Towards a Classification of Knowledge-of-Exponent Assumptions in Cyclic Groups

by

Firas Kraiem

Department of Computer and Mathematical Sciences
Graduate School of Information Sciences
Tohoku University

17 July 2019

Submitted to Tohoku University
in partial fulfillment of the requirements for the degree of
Ph.D. (Information Sciences)

Abstract

Inspired by the work of Ghadafi and Groth on a certain type of computational hardness assumptions in cyclic groups (which they call "target assumptions"), we initiate in this thesis an analogous work on another type of hardness assumptions, namely the "knowledge-of-exponent" assumptions (KEAs). Originally introduced by Damgård to construct practical encryption schemes secure against chosen ciphertext attacks, KEAs have subsequently been used primarily to construct succinct non-interactive arguments of knowledge (SNARKs), and proven to be inherent to such constructions.

Using a proof technique first introduced by Bellare and Palacio (but acknowledged by them as being due to Halevi), we first investigate the internal structure of the q-power knowledge-of-exponent (q-PKE) family of assumptions introduced by Groth, which is thus far the most general instance of KEAs in the literature. Namely, we give evidence showing that the assumptions in the q-PKE family appear to increase in strength as the parameter q grows.

We then introduce a generalisation of the q-PKE family, which is again inspired by the "target assumptions" of Ghadafi and Groth. Namely, the univariate monomial parameters that appear in the q-PKE assumptions are replaced by multivariate rational functions. We call the resulting assumptions "rational knowledge-of-exponent assumptions" (RKEAs), and as a first step towards a full classification of this class of assumptions, we show that the rational functions may be replaced by polynomials.

Acknowledgments

First of all, I would like to express my deepest gratitude to my supervisor, Professor Hiroki Shizuya, for his careful guidance during the work that led to this thesis, as well as for supporting me in many other ways. It is absolutely no exaggeration to say that without his support this thesis would not have been possible.

I would also like to thank the other members of the jury, Professors Xiao Zhou and Takuo Suganuma and Associate Professor Shuji Isobe, whose valuable comments and suggestions allowed me to improve this thesis considerably.

Assistant Professor Eisuke Koizumi has been my thesis advisor and also contributed significantly towards its completion. My thanks are due to him as well.

I have also had many interesting discussions with Associate Professor Masao Sakai and Assistant Professor Shingo Hasegawa, and extend my sincere thanks to them.

Of course, I have also enjoyed many services provided by Tohoku University at large; I want to also express my gratitude to all those who made it such a pleasant place for me to work at.

Finally, I would like to thank my mother and my sister, as well as the Kitashoji family, for all their support throughout the years.

Contents

Chapter 1

Introduction

1.1 Background

Cryptography can be broadly defined as the design of systems, called *cryptographic systems* or *cryptographic schemes*, that are capable of maintaining their functionality in the presence of adversarial entities that attempt to make them deviate from their intended behaviour [16]. In the classical cryptographic task of *encryption*, for instance, a cryptographic system called an *encryption scheme* is used by two parties to exchange messages over a public channel in such a way as to make it impossible for any third party to obtain the contents of the exchanged messages. This property, called *privacy* or *confidentiality*, must be maintained regardless of the strategy employed by such a third party in its attempts. In such schemes, one of the parties (the *sender*) first applies a process called *encryption* to the message they wish to send, yielding an "encrypted form" thereof, called a *ciphertext*. The sender then transmits (over the public channel) the ciphertext to the other party (the *receiver*), who applies a process called *decryption* to reverse the encryption process and recover the original message (the *plaintext*). Since the ciphertext is sent over the public channel, any third party can obtain it; privacy requires that even such a third party should not be able to obtain the corresponding plaintext.

A question that immediately arises when considering such schemes is how one should evaluate their "security", *i.e.*, whether and to what extent they satisfy the privacy requirement. The "classical" approach, based on information theory and pioneered by Shannon during the Second World War [28], asserts that such schemes should be considered secure if (and only if) the ciphertexts they produce

do not contain any "information" about the corresponding plaintexts. However, a major drawback of this approach is that any scheme that is "secure" in this sense requires the parties to exchange *a priori* a secret "encryption key" which must be as long as the messages they wish to subsequently exchange securely. This requirement severely limits the applicability of such schemes when the amount of data to be transmitted securely is large, as is routine over the modern Internet.

The "modern" approach, based on computational complexity theory and pioneered around 1980 [10, 18, 27] asserts that such schemes should be considered secure if (and only if) any information about a plaintext that is contained in a corresponding ciphertext cannot be "efficiently" obtained by any third party. In other words, it is permissible for a ciphertext to contain information about the corresponding plaintext, as long as such information cannot be efficiently obtained by a third party. Under this new paradigm, encryption schemes that make it possible to securely and efficiently transmit large amounts of data became conceivable, and indeed such schemes are routinely used today. On the other hand, because such schemes are based on computational complexity theory, our ability to reason about them, and in particular to prove their security, is constrained by our knowledge in that field, which at this point in history can be fairly described as rudimentary (as illustrated for instance by our inability to answer the fundamental question of whether or not it is more difficult to discover a solution to a problem than to merely check whether a given solution is correct).

Therefore, an important drawback of the complexity-theoretic approach is that the security of most cryptographic systems cannot currently be proved unconditionally, and must be proved under the assumption that certain computational tasks are difficult (in a suitable sense). Of course, in order to increase our confidence in the security of such systems, it is necessary to increase our confidence in the validity of the assumptions under which their security is proved. Traditionally, this was done by admitting as valid the assumption that a problem is difficult when a considerable amount of research effort had been devoted to the search of efficient solutions to it without any (or much) success. (One such problem is the *integer factoring problem* [7]: given an integer, find a non-trivial factor of it.) In recent years, however, new assumptions are introduced very frequently, and, as pointed out for instance by Naor [23], it is sometimes not clear whether proving the security of a system under a new assumption is much different from simply assuming that the system is secure.

This proliferation of new assumptions raises questions both for cryptographers, who design new cryptographic systems, and for cryptanalysts, who attempt to "break" those systems by showing that the underlying assumptions are in fact false. For the former, what are the best assumptions on which to base their constructions? And for the latter, what are the best assumptions on which to focus their efforts? A solution to these dilemmas was proposed by Ghadafi and Groth in 2017 [15] for a class of assumptions which they call "target assumptions" and which includes for instance the well-known computational Diffie-Hellman (CDH) assumption [10]. Their idea was to firstly identify a large class of assumptions (the "target assumptions") which captures many assumptions already used in the literature as well as some which may appear in the future. In a target assumption, an adversary is given some elements of a cyclic group, and must compute another, prescribed group element, the assumption being that *no* adversary can efficiently perform this task. In the CDH assumption, for instance, the adversary is given a triple of the form (g, g^a, g^b), where g is a generator of a cyclic group G of prime order q and a, b are random integers in $\{0, \ldots, q-1\}$, and must compute the element g^{ab} of G. Secondly, they identify a small subclass of assumptions (called "Uber-assumptions") within the large class, and show that if all the Uber-assumptions hold, then all the target assumptions hold as well. Namely, they identify as Uber-assumptions the family consisting of the q-Generalised Diffie-Hellman Exponent (q-GDHE) family of assumptions [4, 5] and a new family of assumptions which they name the q-Simple Fractional (q-SFrac) family. Ghadafi and Groth additionally study the internal structure of both families of assumptions, and in particular they show that the assumptions in the q-GDHE family appear to strictly increase in strength as the parameter q grows (*i.e.*, for any q it is true that $(q+1)$-GDHE implies q-GDHE, but the converse is not true *for generic group adversaries* [6, 24, 29]).

Such a result is useful both to cryptographers and to cryptanalysts. Cryptographers can use any target assumption as the basis of their systems, and be confident that they will remain secure at least as long as none of the Uber-assumptions is broken (since if their chosen assumption is false, then at least one Uber-assumption is false as well). Cryptanalysts, meanwhile, have a higher chance of success if they focus on the Uber-assumptions, since they give a small set of assumptions that is guaranteed to contain at least one false assumption (unless all the assumptions in the large class are true, in which case there is no

hope of proving that any assumption is false anyway). Of course, the usefulness of such a result can be increased either by increasing the size of the large class (since then cryptographers have more assumptions at their disposal) or by decreasing the size of the class of Uber-assumptions (since then cryptanalysts can focus their efforts on a smaller number of assumptions). It is for this reason that Ghadafi and Groth apply their analysis to a large class of assumptions (the "target assumptions"), which is a generalisation of existing assumptions.

In this thesis, we attempt to apply a similar analysis to another type of assumptions, called "knowledge-of-exponent assumptions" (KEAs). In a KEA, as in a target assumption, an adversary is given some elements of a cyclic group of prime order. However, the adversary is not tasked with computing a prescribed group element. Rather, he must compute a pair of group elements with a prescribed *relationship* between them. Moreover, the assumption is not that performing this task is infeasible, but that it is feasible *only in a prescribed manner*. For instance, the first KEA (introduced by Damgård in 1991 [8] and which we call KEA1 following Bellare and Palacio [3]) asserts that an adversary which is given a pair (g, g^a) (where again g is a generator of a cyclic group G of prime order q and a is a random integer in $\{0, \ldots, q-1\}$) has only one feasible way of computing a pair (u, v) of elements of G where $v = u^a$. Namely, the adversary must choose an integer k and compute $u = g^k$ and $v = (g^a)^k$. This is formalised by saying that for every efficient adversary that outputs such a pair (u, v), there must exist an *extractor* which outputs an integer k such that $u = g^k$. Intuitively, because the adversary must, at some point during its execution, compute such an integer k, it must be possible to construct an extractor that computes k in the same manner and outputs it instead of (u, v).

KEAs were initially criticised for not being *falsifiable*, in a sense first made precise in the context of cryptography by Naor in 2003 [23]. In Naor's sense, an assumption is falsifiable if there is "a (constructive) way to demonstrate that it is false, *if this is the case*." Concretely, he considers assumptions in which a *challenger* publishes a *challenge*, and a *verifier* verifies whether a given *solution* to the challenge is correct (possibly with the help of some additional secret information that is generated together with the challenge but not published). The assumption, then, is that it is infeasible to produce a valid solution when given only the challenge. The CDH assumption, for instance, is of this form: the challenger generates the generator g and the integers a and b, computes and pub-

lishes (g, g^a, g^b), and keeps a and b as secret information. The verifier, for its part, uses g and the secret integers a and b to compute g^{ab} and accepts a purported solution h if and only if $h = g^{ab}$. KEA1, on the other hand, is not of this type, for in order to verify that a solution is valid, the verifier would need to verify that it was produced in a manner other than the prescribed one (formally, by an adversary for which there is no extractor). We note, however, that a KEA was falsified, in a different sense, by Bellare and Palacio [3].

Despite the questions surrounding their non-falsifiability, KEAs have still been used to construct systems for which no construction was known under falsifiable assumptions. Such constructions include efficient encryption schemes secure against chosen ciphertext attacks [8], 3-round zero-knowledge protocols [3, 21, 22], and succinct non-interactive zero-knowledge protocols [19, 13], a construct for which non-falsifiable assumptions have been shown to be inherent [14]. Moreover, at least one such construction (a variant of the construction of [13, 25]) is already being used in a practical system, namely the Zcash digital currency [30]. Roughly speaking, unlike prior digital currencies such as Bitcoin which are only *pseudonymous* (meaning that each transaction publicly reveals the addresses of the participants), Zcash uses the zero-knowledge property of the construction of [13, 25] in order to be completely *anonymous*, meaning that no information is revealed. However, for efficiency reasons, classical zero-knowledge protocols such as those based on one-way functions [17] are not suitable, and succinct, non-interactive protools must be used. Since such protocols require KEAs or other non-falsifiable assumptions [14], it can be expected that KEAs will become increasingly popular in the future, which makes it all the more important to have a solid understanding of them.

