encryptions under non-adaptive chosen ciphertext attack (or, it is QIND-CCA1 secure) *iff, for any QIND adversary $\mathcal{A}$ it holds:*

$$\mathsf{Adv}^{\mathsf{QIND-CCA1}}_{\mathcal{E},\mathcal{A}} \leq \mathsf{negl}.$$

As in the classical case, in a completely specular way to Theorems 3.31 and 3.32, one can show that QIND-CCA1 is strictly stronger than QIND-CPA.

Theorem 6.11 (QIND-CCA1 $\Longrightarrow$ QIND-CPA)**.** *If a SKQES is QIND-CCA1 secure, then it is also QIND-CPA secure.*

Theorem 6.12 (QIND-CPA $\not\Longrightarrow$ QIND-CCA1)**.** *There exists a SKQES which is QIND-CPA secure, but not QIND-CCA1 secure.*

Secure Construction

QIND-CCA1 secure SKQES can be constructed given the existence of pqPRF (and hence from pqOWF, as from Corollary 4.15), as shown in [ABF$^+$16]. The idea of the construction is analogous to the one for Construction 3.26: a (classical) randomness is processed by the keyed pqPRF, and the output is used as a key for the QOTP; the ciphertext is composed by the output of the QOTP, plus the classical randomness.

Construction 6.13 ([ABF$^+$16, Scheme 1]). *Let $\mathcal{F} : \mathcal{K} \times \{0,1\}^{2n} \to \{0,1\}^{2n}$ be a pqPRF as from Definition 4.13, and let $\mathfrak{H}_\mathcal{X}$ be a complex Hilbert space of dimension 2^n. Define $\mathcal{E} = \mathcal{E}_{\mathcal{K},\mathfrak{D}(\mathfrak{H}_\mathcal{X}),\mathfrak{D}(\mathfrak{H}_\mathcal{X})} := (\mathsf{KGen}, \mathsf{QEnc}, \mathsf{QDec})$ as the SKQES with key space $\mathcal{K}$, plaintext and ciphertext space $\mathfrak{D}\left(\mathfrak{H}_\mathcal{X}\right)$, as follows:*

1. *$\mathsf{KGen} \to k$, with $k \xleftarrow{\$} \mathcal{K}$;*
2. *$\mathsf{QEnc}_k(\varphi) \to \psi \otimes |r\rangle\langle r|$, with $\psi := \mathsf{QOTP}_{\mathcal{F}_k(r)}(\varphi)$, where $r \xleftarrow{\$} \{0,1\}^{2n}$;*
3. *$\mathsf{QDec}_k(\rho) \to \mathsf{QOTP}_{\mathcal{F}_k(s)}(\sigma)$, where s is obtained by measuring the last $2n$ qubits of ρ, while σ is the reduced state left after such a measurement.*

The above construction is QIND-CCA1 secure.

Theorem 6.14 ([ABF$^+$16, Lemma 14]). *Let $\mathcal{E}$ be the SKQES from Construction 6.13 built using a pqPRF $\mathcal{F}$. Then $\mathcal{E}$ is QIND-CCA1 secure.*

Proof. First, we analyze the security of the scheme in an idealized scenario where $\mathcal{F}$ is replaced by a function $f : \{0,1\}^{2n} \to \{0,1\}^{2n}$ selected truly at random. We show that, in this case, $\mathcal{A}$ correctly guesses the challenge state in the QIND-CCA1 game with probability at most $\frac{1}{2} + \mathsf{negl}$. In fact, this bound holds for a stronger adversary $\mathcal{A}^*$, who has access to a classical oracle for f prior to the challenge, and access to polynomially-many pairs $(r_i, f(r_i))$ where $r_i \xleftarrow{\$} \{0,1\}^{2n}$ for $1 = 1, \dots, q = \mathsf{poly}(n)$, after the challenge. This adversary is stronger than $\mathcal{A}$ since it can simulate $\mathcal{A}$ by implementing the oracles Enc_f and Dec_f using its f oracles. Since the input r into f in the challenge ciphertext is uniformly random, the probability that any of the polynomially-many oracle calls of $\mathcal{A}^*$ uses the same r is negligible. In the case that no oracle calls use r, the mixtures of the inputs to $\mathcal{A}^*$ (including the pairs $(r_i, f(r_i))$) are the same for any of the two original challenge states. This fact can be verified by first averaging over the values of $f(r)$: since f is uniformly random, $f(r)$ is also uniformly random as well as independent of the other values of f. In both cases, applying the quantum one-time pad results in the state:

$$\frac{1}{2^n}\mathbb{I} \otimes |r\rangle\langle r| \otimes \sigma \otimes |r_1\rangle\langle r_1| \otimes |f(r_1)\rangle\langle f(r_1)| \otimes \cdots \otimes |r_q\rangle\langle r_q| \otimes |f(r_q)\rangle\langle f(r_q)|\,,$$

where σ is the state in the 'environment register' of $\mathcal{A}^*$ (communication channel in Experiment 6.9), and hence indistinguishability follows.

Next, we consider the case that f is replaced by a post-quantum pseudorandom function $\mathcal{F}_k$ for a random key k. We show that a successful QIND-CCA1 adversary $\mathcal{A}$ (i.e., one that distinguishes challenges with probability at least $\frac{1}{2}+\varepsilon$ for non-negligible ε) can be used to construct a successful adversary $\mathcal{B}$ for the pqPRF, i.e., one that distinguishes $\mathcal{F}_k$ from random with non-negligible advantage over guessing. The adversary $\mathcal{B}$ is a QPT algorithm with classical oracle access to a function $\hbar : \{0,1\}^{2n} \to \{0,1\}^{2n}$, and his goal is to output 0 if $\hbar$ is selected perfectly at random, and 1 if $\hbar = \mathcal{F}_k$ for some k. Define the simulated oracles:

$$\begin{aligned} &\mathsf{QEnc}_{\hbar} : \varphi \mapsto \mathsf{QOTP}_{\hbar(r)}(\varphi) \otimes |r\rangle\langle r| \text{ for } r \overset{\$}{\leftarrow} \{0,1\}^{2n}; \quad \text{and} \\ &\mathsf{QDec}_{\hbar} : \psi \otimes |r'\rangle\langle r'| \mapsto \mathsf{QOTP}_{\hbar(r')}(\psi), \end{aligned}$$

where, as before, we assume that $\mathsf{QDec}_{\hbar}$ measures the second register before decrypting the first one. Note that if $\hbar = \mathcal{F}_k$ then these are exactly the encryption and decryption oracles (with key k) of the real SKQES scheme.

The algorithm $\mathcal{B}$ proceeds as follows. First, it executes $\mathcal{A}$, and replies to $\mathcal{A}$'s encryption queries with $\mathsf{QEnc}_{\hbar}$ and to $\mathcal{A}$'s decryption queries with $\mathsf{QDec}_{\hbar}$. When $\mathcal{A}$ performs the QIND challenge query with plaintext states φ^0 and φ^1, $\mathcal{B}$ replies with the encryption of either of the two, each with probability $\frac{1}{2}$, and traces out the other one. Then $\mathcal{B}$ keeps answering $\mathcal{A}$'s encryption queries as before with his simulated oracle. If eventually $\mathcal{A}$ correctly guesses the plaintext selcted by $\mathcal{B}$, then $\mathcal{B}$ outputs 1; otherwise it outputs a random bit. If $\hbar = \mathcal{F}_k$ then we have exactly simulated the QIND-CCA1 game with adversary $\mathcal{A}$; otherwise, $\mathcal{B}$ still correctly distinguishes the PRF from random with probability $\frac{1}{2}$. So, the overall success probability of $\mathcal{B}$ is $\frac{1}{2} + \frac{\varepsilon}{2}$, which is non-negligible over guessing. This concludes the proof. □

Notice how the security of Construction 6.13 only relies on the post-quantum security of the PRF, in the **QS**1 sense. In particular, from Corollary 4.15, this gives a construction of QIND-CCA1 secure SKQES from the existence of pqOWF.

Corollary 6.15 (of Theorem 6.14 and Corollary 4.15)**.** *If pqOWF exist, then QIND-CCA1 SKQES exist.*

Another way to build secure SKQES is to rely on the security of some (classical) SKES in **QS**2, and 'lift' the SKES construction to the **QS**3 scenario through the use of type-(2) operators. The following theorem is not found in the literature, but is a direct consequence of [GHS16, Theorem 3.4] and the observation made after Definition 6.4, i.e., the adversarial model (and corresponding security notions) for (**QS**2) qIND and (**QS**3) QIND are basically the same.

