UNIVERSITY OF CALIFORNIA, SAN DIEGO

On Identification, Zero-Knowledge, and Plaintext-Aware-Encryption

A dissertation submitted in partial satisfaction of the
requirements for the degree Doctor of Philosophy
in
Computer Science

by

Adriana Maria Palacio

Committee in charge:

Professor Mihir Bellare, Chair
Professor Samuel R. Buss
Professor Sanjoy Dasgupta
Professor Adriano Garsia
Professor Daniele Micciancio

2006

The dissertation of Adriana Maria Palacio is approved, and it is acceptable in quality and form for publication on microfilm:

Chair

University of California, San Diego

2006

To my parents, Tomás and Mercedes,
my brother, Ricardo, and my sister, Mónica

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGEMENTS

I thank my advisor, Professor Mihir Bellare, for his continuous support during my doctoral studies. Mihir engendered my interest in the field of Cryptography, and showed me what research is all about. He taught me how to formulate questions, approach a problem from different angles, and express my ideas clearly. He encouraged me to become more critical and much more persistent. I want to thank Mihir especially for always being there to listen and offer advice, and for helping me get through difficult times.

I thank the members of my doctoral committee, Professor Samuel R. Buss, Professor Sanjoy Dasgupta, Professor Adriano Garsia, and Professor Daniele Micciancio, for supervising this dissertation. Thanks are also due to Professor Garrison Cottrell, Professor Charles Elkan, and Professor Victor Vianu, for their support and encouragement during my initial years in the graduate program.

I am very grateful for the financial assistance I received while at UCSD. I was supported by an NSF Graduate Research Fellowship; Mihir's NSF grants CCR-0098123, ANR-0129617 and CCR-0208842, and his IBM Faculty Partnership Development Award; and a MASEM Fellowship, funded by NSF and sponsored by UCSD.

My co-authors, Mihir, John Black, Alexandra (Sasha) Boldyreva, Marc Fischlin, Tadayoshi (Yoshi) Kohno, and Bogdan Warinschi, have my gratitude for sharing their ideas about research with me, and for making the experience of working with them very enjoyable.

Several friends and colleagues at UCSD have contributed significantly to my development as a researcher and as a person: Michel Abdalla, Mihir, Sasha, Alejandro Hevia, Matthew Hohlfeld, Yoshi, Jee Hea Lee, Vadim Lyubashevsky, Daniele, Sara More, Chanathip (Meaw) Namprempre, Gregory Neven, Saurabh Panjwani, Sherief Reda, Thomas Ristenpart, Bogdan, and Bianca Zadrozny. I am grateful for the opportunity to meet each of them, and I thank them for the many

thought-provoking conversations we have had. During the final year of my doctoral studies, I have enjoyed the company of Anton Mityagin, Sara Shoup, and Scott Yilek, and I am thankful for the laughs we have shared.

My longtime friends Adriana Arango, Adriana DaCosta, Alexandra Howell, Teresa Tsai, and Denise Leon, have been very supportive and encouraging, while reminding me of the importance of taking time to enjoy non-academic life. I thank each of them for their friendship. I am grateful to Adriana Arango for being there for me every step of the way. Her support has been invaluable. Thanks to Adriana DaCosta for her trust, concern, and loyalty.

I also wish to thank my dear friend Andrés Caicedo for his unconditional love and understanding, and for scolding me each time I go astray. Andrés's cynical humor has always managed to put a smile on my face. It has been wonderful to have a close friend that understands the experience of graduate school, with whom to exchange "war stories."

Thanks to Vadim for his constructive criticism about my job application materials, my dissertation, and my defense, and for all his additional help while I was preparing each of these. I appreciate his efforts to make my life easier during stressful times, and to get me to pay less attention to minutiae. He has brought me much happiness this year, and his love has had a healing effect on me. For that, I am especially grateful.

I am indebted to my parents, my brother, and my sister, for encouraging me endlessly, letting me know that they believe in me, and taking care of me when I needed it. This dissertation is dedicated to them, for without their help and encouragement, I do not think I would have completed it.

The text of Chapter 3, in part, is a reprint of the material as it appears in M. Bellare and A. Palacio, "GQ and Schnorr Identification Schemes: Proofs of Security against Impersonation under Active and Concurrent Attacks," *Advances in Cryptology - Crypto 2002 Proceedings*, Lecture Notes in Computer Science Vol. 2442, M. Yung ed., Springer-Verlag, 2002. The text of Chapter 4, in part, is a

reprint of the material as it appears in M. Bellare and A. Palacio, "The Knowledge-of-Exponent Assumptions and 3-Round Zero-Knowledge Protocols," *Advances in Cryptology - Crypto 2004 Proceedings*, Lecture Notes in Computer Science Vol. 3152, M. Franklin ed., Springer-Verlag, 2004. The text of Chapter 5, in full, is a reprint of the material as it appears in M. Bellare and A. Palacio, "Towards Plaintext-Aware Public-Key Encryption without Random Oracles," *Advances in Cryptology - Asiacrypt 2004 Proceedings*, Lecture Notes in Computer Science Vol. 3329, P. J. Lee ed., Springer-Verlag, 2004.

VITA

1973	Born, Chicago, IL
1998	B.S. in Systems and Computer Engineering, Universidad de los Andes, Bogotá, Colombia
1998	B.S. in Mathematics, Universidad de los Andes, Bogotá, Colombia
1998 - 2000	Software Engineer, OPUS Ingenería Ltda., Bogotá, Colombia
2004	M.S. in Computer Science, University of California, San Diego
2004	Summer Graduate Teaching Fellow, University of California, San Diego
2006	Ph.D. in Computer Science, University of California, San Diego

PUBLICATIONS

M. Bellare and A. Palacio. "GQ and Schnorr Identification Schemes: Proofs of Security against Impersonation under Active and Concurrent Attacks." In *Advances in Cryptology - Crypto 2002 Proceedings*, Lecture Notes in Computer Science Vol. 2442, M. Yung ed., Springer-Verlag, 2002.

M. Bellare, A. Boldyreva, and A. Palacio. "An Uninstantiable Random-Oracle-Model Scheme for a Hybrid-Encryption Problem." In *Advances in Cryptology - Eurocrypt 2004 Proceedings*, Lecture Notes in Computer Science Vol. 3027, C. Cachin and J. Camenisch eds, Springer-Verlag, 2004.

M. Bellare and A. Palacio. "The Knowledge-of-Exponent Assumptions and 3-Round Zero-Knowledge Protocols." In *Advances in Cryptology - Crypto 2004 Proceedings*, Lecture Notes in Computer Science Vol. 3152, M. Franklin ed., Springer-Verlag, 2004.

M. Bellare and A. Palacio. "Towards Plaintext-Aware Public-Key Encryption without Random Oracles." In *Advances in Cryptology - Asiacrypt 2004 Proceedings*, Lecture Notes in Computer Science Vol. 3329, P. J. Lee ed., Springer-Verlag, 2004.

M. Bellare and A. Palacio. "Protecting against Key Exposure: Strongly Key-Insulated Encryption with Optimal Threshold." *Applicable Algebra in Engineering, Communication and Computing*, Vol. 16, No. 6, Springer-Verlag, 2006.

FIELDS OF STUDY

Major Field: Computer Science

Studies in Cryptography.

Professor Mihir Bellare, University of California, San Diego

ABSTRACT OF THE DISSERTATION

On Identification, Zero-Knowledge, and Plaintext-Aware-Encryption

by

Adriana Maria Palacio
Doctor of Philosophy in Computer Science
University of California, San Diego, 2006
Professor Mihir Bellare, Chair

This dissertation studies three cryptographic tools: *identification schemes* – collections of algorithms that enable a party to identify itself to another without revealing information that would facilitate impersonation; *zero-knowledge proofs* – interactive protocols that efficiently demonstrate the validity of an assertion without conveying any additional knowledge; and *plaintext-aware encryption schemes* – public-key encryption protocols with the property that the "only" way to efficiently produce a valid ciphertext is to encrypt a message; hence the creator of a ciphertext must "know" the corresponding plaintext.

We first consider two of the most efficient and best-known identification schemes: GQ and Schnorr. The question of whether they can be proved secure against impersonation under active attack had remained open for over ten years. This dissertation provides such a proof for GQ based on the one-more-RSA-inversion assumption, an extension of the usual one-wayness assumption. It also provides such a proof for Schnorr based on a corresponding discrete-logarithm-related assumption. Both results extend to establish security against impersonation under concurrent attack.

Next, we falsify an assumption, here called KEA2, underlying the Hada-Tanaka 3-round negligible-error zero-knowledge protocol for $\mathbb{NP}$. Providing such a protocol is a challenging problem that has attracted considerable research effort.

The fact that KEA2 is false means that we "lose" one of the few positive results on this subject. To recover the result, we propose a modification of KEA2. After removing a small bug in the Hada-Tanaka protocol that renders it unsound, we obtain a 3-round, negligible-error zero-knowledge protocol for $\mathbb{NP}$ under the discrete-logarithm assumption and our new, suitably modified, assumption.

Finally, we address the problem of defining and achieving plaintext-aware encryption in the standard public-key setting. We provide definitions for three notions of increasing strength: PA0, PA1, PA2, chosen so that security against chosen-plaintext attack (IND-CPA) coupled with PA1 implies security against non-adaptive chosen-ciphertext attack (IND-CCA1), and IND-CPA coupled with PA2 implies security against adaptive chosen-ciphertext attack (IND-CCA2). Towards achieving the new notions, we show that a scheme due to Damgård, denoted DEG, and Cramer-Shoup "lite" are both PA0 under Damgård's DHK0 assumption, and PA1 under an extension of DHK0. DEG is thus the most efficient scheme proved IND-CCA1-secure.

1 Introduction

With the proliferation of computers and communications systems, digital information has become one of the most important resources of the economy and society at large. The development of effective techniques for securing digital information is vital to sustenance of modern society.

It is not possible to eradicate all risks associated to transmission of digital information, but communicating parties expect certain security objectives to be met. Typical requirements for digital communication include *privacy*, which guarantees that information is kept secret from all but those authorized to have it; *authenticity*, which ensures that the receiver of a message can determine that it has not been tampered with; and *entity authentication*, which enables corroboration of the identity of a party. These security goals, and many more, can be satisfied with the use of *cryptographic protocols* – communication protocols designed to withstand the influence of adversaries attempting to make them deviate from their specified functionality.

1.1 The role of cryptography

Protection of digital communications requires much more than the correct implementation of adequate cryptographic protocols. Appropriate security policies, legislation, physical security mechanisms, and measures to detect and document

threats are also necessary. Nevertheless, designing efficient quality cryptographic protocols is a crucial step in securing digital communication systems.

A *secure* communication protocol is one that achieves security goals such as those mentioned above. *Cryptography* is concerned with the design and analysis of such protocols. A cryptographic protocol, also called a *scheme*, is a collection of algorithms, one for each party involved in the communication. The algorithms themselves are usually publicly known, but they are a function of some secret information, called a *cryptographic key*, and security of the protocol relies on secrecy of this key.

Public-key cryptography, which was proposed in the revolutionary work of Diffie and Hellman [39], assumes that a party possesses a pair of keys – a *public key* and an associated *secret key*. The public key is bound to the party's identity and made publicly known; for example, by listing it in a public directory. The secret key is assumed to be kept securely by the party. (How this is accomplished is outside the scope of cryptography.) Henceforth, we will assume that the infrastructure required to support public-key cryptography is available.

Ad hoc approaches and heuristics for the design of cryptographic protocols have proved to be perilous. Proposed schemes are often broken, sometimes after they have been implemented and disseminated (e.g., wireless LAN protocol WEP [27, 46, 85], Diebold's AccuVote-TS voting system [63]). Protocol designers cannot envision all possible strategies that an adversary may use to attack a system. Therefore, countermeasures designed to prevent specific attacks are invariably insufficient. A more sound approach is required.

1.2 Practice-oriented provable security

Researchers, practitioners, and standards bodies have embraced a methodology for cryptographic protocol design called *practice-oriented provable security* [10, 12, 20]. This approach is a refinement of the *provable-security* approach introduced

in the seminal work of Goldwasser and Micali [55].

Provable security is achieved for a given cryptographic problem when one formally defines a security goal, provides a protocol to meet this goal, and proves that the protocol satisfies the security definition, under some widely-believed computational-hardness assumption. Formulating a security goal involves making a formal adversarial model that captures an attacker's capabilities, and defining exactly what it means for a protocol to be secure. The latter requires specifying what an adversary must do to be successful in attacking a scheme. A scheme is proved secure via a *reduction*. A reduction shows that the only way to break the protocol is to solve the underlying computationally-hard problem. Thus, a formal definition for this problem must also be provided. A security proof guarantees that the protocol not only withstands known attacks, but is secure against all (computationally-bounded) adversarial strategies as long as the underlying hardness assumption holds.

The impact of traditional provable security on practical cryptography was very limited due in part to the inefficiency and complexity of the provably-secure constructions first proposed. *Practice-oriented provable security* was introduced by Mihir Bellare and Phillip Rogaway [10, 12, 20] to address this issue and bring the benefits of provable security to cryptographic practice. This approach combines the principles of provable security with traditional practical cryptography, enabling the design of efficient, proved-secure protocols. It has had a significant impact on the application of cryptography in industry and has enriched both theory and practice. Indeed, modern cryptographic standards often employ provably-secure protocols and these are increasingly preferred to ad hoc schemes in real-world applications.

In this dissertation we employ the principles of practice-oriented provable security. These include a focus on concrete security analysis, which quantifies how much security a protocol provides, giving practitioners guidance in choosing among protocols. For each cryptographic problem considered, we present definitions for the security goals, describe protocols designed to achieve these goals, and analyze

these protocols. To show that a scheme is insecure, we present an attack, that is, a successful adversarial strategy. To show that a scheme is secure, we present a proof based on appropriate cryptographic assumptions, providing concrete bounds for the reductions. These bounds can then be used to compute the maximum probability of success possible for an adversary attacking the scheme, based on the best solution known for the underlying hard problem.

1.3 Cryptographic assumptions

Standard cryptographic assumptions involve well-studied problems that are believed to be difficult to solve using reasonable computational resources. A proof of security based on such an assumption is a strong, albeit not absolute, security guarantee. This guarantee is backed by decades of failed attempts to solve a well-known, simply-stated problem.

While provably-secure protocols guarantee protection against all attackers with the abilities specified in the adversarial model (if the underlying assumption holds), ad hoc protocols can at best guarantee protection against specific attacks. Thus, schemes designed using the provable-security approach and standard cryptographic assumptions have superior security guarantees.

Numerous protocols have been proposed for which there are no proofs of security and no known attacks. What can be said about the security of such schemes? The fact that an attack has not been discovered does not imply that none exists. Until a proof of security is provided, we cannot conclude that the scheme is secure. Unless an attack is found, we cannot conclude that it is insecure either. In such cases, less well-studied, sometimes even far-fetched, assumptions might be considered. Is there anything to gain from such musings?

Basing proofs of security on non-standard, but plausible, simply-stated assumptions frees cryptanalysts from the details of the protocols and the security models, allowing them to focus on disproving the assumptions. Furthermore, re-

ducing the security of several protocols (with the same or different security goals) to a single assumption, standard or not, helps to clarify and unify the global picture of protocol security by showing that the properties underlying the security of all of the schemes are the same. Thus, we consider a reduction to a new, less well-studied, or otherwise non-standard, cryptographic assumption to be of value.

With the above in mind, in this dissertation we use some recent RSA- and discrete-logarithm-based assumptions that are strong, but plausible, to provide security proofs for protocols whose security status was unknown for many years. We also make use of a couple of non-standard discrete-logarithm-related assumptions which are considered problematic by some, to achieve security goals that have not been met before. Our results are explained in the following section.

1.4 Contributions

We first briefly summarize the problems addressed in this dissertation and our contributions. Then we discuss each problem in turn, providing some background and motivation for our study, before presenting our results. Readers familiar with any of the problems discussed below may wish to skip the corresponding background and proceed to the results.

1.4.1 Summary of results

We consider three cryptographic problems: *identification*, *zero knowledge*, and *plaintext awareness*. A protocol that provides identification enables a party to prove its identity to another, without revealing information that would facilitate future impersonation. Zero-knowledge protocols are used to efficiently demonstrate the validity of an assertion without conveying any additional knowledge. Plaintext-aware protocols are *public-key encryption schemes* with the property that the "only" way to efficiently produce a valid ciphertext is to encrypt a message; hence the creator of a ciphertext must "know" the corresponding plaintext.

We begin by providing the first proofs of security for two efficient and well-known identification schemes: GQ and Schnorr, based solely on assumptions related to the underlying one-way functions. Then we falsify an assumption on which Hada and Tanaka base their 3-round, negligible-error, zero-knowledge arguments for $\mathbb{NP}$, demonstrating that it is possible to falsifying assumptions that do not lend themselves easily to "efficient falsification" (Naor [70]) due to their nature and quantifier structure. We also recover the results of Hada and Tanaka on 3-round zero knowledge, under a new assumption. Finally, we define and construct plaintext-aware encryption schemes in the standard setting of public-key cryptography. These results are described in more detail below.

1.4.2 Identification

Since the advent of automated teller machines (ATMs), banks and customers have had to deal with a new form of theft. ATMs have been rigged to record account information read from customers' cards. This information has then been used to manufacture counterfeit ATM cards. Using these forged cards, thieves have withdrawn millions of dollars from customer accounts.

In order to prevent ATMs from obtaining information that can be used to manufacture counterfeit cards, it has been proposed to use smart cards containing a secret that will enable identification to the bank. Instead of simply reading the account information from a magnetic stripe on the back of the card, the ATM facilitates an interactive protocol between the smart card and the bank computer. If the protocol is completed correctly, then the bank allows the transaction to procede; otherwise, it does not. Intuitively, by avoiding recovery of the secret stored on the smart card, which makes it possible for the latter to convince the bank of its identity, card forgery can be prevented.

An *identification (ID) scheme* [43] can be used by a smart card to identify itself to a bank. Such a scheme enables a party holding a secret key – the *prover* –

to identify itself to a party holding the corresponding public key – the *verifier*. The main notion of security for identification schemes is security against impersonation under *active* attack [43]. In such an attack, the adversary's goal is *impersonation*: playing the role of the prover, but denied the secret key, it attempts to make the verifier accept. Before the impersonation attempt, the adversary can play the role of cheating verifier, interacting with the prover numerous times to try to obtain information about the secret key. An example of an active attack occurs with the use of a forged ATM card manufactured using information obtained from a rigged ATM. Indeed, the rigged ATM interacts with the prover, possibly numerous times, before the actual impersonation attempt.

A weaker notion of security for ID schemes, namely security against impersonation under *passive* attack, considers adversaries that can eavesdrop on the communication between the prover and the verifier, but cannot interact with the prover directly. Note that security against impersonation under passive attack is not sufficient to solve the problem of rigged ATMs described above.

The *Guillou-Quisquater (GQ)* [58] and *Schnorr* [79] identification schemes are among the most efficient and best-known such schemes. They are known to be secure against impersonation under passive attack, assuming, respectively, hardness of RSA-inversion (i.e., *one-wayness* of RSA), and hardness of the *discrete-logarithm problem (DLP)*.

Our results

The question of whether the GQ [58] and Schnorr [79] identification schemes can be proved secure against impersonation under active attack had remained open for more than 10 years. In Chapter 3 of this dissertation, we provide such a proof for the GQ scheme based on the *one-more-RSA-inversion assumption*, an extension of the usual one-wayness assumption, which was introduced in [16]. We then provide such a proof for the Schnorr ID scheme based on the *one-more-discrete-logarithm assumption*, also introduced in [16], which is an extension of the usual discrete-loga-

rithm assumption (DLA). Our results extend to establish security against stronger attacks, namely *concurrent* ones, based on the same assumptions.

The one-more-RSA-inversion assumption was used in [16] to provide a security proof for Chaum's RSA-based blind-signature scheme [32]. It was also used in [17] to prove that an RSA-based transitive signature scheme due to Micali and Rivest [67] is secure. Our result about GQ then implies that the properties of RSA underlying the security of these three schemes are the same. Thus we obtain the benefit one usually expects with a proof of security, namely reduction of the security of several cryptographic protocols to a single number-theoretic problem. As discussed in Section 1.3, such reductions free cryptanalysts from the details of the protocols and security models, allowing them to focus on disproving a simply-stated assumption.

The benefits of our second result above are analogous to those of our result for the GQ scheme. Although the assumption used is relatively novel and strong, our proof reduces the security of the Schnorr identification scheme to a question about the hardness of a number-theoretic problem, thereby freeing cryptanalysts from consideration of attacks related to the identification problem itself.

The GQ and Schnorr ID schemes are Fiat-Shamir (FS) [45] follow-ons. FS is an efficient identification scheme based on a *zero-knowledge proof* due to Goldwasser, Micali, and Rackoff [56]. Feige, Fiat, and Shamir [45, 43] were the first to use zero-knowledge techniques to address the problem of identification. Their work paved the road for numerous successors, including GQ and Schnorr. These two schemes are comparable to FS in computational cost but have much smaller key sizes. FS, GQ, and Schnorr are 3-round *honest-verifier zero-knowledge* [13] *proofs of knowledge* [56, 9], under the standard hardness assumptions for factoring, RSA-inversion, and DLP, respectively.

We consider three-round zero-knowledge protocols in the following section.

1.4.3 Zero knowledge

Loosely speaking, a *zero-knowledge (ZK) proof system* [56] is a probabilistic and interactive protocol for two parties, called *prover* and *verifier*, that enables the computationally-unbounded prover to convince the computationally-bounded verifier of the validity of an assertion without revealing any additional knowledge (i.e., a verifier obtaining such a proof only gains conviction in the validity of the assertion). The *zero-knowledge* requirement is captured by saying that anything that is computable in polynomial time from the proof is also computable in polynomial time from the assertion itself, using a so-called *simulator*. The proof system must also satisfy a *completeness* condition requiring that if the assertion holds then the verifier always accepts, and a *soundness* condition requiring that if the assertion is false then the verifier rejects with "noticeable" probability, no matter what strategy is employed by a prover.

The ZK property of a proof system (or argument system, defined below) guarantees security against any adversarial strategy that a polynomial-time verifier may use in an attempt to extract knowledge from the prover who tries to convince the verifier to accept a valid assertion. The soundness property of a proof system (resp., computational-soundness property of an argument system) guarantees security against all possible (resp., polynomial-time) strategies that a prover may use in an attempt to fool the verifier into accepting a false assertion. It is desirable to have ZK protocols with *negligible soundness error*.

ZK proofs are widely applicable in cryptography and have had a dramatic effect on cryptographic protocol design. Their power comes from the fact that $\mathbb{NP}$-statements can be proved in ZK [53] using a prover strategy implemented by a probabilistic polynomial-time algorithm that is given as an auxiliary input an $\mathbb{NP}$-witness to the assertion to be proved (provided that one-way functions exist – a requirement for most of modern cryptography). This makes ZK proofs useful in the design of protocols for various tasks. They are often used to force malicious parties

to properly follow a given protocol (which requires parties to prove correctness of their actions, revealing nothing but the validity of the assertion).

As an example of an application of ZK in cryptography (due to [43]), consider the problem of identification discussed in Section 1.4.2. A party can prove its identity by demonstrating that it "knows" a witness for an $\mathbb{NP}$-statement. If the prover simply transmits the witness, an adversary that eavesdrops on one interaction in which the prover identifies itself, can then impersonate the prover. If the prover uses a ZK proof instead, the adversary will not be able to impersonate the prover even after eavesdropping on several successful interactions between the prover and the verifier. In fact, even a *cheating* verifier will not be able to impersonate the prover, that is, convince the honest verifier that it is the prover. The protocol is thus secure against impersonation under active attack.

Brassard, Chaum, and Crépeau [28] introduced a variant of ZK proof systems in which the soundness condition is replaced by a *computational-soundness* condition requiring that it is infeasible (not impossible) to fool the verifier into accepting false statements with high probability. These protocols are called *ZK arguments*. Arguments may be more efficient than proof systems (see [62]), and they suffice for many cryptographic applications.

The *round complexity* of a ZK protocol is the number of message exchanges. This measure is typically considered the most important efficiency criteria. It is desirable for the number of rounds of a ZK protocol to be a small constant.

Bellare, Jakobsson, and Yung [11] showed how to construct *4-round* negligible soundness-error ZK arguments for every language in $\mathbb{NP}$, assuming the existence of one-way functions.

Goldreich and Krawczyk [52] showed that only languages in $\mathbb{BPP}$ have *3-round* negligible-error proofs, or arguments, that are *black-box simulation ZK (BBZK)*. Roughly, BBZK requires that there exist a *universal* simulator that using any cheating verifier $\widehat{V}$ as a *black box*, produces a probability distribution like that of the distribution of conversations between $\widehat{V}$ and the prover. This notion

of ZK is stronger than the original one, which allows each verifier $\widehat{V}$ to have a particular simulator $S_{\widehat{V}}$.

For nearly two decades all known ZK protocols were BBZK. Since it was hard to imagine an alternative way to prove the ZK property of a given protocol, it was believed that impossibility results for BBZK hold also for ZK. Barak [3] refuted this belief by showing how to construct non-black-box simulators, and obtaining several results that were known to be unachievable using black-box simulators. These do not include, however, a 3-round, negligible-error ZK protocol for $\mathbb{NP}$.

Our results

Whether there exist 3-round negligible-error zero-knowledge proofs or arguments for $\mathbb{NP}$ is an intriguing open question in the theory of zero knowledge [56]. The difficulty in answering this question stems from the fact that such protocols would have to be non-black-box simulation ZK [52], and there are few approaches or techniques to this end. A positive answer has been provided, however, by Hada and Tanaka [59, 60]. They prove the existence of such arguments based on a pair of non-standard assumptions which we call *KEA1* and *KEA2* (for "***K****nowledge-of-****E****xponent* ***A****ssumption*s"). In Chapter 4 of this dissertation, we show that KEA2 is false. This renders vacuous the results of [59, 60]. We recover these results, however, under a suitably modified new assumption called *KEA3*.

What we believe is most interesting about these results is that we show that it is possible to "falsify" assumptions such as KEA2 that, due to their nature and quantifier structure, do not lend themselves easily to "*efficient falsification*" (Naor [70]) and do not appear to be even "*somewhat falsifiable*" [70]. Indeed, KEA2 (and KEA1) is an assumption of the form: "For every adversary there exists an extractor such that ...". To show that KEA2 is false, we must show that there is an adversary for which there exists no extractor. It is not difficult to identify an adversary for which there does not appear to exist an extractor, but how can one actually prove that none exists?

In contrast, most standard assumptions require that the probability that an adversary produces a certain output given certain inputs is negligible. (For example, DLA is of this type, asking that the probability that a polynomial-time adversary can output the discrete logarithm of a random group element is negligible.) To falsify such an assumption, one can present an "attack," in the form of an adversary whose success probability is not negligible. (For example, a polynomial-time algorithm that computes discrete logarithms.) Alas, we cannot apply this strategy for falsification of KEA2.

The approach we use instead is to define an adversary such that if there exists an extractor for it, then this extractor can be used to solve DLP. Hence our result is conditional: we show that if DLA holds, then KEA2 is false. We also observe that DLA is necessary to falsify KEA2.

The fact that KEA2 is false renders vacuous the proof of existence of 3-round negligible-error ZK arguments for $\mathbb{NP}$ presented in [60]. Constructing such ZK protocols (or proving that they only exist for $\mathbb{BPP}$ languages) has proved to be difficult thus far, so we would like to recover the result of [60]. To this end, we propose a modification of KEA2 that addresses the weakness we found in this assumption. We call the new assumption KEA3. The Hada-Tanaka protocol has a small bug that renders it unsound. We fix it and then prove that the resulting protocol is a 3-round, negligible-error ZK argument for $\mathbb{NP}$ under DLA, KEA3, and the variant of KEA1 used in [60] to prove the ZK property of the protocol.

Caveats. In recovering the results of [60] on 3-round ZK, we have not weakened the assumptions on which it is based, for KEA3 certainly remains a strong assumption of the same non-standard nature as KEA1.

The knowledge-of-exponent assumptions have been criticized for assuming that one can perform what some people call "reverse engineering" of an adversary. These critiques are certainly valid. Our results do not offer insight into this aspect of the assumptions, but by showing that such assumptions can be falsified, we open

the door to further analyses.

We emphasize that we have not found any weaknesses in KEA1. Recently, Dent [38] proved that this assumption is true in the *generic-group model* (cf. [81]) (where adversaries are restricted to be algorithms that do not exploit any special property of the encoding of group elements), thus providing some evidence for its validity.

KEA1 was proposed by Damgård [36], who used it to construct a *public-key encryption scheme* secure against certain *chosen-ciphertext attacks*. In the last chapter of this dissertation, we show that Damgård's scheme actually satisfies a stronger notion of security, namely, *plaintext awareness*, based on the same assumption.

Plaintext-aware public-key encryption is discussed in the following section.

1.4.4 Plaintext awareness

Encryption protocols are used to guarantee privacy when parties communicate through insecure channels. In the setting of public-key cryptography, a *sender* wishing to privately transmit data to a *receiver*, uses the receiver's public key to *encrypt* a message (aka *plaintext*). The resulting *ciphertext* is sent to the receiver, who uses its secret key to *decrypt* it and obtain the original message. Numerous such *public-key encryption schemes* have been constructed. The best-known examples are RSA [76], ElGamal [42], and Cramer-Shoup [33, 35].