1.2 Summary of Results

As mentioned above, Ghadafi and Groth firstly introduced a new class of assumptions (the "target assumptions") which generalises many pre-existing assumptions. Secondly, they identify Uber-assumptions for this class, which include a pre-existing family of assumptions (the q-GDHE family). Thirdly, they investigate the internal structure of the class of Uber-assumptions, and they show among other results that the q-GDHE family appears to be strictly increasing.

In this thesis, we perform a similar analysis for KEAs, albeit in a slightly

different order. Namely, we firstly investigate the internal structure of the q-PKE family of assumptions of Groth [19], showing that this family is increasing in groups where a certain decisional assumption holds. (Roughly speaking, the assumption is that given $(g, g^a, \ldots, g^{a^q})$ it is infeasible to distinguish $g^{a^{q+1}}$ from a random group element.)

Theorem 3.13 (Chapter 3)
Let $\mathcal{G}$ be a group generator, and $q \in \mathbf{N}^*$. If q-DDHE and $(q+1)$-PKE hold for $\mathcal{G}$, then q-PKE holds for $\mathcal{G}$.

We are unfortunately unable to prove that it is increasing in all groups, or that it is *strictly* increasing, and leave those questions open for future work. Nevertheless, we are able to prove a partial result that holds in all groups, as a generalisation of a prior result (Proposition 2) from [3]. Namely, that result showed that 1-PKE implies 0-PKE; we generalise it to show that q-PKE implies 0-PKE for all q.

Theorem 3.11 (Chapter 3)
Let $\mathcal{G}$ be a group generator, and $q \in \mathbf{N}$. If q-PKE holds for $\mathcal{G}$, then 0-PKE holds for $\mathcal{G}$.

Our reasons for focusing on the q-PKE family first are twofold. Firstly, since the q-PKE family is to our knowledge the most general instance of KEAs in the literature, its structure may be of independent interest. Secondly, studying it provides useful background for our more general study of KEAs and can be helpful in understanding both the general notion of KEAs, which is very different from that of more common assumptions such as CDH, and the proof technique we use. That proof technique was, as previously mentioned, introduced by Bellare and Palacio in [3] but acknowledged by them as being due to Halevi. It is used in that work to prove the result (Proposition 2) mentioned above, which to our knowledge is the only result in the literature showing an implication between two KEAs. Thus that technique is the only one currently known for proving implications between KEAs, and it is the one we use.

Turning to our main goal of a classification of KEAs, we firstly define a large class of assumptions, which we call "rational knowledge-of-exponent assumptions" (RKEAs) and which generalises the q-PKE family. As mentioned previously, our goal is to capture not only the KEAs that have appeared in the literature thus far, but also some that may appear in the future. Roughly speaking, whereas in a q-PKE assumption the adversary is given group elements of the form g^{x^i} for a

random integer x, in a RKEA it is given elements of the form $g^{a_i(\boldsymbol{x})/b_i(\boldsymbol{x})}$, where the a_i and b_i are arbitrary polynomials in m variables and $\boldsymbol{x}$ is a random vector of m integers. Note that q-PKE can be seen as a RKEA where the polynomials are in one variable, $a_i(x) = x^i$, and $b_i(x) = 1$. Thus RKEAs are indeed a generalisation of the q-PKE family.

Finally, as a first step towards a full classification of RKEAs we define a subclass of RKEAs which we call "simple RKEAs", analogously to the simple target assumptions of Ghadafi and Groth, and as they show that simple target assumptions imply all target assumptions, we show that simple RKEAs imply all RKEAs.

Theorem 4.6 (Chapter 4)
For any (d, m, n)*-RKEA* $\mathsf{A} = (\mathcal{I}_\mathsf{A}, \mathcal{V}_\mathsf{A}, \overline{\mathcal{V}}_\mathsf{A})$ *there is a* (nd, m, n)*-simple RKEA* $\mathsf{B} = (\mathcal{I}_\mathsf{B}, \mathcal{V}_\mathsf{B}, \overline{\mathcal{V}}_\mathsf{B})$ *such that* B *implies* A.

Unfortunately, we are unable to follow their next step and define a class of univariate simple RKEAs which would imply all simple RKEAs, and we leave this question open for future work.

1.3 Organisation

The rest of this thesis is organised as follows. In Chapter 2, we review some basic definitions and notation, as well as some background in probability and complexity theory. In Chapter 3, we discuss the q-power knowledge-of-exponent (q-PKE) family of assumptions, starting from the first KEA introduced by Damgård [8], which as we will see can be viewed as a member of this family, up to its formal introduction by Groth [19]. We then study its internal structure and show that it is increasing in some cases. In Chapter 4, we propose a generalisation of the q-PKE family which we call "rational knowledge-of-exponent assumptions" (RKEAs) and, as a first step towards identifying Uber-assumptions for RKEAs, we show that all RKEAs are implied by a slightly smaller class of assumptions (the "simple RKEAs"). Finally, in Chapter 5 we summarise our results and point out possible directions for future work.

Chapter 2

Preliminaries

2.1 Sets, integers, and strings

> *We shall not define* set*, but shall just hope that when such expressions as "the set of all real numbers" or "the set of all members of the United States Senate" are used, people's various ideas of what is meant are sufficiently similar to make communication feasible.*
>
> John B. Fraleigh, *A First Course in Abstract Algebra* [12]

Operations on sets. Given a sequence $A_1, A_2, \ldots$ of subsets of some "universal set" Ω, their *union*, noted $\bigcup_{i=1}^{\infty} A_i$ is the set of elements of Ω that are in at least one of the A_i, and their *intersection*, noted $\bigcap_{i=1}^{\infty} A_i$ is the set of elements of Ω that are in all of them. Given a subset A of Ω, its *complement*, noted A^c or $\overline{A}$ or $\Omega \setminus A$, is the set of all elements of Ω that are *not* in A. We have the *de Morgan laws*.

Proposition 2.1 (The de Morgan laws)
For any sequence $A_1, A_2, \ldots$ of subsets of Ω, we have

$$\overline{\bigcup_{i=1}^{\infty} A_i} = \bigcap_{i=1}^{\infty} \overline{A_i} \quad \textit{and} \quad \overline{\bigcap_{i=1}^{\infty} A_i} = \bigcup_{i=1}^{\infty} \overline{A_i}.$$

Integers and strings. $\mathbf{N} = \{0, 1, 2, \ldots\}$ is the set of *natural numbers* (or *natural integers*); $\mathbf{N}^* = \mathbf{N} \setminus \{0\}$ is the set of *positive integers*. For any $n \in \mathbf{N}$, $\{0,1\}^n$ is the set of binary strings of length n, and $\{0,1\}^* = \bigcup_{i \in \mathbf{N}} \{0,1\}^i$ is the

set of all finite binary strings. For any $n \in \mathbf{N}^*$, $|n| = \lfloor \log_2(n) \rfloor + 1$ is the length of the usual binary representation of n, and for convenience we set $|0| = 0$. (We also use $|x|$ to denote the absolute value of a real number x, but the meaning of such notation should always be clear from the context.) For any $n \in \mathbf{N}$, 1^n is the string of length n with all bits 1. $\mathbf{F}_p$ denotes the finite field with p elements, represented as the integers $\{0, \ldots, p-1\}$ with addition and multiplication modulo p (we only consider prime finite fields). $\mathbf{R}$ denotes the field of real numbers. ε denotes the empty string.

Asymptotics. Let $f : \mathbf{N} \to \mathbf{R}$ be a function. We say that f is *positive* if $f(n) > 0$ for all n. We say that f is *polynomial* (in n), and we write $f(n) \leq \text{poly}(n)$ if there is a polynomial p such that $f(n) \leq p(n)$ for all n. We say that f is *negligible* if for all positive polynomials p and all sufficiently large n we have $f(n) < 1/p(n)$. When such is the case we write $f(n) \leq \text{negl}(n)$, and if n is the security parameter κ, we omit it and write $\nu \leq \text{negl}$. Given another function $g : \mathbf{N} \to \mathbf{R}$, we note $f = \Theta(g)$ if there are two positive real numbers k_1, k_2 such that for all sufficiently large n we have $k_1 \cdot g(n) \leq f(n) \leq k_2 \cdot g(n)$.

2.2 Probability

We quickly review the basic objects of probability theory [20].

2.2.1 Probability spaces

Definition 2.2 (σ-algebras)
Let $\mathcal{A}$ be a collection of subsets of Ω. $\mathcal{A}$ is a σ-algebra *(on Ω) if the following conditions hold.*

1. $\Omega \in \mathcal{A}$.
2. *For any $A \in \mathcal{A}$, $A^c \in \mathcal{A}$.*
3. *For any sequence $A_1, A_2, \ldots$ of elements of $\mathcal{A}$, $\bigcup_{i=1}^{\infty} A_i \in \mathcal{A}$.*

Definition 2.3 (Measurable spaces)
A pair (Ω, Σ) where Ω is a set and Σ is a σ-algebra on Ω is called a measurable space. *The elements of Σ are called the* measurable subsets *of Ω (in the measurable space (Ω, Σ)).*

Definition 2.4 (Probability measures)
Given a measurable space (Ω, Σ)*,* P *is a* probability measure *on* (Ω, Σ) *if the following conditions hold.*

1. *For any* $A \in \Sigma$*,* $P(A)$ *is a non-negative real number, called the* probability of A.
2. $P(\Omega) = 1$.
3. *If* $A_1, A_2, \ldots$ *are* pairwise disjoint *elements of* Σ *(*i.e.*, if* $A_i \cap A_j = \emptyset$ *whenever* $i \neq j$*), then*
$$P\left(\bigcup_{i=1}^{\infty} A_i\right) = \sum_{i=1}^{\infty} P(A_i).$$

Definition 2.5 (Probability spaces)
A probability space *is a triple* (Ω, Σ, P) *such that the following conditions hold.*

1. Ω *is a set, called the* sample space.
2. Σ *is a* σ*-algebra on* Ω *(*i.e.*,* (Ω, Σ) *is a measurable space). The elements of* Σ *are called* events.
3. P *is a probability measure on* (Ω, Σ)*, called the* probability distribution.

For each $\omega \in \Omega$, if $\{\omega\}$ is an event (*i.e.*, if it is in Σ), it is called an *elementary event*. A probability space is called *finite* if its sample space is finite. In the following, we assume that Ω is finite and not empty, and that $\Sigma = \mathfrak{P}(\Omega)$, the set of all subsets of Ω. (Most commonly, Ω will be the set of strings of some fixed length.) It is clear that the probability distribution is then completely determined by the probabilities of all the elementary events, and that the sum of those probabilities equals 1. An important special case is when $P(\{\omega\}) = 1/|\Omega|$ for all $\omega \in \Omega$; the probability distribution P is then said to be *uniform*. In the following, we will sometimes abuse notation and write $P(\omega)$ instead of $P(\{\omega\})$ to denote the probability of the elementary event $\{\omega\}$.