Theorem 6.16. *Let* $\mathcal{E} = \mathcal{E}_{\mathcal{K},\mathcal{X},\mathcal{Y}} := (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ *be a SKES, and let* $\mathcal{E}' = \mathcal{E}'_{\mathcal{K},\mathfrak{D}(\mathfrak{H}_{\mathcal{X}}),\mathfrak{D}(\mathfrak{H}_{\mathcal{Y}})} := (\mathsf{KGen}', \mathsf{QEnc}, \mathsf{QDec})$ *be a SKQES constructed as follows:*

1. $\mathsf{KGen}' \to k$, *with* $k \leftarrow \mathsf{KGen}$*;*
2. $\mathsf{QEnc}_k(\varphi) \to |\mathsf{Enc}_k\rangle_{(2)}\varphi\langle\mathsf{Enc}_k|^{\dagger}_{(2)}$*;*
3. $\mathsf{QDec}_k(\psi) \to |\mathsf{Dec}_k\rangle_{(2)}\varphi\langle\mathsf{Dec}_k|^{\dagger}_{(2)}$*,*

where $|\mathsf{Enc}_k\rangle_{(2)}, |\mathsf{Dec}_k\rangle_{(2)}$ *are type-*(2) *unitary operators associated to* $\mathsf{Enc}, \mathsf{Dec}$*. If* $\mathcal{E}$ *is qIND(-qCPA/qCCA1), then* $\mathcal{E}'$ *is QIND(-CPA/CCA1).*

Proof (sketch). The proof follows from [GHS16, Appendix C], but it basically boils down to what already discussed after Definition 6.4. Namely, the experiments for qIND-qCCA1 and QIND-CCA1 are fundamentally the same, the only difference is that in the qIND- version, encryption and decryption oracles are specifically type-(2) operators derived from classical SKES. So the only thing left to show is that the scheme defined by such encryption/decryption operators as in the statement of the theorem is actually a SKQES. This is trivially shown by observing that:

$$(\mathsf{QDec}_k \circ \mathsf{QEnc}_k)(\varphi) = |\mathsf{Dec}_k\rangle_{(2)}|\mathsf{Enc}_k\rangle_{(2)}\varphi\langle\mathsf{Enc}_k|^{\dagger}_{(2)}\langle\mathsf{Dec}_k|^{\dagger}_{(2)} = \varphi$$

so that QEnc and QDec respect Definition 6.1. □

The above is a typical example of what discussed in *Reason #4* of Section 5.1, about the necessity of superposition-based quantum security for composition results in fully quantum scenarios. Notice in fact that in the above theorem it is *crucial* that $\mathcal{E}$ is a scheme secure in the **QS**2 sense: a 'simply' post-quantum $\mathcal{E}$ (in the **QS**1 sense) would not work, because the same impossibility result described in Section 5.3 would apply.

6.2 Public-Key Quantum Encryption

When we move to the public-key scenario for quantum encryption schemes, intuitively we want the same kind of functionality offered by classical PKES, but with the possibility of encrypting arbitrary quantum states. As usual, we assume classical public/private key pairs $(\mathsf{pk}, \mathsf{sk})$, where w.l.o.g. we assume that, for security parameter n, public keys are of bit size $p(n)$, while secret keys are of bit size $\jmath(n)$, for polynomial functions p,$\jmath$. Under this notation, we identify the keyspace $\mathcal{K}$ as $(\mathcal{K}_n)_n = (\mathcal{K}^p{}_n)_n \times (\mathcal{K}^{\jmath}{}_n)_n =: \mathcal{K}^p \times \mathcal{K}^{\jmath} \subset \{0,1\}^{p(n)} \times \{0,1\}^{\jmath(n)}$. We define a *quantum public-key encryption scheme (PKQES)* as in [ABF⁺16], in the following way.

Definition 6.17 (Public-Key Quantum Encryption Scheme (PKQES))**.** *A* public-key quantum encryption scheme (PKQES) *with plaintext space* $\mathfrak{D}(\mathfrak{H}_{\mathcal{X}})$*, ciphertext space* $\mathfrak{D}(\mathfrak{H}_{\mathcal{Y}})$*, and key space* $\mathcal{K} := \mathcal{K}^p \times \mathcal{K}^{\jmath}$ *is a tuple of* QPT *algorithms* $\mathcal{E} := \mathcal{E}_{\mathcal{K},\mathfrak{D}(\mathfrak{H}_{\mathcal{X}}),\mathfrak{D}(\mathfrak{H}_{\mathcal{Y}})} := (\mathsf{KGen}, \mathsf{QEnc}, \mathsf{QDec})$*:*

1. $\mathsf{KGen} :\to \mathcal{K}$;

2. $\mathsf{QEnc} : \mathcal{K}^{p} \times \mathfrak{D}\left(\mathfrak{H}_{\mathcal{X}}\right) \to \mathfrak{D}\left(\mathfrak{H}_{\mathcal{Y}}\right)$;

3. $\mathsf{QDec} : \mathcal{K}^{s} \times \mathfrak{D}\left(\mathfrak{H}_{\mathcal{Y}}\right) \to \mathfrak{D}\left(\mathfrak{H}_{\mathcal{X}}\right)$;

such that $\left|\mathsf{QDec}_{\mathsf{sk}} \circ \mathsf{QEnc}_{\mathsf{pk}} - \mathbb{I}_{\mathfrak{H}_{\mathcal{X}}}\right|_{\diamond} \leq \mathsf{negl}$ *for all* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}$.

For the security model, as usual, we use the same QIND indistinguishability notion for SKQES, but recalling that (as explained in Section 3.3 for classical SKES) in the public-key scenario the minimum meaningful security notion is QIND-CPA as from Definition 6.6.

Secure Construction

QIND-CPA secure PKQES can be constructed given the existence of pqOWTP, as shown in [ABF^{+}16]. The idea of the construction is analogous to the one for Construction 3.41: a (classical) randomness is sampled, and used as an input to the Goldreich-Levin PRNG to generate a key for the QOTP on the plaintext state; then the pqOWTP is applied to that randomness, and the result appended to the output of the QOTP. For the decryption, the trapdoor of the pqOWTP is used to recover the randomness, and hence the key for the QOTP, inverting the encryption. Assume for simplicity that $\mathcal{X} = \{0,1\}^n$. Then we define the following.

Construction 6.18 (PKQES from pqOWTP)**.** *Let* $\mathcal{P} := (\mathsf{Gen}, \mathsf{Eval}, \mathsf{Invert})$ *be a pqOWTP on* $\mathcal{X}^2$*, with index and trapdoor spaces* $\mathcal{I}$ *and* $\mathcal{T}$ *respectively, and let* $\mathcal{G}_{\mathcal{P}} : \mathcal{X}^2 \to \mathcal{X}^2$ *be the Goldreich-Levin PRNG for* $\mathcal{P}$ *(seen as a OWF with hard-core predicates). Define* $\mathcal{E} = \mathcal{E}_{\mathcal{K},\mathfrak{D}(\mathfrak{H}_{\mathcal{X}}),\mathfrak{D}\left(\mathfrak{H}_{\mathcal{X}}{}^{\otimes 3}\right)} := (\mathsf{KGen}, \mathsf{QEnc}, \mathsf{QDec})$ *as a PKQES with (public,private) key space* $\mathcal{K} = \mathcal{K}^{p} \times \mathcal{K}^{s}$ *(where* $\mathcal{K}^{p} := \mathcal{I}$ *and* $\mathcal{K}^{s} := \mathcal{T}$*, plaintext space* $\mathfrak{D}\left(\mathfrak{H}_{\mathcal{X}}\right)$*, and ciphertext space* $\mathfrak{D}\left(\mathfrak{H}_{\mathcal{X}}{}^{\otimes 3}\right)$*, in the following way:*

1. $\mathsf{KGen} \to (\mathsf{pk}, \mathsf{sk})$*, with* $(\mathsf{pk}, \mathsf{sk}) := (i, t) \leftarrow \mathsf{Gen}$;

2. $\mathsf{QEnc}_{\mathsf{pk}}(\varphi) \to \psi \otimes |z\rangle\langle z|$,
   with $\psi := \mathsf{QOTP}_{\mathcal{G}_{\mathcal{P}}(r)}(\varphi)$ *and* $z \leftarrow \mathsf{Eval}(\mathsf{pk}, r)$*, where* $r \xleftarrow{\$} \mathcal{X}^2$;

3. $\mathsf{QDec}_{\mathsf{sk}}(\rho) \to \mathsf{QOTP}_{\mathcal{G}_{\mathcal{P}}(s)}(\sigma)$,
   with $s \leftarrow \mathsf{Invert}(\mathsf{pk}, \mathsf{sk}, z)$*, where* z *is obtained by measuring the last* $2n$ *qubits of* ρ*, while* σ *is the reduced state left after such a measurement.*

The above construction is a simplified version of [ABF^{+}16, Scheme 2], and it can be shown to be QIND-CPA secure.

Theorem 6.19 ([ABF^{+}16, Lemma 14])**.** *Construction 3.41 is a QIND-CPA secure PKQES.*

Proof (sketch). The proof is as in Theorem 3.42: recall that the QOTP is information-theoretically secure for independent keys sampled uniformly at random, and hence computationally secure for keys output by the pqPRNG in the construction here. Then, the only way for an adversary to attack the scheme would be to extract information about r by looking at the OWTP image z obtained through Eval, but this is impossible because $\mathcal{P}$ is a pqOWTP family, and $\mathcal{G}_{\mathcal{P}}$ only outputs (post-quantum) hard-core bits. □

6.3 Quantum ORAM

In this section we study *quantum ORAMs (QORAM)*, that is, ORAM constructions operating on *quantum data*. This new cryptographic primitive defined in [GKK17] considers the same scenario as in the ORAM case, but where all the parties have quantum computing and communication capabilities. As we will see, many difficulties arise in modeling this scenario.

In the QORAM model, the client $\mathcal{C}$ and the server $\mathcal{S}$ are both QPT algorithms, sharing a *quantum communication channel* (quantum register) Ψ. Since such a quantum channel can also be used to share classical information, we assume without loss of generality that $\mathcal{A}$ and $\mathcal{S}$ also share a classical channel Ξ. In the following, if not otherwise stated, we will always assume that all the classical communication between $\mathcal{A}$ and $\mathcal{S}$ happens through Ξ, and all the quantum communication happens through Ψ. In this scenario, a computationally limited $\mathcal{C}$ wants to outsource a *quantum database (QDB)* to the more powerful $\mathcal{S}$, and perform operations on the QDB in a secure way, as in the ORAM case.