The strongest notion of security for public-key encryption schemes suggested to date is *plaintext awareness*. Intuitively, an encryption scheme is plaintext aware if the "only" way to efficiently produce a valid ciphertext is to encrypt a message; hence, the creator of a ciphertext must "know" the corresponding plaintext. This notion was first defined by Bellare and Rogaway [22] who provided a formalization in the random-oracle (RO) model [21], where a random function is accessible through oracle calls to all parties, adversary included. The definition was enhanced

in [7] to show that security against chosen-plaintext attack (IND-CPA) coupled with plaintext awareness (PA) implies security against adaptive chosen-ciphertext attack (IND-CCA2), currently the target notion of security for public-key encryption schemes in practice.

The natural counterpart standard (i.e., non-RO) model definitions of PA are not achievable without sacrificing privacy.

Our results

Chapter 5 of this dissertation addresses the problem of defining and achieving plaintext-aware encryption in the standard public-key setting. We provide definitions for a hierarchy of notions of increasing strength: PA0, PA1 and PA2, chosen so that IND-CPA+PA1 $\rightarrow$ IND-CCA1 (security against non-adaptive chosen-ciphertext attacks) and IND-CPA+PA2 $\rightarrow$ IND-CCA2.

Towards achieving the new notions of plaintext awareness, we show that the scheme due to Damgård [36] mentioned above, which we refer to as DEG, and the "lite" version of the Cramer-Shoup scheme [35] (CSL) are both PA0 under a variant of the assumption KEA1 [36] discussed in Section 1.4.3, and PA1 under an extension of this assumption. As a result, DEG is the most efficient scheme proven IND-CCA1.

Dent [37] recently showed that some specific *hybrid encryption schemes* obtained using the Cramer Shoup scheme are PA2 under the assumption we used to prove that CSL is PA1. His result shows that PA2 is achievable, albeit under a strong assumption.

2 Preliminaries

In this chapter we introduce notation and definitions that will be used in several places throughout the dissertation.

2.1 Notation and terminology

We let $\mathbb{N} = \{1, 2, 3, \ldots\}$ be the set of positive integers. For $N \in \mathbb{N}$, $\mathbb{Z}_N$ denotes the ring of integers modulo N, $\mathbb{Z}_N^*$ denotes the multiplicative group of integers modulo N, and $\equiv_N$ denotes congruence modulo N. If $N \geq 1$ is an integer, then $|N|$ denotes the length of its binary encoding, i.e., the unique integer ℓ such that $2^{\ell-1} \leq N < 2^\ell$.

If S is a set, then $|S|$ denotes its size and $s \xleftarrow{\$} S$ denotes the operation of selecting an element $s \in S$ uniformly at random.

We denote by $\{0,1\}^*$ the set of all binary strings of finite length. The empty string is denoted ε. If x is a binary string, then we denote by $|x|$ its length and by $\bar{x}$ its bitwise complement. We denote by "$\|$" the string-concatenation operator, and by 1^k the string of $k \in \mathbb{N}$ ones.

If A is a deterministic algorithm, then $a \leftarrow A(x, y, \ldots)$ denotes the operation of assigning to a the outcome of the experiment of running A on inputs $x, y, \ldots$. If A is a randomized algorithm, then $A(x, y, \ldots; R)$ denotes its output on inputs $x, y, \ldots$ and coins R; $a \xleftarrow{\$} A(x, y, \ldots)$ denotes the result of picking R at random and

setting $a = A(x, y, \ldots; R)$; and $[A(x, y, \ldots)]$ denotes the set of all points having positive probability of being output by A on inputs $x, y, \ldots$.

For $k \in \mathbb{N}$, we say that an algorithm A is poly(k)-*time* if given inputs of length bounded by a polynomial in k, A halts in time bounded by a polynomial in k. A function $\nu\colon \mathbb{N} \to [0,1]$ is *negligible* if it approaches zero faster than the reciprocal of any polynomial, that is, for every $c \in \mathbb{N}$ there exists $k_c \in \mathbb{N}$ such that $\nu(k) \leq k^{-c}$ for all $k \geq k_c$.

2.2 Definitions

We first recall the formal definitions of the RSA, discrete-logarithm (DL), and decisional Diffie-Hellman (DDH) assumptions. Then in Section 2.2.2, we review the random-oracle model, which we will refer to in Chapter 5.

2.2.1 RSA, DL, and DDH assumptions

Assumption 2.2.1 [RSA one-wayness] Let $\mathcal{K}_{\text{rsa}}$ be a randomized, poly(k)-time algorithm that on input security parameter $k \in \mathbb{N}$, returns a triple (N, e, d) where N is the product of two distinct primes, $|N| = k$, $e < \varphi(N)$, $\gcd(d, \varphi(N)) = 1$, and $ed \equiv_{\varphi(N)} 1$. Let I be a randomized, polynomial-time algorithm that takes input N, e, y, where $y \in \mathbb{Z}_N^*$, and returns $x \in \mathbb{Z}_N^*$. We call I an *rsa-ow-adversary*. We associate to $\mathcal{K}_{\text{rsa}}$, I, and any $k \in \mathbb{N}$ the following experiment:

Experiment $\mathbf{Exp}^{\text{rsa-ow}}_{\mathcal{K}_{\text{rsa}},I}(k)$

$(N, e, d) \xleftarrow{\$} \mathcal{K}_{\text{rsa}}(k)\,;\ x \xleftarrow{\$} \mathbb{Z}_N^*\,;\ y \leftarrow x^e \bmod N\,;\ x' \xleftarrow{\$} I(N, e, y)$

If $(x' = x)$ then return 1 else return 0

We let

$$\mathbf{Adv}^{\text{rsa-ow}}_{\mathcal{K}_{\text{rsa}},I}(k) \;=\; \Pr\left[\,\mathbf{Exp}^{\text{rsa-ow}}_{\mathcal{K}_{\text{rsa}},I}(k) = 1\,\right]$$

denote the *rsa-ow-advantage* of I, the probability being over the coins of $\mathcal{K}_{\text{rsa}}$, the choice of x, and the coins of I. We say that *RSA one-wayness holds* (or *RSA is*

one-way) *for* $\mathcal{K}_{\rm rsa}$ if the function $\mathbf{Adv}^{\text{rsa-ow}}_{\mathcal{K}_{\rm rsa},I}(\cdot)$ is negligible for any rsa-ow-adversary I of time complexity polynomial in k. ▮

We adopt the convention that the *time complexity* of an rsa-ow-adversary I is the execution time of the entire experiment above.

Assumption 2.2.2 [DL assumption] Let $\mathcal{K}_{\rm dl}$ be a randomized, poly(k)-time algorithm that on input security parameter $k \in \mathbb{N}$, returns a pair (q, g) where q is a prime such that $q \,|\, p-1$ for a prime p with $|p| = k$, and g is a generator of G_q, a subgroup of $\mathbb{Z}_p^*$ of order q. Let I be a randomized, polynomial-time algorithm that takes input q, g, X, where $X \in G_q$, and returns $x' \in \mathbb{Z}_q$. We call I a *dl-adversary*. We associate to $\mathcal{K}_{\rm dl}$, I, and any $k \in \mathbb{N}$ the following experiment:

Experiment $\mathbf{Exp}^{\rm dl}_{\mathcal{K}_{\rm dl},I}(k)$

$(q, g) \xleftarrow{\$} \mathcal{K}_{\rm rsa}(k)\,;\ x \xleftarrow{\$} \mathbb{Z}_q\,;\ X \leftarrow g^x\,;\ x' \xleftarrow{\$} I(q, g, X)$

If $(g^{x'} = X)$ then return 1 else return 0

We let

$$\mathbf{Adv}^{\rm dl}_{\mathcal{K}_{\rm dl},I}(k) \;=\; \Pr\left[\,\mathbf{Exp}^{\rm dl}_{\mathcal{K}_{\rm dl},I}(k) = 1\,\right]$$

denote the *dl-advantage* of I, the probability being over the coins of $\mathcal{K}_{\rm dl}$, the choice of x, and the coins of I. We say that *DLA holds* (or *DLP is hard*) *for* $\mathcal{K}_{\rm dl}$ if the function $\mathbf{Adv}^{\rm dl}_{\mathcal{K}_{\rm dl},I}(\cdot)$ is negligible for any dl-adversary I of time complexity polynomial in k. ▮

We adopt the same convention regarding time complexity as in the case of an rsa-adversary.

Assumption 2.2.3 [DDH assumption] Let $\mathcal{K}_{\rm ddh}$ be a randomized, poly(k)-time algorithm that on input security parameter $k \in \mathbb{N}$, returns a triple (p, q, g) where p, q are primes with $|p| = k$ and $p = 2q + 1$, and g is a generator of G_q, a subgroup of $\mathbb{Z}_p^*$ of order q. Let D be a randomized, polynomial-time algorithm that takes input p, q, g, X, Y, Z, where $X, Y, Z \in G_q$, and returns a bit d. We call D a *ddh-adversary*. We associate to $\mathcal{K}_{\rm ddh}$, D, and any $k \in \mathbb{N}$ the following experiments:

Experiment $\mathbf{Exp}^{\text{ddh-1}}_{\mathcal{K}_{\text{ddh}},D}(k)$	Experiment $\mathbf{Exp}^{\text{ddh-0}}_{\mathcal{K}_{\text{ddh}},D}(k)$
$(p,q,g) \xleftarrow{\$} \mathcal{K}_{\text{rsa}}(k)$	$(p,q,g) \xleftarrow{\$} \mathcal{K}_{\text{rsa}}(k)$
$x \xleftarrow{\$} Z_q$	$x \xleftarrow{\$} Z_q$
$y \xleftarrow{\$} Z_q$	$y \xleftarrow{\$} Z_q$
$z \leftarrow xy \bmod q$	$z \xleftarrow{\$} Z_q$
$X \leftarrow g^x \,;\, Y \leftarrow g^y \,;\, Z \leftarrow g^z$	$X \leftarrow g^x \,;\, Y \leftarrow g^y \,;\, Z \leftarrow g^z$
$d \xleftarrow{\$} D(p,q,g,X,Y,Z)$	$d \xleftarrow{\$} D(p,q,g,X,Y,Z)$
Return d	Return d

We let

$$\mathbf{Adv}^{\text{ddh}}_{\mathcal{K}_{\text{ddh}},D}(k) \;=\; \Pr\left[\,\mathbf{Exp}^{\text{ddh-1}}_{\mathcal{K}_{\text{ddh}},D}(k) = 1\,\right] - \Pr\left[\,\mathbf{Exp}^{\text{ddh-0}}_{\mathcal{K}_{\text{ddh}},D}(k) = 1\,\right]$$

denote the *ddh-advantage* of D, the probability being over the coins of $\mathcal{K}_{\text{ddh}}$, the choices of x, y, z, and the coins of D. We say that the *DDH assumption holds* (or *DDH is hard*) *for* $\mathcal{K}_{\text{ddh}}$ if the function $\mathbf{Adv}^{\text{ddh}}_{\mathcal{K}_{\text{ddh}},D}(\cdot)$ is negligible for any ddh-adversary D of time complexity polynomial in k. ∎

We adopt the convention that the *time complexity* of a ddh-adversary D is the execution time of the worst-case experiment above.

2.2.2 Random-oracle model

A *random-oracle (RO) model* scheme is one whose algorithms have oracle access to a random function. Its security is evaluated with respect to an adversary with oracle access to the same function. An "instantiation" of such a scheme is the standard-model scheme obtained by replacing this function with a member of a polynomial-time computable family of functions, described by a short key. The security of the scheme is evaluated with respect to an adversary given the same key. In the *random-oracle paradigm*, as enunciated by Bellare and Rogaway [21], one first designs and proves secure a scheme in the RO model, and then instantiates it to get a (hopefully still secure) standard-model scheme.

The RO model is quite popular and there are now numerous practical schemes designed and proved secure in this model. A proof in the RO model, however,

does not guarantee security in the standard model. Numerous examples have been provided of *uninstantiable* schemes, i.e., schemes that meet their cryptographic goal in the RO model, but such that no instantiation of the scheme meets the goal in question (e.g., [29, 70, 57, 6]). Thus, we know that RO model schemes might not provide real-world security guarantees at all. Hence proofs of security in the standard model are usually considered preferable to RO-model proofs.

3 GQ and Schnorr Identification Schemes

3.1 Introduction

The Guillou-Quisquater (GQ) [58] and Schnorr [79] identification schemes are amongst the most efficient and best known Fiat-Shamir [45] follow-ons, but the question of whether they can be proved secure against impersonation under active attack has remained open. This chapter addresses this question, as well as its extension to even stronger attacks, namely concurrent ones. We begin with some background.

3.1.1 Identification schemes and their security

An identification (ID) scheme enables a prover holding a secret key to identify itself to a verifier holding the corresponding public key. Fiat and Shamir (FS) [45] showed how the use of zero-knowledge techniques [56] in this area could lead to efficient schemes, paving the road for numerous successors including [58, 79], which are comparable to FS in computational cost but have much smaller key sizes.

The accepted framework for security notions for identification schemes is that of Feige, Fiat and Shamir [43]. As usual, one considers adversary goals as well as adversary capabilities, or attacks. The adversary goal is impersonation: playing

the role of prover but denied the secret key, it tries to make the verifier accept. Towards this goal, one can allow it various attacks on the honest, secret-key equipped prover which, as per [43], take place and complete before the impersonation attempt. The weakest reasonable attack is a passive attack in which the adversary obtains transcripts of interactions between the prover and the verifier. The attack suggested by [43] as defining the main notion of security, however, is an active attack in which the adversary plays the role of cheating verifier, interacting with the prover numerous times before the impersonation attempt. An identification scheme is secure if all such adversaries have a negligible probability of success.

Security against impersonation under active attack has been the classical goal of identification schemes. Interest has been growing, however, in stronger attacks, namely concurrent ones. Here, the adversary still plays the role of cheating verifier prior to impersonation, but it can interact with many different prover "clones" concurrently. The clones all have the same secret key but are initialized with independent coins and maintain their own state. Security against impersonation under concurrent attack implies security against impersonation under active attack.

Analyses often approach the establishment of security against impersonation via consideration of whether or not the protocol is a proof of knowledge [9], honest-verifier zero knowledge [13], witness indistinguishable [44], and so on. These auxiliary properties are important and useful tools, but not the end goal, which remains establishing security against impersonation.

3.1.2 The GQ scheme and our results about it

GQ is RSA based. The prover's public key is (N, e, X), where N is an RSA modulus, e is a prime RSA exponent, and $X \equiv_N x^e$ where $x \in \mathbb{Z}_N^*$ is the prover's secret key. As typical for practical ID schemes, the protocol, depicted in Figure 3.4, has three moves: the prover sends a "commitment," the verifier sends a random challenge, the prover sends a "response," and the verifier then accepts or rejects.

The protocol is honest-verifier zero knowledge and a proof of knowledge of x [58], and it follows easily that it is secure against impersonation under passive attack, assuming RSA is one-way.

The main question is whether the protocol is secure against impersonation under active attack. No attack has been found. However, no proof of security has been provided either. Furthermore, it is difficult to imagine such a proof being based solely on the assumption that RSA is one-way. (The prover response is the RSA inverse of a point that is a function of the verifier challenge, giving a cheating verifier some sort of limited chosen-ciphertext attack capability, something one-wayness does not consider.) In other words, the protocol seems to be secure against impersonation under active attack, but due to properties of RSA that go beyond mere one-wayness.

The research community is well aware that RSA has important strengths beyond one-wayness, and have captured some of them with novel assumptions. Examples include the strong RSA assumption, introduced in [4, 47] and exploited in [49, 34]; the dependent-RSA assumptions [73]; and the one-more-RSA-inversion assumption [16]. The intent, or hope, of introducing such assumptions is that they underlie not one but numerous uses or protocols. Thus our approach is to attempt to build on this existing experience, and prove security based on one of these assumptions.

We prove that the GQ identification scheme is secure against impersonation, under both active and concurrent attacks, if the one-more-RSA-inversion assumption holds. The precise statement of the result is Corollary 3.4.3. Let us now explain the assumption.

The one-more-RSA-inversion assumption, as introduced in [16], considers an adversary given input an RSA public key N, e, and access to two oracles. The *challenge oracle* takes no inputs and returns a random target point in $\mathbb{Z}_N^*$, chosen anew each time the oracle is invoked. The *inversion oracle* given $y \in \mathbb{Z}_N^*$ returns $y^d \bmod N$, where d is the decryption exponent corresponding to e. The assump-

tion states that it is computationally infeasible for an adversary to output correct inverses of all the target points if the number of queries it makes to its inversion oracle is strictly less than the number of queries it makes to its challenge oracle. (When the adversary is allowed to make only one challenge query and no inversion queries, this is the standard one-wayness assumption, which is why the one-more-RSA-inversion assumption is considered an extension of the standard one-wayness assumption.) This assumption was used in [16] to prove the security of Chaum's RSA-based blind-signature scheme [32] in the random-oracle model [21]. (Our results, however, do not involve random oracles.) It was also used in [17] to prove the security of an RSA-based transitive signature scheme due to Micali and Rivest [67].

Our result is based on a relatively novel and strong assumption that should be treated with caution. But the result still has value. It reduces the security of the GQ identification scheme to a question that is solely about the security of the RSA function. Cryptanalysts need no longer attempt to attack the identification scheme, but can instead concentrate on a simply stated assumption about RSA, freeing themselves from the details of the identification model. Furthermore, our result helps clarify and unify the global picture of protocol security by showing that the properties of RSA underlying the security of the GQ identification scheme, Chaum's RSA-based blind-signature scheme, and Micali and Rivest's RSA-based transitive signature scheme are the same. Thus our result brings the benefit we usually expect with a proof of security, namely reduction of the security of many cryptographic problems to a single number-theoretic problem. Finally, a proof under a stronger-than-standard assumption is better than no proof at all in the context of a problem whose provable security has remained an open question for more than ten years.

3.1.3 The Schnorr scheme and our results about it

The Schnorr identification scheme is discrete-logarithm based. The prover's public key is (q, g, X), where g is a generator of a suitable group of prime order q, and $X = g^x$ where x is the prover's secret key. The protocol, having the usual three-move format, is depicted in Figure 3.6. Again the protocol is honest-verifier zero knowledge and a proof of knowledge of x [79], and it follows easily that it is secure against impersonation under passive attack, assuming hardness of computation of discrete logarithms in the underlying group. (That is, one-wayness of the discrete-exponentiation function.) As with GQ, the scheme appears to be secure against impersonation under active attack in the sense that no attacks are known, but proving security has remained open.

We prove that the Schnorr scheme is secure against impersonation, under both active and concurrent attacks, if the one-more-discrete-logarithm assumption holds in the underlying group. The precise statement of the result is Corollary 3.5.3. The assumption, also introduced in [16], is the natural analogue of the one we used for RSA. The adversary gets input the generator g. Its challenge oracle returns a random target point in the group, and its inversion oracle computes discrete logarithms relative to g. The assumption states that it is computationally infeasible for an adversary to output correct discrete logarithms of all the target points if the number of queries it makes to its inversion oracle is strictly less than the number of queries it makes to its challenge oracle. (When the adversary is allowed to make only one challenge query and no inversion queries, this is the standard discrete-logarithm assumption, meaning the standard assumption of one-wayness of the discrete-exponentiation function.)

The benefits of this result are analogous to those for GQ. Although the assumption is relatively novel and strong, our result reduces the security of the Schnorr identification scheme to a question about the hardness of a number-theoretic problem, thereby freeing a cryptanalyst from consideration of attacks

ID Scheme	POK	HVZK	WI	IMP-PA	IMP-AA	IMP-CA
Fiat-Shamir	Yes	Yes	Yes	Yes	Yes	Yes
GQ	Yes	Yes	No	Yes	**YES**	**YES**
2^m*-th root*	No	Yes	No	Yes	Yes	Unknown
Ong-Schnorr	No	Yes	No	Yes	Yes	Unknown
Schnorr	Yes	Yes	No	Yes	**YES**	**YES**
Okamoto	Yes	Yes	Yes	Yes	Yes	Yes

Figure 3.1 **Properties of popular ID schemes**. Results presented in this chapter are highlighted. The assumptions on which proofs of security against impersonation are based are described in Section 3.1.4.

---

related to the identification problem itself.

3.1.4 Discussion and related work

Within the large class of FS follow-on identification schemes, proven security properties vary. Some, like GQ and Schnorr, did not have proofs of security against active or concurrent attacks. The FS scheme itself, however, can be proved secure against impersonation under active and concurrent attacks assuming factoring is hard by exploiting its witness-indistinguishability (WI) and proof-of-knowledge (POK) properties. Okamoto's discrete-logarithm-based scheme [71] is also WI and a POK, and can thus be proved secure against impersonation under active and concurrent attacks, assuming hardness of the discrete-logarithm problem. Similar results hold for other schemes having the WI and POK properties. GQ and Schnorr, however, are not WI, since there is only one secret key corresponding to a given public key, so these techniques do not work for them. On the other hand, they are preferable in terms of cost. Both have smaller key size than FS, and Schnorr is more efficient than Okamoto.

The so-called 2^m-th root identification scheme can be viewed as the analogue of the GQ scheme with the RSA encryption exponent e replaced by a power of two, or as a special case of the Ong-Schnorr scheme [72]. The 2^m-th root scheme

and the Ong-Schnorr scheme have been proved secure against impersonation under active attack assuming factoring is hard [83, 80]. As far as we know, their security against impersonation under concurrent attack is an open question.

The security properties of the aforementioned schemes are summarized in Figure 3.1, where IMP-PA, IMP-AA, and IMP-CA denote, respectively, security against impersonation under passive, active, and concurrent attack.

Shoup [81] had proved that the Schnorr identification scheme is secure against impersonation under active attack in the generic-group model, where the attacker is restricted to be an algorithm that does not exploit any special property of the encoding of group elements. Our results are in the standard and less restrictive model where the adversary is an arbitrary algorithm.

The signature schemes obtained from the GQ and Schnorr identification schemes via the Fiat-Shamir transform are already known to be provably secure in the random-oracle model assuming, respectively, the one-wayness of RSA and the hardness of the discrete-logarithm problem [74], yet the security of the ID schemes against impersonation under active attack has remained open. This is not a contradiction, since the security of the signature scheme in the random-oracle model relies on relatively weak security properties of the ID scheme, namely the security of the latter against impersonation under passive attack [1].

Bellare et al. [15] showed that the GQ scheme can be transformed into an identity-based identification scheme, called GQ-IBI, via a random-oracle-using transform that preserves security. As observed in that paper, our result implies that GQ-IBI is secure against impersonation under active and concurrent attacks in the random-oracle model, if the one-more-RSA-inversion assumption holds. Bellare et al. also showed how to turn the signature scheme obtained from the GQ scheme via the Fiat-Shamir transform into an identity-based signature scheme, called GQ-IBS, using a security-preserving transform. It follows that GQ-IBS is provably secure in the random-oracle model assuming one-wayness of RSA. We comment that the schemes originally proposed by Guillou and Quisquater [58] are

actually GQ-IBI and GQ-IBS. In contrast, the Schnorr identification scheme has no counterpart identity-based identification or signature scheme.

Reset attacks (where the cheating verifier can reset the internal state of prover clones with which it interacts [30, 8]) are not considered here since GQ and Schnorr, as with all proof-of-knowledge-based schemes, are insecure against these attacks.

3.2 Definitions

3.2.1 ID schemes

An *identification (ID) scheme* $\mathcal{ID} = (\mathcal{K}, P, V)$ is a triple of randomized algorithms. On input security parameter $k \in \mathbb{N}$, the poly(k)-time key-generation algorithm $\mathcal{K}$ returns a pair consisting of a public key pk and a matching secret key sk. P and V are polynomial-time interactive algorithms that implement the prover and verifier, respectively. We require the natural correctness condition, namely that the boolean decision produced by V in the interaction in which P has input pk, sk and V has input pk, is 1 with probability one. This probability is over the coin tosses of both parties. We assume that the first and last moves in the interaction always belong to the prover.

The following security notion uses the basic two-phase framework of [43] in which, in a first phase, the adversary attacks the secret-key equipped P, and then, in a second phase, plays the role of cheating prover, trying to make V accept. We define and prove security only for impersonation under concurrent attack, since the usual (serial) active attack [43] is a special case of a concurrent attack.

3.2.2 Impersonation under concurrent attack

An *imp-ca-adversary* $A = (\widehat{V}, \widehat{P})$ is a pair of randomized polynomial-time algorithms, the *cheating verifier* and *cheating prover*, respectively. We consider a game having two phases. In the first phase, $\mathcal{K}$ is run on input k to produce

(pk, sk), a random tape is chosen for $\widehat{V}$ and it is given input pk. It then interacts concurrently with different *clones* of prover P, all clones having independent random tapes and being initialized with pk, sk. Specifically, we view P as a function that takes an incoming message and current state and returns an outgoing message and updated state. Cheating verifier $\widehat{V}$ can issue a request of the form (ε, i). As a result, a fresh random tape R_i is chosen, the initial state St_i of clone i is set to (pk, sk, R_i), the operation $(M_{\text{out}}, St_i) \leftarrow P(\varepsilon; St_i)$ is executed, M_{out} is returned to $\widehat{V}$, and the updated St_i is saved as the new state of clone i. Subsequently, $\widehat{V}$ can issue a request of the form (M, i), in which case message M is sent to clone i, who computes $(M_{\text{out}}, St_i) \leftarrow P(M; St_i)$, returns M_{out} to $\widehat{V}$, and saves the updated state St_i. These requests of $\widehat{V}$ can be arbitrarily interleaved. Eventually, $\widehat{V}$ outputs some state information St and stops, ending the first phase. In the second phase of the game, the cheating prover $\widehat{P}$ is initialized with St, verifier V is initialized with pk and freshly chosen coins, and $\widehat{P}$ and V interact. We say that adversary A *wins* if V accepts in this interaction, and the *imp-ca-advantage* of A, denoted

$$\mathbf{Adv}^{\text{imp-ca}}_{\mathcal{ID}, A}(k)$$

is the probability that A wins, taken over the coins of $\mathcal{K}$, the coins of $\widehat{V}$, the coins of the prover clones, and the coins of V. (There is no need to give $\widehat{P}$ separate coins, or even pk, since it can get them from $\widehat{V}$ via St.) We say that $\mathcal{ID}$ is *secure against impersonation under concurrent attack* (IMP-CA-secure) if the function

$$\mathbf{Adv}^{\text{imp-ca}}_{\mathcal{ID}, A}(\cdot)$$

is negligible for all imp-ca-adversaries A of time complexity polynomial in the security parameter k.

We adopt the convention that the *time complexity* of imp-ca-adversary A does not include the time taken by the prover clones and the verifier to compute replies to the adversary's requests. Rather we view these as oracles, each returning replies in unit time. Barring this, the time complexity of A is the execution time

of the entire two-phase game, including the time taken for key generation and initializations. This convention simplifies concrete security considerations.

An active attack [43] is captured by considering cheating verifiers that interact serially, one by one, with prover clones. (This means the cheating verifier initializes a clone and finishes interacting with it before starting up another one.)

3.2.3 Comments

We clarify that we do *not* allow reset attacks such as considered in [30, 8]: although $\widehat{V}$ can interact concurrently and in interleaved fashion with the prover clones, the internal state of a clone progresses in a normal serial fashion and cannot be reset by $\widehat{V}$. Indeed, the GQ and Schnorr protocols, object of our study, are both insecure under reset attacks.

We also clarify that we stay within the two-phase framework of [43] even while considering concurrent attacks, in the sense that the first phase (in which the adversary mounts a concurrent attack on the secret-key equipped P) is assumed to be completed before the start of the second phase (in which the adversary plays the role of cheating prover and tries to make V accept). This reflects applications such as smart card based identification for ATMs [43]. For identification over the Internet, it is more suitable to consider adversaries that can interact with the prover or prover clones even while they are interacting with the verifier in an attempt to make the latter accept. With this, one moves into the domain of authenticated key-exchange protocols which is definitionally more complex (see for example [20, 19, 82, 31]) and where identification without an associated exchange of a session-key is of little practical value.

3.3 Reset lemma

We refer to a three-move protocol of the form depicted in Figure 3.2 as *canonical*. The prover's first message is called its *commitment*. The verifier selects

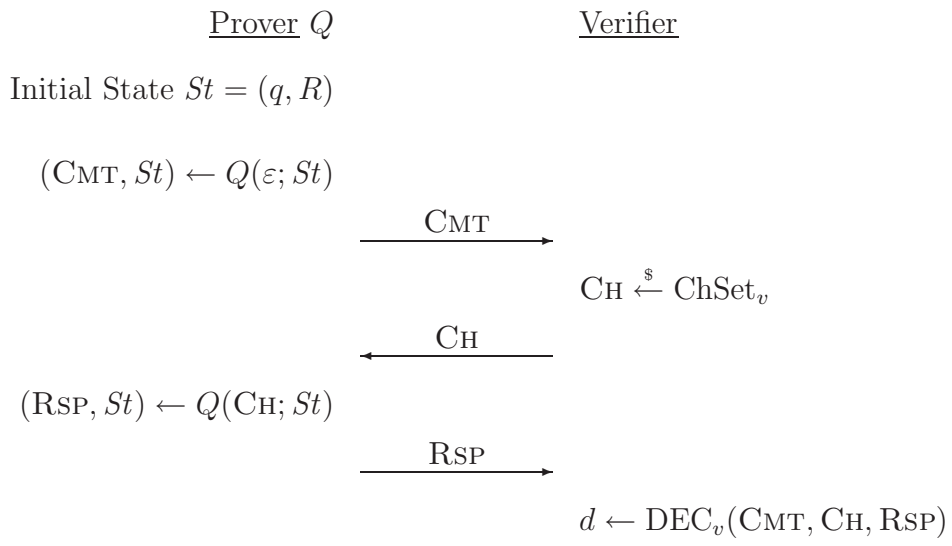

Figure 3.2 **A canonical protocol**. Prover Q has input q and random tape R, and maintains state St. The verifier has input v and returns boolean decision d.