Example 2.6 (Throw of a die)
The experiment of throwing a (fair) six-sided die can be formalised by a probability space with sample space $\{1, \ldots, 6\}$ where the probability of each elementary event is $1/6$ (*i.e.*, the probability distribution is uniform). The elementary event

$\{n\}$ naturally represents the case when the n side comes up. We can then consider non-elementary events, for example the event $\{2, 4, 6\}$ represents the case where the side which comes up is even, and the probability of this event is $1/2$, as intuition dictates.

2.2.2 Independence and conditional probabilities

The notion of *independence* formalises the perception that past events do not influence the outcome of future ones or provide any information about them. Put another way, if two events are independent, the order in which they occur is of no consequence, and it may sometimes help to think of one occurring after the other, even if it actually occurs before.

Definition 2.7 (Independent events)
Let $A_1, \ldots, A_n$ be events on (Ω, Σ, P). Those events are said to be independent *if for all subsets I of $\{1, \ldots, n\}$ we have*

$$P\left(\bigcap_{i \in I} A_i\right) = \prod_{i \in I} P\left(A_i\right).$$

They are said to be pairwise independent *if any two of them are independent,* i.e.*, if*

$$P(A_i \cap A_j) = P(A_i) \cdot P(A_j)$$

whenever $i \neq j$.

Example 2.8
Considering again the case of a fair six-sided die, the events $\mathsf{odd} = \{1, 3, 5\}$ and $\mathsf{lowerhalf} = \{1, 2, 3\}$ are not independent, because $P(\mathsf{odd}) = P(\mathsf{lowerhalf}) = 1/2$, whereas $P(\mathsf{odd} \cap \mathsf{lowerhalf}) = 1/3 \neq 1/2 \cdot 1/2$. On the other hand the events odd and $\mathsf{lowerthird} = \{1, 2\}$ are independent.

The definition of independence naturally leads to conditional probabilities. Intuitively, independence of two events A and B means that the probability that B occurs is not changed if we "know" that A has occurred. Conditional probabilities formalise "the probability that B occurs if we know that A has occurred".

Definition 2.9 (Conditional probabilities)
Let A and B be two events, with $P(A) > 0$. Then the conditional probability of

B given A *is*

$$P(B \mid A) = \frac{P(A \cap B)}{P(A)}.$$

Clearly, A and B are independent if and only if $P(B \mid A) = P(B)$.

Example 2.10
With the notation above, we have $P(\mathsf{odd} \mid \mathsf{lowerhalf}) = 2/3$. Intuitively, of the three possible "lower half" results, two are odd, so if we know that the result is in the lower half, the probability that it is odd becomes 2/3. (An equivalent experiment would be to throw an imaginary three-sided die.)

2.2.3 Random variables

Definition 2.11 (Random variables)
A random variable *on a probability space* (Ω, Σ, P) *is a function from* Ω *to* $\mathbf{R}$.

A random variable X on a probability space (Ω, Σ, P) induces a probability distribution $\mathbf{P}$ on the measurable space $(\mathbf{R}, \mathfrak{P}(\mathbf{R}))$ by

$$\mathbf{P}(A) = P(X^{-1}(A)) = P(\{\omega \in \Omega \mid X(\omega) \in A\}) = \sum_{\substack{\omega \in \Omega \\ X(\omega) \in A}} P(\omega).$$

For a subset A of $\mathbf{R}$, $\mathbf{P}(A)$ naturally represents the probability that the random variable X takes a value in A, and in particular $\mathbf{P}(\{a\})$ for some $a \in \mathbf{R}$ represents the probability that the value of X equals a.

Example 2.12 (A dice game)
Suppose we play a game where, upon the throw of a fair six-sided die, we gain three dollars if the five or six side comes up and lose one dollar otherwise. We have as before $\Omega = \{1, \ldots, 6\}$, and the random variable X representing the amount of money gained is defined as

$$X(5) = X(6) = 3$$

and

$$X(1) = X(2) = X(3) = X(4) = -1.$$

We thus have

$$\mathbf{P}(\{3\}) = P(\{5, 6\}) = \frac{1}{3},$$

and likewise $\mathbf{P}(\{-1\}) = 2/3$, which means that the probability that we gain three dollars (resp., lose one dollar) is 1/3 (resp., 2/3).

In the following, P will usually be uniform, and we will only be interested in $\mathbf{P}$. Thus we identify the two, and for a subset A of $\mathbf{R}$ we write $P(X \in A)$ instead of $\mathbf{P}(A)$. Moreover, if $A = \{a\}$ for some $a \in \mathbf{R}$, we write $P(X = a)$. Thus in the end when we write $P(X \in A)$ (resp., $P(X = a)$) we mean $P(X^{-1}(A))$ (resp., $P(X^{-1}(\{a\}))$). Similarly, we will write $P(X < a)$, $P(X \leq a)$, etc. When we write $P(A)$, without any symbol, A is an element of Σ, as in the formal definition of P. We will also abuse terminology slightly and consider random variables that are mappings from Ω to some set S other than $\mathbf{R}$. If S is finite, we then say that a random variable $X : \Omega \to S$ is *uniformly distributed* if $P(X = s) = 1/|S|$ for all $s \in S$.

2.2.4 The Schwartz-Zippel lemma

We will use the following result, often called the *Schwartz-Zippel lemma* although its origin is unclear [2, Lemma A.36].

Proposition 2.13 (The Schwartz-Zippel lemma)
Let f be a non-zero polynomial in n variables and of degree at most d over a finite field $\mathbf{F}_q$. Then if a vector $(x_1, x_2, \ldots, x_n)$ is chosen uniformly in $\mathbf{F}_q^n$ we have

$$P\left[f(x_1, x_2, \ldots, x_n) = 0\right] \leq \frac{d}{q}.$$

Proof. The proof is by induction on n. The case $n = 1$ follows from the well-known fact that a non-zero polynomial in one variable and of degree d over a field has at most d roots in that field.

Suppose now that the result holds for some $n \geq 1$, and consider a non-zero polynomial f in $n + 1$ variables and of degree $r \leq d$. We can view f as a polynomial in one variable X_{n+1} and whose coefficients are polynomials in n variables $X_1, X_2, \ldots, X_n$:

$$f(X_1, X_2, \ldots, X_n, X_{n+1}) = \sum_{k=0}^{r} f_k(X_1, X_2, \ldots, X_n) \cdot X_{n+1}^k.$$

Since f is a non-zero polynomial, there is a k such that f_k is a non-zero polynomial; we consider the largest such k, and we note that the degree of f_k equals $r - k \leq d - k$. Since we assume that the result holds for all polynomials in n variables, we obtain that, if $x_1, x_2, \ldots, x_n$ are chosen uniformly, $f_k(x_1, x_2, \ldots, x_n) = 0$ with probability at most $(d - k)/q$.

If $f_k(x_1, x_2, \ldots, x_n) \neq 0$, then the polynomial $f(x_1, x_2, \ldots, x_n, X_{n+1})$ in one variable X_{n+1} is of degree k, and so if x_{n+1} is uniformly chosen we have $f(x_1, x_2, \ldots, x_{n+1}) = 0$ with probability at most k/q.

On the sample space $\mathbf{F}_q^{n+1}$ with uniform probability distribution, we consider two events. A is the set of vectors $(x_1, x_2, \ldots, x_{n+1})$ such that $f(x_1, x_2, \ldots, x_{n+1}) = 0$; we want to show that $P(A) \leq d/q$. B is the set of vectors such that $f_k(x_1, x_2, \ldots, x_n) = 0$. We have seen above that $P(B) \leq (d-k)/q$ and that $P(A|\overline{B}) \leq k/q$. We thus obtain

$$\begin{aligned} P(A) &= P(A \cap B) + P(A \cap \overline{B}) \\ &= P(A \cap B) + P(\overline{B}) \cdot P(A \mid \overline{B}) \\ &\leq P(B) + P(A \mid \overline{B}) \\ &\leq \frac{d-k}{q} + \frac{k}{q} \\ &\leq \frac{d}{q}. \end{aligned}$$

□

2.3 Computational complexity and algorithms

As mentioned in the Introduction, our approach is based on computational complexity theory: we wish to argue that some computational tasks are impossible to perform efficiently. Of course, this means that we first need a notion of what it means for a computation to be efficient. At first glance, this seems to depend heavily on the computing hardware: we intuitively know that what is "efficient" on a modern computer might not be so on a more ancient one. Nevertheless, we can devise a computation model, the *Turing machine*, which can simulate any physically realisable computing machinery sufficiently accurately to enable us to have a sensible notion of "efficient" computation (and of computational complexity in general), independently of technology[*1)].

2.3.1 Turing machines

Although we rarely need to delve into the details of Turing machines, it is still useful to have at our disposal a precise model to which we can refer when needed,

[*1)] A possible exception is quantum computers, which are suspected (but not proven) to be impossible to simulate with Turing machines. Quantum computers are also not proven to be physically realisable.

and so we will briefly introduce them here (for details, see any text on complexity theory, such as [2]). Jumping ahead, we note that the actual computational model we will use is the one introduced by Abe and Fehr [1], which we will describe at the end of this chapter as a variant of those described beforehand.

Definition

A Turing machine is composed of one or more *tapes* (analogous to the memory of modern computers) of infinite length. Each tape is divided into *cells*, and each cell contains a *symbol*. Each tape also comes with a *head* (which is analogous to the processor). At each step of the computation, the machine starts by reading the symbols which lie under the head of each tape. Then, depending on the symbols read and on its program, it performs the following actions.

1. It writes a new symbol under the head of each tape (the new symbol may be the same as the old one, so the machine is allowed to effectively not write anything).

2. It separately moves each head one cell to the left or to the right on the corresponding tape (again, it is also allowed to leave one or more heads in their current positions).

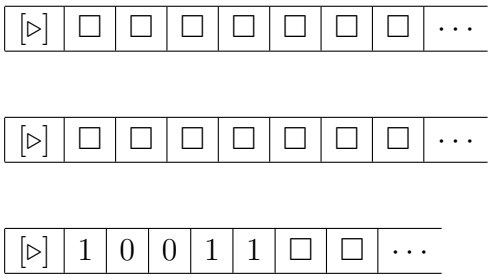
3. It proceeds to the next step of the computation.

Concretely, we define our flavour of Turing machines (following [2]) as follows.

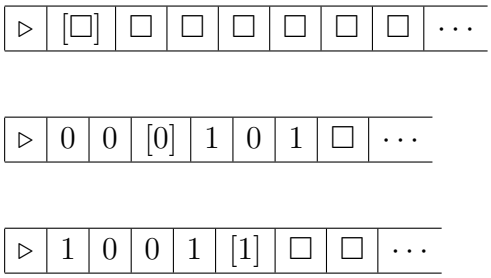
- Our machines have $k > 1$ tapes. One of them is the *input tape*, which contains the input of the computation and which is "read-only". All the other tapes are "read-write", and one of them is the *output tape*, where the output of the computation is written (the other tapes are generally called the *work tapes*).
- The set of symbols, or *alphabet*, is noted Γ. It contains the symbols 0 and 1, as well as two special symbols: the *blank symbol* $\square$ and the *start symbol* $\triangleright$.
- The set of *states* of the machine is noted Q. It contains at least the *initial state* q_{start} and the *final state* (or *halting state*) q_{halt}.
- The *transition function* of the machine, which is the "program" indicating which actions the machine performs as a function of its read symbols and current state, is noted δ. Formally, it is a function from $\Gamma^k \times Q$ (read

symbols and current state) to $\Gamma^{k-1} \times \{\blacktriangleleft, \blacktriangledown, \blacktriangleright\}^k \times Q$ (written symbols, head movements, and new state).