We have first to define what it means to have a 'quantum database'. In our case, this will be a structure of *quantum blocks*. A quantum block is a n_{blk}-qubit quantum state $\psi \in \mathfrak{D}\left(\mathfrak{H}_{n_{\mathsf{blk}}}\right)$ for a fixed parameter $n_{\mathsf{blk}} \in \mathbb{N}$ which depends on $\mathcal{C}$'s and $\mathcal{S}$'s architectures. A *quantum database* (QDB) of size $n_{\mathsf{db}} \in \mathbb{N}$ is a quantum register of $\mathcal{S}$ which stores n_{db} quantum blocks. It is important to notice that we impose no restriction on the nature of the states stored in the quantum blocks, i.e., these states could be mixed or entangled, amongst them or with states stored in other, external registers. As explained in the preliminaries, in the following, for simplicity, we abuse notation and denote such multipartite system with a tuple of quantum blocks $(\psi_1, \ldots, \psi_{n_{\mathsf{db}}})$. Since we assume this quantum register to reside on the server's side, we will denote it as $\mathcal{S}.|\mathtt{QDB}\rangle$. As in the ORAM case, the precise way this system of quantum blocks is represented in the quantum database is unspecified, and left to the exact implementation of the QORAM scheme taken into account. As usual, we will abuse notation and write that $\mathcal{S}.|\mathtt{QDB}\rangle(i) = \psi$ if ψ is the state obtained by tracing out all but the i-th subsystem of $\mathcal{S}.|\mathtt{QDB}\rangle$, and that $\psi \in \mathcal{S}.|\mathtt{QDB}\rangle$ if $\mathcal{S}.|\mathtt{QDB}\rangle(i) = \psi$ for some $i \in \mathbb{N}$.

A quantum block encodes (usually in an encrypted form) a *quantum data* *unit*, which is another quantum state representing the information that the client actually wants to access or modify, and possibly additional (quantum or classical) auxiliary information. Formally, a quantum data unit is a quantum state $\varphi \in \mathfrak{D}(\mathfrak{H}_{n_{\mathsf{dat}}})$ of n_{dat} qubits, where $n_{\mathsf{dat}} \leq n_{\mathsf{blk}}$ depends on $\mathcal{C}$'s and $\mathcal{S}$'s architecture. As before, no assumption is made about the nature of these quantum states. Every quantum block can encode a single quantum data unit, therefore at any given time t it is defined a CPTP map $|\mathsf{QData}\rangle_t : \mathcal{S}.|\mathtt{QDB}\rangle \to \mathfrak{D}(\mathfrak{H}_{n_{\mathsf{dat}}})$. With abuse of notation, we will denote by $|\mathsf{QData}\rangle(\psi)$ the quantum data unit encoded in the block ψ at a certain time. The client $\mathcal{C}$ can operate on the quantum database through *quantum data requests*.

Definition 6.20 (Quantum Data Request)**.** *A* quantum data request *to a database $\mathcal{S}.|QDB\rangle$ of size n_{db} is a tuple of the form $|\mathsf{qdr}\rangle = (\mathsf{op}, i, \varphi)$, where* $\mathsf{op} \in \{read, write\}, i \in \{1, \ldots, n_{\mathsf{db}}\}$*, and $\varphi \in \mathfrak{D}(\mathfrak{H}_{n_{\mathsf{dat}}})$ is a quantum data unit (φ can also be $|\perp\rangle$ if* $\mathsf{op} = read$*).*

Finally, we define the meaning of a *quantum communication transcript* during an execution of a QORAM protocol. As in the ORAM case, we will use the following definition.

Definition 6.21 (Quantum Communication Transcript)**.** *A* quantum communication transcript $|\mathsf{qcom}\rangle$ *at time t is the content of the communication registers (Ξ, Ψ) at time t of the protocol's execution.*

As in the ORAM case, in the following we will consider $|\mathsf{qcom}\rangle$ as a discrete function of the round $1, 2, \ldots$ of the protocol. Notice the following difference from the classical case: as this time $\mathcal{C}$ and $\mathcal{S}$ are also allowed to exchange quantum data through Ψ, it might not be possible for an adversary to obtain a full transcript of $|\mathsf{qcom}\rangle$ without disturbing the protocol. We will address this issue in the next section about security.

From now on, n_{blk} and n_{dat} will be fixed constants (the *quantum block size*, and *quantum data unit size*, resp.) As in the classical case, we assume that a server's QDB is always initialized empty (that is, with randomized encryptions of $|0\ldots0\rangle$ as data), and it is left up to the client the task of 'populating' the database. We are now ready to define a QORAM as follows.

Definition 6.22 (QORAM)**.** *Let $n_{\mathsf{Max}} \in \mathbb{N}, n_{\mathsf{msg}} \geq n_{\mathsf{dat}} \in \mathbb{N}$, and $\mathcal{E} =$* $(\mathsf{KGen}, \mathsf{QEnc}, \mathsf{QDec})$ *be a SKQES scheme mapping n_{msg}-qubit plaintext states to n_{blk}-qubit ciphertext states. A* QORAM (quantum oblivious random access machine) $\mathsf{QORAM}_{\mathcal{E}}$ *with parameters $(n_{\mathsf{Max}}, n_{\mathsf{dat}}, \mathcal{E})$ is a pair of two-party interactive* QPT *algorithms* $(\mathsf{QInit}, \mathsf{QAccess})$*, such that:*

- $\mathsf{QInit}(n, n_{\mathsf{db}}) \to (\mathcal{C}, \mathcal{S})$ *in the following way:*

  1. n is the security parameter, $n_{\mathsf{db}} \leq n_{\mathsf{Max}}$;

  2. $k \leftarrow$ $\mathsf{KGen}(n)$ is generated by $\mathcal{C}$;

3. *$\mathcal{S}$ includes a QDB $\mathcal{S}.|\texttt{QDB}\rangle = (\psi_1, \dots, \psi_{n_{\mathsf{db}}})$, where $\forall i \implies \psi_i \leftarrow \mathsf{QEnc}_k(|0\rangle\langle 0|)$;*

- $\mathsf{QAccess}(\mathcal{C}, \mathcal{S}, |\mathsf{qdr}\rangle) \rightarrow (\mathcal{C}', \mathcal{S}', |\mathsf{qcom}\rangle)$ *in the following way:*

  1. *$\mathcal{C}$ issues a quantum data request $|\mathsf{qdr}\rangle$;*

  2. *$\mathcal{C}$ and $\mathcal{S}$ communicate via (Ξ, Ψ) and produce the quantum communication transcript $|\mathsf{qcom}\rangle$.*

The same considerations about soundness hold as in the classical case.

QORAM Security

We now look at the security model for QORAMs. As in the classical model, security will be given in terms of adaptive access pattern indistinguishability.

Our threat model considers a quantum adversary $\mathcal{A}$, which we identify as $\mathcal{S}$ himself, and who wants to compromise $\mathcal{C}$'s privacy by having access to the communication channel (Ξ, Ψ) and $\mathcal{S}$'s internal memory, but who is not allowed to modify the content of the channel against the soundness of the protocol. Without loss of generality, we assume that the only meaningful changes in the database area $\mathcal{S}.|\texttt{QDB}\rangle$ only happen between the beginning and the end of a $\mathsf{QAccess}$ execution.

As it often happens in the quantum world, there is a caveat here: it is unclear what a 'honest-but-curious' quantum adversary is. In fact, the problem is even more general: we do not have a notion of 'read-only' for quantum channels, as the mere act of observing the data in transit through Ψ can destroy such data. For example, suppose that a quantum state φ is sent through Ψ. Because of the No-Cloning Theorem, $\mathcal{S}$ cannot store a local copy of φ; at the same time, measuring φ in transit through Ψ without any knowledge of such state, would disturb it with high probability. Therefore, it seems hard to justify the inclusion of the state φ in the adversarial view (the *quantum access pattern*) of a honest-but-curious quantum adversary.

Nevertheless, it is important to allow the adversary $\mathcal{A}$ to know some information about the quantum state φ. There are many reasons for this choice. First of all, remember that we are defining QORAMs in a very abstract and general way, and the exact details of how the communication and storage of quantum information works is left to the particular QORAM construction. For example, there might be constructions which only use quantum states from a finite, fixed set of orthogonal states, or which only use subsets of quantum states admitting efficient classical representations (and encoding them in a classical way during the communication). Moreover, it might be possible that the adversary $\mathcal{A}$ at some point obtains access to some side-information which allows him to know something about the content of the database or the data transferred in a sound way, e.g., by applying some quantum operation or partial measurement which does not disturb the state too much. As we need to cover all these possibilities, the option of not including at all the quantum data in the access pattern would be too restrictive. On the other hand, the adversary $\mathcal{A}$ should not be able to modify too much (from $\mathcal{C}$'s point of view) any quantum state, as this would go beyond the notion of honest-but-curious adversary usually considered in the ORAM scenario.

We solve this issue by introducing a *safe extractor*. The intuition behind this technique is to allow our adversary to extract any kind of (quantum) information he wants from a certain physical system, *as long as such extraction is hardly noticeable by any other party*. In this case we say that the action of the adversary on the physical system is *computationally undetectable*, meaning that no QPT algorithm can reliably distinguish whether a quantum operation takes place or not by just looking at the processed quantum state, even in presence of auxiliary information such as, e.g., additional entangled registers. More formally we define the following.