---

a *challenge* uniformly at random from a set ChSet_v associated to its input v, and, upon receiving a *response* $\textsc{Rsp}$ from the prover, applies a deterministic *decision predicate* $\text{DEC}_v(\textsc{Cmt}, \textsc{Ch}, \textsc{Rsp})$ to compute a boolean decision. The verifier is said to be *represented* by the pair $(\text{ChSet}, \text{DEC})$ which, given the verifier input v, defines the challenge set and decision predicate.

A prover is identified with a function Q that given an incoming message M_{in} (this is ε when the prover is initiating the protocol) and its current state St, returns an outgoing message M_{out} and an updated state. The initial state of the prover is (q, R), where q is an input for the prover and R is a random tape.

The following lemma, which we call the Reset Lemma, upper bounds the probability that a (cheating) prover Q can convince the verifier to accept as a function of the probability that a certain experiment based on resetting the prover yields two accepting conversation transcripts. We will use this lemma in our proofs of security of both the GQ and the Schnorr schemes at the time of exploiting their proof-of-knowledge properties. In the past such analyses were based on the

techniques of [43] who considered certain "execution trees" corresponding to the interaction, and their "heavy nodes." The Reset Lemma provides a slightly better bound, has a simple proof, and is general enough to be applicable in numerous settings, saving the need to apply the techniques of [43] from scratch in each analysis, and may thus be of independent interest. Note that the lemma makes no mention of proofs of knowledge; it is just about relating two probabilities. The formulation and proof of the lemma generalize some analyses in [14].

Lemma 3.3.1 [Reset Lemma] Let Q be a prover in a canonical protocol with a verifier V represented by $(\mathrm{ChSet}, \mathrm{DEC})$. Let IG be a randomized algorithm that takes a security parameter $k \in \mathbb{N}$ and returns a pair of strings (q, v). We call IG an *input generator*. For any $(q, v) \in [\mathsf{IG}(k)]$, associate the experiments defined in Figure 3.3 to Q, V, q, v. For $k \in \mathbb{N}$, let

$$\begin{aligned}
\mathsf{acc}(k) &= \Pr\left[\, \mathbf{Exp}^{\mathrm{acc}}_{Q,V}(q,v) = 1 \;:\; (q,v) \xleftarrow{\$} \mathsf{IG}(k) \,\right], \\
\mathsf{res}(k) &= \Pr\left[\, \mathbf{Exp}^{\mathrm{reset}}_{Q,V}(q,v) = 1 \;:\; (q,v) \xleftarrow{\$} \mathsf{IG}(k) \,\right], \text{ and} \\
c(k) &= \min\{\, |\mathrm{ChSet}_v| \,:\, \text{there exists } q \text{ such that } (q,v) \in [\mathsf{IG}(k)] \,\}\,.
\end{aligned}$$

Then

$$\mathsf{res}(k) \;\geq\; \mathsf{acc}(k)\left(\mathsf{acc}(k) - \frac{1}{c(k)}\right)\,. \tag{3.1}$$

Alternatively,

$$\mathsf{acc}(k) \;\leq\; \frac{1}{c(k)} + \sqrt{\mathsf{res}(k)}\;. \quad \blacksquare \tag{3.2}$$

To prove Lemma 3.3.1 we will use the following standard fact, which can be derived from Jensen's inequality or as a consequence of the fact that the variance of any random variable is non-negative. For the sake of self-containment, we provide a direct proof based on the latter approach.

Lemma 3.3.2 Let X be a real-valued random variable. Then $\mathbf{E}\left[X^2\right] \geq \mathbf{E}\left[X^2\right]$.

Proof of Lemma 3.3.2: Let $\mu = \mathbf{E}\left[X\right]$. The random variable $(X-\mu)^2$ is non-negative. Thus

$$0 \;\leq\; \mathbf{E}\left[(X-\mu)^2\right] \;=\; \mathbf{E}\left[X^2\right] - 2\mu\mathbf{E}\left[X\right] + \mu^2 \;=\; \mathbf{E}\left[X^2\right] - 2\mu^2 + \mu^2 \;=\; \mathbf{E}\left[X^2\right] - \mu^2\,,$$

Experiment $\mathbf{Exp}^{\text{acc}}_{Q,V}(q,v)$ // $(q,v) \in [\mathsf{IG}(k)]$
Choose random tape R for Q ; $St \leftarrow (q,R)$; $(\text{CMT}, St) \leftarrow Q(\varepsilon; St)$
$\text{CH} \xleftarrow{\$} \text{ChSet}_v$; $(\text{RSP}, St) \leftarrow Q(\text{CH}; St)$; $d \leftarrow \text{DEC}_v(\text{CMT}, \text{CH}, \text{RSP})$
Return d

Experiment $\mathbf{Exp}^{\text{reset}}_{Q,V}(q,v)$ // $(q,v) \in [\mathsf{IG}(k)]$
Choose random tape R for Q ; $St \leftarrow (q,R)$; $(\text{CMT}, St) \leftarrow Q(\varepsilon; St)$
$\text{CH}_1 \xleftarrow{\$} \text{ChSet}_v$; $(\text{RSP}_1, St_1) \leftarrow Q(\text{CH}_1; St)$; $d_1 \leftarrow \text{DEC}_v(\text{CMT}, \text{CH}_1, \text{RSP}_1)$
$\text{CH}_2 \xleftarrow{\$} \text{ChSet}_v$; $(\text{RSP}_2, St_2) \leftarrow Q(\text{CH}_2; St)$; $d_2 \leftarrow \text{DEC}_v(\text{CMT}, \text{CH}_2, \text{RSP}_2)$
If ($d_1 = 1$ AND $d_2 = 1$ AND $\text{CH}_1 \neq \text{CH}_2$) then return 1 else return 0 EndIf

Figure 3.3 Experiments used to define functions acc and res in the Reset Lemma (Lemma 3.3.1).

---

and hence $\mathbf{E}[X^2] \geq \mu^2$. ∎

Proof of Lemma 3.3.1: Fix $k \in \mathbb{N}$. We will first establish Equation (3.1) and then show that it implies Equation (3.2). For $(q,v) \in [\mathsf{IG}(k)]$, let

$$\mathsf{acc}(q,v) \;=\; \Pr\left[\,\mathbf{Exp}^{\text{acc}}_{Q,V}(q,v) = 1\,\right] \;\text{ and }\; \mathsf{res}(q,v) \;=\; \Pr\left[\,\mathbf{Exp}^{\text{reset}}_{Q,V}(q,v) = 1\,\right].$$

We will show that for all $(q,v) \in [\mathsf{IG}(k)]$:

$$\mathsf{res}(q,v) \;\geq\; \mathsf{acc}(q,v)^2 - \frac{1}{|\text{ChSet}_v|} \cdot \mathsf{acc}(q,v)\,. \tag{3.3}$$

Therefore, if $\mathbf{E}$ denotes the expectation taken over $(q,v) \xleftarrow{\$} \mathsf{IG}(k)$, then

$$\begin{aligned}
\mathsf{res}(k) \;=\; \mathbf{E}[\,\mathsf{res}(q,v)\,] \;&\geq\; \mathbf{E}\left[\,\mathsf{acc}(q,v)^2 - \frac{1}{|\text{ChSet}_v|} \cdot \mathsf{acc}(q,v)\,\right] \\
&\geq\; \mathbf{E}\left[\,\mathsf{acc}(q,v)^2\,\right] - \mathbf{E}\left[\,\frac{1}{c(k)} \cdot \mathsf{acc}(q,v)\,\right] \\
&\geq\; \mathbf{E}[\,\mathsf{acc}(q,v)\,]^2 - \frac{1}{c(k)} \cdot \mathbf{E}[\,\mathsf{acc}(q,v)\,] \\
&=\; \mathsf{acc}(k)^2 - \frac{1}{c(k)} \cdot \mathsf{acc}(k) \\
&=\; \mathsf{acc}(k)\left(\mathsf{acc}(k) - \frac{1}{c(k)}\right),
\end{aligned}$$

where the third inequality above follows from Lemma 3.3.2. This establishes Equation (3.1).

We now prove that Equation (3.3) holds for all $(q, v) \in [\mathsf{IG}(k)]$. Fix such (q, v) and let r denote the length of the prover's random tape. For $R \in \{0,1\}^r$ and $\textsc{Ch} \in \mathrm{ChSet}_v$, let $d(R, \textsc{Ch})$ denote the verifier's boolean decision when Q has input q and random tape R, and V has input v and selects challenge $\textsc{Ch}$. We define functions $\mathsf{X}, \mathsf{Y} \colon \{0,1\}^r \to [0,1]$ as follows. For each $R \in \{0,1\}^r$, let

$$\mathsf{X}(R) \;=\; \Pr\left[\, d(R, \textsc{Ch}) = 1 \,\right],$$

the probability being over a random choice of $\textsc{Ch}$ from ChSet_v. For each $R \in \{0,1\}^r$, let

$$\mathsf{Y}(R) \;=\; \Pr\left[\, d(R, \textsc{Ch}_1) = 1 \wedge d(R, \textsc{Ch}_2) = 1 \wedge \textsc{Ch}_1 \neq \textsc{Ch}_2 \,\right],$$

the probability being over random and independent choices of $\textsc{Ch}_1$ and $\textsc{Ch}_2$ from ChSet_v. Then for any $R \in \{0,1\}^r$,

$$\begin{aligned}
\mathsf{Y}(R) \;&=\; \Pr\left[\, d(R, \textsc{Ch}_1) = 1 \,\right] \cdot \Pr\left[\, d(R, \textsc{Ch}_2) = 1 \wedge \textsc{Ch}_1 \neq \textsc{Ch}_2 \;\middle|\; d(R, \textsc{Ch}_1) = 1 \,\right] \\
&=\; \mathsf{X}(R) \cdot \Pr\left[\, d(R, \textsc{Ch}_2) = 1 \wedge \textsc{Ch}_1 \neq \textsc{Ch}_2 \;\middle|\; d(R, \textsc{Ch}_1) = 1 \,\right] \\
&\geq\; \mathsf{X}(R) \cdot \Big(\Pr\left[\, d(R, \textsc{Ch}_2) = 1 \;\middle|\; d(R, \textsc{Ch}_1) = 1 \,\right] \;- \\
&\qquad\qquad \Pr\left[\, \textsc{Ch}_1 = \textsc{Ch}_2 \;\middle|\; d(R, \textsc{Ch}_1) = 1 \,\right] \Big) \\
&\geq\; \mathsf{X}(R) \cdot \Big(\Pr\left[\, d(R, \textsc{Ch}_2) = 1 \,\right] - \Pr\left[\, \textsc{Ch}_1 = \textsc{Ch}_2 \,\right] \Big) \\
&=\; \mathsf{X}(R) \cdot \left(\mathsf{X}(R) - \frac{1}{|\mathrm{ChSet}_v|} \right)
\end{aligned}$$

We view X, Y as random variables over the sample space $\{0,1\}^r$ of coins of Q. Then letting $p = 1/|\mathrm{ChSet}_v|$ and using the above we have

$$\begin{aligned}
\mathsf{res}(q, v) \;=\; \mathbf{E}\left[\mathsf{Y}\right] \;&\geq\; \mathbf{E}\left[\mathsf{X} \cdot (\mathsf{X} - p)\right] \\
&=\; \mathbf{E}\left[\mathsf{X}^2\right] - p \cdot \mathbf{E}\left[\mathsf{X}\right] \\
&\geq\; \mathbf{E}\left[\mathsf{X}\right]^2 - p \cdot \mathbf{E}\left[\mathsf{X}\right] \\
&=\; \mathsf{acc}(q, v)^2 - p \cdot \mathsf{acc}(q, v)\,.
\end{aligned}$$

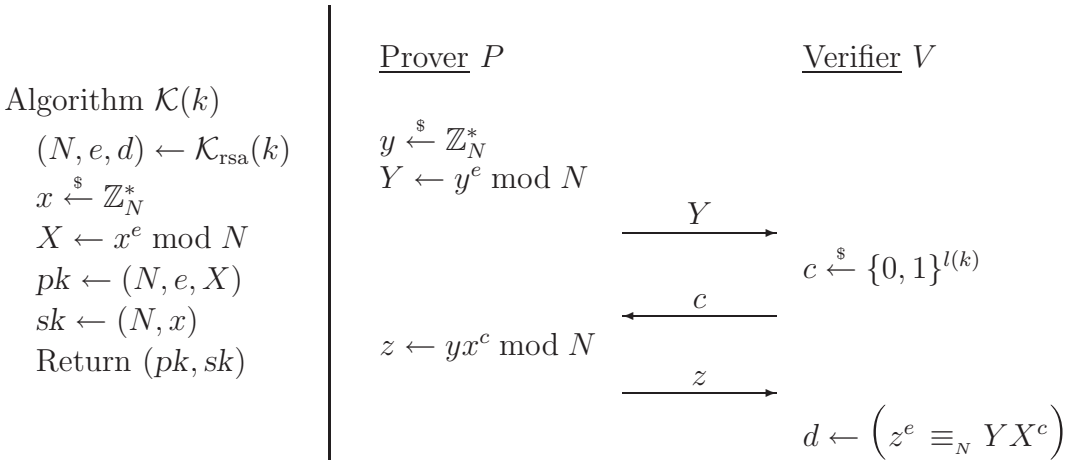

Figure 3.4 **GQ identification scheme**. Prover P has input $pk = (N, e, X)$ and $sk = (N, x)$. Verifier V has input pk and returns boolean decision d.

In the third line above, we used Lemma 3.3.2. This completes the proof of Equation (3.3) and thus of Equation (3.1).

We now show how to obtain Equation (3.2). Using Equation (3.1) we have

$$\left(\mathsf{acc}(k) - \frac{1}{2c(k)}\right)^2 \;=\; \mathsf{acc}(k)^2 - \frac{1}{c(k)} \cdot \mathsf{acc}(k) + \frac{1}{4c(k)^2} \;\leq\; \mathsf{res}(k) + \frac{1}{4c(k)^2} \;.$$

Taking the square-root of both sides of the above, and using the fact that $\sqrt{a+b} \leq \sqrt{a} + \sqrt{b}$ for all real numbers $a, b \geq 0$, we get

$$\mathsf{acc}(k) - \frac{1}{2c(k)} \;\leq\; \sqrt{\mathsf{res}(k) + \frac{1}{4c(k)^2}} \;\leq\; \sqrt{\mathsf{res}(k)} + \sqrt{\frac{1}{4c(k)^2}} \;=\; \sqrt{\mathsf{res}(k)} + \frac{1}{2c(k)} \;.$$

Re-arranging terms and simplifying gives us the desired conclusion. ∎

3.4 Security of GQ under concurrent attack

A randomized, poly(k)-time algorithm $\mathcal{K}_{\rm rsa}$ is said to be a *prime-exponent RSA key generator* if on input security parameter $k \in \mathbb{N}$, its output is a triple (N, e, d) where N is the product of two distinct primes, $|N| = k$ (N is k bits long),

$e < \varphi(N)$ is an odd prime, $\gcd(d, \varphi(N)) = 1$, and $ed \equiv_{\varphi(N)} 1$. We do not pin down any specific such generator. Rather it is a parameter of the GQ identification scheme, and security is proved based on an assumption about it.

3.4.1 GQ identification scheme

Let $\mathcal{K}_{\text{rsa}}$ be a prime-exponent RSA key generator and let $l\colon \mathbb{N} \to \mathbb{N}$ be a polynomial-time computable, polynomially bounded function such that $2^{l(k)} < e$ for any e output by $\mathcal{K}_{\text{rsa}}$ on input k. The *GQ identification scheme* associated to $\mathcal{K}_{\text{rsa}}$ and challenge length l is the ID scheme whose constituent algorithms are depicted in Figure 3.4. The prover's commitment is a random element $Y \in \mathbb{Z}_N^*$. For any verifier input $pk = (N, e, X)$, $\text{ChSet}_{pk} = \{0,1\}^{l(k)}$. A challenge $c \in \text{ChSet}_{pk}$ is interpreted in the natural way as an integer in the set $\left\{0, \ldots, 2^{l(k)} - 1\right\}$ in the ensuing computations. Due to the assumption that $2^{l(k)} < e$, the challenge is in $\mathbb{Z}_e$. The verifier's decision predicate $\text{DEC}_{pk}(Y, c, z)$ evaluates to 1 if and only if z is the RSA-inverse of $YX^c \bmod N$.

3.4.2 RSA assumption

We recall the one-more-RSA-inversion assumption [16], RSA-OMI.

Assumption 3.4.1 [One-more-RSA-inversion: RSA-OMI] Let $\mathcal{K}_{\text{rsa}}$ be a prime-exponent RSA key generator. Let I be a randomized, polynomial-time algorithm that takes input N, e and has access to two oracles. The first is an *RSA-inversion oracle* $(\cdot)^d \bmod N$ that given $Y \in \mathbb{Z}_N^*$ returns $Y^d \bmod N$, where d is the decryption exponent corresponding to e. The second is a *challenge oracle* $\mathcal{O}_N$ that takes no inputs and returns a random challenge point $W \in \mathbb{Z}_N^*$ each time it is invoked. We call I an *rsa-omi-adversary*. We associate to $\mathcal{K}_{\text{rsa}}$, I, and any $k \in \mathbb{N}$ the following experiment:

Experiment $\mathbf{Exp}^{\text{rsa-omi}}_{\mathcal{K}_{\text{rsa}},I}(k)$

$(N, e, d) \xleftarrow{\$} \mathcal{K}_{\text{rsa}}(k)\,;\; w_1, \ldots, w_n \xleftarrow{\$} I^{(\cdot)^d \bmod N,\, \mathcal{O}_N}(N, e)$

Let $W_1, \ldots, W_n$ denote the challenges returned by $\mathcal{O}_N$ in response to queries from I, and m denote the number of queries made by I to its RSA-inversion oracle

If ($w_i \equiv_N W_i^d$ for $i = 1, \ldots, n$ AND $m < n$) then return 1 else return 0

We let

$$\mathbf{Adv}^{\text{rsa-omi}}_{\mathcal{K}_{\text{rsa}},I}(k) \;=\; \Pr\left[\,\mathbf{Exp}^{\text{rsa-omi}}_{\mathcal{K}_{\text{rsa}},I}(k) = 1\,\right]$$

denote the *rsa-omi-advantage* of I, the probability being over the coins of $\mathcal{K}_{\text{rsa}}$, the coins of I, and the coins used by the challenge oracle across its invocations. We say that $\mathcal{K}_{\text{rsa}}$ is *RSA-OMI-secure* if the function $\mathbf{Adv}^{\text{rsa-omi}}_{\mathcal{K}_{\text{rsa}},I}(\cdot)$ is negligible for any rsa-omi-adversary I of time complexity polynomial in k. ▮

We adopt the convention that the *time complexity* of an rsa-omi-adversary I is the execution time of the entire experiment, including the time taken for key generation and one time unit for each reply to an oracle query. (The time taken by the oracles to compute replies to the adversary's queries is not included.)

3.4.3 Result

The following theorem shows that the advantage of any imp-ca-adversary against the GQ scheme can be upper bounded via the advantage of a related rsa-omi-adversary and a function of the challenge length. The theorem shows the concrete security of the reduction.

Theorem 3.4.2 Let $\mathcal{ID} = (\mathcal{K}, P, V)$ be the GQ identification scheme associated to prime-exponent RSA key generator $\mathcal{K}_{\text{rsa}}$ and challenge length l. Let $A = (\widehat{V}, \widehat{P})$ be an imp-ca-adversary of time complexity $t(\cdot)$ attacking $\mathcal{ID}$. Then there exists

an rsa-omi-adversary I attacking $\mathcal{K}_{\text{rsa}}$ such that for every $k \in \mathbb{N}$:

$$\mathbf{Adv}^{\text{imp-ca}}_{\mathcal{ID},A}(k) \;\leq\; 2^{-l(k)} + \sqrt{\mathbf{Adv}^{\text{rsa-omi}}_{\mathcal{K}_{\text{rsa}},I}(k)} \;. \tag{3.4}$$

Furthermore, the time complexity of I is $2t(k) + O(k^4 + (n(k)+1) \cdot l(k) \cdot k^2)$, where $n(k)$ is the number of prover clones with which $\widehat{V}$ interacts. ▌

Based on this theorem, which we will prove later, we can easily provide the following security result for the GQ scheme. In this result, we assume that the challenge length l is super-logarithmic in the security parameter, which means that $2^{-l(\cdot)}$ is negligible. This assumption is necessary, since otherwise the GQ scheme can be broken merely by guessing the verifier's challenge.

Corollary 3.4.3 If prime-exponent RSA key generator $\mathcal{K}_{\text{rsa}}$ is RSA-OMI-secure and challenge length l satisfies $l(k) = \omega(\log(k))$, then the GQ identification scheme associated to $\mathcal{K}_{\text{rsa}}$ and l is secure against impersonation under both active and concurrent attacks. ▌

Proof of Corollary 3.4.3: Let A be an imp-ca-adversary of polynomial time complexity attacking $\mathcal{ID}$. Then the rsa-omi-adversary given by Theorem 3.4.2 also has polynomial time complexity. The assumption that $\mathcal{K}_{\text{rsa}}$ is RSA-OMI-secure implies that $\mathbf{Adv}^{\text{rsa-omi}}_{\mathcal{K}_{\text{rsa}},I}(\cdot)$ is negligible, and the condition on the challenge length implies that $2^{-l(\cdot)}$ is negligible. Equation (3.4) then implies that $\mathbf{Adv}^{\text{imp-ca}}_{\mathcal{ID},A}(\cdot)$ is negligible. This shows that any imp-ca-adversary of polynomial time complexity attacking the scheme has a negligible advantage. Since an active attack is a particular case of a concurrent attack, the conclusion holds. ▌

We proceed to prove Theorem 3.4.2.

Proof of Theorem 3.4.2: We assume wlog that $\widehat{V}$ never repeats a request. Fix $k \in \mathbb{N}$ and let (N, e, d) be an output of $\mathcal{K}_{\text{rsa}}$ running on input k. Adversary I has access to an RSA-inversion oracle $(\cdot)^d \bmod N$ and a challenge oracle $\mathcal{O}_N$ that takes

no inputs and returns a random challenge point $W \in \mathbb{Z}_N^*$ each time it is invoked. The adversary's goal is to invert all the challenges returned by $\mathcal{O}_N$, while making fewer queries to its RSA-inversion oracle then the number of such challenges.

A detailed description of the adversary is in Figure 3.5. It first queries its challenge oracle to obtain a random element $W_0 \in \mathbb{Z}_N^*$ and uses it to create a public key pk for the imp-ca-adversary A. It then uses A to achieve its goal by running $\widehat{V}$ and playing the role of the prover clones to answer its requests. In response to a request of the form (ε, i), I queries its challenge oracle $\mathcal{O}_N$ and returns the answer W_i to $\widehat{V}$. By the definition of prover P, from $\widehat{V}$'s perspective, this is equivalent to picking a random tape R_i for prover clone i, initializing clone i with state pk, R_i, computing clone i's commitment W_i, and returning the commitment to $\widehat{V}$. I is not in possession of the secret key $sk = (N, W_0^d \bmod N)$ corresponding to pk, which the prover clones would use to respond to $\widehat{V}$'s requests of the form (c, i), where $c \in \{0,1\}^{l(k)}$, but it compensates using its access to the RSA-inversion oracle to answer these requests. Specifically, in response to request (c, i), I makes the query $W_i W_0^c$ to its inversion oracle and returns the answer z_i to $\widehat{V}$. Since $z_i = (W_i W_0^c)^d \bmod N = W_i^d (W_0^d)^c \bmod N$, this is exactly the response that clone i would return to $\widehat{V}$. Hence I simulates the behavior of the prover clones perfectly.

If $n(k)$ is the number of prover clones with which $\widehat{V}$ interacts, when $\widehat{V}$ stops I has made $n(k)$ queries to its RSA-inversion oracle and it needs to invert $n(k)+1$ challenge points. It cannot use the inversion oracle to obtain the desired inverses. Instead, I attempts to extract from $\widehat{P}$, initialized with the output of $\widehat{V}$, the RSA-inverse of challenge W_0. It can then use this value to compute the inverse of each of the other challenge points. To do so, I runs $\widehat{P}$ obtaining its commitment, selects an $l(k)$-bit challenge uniformly at random, runs $\widehat{P}$ to obtain its response to this challenge, and evaluates the verifier's decision predicate. It then selects another random challenge, re-runs $\widehat{P}$ (with the same state as before) to obtain its response to the new challenge, and evaluates the verifier's decision predicate.

Adversary $I^{(\cdot)^d \bmod N, \mathcal{O}_N}(N, e)$

Make a query to $\mathcal{O}_N$ and let W_0 be the response; $pk \leftarrow (N, e, W_0)$
Choose a random tape R for $\widehat{V}$; Initialize $\widehat{V}$ with (pk, R); $n \leftarrow 0$
Run $\widehat{V}$ answering its requests as follows:
  When $\widehat{V}$ issues a request of the form (ε, i) do
    $n \leftarrow n + 1$; Make a query to $\mathcal{O}_N$, let W_i be the response and return W_i to $\widehat{V}$
  When $\widehat{V}$ issues a request of the form (c, i), where $c \in \{0,1\}^{l(k)}$, do
    $c_i \leftarrow c$; Make query $W_i W_0^{c_i}$ to $(\cdot)^d \bmod N$, let z_i be the response and return z_i to $\widehat{V}$
Until $\widehat{V}$ outputs state information St and stops
$R \leftarrow \varepsilon$; $St \leftarrow (St, R)$; $(Y, St) \leftarrow \widehat{P}(\varepsilon; St)$
$\textsc{Ch}_1 \xleftarrow{\$} \{0,1\}^{l(k)}$; $(\textsc{Rsp}_1, St_1) \leftarrow \widehat{P}(\textsc{Ch}_1; St)$; $d_1 \leftarrow \left(\textsc{Rsp}_1^e \equiv_N Y W_0^{\textsc{Ch}_1}\right)$
$\textsc{Ch}_2 \xleftarrow{\$} \{0,1\}^{l(k)}$; $(\textsc{Rsp}_2, St_2) \leftarrow \widehat{P}(\textsc{Ch}_2; St)$; $d_2 \leftarrow \left(\textsc{Rsp}_2^e \equiv_N Y W_0^{\textsc{Ch}_2}\right)$
If $(d_1 = 1$ AND $d_2 = 1$ AND $\textsc{Ch}_1 \neq \textsc{Ch}_2)$ then
  $z \leftarrow \textsc{Rsp}_1 \cdot \textsc{Rsp}_2^{-1} \bmod N$; $(\overline{d}, a, b) \leftarrow \mathrm{EGCD}(e, \textsc{Ch}_1 - \textsc{Ch}_2)$
  $w_0 \leftarrow W_0^a z^b \bmod N$; For $i = 1$ to n do $w_i \leftarrow z_i w_0^{-c_i} \bmod N$
  Return $w_0, w_1, \ldots, w_n$
else Return $\perp$ EndIf

Figure 3.5 Rsa-omi-adversary I for the proof of Theorem 3.4.2. EGCD is a routine that implements the extended Euclid algorithm which given x, y returns (d, a, b) such that $d = \gcd(x, y)$ and $ax + by = d$.

If the decision predicate evaluates to 1, meaning $\widehat{P}$ makes the verifier accept, on both accounts and the challenges are different, then I extracts the inverse of W_0 as follows. It computes the quotient mod N of the cheating prover's responses to the challenges and sets z to this value. We observe that $z^e \equiv_N W_0^{\textsc{Ch}_1 - \textsc{Ch}_2}$. Then I uses the routine EGCD, which implements the extended Euclid algorithm, to compute $(\overline{d}, a, b)$, where $\overline{d} = \gcd(e, \textsc{Ch}_1 - \textsc{Ch}_2)$ and $a, b \in \mathbb{Z}$ are such that $ae + b(\textsc{Ch}_1 - \textsc{Ch}_2) = \overline{d}$. By the assumptions that e is prime and $2^{l(k)} < e$ (which implies $\textsc{Ch}_1, \textsc{Ch}_2 \in \mathbb{Z}_e$), $\overline{d} = 1$. Hence $ae + b(\textsc{Ch}_1 - \textsc{Ch}_2) = 1$. Therefore, we have

$$W_0 \equiv_N W_0^{ae} W_0^{b(\textsc{Ch}_1 - \textsc{Ch}_2)} \equiv_N W_0^{ae} (W_0^{\textsc{Ch}_1 - \textsc{Ch}_2})^b \equiv_N W_0^{ae} (z^e)^b \equiv_N (W_0^a z^b)^e \,.$$

This shows that $w_0 = W_0^a z^b \bmod N$ is the RSA-inverse of W_0. For $i = 1, \ldots, n(k)$, I computes the inverse of the i-th challenge point as $w_i = z_i w_0^{-c_i} \bmod N$. To prove that this computation yields the desired RSA-inverse, we show that $w_i^e \equiv_N W_i$. Since z_i is the inverse of $W_i W_0^{c_i}$ and w_0 is the inverse of W_0,

$$w_i^e \equiv_N (z_i w_0^{-c_i})^e \equiv_N z_i^e (w_0^e)^{-c_i} \equiv_N W_i W_0^{c_i} W_0^{-c_i} \equiv_N W_i \,.$$

If the decision predicate does not evaluate to 1 on both occasions or the challenges coincide, then I fails. Therefore, I wins if and only if $d_1 = 1$, $d_2 = 1$ and $\textsc{Ch}_1 \neq \textsc{Ch}_2$. We proceed to relate the probability of this event with the imp-ca-advantage of adversary A.