At the start of the computation, the machine is in the initial state q_{start}. The tapes contain the start symbol $\triangleright$ followed by infinitely many blank symbols $\square$, except the input tape, which contains the start symbol, then the input of the computation, then the blank symbols. The heads are all at the beginning of their respective tapes. For example, with three tapes (the brackets indicate the positions of the heads):



The computation then begins; it simply consists of repeated application of the transition function. During the computation, the tapes may look like this:



The computation ends when the machine enters the terminal state q_{halt} (we then say that the machine *halts*). Then it performs no further action, and the output of the computation is the contents of the output tape. The output of a machine M on input x, assuming that M halts on input x, is noted $M(x)$. Given a function $T : \mathbf{N} \to \mathbf{N}$, we say that a machine *runs in time* $T(n)$ if for all input strings $x \in \{0,1\}^*$ it halts after at most $T(|x|)$ steps. We say that the machine runs in *polynomial time* if there is a polynomial $p(n)$ such that the machine runs in time $p(n)$. A machine that runs in polynomial time is also called a *polynomial-time machine*.

Example: Palindromes

We will define a Turing machine which decides whether its input string is a *palindrome*, *i.e.*, whether it is read in the same way from left to right and from right to left (for example 0010100 and 10001 are palindromes, but 100110 is not), again following [2]. As above, our machine has three tapes (input, work and output), and its alphabet is $\Gamma = \{0, 1, \square, \triangleright\}$. To determine its states and transition function, it is helpful to start with a high-level description of the computation.

1. Copy the entire input on the work tape.
2. Go back to the start of the input tape (staying at the end of the work tape).
3. [Finished?] If we are at the end of the input tape *and* at the start of the work tape, output 1 and halt.
4. [Test] If the symbols under the input and work tape are different, output 0 and halt. Otherwise, move one position to the right on the input tape, one position to the left on the work tape, and go to step 3.

In detail, we have five states $Q = \{q_{\mathsf{start}}, q_{\mathsf{copy}}, q_{\mathsf{left}}, q_{\mathsf{test}}, q_{\mathsf{halt}}\}$, and the transition function is defined as follows.

- In state q_{start}, the machine moves all the heads one position to the right without writing anything (more accurately, rewriting the $\triangleright$ symbol) and enters state q_{copy}.
- In state q_{copy}, if the symbol on the input tape is *not* the $\square$ symbol, the machine copies it on the work tape, moves both heads to the right, and remains in state q_{copy}. Otherwise, it moves them to the left, writes nothing, and enters state q_{left}. (In both cases, the head of the output tape does not move and writes nothing.)
- In state q_{left}, if the symbol on the input tape is not $\triangleright$, the machine moves its head to the left, writes nothing, and remains in state q_{left}. Otherwise, it moves it to the right, still writing nothing, and enters state q_{test}.
- In state q_{test}, if the symbol on the input tape is $\square$ *and* the symbol on the work tape is $\triangleright$, the machine writes 1 on the output tape and enters q_{halt}. Otherwise, if the symbols on the input and work tapes are different, it writes 0 on the output tape and enters q_{halt}. Otherwise, it moves the

heads of the input and work tapes to the right and the left respectively, and remains in state q_{test}.

It is easily seen that this machine correctly decides whether its input is a palindrome. As for its running time, we see that on input x it always spends 1 step in state q_{start}, then $|x|+1$ steps in state q_{copy}, then again $|x|+1$ steps in state q_{left}, then finally *at most* $|x|+1$ steps in state q_{test} (if the string is indeed a palindrome). It will therefore always halt after at most $3|x|+4$ steps, and so it runs in polynomial time.

2.3.2 Probabilistic Turing machines

One aspect of "real programs" that does not seem to be captured by the foregoing definition of Turing machines is the ability to make *random choices*. Since our approach is to not make any assumption about the strategy employed by an adversary, and since many practical programming languages provide a random number generator, it seems reasonable to model adversaries with machines that can access a source of randomness. Such machines were first introduced by Rabin in 1976 [26]; we discuss them following [2, Chap. 7].

Definition 2.14 (Probabilistic polynomial-time (PPT) Turing machines)
A probabilistic Turing machine *is a Turing machine with an additional, read-only tape called the* random tape. *A probabilistic Turing machine is said to run in polynomial time if there is a polynomial p such that for any strings $x \in \{0,1\}^*$ and $r \in \{0,1\}^{p(|x|)}$, the machine halts after at most $p(|x|)$ steps on input x when its random tape initially contains the string r. A probabilistic Turing machine that runs in polynomial time is called a* probabilistic polynomial-time *(PPT) Turing machine. We call the string r the* random coins *of the machine, and we denote by $M(x;r)$ the output of a PPT machine M on input x and random coins r.*

Remark 2.15
When we need to make explicit the fact that a polynomial-time Turing machine is *not* probabilistic, we will call it a *deterministic polynomial-time* (DPT) machine.

The execution of a PPT Turing machine M on inputs of length n naturally induces a probability space with sample space $\{0,1\}^{p(n)}$ and uniform probability distribution. For any $x \in \{0,1\}^n$, we can then define a random variable by mapping any $r \in \{0,1\}^{p(n)}$ to the string $M(x;r)$. This random variable naturally

represents the output of M on input x, and we note it $M(x)$; thus for example the probability that M outputs 1 on input x is

$$\begin{aligned} P(M(x) = 1) &= P(\{r \in \{0,1\}^{p(n)} \mid M(x;r) = 1\}) \\ &= \frac{1}{2^{p(n)}} \cdot \left|\{r \in \{0,1\}^{p(n)} \mid M(x;r) = 1\}\right|. \end{aligned}$$

2.3.3 Non-uniform Turing machines

Definition 2.16 (Non-uniform polynomial-time Turing machines)
A non-uniform Turing machine *is a Turing machine with an additional, read-only tape called the* advice tape*, together with an infinite sequence of* advice strings $\mathrm{adv}_0, \mathrm{adv}_1, \ldots$. *A non-uniform Turing machine is said to run in polynomial time if there is a polynomial p such that for any string $x \in \{0,1\}^*$, the machine halts after at most $p(|x|)$ steps on input x when its advice tape initially contains the string* $\mathrm{adv}_{|x|}$. *A non-uniform Turing machine that runs in polynomial time is called a* non-uniform polynomial-time *Turing machine, and we denote by $M(x)$ the output of such a machine M on input x.*

We can combine probabilistic and non-uniform Turing machines to yield *non-uniform probabilistic Turing machines*, which have both a random tape and an advice tape. We say that a machine runs in *non-uniform probabilistic polynomial time* if there is a polynomial p such that for all strings $x \in \{0,1\}^*$ and $r \in \{0,1\}^{p(|x|)}$ it halts after at most $p(|x|)$ steps on input x when the strings r and $\mathrm{adv}_{|x|}$ are initially written respectively on its random and advice tapes. Such a machine is called a *non-uniform probabilistic polynomial-time* (non-uniform PPT) machine.

2.3.4 The model we use

As mentioned at the beginning of this section, we will use a variant of Turing machines first introduced by Abe and Fehr [1]. All our machines will have an additional, read-only tape that we will call the *security parameter tape* and that will initially contain the string 1^κ, for a security parameter κ. All machines will run in time polynomial in κ, meaning that whenever we consider a machine it is assumed that there is a polynomial p such that the machine always halts after at most $p(\kappa)$ steps (where κ is the length of the string on its security parameter

tape), regardless of the initial contents of all its other tapes (in particular, of its input tape, which we may thus assume to have length at most $p(\kappa)$).

Such machines can be probabilistic, meaning that they have a random tape which initially contain the random coins (of length $p(\kappa)$) and that their output (on a given input and security parameter) can be viewed as a random variable over the sample space of all possible random coins with uniform distribution. They can also be non-uniform, but with the advice string depending on the security parameter instead of on the length of the input (*i.e.*, the string written on the advice tape is adv_κ, which may also be assumed to have length at most $p(\kappa)$).

Thus the output of such a machine $\mathcal{A}$ on input x and security parameter κ is $\mathcal{A}(1^\kappa, x)$ or $\mathcal{A}(1^\kappa, x, \mathrm{adv}_\kappa)$. To ease notation, however, 1^κ and adv_κ will often be omitted (*i.e.*, we will often write $\mathcal{A}(x)$ instead of $\mathcal{A}(1^\kappa, x)$ or $\mathcal{A}(1^\kappa, x, \mathrm{adv}_\kappa)$). For two machines $\mathcal{A}$ and $\mathcal{B}$ we denote by $\mathcal{A}||\mathcal{B}$ their joint execution on a common input and random tape, and we write $(u; v) \leftarrow (\mathcal{A}||\mathcal{B})(x)$ to say that the output of $\mathcal{A}$ on input x is assigned to u and the output of $\mathcal{B}$ on the same input x *and the same random coins* is assigned to v. We note that it makes sense to say that $\mathcal{A}$ and $\mathcal{B}$ are executed on the same random coins: if $\mathcal{A}$ (resp., $\mathcal{B}$) runs in time $p_{\mathcal{A}}(\kappa)$ (resp., $p_{\mathcal{B}}(\kappa)$), then both machines can be seen as running in time $(p_{\mathcal{A}} + p_{\mathcal{B}})(\kappa)$, with random coins of equal length.

Chapter 3

The q-PKE family of assumptions

In this chapter we discuss the KEAs that have appeared in the literature thus far, from the first KEA of Damgård [8] to the q-PKE family of Groth [19]. (The title of this chapter is justified by the fact that all those KEAs can be seen as members of the q-PKE family, or close variants thereof.) We then show two theorems regarding the internal structure of the q-PKE family, which, as mentioned in the Introduction, indicate that it is increasing.

3.1 Group generators

As is common in modern practice, we will define assumptions relative to an abstract *group generator*, as opposed to defining them in specific groups. We define group generators following Ghadafi and Groth [15].

Definition 3.1 (Group generators)
A group generator *is a uniform probabilistic algorithm $\mathcal{G}$ which on security parameter κ outputs group parameters (G_p, g), where*

- *p is a prime with $|p| = \Theta(\kappa)$;*
- *G_p is (a description of) a (cyclic) group of order p, with canonical representations of group elements as bitstrings and efficient algorithms for performing the group operation and deciding membership; and*
- *g is a uniformly random generator of G_p.*

Example 3.2
A simple example of a group generator that is used often in practice is one that produces a prime-order subgroup of the multiplicative group of a prime finite

field. Namely, on input 1^κ it generates a prime q of length $\kappa + 1$ such that $p = (q-1)/2$ (which is of length κ) is also prime. The group G_p is then the order-p subgroup of $\mathbf{F}_q^*$, which is the group of *quadratic residues* modulo q; its elements are represented as bitsrings by their usual binary representation. The integer q suffices as a description of this group: the group operation is just multiplication modulo q, and testing whether an element $a \in \mathbf{F}_q^*$ is in G_p can be done by testing whether $a^{(q-1)/2} \bmod q$ equals 1. Finally, a generator g can be chosen uniformly by choosing uniformly an integer $i \in \{1, \ldots, q-1\}$ and outputting $g = 4^i \bmod q$ ($4 = 2^2$ is a generator of G_p since it is a quadratic residue, it is not 1, and the group has prime order).

As in [15], given a group G_p, a generator g, and an element $x \in \mathbf{F}_p$, we will denote by $[x]$ the element of G_p with discrete logarithm x relative to the generator g and the group operation of G_p, *i.e.*, $[x] = g \circ g \circ \cdots \circ g$ for x terms. Thus the generator g is $[1]$ and the identity element is $[0]$. We will also denote the group operation *additively*, so that we have $[x+y] = [x] + [y]$ and $[kx] = k[x]$ (where $k[x] = [x] + [x] + \cdots + [x]$ for k terms).