Definition 6.23 (Computational Undetectability of Quantum Action)**.** *Let* $\mathfrak{H}_\Lambda, \mathfrak{H}_\Sigma, \mathfrak{H}_{\mathsf{Env}}$ *be Hilbert spaces of dimension polynomial in* 2^n *associated to quantum register* $\Lambda, \Sigma, \mathsf{Env}$ *respectively, and let* φ_Σ *be an arbitrary quantum state on register* Σ*. A quantum algorithm* $\mathcal{B} : \mathfrak{D}\left(\mathfrak{H}_\Lambda \otimes \mathfrak{H}_\Sigma\right) \to \mathfrak{D}\left(\mathfrak{H}_\Lambda \otimes \mathfrak{H}_\Sigma\right)$ *acting on registers* Λ *and* Σ *has* computationally undetectable action on φ_Σ *iff for any bipartite state* $\varphi_{\Sigma\mathsf{Env}}$ *such that* $(\varphi_{\Sigma\mathsf{Env}})_\Sigma = \varphi_\Sigma$*, and for any* QPT *algorithm* $\mathcal{D}$ *acting on registers* Σ *and* Env *and outputting* 0 *or* 1*, it holds:*

$$\left|\Pr\left[\mathcal{D}\left(\varphi_{\Sigma\mathsf{Env}}\right) \to 1\right] - \Pr\left[\mathcal{D}\left(\left(\mathcal{B} \otimes \mathbb{I}_{\mathfrak{H}_{\mathsf{Env}}}\right)\left(|0\rangle\langle 0|_\Lambda \otimes \varphi_{\Sigma\mathsf{Env}}\right)_{\Sigma\mathsf{Env}}\right) \to 1\right]\right| \leq \mathsf{negl}.$$

Definition 6.24 (Safe Extractor)**.** *Let* $\varphi_\Sigma \in \mathfrak{D}\left(\mathfrak{H}_\Sigma\right)$ *be the quantum state contained in a quantum register* Σ*. A* safe extractor *for* Σ *in the state* φ_Σ *is a* QPT *algorithm* $\mathcal{B}$ *with additional classical input* x *of size polynomial in* n*, acting on* Σ *and outputting a quantum state* ψ *of qubit size polynomial in* n*, and such that the action of* $\mathcal{B}$ *on* φ_Σ *is computationally undetectable.*

Notice that Definition 6.24 depends on the state contained in the quantum register considered. That is, $\mathcal{B}$ might be a safe extractor for a given quantum register if that register is in a certain state, but not in a different one. Of course one could define $\mathcal{B}$ to be a safe extractor for a register *'tout-court'* if it is a safe extractor for *any* state of that register according to Definition 6.24, but this would considerably reduce the power of the adversary. Instead, this definition allows the adversary to use $\mathcal{B}$ adaptively, only at certain points of his execution, when he knows that the action of $\mathcal{B}$ on the current state of the QORAM will be computationally undetectable. The additional classic input to $\mathcal{B}$ serves a useful purpose here, as it can be seen as a way for the adversary to communicate instructions to $\mathcal{B}$ about how to perform the extraction in a safe way (for example, $\mathcal{A}$ might encode a certain measurement basis through this classical input.) With abuse of notation, and without loss of generality, we will write $\psi \leftarrow \mathcal{B}(|\mathsf{qcom}\rangle, \mathcal{S}.|\mathtt{QDB}\rangle)$ to denote that $\mathcal{B}$ performs the following:

- as a classical input, $\mathcal{B}$ gets the classical part of a quantum communication transcript $|\mathsf{qcom}\rangle$ (that is, the content of the classical channel Ξ) and additional classical information by the adversary $\mathcal{A}$;
- $\mathcal{B}$ acts on the quantum registers Ψ and $\mathcal{S}.|\mathsf{QDB}\rangle$;
- finally, $\mathcal{B}$ produces a quantum output ψ.

The intuition of a safe extractor is that we need a way to formalize the (quantum or classical) information that an adversary is able to extract by observing the changes in the quantum database and communication channel. However, we still require that such extraction does not lead to a meaningful deviation from the 'regular' execution of the QORAM protocol. Computational undetectability of quantum action is a *strong* guarantee, because if such action is undetectable, in particular it means that such action cannot modify the QORAM soundness. The converse does not hold: it might be the case that an adversary manipulates the quantum channel or database in such a way that it is *theoretically possible* to detect this manipulation (for some distinguisher $\mathcal{D}$), but *not* for any QORAM client, and therefore the QORAM soundness would be still preserved. However, for our purposes the above restriction on the power of the QORAM adversary is sufficient to define meaningful notions of security, and it is analogous to the (classical) restriction of a honest-but-curious adversary in the ORAM case commonly used in the literature.

More formally, we define a QORAM adversary as follows.

Definition 6.25 (QORAM Adversary)**.** *Let $\mathfrak{H}_{|\mathsf{QDB}\rangle}, \mathfrak{H}_\Psi, \mathfrak{H}_\Lambda$ be complex Hilbert spaces associated to quantum registers $|\mathsf{QDB}\rangle$ (the quantum database), Ψ (the quantum communication channel) and Λ (the quantum access pattern register). A* QORAM adversary *is a* QPT *algorithm $\mathcal{A}^{\mathcal{B}}$ with quantum oracle access to a CPTP map $\mathcal{B} : \Xi \times \mathfrak{D}\left(\mathfrak{H}_{|\mathsf{QDB}\rangle} \otimes \mathfrak{H}_\Psi\right) \to \mathfrak{D}\left(\mathfrak{H}_\Lambda\right)$, such that:*

1. *$\mathcal{B}$ is a* safe extractor *for the joint register $(|\mathsf{QDB}\rangle, \Psi)$ for any of its states during any invocation of $\mathcal{B}$ by $\mathcal{A}$;*
2. *$\mathcal{A}^{\mathcal{B}}$ is computationally indistinguishable from an honest server $\mathcal{S}$ for every QORAM client $\mathcal{C}$.*

As already discussed notice that, in the definition above, conditions 1 and 2 are independent: if $\mathcal{B}$ is *not* a safe extractor during the execution, it means that there exists *some* quantum distinguisher $\mathcal{D}$ able to detect $\mathcal{B}$'s action on the joint register $(|\mathsf{QDB}\rangle, \Psi)$, but $\mathcal{A}^{\mathcal{B}}$ might still remain indistinguishable from an honest server for any honest quantum client. On the other hand, $\mathcal{A}$ might be a misbehaving adversary which deviates 'too much' from the execution of an honest server (and therefore might compromise the QORAM's soundness), even if $\mathcal{B}$ behaves always as a safe extractor. For a meaningful notion of security akin to the **QS**0 case, we require that a QORAM adversary respects both conditions.

We are now able to define *quantum access patterns*, as the outputs of the safe extractor before and after the execution of a quantum data request.

Definition 6.26 (Quantum Access Pattern)**.** *Given QORAM client and server $\mathcal{C}$ and $\mathcal{S}$, a quantum data request $|\mathsf{qdr}\rangle$, and a QORAM adversary $\mathcal{A} = \mathcal{A}^{\mathcal{B}}$, the* quantum access pattern observed by $\mathcal{A}$*, denoted by $|\mathsf{qap}\rangle_{\mathcal{A}}(|\mathsf{qdr}\rangle)$, is the pair of quantum states (ψ, ψ'), where:*

- $\psi \leftarrow \mathcal{B}(|\mathsf{qcom}\rangle, \mathcal{S}.|\mathit{QDB}\rangle)$*;*
- $(\mathcal{C}', \mathcal{S}', |\mathsf{qcom}\rangle') \leftarrow \mathsf{QAccess}(\mathcal{C}, \mathcal{S}, |\mathsf{qdr}\rangle)$
- $\psi' \leftarrow \mathcal{B}(|\mathsf{qcom}\rangle', \mathcal{S}'.|\mathit{QDB}\rangle)$*.*

Notice that, since the action of the safe extractor is computationally undetectable, running it on two consecutive quantum data requests does not allow, in any case, to clone arbitrary quantum states. We define the new security game as follows.