We observe that pk has the same distribution as in the two-phase game that defines a concurrent attack. Since I simulates the environment provided to $\widehat{V}$ in that game perfectly, $\widehat{V}$ behaves as it does when performing a concurrent attack against $\mathcal{ID}$, and $\widehat{P}$ is given state information with the same distribution as in that case. Therefore, the probability that $d_1 = 1$ is exactly $\mathbf{Adv}^{\text{imp-ca}}_{\mathcal{ID},A}(k)$.

To relate this probability with the probability that I wins, we will apply the Reset Lemma to the deterministic cheating prover $\widehat{P}$, verifier V, implemented by I, and the input generator IG defined below, which returns a pair (St, pk) consisting of the output St of $\widehat{V}$ on input pk and the public key pk.

Algorithm $\mathsf{IG}(k)$

$(N, e, d) \xleftarrow{\$} \mathcal{K}_{\text{rsa}}(k)$

$W_0 \xleftarrow{\$} \mathbb{Z}_N^* \,;\; pk \leftarrow (N, e, W_0)$

Choose a random tape R for $\widehat{V}$; Initialize $\widehat{V}$ with (pk, R) ; $n \leftarrow 0$

Run $\widehat{V}$ answering its requests as follows:

When $\widehat{V}$ issues a request of the form (ε, i) do

$n \leftarrow n + 1 \,;\; W_i \xleftarrow{\$} \mathbb{Z}_N^* \,;$ return W_i to $\widehat{V}$

When $\widehat{V}$ issues a request of the form (c, i), where $c \in \{0,1\}^{l(k)}$, do

$c_i \leftarrow c \,;\; z_i \leftarrow (W_i W_0^{c_i})^d \bmod N \,;$ return z_i to $\widehat{V}$

Until $\widehat{V}$ outputs state information St and stops

Return (St, pk)

We observe that the environment provided to $\widehat{V}$ by IG perfectly simulates the one provided by I in $\mathbf{Exp}^{\text{rsa-omi}}_{\mathcal{K}_{\text{rsa}},I}(k)$. Let acc, res, and c be defined as in Lemma 3.3.1. Note that $c(k) = 2^{l(k)}$. Comparing the definitions of I and IG, it is easy to see that $\mathsf{acc}(k) = \mathbf{Adv}^{\text{imp-ca}}_{\mathcal{ID},A}(k)$ and $\mathsf{res}(k) = \mathbf{Adv}^{\text{rsa-omi}}_{\mathcal{K}_{\text{rsa}},I}(k)$. Applying the Reset Lemma, we have

$$\mathbf{Adv}^{\text{imp-ca}}_{\mathcal{ID},A}(k) \;\leq\; 2^{-l(k)} + \sqrt{\mathbf{Adv}^{\text{rsa-omi}}_{\mathcal{K}_{\text{rsa}},I}(k)}\ .$$

To complete the proof of Theorem 3.4.2, it remains to justify the claim about the time complexity of adversary I. Consider the experiment that defines the rsa-omi-advantage of I. Our conventions for measuring time complexity imply that the cost of all the steps of this experiment before the execution of the final "If" in the algorithm of adversary I is at most $2t(k)$ plus the cost of evaluating the verifier's decision predicate twice. The latter involves computing two exponentiations of $|e|$-bit exponents and two exponentiations of $l(k)$-bit exponents. Since e is at most k bits long, this is $O(k^3 + l(k) \cdot k^2)$. We now calculate the cost of the remaining operations performed by I. The computation of the quotient mod N of the cheating prover's responses has cost $O(k^2)$. The extended Euclid algorithm runs in time the product of the lengths of its inputs. Hence the cost of computing $(\overline{d}, a, b)$ is $O(|e| \cdot |\text{CH}_1 - \text{CH}_2|)$, which is $O(k^2)$ because $\text{CH}_1, \text{CH}_2 \in \mathbb{Z}_e$ and $|e| \leq k$. The lengths of a and b cannot exceed the running time of EGCD, and they are exponents in the computation of w_0. Therefore, the cost of this computation is $O(k^2 \cdot k^2) = O(k^4)$. The "For" loop has cost $n(k) \cdot O(l(k) \cdot k^2)$. The time complexity of I is then $2t(k) + O(k^4 + (n(k) + 1) \cdot l(k) \cdot k^2)$. ∎

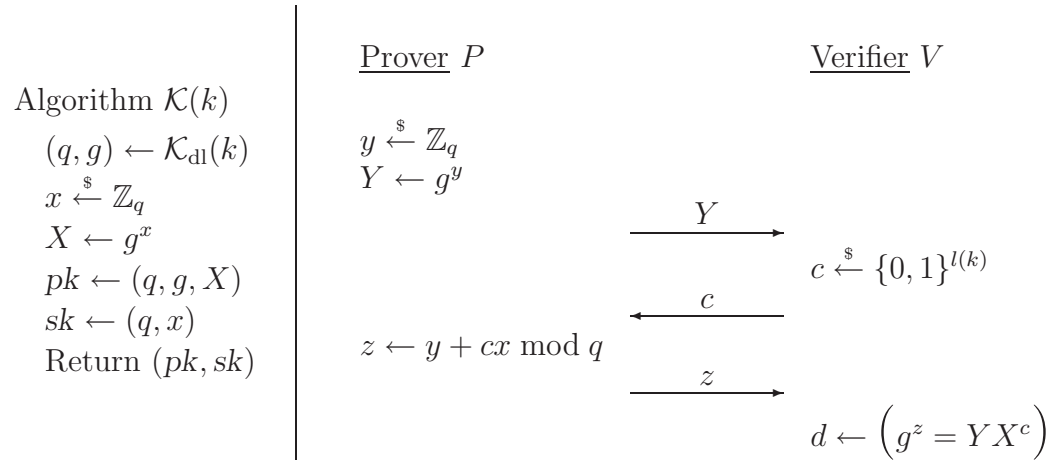

Figure 3.6 **Schnorr identification scheme**. Prover P has input $pk = (q, g, X)$ and $sk = (q, x)$. Verifier V has input pk.

3.5 Security of Schnorr under concurrent attack

A randomized, poly(k)-time algorithm $\mathcal{K}_{\mathrm{dl}}$ is said to be a *discrete-logarithm parameter generator* if given security parameter $k \in \mathbb{N}$, it outputs a pair (q, g) where q is a prime such that $q \,|\, p - 1$ for a prime p with $|p| = k$ (p is k bits long), and g is a generator of G_q, a subgroup of $\mathbb{Z}_p^*$ of order q. As before, we do not pin down any specific such generator. The generator is a parameter of the Schnorr scheme, and security is proved based on an assumption about it.

3.5.1 Schnorr identification scheme

Let $\mathcal{K}_{\mathrm{dl}}$ be a discrete-logarithm parameter generator and let $l \colon \mathbb{N} \to \mathbb{N}$ be a polynomial-time computable, polynomially bounded function such that $2^{l(k)} < q$ for any q output by $\mathcal{K}_{\mathrm{dl}}$ on input k. The *Schnorr identification scheme* associated to $\mathcal{K}_{\mathrm{dl}}$ and challenge length l is the ID scheme whose constituent algorithms are depicted in Figure 3.6. The prover's commitment is a random element $Y \in G_q$. For any verifier input $pk = (q, g, X)$, $\mathrm{ChSet}_{pk} = \{0,1\}^{l(k)}$. A challenge $c \in \mathrm{ChSet}_{pk}$ is interpreted as an integer in the set $\left\{0, \ldots, 2^{l(k)} - 1\right\}$ in the ensuing computations.

The assumption that $2^{l(k)} < q$ implies that the challenge is in $\mathbb{Z}_q$. The verifier's decision predicate $\mathrm{DEC}_{pk}(Y, c, z)$ evaluates to 1 if and only if z is the discrete logarithm of YX^c.

3.5.2 DL assumption

We recall the one-more-discrete-logarithm assumption [16], OMDL.

Assumption 3.5.1 [One-more-discrete-logarithm: OMDL] Let $\mathcal{K}_{\mathrm{dl}}$ be a discrete-logarithm parameter generator. Let I be a randomized, polynomial-time algorithm that takes input q, g and has access to two oracles. The first is a discrete-logarithm oracle $\mathrm{DLog}_{G_q,g}(\cdot)$ that given $Y \in G_q$ returns $y \in \mathbb{Z}_q$ such that $g^y = Y$. The second is a *challenge oracle* $\mathcal{O}_N$ that takes no inputs and returns a random challenge point $W \in G_q$ each time it is invoked. We call I an *omdl-adversary*. We associate to $\mathcal{K}_{\mathrm{dl}}$, I, and any $k \in \mathbb{N}$ the following experiment:

Experiment $\mathbf{Exp}^{\mathrm{omdl}}_{\mathcal{K}_{\mathrm{dl}},I}(k)$

$(q, g) \xleftarrow{\$} \mathcal{K}_{\mathrm{rsa}}(k)$; $w_1, \ldots, w_n \xleftarrow{\$} I^{\mathrm{DLog}_{G_q,g}(\cdot),\mathcal{O}_N}(q, g)$

Let $W_1, \ldots, W_n$ denote the challenges returned by $\mathcal{O}_N$ in response to queries from I, and m denote the number of queries made by I to its discrete-logarithm oracle

If ($g^{w_i} = W_i$ for $i = 1, \ldots, n$ AND $m < n$) then return 1 else return 0

We let

$$\mathbf{Adv}^{\mathrm{omdl}}_{\mathcal{K}_{\mathrm{dl}},I}(k) \;=\; \Pr\left[\,\mathbf{Exp}^{\mathrm{omdl}}_{\mathcal{K}_{\mathrm{dl}},I}(k) = 1\,\right]$$

denote the *omdl-advantage* of I, the probability being over the coins of $\mathcal{K}_{\mathrm{dl}}$, the coins of I, and the coins used by the challenge oracle across its invocations. We say that $\mathcal{K}_{\mathrm{dl}}$ is *OMDL-secure* if the function $\mathbf{Adv}^{\mathrm{omdl}}_{\mathcal{K}_{\mathrm{dl}},I}(\cdot)$ is negligible for any omdl-adversary I of time complexity polynomial in k. ▮

We adopt the same convention regarding time complexity as in the case of an rsa-omi-adversary.

3.5.3 Result

The following theorem guarantees that the advantage of any imp-ca-adversary attacking the Schnorr scheme can be upper bounded via the advantage of a related omdl-adversary and a function of the challenge length.

Theorem 3.5.2 Let $\mathcal{ID} = (\mathcal{K}, P, V)$ be the Schnorr identification scheme associated to discrete-logarithm parameter generator $\mathcal{K}_{\rm dl}$ and challenge length l. Let $A = (\widehat{V}, \widehat{P})$ be an imp-ca-adversary of time complexity $t(\cdot)$ attacking $\mathcal{ID}$. Then there exists an omdl-adversary I attacking $\mathcal{K}_{\rm dl}$ such that for every $k \in \mathbb{N}$:

$$\mathbf{Adv}^{\text{imp-ca}}_{\mathcal{ID},A}(k) \;\leq\; 2^{-l(k)} + \sqrt{\mathbf{Adv}^{\text{omdl}}_{\mathcal{K}_{\rm dl},I}(k)}\ . \tag{3.5}$$

Furthermore, the time complexity of I is $2t(k) + O(k^3 + (l(k) + n(k)) \cdot k^2)$, where $n(k)$ is the number of prover clones with which $\widehat{V}$ interacts. ▮

Before proving this theorem we note that it implies the following security result for the Schnorr scheme.

Corollary 3.5.3 If discrete-logarithm parameter generator $\mathcal{K}_{\rm dl}$ is OMDL-secure and challenge length l satisfies $l(k) = \omega(\log(k))$, then the Schnorr identification scheme associated to $\mathcal{K}_{\rm dl}$ and l is secure against impersonation under both active and concurrent attacks. ▮

As in the case of the GQ scheme, the assumption that the challenge length l is super-logarithmic in the security parameter is necessary since otherwise the Schnorr scheme can be broken by guessing the verifier's challenge. The proof of this corollary is completely analogous to the proof of Corollary 3.4.3.

We proceed to prove Theorem 3.5.2.

Proof of Theorem 3.5.2: The proof is similar to the proof of Theorem 3.4.2. We assume wlog that $\widehat{V}$ never repeats a request. Fix $k \in \mathbb{N}$ and let (q, g) be an output of $\mathcal{K}_{\rm dl}$ running on input k. Adversary I has access to a discrete-logarithm

Adversary $I^{\mathrm{DLog}_{G_q,g}(\cdot),\mathcal{O}_N}(q,g)$

Make a query to $\mathcal{O}_N$ and let W_0 be the response; $pk \leftarrow (q,g,W_0)$
Choose a random tape R for $\widehat{V}$; Initialize $\widehat{V}$ with (pk,R); $n \leftarrow 0$
Run $\widehat{V}$ answering its requests as follows:
  When $\widehat{V}$ issues a request of the form (ε,i) do
    $n \leftarrow n+1$; Make a query to $\mathcal{O}_N$, let W_i be the response and return W_i to $\widehat{V}$
  When $\widehat{V}$ issues a request of the form (c,i), where $c \in \{0,1\}^{l(k)}$, do
    $c_i \leftarrow c$; Make query $W_i W_0^{c_i}$ to $\mathrm{DLog}_{G_q,g}(\cdot)$, let z_i be the response and return z_i to $\widehat{V}$
Until $\widehat{V}$ outputs state information St and stops
$R \leftarrow \varepsilon$; $St \leftarrow (St,R)$; $(Y,St) \leftarrow \widehat{P}(\varepsilon;St)$
$\mathrm{CH}_1 \xleftarrow{\$} \{0,1\}^{l(k)}$; $(\mathrm{RSP}_1,St_1) \leftarrow \widehat{P}(\mathrm{CH}_1;St)$; $d_1 \leftarrow \left(g^{\mathrm{RSP}_1} = Y W_0^{\mathrm{CH}_1}\right)$
$\mathrm{CH}_2 \xleftarrow{\$} \{0,1\}^{l(k)}$; $(\mathrm{RSP}_2,St_2) \leftarrow \widehat{P}(\mathrm{CH}_2;St)$; $d_2 \leftarrow \left(g^{\mathrm{RSP}_2} = Y W_0^{\mathrm{CH}_2}\right)$
If $(d_1 = 1$ AND $d_2 = 1$ AND $\mathrm{CH}_1 \neq \mathrm{CH}_2)$ then
  $w_0 \leftarrow (\mathrm{RSP}_1 - \mathrm{RSP}_2)(\mathrm{CH}_1 - \mathrm{CH}_2)^{-1} \bmod q$
  For $i = 1$ to n do $w_i \leftarrow z_i - c_i w_0 \bmod q$
  Return $w_0, w_1, \ldots, w_n$
else Return $\perp$ EndIf

Figure 3.7 Omdl-adversary I for the proof of Theorem 3.5.2.

---

oracle $\mathrm{DLog}_{G_q,g}(\cdot)$ and a challenge oracle $\mathcal{O}_N$ that takes no inputs and returns a random challenge point $W \in G_q$ each time it is invoked. The adversary attempts to invert all the challenges returned by $\mathcal{O}_N$, while making fewer queries to its discrete-logarithm oracle then the number of challenge points.

A detailed description of the adversary is in Figure 3.7. I simulates an interaction between $\widehat{V}$ and the prover clones. To do so, it first queries its challenge oracle obtaining a random group element $W_0 \in G_q$ and uses it to create a public key pk for the imp-ca-adversary A. It then runs $\widehat{V}$ and answers its requests. In response to a request of the form (ε,i), I queries its challenge oracle $\mathcal{O}_N$ and returns the answer to $\widehat{V}$. By the definition of prover P, from $\widehat{V}$'s perspective, this is equivalent to picking a random tape R_i for prover clone i, initializing clone i with

state pk, R_i, computing clone i's commitment W_i, and returning the commitment to $\widehat{V}$. I is not in possession of the secret key $sk = (q, \mathrm{DLog}_{G_q,g}(W_0))$ corresponding to pk, which the prover clones would use to respond to $\widehat{V}$'s requests of the form (c, i), where $c \in \{0,1\}^{l(k)}$, but it compensates using its access to the discrete-logarithm oracle to answer these requests. Specifically, in response to request (c, i), I makes the query $W_i W_0^c$ to its discrete-logarithm oracle and returns the answer z_i to $\widehat{V}$. This is exactly the response that clone i would return to $\widehat{V}$ because $z_i = \mathrm{DLog}_{G_q,g}(W_i W_0^c) = \mathrm{DLog}_{G_q,g}(W_i) + c\,\mathrm{DLog}_{G_q,g}(W_0) \bmod q$. Hence I simulates the behavior of the prover clones perfectly.

Since $n(k)$ is the number of prover clones $\widehat{V}$ interacts with, when $\widehat{V}$ stops, I has made $n(k)$ queries to its discrete-logarithm oracle and it needs to find the discrete logarithm of $n(k)+1$ challenge points. I attempts to extract from $\widehat{P}$, initialized with the output of $\widehat{V}$, the discrete logarithm of challenge W_0. It can then use this value to compute the discrete logarithm of each of the other challenge points. To do so, I runs $\widehat{P}$ obtaining its commitment, selects a challenge uniformly at random from $\{0,1\}^{l(k)}$, runs $\widehat{P}$ to obtain its response to this challenge, and evaluates the verifier's decision predicate. It then selects another random challenge, re-runs $\widehat{P}$ (with the same state as before) to obtain its response to the new challenge, and evaluates the verifier's decision predicate. If the decision predicate evaluates to 1, meaning $\widehat{P}$ makes the verifier accept, on both accounts and the challenges are different, then I extracts the discrete logarithm of W_0 as $w_0 = (\mathrm{RSP}_1 - \mathrm{RSP}_2)(\mathrm{CH}_1 - \mathrm{CH}_2)^{-1} \bmod q$. We observe that since $\mathrm{CH}_1 \neq \mathrm{CH}_2$ and q is prime, $\mathrm{CH}_1 - \mathrm{CH}_2$ has a multiplicative inverse in $\mathbb{Z}_q$. To prove that the computation yields the desired value, we show that $g^{w_0} = W_0$. Since RSP_1 is the discrete logarithm of $YW_0^{\mathrm{CH}_1}$ and RSP_2 is the discrete logarithm of $YW_0^{\mathrm{CH}_2}$, we have

$$\begin{aligned} g^{w_0} &= g^{(\mathrm{RSP}_1 - \mathrm{RSP}_2)(\mathrm{CH}_1 - \mathrm{CH}_2)^{-1} \bmod q} \\ &= \left(g^{\mathrm{RSP}_1} \left(g^{\mathrm{RSP}_2}\right)^{-1}\right)^{(\mathrm{CH}_1 - \mathrm{CH}_2)^{-1} \bmod q} \\ &= \left(YW_0^{\mathrm{CH}_1} \left(YW_0^{\mathrm{CH}_2}\right)^{-1}\right)^{(\mathrm{CH}_1 - \mathrm{CH}_2)^{-1} \bmod q} \end{aligned}$$

$$
\begin{aligned}
&= \left(W_0^{\mathrm{CH}_1 - \mathrm{CH}_2}\right)^{(\mathrm{CH}_1 - \mathrm{CH}_2)^{-1} \bmod q} \\
&= W_0 \,.
\end{aligned}
$$

For $i = 1, \ldots, n(k)$, I computes the discrete logarithm of the i-th challenge point as $w_i = z_i - c_i w_0 \bmod q$. To prove that this computation is correct, we show that $g^{w_i} = W_i$. Since z_i is the discrete logarithm of $W_i W_0^{c_i}$ and w_0 is the discrete logarithm of W_0, we have

$$
g^{w_i} \;=\; g^{z_i - c_i w_0 \bmod q} \;=\; g^{z_i} \left(g^{w_0}\right)^{-c_i} \;=\; W_i W_0^{c_i} W_0^{-c_i} \;=\; W_i \,.
$$

If the decision predicate does not evaluate to 1 on both occasions or the challenges coincide, then I fails. Therefore, I wins if and only if $d_1 = 1$, $d_2 = 1$ and $\mathrm{CH}_1 \neq \mathrm{CH}_2$. We proceed to relate the probability of this event with the imp-ca-advantage of adversary A.

We observe that pk has the same distribution as in the two-phase game that defines a concurrent attack. Since I simulates the environment provided to $\widehat{V}$ in that game perfectly, $\widehat{V}$ behaves as it does when performing a concurrent attack against $\mathcal{ID}$, and $\widehat{P}$ is given state information with the same distribution as in that case. Therefore, the probability that $d_1 = 1$ is exactly $\mathbf{Adv}^{\text{imp-ca}}_{\mathcal{ID}, A}(k)$.

To relate this probability with the probability that I wins, we will apply the Reset Lemma to the deterministic cheating prover $\widehat{P}$, verifier V, implemented by I, and the input generator IG defined below, which returns a pair (St, pk) consisting of the output St of $\widehat{V}$ on input pk and the public key pk.

Algorithm $\mathsf{IG}(k)$

$(q, g) \xleftarrow{\$} \mathcal{K}_{\mathrm{dl}}(k)$

$w_0 \xleftarrow{\$} \mathbb{Z}_q$; $W_0 \leftarrow g^{w_0}$; $pk \leftarrow (q, g, W_0)$

Choose a random tape R for $\widehat{V}$; Initialize $\widehat{V}$ with (pk, R) ; $n \leftarrow 0$

Run $\widehat{V}$ answering its requests as follows:

When $\widehat{V}$ issues a request of the form (ε, i) do

$n \leftarrow n + 1$; $w_i \stackrel{\$}{\leftarrow} \mathbb{Z}_q$; $W_i \leftarrow g^{w_i}$; return W_i to $\widehat{V}$

When $\widehat{V}$ issues a request of the form (c, i), where $c \in \{0,1\}^{l(k)}$, do

$c_i \leftarrow c$; $z_i \leftarrow w_i + c_i w_0 \bmod q$; return z_i to $\widehat{V}$

Until $\widehat{V}$ outputs state information St and stops

Return (St, pk)

The environment provided to $\widehat{V}$ by IG perfectly simulates the one provided by I in $\mathbf{Exp}^{\text{omdl}}_{\mathcal{K}_{\text{dl}},I}(k)$. Let acc, res, and c be defined as in Lemma 3.3.1. Note that $c(k) = 2^{l(k)}$. Comparing the definitions of I and IG, it is easy to see that $\mathsf{acc}(k) = \mathbf{Adv}^{\text{imp-ca}}_{\mathcal{ID},A}(k)$ and $\mathsf{res}(k) = \mathbf{Adv}^{\text{omdl}}_{\mathcal{K}_{\text{dl}},I}(k)$. Applying the Reset Lemma, we have

$$\mathbf{Adv}^{\text{imp-ca}}_{\mathcal{ID},A}(k) \;\leq\; 2^{-l(k)} + \sqrt{\mathbf{Adv}^{\text{omdl}}_{\mathcal{K}_{\text{dl}},I}(k)} \;.$$

To complete the proof of Theorem 3.4.2, it remains to justify the claim about the time complexity of adversary I. Consider the experiment that defines the omdl-advantage of I. Our conventions for measuring time complexity imply that the cost of all the steps of this experiment before the execution of the final "If" in the algorithm of adversary I is at most $2t(k)$ plus the cost of evaluating the verifier's decision predicate twice. The latter involves computing two exponentiations of $|q|$-bit exponents and two exponentiations of $l(k)$-bit exponents. Since p is k bits long and q is at most k bits long, this is $O(k^3 + l(k) \cdot k^2)$. We now calculate the cost of the remaining operations performed by I. The computation of w_0 has cost $O(|q|^2) = O(k^2)$. The "For" loop has cost $n(k) \cdot O(k^2)$. The time complexity of I is then $2t(k) + O(k^3 + (l(k) + n(k)) \cdot k^2)$. ∎

This chapter, in part, is a reprint of the material as it appears in M. Bellare and A. Palacio, "GQ and Schnorr Identification Schemes: Proofs of Security against Impersonation under Active and Concurrent Attacks," *Advances in Cryptology - Crypto 2002 Proceedings*, Lecture Notes in Computer Science Vol. 2442, M. Yung ed., Springer-Verlag, 2002.

4 Knowledge-of-Exponent Assumptions and 3-round Zero-Knowledge Protocols

4.1 Introduction

A classical question in the theory of zero knowledge (ZK) [56] is whether there exist 3-round, negligible-error ZK proofs or arguments for $\mathbb{NP}$. The difficulty in answering this question stems from the fact that such protocols would have to be non-black-box simulation ZK [52], and there are few approaches or techniques to this end. A positive answer has, however, been provided by Hada and Tanaka [59, 60]. Their result (a negligible-error, 3-round ZK argument for $\mathbb{NP}$) requires a pair of non-standard assumptions that we will denote by KEA1 and KEA2.

4.1.1 The assumptions, roughly

Let q be a prime such that $2q + 1$ is also prime, and let g be a generator of the order q subgroup of $\mathbb{Z}^*_{2q+1}$. Suppose we are given inputs q, g, g^a and want to output a pair (C, Y) such that $Y = C^a$. One way to do this is to pick some $c \in \mathbb{Z}_q$, let $C = g^c$, and let $Y = (g^a)^c$. Intuitively, KEA1 can be viewed as saying that this is the "only" way to produce such a pair. The assumption captures this intuition

by saying that any adversary outputting such a pair must "know" an exponent c such that $g^c = C$. The formalization asks that there be an "extractor" that can return c. Roughly:

KEA1: For any adversary $\mathbf{A}$ that takes inputs q, g, g^a and returns (C, Y) with $Y = C^a$, there exists an "extractor" $\bar{\mathbf{A}}$, which given the same inputs as $\mathbf{A}$ returns c such that $g^c = C$.

Suppose we are given inputs q, g, g^a, g^b, g^{ab} and want to output a pair (C, Y) such that $Y = C^b$. One way to do this is to pick some $c \in \mathbb{Z}_q$, let $C = g^c$, and let $Y = (g^b)^c$. Another way is to pick some $c \in \mathbb{Z}_q$, let $C = (g^a)^c$, and let $Y = (g^{ab})^c$. Intuitively, KEA2 can be viewed as saying that these are the "only" ways to produce such a pair. The assumption captures this intuition by saying that any adversary outputting such a pair must "know" an exponent c such that either $g^c = C$ or $(g^a)^c = C$. The formalization asks that there be an "extractor" that can return c. Roughly:

KEA2: For any adversary $\mathbf{A}$ that takes inputs q, g, g^a, g^b, g^{ab} and returns (C, Y) with $Y = C^b$, there exists an "extractor" $\bar{\mathbf{A}}$, which given the same inputs as $\mathbf{A}$ returns c such that either $g^c = C$ or $(g^a)^c = C$.

As per [59, 60], adversaries and extractors are poly-size families of (deterministic) circuits. See Assumption 4.3.1 for a formalization of KEA2, and Assumption 4.4.2 for a formalization of KEA1.

4.1.2 History and nomenclature of the assumptions

KEA1 is due to Damgård [36]. Variants of this assumption are used by Hada and Tanaka [59, 60] to prove that their protocol satisfies various notions of ZK. To prove soundness of their protocol, they introduce and use KEA2. (In addition, they make the Discrete-Logarithm Assumption, DLA.) The preliminary version of their work [59] referred to the assumptions as SDHA1 and SDHA2 (**S**trong **D**iffie-**H**ellman **A**ssumptions 1 and 2), respectively. The full version [60], however, points

out that the formalizations in the preliminary version are flawed, and provides corrected versions called non-uniform-DA1 and non-uniform-DA2. The latter are the assumptions considered in this chapter, but we use the terminology of Naor [68] which we feel is more reflective of the content of the assumption: "KEA" stands for "**K**nowledge-of-**E**xponent **A**ssumption", the exponent being the value c above.

4.1.3 Falsifying KEA2

In this chapter we show that KEA2 is false. What is interesting about this —besides the fact that it renders the results of [59, 60] vacuous— is that we are able to "falsify" an assumption whose nature, as pointed out by Naor [68], does not lend itself easily to "efficient falsification." Let us explain this issue before expanding more on the result itself.

The most standard format for an assumption is to ask that the probability that an adversary produces a certain output on certain inputs is negligible. For example, the Factoring assumption is of this type, asking that the probability that a polynomial-time adversary can output the prime factors of an integer (chosen by multipling a pair of random primes) is negligible. To show that such an assumption is false, one can present an "attack," in the form of an adversary whose success probability is not negligible. (For example, a polynomial-time factoring algorithm.) KEA1 and KEA2 are not of this standard format. They involve a more complex quantification: "For every adversary there exists an extractor such that ...". To show that KEA2 is false, we must show that there is an adversary *for which there exists no extractor*. As we will see later, it is relatively simple to identify an adversary for which there does not *appear* to exist an extractor, but how can we actually show that none of the infinite number of possible extractors succeeds?