3.2 The first knowledge-of-exponent assumption (KEA1)

As mentioned, the first knowledge-of-exponent assumption, which we call KEA1 following Bellare and Palacio [3], was introduced by Damgård in 1991 [8]. Roughly, it says that given a pair $([1], [\alpha])$ of elements of G_p, the only way to generate a pair $([k], [k\alpha])$ is the obvious way: pick k in some fashion, and compute $[k] = k[1]$ and $[k\alpha] = k[\alpha]$. In other words, any algorithm (adversary) which outputs such a pair must "know" k. This is formalised by saying that there must exist another algorithm, called an extractor, which, also given $([1], [\alpha])$, outputs k.

Assumption 3.3 (KEA1)
Let $\mathcal{G}$ be a group generator. We say that KEA1 holds (relative to $\mathcal{G}$) if for every non-uniform probabilistic algorithm $\mathcal{A}$ (the adversary*) there is a non-uniform probabilistic algorithm $\chi_\mathcal{A}$ (the* extractor*) such that*

$$\begin{aligned}\Pr\big[&(G_p, [1]) \leftarrow \mathcal{G}; \alpha \leftarrow \mathbf{F}_p; \sigma := (G_p, [1], [\alpha]);\\ &\big(([u], [v]); k\big) \leftarrow (\mathcal{A}||\chi_\mathcal{A})(\sigma) :\\ &\big([v] = \alpha[u]\big) \wedge \big([u] \neq k[1]\big)\big] \leq \text{negl}.\end{aligned}$$

Remark 3.4
In the above probabilistic statement, the probability is taken over the random coins of $\mathcal{G}$ and $\mathcal{A}||\chi_{\mathcal{A}}$ and the uniform choice of $\alpha \in \mathbf{F}_p$. In other words, the sample space is $\{0,1\}^{p_{\mathcal{G}}(\kappa)} \times \{0,1\}^{p_{\mathcal{A}||\chi_{\mathcal{A}}}(\kappa)} \times \mathbf{F}_p$ with uniform probability distribution. Other probabilistic statements throughout should be read similarly.

Remark 3.5
In [13, 25], KEAs are augmented to take into account any prior information the adversary might possess. Namely, the adversary has an additional auxiliary input z, and the condition must hold for all z generated independently of α. (Of course, the extractor is given z as well.)

In [8], KEA1 is used to show that a variant of the ElGamal encryption scheme [11] is secure against chosen ciphertext attacks by asserting that the only way for an adversary to produce a valid ciphertext (*i.e.*, one that is accepted by the decryption oracle) is to encrypt a known plaintext (in which case the decryption gives no advantage).

KEA1 is a very strong assumption. For instance, it is easily shown that, under KEA1, the hardness of the discrete logarithm problem implies that of the computational Diffie-Hellman problem, whereas the same implication in the general case is a longstanding open question. Nevertheless, KEA1 has been shown to hold in the generic group model [6, 24, 29] independently by Abe and Fehr [1] and by Dent [9].

3.3 The discrete logarithm assumption (DLA)

As in [3], we remark that if the discrete logarithm problem is easy (in groups generated by $\mathcal{G}$), then KEA1 trivially holds (in $\mathcal{G}$), for in that case we can trivially construct a KEA1-extractor $\chi_{\mathcal{A}}$ for any $\mathcal{A}$ as follows. Since $\chi_{\mathcal{A}}$ is given $\mathcal{A}$'s input and random coins, it can compute $\mathcal{A}$'s output $([u], [v])$, and furthermore, since the discrete logarithm problem is easy, it can compute u from $[u]$. It then outputs u, and if the discrete logarithm computation was successful (which happens with high probability since the discrete logarithm problem is easy), it will be successful as well.

Therefore, KEAs are only interesting in groups where the discrete logarithm problem is (believed to be) hard, which are the groups commonly used in cryptographic systems. We will thus assume throughout that the discrete logarithm

problem is hard in the group generators we will consider, and we formalise this assumption as follows.

Assumption 3.6 (The discrete logarithm assumption (DLA))
We say that DLA holds (relative to the group generator $\mathcal{G}$) if for every non-uniform probabilistic adversary $\mathcal{A}$ we have

$$\Pr\big[(G_p, [1]) \leftarrow \mathcal{G}; \alpha \leftarrow \mathbf{F}_p : \mathcal{A}(G_p, [1], [\alpha]) = \alpha\big] \leq \text{negl}.$$

3.4 More assumptions: KEA2 and KEA3

A second knowledge-of-exponent assumption (KEA2) was introduced by Hada and Tanaka in 1998 [21, 22] to construct 3-round zero-knowlege protocols, which had been a longstaning open problem. However, KEA2 was proven false (under the DLA) by Bellare and Palacio in 2004 [3], a result which, besides rendering the results of Hada and Tanaka vacuous, showed that it was possible to falsify a KEA, albeit in a different sense than that of Naor [23]. Fortunately, Bellare and Palacio were able to recover the results of Hada and Tanaka by using another new assumption, named KEA3, and they also showed that KEA3 implies KEA1, a result which we extend below.

Roughly, KEA2 states that given $([1], [x], [\alpha], [\alpha x])$, there are only two ways to produce a pair $([k], [k\alpha])$: generate k in some fashion and output either $(k[1], k[\alpha])$ or $(k[x], k[\alpha x])$. Intuitively, it is easy to see why this assumption should be false under the DLA: what about an adversary which generates k_1, k_2 and outputs $(k_1[1] + k_2[x], k_1[\alpha] + k_2[\alpha x])$? KEA2 asserts that such an adversary should know either $k_1 + k_2 x$ or $k_1 x^{-1} + k_2$, but it seems impossible to compute them without computing x and breaking the DLA. KEA3 addresses this issue in the obvious manner, by asserting that the only way to generate a pair $([k], [k\alpha])$ is as above: generate k_1, k_2 and output $(k_1[1] + k_2[x], k_1[\alpha] + k_2[\alpha x])$. We now turn to the formalisation of both assumptions.

Assumption 3.7 (KEA2)
Let $\mathcal{G}$ be a group generator. We say that KEA2 holds (relative to $\mathcal{G}$) if for every non-uniform probabilistic adversary $\mathcal{A}$ there is a non-uniform probabilistic

extractor $\chi_{\mathcal{A}}$ such that

$$\begin{aligned}\Pr\big[&(G_p,[1]) \leftarrow \mathcal{G}; x, \alpha \leftarrow \mathbf{F}_p;\\ &\sigma \coloneqq (G_p,[1],[x],[\alpha],[\alpha x]);\\ &\big(([u],[v]);k\big) \leftarrow (\mathcal{A}||\chi_{\mathcal{A}})(\sigma) :\\ &\big([v]=\alpha[u]\big) \wedge \big([u] \neq k[1]\big) \wedge \big([u] \neq k[x]\big)\big] \leq \text{negl}.\end{aligned}$$

Assumption 3.8 (KEA3)
Let $\mathcal{G}$ be a group generator. We say that KEA3 holds (relative to $\mathcal{G}$) if for every non-uniform probabilistic adversary $\mathcal{A}$ there is a non-uniform probabilistic extractor $\chi_{\mathcal{A}}$ such that

$$\begin{aligned}\Pr\big[&(G_p,[1]) \leftarrow \mathcal{G}; x, \alpha \leftarrow \mathbf{F}_p;\\ &\sigma \coloneqq (G_p,[1],[x],[\alpha],[\alpha x]);\\ &\big(([u],[v]);(k_1,k_2)\big) \leftarrow (\mathcal{A}||\chi_{\mathcal{A}})(\sigma) :\\ &\big([v]=\alpha[u]\big) \wedge \big([u] \neq k_1[1]+k_2[x]\big)\big] \leq \text{negl}.\end{aligned}$$

3.5 The q-PKE family of assumptions

In this section we describe the q-power knowledge-of-exponent (q-PKE) family of assumptions, which was introduced by Groth in 2010 [19], and we prove two theorems which indicate that this family of assumptions is increasing in strength as the parameter q grows (*i.e.*, that assumptions with a higher value of q imply those with a lower value). We note that Groth showed in [19] that q-PKE holds in the generic group model for any q.

Assumption 3.9 (The q-power knowledge-of-exponent assumption (q-PKE))
Let $\mathcal{G}$ be a group generator, and $q \in \mathbf{N}$. We say that q-PKE holds (relative to $\mathcal{G}$) if for every non-uniform probabilistic adversary $\mathcal{A}$ there is a non-uniform probabilistic extractor $\chi_{\mathcal{A}}$ such that

$$\begin{aligned}\Pr\big[&(G_p,[1]) \leftarrow \mathcal{G}; x, \alpha \leftarrow \mathbf{F}_p;\\ &\sigma \coloneqq (G_p,[1],[x],\ldots,[x^q],[\alpha],[\alpha x],\ldots,[\alpha x^q]);\\ &\big(([u],[v]);(k_0,\ldots,k_q)\big) \leftarrow (\mathcal{A}||\chi_{\mathcal{A}})(\sigma) :\\ &\big([v]=\alpha[u]\big) \wedge \big([u] \neq \textstyle\sum_{i=0}^{q} k_i[x^i]\big)\big] \leq \text{negl}.\end{aligned}$$

Remark 3.10
We can allow the parameter q to be any function of the security parameter κ; in that case, the experiment on security parameter κ has $\sigma := (G_p, [1], [x], \ldots, [x^{q(\kappa)}], [\alpha], [\alpha x], \ldots, [\alpha x^{q(\kappa)}])$. Of course, since our algorithms run in time polynomial in κ, we can assume that $q(\kappa)$ is polynomial as well. To ease notation, we will always simply write q.

We note that KEA1 is 0-PKE and that KEA3 is 1-PKE. As previously mentioned, it was shown by Bellare and Palacio [3] that 1-PKE implies 0-PKE; the proof there readily extends to show that, for any q, q-PKE implies 0-PKE. For completeness and as a warm-up for what follows, we restate it here.

Theorem 3.11 (Generalisation of Proposition 2 from [3])
Let $\mathcal{G}$ be a group generator, and $q \in \mathbf{N}$. If q-PKE holds for $\mathcal{G}$, then 0-PKE holds for $\mathcal{G}$.

Proof. Let $\mathcal{A}$ be an adversary against 0-PKE; we first construct an adversary $\mathcal{B}$ against q-PKE that uses $\mathcal{A}$ in a black-box manner. $\mathcal{B}$ has input

$$(G_p, [1], [x], \ldots, [x^q], [\alpha], [\alpha x], \ldots, [\alpha x^q]);$$

it runs $\mathcal{A}$ on input $(G_p, [1], [\alpha])$ and with its own random tape, and outputs the pair $([u], [v])$ output by $\mathcal{A}$. Since q-PKE holds, there is an extractor $\chi_{\mathcal{B}}$ for $\mathcal{B}$ with negligible error probability ν; we construct an extractor $\chi_{\mathcal{A}}$ for $\mathcal{A}$ that uses $\chi_{\mathcal{B}}$ in a black-box manner. $\chi_{\mathcal{A}}$ proceeds as follows on input $(G_p, [1], [\alpha])$.

- $x \leftarrow \mathbf{F}_p$.
- $\sigma := (G_p, [1], [x], \ldots, [x^q], [\alpha], [\alpha x], \ldots, [\alpha x^q])$.
- $(k_0, \ldots, k_q) \leftarrow \chi_{\mathcal{B}}(\sigma)$.
- Output $\sum_{i=0}^{q} k_i x^i$.