Experiment 6.27 ($\mathsf{Game}^{\mathsf{QAP-IND-CQA}}_{\mathsf{QORAM},\mathcal{A}^{\mathcal{B}}}$)**.** *Let* $\mathsf{QORAM} = (\mathsf{QInit}, \mathsf{QAccess})$ *be a QORAM construction with parameters $(n_{\mathsf{Max}}, n_{\mathsf{dat}}, \mathcal{E})$, n a security parameter and $\mathcal{A} = \mathcal{A}^{\mathcal{B}}$ a QORAM adversary. The* computational indistinguishability of quantum access patterns under adaptive chosen query attack game $\mathsf{Game}^{\mathsf{QAP-IND-CQA}}_{\mathsf{QORAM},\mathcal{A}^{\mathcal{B}}}$ *proceeds as follows:*

```
Input: $n \in \mathbb{N}$
$\mathcal{A} \to (\mathcal{A}_0, |\mathsf{qdr}\rangle_1, n_{\mathsf{db}} \leq n_{\mathsf{Max}})$
$(\mathcal{C}_0, \mathcal{S}_0) \leftarrow \mathsf{QInit}(n, n_{\mathsf{db}})$
loop for $i = 1, \dots, q_1 \in \mathbb{N}$:                    ▷ first quantum CQA phase
    $\mathsf{QAccess}(\mathcal{C}_{i-1}, \mathcal{S}_{i-1}, |\mathsf{qdr}\rangle_i) \to (\mathcal{C}_i, \mathcal{S}_i, |\mathsf{qap}\rangle_i)$
    $\mathcal{A}_{i-1}(|\mathsf{qap}\rangle_i, \mathcal{S}_i) \to (\mathcal{A}_i, |\mathsf{qdr}\rangle_{i+1})$
$\mathcal{A}_{q_1}(|\mathsf{qdr}\rangle_{q_1+1}) \to (\mathcal{A}', |\mathsf{qdr}\rangle^0, |\mathsf{qdr}\rangle^1)$
$b \xleftarrow{\$} \{0,1\}$
$\mathsf{Access}(\mathcal{C}_{q_1}, \mathcal{S}_{q_1}, |\mathsf{qdr}\rangle^b) \to (\mathcal{C}_{q_1+1}, \mathcal{S}_{q_1+1}, |\mathsf{qap}\rangle_{q_1+1})$   ▷ QAP-IND challenge
trace out the quantum data contained in $|\mathsf{qdr}\rangle^{1-b}$
$\mathcal{A}'(\mathsf{ap}_{q_1+1}, \mathcal{S}_{q_1+1}) \to (\mathcal{A}_{q_1+1}, |\mathsf{qdr}\rangle_{q_1+2})$
loop for $i = q_1 + 2, \dots, q_2 \geq q_1 + 2$:               ▷ second quantum CQA phase
    $\mathsf{Access}(\mathcal{C}_{i-1}, \mathcal{S}_{i-1}, |\mathsf{qdr}\rangle_i) \to (\mathcal{C}_i, \mathcal{S}_i, |\mathsf{qap}\rangle_i)$
    $\mathcal{A}_{i-1}(|\mathsf{qap}\rangle_i, \mathcal{S}_i) \to (\mathcal{A}_i, |\mathsf{qdr}\rangle_{i+1})$
$\mathcal{A}_{q_2}(|\mathsf{qdr}\rangle_{q_2+1}) \to b' \in \{0,1\}$
if $b = b'$ then
    Output: 1
else
    Output: 0
```

The advantage of $\mathcal{A}$ *is defined as:*

$$\mathsf{Adv}^{\mathsf{QAP-IND-CQA}}_{\mathsf{QORAM},\mathcal{A}^{\mathcal{B}}} := \Pr\left[\mathsf{Game}^{\mathsf{QAP-IND-CQA}}_{\mathsf{QORAM},\mathcal{A}^{\mathcal{B}}} \to 1\right] - \frac{1}{2}.$$

The idea of the above game follows specularly the classical intuition: the adversary is first allowed to enforce (adaptively) the execution of quantum data requests of his choice, and to observe the related access patterns. Then he issues the challenge query, composed of two different quantum data requests, one of which is executed, and the other discarded. After that, the adversary is allowed another adaptive learning phase, and finally he has to output a bit indicating the challenge data request which was executed. We are now ready to define the security notion for QORAMs.

Definition 6.28 (Quantum Access Pattern Indistinguishability Under Adaptive Chosen Query Attack)**.** *A QORAM construction* QORAM *has* computationally indistinguishable quantum access patterns under adaptive chosen query attack *(or, it is QAP-IND-CQA-secure) iff for any QORAM adversary* $\mathcal{A}^{\mathcal{B}}$ *it holds:* $\mathsf{Adv}^{\mathsf{QAP-IND-CQA}}_{\mathsf{QORAM},\mathcal{A}^{\mathcal{B}}} \leq \mathsf{negl}$.

PathQORAM

In this section we describe the construction for a novel QAP-IND-CQA-secure QORAM scheme, which we call ***PathQORAM***, and which has the interesting property that read and write operations are *inherently equivalent*. The idea is to modify `PathORAM` with the SKQES from Construction 6.13, but we need some additional care for ensuring soundness. In fact, we have the following problem. Suppose the client issues a quantum data request for block i. This will be translated to a leaf in $\mathcal{S}$'s quantum database, and the resulting tree branch $|\texttt{QBranch}\rangle$ will be sent to $\mathcal{C}$. Now $\mathcal{C}$ knows that the data he is looking for is encoded in one of $|\texttt{QBranch}\rangle$'s nodes, but he does not know which one. Classically, $\mathcal{C}$ would proceed by decrypting and inspecting every node in $|\texttt{QBranch}\rangle$ until he finds what he is looking for, then he would perform some operation on that element, before re-encrypting it again, and then complete the re-randomization of $|\texttt{QBranch}\rangle$ before re-sending the whole branch to $\mathcal{S}$. This operation might be problematic in the quantum world though: inspecting an unknown quantum state will destroy it with high probability. We have therefore to find a way to signal $\mathcal{C}$ when he reaches the right node in the path without disturbing the quantum data unit itself.

The solution is to notice that, in our formalization of PathORAM, the client stores the classical identifier i together with the data unit in the block. In the quantum version `PathQORAM`, this identifier is still classical, and of a fixed length $n_{\mathtt{tag}}$. Once a node in $|\texttt{QBranch}\rangle$ is decrypted, it will be transformed to $|i\rangle\langle i| \otimes \varphi$. The first register can then be measured in the computational basis without being disturbed, and without disturbing the state φ (which is not entangled with $|i\rangle$). So the trick for $\mathcal{C}$ is to find out when he is decrypting the right element by *only* measuring the first n_{tag} qubits of the decrypted block, and then only act on the quantum data unit when the right identifier is found. Notice how other different approaches used classically to instantiate PathORAM, such as identifying blocks by storing a local table with the hash values of the data units, might not work so smoothly when translated to the quantum world.

More concretely, we give here a full description of PathQORAM (which from now on we denote as `PathQORAM`) according to our new formalism. The meaning of the parameters is as in Definition 3.63.

Construction 6.29 (`PathQORAM` [GKK17, Definition 36])**.** *For fixed parameters $n_{\mathsf{dat}}, n_{\mathsf{Max}} \in \mathbb{N}$, let $n_{\mathsf{tag}} = \lceil \log_2 n_{\mathsf{Max}} \rceil, n_{\mathsf{bkt}} \in \mathbb{N}, n_{\mathsf{msg}} = n_{\mathsf{dat}} + n_{\mathsf{tag}}$, and $n_{\mathsf{blk}} \geq n_{\mathsf{msg}}$. Let $\mathcal{G}$ be a pqPRNG outputting n_{tag}-bit pseudorandom values, and let $\mathcal{E} = (\mathsf{KGen}, \mathsf{QEnc}, \mathsf{QDec})$ be a QIND-CPA SKQES with n_{msg}-qubit plaintexts and n_{blk}-qubit ciphertexts. We define a QORAM construction called* `PathQORAM` $=$ `PathQORAM`$_{\mathcal{E},\mathcal{G}}$ *as follows:*

- $\mathsf{QInit}(n, n_{\mathsf{db}}) \to (\mathcal{C}, \mathcal{S})$ *in the following way:*

```
$\mathcal{C}$ generates a secret key $k \leftarrow \mathsf{KGen}$
set $n_{\mathsf{tree}} := \lceil \log_2 n_{\mathsf{db}} \rceil$
$\mathcal{C}$ initializes a lookup table (the position map) of the form
  $((1, r_1), \dots, (n_{\mathsf{db}}, r_{n_{\mathsf{db}}}))$, where $r_i$ are $n_{\mathsf{tree}}$-bit values generated by
  truncating the first $n_{\mathsf{tag}} - n_{\mathsf{tree}}$ bits of $\mathcal{G}$'s output
$\mathcal{S}.|\mathtt{QDB}\rangle$ is stored in a binary tree of height $n_{\mathsf{tree}}$, with root $|\mathtt{QRoot}\rangle$
  and leaves $|\mathtt{QLeaf}\rangle_0, \dots, |\mathtt{QLeaf}\rangle_{2^{n_{\mathsf{tree}}}-1}$, and such that:
    1. each node of the tree stores up to $n_{\mathsf{bkt}}$ quantum blocks;
    2. every quantum block of every node is initialized
       to $\mathsf{QEnc}_k(|0^{n_{\mathsf{msg}}}\rangle\langle 0^{n_{\mathsf{msg}}}|)$.
```