4.1.4 An analogy

The difficulty of falsifying an assumption with the quantifier format of KEA2 may be better appreciated via an analogy. The definition of ZK has a similar quantifier format: "For every (cheating) verifier there exists a simulator such that ...". This makes it hard to show that a protocol is not ZK, for, even though we may be able to identify a cheating-verifier strategy that appears hard to simulate, it is not clear how we can actually show that no simulator exists. (For example, it is hard to imagine how one could find a simulator for the cheating verifier, for Blum's ZK proof of Hamiltonian Cycle [23], that produces its challenges by hashing the permuted graphs sent by the prover in the first step. But there is to date no proof that such a simulator does not exist). It has been possible, however, to show that protocols are not black-box simulation ZK [52], taking advantage of the fact that the quantification in this definition is different from that of ZK itself. It has also been possible to show conditional results, for example that the parallel version of the Fiat-Shamir [45] protocol is not ZK, unless there is no hash function that, when applied to collapse this protocol, results in a secure signature scheme [77]. Our result too is conditional.

4.1.5 Falsification result

At an intuitive level, the weakness in KEA2 is easy to see, and indeed it is surprising that this was not noted before. Indeed, consider an adversary $\mathbf{A}$ that on inputs q, g, g^a, g^b, g^{ab} picks c_1, c_2 in some fashion, and outputs (C, Y) where $C = g^{c_1}(g^a)^{c_2}$ and $Y = (g^b)^{c_1}(g^{ab})^{c_2}$. Then $Y = C^b$ but this adversary does not appear to "know" c such that either $g^c = C$ or $(g^a)^c = C$. The difficulty, however, as indicated above, is to prove that there does not exist an extractor. We do this by first specifying a particular strategy for choosing c_1 and c_2 and then showing that if there exists an extractor for the resulting adversary, then this extractor can be used to solve the discrete-logarithm problem (DLP). Thus, our result (cf. The-

orem 4.3.2) is that if DLP is hard (equivalently, DLA holds) then KEA2 is false. Note that if DLP is easy, then KEA2 is true, for the extractor can simply compute a discrete logarithm of C and output it, and thus the assumption that it is hard is necessary to falsify KEA2.

4.1.6 Remark

We emphasize that we have not found any weaknesses in KEA1, an assumption used not only in [36, 59, 60] but also elsewhere.

4.1.7 KEA3

Providing a 3-round, negligible-error ZK protocol for $\mathbb{NP}$ is a challenging problem that has attracted considerable research effort. The fact that KEA2 is false means that we "lose" one of the few positive results [59, 60] that exist on this subject. Accordingly, we would like to "recover" it. To this end, we propose a modification of KEA2 that addresses the weakness we found. The new assumption is, roughly, as follows:

KEA3: For any adversary $\mathbf{A}$ that takes inputs q, g, g^a, g^b, g^{ab} and returns (C, Y) with $Y = C^b$, there exists an "extractor" $\bar{\mathbf{A}}$, which given the same inputs as $\mathbf{A}$ returns c_1, c_2 such that $g^{c_1}(g^a)^{c_2} = C$.

Before proceeding to use this assumption, we note a relation that we consider interesting, namely, that KEA3 implies KEA1 (cf. Proposition 4.4.3).[1] This relation means that KEA3 is a natural extension of KEA1.

[1] KEA2 was not shown by [60] to imply KEA1. Our proof of Proposition 4.4.3 can be adapted to establish this, but the point is moot since KEA2 is false (if DLP is hard) and hence, of course, implies everything anyway.

4.1.8 Recovering the ZK results

Let HTP denote the 3-round protocol of Hada and Tanaka, which they claim to be sound (i.e., have negligible error) and ZK. The falsity of KEA2 invalidates their proof of soundness. This does not mean, however, that HTP is not sound; perhaps it is and this could be proved under another assumption, such as KEA3. This turns out to be almost, but not quite, true. We identify a small bug in HTP based on which we can present a successful cheating-prover strategy, showing that HTP is not sound. This is easily fixed, however, to yield a protocol that we call PHTP (**P**atched HTP). The proof of soundness of HTP provided in [60] extends with very minor modifications to prove soundness of PHTP based on KEA3 and DLA (cf. Lemma 4.5.3). On the other hand, PHTP is close enough to HTP that the proofs of ZK (based on variants of KEA1) are unchanged.

To prove that HTP has ZK properties, Hada and Tanaka use variants of KEA1 that consider an adversary and an extractor who are given an auxiliary input. We formalize KEA1-A(p), where p is a polynomial restricting the length of the auxiliary input, in Assumption 4.5.7. The proponents of HTP show that this protocol is non-uniform ZK if KEA1-A(p) holds for a particular polynomial p, and it is auxiliary-input non-uniform ZK if KEA1-A(p) holds for every polynomial p. They also consider uniform variants of KEA1. We formalize UKEA1-A(p), where p is a polynomial restricting the length of an auxiliary input given to the adversary and the extractor, in Assumption 4.5.8. Hada and Tanaka show that HTP is ZK if UKEA1-A(p) holds for a particular polynomial p, and it is auxiliary-input ZK if UKEA1-A(p) holds for every polynomial p. PHTP inherits these ZK properties, under the same assumptions.

In summary, assuming KEA3 and DLA, there exists a 3-round, negligible-error argument for $\mathbb{NP}$ that is non-uniform ZK if KEA1-A(p) holds for a polynomial p specified in Section 4.5, auxiliary-input non-uniform ZK if KEA1-A(p) holds for every polynomial p, ZK if UKEA1-A(p) holds for a polynomial p specified in

Section 4.5, and auxiliary-input ZK if UKEA1-A(p) holds for every polynomial p.

4.1.9 Strength of the assumptions

The knowledge-of-exponent assumptions are strong and non-standard ones, and have been criticized for assuming that one can perform what some people call "reverse engineering" of an adversary. These critiques are certainly valid. Our falsification of KEA2 does not provide information on this aspect of the assumptions, uncovering, rather, other kinds of problems. By showing that such assumptions can be falsified, however, we open the door to further analyses.

We also stress that in recovering the result of [60] on 3-round ZK we have not succeeded in weakening the assumptions on which it is based, for KEA3 certainly remains a strong assumption of the same non-standard nature as KEA1 and its variants.

4.1.10 Related work

Since [59, 60] there has been more progress with regard to the design of non-black-box simulation ZK protocols, most notably [3]. That work, however, does not provide a 3-round, negligible-error ZK protocol for $\mathbb{NP}$. To date, there have been only two positive results in this regard. One is that of [59, 60], broken and recovered in this dissertation. The other, which builds a proof system rather than an argument, is reported in [65] and further documented in [64]. It also relies on non-standard assumptions, but these are of a different nature than the Knowledge-of-Exponent ones. Roughly, the authors assume the existence of a hash function such that a certain discrete-logarithm-based protocol, that uses this hash function and is related to the non-interactive oblivious-transfer protocol of [14], is a proof of knowledge.

4.2 Preliminaries

If q is a prime number such that $2q+1$ is also prime, then we denote by G_q the subgroup of quadratic residues of $\mathbb{Z}^*_{2q+1}$. (Operations are modulo $2q+1$ but we will omit writing "mod $2q+1$" for simplicity.) Recall that this is a cyclic group of order q. If g is a generator of G_q then we let $\mathrm{DLog}_{q,g}\colon G_q \to \mathbb{Z}_q$ denote the associated discrete-logarithm function, meaning $\mathrm{DLog}_{q,g}(g^a) = a$ for any $a \in \mathbb{Z}_q$. We let

$$GL \;=\; \{\, (q,g) \;:\; q, 2q+1 \text{ are primes and } g \text{ is a generator of } G_q \,\}\,,$$

and for every $n \in \mathbb{N}$, we let

$$GL_n \;=\; \{\, (q,g) \in GL \;:\; |2q+1| = n \,\}\,.$$

Assumptions and problems in [59, 60] involve circuits. A family of circuits $\mathbf{C} = \{\mathbf{C}_n\}_{n\in\mathbb{N}}$ contains one circuit for each value of $n \in \mathbb{N}$. It is *poly-size* if there is a polynomial p such that the size of $\mathbf{C}_n$ is at most $p(n)$ for all $n \in \mathbb{N}$. Unless otherwise stated, *circuits are deterministic*. If they are randomized, we will say so explicitly. We now recall DLA, following [60].

Assumption 4.2.1 [DLA] Let $\mathbf{I} = \{\mathbf{I}_n\}_{n\in\mathbb{N}}$ be a family of randomized circuits, and $\nu\colon \mathbb{N} \to [0,1]$ a function. We associate to any $n \in \mathbb{N}$ and any $(q,g) \in GL_n$ the following experiment:

Experiment $\mathbf{Exp}^{\mathrm{dl}}_{\mathbf{I}}(n,q,g)$

$a \xleftarrow{\$} \mathbb{Z}_q\,;\; A \leftarrow g^a\,;\; \bar{a} \xleftarrow{\$} \mathbf{I}_n(q,g,A)\,;$ If $a = \bar{a}$ then return 1 else return 0

We let

$$\mathbf{Adv}^{\mathrm{dl}}_{\mathbf{I}}(n,q,g) \;=\; \Pr\left[\,\mathbf{Exp}^{\mathrm{dl}}_{\mathbf{I}}(n,q,g) = 1\,\right]$$

denote the *advantage* of adversary $\mathbf{I}$ on inputs n,q,g, the probability being over the random choice of a and the coins of $\mathbf{I}_n$, if any. We say that $\mathbf{I}$ has *success bound* ν if

$$\forall n \in \mathbb{N}\;\; \forall (q,g) \in GL_n :\;\; \mathbf{Adv}^{\mathrm{dl}}_{\mathbf{I}}(n,q,g) \le \nu(n)\,.$$

We say the *Discrete-Logarithm Assumption (DLA) holds* (i.e., the *Discrete-Logarithm Problem (DLP) is hard*) if for every poly-size family of (deterministic) circuits $\mathbf{I}$ there exists a negligible function ν such that $\mathbf{I}$ has success bound ν. ▮

The above formulation of DLA, which, as we have indicated, follows [60], has some non-standard features that are important for their results. Let us discuss these briefly.

First, we note that the definition of the success bound is not with respect to (q, g) being chosen according to some distribution as is standard, but rather makes the stronger requirement that the advantage of $\mathbf{I}$ is small for all (q, g).

Second, we stress that the assumption only requires poly-size families of *deterministic* circuits to have a negligible success bound. However, in their proofs, which aim to contradict DLA, Hada and Tanaka [59, 60] build adversaries that are poly-size families of randomized circuits, and then argue that these can be converted to related poly-size families of deterministic circuits that do not have a negligible success bound. We will also need to build such randomized adversaries, but, rather than using ad hoc conversion arguments repeated across proofs, we note the following more general proposition, which simply says that DLA, as per Assumption 4.2.1, implies that poly-size families of randomized circuits also have a negligible success bound. We will appeal to this in several later places in this chapter.

Proposition 4.2.2 Assume DLA, and let $\mathbf{J} = \{\mathbf{J}_n\}_{n\in\mathbb{N}}$ be a poly-size family of *randomized* circuits. Then there exists a negligible function ν such that $\mathbf{J}$ has success bound ν. ▮

As is typical in such claims, the proof proceeds by showing that for every n there exists a "good" choice of coins for $\mathbf{J}_n$, and by embedding these coins we get a deterministic circuit. For completeness, we include the proof below.

Proof of Proposition 4.2.2: Let $K = \{\, n \in \mathbb{N} \,:\, GL_n \neq \emptyset \,\}$. For each $n \in K$,

let $(q_n, g_n) \in GL_n$ be such that

$$\forall (q,g) \in GL_n : \ \mathbf{Adv}^{\mathrm{dl}}_{\mathbf{J}}(n,q,g) \ \leq \ \mathbf{Adv}^{\mathrm{dl}}_{\mathbf{J}}(n,q_n,g_n) \ . \tag{4.1}$$

For $n \in K$, let $R(n)$ denote the set from which $\mathbf{J}_n$ draws its coins on inputs n, q_n, g_n. We say that $r \in R(n)$ is *n-good* if

$$\Pr\left[\ g^{\bar{a}} = A \ \ : \ \ A \xleftarrow{\$} G_{q_n} \ ; \ \bar{a} \leftarrow \mathbf{J}_n(q_n, g_n, A; r) \ \right] \ \geq \ \mathbf{Adv}^{\mathrm{dl}}_{\mathbf{J}}(n, q_n, g_n) \ .$$

Claim 4.2.3 For each $n \in K$ there exists $r \in R(n)$ such that r is n-good. ∎

Proof of Claim 4.2.3: Define $X : \ G_{q_n} \times \mathbb{Z}_{q_n} \to \{0,1\}$ as follows:

$X(A, r)$
  $\bar{a} \leftarrow \mathbf{J}_n(q_n, g_n, A; r)$
  If $g^{\bar{a}} = A$ then return 1 else return 0

Then we have:

$$\begin{aligned}
&\sum_{r \in R(n)} \frac{1}{|R(n)|} \cdot \Pr\left[\ g^{\bar{a}} = A \ \ : \ \ A \xleftarrow{\$} G_{q_n} \ ; \ \bar{a} \leftarrow \mathbf{J}_n(q_n, g_n, A; r) \ \right] \\
&\qquad = \ \sum_{r \in R(n)} \frac{1}{|R(n)|} \sum_{A \in G_{q_n}} \frac{1}{q_n} \cdot X(A, r) \\
&\qquad = \ \sum_{A \in G_{q_n}} \frac{1}{q_n} \sum_{r \in R(n)} \frac{1}{|R(n)|} \cdot X(A, r) \\
&\qquad = \ \mathbf{Adv}^{\mathrm{dl}}_{\mathbf{J}_n}(n, q_n, g_n) \ .
\end{aligned}$$

This means that there must exist $r \in R(n)$ such that

$$\Pr\left[\ g^{\bar{a}} = A \ \ : \ \ A \xleftarrow{\$} G_{q_n} \ ; \ \bar{a} \leftarrow \mathbf{J}_n(q_n, g_n, A; r) \ \right] \ \geq \ \mathbf{Adv}^{\mathrm{dl}}_{\mathbf{J}_n}(n, q_n, g_n) \ ,$$

which proves the claim. ∎

We now define a poly-size family $\mathbf{I} = \{\mathbf{I}_n\}_{n \in \mathbb{N}}$ of (deterministic) circuits, as follows. Let $n \in \mathbb{N}$. If $n \notin K$ then we define $\mathbf{I}_n$ arbitrarily. If $n \in K$ then Claim 4.2.3 tells us that there exists a string, which we denote by r_n, that is n-good. We then define $\mathbf{I}_n$ as follows:

$\mathbf{I}_n(q, g, A)$

If $q \neq q_n$ or $g \neq g_n$ then abort EndIf

$\bar{a} \leftarrow \mathbf{J}_n(q_n, g_n, A; r_n)$

Return $\bar{a}$

Since $\mathbf{I}$ is a poly-size family of deterministic circuits, the assumption that DLP is hard says that there is a negligible function ν such that $\mathbf{I}$ has success bound ν. Now putting this together with Equation (4.1) and Claim 4.2.3 we have

$$\forall n \in K \ \forall (q, g) \in GL_n :$$
$$\mathbf{Adv}^{\mathrm{dl}}_{\mathbf{J}}(n, q, g) \ \leq \ \mathbf{Adv}^{\mathrm{dl}}_{\mathbf{J}}(n, q_n, g_n) \ \leq \ \mathbf{Adv}^{\mathrm{dl}}_{\mathbf{I}}(n, q_n, g_n) \ \leq \ \nu(n) \ .$$

This means that $\mathbf{J}$ also has success bound ν, which proves the proposition. ∎

4.3 KEA2 is false

We begin by recalling the assumption. Our presentation is slightly different from, but clearly equivalent to, that of [60]: we have merged the two separate conditions of their formalization into one. Recall that in [60], this assumption was referred to as "non-uniform-DA2," and it was referred to, under a different and incorrect formalization, as SDHA2 in [59].

Assumption 4.3.1 [KEA2] Let $\mathbf{A} = \{\mathbf{A}_n\}_{n\in\mathbb{N}}$ and $\bar{\mathbf{A}} = \{\bar{\mathbf{A}}_n\}_{n\in\mathbb{N}}$ be families of circuits, and $\nu : \mathbb{N} \to [0, 1]$ a function. We associate to any $n \in \mathbb{N}$, any $(q, g) \in GL_n$, and any $A \in G_q$ the following experiment:

Experiment $\mathbf{Exp}^{\mathrm{kea2}}_{\mathbf{A},\bar{\mathbf{A}}}(n, q, g, A)$

$b \xleftarrow{\$} \mathbb{Z}_q \,;\ B \leftarrow g^b \,;\ X \leftarrow A^b$

$(C, Y) \leftarrow \mathbf{A}_n(q, g, A, B, X) \,;\ c \leftarrow \bar{\mathbf{A}}_n(q, g, A, B, X)$

If $(Y = C^b$ AND $g^c \neq C$ AND $A^c \neq C)$ then return 1 else return 0

We let

$$\mathbf{Adv}^{\mathrm{kea2}}_{\mathbf{A},\bar{\mathbf{A}}}(n,q,g,A) \;=\; \Pr\left[\,\mathbf{Exp}^{\mathrm{kea2}}_{\mathbf{A},\bar{\mathbf{A}}}(n,q,g,A)=1\,\right]$$

denote the *advantage* of adversary $\mathbf{A}$ relative to $\bar{\mathbf{A}}$ on inputs n, q, g, A. We say that $\bar{\mathbf{A}}$ is a *kea2-extractor for* $\mathbf{A}$ *with error bound* ν if

$$\forall n \in \mathbb{N}\ \forall (q,g) \in GL_n\ \forall A \in G_q :\ \mathbf{Adv}^{\mathrm{kea2}}_{\mathbf{A},\bar{\mathbf{A}}}(n,q,g,A) \;\leq\; \nu(n)\ .$$

We say that *KEA2 holds* if for every poly-size family of circuits $\mathbf{A}$ there exist a poly-size family of circuits $\bar{\mathbf{A}}$ and a negligible function ν such that $\bar{\mathbf{A}}$ is a kea2-extractor for $\mathbf{A}$ with error bound ν. ▮

We stress again that in the above formulation, following [60], both the adversary and the extractor are families of *deterministic* circuits. One can consider various variants of the assumption, including an extension to families of randomized circuits, and we discuss these following the theorem below.

Theorem 4.3.2 If DLA holds then KEA2 is false. ▮

The basic idea behind the failure of the assumption, as sketched in Section 4.1, is simple. Consider an adversary given inputs q, g, A, B, X, where $A = g^a, B = g^b$ and $X = g^{ab}$. The assumption says that there are only two ways for the adversary to output a pair C, Y satisfying $Y = C^b$. One way is to pick some c, let $C = g^c$ and let $Y = B^c$. The other way is to pick some c, let $C = A^c$ and let $Y = X^c$. The assumption thus states that the adversary "knows" c such that either $C = g^c$ (i.e., $c = \mathrm{DLog}_{q,g}(C)$) or $C = A^c$ (i.e., $c = \mathrm{DLog}_{q,A}(C)$). This ignores the possibility of performing a linear combination of the two steps above. In other words, an adversary might pick c_1, c_2, let $C = g^{c_1}A^{c_2}$ and $Y = B^{c_1}X^{c_2}$. In this case, $Y = C^b$ but the adversary does not appear to necessarily know $\mathrm{DLog}_{q,g}(C) = c_1 + c_2\mathrm{DLog}_{q,g}(A)$ or $\mathrm{DLog}_{q,A}(C) = c_1\mathrm{DLog}_{q,A}(g) + c_2$.

Going from this intuition to an actual proof that the assumption is false, however, takes some work, for several reasons. The above may be intuition that

there exists an adversary for which there would not exist an extractor, but we need to *prove* that there is no extractor. This cannot be done unconditionally, since certainly if DLP is easy, then in fact there is an extractor: it simply computes $\mathrm{DLog}_{q,g}(C)$ and returns this value. Accordingly, our strategy will be to present an adversary $\mathbf{A}$ for which we can prove that if there exists a kea2-extractor $\bar{\mathbf{A}}$ then there is a method to efficiently compute the discrete logarithm of A.

An issue in implementing this is that the natural adversary $\mathbf{A}$ arising from the above intuition is randomized, picking c_1, c_2 at random and forming C, Y as indicated, but our adversaries must be deterministic. We resolve this by designing an adversary that makes certain specific choices of c_1, c_2. We now proceed to the formal proof.

4.3.1 Proof of Theorem 4.3.2

Assume to the contrary that KEA2 is true. We show that DLP is easy. The outline of the proof is as follows. We first construct an adversary $\mathbf{A}$ for the KEA2 problem. By assumption, there exists for it a kea2-extractor $\bar{\mathbf{A}}$ with negligible error bound. Using $\bar{\mathbf{A}}$, we then present a poly-size family of randomized circuits $\mathbf{J} = \{\mathbf{J}_n\}_{n\in\mathbb{N}}$ for DLP, and show that it does not have a negligible success bound. By Proposition 4.2.2, this contradicts DLA.

The poly-size family of circuits $\mathbf{A} = \{\mathbf{A}_n\}_{n\in\mathbb{N}}$ is presented in Figure 4.1. Now, under KEA2, there exist a poly-size family of circuits $\bar{\mathbf{A}} = \{\bar{\mathbf{A}}_n\}_{n\in\mathbb{N}}$ and a negligible function ν such that $\bar{\mathbf{A}}$ is a kea2-extractor for $\mathbf{A}$ with error bound ν. Using $\bar{\mathbf{A}}$, we define the poly-size family of circuits $\mathbf{J} = \{\mathbf{J}_n\}_{n\in\mathbb{N}}$ shown in Figure 4.1.

Claim 4.3.3 For all $n \in \mathbb{N}$, all $(q, g) \in GL_n$ and all $A \in G_q$:

$$\Pr\left[\, g^{\bar{a}} \neq A \;\; : \;\; \bar{a} \xleftarrow{\$} \mathbf{J}_n(q, g, A) \,\right] \;\leq\; \nu(n)\ . \quad \blacksquare$$

Note that this claim says much more than what we need. Indeed, $\mathbf{J}$ does

$\mathbf{A}_n(q, g, A, B, X)$	$\mathbf{J}_n(q, g, A)$
$C \leftarrow gA$	$b \xleftarrow{\$} \mathbb{Z}_q$; $B \leftarrow g^b$; $X \leftarrow A^b$
$Y \leftarrow BX$	$c \leftarrow \bar{\mathbf{A}}_n(q, g, A, B, X)$
Return (C, Y)	$C \leftarrow gA$
	If $g^c = C$ then $\bar{a} \leftarrow (c-1) \bmod q$ EndIf
	If $A^c = C$ then $\bar{a} \leftarrow (c-1)^{-1} \bmod q$ EndIf
	Return $\bar{a}$

Figure 4.1 Adversary $\mathbf{A} = \{\mathbf{A}_n\}_{n\in\mathbb{N}}$ for KEA2 and adversary $\mathbf{J} = \{\mathbf{J}_n\}_{n\in\mathbb{N}}$ for DLP, for the proof of Theorem 4.3.2.

---

not merely have a success bound that is not negligible. In fact, it succeeds with probability almost one.

Proof of Claim 4.3.3: We let $\Pr[\cdot]$ denote the probability in the experiment of executing $\mathbf{J}_n(q, g, A)$. We first write some inequalities leading to the claim and then justify them.

$$\begin{aligned} \Pr[\, g^{\bar{a}} \neq A \,] &\leq \Pr[\, g^c \neq C \wedge A^c \neq C \,] && (4.2) \\ &\leq \mathbf{Adv}^{\mathrm{kea2}}_{\mathbf{A},\bar{\mathbf{A}}}(n, q, g, A) && (4.3) \\ &\leq \nu(n)\ . && (4.4) \end{aligned}$$

We justify Equation (4.2) by showing that if $g^c = C$ or $A^c = C$ then $g^{\bar{a}} = A$. First assume $g^c = C$. Since $C = gA$, we have $g^c = gA$, whence $A = g^{c-1}$. Since we set $\bar{a} = (c-1) \bmod q$, we have $A = g^{\bar{a}}$. Next assume $A^c = C$. Since $C = gA$, we have $A^c = gA$, whence $A^{c-1} = g$. Now observe that $c \neq 1$, because otherwise $A^c = A \neq gA$. (Since g is a generator, it is not equal to 1). Since $c \neq 1$ and q is prime, $c - 1$ has an inverse modulo q which we have denoted by $\bar{a}$. Raising both sides of the equation "$A^{c-1} = g$" to the power $\bar{a}$ we get $A = g^{\bar{a}}$.

$\mathbf{Exp}^{\mathrm{kea2}}_{\mathbf{A},\bar{\mathbf{A}}}(n, q, g, A)$ returns 1 exactly when $Y = C^b$ and $g^c \neq C$ and $A^c \neq C$. By construction of $\mathbf{A}$, we have $C = gA$ and $Y = BX$, and thus $Y = C^b$, so $\mathbf{Exp}^{\mathrm{kea2}}_{\mathbf{A},\bar{\mathbf{A}}}(n, q, g, A)$ returns 1 exactly when $g^c \neq C$ and $A^c \neq C$. This justifies Equation (4.3).

Equation (4.4) is justified by the assumption that $\bar{\mathbf{A}}$ is a kea2-extractor for $\mathbf{A}$ with error bound ν. ∎

Claim 4.3.3 implies that $\mathbf{J}$ does not have a negligible success bound, which, by Proposition 4.2.2, shows that DLP is not hard, contradicting the assumption made in this theorem. This completes the proof of Theorem 4.3.2.

4.3.2 Extensions and variants

There are many ways in which the formalization of Assumption 4.3.1 can be varied to capture the same basic intuition. Theorem 4.3.2, however, extends to these variants as well. Let us discuss this briefly.

As mentioned above, we might want to allow the adversary to be randomized. (In that case, it is important that the extractor get the coins of the adversary as an additional input, since otherwise the assumption is clearly false.) Theorem 4.3.2 remains true for the resulting assumption, in particular because it is stronger than the original assumption. (Note however that the proof of the theorem would be easier for this stronger assumption.)

Another variant is that adversaries and extractors are uniform, namely standard algorithms, not circuits. (In this case we should certainly allow both to be randomized, and should again give the extractor the coins of the adversary.) Again, it is easy to see that Theorem 4.3.2 extends to show that this variant of the assumption is also false.

4.4 The KEA3 assumption

The obvious fix to KEA2 is to take into account the possibility of linear combinations by saying that this is the only thing the adversary can do. This leads to the following.

Assumption 4.4.1 [KEA3] Let $\mathbf{A} = \{\mathbf{A}_n\}_{n\in\mathbb{N}}$ and $\bar{\mathbf{A}} = \{\bar{\mathbf{A}}_n\}_{n\in\mathbb{N}}$ be families of circuits, and $\nu\colon \mathbb{N} \to [0,1]$ a function. We associate to any $n \in \mathbb{N}$, any $(q, g) \in GL_n$, and any $A \in G_q$ the following experiment:

Experiment $\mathbf{Exp}^{\mathrm{kea3}}_{\mathbf{A},\bar{\mathbf{A}}}(n, q, g, A)$
$b \xleftarrow{\$} \mathbb{Z}_q\,;\ B \leftarrow g^b\,;\ X \leftarrow A^b$
$(C, Y) \leftarrow \mathbf{A}_n(q, g, A, B, X)\,;\ (c_1, c_2) \leftarrow \bar{\mathbf{A}}_n(q, g, A, B, X)$
If $(Y = C^b$ AND $g^{c_1}A^{c_2} \neq C)$ then return 1 else return 0

We let

$$\mathbf{Adv}^{\mathrm{kea3}}_{\mathbf{A},\bar{\mathbf{A}}}(n, q, g, A) \;=\; \Pr\left[\,\mathbf{Exp}^{\mathrm{kea3}}_{\mathbf{A},\bar{\mathbf{A}}}(n, q, g, A) = 1\,\right]$$

denote the *advantage* of adversary $\mathbf{A}$ relative to $\bar{\mathbf{A}}$ on inputs n, q, g, A. We say that $\bar{\mathbf{A}}$ is a *kea3-extractor for* $\mathbf{A}$ *with error bound* ν if

$$\forall n \in \mathbb{N}\ \forall (q, g) \in GL_n\ \forall A \in G_q :\ \mathbf{Adv}^{\mathrm{kea3}}_{\mathbf{A},\bar{\mathbf{A}}}(n, q, g, A) \;\leq\; \nu(n)\ .$$

We say that *KEA3 holds* if for every poly-size family of circuits $\mathbf{A}$ there exist a poly-size family of circuits $\bar{\mathbf{A}}$ and a negligible function ν such that $\bar{\mathbf{A}}$ is a kea3-extractor for $\mathbf{A}$ with error bound ν. ∎

We have formulated this assumption in the style of the formalization of KEA2 of [60] given in Assumption 4.3.1. Naturally, variants such as discussed above are possible. Namely, we could strengthen the assumption to allow the adversary to be a family of randomized circuits, of course then giving the extractor the adversary's coins as an additional input. We do not do this because we do not need it for what follows. We could also formulate a uniform-complexity version of the assumption. We do not do this because it does not suffice to prove the results that follow. These extensions or variations, however, might be useful in other contexts.

We now recall KEA1 and show that KEA3 is a natural extension of this assumption. Our formalization follows [60], but we apply the same simplifications as we did for KEA2, merging their two conditions into one.