We claim that $\chi_{\mathcal{A}}$ has (negligible) error probability ν.

We run the 0-PKE experiment. Firstly, $\mathcal{A}$ is run on input $(G_p, [1], [\alpha])$, and we let $([u], [v])$ be its output. Then $\chi_{\mathcal{A}}$ is run, again on input $(G_p, [1], [\alpha])$ and with the same random tape as $\mathcal{A}$, and it runs $\chi_{\mathcal{B}}$ on input σ and with its own random tape. Now, observe that σ is distributed identically to the input to $\chi_{\mathcal{B}}$ in the experiment for q-PKE, and so, letting $\big(([u'], [v']); (k_0, \ldots, k_q)\big)$ be the output

of $\mathcal{B}||\chi_{\mathcal{B}}$ on input σ and with the same random tape as that of $\mathcal{A}$, we have

$$\big([v'] = \alpha[u']\big) \wedge \Big(\sum_{i=0}^{q} k_i[x^i] \neq [u']\Big)$$

with probability ν. Since $\mathcal{B}$ on input σ runs $\mathcal{A}$ on input $(G_p, [1], [\alpha])$ and with the same random tape as that with which $\mathcal{A}$ was run originally, we have $([u'], [v']) = ([u], [v])$. Observing that

$$\sum_{i=0}^{q} k_i[x^i] = \Big(\sum_{i=0}^{q} k_i x^i\Big)[1]$$

completes the proof. □

As mentioned, the above result shows in particular that 1-PKE implies 0-PKE. A natural question is then to ask whether this can be generalised to show that in general $(q+1)$-PKE implies q-PKE. We show that this is the case under a variant of the decisional Diffie-Hellman exponent (DDHE) assumption.[*1)]

Assumption 3.12 (The q-decisional Diffie-Hellman exponent (q-DDHE) assumption)
Let $\mathcal{G}$ be a group generator, $\mathcal{A}$ be a non-uniform probabilistic adversary, $q \in \mathbf{N}^*$, $b \in \{0,1\}$. Consider the following experiment $\mathsf{Exp}^{q\text{-}ddhe\text{-}b}_{\mathcal{G},\mathcal{A}}(\kappa)$.

- $(G_p, [1]) \leftarrow \mathcal{G}; x, r \leftarrow \mathbf{F}_p$.
- *If $b = 0$, then $\sigma \coloneqq (G_p, [1], [x], \ldots, [x^q], [r])$; otherwise, $\sigma \coloneqq (G_p, [1], [x], \ldots, [x^q], [x^{q+1}])$.*
- $b' \leftarrow \mathcal{A}(\sigma)$.
- *Output b'.*

We let

$$\mathsf{Adv}^{q\text{-}ddhe}_{\mathcal{G},\mathcal{A}}(\kappa) = \Big|\Pr\big[\mathsf{Exp}^{q\text{-}ddhe\text{-}1}_{\mathcal{G},\mathcal{A}}(\kappa) = 1\big] - \Pr\big[\mathsf{Exp}^{q\text{-}ddhe\text{-}0}_{\mathcal{G},\mathcal{A}}(\kappa) = 1\big]\Big|$$

be the advantage *of $\mathcal{A}$ (in q-DDHE) relative to $\mathcal{G}$, and we say that q-DDHE holds in $\mathcal{G}$ if every non-uniform adversary has negligible advantage,* i.e., *if for every non-uniform adversary $\mathcal{A}$, we have* $\mathsf{Adv}^{q\text{-}ddhe}_{\mathcal{G},\mathcal{A}} \leq \mathrm{negl}$.

[*1)] Unfortunately, since our proof relies on a decisional assumption, it does not apply in the bilinear setting, which is the setting in which q-PKE was introduced in [19] and subsequently used in [13, 25].

Theorem 3.13
Let $\mathcal{G}$ be a group generator, and $q \in \mathbf{N}^*$. If q-DDHE and $(q+1)$-PKE hold for $\mathcal{G}$, then q-PKE holds for $\mathcal{G}$.

Proof. Let $\mathcal{A}$ be an adversary against q-PKE; we first construct an adversary $\mathcal{B}$ against $(q+1)$-PKE that uses $\mathcal{A}$ in a black-box manner. $\mathcal{B}$ has input

$$\big(G_p, [1], [x], \ldots, [x^{q+1}], [\alpha], [\alpha x], \ldots, [\alpha x^{q+1}]\big);$$

it runs $\mathcal{A}$ on input

$$\big(G_p, [1], [x], \ldots, [x^{q}], [\alpha], [\alpha x], \ldots, [\alpha x^{q}]\big)$$

and with its own random tape, and outputs the pair $([u], [v])$ output by $\mathcal{A}$. Since $(q+1)$-PKE holds, there is an extractor $\chi_{\mathcal{B}}$ for $\mathcal{B}$ with negligible error probability ν; we construct an extractor $\chi_{\mathcal{A}}$ for $\mathcal{A}$ that uses $\chi_{\mathcal{B}}$ in a black-box manner. $\chi_{\mathcal{A}}$ proceeds as follows on input $(G_p, [1], [x], \ldots, [x^q], [\alpha], [\alpha x], \ldots, [\alpha x^q])$.

- $r \leftarrow \mathbf{F}_p$.
- $\sigma := (G_p, [1], \ldots, [x^q], [r], [\alpha], \ldots, [\alpha x^q], [\alpha r])$.
- $(k_0, k_1, \ldots, k_{q+1}) \leftarrow \chi_{\mathcal{B}}(\sigma)$.
- Output $(k_0 + k_{q+1} r, k_1, \ldots, k_q)$.

We claim that $\chi_{\mathcal{A}}$ has negligible error probability.

We run the q-PKE experiment. Firstly, $\mathcal{A}$ is run on input

$$(G_p, [1], [x], \ldots, [x^q], [\alpha], [\alpha x], \ldots, [\alpha x^q]),$$

and we let $([u], [v])$ be its output. Then $\chi_{\mathcal{A}}$ is run, on the same input and with the same random tape as $\mathcal{A}$, and it runs $\chi_{\mathcal{B}}$ on input σ and with its own random tape. We claim that, letting $\big(([u'], [v']); (k_0, \ldots, k_{q+1})\big)$ be the output of $\mathcal{B}||\chi_{\mathcal{B}}$ on input σ, we have

$$\big([v'] = \alpha[u']\big) \wedge \big(k_{q+1}[r] + \sum_{i=0}^{q} k_i [x^i] \neq [u']\big)$$

with negligible probability. Intuitively, this follows from the fact that, under q-DDHE, σ is indistinguishable from the input to $\mathcal{B}||\chi_{\mathcal{B}}$ in the $(q+1)$-PKE experiment. To show it formally, we consider the following adversary $\mathcal{Z}$ against q-DDHE, which uses $\mathcal{B}$ and $\chi_{\mathcal{B}}$ in a black-box manner.

$\mathcal{Z}$ proceeds as follows on input $(G_p, [1], [x], \ldots, [x^q], [z])$ (where x is random and z is either x^{q+1} or random).

- $\alpha \leftarrow \mathbf{F}_p$.
- $\sigma := (G_p, [1], [x], \ldots, [x^q], [z], [\alpha], [\alpha x], \ldots, [\alpha x^q], [\alpha z])$.
- $\big(([u], [v]); (k_0, k_1, \ldots, k_{q+1})\big) \leftarrow (\mathcal{B}||\chi_{\mathcal{B}})(\sigma)$.
- If $[v] = \alpha[u]$ and $k_{q+1}[z] + \sum_{i=0}^{q} k_i[x^i] \neq [u]$, output 1; else, output 0.

If $z = x^{q+1}$, the q-DDHE experiment for $\mathcal{Z}$ is exactly the $(q+1)$-PKE experiment for $\mathcal{B}||\chi_{\mathcal{B}}$, and so $\mathcal{Z}$ outputs 1 with (negligible) probability ν. On the other hand, if z is random, then σ is distributed identically to the input to $\mathcal{B}||\chi_{\mathcal{B}}$ when it is run by $\chi_{\mathcal{A}}$ in the q-PKE experiment. Let μ be the probability that $\mathcal{Z}$ outputs 1 in that latter case; then $|\nu - \mu|$ is negligible since q-DDHE holds, and since ν is negligible as well, so is μ.

Finally, since $\mathcal{B}$ on input σ runs $\mathcal{A}$ on input $(G_p, [1], \ldots, [x^q], [\alpha], \ldots, [\alpha x^q])$ and with the same random tape as that with which $\mathcal{A}$ was run originally, we have $([u'], [v']) = ([u], [v])$. Observing that

$$k_{q+1}[r] + \sum_{i=0}^{q} k_i[x^i] = (k_0 + k_{q+1}r)[1] + \sum_{i=1}^{q} k_i[x^i]$$

completes the proof. □

Chapter 4

Rational KEAs (RKEAs)

In this chapter, we propose a definition of a large class of assumptions, with the goal of capturing not only the KEAs that have appeared in the literature thus far, but also some that are likely to appear in the future. We then show that this large class is implied by a slightly smaller subclass.

4.1 Definition of RKEAs

Along the lines of the definition of target assumptions by Ghadafi and Groth [15], we generalise the PKE family of assumptions by allowing arbitrary rational functions of several variables instead of just powers of x. We call the resulting class of assumptions *rational knowledge-of-exponent assumptions* (RKEAs). Analogously to the target assumptions of [15], RKEAs are parameterised by three integers[*1)] d (the maximal degree of the polynomials involved), m (the number of variables) and n (the number of rational functions). We first define a very general notion of *non-interactive knowledge assumptions* (NIKAs) analogous to the non-interactive computational assumptions of Ghadafi and Groth [15]. (The intuitive meanings of the quoted terms should be clear from the previous examples of KEAs.)

Definition 4.1 (Non-interactive knowledge assumptions (NIKAs))
A non-interactive knowledge assumption *consists of an instance generator $\mathcal{I}$, a verifier $\mathcal{V}$, and a knowledge verifier $\overline{\mathcal{V}}$, defined as follows.*

[*1)] Again, we can allow d, m, n to be any functions of the security parameter κ, and assume that they are polynomial.