- *If* $|\mathsf{qdr}\rangle = (\mathsf{op}, i, \varphi)$, *then* $\mathsf{QAccess}(\mathcal{C}, \mathcal{S}, |\mathsf{qdr}\rangle) \to (\mathcal{C}', \mathcal{S}', |\mathsf{qcom}\rangle)$ *in the following way:*

```
$\mathcal{C}$ reads $r_i$ from his position map and sends it to $\mathcal{S}$
$\mathcal{S}$ sends to $\mathcal{C}$ the quantum system containing the path $|\mathtt{QBranch}\rangle$ from
  $|\mathtt{QRoot}\rangle$ to $|\mathtt{QLeaf}\rangle_{r_i}$
remap $(i, r_i)$ to $(i, r_i')$ in the position map of $\mathcal{C}$, where $r_i'$ is a fresh
  pseudorandom $n_{\mathsf{tree}}$-bit value (generated by truncating the first
  $n_{\mathsf{tag}} - n_{\mathsf{tree}}$ bits of $\mathcal{G}$'s output), obtaining $\mathcal{C}'$
for all quantum block $\psi$ contained in $|\mathtt{QBranch}\rangle$ do
    $\mathcal{C}'$ decrypts $\mathsf{QDec}_k(\psi) \to (|j\rangle\langle j| \otimes \sigma)$,
      where $|j\rangle \in \mathfrak{H}_{n_{\mathsf{tag}}}$, and $\sigma \in \mathfrak{D}(\mathfrak{H}_{n_{\mathsf{dat}}})$
    $\mathcal{C}'$ measures the first $n_{\mathsf{tag}}$ qubits of the decrypted state in the
      computational basis, obtaining $j$
```

```
7:      if $j = i$ then
8:          swap $\sigma$ with $\varphi$
9:      $\mathcal{C}'$ re-encrypts (re-randomizing) the current quantum block,
          obtaining $\psi'$
10:     find in $|QBranch\rangle$ the common parent node $|QNode\rangle$ between
          $|QLeaf\rangle_{r_i}$ and $|QLeaf\rangle_{r_j}$, closer to the leaf level
11:     set $b_{\mathsf{swap}} :=$ 'false'
12:     for all $\rho$ in $|QNode\rangle$ do
13:         $\mathcal{C}'$ decrypts $\mathsf{QDec}_k(\rho) \to (|j'\rangle\langle j'| \otimes \sigma')$
14:         $\mathcal{C}'$ re-encrypts (re-randomizing) $\rho' \leftarrow \mathsf{QEnc}_k(|j'\rangle\langle j'| \otimes \sigma')$
15:         if $j' = 0 \ldots 0$ then                    ▷ $\rho'$ is empty, can be used
16:             swap $\psi'$ and $\rho'$
17:             set $b_{\mathsf{swap}} :=$ 'true'
18:     if $b_{\mathsf{swap}} =$ 'false' then        ▷ no empty blocks in current $|QNode\rangle$
19:         if $|QNode\rangle \neq |QRoot\rangle$ then
20:             set $|QNode\rangle$ to be one level up in the tree
21:             go to step 12
22:         else
23:             store the current quantum block in the $|QStash\rangle$
24: $\mathcal{C}'$ sends back the updated tree branch, $|NewQBranch\rangle$, to $\mathcal{S}$
25: update $\mathcal{S}.|QDB\rangle$ with $|NewQBranch\rangle$, obtaining $\mathcal{S}'$
26: produce $|\mathsf{qcom}\rangle$, which contains $r_i, |QBranch\rangle, |NewQBranch\rangle$
```

Notice that the following interesting property holds: the operations of 'write' and 'read' have the *same* effect. Namely: since qubits from the server's database cannot be copied, and cannot be removed or added (otherwise this would compromise indistinguishability), the action of a read or write operation is simply to swap a state in the database with a state in $\mathcal{C}$'s memory. In fact, QAccess swaps φ known by $\mathcal{C}$ with σ stored in $\mathcal{S}$. Also notice how $|\mathsf{qcom}\rangle$ containing $|\mathtt{QBranch}\rangle, |\mathtt{NewQBranch}\rangle$ would imply a cloning of quantum states. This is just a formal artifice, because in the case of QORAMs as we defined them, $|\mathsf{qcom}\rangle$ is only used in respect to a safe extractor $\mathcal{B}$, which processes $|\mathtt{NewQBranch}\rangle$ only after $\mathcal{C}$ has processed $|\mathtt{QBranch}\rangle$, so information is never copied. For the soundness of the `PathQORAM` construction we have left unexplained the use of a *quantum stash* $|QStash\rangle$. This is an area of quantum memory basically used as the classical stash of `PathORAM`, but every time an element is 'written' in the stash, it is actually 'swapped' with an empty block in the tree. The security of the construction follows from the QIND-CPA security of the SKQES $\mathcal{E}$, and from the security of the pqPRNG $\mathcal{G}$.

Theorem 6.30 ([GKK17, Theorem 34])**.** *Let $\mathcal{E}$ be a QIND-CPA SKQES, and let $\mathcal{G}$ be a pqPRNG. Then, `PathQORAM` instantiated using $\mathcal{E}$ and $\mathcal{G}$ is a QAP-IND-CQA secure QORAM.*

Proof. The proof follows step-by-step the proof of Theorem 3.64 with some important differences. First of all, $\mathcal{D}$ cannot store a local mirrored tree of plaintexts of the form $(|i\rangle\langle i| \otimes \sigma)$ because of the No-Cloning Theorem, so he cannot simulate $\mathcal{C}$ perfectly. But he can store a mirrored tree which contains *only* the classical identifiers i, at the right positions of every block throughout the execution of the protocol.

At this point, $\mathcal{D}$ can simulate a decryption oracle for a certain block ψ in a downloaded branch by fetching the cleartext identifier i found at the corresponding position in the 'mirrored' tree, and creating a 'simulated' plaintext of the form $(|i\rangle\langle i| \otimes |0^{n_{\mathsf{dat}}}\rangle\langle 0^{n_{\mathsf{dat}}}|)$, i.e., replacing the 'real' quantum data unit σ with a zero state. Since $\mathcal{A}$ never 'sees' a decrypted block, this substitution is not immediately apparent to him. Moreover, whenever $\mathcal{C}$ would create a block by encrypting $\psi \leftarrow \mathsf{QEnc}_k(|i\rangle\langle i| \otimes \sigma)$, $\mathcal{D}$ can simulate this by doing $\psi \leftarrow \mathsf{QEnc}_k(|i\rangle\langle i| \otimes |0^{n_{\mathsf{dat}}}\rangle\langle 0^{n_{\mathsf{dat}}}|)$. By the QIND-CPA security of $\mathcal{E}$, $\mathcal{A}$ cannot detect this substitution with more than negligible advantage over guessing. Therefore, $\mathcal{D}$ can still simulate $\mathcal{C}$ (with overwhelming, albeit not 100%, probability) at any data request.

Another issue appears during the challenge phase, as this time the concept of *non-meaningful* challenge must be redefined. For the same argument as above, from $\mathcal{A}$'s perspective it does not matter whether two data requests lead to two 'different' quantum data units σ^0, σ^1 (the analogue of data units $\mathsf{data}^0, \mathsf{data}^1$ in the classical proof) or not. Therefore, $\mathcal{D}$ can ignore the quantum data units at all. Moreover, as discussed above, in `PathQORAM` there is no difference between 'read' and 'write' operations. It follows, from the same argument as in the proof of Theorem 3.64, that the two challenge quantum data requests $|\mathsf{qdr}\rangle^0, |\mathsf{qdr}\rangle^1$ must differ on the identifiers i^0, i^1. Then, $\mathcal{D}$ plays the QIND-CPA game with challenge plaintexts $\varphi^a = |i^a\rangle\langle i^a| \otimes |0^{n_{\mathsf{dat}}}\rangle\langle 0^{n_{\mathsf{dat}}}|$ for $a \in \{0, 1\}$, following the same strategy as in the classical case (by guessing a bit, injecting the challenge ciphertext, and observing $\mathcal{A}$'s output), with only a negligible loss in the success probability because he is simulating fake plaintexts. This concludes the proof. □

Bibliography

[Aar09] Scott Aaronson. Quantum copy-protection and quantum money. In *Proceedings of the 24th Annual IEEE Conference on Computational Complexity, CCC 2009, Paris, France, 15-18 July 2009*, pages 229–242, 2009.

[ABF^{+}16] Gorjan Alagic, Anne Broadbent, Bill Fefferman, Tommaso Gagliardoni, Christian Schaffner, and Michael St. Jules. Computational security of quantum encryption. In *Information Theoretic Security - 9th International Conference, ICITS 2016, Tacoma, WA, USA, August 9-12, 2016, Revised Selected Papers*, pages 47–71, 2016.

[AM16] Gorjan Alagic and Christian Majenz. Quantum non-malleability and authentication. *CoRR*, abs/1610.04214, 2016.

[AR16] Gorjan Alagic and Alexander Russell. Quantum-secure symmetric-key cryptography based on hidden shifts. *IACR Cryptology ePrint Archive*, 2016:960, 2016.

[ARTL15] Tameen Albash, Troels F. Rønnow, Matthias Troyer, and Daniel A. Lidar. Reexamining classical and quantum models for the D-Wave One processor. *The European Physical Journal Special Topics*, 224(1):111–129, 2015.

[AMTdW00] Andris Ambainis, Michele Mosca, Alain Tapp, and Ronald de Wolf. Private quantum channels. In *41st Annual Symposium on Foundations of Computer Science, FOCS 2000, 12-14 November 2000, Redondo Beach, California, USA*, pages 547–553, 2000.

[ARU14] Andris Ambainis, Ansis Rosmanis, and Dominique Unruh. Quantum attacks on classical proof systems: The hardness of quantum rewinding. In *55th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2014, Philadelphia, PA, USA, October 18-21, 2014*, pages 474–483, 2014.

[ATTU16] Mayuresh Vivekanand Anand, Ehsan Ebrahimi Targhi, Gelo Noel Tabia, and Dominique Unruh. Post-quantum security of the CBC, CFB, OFB, CTR, and XTS modes of operation. In *Post-Quantum Cryptography - 7th International Workshop, PQCrypto 2016, Fukuoka, Japan, February 24-26, 2016, Proceedings*, pages 44–63, 2016.

[AB09] Sanjeev Arora and Boaz Barak. *Computational Complexity - A Modern Approach.* Cambridge University Press, 2009.

[BCG+02a] Howard Barnum, Claude Crépeau, Daniel Gottesman, Adam D. Smith, and Alain Tapp. Authentication of quantum messages. In *43rd Symposium on Foundations of Computer Science (FOCS 2002), 16-19 November 2002, Vancouver, BC, Canada, Proceedings*, pages 449–458, 2002.

[BCG+02b] Howard Barnum, Claude Crépeau, Daniel Gottesman, Adam D. Smith, and Alain Tapp. Authentication of quantum messages. In *43rd Symposium on Foundations of Computer Science (FOCS 2002), 16-19 November 2002, Vancouver, BC, Canada, Proceedings*, pages 449–458, 2002.