Assumption 4.4.2 [KEA1] Let $\mathbf{A} = \{\mathbf{A}_n\}_{n\in\mathbb{N}}$ and $\bar{\mathbf{A}} = \{\bar{\mathbf{A}}_n\}_{n\in\mathbb{N}}$ be families of circuits, and $\nu\colon \mathbb{N} \to [0,1]$ a function. We associate to any $n \in \mathbb{N}$ and any $(q,g) \in GL_n$ the following experiment:

Experiment $\mathbf{Exp}^{\mathrm{kea1}}_{\mathbf{A},\bar{\mathbf{A}}}(n,q,g)$
$b \xleftarrow{\$} \mathbb{Z}_q$; $B \leftarrow g^b$
$(C,Y) \leftarrow \mathbf{A}_n(q,g,B)$; $c \leftarrow \bar{\mathbf{A}}_n(q,g,B)$
If ($Y = C^b$ AND $g^c \neq C$) then return 1 else return 0

We let

$$\mathbf{Adv}^{\mathrm{kea1}}_{\mathbf{A},\bar{\mathbf{A}}}(n,q,g) \;=\; \Pr\left[\,\mathbf{Exp}^{\mathrm{kea1}}_{\mathbf{A},\bar{\mathbf{A}}}(n,q,g) = 1\,\right]$$

denote the *advantage* of adversary $\mathbf{A}$ relative to $\bar{\mathbf{A}}$ on inputs n,q,g. We say that $\bar{\mathbf{A}}$ is a *kea1-extractor for* $\mathbf{A}$ *with error bound* ν if

$$\forall n \in \mathbb{N}\ \forall (q,g) \in GL_n :\ \mathbf{Adv}^{\mathrm{kea1}}_{\mathbf{A},\bar{\mathbf{A}}}(n,q,g) \;\leq\; \nu(n)\ .$$

We say that *KEA1 holds* if for every poly-size family of circuits $\mathbf{A}$ there exist a poly-size family of circuits $\bar{\mathbf{A}}$ and a negligible function ν such that $\bar{\mathbf{A}}$ is a kea1-extractor for $\mathbf{A}$ with error bound ν. ▮

Proposition 4.4.3 KEA3 implies KEA1. ▮

Proof of Proposition 4.4.3: Let $\mathbf{A} = \{\mathbf{A}_n\}_{n\in\mathbb{N}}$ be a poly-size family of circuits for KEA1. We need to show that there exist a poly-size family of circuits $\bar{\mathbf{A}} = \{\bar{\mathbf{A}}_n\}_{n\in\mathbb{N}}$ and a negligible function ν such that $\bar{\mathbf{A}}$ is a kea1-extractor for $\mathbf{A}$ with error-bound ν.

We begin by constructing from $\mathbf{A}$ an adversary $\mathbf{A}' = \{\mathbf{A}'_n\}_{n\in\mathbb{N}}$ for KEA3 as follows:

Adversary $\mathbf{A}'_n(q,g,A,B,X)$
$(C,Y) \leftarrow \mathbf{A}_n(q,g,B)$
Return (C,Y)

We have assumed KEA3. Thus there exist an extractor $\bar{\mathbf{A}}' = \{\bar{\mathbf{A}}'_n\}_{n\in\mathbb{N}}$ and a negligible function ν such that $\bar{\mathbf{A}}'$ is a kea3-extractor for $\mathbf{A}'$ with error bound ν. We define an extractor $\bar{\mathbf{A}} = \{\bar{\mathbf{A}}_n\}_{n\in\mathbb{N}}$ for $\mathbf{A}$ as follows:

Extractor $\bar{\mathbf{A}}_n(q, g, B)$
$(c_1, c_2) \leftarrow \bar{\mathbf{A}}'_n(q, g, 1, B, 1)$
Return c_1

We claim that $\bar{\mathbf{A}}$ is a kea1-extractor for $\mathbf{A}$ with error bound ν. To see this, assume that $\bar{\mathbf{A}}'_n(q, g, 1, B, 1)$ is successful, meaning $g^{c_1}1^{c_2} = C$. Then $g^{c_1} = C$, so $\bar{\mathbf{A}}_n(q, g, B)$ is successful as well. ∎

4.5 Three-round zero knowledge

The falsity of KEA2 renders vacuous the result of [59, 60] saying that there exists a negligible-error, 3-round ZK argument for $\mathbb{NP}$. In this section we look at recovering this result.

We first consider the protocol of [59, 60], here called HTP – **H**ada-**T**anaka **P**rotocol. What has been lost is the proof of soundness (i.e., of negligible error). The simplest thing one could hope for is to re-prove soundness of HTP under KEA3 without modifying the protocol. We identify a bug in HTP, however, that renders it unsound. This bug has nothing to do with the assumptions on which the proof of soundness was or can be based.

The bug is, however, small and easily fixed. We consider a modified protocol which we call PHTP – **P**atched HTP. We are able to show that it is sound (i.e., has negligible error) under KEA3 and DLA. Since we have modified the protocol, we need to re-establish ZK under variants of KEA1 as well, but this is easily done.

4.5.1 Arguments

We begin by recalling some definitions. An argument for an $\mathbb{NP}$ language L [28] is a two-party protocol in which a polynomial-time prover tries to "convince" a polynomial-time verifier that their common input x belongs to L. (A party is said to be polynomial time if its running time is polynomial in the length of the common input.) In addition to x, the prover has an auxiliary input a. The protocol is a message exchange at the end of which the verifier outputs a bit indicating its decision to accept or reject. The probability (over the coin tosses of both parties) that the verifier accepts is denoted $\mathbf{Acc}_V^{P,a}(x)$. The formal definition follows.

Definition 4.5.1 [**Argument**] A two-party protocol (P, V), where P and V are both polynomial time, is an *argument for L with error probability δ*: $\mathbb{N} \to [0,1]$, if the following conditions are satisfied:

COMPLETENESS: For all $x \in L$ there exists $w \in \{0,1\}^*$ such that $\mathbf{Acc}_V^{P,w}(x) = 1$.

SOUNDNESS: For all probabilistic polynomial-time algorithms $\widehat{P}$, all sufficiently long $x \notin L$, and all $a \in \{0,1\}^*$: $\mathbf{Acc}_V^{\widehat{P},a}(x) \leq \delta(|x|)$.

We say that (P, V) is a *negligible-error argument for L* if there exists a negligible function δ such that (P, V) is an argument for L with error probability δ. ∎

4.5.2 Canonical arguments

The 3-round protocol proposed by [59, 60], which we call HTP, is based on a 3-round argument $(\bar{P}, \bar{V})$ for an $\mathbb{NP}$-complete language L with the following properties:

(1) The protocol is of the form depicted in Figure 4.2. The prover is identified with a function $\bar{P}$ that given an incoming message M_{in} (this is ε when the prover is initiating the protocol) and its current state St, returns an outgoing message M_{out} and an updated state. The initial state of the prover is

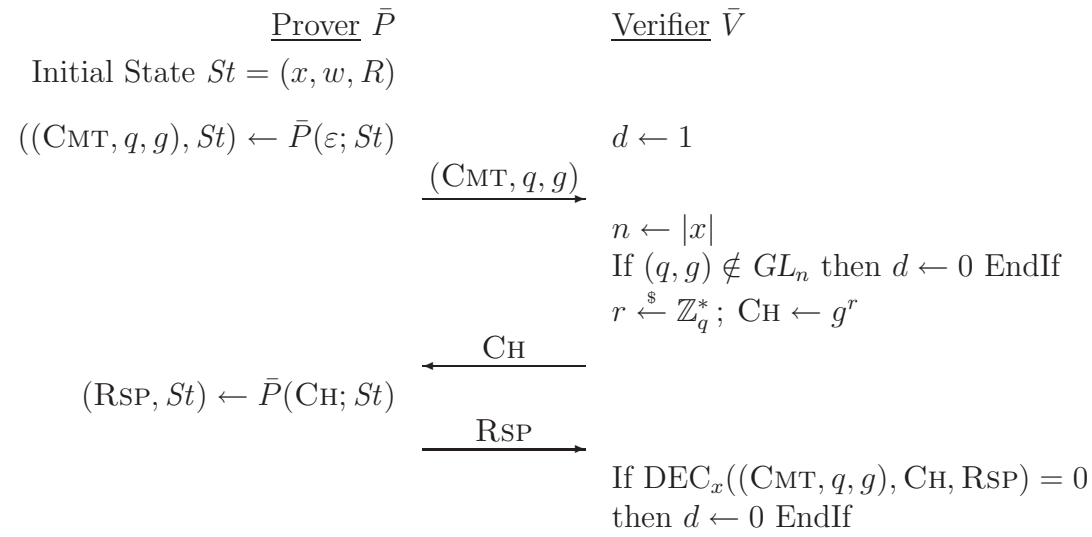

Figure 4.2 **A 3-round argument**. The common input is x. Prover $\bar{P}$ has auxiliary input w and random tape R, and maintains state St. Verifier $\bar{V}$ returns boolean decision d.

(x, w, R), where x is the common input, w is an auxiliary input and R is a random tape. The prover's first message is called its *commitment*. This is a tuple consisting of a string $\textsc{Cmt}$, a prime number q and an element g, where $(q, g) \in GL_{|x|}$. The verifier selects a *challenge* $\textsc{Ch}$ uniformly at random from G_q, and, upon receiving a *response* $\textsc{Rsp}$ from the prover, applies a deterministic *decision predicate* $\mathrm{DEC}_x((\textsc{Cmt}, q, g), \textsc{Ch}, \textsc{Rsp})$ to compute a boolean decision.

(2) For any $x \notin L$ and any commitment $(\textsc{Cmt}, q, g)$, where $(q, g) \in GL_{|x|}$, there is at most one challenge $\textsc{Ch} \in G_q$ for which there exists a response $\textsc{Rsp} \in \{0, 1\}^*$ such that $\mathrm{DEC}_x((\textsc{Cmt}, q, g), \textsc{Ch}, \textsc{Rsp}) = 1$. This property is called *strong soundness*.

(3) The protocol is honest-verifier zero knowledge (HVZK), meaning there exists a probabilistic polynomial-time *simulator* S such that the following two

ensembles are computationally indistinguishable:

$$\{S(x)\}_{x\in L} \quad \text{and} \quad \left\{\mathbf{View}_{\bar{V}}^{\bar{P},W(x)}(x)\right\}_{x\in L},$$

where W is any function that given an input in L returns a witness to its membership in L, and $\mathbf{View}_{\bar{V}}^{\bar{P},W(x)}(x)$ is a random variable taking value $\bar{V}$'s internal coin tosses and the sequence of messages it receives during an interaction between prover $\bar{P}$, with auxiliary input $W(x)$, and verifier $\bar{V}$ on common input x.

If $(\bar{P},\bar{V})$ is a 3-round argument for an $\mathbb{NP}$-complete language, meeting the three conditions above, then we refer to $(\bar{P},\bar{V})$ as a *canonical argument*. In what follows, we assume that we have such canonical arguments. They can be constructed in various ways. For example, a canonical argument can be constructed by modifying the parallel composition of Blum's zero-knowledge protocol for the Hamiltonian circuit problem [23], as described in [59, 60].

4.5.3 The Hada-Tanaka protocol

Let $(\bar{P},\bar{V})$ be a canonical argument for an $\mathbb{NP}$-complete language L, and let DEC be the verifier's decision predicate. The Hada-Tanaka protocol HTP $= (P,V)$ is described in Figure 4.3. Note that V's decision predicate does not include the highlighted portion of its code.

We now observe that HTP is unsound. More precisely, there exist canonical arguments such that the Hada-Tanaka protocol based on them does not have negligible error. This is true for any canonical argument $(\bar{P},\bar{V})$ satisfying the extra condition that for infinitely many $x \notin L$ there exists a commitment $(\textsc{Cmt}_x, q_x, g_x)$ for which there is a response $\textsc{Rsp}_x$ to challenge 1 that will make the verifier accept. There are many such canonical arguments. For instance, a canonical argument satisfying this condition results from using an appropriate encoding of group elements in Hada and Tanaka's modification of the parallel composition of Blum's zero-knowledge protocol for the Hamiltonian circuit problem.

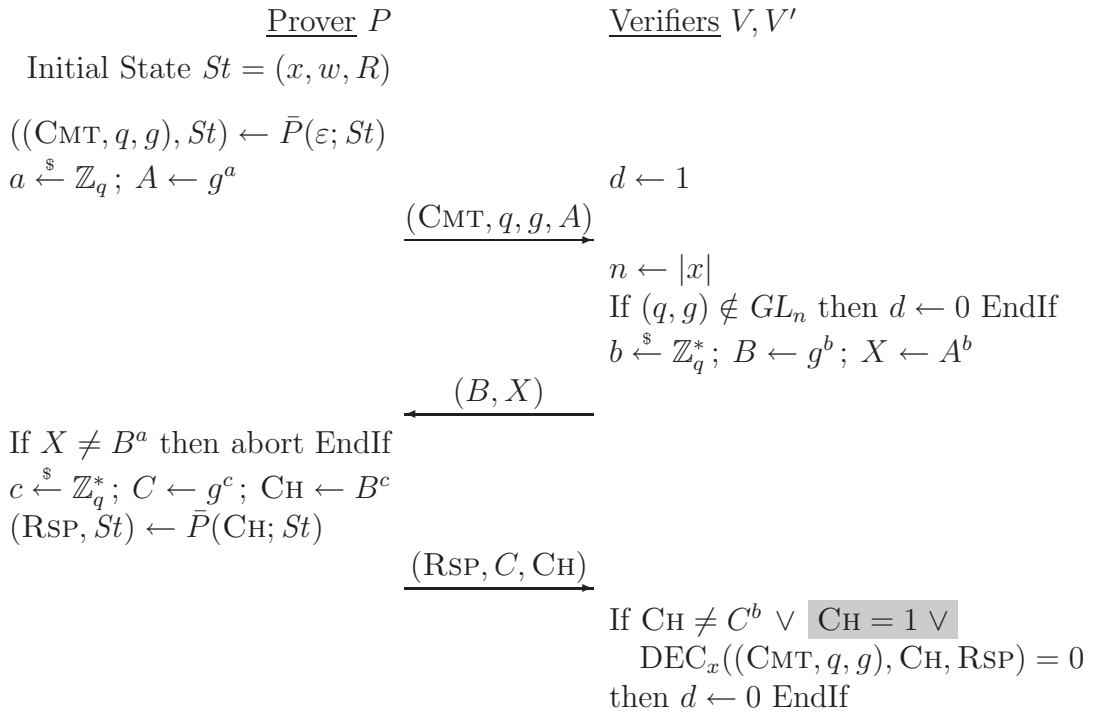

Figure 4.3 **HTP and PHTP**. Verifier V of HTP $= (P, V)$ does not include the highlighted portion. Verifier V' of PHTP $= (P, V')$ does.

Proposition 4.5.2 Let HTP be the Hada-Tanaka protocol based on a canonical argument satisfying the condition stated above. Then there exists a polynomial-time prover for HTP that can make the verifier accept with probability one for infinitely many common inputs not in L. ∎

Proof of Proposition 4.5.2: Let $(\bar{P}, \bar{V})$ be the canonical argument and let V be the verifier of the corresponding protocol HTP. Consider a cheating prover $\widehat{P}$ that on initial state $(x, ((\text{CMT}_x, q_x, g_x), \text{RSP}_x), \varepsilon)$ selects an exponent $a \in \mathbb{Z}_{q_x}$ uniformly at random, and sends $(\text{CMT}_x, q_x, g_x, g_x^a)$ as its commitment to verifier V. Upon receiving a challenge (B, X), it checks if $X = B^a$. If not, it aborts. Otherwise, it sends $(\text{RSP}_x, 1, 1)$ as its response to V. By the assumption about protocol $(\bar{P}, \bar{V})$, for infinitely many $x \notin L$ there exists an auxiliary input $y = ((\text{CMT}_x, q_x, g_x), \text{RSP}_x) \in \{0,1\}^*$ such that $\mathbf{Acc}_V^{\widehat{P}, y}(x) = 1$. ∎

4.5.4 Protocol PHTP

The above attack can be avoided by modifying the verifier to include the highlighted portion of the code in Figure 4.3. We call the resulting verifier V'. The following guarantees that the protocol PHTP $= (P, V')$ is sound under KEA3, if DLP is hard.

Lemma 4.5.3 If KEA3 holds, DLA holds, and $(\bar{P}, \bar{V})$ is a canonical argument for an $\mathbb{NP}$-complete language L, then PHTP $= (P, V')$ as defined in Figure 4.3 is a negligible-error argument for L. ▌

Proof of Lemma 4.5.3: The proof is almost identical to that of Lemma 5.2 in [60]. For completeness, however, we provide it.

Completeness follows directly from the completeness of protocol $(\bar{P}, \bar{V})$. To prove soundness, we proceed by contradiction. Assume that PHTP is not sound, i.e., there is no negligible function δ such that the soundness condition in Definition 4.5.1 holds with respect to δ. We show that DLP is easy under KEA3.

By the assumption that PHTP is not sound and a result of [5], there exists a probabilistic polynomial-time algorithm $\widehat{P}$ such that the function

$$\mathbf{Err}_{\widehat{P}}(n) \;=\; \max\{\; \mathbf{Acc}_{V'}^{\widehat{P},a}(x) \;:\; x \in \{0,1\}^n \wedge x \notin L \wedge a \in \{0,1\}^* \;\}^2$$

is not negligible. Hence there exist a probabilistic polynomial-time algorithm $\widehat{P}$, a polynomial p, and an infinite set $S = \{\; (x,a) \;:\; x \in \{0,1\}^* \setminus L \;\wedge\; a \in \{0,1\}^* \;\}$ such that for every $(x,a) \in S$:

$$\mathbf{Acc}_{V'}^{\widehat{P},a}(x) \;>\; 1/p(|x|)\;, \tag{4.5}$$

and $\{\; x \in \{0,1\}^* \;:\; \exists a \in \{0,1\}^*$ such that $(x,a) \in S \;\}$ is infinite.

[2]We note that this set is finite since $\widehat{P}$ is a polynomial-time algorithm and $\mathbf{Acc}_{V'}^{\widehat{P},a}(x)$ depends only on the first $t_{\widehat{P}}(|x|)$ bits of a, where $t_{\widehat{P}}(\cdot)$ is the running time of $\widehat{P}$.

Since $\widehat{P}$ takes an auxiliary input a, we may assume, without loss of generality, that $\widehat{P}$ is deterministic. We also assume that, if (CMT, q', g', A') is $\widehat{P}$'s commitment on input ε when the initial state is (x, a, ε), for some $x, a \in \{0,1\}^*$ with $|x| = n$, then $(q', g') \in GL_n$. (There exists a prover $\widehat{P}'$ for which $\mathbf{Acc}_{V'}^{\widehat{P}',a}(x) = \mathbf{Acc}_{V'}^{\widehat{P},a}(x)$ for every $x, a \in \{0,1\}^*$ and this assumption holds.) We will use $\widehat{P}$ to construct an adversary $\mathbf{A}$ for the KEA3 problem. By assumption, there exists for it a kea3-extractor $\bar{\mathbf{A}}$ with negligible error bound. Using $\bar{\mathbf{A}}$ and $\widehat{P}$, we then present a poly-size family of randomized circuits $\mathbf{J} = \{\mathbf{J}_n\}_{n\in\mathbb{N}}$ for DLP and show that it does not have a negligible success bound. By Proposition 4.2.2, this implies that DLP is not hard.

Let $K = \{\, n \in \mathbb{N} \;:\; \exists\, (x,a) \in S \text{ such that } |x| = n \,\}$. We observe that K is an infinite set. For each $n \in K$, fix $(x_n, a_n) \in S$ such that $|x_n| = n$. The poly-size family of circuits $\mathbf{A} = \{\mathbf{A}_n\}_{n\in\mathbb{N}}$ for KEA3 is presented in Figure 4.4. Now, under KEA3, there exist a poly-size family of circuits $\bar{\mathbf{A}} = \{\bar{\mathbf{A}}_n\}_{n\in\mathbb{N}}$ and a negligible function ν such that $\bar{\mathbf{A}}$ is a kea3-extractor for $\mathbf{A}$ with error bound ν. For each $n \in K$, let $a'_n = \text{DLog}_{q',g'}(A')$, where (CMT, q', g', A') is $\widehat{P}$'s commitment on input ε when the initial state is (x_n, a_n, ε). Using $\bar{\mathbf{A}}$, we define the poly-size family of circuits $\mathbf{J} = \{\mathbf{J}_n\}_{n\in\mathbb{N}}$ shown in Figure 4.4. The following claim implies that $\mathbf{J}$ does not have a negligible success bound.

Claim 4.5.4 For infinitely many $n \in \mathbb{N}$ there exists $(q, g) \in GL_n$ such that for every $A \in G_q$:

$$\Pr\left[\, g^{\bar{a}} = A \;:\; \bar{a} \xleftarrow{\$} \mathbf{J}_n(q, g, A) \,\right] \;>\; \frac{1}{p(n)^2} - \frac{8}{2^n p(n)} - 2\nu(n)\ . \quad \blacksquare$$

Before proving Claim 4.5.4, we use it to complete the proof of the lemma. The claim and Proposition 4.2.2, DLP is not hard. This contradicts the assumption made in Lemma 4.5.3. $\blacksquare$

Proof of Claim 4.5.4: We let $\Pr[\cdot]$ denote the probability in the experiment of executing $\mathbf{J}_n(q, g, A)$. We show that for every $n \in K$ such that $n \geq 4$, if

$\mathbf{A}_n(q, g, A, B, X)$ // $n \in K$
$St \leftarrow (x_n, a_n, \varepsilon)$; $((\text{CMT}, q', g', A'), St) \leftarrow \widehat{P}(\varepsilon; St)$
If $q' \neq q \vee g' \neq g \vee A' \neq A$ then return $(1, 1)$
else $((\text{RSP}, C, \text{CH}), St) \leftarrow \widehat{P}((B, X); St)$; return (C, CH) EndIf

$\mathbf{A}_n(q, g, A, B, X)$ // $n \notin K$
Return $(1, 1)$

$\mathbf{J}_n(q, g, A)$ // $n \in K$
$St \leftarrow (x_n, a_n, \varepsilon)$; $((\text{CMT}, q', g', A'), St) \leftarrow \widehat{P}(\varepsilon; St)$
If $q' \neq q \vee g' \neq g$ then return $\perp$ EndIf
$b \xleftarrow{\$} \mathbb{Z}_q$; $B \leftarrow A \cdot g^b$; $X \leftarrow B^{a'_n}$
$((\text{RSP}, C, \text{CH}), St_1) \leftarrow \widehat{P}((B, X); St)$; $(c_1, c_2) \leftarrow \bar{\mathbf{A}}_n(q, g, A', B, X)$
If $\text{DEC}_{x_n}((\text{CMT}, q, g), \text{CH}, \text{RSP}) = 0 \vee \text{CH} \neq B^{c_1} X^{c_2}$ then return $\perp$ EndIf
$b' \xleftarrow{\$} \mathbb{Z}_q$; $B' \leftarrow g^{b'}$; $X' \leftarrow B'^{a'_n}$
If $B = B'$ then $\bar{a} \leftarrow b' - b \bmod q$; return $\bar{a}$ EndIf
$((\text{RSP}', C', \text{CH}'), St'_1) \leftarrow \widehat{P}((B', X'); St)$; $(c'_1, c'_2) \leftarrow \bar{\mathbf{A}}_n(q, g, A', B', X')$
If $\text{DEC}_{x_n}((\text{CMT}, q, g), \text{CH}', \text{RSP}') = 0 \vee \text{CH}' \neq B'^{c'_1} X'^{c'_2}$ then
return $\perp$ EndIf
If $c_1 + a'_n c_2 \not\equiv 0 \pmod q$ then
  $\bar{a} \leftarrow (b'c'_1 + b'a'_n c'_2 - bc_1 - ba'_n c_2) \cdot (c_1 + a'_n c_2)^{-1} \bmod q$; return $\bar{a}$
else return $\perp$ EndIf

$\mathbf{J}_n(q, g, A)$ // $n \notin K$
Return $\perp$

Figure 4.4 Adversary $\mathbf{A} = \{\mathbf{A}_n\}_{n \in \mathbb{N}}$ for KEA3 and adversary $\mathbf{J} = \{\mathbf{J}_n\}_{n \in \mathbb{N}}$ for DLP, for the proof of Lemma 4.5.3.

(CMT, q, g, A') is $\widehat{P}$'s commitment on input ε when the initial state is (x_n, a_n, ε), then for every $A \in G_q$:

$$\Pr[\, g^{\bar{a}} = A \,] \;>\; \frac{1}{p(n)^2} - \frac{8}{2^n p(n)} - 2\nu(n) \,.$$

Since K is infinite and, by our assumption about the output of $\widehat{P}$, q, g are such that $(q, g) \in GL_n$, this proves the claim.

Fix $n \in K$ such that $n \geq 4$. Let $(\textsc{Cmt}, q, g, A')$ be $\widehat{P}$'s commitment on input ε when the initial state is (x_n, a_n, ε), and let $A \in G_q$. We first write some inequalities leading to the claim and then justify them:

$$
\begin{aligned}
&\Pr\left[\, g^{\bar{a}} = A \,\right] \\
&\geq \Pr\Big[\mathrm{DEC}_{x_n}((\textsc{Cmt}, q, g), \textsc{Ch}, \textsc{Rsp}) = 1 \wedge \textsc{Ch} = B^{c_1} X^{c_2} \wedge B \neq B' \wedge \\
&\qquad \mathrm{DEC}_{x_n}((\textsc{Cmt}, q, g), \textsc{Ch}', \textsc{Rsp}') = 1 \wedge \textsc{Ch}' = B'^{c'_1} X'^{c'_2} \wedge \\
&\qquad c_1 + a'_n c_2 \not\equiv 0 \pmod{q} \Big] && (4.6) \\
&\geq \Pr\Big[\mathrm{DEC}_{x_n}((\textsc{Cmt}, q, g), \textsc{Ch}, \textsc{Rsp}) = 1 \wedge \textsc{Ch} = B^{c_1} X^{c_2} \wedge \textsc{Ch} \neq 1 \wedge \\
&\qquad B \neq B' \wedge \\
&\qquad \mathrm{DEC}_{x_n}((\textsc{Cmt}, q, g), \textsc{Ch}', \textsc{Rsp}') = 1 \wedge \textsc{Ch}' = B'^{c'_1} X'^{c'_2} \wedge \textsc{Ch}' \neq 1 \Big] && (4.7) \\
&\geq \Pr\Big[\mathrm{DEC}_{x_n}((\textsc{Cmt}, q, g), \textsc{Ch}, \textsc{Rsp}) = 1 \wedge \textsc{Ch} = C^{\mathrm{DLog}_{q,g}(B)} \wedge \textsc{Ch} \neq 1 \wedge \\
&\qquad B \neq B' \wedge \\
&\qquad \mathrm{DEC}_{x_n}((\textsc{Cmt}, q, g), \textsc{Ch}', \textsc{Rsp}') = 1 \wedge \textsc{Ch}' = C'^{\mathrm{DLog}_{q,g}(B')} \wedge \textsc{Ch}' \neq 1 \Big] \\
&\quad - \Big(\Pr\Big[\textsc{Ch} \neq B^{c_1} X^{c_2} \wedge \textsc{Ch} = C^{\mathrm{DLog}_{q,g}(B)} \Big] + \\
&\qquad \Pr\Big[\textsc{Ch}' \neq B'^{c'_1} X'^{c'_2} \wedge \textsc{Ch}' = C'^{\mathrm{DLog}_{q,g}(B')} \Big] \Big) && (4.8) \\
&\geq \Big(\mathbf{Acc}^{\widehat{P}, a_n}_{V'}(x_n) \Big)^2 - \frac{1}{q-1} \mathbf{Acc}^{\widehat{P}, a_n}_{V'}(x_n) - 2\mathbf{Adv}^{\mathrm{kea3}}_{\mathbf{A}, \bar{\mathbf{A}}}(n, q, g, A') && (4.9) \\
&> \frac{1}{p(n)^2} - \frac{1}{(q-1)p(n)} - 2\nu(n) && (4.10) \\
&\geq \frac{1}{p(n)^2} - \frac{8}{2^n p(n)} - 2\nu(n) \,. && (4.11)
\end{aligned}
$$

We justify Equation (4.6) by showing that if $\mathrm{DEC}_{x_n}((\textsc{Cmt}, q, g), \textsc{Ch}, \textsc{Rsp}) = 1$, $\textsc{Ch} = B^{c_1} X^{c_2}$, $B \neq B'$, $\mathrm{DEC}_{x_n}((\textsc{Cmt}, q, g), \textsc{Ch}', \textsc{Rsp}') = 1$, $\textsc{Ch}' = B'^{c'_1} X'^{c'_2}$ and $c_1 + a'_n c_2 \not\equiv 0 \pmod{q}$ then $g^{\bar{a}} = A$. Assume that the former statement holds. By the strong soundness property of protocol $(\bar{P}, \bar{V})$, $\textsc{Ch} = \textsc{Ch}'$, whence $B^{c_1} X^{c_2} = B'^{c'_1} X'^{c'_2}$. Thus we have

$$\begin{aligned} g^{\bar{a}} &= g^{(b'c'_1+b'a'_nc'_2-bc_1-ba'_nc_2)\cdot(c_1+a'_nc_2)^{-1} \bmod q} &&= \left(g^{b'c'_1+b'a'_nc'_2}\right)^{(c_1+a'_nc_2)^{-1}} g^{-b} \\ &= \left(B'^{c'_1}X'^{c'_2}\right)^{(c_1+a'_nc_2)^{-1}} g^{-b} &&= \left(B^{c_1}X^{c_2}\right)^{(c_1+a'_nc_2)^{-1}} g^{-b} \\ &= \left(B^{c_1}B^{a'_nc_2}\right)^{(c_1+a'_nc_2)^{-1}} g^{-b} &&= \left(B^{c_1+a'_nc_2}\right)^{(c_1+a'_nc_2)^{-1}} g^{-b} \\ &= Bg^{-b} &&= A, \end{aligned}$$

as desired.