- $(\mathsf{pub}, \mathsf{priv}) \leftarrow \mathcal{I}$: *$\mathcal{I}$ is a uniform probabilistic algorithm which, on input 1^κ (where κ is a security parameter), outputs a pair of public/private information* $(\mathsf{pub}, \mathsf{priv})$. *We omit the input 1^κ as usual.*
- $0/1 \leftarrow \mathcal{V}(\mathsf{pub}, \mathsf{priv}, \mathsf{sol})$: *$\mathcal{V}$ is a uniform deterministic algorithm which, on input* $(\mathsf{pub}, \mathsf{priv})$ *and a purported solution* sol, *outputs* 1 *if the solution is "correct" and* 0 *otherwise.*
- $0/1 \leftarrow \overline{\mathcal{V}}(\mathsf{pub}, \mathsf{priv}, \mathsf{sol}, \mathsf{sec})$: *$\overline{\mathcal{V}}$ is a uniform deterministic algorithm which, on input* $(\mathsf{pub}, \mathsf{priv}, \mathsf{sol})$ *and a purported "secret"* sec, *outputs* 1 *if the secret is "correct" and* 0 *otherwise.*

We say that the assumption holds if for any non-uniform probabilistic algorithm $\mathcal{A}$ (the adversary*) there is a non-uniform probabilistic algorithm $\chi_\mathcal{A}$ (the* knowledge extractor*, or just the* extractor*) such that*

$$\Pr\big[(\mathsf{pub}, \mathsf{priv}) \leftarrow \mathcal{I}; (\mathsf{sol}; \mathsf{sec}) \leftarrow (\mathcal{A}||\chi_\mathcal{A})(\mathsf{pub}) : \mathcal{V}(\mathsf{pub}, \mathsf{priv}, \mathsf{sol}) = 1 \wedge \overline{\mathcal{V}}(\mathsf{pub}, \mathsf{priv}, \mathsf{sol}, \mathsf{sec}) = 0\big] \leq \text{negl}.$$

We call the above probability the error probability *of $\chi_\mathcal{A}$ relative to $\mathcal{A}$, and express it as a function of the security parameter κ; thus the assumption holds if for every adversary $\mathcal{A}$ there is an extractor $\chi_\mathcal{A}$ with negligible error probability relative to $\mathcal{A}$. We also say that $\chi_\mathcal{A}$ is* successful *(relative to $\mathcal{A}$) if the condition above does* not *hold,* i.e.*, if $\chi_\mathcal{A}$ "successfully extracts" $\mathcal{A}$'s secret (hence the error probability is the probability that the extractor is not successful).*

Definition 4.2 (Rational knowledge-of-exponent assumptions (RKEAs))
For $d, m, n \in \mathbf{N}^*$ and a group generator $\mathcal{G}$, we say that $(\mathcal{I}, \mathcal{V}, \overline{\mathcal{V}})$ is a (d, m, n)-RKEA if there is a uniform probabilistic algorithm $\mathcal{I}^{\text{core}}$ such that $\mathcal{I}$, $\mathcal{V}$ and $\overline{\mathcal{V}}$ are of the following forms.

- $(\mathsf{pub}, \mathsf{priv}) \leftarrow \mathcal{I}$:
  - $(G_p, [1]) \leftarrow \mathcal{G}$.
  - $\left(\left\{\frac{a_i(\mathbf{X})}{b_i(\mathbf{X})}\right\}_{i=1}^n, \mathsf{pub}', \mathsf{priv}'\right) \leftarrow \mathcal{I}^{\text{core}}(G_p)$, *where the a_is and b_is are polynomials in m variables and of total degree at most d.*
  - $\boldsymbol{x} \leftarrow \mathbf{F}_p^m$ *conditioned on $b_i(\boldsymbol{x}) \neq 0$ for all i.*
  - $\alpha \leftarrow \mathbf{F}_p$.
  - $\mathsf{pub} := \left(G_p, \left\{\left[\frac{a_i(\boldsymbol{x})}{b_i(\boldsymbol{x})}\right]\right\}_{i=1}^n, \left\{\left[\frac{\alpha \cdot a_i(\boldsymbol{x})}{b_i(\boldsymbol{x})}\right]\right\}_{i=1}^n, \left\{\frac{a_i(\mathbf{X})}{b_i(\mathbf{X})}\right\}_{i=1}^n, \mathsf{pub}'\right)$.

– *Return* $\big(\mathsf{pub}, \mathsf{priv} \coloneqq ([1], \boldsymbol{x}, \alpha, \mathsf{priv}')\big)$.

- $0/1 \leftarrow \mathcal{V}\big(\mathsf{pub}, \mathsf{priv}, \mathsf{sol} = ([u], [v])\big)$*: if* $[v] = \alpha[u]$*, return* 1*; else, return* 0.
- $0/1 \leftarrow \overline{\mathcal{V}}\big(\mathsf{pub}, \mathsf{priv}, \mathsf{sol}, \mathsf{sec} = (k_1, \ldots, k_n)\big)$*: if* $\sum_{i=1}^{n} k_i \big[\frac{a_i(\boldsymbol{x})}{b_i(\boldsymbol{x})}\big] = [u]$*, return* 1*; else, return* 0.

Remark 4.3
We note that in an RKEA the knowledge verifier $\overline{\mathcal{V}}$ does not use the private information priv; thus RKEAs would also satisfy an alternative definition of NIKAs where $\overline{\mathcal{V}}$ is not given priv.

Example 4.4 (q-PKE)
q-PKE is a $(q, 1, q{+}1)$-RKEA, meaning that $\mathcal{I}^{\mathrm{core}}$ generates $q{+}1$ rational functions consisting of polynomials in one variable and of degree at most q. Namely, we have $a_i(x) = x^{i-1}$ and $b_i(x) = 1$ for all $i = 1, \ldots, q+1$.

4.2 Simplifications of RKEAs

As a first step towards identifying Uber-assumptions for the class of RKEAs, in this section we define *simple RKEAs* analogously to the simple target assumptions of Ghadafi and Groth [15], and we show that simple RKEAs imply all RKEAs.

Definition 4.5 (Simple RKEAs)
We say that an RKEA is simple *if* $b_i(\mathbf{X}) = 1$ *for all* $i = 1, \ldots, n$, i.e., *all the rational functions output by* $\mathcal{I}^{\mathrm{core}}$ *are just polynomials.*

Theorem 4.6
For any (d, m, n)*-RKEA* $\mathsf{A} = (\mathcal{I}_{\mathsf{A}}, \mathcal{V}_{\mathsf{A}}, \overline{\mathcal{V}}_{\mathsf{A}})$ *there is an* (nd, m, n)*-simple RKEA* $\mathsf{B} = (\mathcal{I}_{\mathsf{B}}, \mathcal{V}_{\mathsf{B}}, \overline{\mathcal{V}}_{\mathsf{B}})$ *such that* B *implies* A.

Proof. The algorithm $\mathcal{I}^{\mathrm{core}}_{\mathsf{B}}$ of B proceeds as follows on input G_p.

- $\Big(\Big\{\frac{a_i(\mathbf{X})}{b_i(\mathbf{X})}\Big\}_{i=1}^{n}, \mathsf{pub}'_{\mathsf{A}}, \mathsf{priv}'_{\mathsf{A}}\Big) \leftarrow \mathcal{I}^{\mathrm{core}}_{\mathsf{A}}(G_p)$.
- $c_i(\mathbf{X}) \coloneqq a_i(\mathbf{X}) \cdot \prod_{j \neq i} b_j(\mathbf{X})$ for all $i = 1, \ldots, n$.
- $\mathsf{pub}'_{\mathsf{B}} \coloneqq \Big(\Big\{\frac{a_i(\mathbf{X})}{b_i(\mathbf{X})}\Big\}_{i=1}^{n}, \mathsf{pub}'_{\mathsf{A}}\Big)$; $\mathsf{priv}'_{\mathsf{B}} \coloneqq \mathsf{priv}'_{\mathsf{A}}$.
- Return $\big(\{c_i(\mathbf{X})\}_{i=1}^{n}, \mathsf{pub}'_{\mathsf{B}}, \mathsf{priv}'_{\mathsf{B}}\big)$.

Let $\mathcal{A}$ be an adversary against A; we first construct an adversary $\mathcal{B}$ against B which uses $\mathcal{A}$ in a black-box manner. $\mathcal{B}$'s input is

$$\mathsf{pub}_{\mathsf{B}} = \left(G_p, \left\{\left[c_i(\boldsymbol{x})\right]\right\}_{i=1}^{n}, \left\{\left[\alpha \cdot c_i(\boldsymbol{x})\right]\right\}_{i=1}^{n}, \left\{c_i(\mathbf{X})\right\}_{i=1}^{n}, \mathsf{pub}'_{\mathsf{B}}\right);$$

it runs $\mathcal{A}$ on input

$$\left(G_p, \left\{\left[c_i(\boldsymbol{x})\right]\right\}_{i=1}^{n}, \left\{\left[\alpha \cdot c_i(\boldsymbol{x})\right]\right\}_{i=1}^{n}, \left\{\frac{a_i(\mathbf{X})}{b_i(\mathbf{X})}\right\}_{i=1}^{n}, \mathsf{pub}'_{\mathsf{A}}\right)$$

and with its own random tape, and outputs the pair output by $\mathcal{A}$. Since we assume that B holds, there is an extractor $\chi_{\mathcal{B}}$ for $\mathcal{B}$ with negligible error probability ν; we construct an extractor $\chi_{\mathcal{A}}$ for $\mathcal{A}$ which uses $\chi_{\mathcal{B}}$ in a black-box manner. $\chi_{\mathcal{A}}$'s input is

$$\mathsf{pub}_{\mathsf{A}} = \left(G_p, \left\{\left[\frac{a_i(\boldsymbol{x})}{b_i(\boldsymbol{x})}\right]\right\}_{i=1}^{n}, \left\{\left[\frac{\alpha \cdot a_i(\boldsymbol{x})}{b_i(\boldsymbol{x})}\right]\right\}_{i=1}^{n}, \left\{\frac{a_i(\mathbf{X})}{b_i(\mathbf{X})}\right\}_{i=1}^{n}, \mathsf{pub}'_{\mathsf{A}}\right)$$

and its random tape is the same as that of $\mathcal{A}$. $\chi_{\mathcal{A}}$ runs $\chi_{\mathcal{B}}$ on input

$$\sigma_{\mathsf{B}} = \left(G_p, \left\{\left[\frac{a_i(\boldsymbol{x})}{b_i(\boldsymbol{x})}\right]\right\}_{i=1}^{n}, \left\{\left[\frac{\alpha \cdot a_i(\boldsymbol{x})}{b_i(\boldsymbol{x})}\right]\right\}_{i=1}^{n}, \left\{c_i(\mathbf{X})\right\}_{i=1}^{n}, \mathsf{pub}'_{\mathsf{B}}\right)$$

and with its own random tape, and outputs the values $(k_1, \ldots, k_n)$ output by $\chi_{\mathcal{B}}$. We claim that $\chi_{\mathcal{A}}$ is an extractor for $\mathcal{A}$ with (negligible) error probability at most $\nu + \frac{dn}{p}$.

We run the NIKA experiment for A. Firstly, $\mathcal{A}$ is run on input $\mathsf{pub}_{\mathsf{A}}$, and we let $([u], [v])$ be its output. Then, $\chi_{\mathcal{A}}$ is run, again on input $\mathsf{pub}_{\mathsf{A}}$ and with the same random tape as $\mathcal{A}$, and it runs $\chi_{\mathcal{B}}$ on input σ_{B} and with its own random tape, outputting the output $(k_1, \ldots, k_n)$ of $\chi_{\mathcal{B}}$. We claim that σ_{B} is distributed identically to $\mathsf{pub}_{\mathsf{B}}$ except with negligible probability. To see this, observe that $\mathcal{I}_{\mathsf{A}}$ generates the polynomials $a_i(\mathbf{X})$ and $b_i(\mathbf{X})$ as well as the vector $\boldsymbol{x}$ independently of the generator [1] output by $\mathcal{G}$. Further, assuming that $\prod_{i=1}^{n} b_i(\boldsymbol{x}) \neq 0$, the only difference between $\mathsf{pub}_{\mathsf{B}}$ and σ_{B} is the choice of generator; namely, if choosing the generator [1] yields the input $\mathsf{pub}_{\mathsf{B}}$, then choosing the generator $\left[\prod_{i=1}^{n} b_i(\boldsymbol{x})\right]$ yields σ_{B}.