[Bel98] Mihir Bellare. Practice-oriented provable security. In *Lectures on Data Security, Modern Cryptology in Theory and Practice, Summer School, Aarhus, Denmark, July 1998*, pages 1–15, 1998.

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *CCS '93, Proceedings of the 1st ACM Conference on Computer and Communications Security, Fairfax, Virginia, USA, November 3-5, 1993.*, pages 62–73, 1993.

[BBBV97] Charles H. Bennett, Ethan Bernstein, Gilles Brassard, and Umesh V. Vazirani. Strengths and weaknesses of quantum computing. *SIAM J. Comput.*, 26(5):1510–1523, 1997.

[BB14] Charles H. Bennett and Gilles Brassard. Quantum cryptography: Public key distribution and coin tossing. *Theor. Comput. Sci.*, 560:7–11, 2014.

[BBD09] Daniel J. Bernstein, Johannes Buchmann, and Erik Dahmen. *Post-Quantum Cryptography.* Springer-Verlag Berlin Heidelberg, 2009.

[BHH+15] Daniel J. Bernstein, Daira Hopwood, Andreas Hülsing, Tanja Lange, Ruben Niederhagen, Louiza Papachristodoulou, Michael Schneider, Peter Schwabe, and Zooko Wilcox-O'Hearn. SPHINCS: practical stateless hash-based signatures. In *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part I*, pages 368–397, 2015.

[BV97] Ethan Bernstein and Umesh V. Vazirani. Quantum complexity theory. *SIAM J. Comput.*, 26(5):1411–1473, 1997.

[Bon98] Dan Boneh. The decision Diffie-Hellman problem. In *Algorithmic Number Theory, Third International Symposium, ANTS-III, Portland, Oregon, USA, June 21-25, 1998, Proceedings*, pages 48–63, 1998.

[BDF+11] Dan Boneh, Özgür Dagdelen, Marc Fischlin, Anja Lehmann, Christian Schaffner, and Mark Zhandry. Random oracles in a quantum world. In *Advances in Cryptology - ASIACRYPT 2011 - 17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea, December 4-8, 2011. Proceedings*, pages 41–69, 2011.

[BZ13a] Dan Boneh and Mark Zhandry. Quantum-secure message authentication codes. In *Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings*, pages 592–608, 2013.

[BZ13b] Dan Boneh and Mark Zhandry. Secure signatures and chosen ciphertext security in a quantum computing world. In *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part II*, pages 361–379, 2013.

[BCD+16] Joppe W. Bos, Craig Costello, Léo Ducas, Ilya Mironov, Michael Naehrig, Valeria Nikolaenko, Ananth Raghunathan, and Douglas Stebila. Frodo: Take off the ring! practical, quantum-secure key exchange from LWE. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, October 24-28, 2016*, pages 1006–1018, 2016.

[BR03] P. Oscar Boykin and Vwani Roychowdhury. Optimal encryption of quantum bits. *Phys. Rev. A*, 67:042317, Apr 2003.

[BHT98] Gilles Brassard, Peter Høyer, and Alain Tapp. *Quantum cryptanalysis of hash and claw-free functions*, pages 163–169. Springer Berlin Heidelberg, Berlin, Heidelberg, 1998.

[BJ15] Anne Broadbent and Stacey Jeffery. Quantum homomorphic encryption for circuits of low T-gate complexity. In *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part II*, pages 609–629, 2015.

[BS16] Anne Broadbent and Christian Schaffner. Quantum cryptography beyond quantum key distribution. *Des. Codes Cryptography*, 78(1):351–382, 2016.

[BFM15] Christina Brzuska, Pooya Farshim, and Arno Mittelbach. Random-oracle uninstantiability from indistinguishability obfuscation. In *Theory of Cryptography - 12th Theory of Cryptography Conference, TCC 2015, Warsaw, Poland, March 23-25, 2015, Proceedings, Part II*, pages 428–455, 2015.

[CNR12] Jan Camenisch, Gregory Neven, and Markus Rückert. Fully anonymous attribute tokens from lattices. In *Security and Cryptography for Networks - 8th International Conference, SCN 2012, Amalfi, Italy, September 5-7, 2012. Proceedings*, pages 57–75, 2012.

[CGH98] Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited (preliminary version). In *Proceedings of the Thirtieth Annual ACM Symposium on the Theory of Computing, Dallas, Texas, USA, May 23-26, 1998*, pages 209–218, 1998.

[CEJvO02] Stanley Chow, Philip A. Eisen, Harold Johnson, and Paul C. van Oorschot. White-box cryptography and an AES implementation. In *Selected Areas in Cryptography, 9th Annual International Workshop, SAC 2002, St. John's, Newfoundland, Canada, August 15-16, 2002. Revised Papers*, pages 250–270, 2002.

[DFG13] Özgür Dagdelen, Marc Fischlin, and Tommaso Gagliardoni. The Fiat-Shamir transformation in a quantum world. In *Advances in Cryptology - ASIACRYPT 2013 - 19th International Conference on the Theory and Application of Cryptology and Information Security, Bengaluru, India, December 1-5, 2013, Proceedings, Part II*, pages 62–81, 2013.

[DFNS13] Ivan Damgaard, Jakob Funder, Jesper Buus Nielsen, and Louis Salvail. Superposition attacks on cryptographic protocols. In *Information Theoretic Security - 7th International Conference, ICITS 2013, Singapore, November 28-30, 2013, Proceedings*, pages 142–161, 2013.

[DH76] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Trans. Information Theory*, 22(6):644–654, 1976.

[DFPR14] Vedran Dunjko, Joseph Fitzsimons, Christopher Portmann, and Renato Renner. Composable security of delegated quantum computation. In *Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C., December 7-11, 2014, Proceedings, Part II*, pages 406–425, 2014.

[ES15] Edward Eaton and Fang Song. Making existential-unforgeable signatures strongly unforgeable in the quantum random-oracle model. In *10th Conference on the Theory of Quantum Computation, Communication and Cryptography, TQC 2015, May 20-22, 2015, Brussels, Belgium*, pages 147–162, 2015.

[FJP14] Luca De Feo, David Jao, and Jérôme Plût. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. *J. Mathematical Cryptology*, 8(3):209–247, 2014.

[Fey82] Richard P. Feynman. Simulating physics with computers. *International Journal of Theoretical Physics*, 21(6):467–488, 1982.

[FS86] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *Advances in Cryptology - CRYPTO '86, Santa Barbara, California, USA, 1986, Proceedings*, pages 186–194, 1986.

[GHS16] Tommaso Gagliardoni, Andreas Hülsing, and Christian Schaffner. Semantic security and indistinguishability in the quantum world. In *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part III*, pages 60–89, 2016.

[GKK17] Tommaso Gagliardoni, Nikolaos P. Karvelas, and Stefan Katzenbeisser. ORAMs in a quantum world. *IACR Cryptology ePrint Archive*, 2017.

[Gam84] Taher El Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *Advances in Cryptology, Proceedings of CRYPTO '84, Santa Barbara, California, USA, August 19-22, 1984, Proceedings*, pages 10–18, 1984.

[GMP16] Sanjam Garg, Payman Mohassel, and Charalampos Papamanthou. TWORAM: efficient oblivious RAM in two rounds with applications to searchable encryption. In *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part III*, pages 563–592, 2016.

[GYZ16] Sumegha Garg, Henry Yuen, and Mark Zhandry. New security notions and feasibility results for authentication of quantum data. *CoRR*, abs/1607.07759, 2016.

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*, pages 197–206, 2008.

[Gol01] Oded Goldreich. *The Foundations of Cryptography - Volume 1, Basic Techniques.* Cambridge University Press, 2001.

[Gol04] Oded Goldreich. *The Foundations of Cryptography - Volume 2, Basic Applications.* Cambridge University Press, 2004.

[Gol11] Oded Goldreich. In a world of P=BPP. In *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation - In Collaboration with Lidor Avigad, Mihir Bellare, Zvika Brakerski, Shafi Goldwasser, Shai Halevi, Tali Kaufman, Leonid Levin, Noam Nisan, Dana Ron, Madhu Sudan, Luca Trevisan, Salil Vadhan, Avi Wigderson, David Zuckerman*, pages 191–232. 2011.

[GGH97] Oded Goldreich, Shafi Goldwasser, and Shai Halevi. Public-key cryptosystems from lattice reduction problems. In *Advances in Cryptology - CRYPTO '97, 17th Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 1997, Proceedings*, pages 112–131, 1997.

[GGM84] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions (extended abstract). In *25th Annual Symposium on Foundations of Computer Science, West Palm Beach, Florida, USA, 24-26 October 1984*, pages 464–479, 1984.

[GL89] Oded Goldreich and Leonid A. Levin. A hard-core predicate for all one-way functions. In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing, May 14-17, 1989, Seattle, Washigton, USA*, pages 25–32, 1989.

[GMW86] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to prove all np-statements in zero-knowledge, and a methodology of cryptographic protocol design. In *Advances in Cryptology - CRYPTO '86, Santa Barbara, California, USA, 1986, Proceedings*, pages 171–185, 1986.

[GO96] Oded Goldreich and Rafail Ostrovsky. Software protection and simulation on oblivious RAMs. *J. ACM*, 43(3):431–473, 1996.

[Gro96] Lov K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing, Philadelphia, Pennsylvania, USA, May 22-24, 1996*, pages 212–219, 1996.