To justify Equation (4.7) we observe that if $\text{CH} = B^{c_1}X^{c_2}$ and $\text{CH} \neq 1$ then $c_1 + a'_nc_2 \not\equiv 0 \pmod{q}$, and that adding the condition $\text{CH}' \neq 1$ can only decrease the probability further.

Now Equation (4.8) is justified as follows.

$$\begin{aligned} &\Pr\Big[\text{DEC}_{x_n}((\text{CMT}, q, g), \text{CH}, \text{RSP}) = 0 \vee \text{CH} \neq B^{c_1}X^{c_2} \vee \text{CH} = 1 \vee B = B' \vee \\ &\quad \text{DEC}_{x_n}((\text{CMT}, q, g), \text{CH}', \text{RSP}') = 0 \vee \text{CH}' \neq B'^{c'_1}X'^{c'_2} \vee \text{CH}' = 1\Big] \\ &\leq \Pr\Big[\text{DEC}_{x_n}((\text{CMT}, q, g), \text{CH}, \text{RSP}) = 0 \vee \text{CH} \neq C^{\text{DLog}_{q,g}(B)} \vee \text{CH} = 1 \vee \\ &\qquad B = B' \vee \\ &\qquad \text{DEC}_{x_n}((\text{CMT}, q, g), \text{CH}', \text{RSP}') = 0 \vee \text{CH}' \neq C'^{\text{DLog}_{q,g}(B')} \vee \text{CH}' = 1 \vee \\ &\qquad \Big(\text{CH} \neq B^{c_1}X^{c_2} \wedge \text{CH} = C^{\text{DLog}_{q,g}(B)}\Big) \vee \\ &\qquad \Big(\text{CH}' \neq B'^{c'_1}X'^{c'_2} \wedge \text{CH}' = C'^{\text{DLog}_{q,g}(B')}\Big)\Big] \\ &\leq \Pr\Big[\text{DEC}_{x_n}((\text{CMT}, q, g), \text{CH}, \text{RSP}) = 0 \vee \text{CH} \neq C^{\text{DLog}_{q,g}(B)} \vee \text{CH} = 1 \vee \\ &\qquad B = B' \vee \\ &\qquad \text{DEC}_{x_n}((\text{CMT}, q, g), \text{CH}', \text{RSP}') = 0 \vee \text{CH}' \neq C'^{\text{DLog}_{q,g}(B')} \vee \text{CH}' = 1\Big] + \\ &\quad \Pr\Big[\text{CH} \neq B^{c_1}X^{c_2} \wedge \text{CH} = C^{\text{DLog}_{q,g}(B)}\Big] + \\ &\quad \Pr\Big[\text{CH}' \neq B'^{c'_1}X'^{c'_2} \wedge \text{CH}' = C'^{\text{DLog}_{q,g}(B')}\Big]. \end{aligned}$$

$\mathbf{Exp}^{\text{kea3}}_{\mathbf{A},\bar{\mathbf{A}}}(n, q, g, A')$ returns 1 exactly when $Y = C^{\text{DLog}_{q,g}(B)}$ and $g^{c_1}A'^{c_2} \neq C$. By construction of $\mathbf{A}$, we have $Y = \text{CH}$, and thus $\text{CH} = C^{\text{DLog}_{q,g}(B)} \wedge \text{CH} \neq B^{c_1}X^{c_2}$ implies that $\mathbf{Exp}^{\text{kea3}}_{\mathbf{A},\bar{\mathbf{A}}}(n, q, g, A')$ returns 1. Similarly, $\text{CH}' = C^{\text{DLog}_{q,g}(B')} \wedge \text{CH} \neq B'^{c'_1}X'^{c'_2}$ implies that $\mathbf{Exp}^{\text{kea3}}_{\mathbf{A},\bar{\mathbf{A}}}(n, q, g, A')$ returns 1. To justify Equation (4.9) it

remains to show that

$$\begin{aligned}
&\Pr\Big[\,\mathrm{DEC}_{x_n}((\textsc{Cmt},q,g),\textsc{Ch},\textsc{Rsp})=1\wedge\textsc{Ch}=C^{\mathrm{DLog}_{q,g}(B)}\wedge\textsc{Ch}\neq 1\wedge\\
&\quad B\neq B'\wedge\\
&\quad \mathrm{DEC}_{x_n}((\textsc{Cmt},q,g),\textsc{Ch}',\textsc{Rsp}')=1\wedge\textsc{Ch}'=C'^{\mathrm{DLog}_{q,g}(B')}\wedge\textsc{Ch}'\neq 1\,\Big]\\
&\geq\ \Big(\mathbf{Acc}_{V'}^{\widehat{P},a_n}(x_n)\Big)^2-\frac{1}{q-1}\mathbf{Acc}_{V'}^{\widehat{P},a_n}(x_n)\,. \qquad (4.12)
\end{aligned}$$

Let RES denote the event in the experiment of executing $\mathbf{J}_n(q,g,A)$ whose probability is bounded from below in Equation (4.12). Note that the corresponding sample space is $\mathbb{Z}_q^*\times\mathbb{Z}_q^*$. Let ACC denote the event that in an interaction between $\widehat{P}$ (with initial state (x_n,a_n,ε)) and V' (with input x_n), the latter accepts (i.e., $\Pr[\,\mathsf{ACC}\,]=\mathbf{Acc}_{V'}^{\widehat{P},a_n}(x_n)$). The sample space of the corresponding experiment is $\mathbb{Z}_q^*$. We observe that if $b\in\mathsf{ACC}$, $b'\in\mathsf{ACC}$ and $b\neq b'$ then $(b,b')\in\mathsf{RES}$. Therefore,

$$\begin{aligned}
|\mathsf{RES}| &\geq |\mathsf{ACC}|(|\mathsf{ACC}|-1) \quad \text{and}\\
\Pr[\,\mathsf{RES}\,]=\frac{|\mathsf{RES}|}{|\mathbb{Z}_q^*\times\mathbb{Z}_q^*|} &\geq \frac{|\mathsf{ACC}|}{|\mathbb{Z}_q^*|}\left(\frac{|\mathsf{ACC}|}{|\mathbb{Z}_q^*|}-\frac{1}{|\mathbb{Z}_q^*|}\right)\\
&= \Big(\mathbf{Acc}_{V'}^{\widehat{P},a_n}(x_n)\Big)^2-\frac{1}{q-1}\mathbf{Acc}_{V'}^{\widehat{P},a_n}(x_n)\,.
\end{aligned}$$

Equation (4.10) is justified by Equation (4.5) and the assumption that $\bar{\mathbf{A}}$ is a kea3-extractor for $\mathbf{A}$ with error bound ν.

The assumption that $(q,g)\in GL_n$ implies that $|2q+1|=n$, i.e., $2^{n-1}\leq 2q+1<2^n$, and hence $q-1\geq 2^{n-3}$ (recall that $n\geq 4$). This justifies Equation (4.11). ∎

4.5.5 Zero knowledge of PHTP

Having modified HTP, we need to revisit the zero knowledge. We observe that PHTP modifies only the verifier, not the prover. Furthermore, only the decision predicate of the verifier is modified, not the messages it sends. This means that the view (i.e., the internal coin tosses and the sequence of messages received during an

interaction with a prover P) of verifier V' of PHTP is identical to that of verifier V of HTP. Thus, whatever zero knowledge property HTP has is inherited by PHTP, under the same assumptions.

We recall the notions of zero knowledge considered by Hada and Tanaka: zero knowledge (ZK) [56], auxiliary-input ZK [54], non-uniform ZK [51], and auxiliary-input non-uniform ZK [51]; and the assumptions they used to prove zero knowledge of HTP, namely KEA1-A(p) and UKEA1-A(p), where p is a polynomial.

Definition 4.5.5 [**ZK and Auxiliary-input ZK**] Let (P, V) be an argument for an $\mathbb{NP}$ language L. We say that (P, V) is *ZK* (respectively, *auxiliary-input ZK*) *for* L if for every probabilistic polynomial-time algorithm $\widehat{V}$ there exists a probabilistic *simulator* $S_{\widehat{V}}$, that runs in time polynomial in the length of its first input, such that the following two ensembles are computationally indistinguishable:

$$\left\{S_{\widehat{V}}(x,\varepsilon)\right\}_{x\in L} \quad \text{and} \quad \left\{\mathbf{View}_{\widehat{V},\,\varepsilon}^{P,\,W(x)}(x)\right\}_{x\in L}$$

$$\left(\text{respectively, } \left\{S_{\widehat{V}}(x,z)\right\}_{x\in L,\,z\in\{0,1\}^*} \quad \text{and} \quad \left\{\mathbf{View}_{\widehat{V},\,z}^{P,\,W(x)}(x)\right\}_{x\in L,\,z\in\{0,1\}^*}\right),$$

where W is any function that given an input in L returns a witness to its membership in L, and $\mathbf{View}_{\widehat{V},\,z}^{P,\,W(x)}(x)$ is a random variable taking value $\widehat{V}$'s internal coin tosses and the sequence of messages it receives during an interaction between prover P, with auxiliary input $W(x)$, and verifier $\widehat{V}$, with auxiliary input z, on common input x. ∎

Definition 4.5.6 [**Non-uniform ZK and Auxiliary-input non-uniform ZK**] Let (P, V) be an argument for an $\mathbb{NP}$ language L. We say that (P, V) is *non-uniform ZK* (respectively, *auxiliary-input non-uniform ZK*) *for* L if for every poly-size family of circuits $\{V_n\}_{n\in\mathbb{N}}$ there exists a poly-size family of randomized circuits $\{S_n\}_{n\in\mathbb{N}}$ such that the following two ensembles are indistinguishable by poly-size circuits:

$$\left\{S_{|x|}(x,\varepsilon)\right\}_{x\in L} \quad \text{and} \quad \left\{\mathbf{View}_{V_{|x|},\,\varepsilon}^{P,\,W(x)}(x)\right\}_{x\in L}$$

$$\left(\text{respectively, } \left\{S_{|x|}(x,z)\right\}_{x\in L,\, z\in\{0,1\}^*} \text{ and } \left\{\mathbf{View}_{V_{|x|},\,z}^{P,W(x)}(x)\right\}_{x\in L,\, z\in\{0,1\}^*}\right),$$

where W and $\mathbf{View}_{V_{|x|},\,z}^{P,W(x)}(x)$ are as in Definition 4.5.5. ▮

In the following formalizations of assumptions KEA1-A(p) and UKEA1-A(p), we merge the two conditions specified in [60] into one, as we did for KEA1.

Assumption 4.5.7 [KEA1-A] Let $\mathbf{A} = \{\mathbf{A}_n\}_{n\in\mathbb{N}}$ and $\bar{\mathbf{A}} = \{\bar{\mathbf{A}}_n\}_{n\in\mathbb{N}}$ be families of circuits, p a polynomial, and $\nu\colon \mathbb{N} \to [0,1]$ a function. We associate to any $n \in \mathbb{N}$, any $(q,g) \in GL_n$, and any $\sigma \in \{0,1\}^{p(n)}$ the following experiment:

Experiment $\mathbf{Exp}^{\text{kea1-a}}_{\mathbf{A},\bar{\mathbf{A}}}(n,q,g,\sigma)$

$b \xleftarrow{\$} \mathbb{Z}_q$; $B \leftarrow g^b$

$(C,Y) \leftarrow \mathbf{A}_n(q,g,B,\sigma)$; $c \leftarrow \bar{\mathbf{A}}_n(q,g,B,\sigma)$

If ($Y = C^b$ AND $g^c \neq C$) then return 1 else return 0

We let

$$\mathbf{Adv}^{\text{kea1-a}}_{\mathbf{A},\bar{\mathbf{A}}}(n,q,g,\sigma) \;=\; \Pr\left[\,\mathbf{Exp}^{\text{kea1-a}}_{\mathbf{A},\bar{\mathbf{A}}}(n,q,g,\sigma) = 1\,\right]$$

denote the *advantage* of adversary $\mathbf{A}$ relative to $\bar{\mathbf{A}}$ on inputs n,q,g,σ. We say that $\bar{\mathbf{A}}$ is a *kea1-a(p)-extractor for* $\mathbf{A}$ *with error bound* ν if

$$\forall n \in \mathbb{N}\ \forall (q,g) \in GL_n\ \forall \sigma \in \{0,1\}^{p(n)} :\ \mathbf{Adv}^{\text{kea1-a}}_{\mathbf{A},\bar{\mathbf{A}}}(n,q,g,\sigma) \;\leq\; \nu(n)\ .$$

We say that *KEA1-A(p) holds* if for every poly-size family of circuits $\mathbf{A}$ there exist a poly-size family of circuits $\bar{\mathbf{A}}$ and a negligible function ν such that $\bar{\mathbf{A}}$ is a kea1-a(p)-extractor for $\mathbf{A}$ with error bound ν. ▮

Assumption 4.5.8 [UKEA1-A] Let $\mathbf{A}$ and $\bar{\mathbf{A}}$ be probabilistic polynomial-time algorithms, p a polynomial, and $\nu\colon \mathbb{N} \to [0,1]$ a function. We associate to any $n \in \mathbb{N}$, any $(q,g) \in GL_n$, and any $\sigma \in \{0,1\}^{p(n)}$ the following experiment:

Experiment $\mathbf{Exp}^{\text{kea1-au}}_{\mathbf{A},\bar{\mathbf{A}}}(n,q,g,\sigma)$

$b \xleftarrow{\$} \mathbb{Z}_q$; $B \leftarrow g^b$

Choose coins R at random

$(C, Y) \leftarrow \mathbf{A}(n, q, g, B, \sigma; R)$; $c \leftarrow \bar{\mathbf{A}}(n, q, g, B, \sigma; R)$

If $(Y = C^b$ AND $g^c \neq C)$ then return 1 else return 0

We let

$$\mathbf{Adv}^{\text{kea1-au}}_{\mathbf{A},\bar{\mathbf{A}}}(n, q, g, \sigma) \;=\; \Pr\left[\, \mathbf{Exp}^{\text{kea1-au}}_{\mathbf{A},\bar{\mathbf{A}}}(n, q, g, \sigma) = 1 \,\right]$$

denote the *advantage* of adversary $\mathbf{A}$ relative to $\bar{\mathbf{A}}$ on inputs n, q, g, σ. We say that $\bar{\mathbf{A}}$ is a *ukea1-a(p)-extractor for* $\mathbf{A}$ *with error bound* ν if

$$\forall n \in \mathbb{N}\ \forall (q, g) \in GL_n\ \forall \sigma \in \{0,1\}^{p(n)} :\ \mathbf{Adv}^{\text{kea1-au}}_{\mathbf{A},\bar{\mathbf{A}}}(n, q, g, \sigma) \;\leq\; \nu(n)\ .$$

We say that *UKEA1-A(p) holds* if for every probabilistic polynomial-time algorithm $\mathbf{A}$ there exist a probabilistic polynomial-time algorithm $\bar{\mathbf{A}}$ and a negligible function ν such that $\bar{\mathbf{A}}$ is a ukea1-a(p)-extractor for $\mathbf{A}$ with error bound ν. ∎

Hada and Tanaka proved that if the underlying canonical argument is HVZK (property (3) above), then HTP is 1) non-uniform ZK if KEA1-A(p) holds for a polynomial p bounding the length of (x, CMT), 2) auxiliary-input non-uniform ZK if KEA1-A(p) holds for every polynomial p, 3) ZK if UKEA1-A(p) holds for a polynomial p bounding the length of (x, CMT), and 4) auxiliary-input ZK if UKEA1-A(p) holds for every polynomial p. By the discussion above, PHTP has these same properties. For ease of reference, we state this result in the following lemma.

Lemma 4.5.9 Let $(\bar{P}, \bar{V})$ be a canonical argument for an $\mathbb{NP}$-complete language L, and let PHTP $= (P, V')$ be the protocol defined in Figure 4.3. Then

1) PHTP is non-uniform ZK for L if KEA1-A(p) holds for a polynomial p bounding the length of (x, CMT),

2) PHTP is auxiliary-input non-uniform ZK for L if KEA1-A(p) holds for every polynomial p,

3) PHTP is ZK for L if UKEA1-A(p) holds for a polynomial p bounding the length of $(x, \textsc{Cmt})$,

and

4) PHTP is auxiliary-input ZK for L if UKEA1-A(p) holds for every polynomial p. ∎

4.5.6 Summary

In summary, we have shown the following:

Theorem 4.5.10 Assuming DLA and KEA3, there exists a 3-round negligible-error argument for an $\mathbb{NP}$-complete language L that is

1) non-uniform ZK if KEA1-A(p) holds for a particular polynomial p,
2) auxiliary-input non-uniform ZK if KEA1-A(p) holds for every polynomial p,
3) ZK if UKEA1-A(p) holds for a particular polynomial p, and
4) auxiliary-input ZK if UKEA1-A(p) holds for every polynomial p. ∎

Proof of Theorem 4.5.10: The theorem follows immediately from Lemma 4.5.3, Lemma 4.5.9, and the existence of canonical arguments for $\mathbb{NP}$-complete languages. The latter can be constructed, for example, by modifying the parallel composition of Blum's zero-knowledge protocol for the Hamiltonian circuit problem [23], as described in [59, 60]. ∎

Acknowledgements

Proposition 4.4.3 is due to Shai Halevi and we thank him for permission to include it.

This chapter, in part, is a reprint of the material as it appears in M. Bellare and A. Palacio, "The Knowledge-of-Exponent Assumptions and 3-Round Zero-Knowledge Protocols," *Advances in Cryptology - Crypto 2004 Proceedings*, Lecture Notes in Computer Science Vol. 3152, M. Franklin ed., Springer-Verlag, 2004.

5 Plaintext-Aware Public-Key Encryption without Random Oracles

5.1 Introduction

The theory of encryption is concerned with defining and implementing notions of security for encryption schemes [55, 66, 50, 69, 75, 40]. One of the themes in its history is the emergence of notions of security of increasing strength that over time find applications and acceptance.

Our work pursues, from the same perspective, a notion that is stronger than any previous ones, namely plaintext awareness. Our goal is to strengthen the foundations of this notion by lifting it out of the random-oracle model [21] where it currently resides. Towards this end, we provide definitions of a hierarchy of notions of plaintext awareness, relate them to existing notions, and implement some of them. We consider this a first step in the area, however, since important questions are left unresolved. We begin below by reviewing existing work and providing some motivation for our work.

5.1.1 Background

Intuitively, an encryption scheme is plaintext aware (PA) if the "only" way that an adversary can produce a valid ciphertext is to apply the encryption algorithm to the public key and a message. In other words, any adversary against a PA scheme that produces a ciphertext "knows" the corresponding plaintext.

Random-oracle model work

The notion of PA encryption was first suggested by Bellare and Rogaway [22], with the motivation that PA + IND-CPA should imply IND-CCA2. That is, security against chosen-plaintext attack coupled with plaintext awareness should imply security against adaptive chosen-ciphertext attack. The intuition, namely, that if an adversary knows the plaintext corresponding to a ciphertext it produces, then a decryption oracle must be useless to it, goes back to [24, 25]. Bellare and Rogaway [22] provided a formalization of PA in the random-oracle (RO) model. They asked that for every adversary A taking the public key and outputting a ciphertext, there exist an extractor that, given the same public key and a transcript of the interaction of A with its RO, is able to decrypt the ciphertext output by A. We will refer to this notion as PA-BR.

Subsequently, it was found that PA-BR was too weak for PA-BR + IND-CPA to imply IND-CCA2. Bellare, Desai, Pointcheval, and Rogaway [7] traced the cause of this to the fact that PA-BR did not capture the ability of the adversary to obtain ciphertexts via eavesdropping on communications made to the receiver. (Such eavesdropping can put into the adversary's hands ciphertexts whose decryptions it does not know, lending it the ability to create other ciphertexts whose decryptions it does not know.) They provided an appropriately enhanced definition (still in the RO model) that we denote by PA-BDPR, and showed that PA-BDPR + IND-CPA $\rightarrow$ IND-CCA2.

Plaintext awareness is exploited, even though typically implicitly rather than

explicitly, in the proofs of the IND-CCA2-security of numerous RO-model encryption schemes, e.g., [48, 84, 26].

PA and the RO model

By restricting the above-mentioned RO-model definitions to schemes and adversaries that do not query the RO, one obtains natural counterpart standard (i.e., non-RO) model definitions of PA. These standard-model definitions turn out, however, not to be achievable without sacrificing privacy, because the extractor can simply be used for decryption. This indicates that the use of the RO model in the definitions of [22, 7] is central.

Indeed, PA as per [22, 7] is "designed" for the RO model in the sense that the definition aims to capture certain properties of certain RO-model schemes, namely, the fact that possession of the transcript of the interaction of an adversary with its RO permits decryption of ciphertexts formed by this adversary. It is not clear what counterpart this intuition has in the standard model.

The lack of a standard-model definition of PA results in several gaps. One such arises when we consider that RO-model PA schemes are eventually instantiated to get standard-model schemes. In that case, what property are these instantiated schemes even supposed to possess? There is no definition that we might even discuss as a target.

PA via key registration

PA without ROs was first considered by Herzog, Liskov and Micali [61], who define and implement it in an extension of the usual public-key setting. In their setting, the sender (not just the receiver) has a public key, and, in a key-registration phase that precedes encryption, proves knowledge of the corresponding secret key to a key-registration authority via an interactive proof of knowledge. Encryption is a function of the public keys of both the sender and the receiver, and the PA extractor works by extracting the sender secret key using the knowledge extractor

of the interactive proof of knowledge.

Their work also points to an application of plaintext-aware encryption where they claim the use of the latter is crucial in the sense that IND-CCA2-secure encryption does not suffice, namely to securely instantiate the ideal encryption functions of the Dolev-Yao model [41].

5.1.2 Our goals and motivation

The goal of this work is to provide definitions and constructions for plaintext-aware public-key encryption in the standard and classical setting of public-key encryption, namely the one where the receiver (but not the sender) has a public key, and anyone (not just a registered sender) can encrypt a message for the receiver as a function of the receiver's public key. In this setting there is no key-registration authority or key-registration protocol akin to [61].

Motivations include the following. As in the RO model, we would like a tool enabling the construction of public-key encryption schemes secure against chosen-ciphertext attack. We would also like to have some well-defined notion that can be viewed as a target for instantiated RO-model PA schemes. (One could then evaluate these schemes with regard to meeting the target.)

Additionally, we would like to enable the possibility of instantiating the ideal encryption functions of the Dolev-Yao model [41] without recourse to either random oracles or the key-registration model. (The last is an application where, as per [61], PA is required and IND-CCA2 does not suffice. However, see also [2].)

As we will see later, consideration of PA in the standard model brings other benefits, such as some insight, or at least an alternative perspective, on the design of existing encryption schemes secure against chosen-ciphertext attack. Let us now discuss our contributions.

5.1.3 Definitions

The first contribution of this paper is to provide definitions for plaintext-aware encryption in the standard model and standard public-key setting.

Overview

We provide a hierarchy consisting of three notions of increasing strength that we denote by PA0, PA1 and PA2. There are several motivations for this. One is that these will be seen (in conjunction with IND-CPA) to imply security against chosen-ciphertext attacks of different strengths. Another is that, as will become apparent, PA is difficult to achieve, and progress can be made by first achieving it in weaker forms. Finally, it is useful, pedagogically, to bring in new definitional elements incrementally.

A closer look

Our basic definitional framework considers a polynomial-time adversary $\boldsymbol{C}$, called a ciphertext creator, that takes input the public key and can query ciphertexts to an oracle. A polynomial-time algorithm $\boldsymbol{C}^*$ is said to be a successful extractor for $\boldsymbol{C}$ if it can provide replies to the oracle queries of $\boldsymbol{C}$ that are computationally indistinguishable from those provided by a decryption oracle.

An important element of the above framework is that the extractor gets as input *the same public key as the ciphertext creator, as well as the coin tosses of the ciphertext creator*. This reflects the intuition that the extractor is the "subconscious" of the adversary, and begins with exactly the same information as the adversary itself.

We say that an encryption scheme is PA0 (respectively, PA1) if there exists a successful extractor for any ciphertext creator that makes only a single oracle query (respectively, a polynomial number of oracle queries).

Eavesdropping capability in PA2 is captured by providing the ciphertext

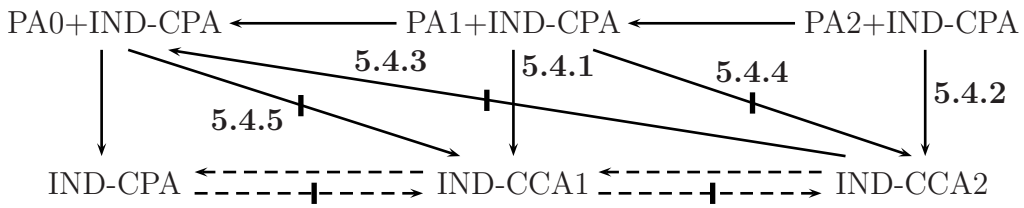

Figure 5.1 **Relations between PA and notions of privacy**. An arrow is an implication, and, in the directed graph given by the arrows, there is a path from A to B if and only if A implies B. The hatched arrows represent separations. Solid lines represent results from this paper, while dashed lines represent results from prior work [7, 40]. The number on an arrow or hatched arrow refers to the theorem in this paper that establishes this relationship. Absence of a number on a solid arrow means the result is trivial.

---

creator $\boldsymbol{C}$ with an additional oracle that returns ciphertexts, but care has to be taken in defining this oracle. It does *not* suffice to let it be an encryption oracle because we want to model the ability of the adversary to obtain ciphertexts whose decryptions it may not know. Our formalization of PA2 allows the additional oracle to compute a plaintext, as a function of the query made to it and coins unknown to $\boldsymbol{C}$, and return the encryption of this plaintext to $\boldsymbol{C}$.

Formal definitions of PA0, PA1 and PA2, based on the above ideas, are in Section 5.3, which includes a discussion of how these definitions compare to the earlier RO-model ones.

5.1.4 Relations

PA by itself is not a notion of privacy, and so we are typically interested in PA coupled with the minimal notion of privacy, namely IND-CPA [55, 66]. We consider six notions, namely, PA0 + IND-CPA, PA1 + IND-CPA and PA2 + IND-CPA, on the one hand, and the standard notions of privacy IND-CPA, IND-CCA1 [69] and IND-CCA2 [75], on the other. We provide implications and separations among these six notions in the style of [7, 40]. The results are depicted in Figure 5.1. For notions A, B, an implication, represented by A $\rightarrow$ B, means that every encryption

scheme satisfying notion A also satisfies notion B, and a separation, represented by A $\not\rightarrow$ B, means that there exists an encryption scheme satisfying notion A but not satisfying notion B. (The latter assumes there exists some encryption scheme satisfying notion A, since otherwise the question is vacuous.)

Figure 5.1 shows a minimal set of arrows and hatched arrows, but the relation between any two notions is resolved by the given relations. For example, IND-CCA1 $\not\rightarrow$ PA1 + IND-CPA, because, otherwise, there would be a path from IND-CCA2 to PA0 + IND-CPA, contradicting the hatched arrow labeled 5.4.3. Similarly, we get PA0 $\not\rightarrow$ PA1 $\not\rightarrow$ PA2, meaning the three notions of plaintext awareness are of increasing strength.

The main implications are that PA1 + IND-CPA implies IND-CCA1 and PA2 + IND-CPA implies IND-CCA2. The PA1 + IND-CPA $\rightarrow$ IND-CCA1 result shows that even a notion of PA not taking eavesdropping adversaries into account is strong enough to imply security against a significant class of chosen-ciphertext attacks. Since the PA + IND-CPA $\rightarrow$ IND-CCA2 implication has been a motivating target for definitions of PA, the PA2 + IND-CPA $\rightarrow$ IND-CCA2 result provides some validation for the definition of PA2.

Among the separations, we note that IND-CCA2 does not imply PA0, meaning even the strongest form of security against chosen-ciphertext attack is not enough to guarantee the weakest form of plaintext awareness.

5.1.5 Constructions

The next problem we address is to find provably-secure plaintext-aware encryption schemes.

Approaches

A natural approach to consider is to include a non-interactive zero-knowledge proof of knowledge [78] of the message in the ciphertext. However, as we explain

in Section 5.5, this fails to achieve PA.

As such approaches are considered and discarded, it becomes apparent that achieving even the weaker forms of PA in the standard (as opposed to RO) model may be difficult. We have been able to make progress, however, under some strong assumptions that we now describe.

DHK assumptions

Let G be the order q subgroup of $\mathbb{Z}^*_{2q+1}$, where $q, 2q+1$ are primes, and let g be a generator of G. Damgård [36] introduced and used an assumption that states, roughly, that an adversary given g^a and outputting a pair of the form (g^b, g^{ab}) must "know" b. The latter is captured by requiring an extractor that given the adversary coins and inputs can output b. We call our formalization of this assumption (cf. Assumption 5.5.2) DHK0.[1] We also introduce an extension of this assumption called DHK1 (cf. Assumption 5.5.1), in which the adversary does not just output one pair (g^b, g^{ab}), but instead interacts with the extractor, feeding it such pairs adaptively and each time expecting back the discrete logarithm of the first component of the pair.

The DEG scheme

Damgård presented a simple ElGamal variant that we call DEG. It is efficient, requiring only three exponentiations to encrypt and two to decrypt.