By the Schwartz-Zippel lemma, the probability that $\prod_{i=1}^{n} b_i(\boldsymbol{x}) = 0$ is at most $\frac{dn}{p}$. Thus, letting $([u'], [v'])$ be the pair output by $\mathcal{B}$ we have

$$\left([v'] = \alpha[u']\right) \wedge \left(\sum_{i=1}^{n} k_i \left[\frac{a_i(\boldsymbol{x})}{b_i(\boldsymbol{x})}\right] \neq [u']\right)$$

with probability at most $\nu + \frac{dn}{p}$. Observing that $\mathcal{B}$, when run on input σ_{B}, runs $\mathcal{A}$ on input $\mathsf{pub}_{\mathsf{A}}$ and with the same random tape as that with which $\mathcal{A}$ was run originally shows that $([u'], [v']) = ([u], [v])$, which completes the proof. □

Chapter 5

Conclusion

In this thesis we have initiated an analysis of knowledge-of-exponent assumptions (KEAs), with the goal of obtaining a classification of those assumptions analogous to that obtained by Ghadafi and Groth for target assumptions [15]. Namely, we sought to identify a large class of KEAs, and then a small subclass thereof such that if all the assumptions in the small class hold, then all the assumptions in the large class hold as well. (The assumptions in the small class are then called "Uber-assumptions".) The aim of such a classification is to provide designers of cryptographic systems with a wide range of assumptions from which they may choose the most suitable one for their purposes, while at the same time ensuring that their validity relies on a small number of assumptions, which can then be thoroughly studied.

In Chapter 3 we studied the internal structure of the q-power knowledge-of-exponent (q-PKE) family of assumptions. This family of assumptions was introduced by Groth [19] as a generalisation of assumptions that had been introduced earlier by Damgård [8] and by Bellare and Palacio [3] (the latter as a modification of a faulty assumption of Hada and Tanaka [21, 22]), and was the most general instance of KEAs in the literature. We proved two results which give evidence that the assumptions in the q-PKE family increase in strength as the parameter q grows. Namely, we showed that, for any q, q-PKE unconditionally implies 0-PKE, and implies $(q-1)$-PKE under a decisional Diffie-Hellman-type assumption. The first result is an extension of a prior result of Bellare and Palacio (due to Halevi) [3], which showed merely that 1-PKE implies 0-PKE. Regarding the second result, we remark that its reliance on a decisional assumption is unfortunate, since it means that it does not apply in bilinear groups.

In Chapter 4, we introduced a further generalisation of the q-PKE family of assumptions, which we named *rational knowledge-of-exponent assumptions* (RKEAs), with the aim of capturing not only existing assumptions, but also others which may appear in the future. This is again motivated by the desire to have as large a class of assumptions as possible. Then, as a first step towards identifying Uber-assumptions for RKEAs, we show that all RKEAs are implied by the smaller subclass of *simple RKEAs*.

All our results were obtained using the proof technique from [3], which to our knowledge, is thus far the only known method for proving implications between KEAs. Indeed, to our knowledge, Proposition 4.2 from [3] was prior to this work the only result in the literature that proved an implication between two KEAs. Many directions for future work remain open, which might require the introduction of new proof techniques.

- Is the q-PKE family *strictly* increasing? In other words, can it be shown in some sense that q-PKE does *not* imply $(q+1)$-PKE?
- Can RKEAs be simplified further as in [15]? In particular, can Uber-assumptions be found? If not, can RKEAs be proved secure in the generic group model [6, 24, 29]?
- Can RKEAs be generalised further, for instance by allowing $\mathcal{V}$ and $\overline{\mathcal{V}}$ to be of a more general form?
- Perhaps most importantly, can a similar analysis be done in the bilinear setting, where KEAs are now primarily used?

Bibliography

[1] M. Abe and S. Fehr, *Perfect NIZK with Adaptive Soundness*. S. P. Vadhan (Ed.), *TCC 2007*, *Lecture Notes in Computer Science*, vol. 4392, pp. 118–136, Springer, 2007.
doi:10.1007/978-3-540-70936-7_7

[2] S. Arora and B. Barak, *Computational Complexity: A Modern Approach*. Cambridge University Press, New York, 2009.
http://theory.cs.princeton.edu/complexity/ (Accessed 9 July 2019.)

[3] M. Bellare and A. Palacio, *The Knowledge-of-Exponent Assumptions and 3-Round Zero-Knowledge Protocols*. M. Franklin (Ed.), *CRYPTO 2004*, *Lecture Notes in Computer Science*, vol. 3152, pp. 273–289, Springer, 2004.
doi:10.1007/978-3-540-28628-8_17

[4] D. Boneh, X. Boyen, and E.-J. Goh, *Hierarchical Identity Based Encryption with Constant Size Ciphertext*. R. Cramer (Ed.), *EUROCRYPT 2005*, *Lecture Notes in Computer Science*, vol. 3494, pp. 440–456, Springer, 2005.
doi:10.1007/11426639_26

[5] D. Boneh, C. Gentry, and B. Waters, *Collusion Resistant Broadcast Encryption with Short Ciphertexts and Private Keys*. V. Shoup (Ed.), *CRYPTO 2005*, *Lecture Notes in Computer Science*, vol. 3621, pp. 258–275, Springer, 2005.
doi:10.1007/11535218_16

[6] M. Chateauneuf, A. C. H. Ling, and D. R. Stinson, *Slope Packings and Coverings, and Generic Algorithms for the Discrete Logarithm Problem*. *Journal of Combinatorial Designs*, vol. 11, no. 1, pp. 36–50, Wiley, 2003.
doi:10.1002/jcd.10033

[7] R. Crandall and C. Pomerance, *Prime Numbers: A Computational Perspective*, second edition. Springer, New York, 2005.
doi:10.1007/0-387-28979-8

[8] I. Damgård, *Towards Practical Public Key Systems Secure Against Chosen Ciphertext Attacks*. J. Feigenbaum (Ed.), *CRYPTO '91*, *Lecture Notes in Computer Science*, vol. 576, pp. 445–456, Springer, 1992.
doi:10.1007/3-540-46766-1_36

[9] A. W. Dent, *The Hardness of the DHK Problem in the Generic Group Model*. *Cryptology ePrint Archive*, Report 2006/156, 2006.
https://ia.cr/2006/156 (Accessed 9 July 2019.)

[10] W. Diffie and M. E. Hellman, *New Directions in Cryptography*. *IEEE Transactions on Information Theory*, vol. 22, no. 6, pp. 644–654, IEEE, 1976.
doi:10.1109/TIT.1976.1055638

[11] T. ElGamal, *A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms*. *IEEE Transactions on Information Theory*, vol. 31, no. 4, pp. 469–472, IEEE, 1985.
doi:10.1109/TIT.1985.1057074

[12] J. B. Fraleigh, *A First Course in Abstract Algebra*, seventh edition. Pearson, 2003.

[13] R. Gennaro, C. Gentry, B. Parno, and M. Raykova, *Quadratic Span Programs and Succinct NIZKs without PCPs*. T. Johansson and P. Nguyen (Eds.), *EUROCRYPT 2013*, *Lecture Notes in Computer Science*, vol. 7881, pp. 626–645, Springer, 2013.
doi:10.1007/978-3-642-38348-9_37

[14] C. Gentry and D. Wichs, *Separating Succinct Non-Interactive Arguments From All Falsifiable Assumptions*. *STOC '11: Proc. of the Forty-third Annual ACM Symposium on Theory of Computing*, pp. 99–108, ACM, 2011.
doi:10.1145/1993636.1993651

[15] E. Ghadafi and J. Groth, *Towards a Classification of Non-interactive Computational Assumptions in Cyclic Groups*. T. Takagi and T. Peyrin (Eds.), *ASIACRYPT 2017*, Part II, *Lecture Notes in Computer Science*, vol. 10625,

pp. 66–96, Springer, 2017.
doi:10.1007/978-3-319-70697-9_3

[16] O. Goldreich, *On the Foundations of Modern Cryptography*. B. S. Kaliski Jr. (Ed.), *CRYPTO '97*, *Lecture Notes in Computer Science*, vol. 1294, pp. 46–74, Springer, 1997.
doi:10.1007/BFB0052227

[17] O. Goldreich, S. Micali, and A. Widgerson, *Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems. Journal of the ACM*, vol. 38, no. 3, pp. 690–728, ACM, 1991.
doi:10.1145/116825.116852

[18] S. Goldwasser and S. Micali, *Probabilistic Encryption. Journal of Computer and System Sciences*, vol. 28, no. 2, pp. 270–299, Academic Press, 1984.
doi:10.1016/0022-0000(84)90070-9

[19] J. Groth, *Short Pairing-Based Non-interactive Zero-Knowledge Arguments*. M. Abe (Ed.), *ASIACRYPT 2010*, *Lecture Notes in Computer Science*, vol. 6477, pp. 321–340, Springer, 2010.
doi:10.1007/978-3-642-17373-8_19

[20] A. Gut, *Probability: A Graduate Course*, second edition. *Springer Texts in Statistics*, Springer, New York, 2013.
doi:10.1007/978-1-4614-4708-5

[21] S. Hada and T. Tanaka, *On the existence of 3-round zero-knowledge protocols*. H. Krawczyk (Ed.), *CRYPTO '98*, *Lecture Notes in Computer Science*, vol. 1462, pp. 408–423, Springer, 1998. (Preliminary version of [22].)
doi:10.1007/BFB0055744

[22] S. Hada and T. Tanaka, *On the Existence of 3-Round Zero-Knowledge Protocols. Cryptology ePrint Archive*, Report 1999/009, 1999. (Full version of [21].)
https://ia.cr/1999/009 (Accessed 9 July 2019.)

[23] M. Naor, *On Cryptographic Assumptions and Challenges*. D. Boneh (Ed.), *CRYPTO 2003*, *Lecture Notes in Computer Science*, vol. 2729, pp. 96–109, Springer, 2003.
doi:10.1007/978-3-540-45146-4_6

[24] V. I. Nechaev, *Complexity of a determinate algorithm for the discrete logarithm. Mathematical Notes*, vol. 55, no. 2, pp. 165–172, Plenum, 1994.
doi:10.1007/BF02113297

[25] B. Parno, J. Howell, C. Gentry, and M. Raykova, *Pinocchio: Nearly Practical Verifiable Computation. 2013 IEEE Symposium on Security and Privacy*, pp. 238–252, IEEE, 2013.
doi:10.1109/SP.2013.47

[26] M. O. Rabin, *Probabilistic Algorithms*. J. F. Traub (Ed.), *Algorithms and Complexity: New Directions and Recent Results*, pp. 21–39, Academic Press, New York, 1976.

[27] R. L. Rivest, A. Shamir, and L. Adleman, *A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. Communications of the ACM*, vol. 21, no. 2, pp. 120–126, ACM, 1978.
doi:10.1145/359340.359342

[28] C. E. Shannon, *Communication Theory of Secrecy Systems. Bell System Technical Journal*, vol. 28, no. 4, pp. 656–715, AT&T, 1949.

[29] V. Shoup, *Lower Bounds for Discrete Logarithms and Related Problems*. W. Fumy (Ed.), *EUROCRYPT '97*, *Lecture Notes in Computer Science*, vol. 1233, pp. 256–266, Springer, 1997.
doi:10.1007/3-540-69053-0_18

[30] Zerocoin Electric Coin Company, *What are zk-SNARKs?*.
https://z.cash/technology/zksnarks/ (Accessed 9 July 2019.)

List of Publications

[1] F. Kraiem, S. Isobe, E. Koizumi, and H. Shizuya, *On the classification of knowledge-of-exponent assumptions in cyclic groups*. To appear in *Interdisciplinary Information Sciences*.