[GdAJ13] Elloá B. Guedes, Francisco Marcos de Assis, and Bernardo Lula Jr. Quantum attacks on pseudorandom generators. *Mathematical Structures in Computer Science*, 23(3):608–634, 2013.

[GQ88] Louis C. Guillou and Jean-Jacques Quisquater. A "paradoxical" indentity-based signature scheme resulting from zero-knowledge. In *Advances in Cryptology - CRYPTO '88, 8th Annual International Cryptology Conference, Santa Barbara, California, USA, August 21-25, 1988, Proceedings*, pages 216–231, 1988.

[HILL99] Johan Haastad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4):1364–1396, 1999.

[HSS11] Sean Hallgren, Adam D. Smith, and Fang Song. Classical cryptographic protocols in a quantum world. In *Advances in Cryptology - CRYPTO 2011 - 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings*, pages 411–428, 2011.

[IR88] Russell Impagliazzo and Steven Rudich. Limits on the provable consequences of one-way permutations. In *Advances in Cryptology - CRYPTO '88, 8th Annual International Cryptology Conference, Santa Barbara, California, USA, August 21-25, 1988, Proceedings*, pages 8–26, 1988.

[JMV01] Don Johnson, Alfred Menezes, and Scott A. Vanstone. The elliptic curve digital signature algorithm (ECDSA). *Int. J. Inf. Sec.*, 1(1):36–63, 2001.

[Jul17] Michael St. Jules. Secure quantum encryption. Master's thesis, Master of Science in Mathematics, School of Graduate Studies and Research, University of Ottawa, Canada, 2017.

[KLLN16] Marc Kaplan, Gaëtan Leurent, Anthony Leverrier, and María Naya-Plasencia. Breaking symmetric cryptosystems using quantum period finding. In *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part II*, pages 207–237, 2016.

[KKVB02] Elham Kashefi, Adrian Kent, Vlatko Vedral, and Konrad Banaszek. Comparison of quantum oracles. *Phys. Rev. A*, 65:050304, May 2002.

[KL07] Jonathan Katz and Yehuda Lindell. *Introduction to Modern Cryptography.* Chapman and Hall/CRC Press, 2007.

[KPG99] Aviad Kipnis, Jacques Patarin, and Louis Goubin. Unbalanced oil and vinegar signature schemes. In *Advances in Cryptology - EUROCRYPT '99, International Conference on the Theory and Application of Cryptographic Techniques, Prague, Czech Republic, May 2-6, 1999, Proceeding*, pages 206–222, 1999.

[KM10] Hidenori Kuwakado and Masakatu Morii. Quantum distinguisher between the 3-round Feistel cipher and the random permutation. In *IEEE International Symposium on Information Theory, ISIT 2010, June 13-18, 2010, Austin, Texas, USA, Proceedings*, pages 2682–2685, 2010.

[KM12] Hidenori Kuwakado and Masakatu Morii. Security on the quantum-type Even-Mansour cipher. In *Proceedings of the International Symposium on Information Theory and its Applications, ISITA 2012, Honolulu, HI, USA, October 28-31, 2012*, pages 312–316, 2012.

[Lyu12] Vadim Lyubashevsky. Lattice signatures without trapdoors. In *Advances in Cryptology - EUROCRYPT 2012 - 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, April 15-19, 2012. Proceedings*, pages 738–755, 2012.

[LPR13] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. *J. ACM*, 60(6):43:1–43:35, 2013.

[McE78] Robert J. McEliece. A public-key cryptosystem based on algebraic coding theory. *Deep Space Network Progress Report*, 44:114–116, January 1978.

[Mic11] Daniele Micciancio. Lattice-based cryptography. In *Encyclopedia of Cryptography and Security, 2nd Ed.*, pages 713–715. 2011.

[NC00] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information.* Cambridge University Press, Cambridge, New York, 2000.

[OBK$^+$16] P. J. J. O'Malley, R. Babbush, I. D. Kivlichan, J. Romero, J. R. McClean, R. Barends, J. Kelly, P. Roushan, A. Tranter, N. Ding, B. Campbell, Y. Chen, Z. Chen, B. Chiaro, A. Dunsworth, A. G. Fowler, E. Jeffrey, E. Lucero, A. Megrant, J. Y. Mutus, M. Neeley, C. Neill, C. Quintana, D. Sank, A. Vainsencher, J. Wenner, T. C. White, P. V. Coveney, P. J. Love, H. Neven, A. Aspuru-Guzik, and J. M. Martinis. Scalable quantum simulation of molecular energies. *Phys. Rev. X*, 6:031007, Jul 2016.

[PS00] David Pointcheval and Jacques Stern. Security arguments for digital signatures and blind signatures. *J. Cryptology*, 13(3):361–396, 2000.

[RSA78] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21(2):120–126, 1978.

[SP92] Alfredo De Santis and Giuseppe Persiano. Zero-knowledge proofs of knowledge without interaction (extended abstract). In *33rd Annual Symposium on Foundations of Computer Science, Pittsburgh, Pennsylvania, USA, 24-27 October 1992*, pages 427–436, 1992.

[SS17] Thomas Santoli and Christian Schaffner. Using Simon's algorithm to attack symmetric-key cryptographic primitives. *Quantum Information & Computation*, 17(1&2):65–78, 2017.

[Sch91] Claus-Peter Schnorr. Efficient signature generation by smart cards. *J. Cryptology*, 4(3):161–174, 1991.

[Sha01] Claude E. Shannon. A mathematical theory of communication. *Mobile Computing and Communications Review*, 5(1):3–55, 2001.

[Sho94] Peter W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In *35th Annual Symposium on Foundations of Computer Science, Santa Fe, New Mexico, USA, 20-22 November 1994*, pages 124–134, 1994.

[Sim97] Daniel R. Simon. On the power of quantum computation. *SIAM J. Comput.*, 26(5):1474–1483, 1997.

[Son14] Fang Song. A note on quantum security for post-quantum cryptography. In *Post-Quantum Cryptography - 6th International Workshop, PQCrypto 2014, Waterloo, ON, Canada, October 1-3, 2014. Proceedings*, pages 246–265, 2014.

[SSS12] Emil Stefanov, Elaine Shi, and Dawn Xiaodong Song. Towards practical oblivious RAM. In *19th Annual Network and Distributed System Security Symposium, NDSS 2012, San Diego, California, USA, February 5-8, 2012*, 2012.

[SvDS$^+$13] Emil Stefanov, Marten van Dijk, Elaine Shi, Christopher W. Fletcher, Ling Ren, Xiangyao Yu, and Srinivas Devadas. Path ORAM: an extremely simple oblivious RAM protocol. In *2013 ACM SIGSAC Conference on Computer and Communications Security, CCS'13, Berlin, Germany, November 4-8, 2013*, pages 299–310, 2013.

[SLB$^+$11] D. Stucki, M. Legré, F. Buntschu, B. Clausen, N. Felber, N. Gisin, L. Henzen, P. Junod, G. Litzistorf, P. Monbaron, L. Monat, J.-B. Page, D. Perroud, G. Ribordy, A. Rochas, S. Robyr, J. Tavares, R. Thew, P. Trinkler, S. Ventura, R. Voirol, N. Walenta, and H. Zbinden. Long-term performance of the SwissQuantum quantum key distribution network in a field environment. *New Journal of Physics*, 13(12):123001, December 2011.

[TCM$^+$16] Maika Takita, Antonio D. Córcoles, Easwar Magesan, Baleegh Abdo, Markus Brink, Andrew Cross, Jerry M. Chow, and Jay M. Gambetta. Demonstration of weight-four parity measurements in the surface code architecture. *Phys. Rev. Lett.*, 117:210505, Nov 2016.

[Unr12] Dominique Unruh. Quantum proofs of knowledge. In *Advances in Cryptology - EUROCRYPT 2012 - 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, April 15-19, 2012. Proceedings*, pages 135–152, 2012.

[Unr13] Dominique Unruh. Everlasting multi-party computation. In *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part II*, pages 380–397, 2013.

[Unr15] Dominique Unruh. Non-interactive zero-knowledge proofs in the quantum random oracle model. In *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part II*, pages 755–784, 2015.

[VW16] Thomas Vidick and John Watrous. Quantum proofs. *Foundations and Trends in Theoretical Computer Science*, 11(1-2):1–215, 2016.

[Wat01] John Watrous. Quantum algorithms for solvable groups. In *Proceedings on 33rd Annual ACM Symposium on Theory of Computing, July 6-8, 2001, Heraklion, Crete, Greece*, pages 60–67, 2001.

[Wat06] John Watrous. Zero-knowledge against quantum attacks. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing, Seattle, WA, USA, May 21-23, 2006*, pages 296–305, 2006.

[Wie83] Stephen Wiesner. Conjugate coding. *SIGACT News*, 15(1):78–88, January 1983.

[Yao82] Andrew Chi-Chih Yao. Theory and applications of trapdoor functions (extended abstract). In *23rd Annual Symposium on Foundations of Computer Science, Chicago, Illinois, USA, 3-5 November 1982*, pages 80–91, 1982.

[Zha12a] Mark Zhandry. How to construct quantum random functions. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pages 679–687, 2012.

[Zha12b] Mark Zhandry. Secure identity-based encryption in the quantum random oracle model. In *Advances in Cryptology - CRYPTO 2012 - 32nd Annual Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2012. Proceedings*, pages 758–775, 2012.

[Zha16] Mark Zhandry. A note on quantum-secure PRPs. *CoRR*, abs/1611.05564, 2016.