We prove that DEG is PA0 under the DHK0 assumption and PA1 under the DHK1 assumption. Since DEG is easily seen to be IND-CPA-secure under the DDH assumption, and we saw above that PA1 + IND-CPA $\rightarrow$ IND-CCA1, a

[1] Another formalization, called DA-1, is used by Hada and Tanaka [60]. (We refer to the full version of their paper [60], which points out that the formalization of the preliminary version [59] is wrong.) This differs from DHK0 in being for a non-uniform setting. DA-1 is called KEA1 in Chapter 3, based on Naor's terminology [68]: KEA stands for "knowledge of exponent." Hada and Tanaka [60] also introduced and used another assumption, that they call DA-2 and is called KEA2 in Chapter 3, but there we show that this assumption is false. The DHK0/DA-1/KEA1 assumptions, to the best of our knowledge, are not known to be false.

consequence is that DEG is IND-CCA1-secure assuming DHK1 and DDH. DEG is in fact the most efficient IND-CCA1-secure scheme known to date to be provably secure in the standard model.

Damgård [36] claims that DEG meets a notion of security under ciphertext attack that we call RPR-CCA1, assuming DHK0 and assuming the ElGamal scheme meets a notion called RPR-CPA. (Both notions are recalled in Section 5.6, and are weaker than IND-CCA1 and IND-CPA, respectively). As we explain in Section 5.6, his proof has a flaw, but his overall approach and intuition are valid, and the proof can be fixed by simply assuming DHK1 in place of DHK0. In summary, our contribution is (1) to show that DEG meets a stronger and more standard notion of security than RPR-CCA1, namely IND-CCA1, and (2) to show it is PA0 and PA1, indicating that it has even stronger properties, and providing some formal support for the intuition given in [36] about the security underlying the scheme.

CSL

CSL is a simpler and more efficient version of the Cramer-Shoup encryption scheme [35] that is IND-CCA1-secure under the DDH assumption. We show that CSL is PA0 under the DHK0 assumption and PA1 under the DHK1 assumption. (IND-CPA-security under DDH being easy to see, this again implies that CSL is IND-CCA1-secure under DHK1 and DDH, but in this case the conclusion is not novel.) What we believe is interesting about our results is that they show that some form of plaintext awareness underlies the CSL scheme, and this provides perhaps an alternative viewpoint on the source of its security. We remark, however, that DEG is more efficient than CSL.

Warning and discussion

DHK0 and DHK1 are strong and non-standard assumptions. As pointed out by Naor [68], they are not efficiently falsifiable. (However, such assumptions can

be shown to be false as exemplified in [18]). However standard-model schemes, even under strong assumptions, might provide better guarantees than RO model schemes, for we know that the latter may not provide real-world security guarantees at all [29, 70, 57, 6]. Also, PA without random oracles is challenging to achieve, and we consider it important to "break ground" by showing it is possible, even if under strong assumptions.

Achieving PA2

The eavesdropping capability provided to an adversary in the PA2 setting seems to render the task of finding constructions sonewhat harder. We were not able to find any, and conjectured that the Cramer-Shoup scheme, already known to be IND-CCA2-secure, could be proved PA2-secure under some appropriate assumption. (Intuitively, it seems to be PA2.)

Dent [37] recently proved that if DHK1 holds, then the Cramer-Shoup *hybrid-encryption scheme* [35] acting on fixed-length messages is PA2-secure. The reader is referred to Dent's paper for a precise statement of the result.

Open problems

It would be nice to achieve PA0, PA1, or PA2 under weaker and more standard assumptions than those used here.

5.2 Notation and standard definitions

We denote by [ ] the empty list. Given a list L and an element x, $L \mathbin{@} x$ denotes the list consisting of the elements in L followed by x.

Unless otherwise indicated, an algorithm is randomized.

5.2.1 Encryption schemes

We recall the standard syntax. An asymmetric (also called public-key) encryption scheme is a tuple $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D}, \mathsf{MsgSp})$ whose components are as follows. The polynomial-time key-generation algorithm $\mathcal{K}$ takes input 1^k, where $k \in \mathbb{N}$ is the security parameter, and returns a pair (pk, sk) consisting of a public key and matching secret key. The polynomial-time encryption algorithm $\mathcal{E}$ takes a public key pk and a message M to return a ciphertext C. We write $C \xleftarrow{\$} \mathcal{E}_{pk}(M)$ to represent the operation of executing $\mathcal{E}$ on pk and M and letting C denote the ciphertext returned. The deterministic, polynomial-time decryption algorithm $\mathcal{D}$ takes a secret key sk and a ciphertext C to return either a message M or the special symbol $\perp$ indicating that the ciphertext is invalid. We write $M \leftarrow \mathcal{D}_{sk}(C)$ to represent the operation of executing $\mathcal{D}$ on sk and C and letting M denote the response. The message-space function MsgSp associates to each public key pk a set $\mathsf{MsgSp}(pk)$ called the message space of pk. It is required that for every $k \in \mathbb{N}$:

$$\Pr\left[\, (pk, sk) \xleftarrow{\$} \mathcal{K}(1^k)\,;\ M \xleftarrow{\$} \mathsf{MsgSp}(pk)\,;\ C \xleftarrow{\$} \mathcal{E}_{pk}(M) \ :\ \mathcal{D}_{sk}(C) = M \,\right] \ =\ 1\ .$$

5.2.2 Standard security notions

We recall the definitions of IND-CPA-, IND-CCA1-, and IND-CCA2-security that originate in [55], [69], and [75], respectively. We use the formalizations of [7]. Let $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D}, \mathsf{MsgSp})$ be an asymmetric encryption scheme, let $k \in \mathbb{N}$ and $b \in \{0, 1\}$. Let $\boldsymbol{X}$ be an algorithm with access to an oracle. For aaa $\in \{\text{cpa}, \text{cca1}, \text{cca2}\}$, consider the following experiment

Experiment $\mathbf{Exp}^{\text{ind-aaa-}b}_{\mathcal{AE},\boldsymbol{X}}(k)$
$(pk, sk) \xleftarrow{\$} \mathcal{K}(1^k)\,;\ (M_0, M_1, St) \xleftarrow{\$} \boldsymbol{X}^{\mathcal{O}_1(\cdot)}(\mathsf{find}, pk)\,;\ C \xleftarrow{\$} \mathcal{E}_{pk}(M_b)$
$d \leftarrow \boldsymbol{X}^{\mathcal{O}_2(\cdot)}(\mathsf{guess}, C, St)\,;$ Return d

where

If aaa = cpa	then	$\mathcal{O}_1(\cdot) = \varepsilon$	and	$\mathcal{O}_2(\cdot) = \varepsilon$
If aaa = cca1	then	$\mathcal{O}_1(\cdot) = \mathcal{D}_{sk}(\cdot)$	and	$\mathcal{O}_2(\cdot) = \varepsilon$
If aaa = cca2	then	$\mathcal{O}_1(\cdot) = \mathcal{D}_{sk}(\cdot)$	and	$\mathcal{O}_2(\cdot) = \mathcal{D}_{sk}(\cdot)$

Experiment $\mathbf{Exp}^{\text{pa1-d}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D}}(k)$
$(pk, sk) \xleftarrow{\$} \mathcal{K}(1^k)$; $x \xleftarrow{\$} \boldsymbol{C}^{\mathcal{D}_{sk}(\cdot)}(pk)$; $d \xleftarrow{\$} \boldsymbol{D}(x)$; Return d

---

Experiment $\mathbf{Exp}^{\text{pa1-x}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D},\boldsymbol{C}^*}(k)$
$(pk, sk) \xleftarrow{\$} \mathcal{K}(1^k)$
Choose coins $R[\boldsymbol{C}], R[\boldsymbol{C}^*]$ for $\boldsymbol{C}, \boldsymbol{C}^*$, respectively ; $St[\boldsymbol{C}^*] \leftarrow (pk, R[\boldsymbol{C}])$
Run $\boldsymbol{C}$ on input pk and coins $R[\boldsymbol{C}]$ until it halts, replying to its oracle queries as follows:
– If $\boldsymbol{C}$ makes query Q then
$(M, St[\boldsymbol{C}^*]) \leftarrow \boldsymbol{C}^*(Q, St[\boldsymbol{C}^*]; R[\boldsymbol{C}^*])$
Return M to $\boldsymbol{C}$ as the reply EndIf
Let x denote the output of $\boldsymbol{C}$; $d \xleftarrow{\$} \boldsymbol{D}(x)$; Return d

Figure 5.2 Experiments used to define PA1 and PA0.

In each case it is required that $M_0, M_1 \in \mathsf{MsgSp}(pk)$ and $|M_0| = |M_1|$. In the case of IND-CCA2, it is also required that $\boldsymbol{X}$ not query its decryption oracle with ciphertext C. We call $\boldsymbol{X}$ an *ind-aaa-adversary*. The *ind-aaa-advantage* of $\boldsymbol{X}$ is

$$\mathbf{Adv}^{\text{ind-aaa}}_{\mathcal{AE},\boldsymbol{X}}(k) \;=\; \Pr\left[\,\mathbf{Exp}^{\text{ind-aaa-1}}_{\mathcal{AE},\boldsymbol{X}}(k) = 1\,\right] - \Pr\left[\,\mathbf{Exp}^{\text{ind-aaa-0}}_{\mathcal{AE},\boldsymbol{X}}(k) = 1\,\right] .$$

For AAA $\in \{\text{CPA}, \text{CCA1}, \text{CCA2}\}$, $\mathcal{AE}$ is said to be IND-AAA-secure if the function $\mathbf{Adv}^{\text{ind-aaa}}_{\mathcal{AE},\boldsymbol{X}}$ is negligible for every polynomial-time ind-aaa-adversary $\boldsymbol{X}$.

5.3 New notions of plaintext awareness

In this section we provide our formalizations of plaintext-aware encryption. We provide the formal definitions first and explanations later. We begin with PA1, then define PA0 via this, and finally define PA2.

Definition 5.3.1 [**PA1**] Let $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D}, \mathsf{MsgSp})$ be an asymmetric encryption scheme. Let $\boldsymbol{C}$ be an algorithm that has access to an oracle, takes as input a public key pk, and returns a string. Let $\boldsymbol{D}$ be an algorithm that takes a string and returns a bit. Let $\boldsymbol{C}^*$ be an algorithm that takes a string and some state information, and

returns a message or the symbol $\perp$, and a new state. We call $\boldsymbol{C}$ a *ciphertext-creator* adversary, $\boldsymbol{D}$ a *distinguisher*, and $\boldsymbol{C}^*$ a *pa1-extractor*. For $k \in \mathbb{N}$, we define the experiments shown in Figure 5.2. The *pa1-advantage* of $\boldsymbol{C}$ relative to $\boldsymbol{D}$ and $\boldsymbol{C}^*$ is

$$\mathbf{Adv}^{\mathrm{pa1}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D},\boldsymbol{C}^*}(k) \;=\; \Pr\left[\,\mathbf{Exp}^{\mathrm{pa1\text{-}d}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D}}(k) = 1\,\right] - \Pr\left[\,\mathbf{Exp}^{\mathrm{pa1\text{-}x}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D},\boldsymbol{C}^*}(k) = 1\,\right].$$

We say that $\boldsymbol{C}^*$ is a *successful pa1-extractor for* $\boldsymbol{C}$ if for every polynomial-time distinguisher $\boldsymbol{D}$ the function $\mathbf{Adv}^{\mathrm{pa1}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D},\boldsymbol{C}^*}$ is negligible. We say $\mathcal{AE}$ is *PA1-secure* if for any polynomial-time ciphertext creator there exists a successful polynomial-time pa1-extractor. ▮

Definition 5.3.2 [**PA0**] Let $\mathcal{AE}$ be an asymmetric encryption scheme. We call a ciphertext-creator adversary that makes *exactly one* oracle query a *pa0 ciphertext creator*. We call a pa1-extractor for a pa0 ciphertext creator a *pa0-extractor*. We say that $\mathcal{AE}$ is *PA0-secure* if for any polynomial-time pa0 ciphertext creator there exists a successful polynomial-time pa0-extractor. ▮

We now explain the ideas behind the above formalisms. The core of the formalization of plaintext awareness of asymmetric encryption scheme $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D}, \mathsf{MsgSp})$ considers a polynomial-time ciphertext-creator adversary $\boldsymbol{C}$ that takes input a public key pk, has access to an oracle and returns a string. The adversary tries to distinguish between the cases that its oracle is $\mathcal{D}_{sk}(\cdot)$, or it is an *extractor* algorithm $\boldsymbol{C}^*$ that takes as input the same public key pk. PA1-security requires that there exist a polynomial-time $\boldsymbol{C}^*$ such that $\boldsymbol{C}$'s outputs in the two cases are indistinguishable. We allow $\boldsymbol{C}^*$ to be stateful, maintaining state $St[\boldsymbol{C}^*]$ across invocations. Importantly, $\boldsymbol{C}^*$ *is provided with the coin tosses of* $\boldsymbol{C}$; otherwise, $\boldsymbol{C}^*$ would be functionally equivalent to the decryption algorithm and thus could not exist unless $\mathcal{AE}$ were insecure with regard to providing privacy. We remark that this formulation is stronger than one not involving a distinguisher $\boldsymbol{D}$, in which $\boldsymbol{C}$ simply outputs a bit representing its guess, since $\boldsymbol{C}^*$ gets the coins of $\boldsymbol{C}$, but not the coins of $\boldsymbol{D}$.

Experiment $\mathbf{Exp}^{\text{pa2-d}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P},\boldsymbol{D}}(k)$

$(pk, sk) \xleftarrow{\$} \mathcal{K}(1^k)$; $\textsc{List} \leftarrow [\,]$
Choose coins $R[\boldsymbol{C}], R[\boldsymbol{P}]$ for $\boldsymbol{C}, \boldsymbol{P}$, respectively ; $St[\boldsymbol{P}] \leftarrow \varepsilon$
Run $\boldsymbol{C}$ on input pk and coins $R[\boldsymbol{C}]$ until it halts, replying to its oracle queries as follows:

- If $\boldsymbol{C}$ makes query (dec, Q) then
  $M \leftarrow \mathcal{D}_{sk}(Q)$; Return M to $\boldsymbol{C}$ as the reply EndIf
- If $\boldsymbol{C}$ makes query (enc, Q) then
  $(M, St[\boldsymbol{P}]) \leftarrow \boldsymbol{P}(Q, St[\boldsymbol{P}]; R[\boldsymbol{P}])$; $C \xleftarrow{\$} \mathcal{E}_{pk}(M)$
  $\textsc{List} \leftarrow \textsc{List} \mathbin{@} C$; Return C to $\boldsymbol{C}$ as the reply EndIf

Let x denote the output of $\boldsymbol{C}$; $d \xleftarrow{\$} \boldsymbol{D}(x)$; Return d

---

Experiment $\mathbf{Exp}^{\text{pa2-x}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P},\boldsymbol{D},\boldsymbol{C}^*}(k)$

$(pk, sk) \xleftarrow{\$} \mathcal{K}(1^k)$; $\textsc{List} \leftarrow [\,]$
Choose coins $R[\boldsymbol{C}], R[\boldsymbol{P}], R[\boldsymbol{C}^*]$ for $\boldsymbol{C}, \boldsymbol{P}, \boldsymbol{C}^*$, respectively
$St[\boldsymbol{P}] \leftarrow \varepsilon$; $St[\boldsymbol{C}^*] \leftarrow (pk, R[\boldsymbol{C}])$
Run $\boldsymbol{C}$ on input pk and coins $R[\boldsymbol{C}]$ until it halts, replying to its oracle queries as follows:

- If $\boldsymbol{C}$ makes query (dec, Q) then
  $(M, St[\boldsymbol{C}^*]) \leftarrow \boldsymbol{C}^*(Q, \textsc{List}, St[\boldsymbol{C}^*]; R[\boldsymbol{C}^*])$
  Return M to $\boldsymbol{C}$ as the reply EndIf
- If $\boldsymbol{C}$ makes query (enc, Q) then
  $(M, St[\boldsymbol{P}]) \leftarrow \boldsymbol{P}(Q, St[\boldsymbol{P}]; R[\boldsymbol{P}])$; $C \xleftarrow{\$} \mathcal{E}_{pk}(M)$
  $\textsc{List} \leftarrow \textsc{List} \mathbin{@} C$; Return C to $\boldsymbol{C}$ as the reply EndIf

Let x denote the output of $\boldsymbol{C}$; $d \xleftarrow{\$} \boldsymbol{D}(x)$; Return d

Figure 5.3 Experiments used to define PA2.

PA0-security considers only adversaries that make a *single* query in their attempt to determine if the oracle is a decryption oracle or an extractor.

Definition 5.3.3 **[PA2]** Let $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D}, \mathsf{MsgSp})$ be an asymmetric encryption scheme. Let $\boldsymbol{C}$ be an algorithm that has access to an oracle, takes as input a public key pk, and returns a string. Let $\boldsymbol{P}$ be an algorithm that takes a string and some state information, and returns a message and a new state. Let $\boldsymbol{D}$ be an algorithm that takes a string and returns a bit. Let $\boldsymbol{C}^*$ be an algorithm that takes a string, a

list of strings and some state information, and returns a message or the symbol $\perp$, and a new state. We call $\boldsymbol{C}$ a *ciphertext-creator* adversary, $\boldsymbol{P}$ a *plaintext-creator* adversary, $\boldsymbol{D}$ a *distinguisher*, and $\boldsymbol{C}^*$ a *pa2-extractor*. For $k \in \mathbb{N}$, we define the experiments shown in Figure 5.3. It is required that, in these experiments, $\boldsymbol{C}$ not make a query (dec, C) for which $C \in \textsc{Clist}$. The *pa2-advantage* of $\boldsymbol{C}$ relative to $\boldsymbol{P}$, $\boldsymbol{D}$ and $\boldsymbol{C}^*$ is

$$\begin{aligned}&\mathbf{Adv}^{\mathrm{pa2}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P},\boldsymbol{D},\boldsymbol{C}^*}(k)\\ &\quad = \Pr\left[\mathbf{Exp}^{\mathrm{pa2\text{-}d}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P},\boldsymbol{D}}(k) = 1\right] - \Pr\left[\mathbf{Exp}^{\mathrm{pa2\text{-}x}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P},\boldsymbol{D},\boldsymbol{C}^*}(k) = 1\right].\end{aligned}$$

We say that $\boldsymbol{C}^*$ is a *successful pa2-extractor for* $\boldsymbol{C}$ if for every polynomial-time plaintext creator $\boldsymbol{P}$ and distinguisher $\boldsymbol{D}$, the function $\mathbf{Adv}^{\mathrm{pa2}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P},\boldsymbol{D},\boldsymbol{C}^*}$ is negligible. We say $\mathcal{AE}$ is *PA2-secure* if for any polynomial-time ciphertext creator there exists a successful polynomial-time pa2-extractor. ▮

In the definition of PA2, the core setting of PA1 is enhanced to model the real-life capability of a ciphertext creator to obtain ciphertexts via eavesdropping on communications made by a third party to the receiver (cf. [7]). Providing $\boldsymbol{C}$ with an encryption oracle does not capture this because eavesdropping puts into $\boldsymbol{C}$'s hands ciphertexts of which it does *not* know the corresponding plaintext, and, although we disallow $\boldsymbol{C}$ to query these to its oracle, it might be able to use them to create other ciphertexts whose corresponding plaintext it does not know and on which the extractor fails.

Modeling eavesdropping requires balancing two elements: providing $\boldsymbol{C}$ with a capability to obtain ciphertexts of plaintexts it does not know, yet capturing the fact that $\boldsymbol{C}$ might have partial information about the plaintexts, or control of the distribution from which these plaintexts are drawn. We introduce a companion *plaintext-creator* adversary $\boldsymbol{P}$ who, upon receiving a communication from $\boldsymbol{C}$, creates a plaintext and forwards it to an encryption oracle. The ciphertext emanating from the encryption oracle is sent to both $\boldsymbol{C}$ and $\boldsymbol{C}^*$. $\boldsymbol{C}$ has some control over $\boldsymbol{P}$

via its communication to $\boldsymbol{P}$, but we ensure this is not total by *denying* $\boldsymbol{C}$ *and* $\boldsymbol{C}^*$ *the coin tosses of* $\boldsymbol{P}$, and also by asking that $\boldsymbol{C}^*$ depend on $\boldsymbol{C}$ but not on $\boldsymbol{P}$.

The extractor $\boldsymbol{C}^*$ is, as before, provided with the coin tosses of $\boldsymbol{C}$. Two types of oracle queries are allowed to $\boldsymbol{C}$. Via a query (dec, Q), it can ask its oracle to decrypt ciphertext Q. Alternatively, it can make a query (enc, Q) to call $\boldsymbol{P}$ with argument Q, upon which the latter computes a message M and forwards it to the encryption oracle, which returns the resulting ciphertext to $\boldsymbol{C}$, and $\boldsymbol{C}^*$ in the case that $\boldsymbol{C}$'s oracle is $\boldsymbol{C}^*$. We observe that if an asymmetric encryption scheme is PA2-secure then it is PA1-secure, and if it is PA1-secure then it is PA0-secure.

5.3.1 Comparison

The RO-model definitions PA-BR [22] and PA-BDPR [7] differ from ours in the following ways.

The RO-model definitions did not give the extractor the coins of the ciphertext creator. As we indicated above, in the absence of ROs, providing the extractor with the coins of the ciphertext creator is necessary for the non-triviality of PA, since otherwise the extractor can be used for decryption and the scheme will not be IND-CPA-secure. Furthermore, the basic intuition is that the extractor is the subconscious of the ciphertext creator, and thus should have all the inputs of the latter, meaning it should be given the public key and the coins that were given to the ciphertext creator.

The RO-model definitions required the extractor to return the decryption of a given ciphertext. We have weakened this requirement, asking only that the outputs of the extractor be computationally indistinguishable from the outputs of the decryption oracle, because this weakening preserves the main implications and applications of PA while increasing the possibility of finding constructions. However, as discussed below, one can consider a stronger, statistical version of our definitions which captures requiring the extractor to return correct decryptions,

and this might be useful in some contexts.

The RO-model definitions only consider ciphertext creators that output a single ciphertext which the extractor must decrypt, while we have considered an oracle-based formulation, corresponding to ciphertext creators that adaptively create multiple ciphertexts for the extractor to decrypt.

The most important changes are with regard to modeling eavesdropping. In PA-BDPR [7], eavesdropping capability was modeled by providing the ciphertext creator with an encryption oracle but denying it and the extractor the so-called indirect random-oracle queries, namely those made by the encryption oracle. Adopting this approach in the absence of ROs would reduce to providing the ciphertext creator with an encryption oracle, which, as we discussed above, does not correctly model eavesdropping because the ciphertext creator knows the decryptions of ciphertexts it obtains via an encryption oracle, and we want to provide it a means of obtaining ciphertexts whose decryptions it may not know. In particular, the encryption-oracle-based notion seems too weak to prove Theorem 5.4.2. We have used the plaintext creator instead.

The plaintext extractor in PA-BDPR [7] is black box, meaning there is a single extractor that works for all ciphertext creators, upon being given the transcript of interactions of the ciphertext creator with its oracles. We have weakened this requirement, allowing the extractor code to depend non-uniformly on the code of the ciphertext creator. Again, this was done in order to increase the possibility of finding constructions. Evidence of the power of non-black-box formulations is provided, in another context, by [3].

We note that it is easy to "lift" our standard-model definitions to counterpart RO-model definitions following the paradigm of [21]. Call these PA0-RO, PA1-RO and PA2-RO. Given the above, we suggest that these make more suitable RO model notions than the existing PA-BR [22] and PA-BDPR [7] ones, since they are rooted in the standard model rather than being RO-model definitions with no standard-model counterparts. We remark that PA-BR implies PA0-RO and PA-

BDPR implies PA2-RO, which says that we have weakened the definitions. Yet we are still in line with the original intuition and have preserved the important implications and application potential.

5.3.2 Statistical PA

Stronger versions of our definitions of PA0, PA1 and PA2 are obtained by requiring that the outputs of the ciphertext creator, in the case where its oracle is the decryption algorithm and in the case where it is the extractor, are statistically rather than computationally indistinguishable. Formally, this can be captured by simply allowing the distinguisher to be computationally unlimited. In other words, let us say $\mathcal{AE}$ is *sPA1-secure* if for any polynomial-time ciphertext creator $\boldsymbol{C}$ there exists a polynomial-time extractor $\boldsymbol{C}^*$ such that the function $\mathbf{Adv}^{\mathrm{pa1}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D},\boldsymbol{C}^*}$ is negligible for every (not necessarily polynomial-time) distinguisher $\boldsymbol{D}$. We can define sPA0- and sPA2-security analogously.

The statistical versions of our definitions amount simply to saying that the extractor must return the correct decryption of any ciphertext it is given, except with negligible probability. Accordingly, this could certainly have been formulated in a simpler way without introducing distinguishers at all, and, indeed, it may have been pedagogically preferable to begin with this simpler and stronger definition and only then get to our current ones. We preferred the current distinguisher-based approach because it allows us to fit the computational and statistical settings into a common definitional framework.

As indicated above, we have chosen to make the computational versions of the definitions the main ones because they suffice for the applications of Theorems 5.4.1 and 5.4.2, and might make future constructions easier to find. We remark, however, that the schemes DEG and CSL that we show to achieve PA0- or PA1-security actually achieve sPA0- and sPA1-security under the same assumptions, meaning that Theorems 5.5.3, and 5.5.4 are true if we replace PA0 by sPA0 and PA1 by

sPA1.

5.4 Relations among notions

We now state the formal results corresponding to Figure 5.1, beginning with the two motivating applications of our notions of plaintext awareness. The proof of the following is in Section 5.4.1.

Theorem 5.4.1 [**PA1 + IND-CPA ⇒ IND-CCA1**] Let $\mathcal{AE}$ be an asymmetric encryption scheme. If $\mathcal{AE}$ is PA1-secure and IND-CPA-secure, then it is IND-CCA1-secure. ∎

The proof of the following is in Section 5.4.2.

Theorem 5.4.2 [**PA2 + IND-CPA ⇒ IND-CCA2**] Let $\mathcal{AE}$ be an asymmetric encryption scheme. If $\mathcal{AE}$ is PA2-secure and IND-CPA-secure, then it is IND-CCA2-secure. ∎

It is natural to ask whether the converse of Theorem 5.4.2 (resp., Theorem 5.4.1) is true, namely whether an asymmetric encryption scheme that is IND-CCA2- (resp., IND-CCA1-) secure is also PA2- (respectively, PA1-) secure. (It is, of course, IND-CPA-secure). The answer is no.

The following theorem implies that PA2- (respectively, PA1-) security (in conjunction with IND-CPA-security) is a strictly stronger requirement than IND-CCA2- (respectively, IND-CCA1-) security, unless there simply do not exist any IND-CCA2-secure schemes. The proof is in Section 5.4.3.

Theorem 5.4.3 [**IND-CCA2 ⇏ PA0 + IND-CPA**] Assume there exists an IND-CCA2-secure asymmetric encryption scheme. Then there exists an IND-CCA2-secure asymmetric encryption scheme that is *not* PA0-secure. ∎

We remind the reader that each notion of PA in the absence of IND-CPA security is trivial to achieve. (In particular, the encryption scheme in which the

encryption function sets the ciphertext equal to the plaintext is PA2-secure, but not IND-CPA-secure.) Thus the fact that the scheme guaranteed by Theorem 5.4.3 is IND-CPA-secure is important.

To complete the picture of implications and separations between the PA + IND-CPA notions and the IND-AAA notions, we now show that PA1-security (in conjunction with IND-CPA-security) is not sufficient to achieve IND-CCA2-security, and PA0-security (in conjunction with IND-CPA-security) is not sufficient to achieve IND-CCA1-security. The proof of the following is in Section 5.4.4.

Theorem 5.4.4 [**PA1 + IND-CPA $\not\Rightarrow$ IND-CCA2**] Assume there exists a PA1-secure and IND-CPA-secure asymmetric encryption scheme. Then there exists a PA1-secure and IND-CPA-secure asymmetric encryption scheme that is *not* IND-CCA2-secure. ∎

The proof of the following is in Section 5.4.5.

Theorem 5.4.5 [**PA0 + IND-CPA $\not\Rightarrow$ IND-CCA1**] Assume there exists a PA0-secure and IND-CPA-secure asymmetric encryption scheme. Then there exists a PA0-secure and IND-CPA-secure asymmetric encryption scheme that is *not* IND-CCA1-secure. ∎

5.4.1 Proof of Theorem 5.4.1

Assume that $\mathcal{AE}$ is PA1-secure and IND-CPA-secure, and let $\boldsymbol{X}$ be a polynomial-time ind-cca1-adversary attacking $\mathcal{AE}$. We construct a polynomial-time ciphertext creator $\boldsymbol{C}$ for $\mathcal{AE}$, based on $\boldsymbol{X}$, and let $\boldsymbol{C}^*$ be a successful polynomial-time pa1-extractor for it. Then we construct a polynomial-time ind-cpa-adversary $\boldsymbol{Y}$ for $\mathcal{AE}$, based on $\boldsymbol{X}$ and $\boldsymbol{C}^*$. Finally, we construct polynomial-time distinguishers $\boldsymbol{D}_0$ and $\boldsymbol{D}_1$ for $\boldsymbol{C}$, and prove that for every $k \in \mathbb{N}$:

$$\mathbf{Adv}^{\text{ind-cca1}}_{\mathcal{AE},\boldsymbol{X}}(k) \;\leq\; \mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{AE},\boldsymbol{Y}}(k) + \mathbf{Adv}^{\text{pa1}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D}_0,\boldsymbol{C}^*}(k) + \mathbf{Adv}^{\text{pa1}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D}_1,\boldsymbol{C}^*}(k)\,. \quad (5.1)$$

The assumption that $\mathcal{AE}$ is PA1-secure and IND-CPA-secure implies that the function $\mathbf{Adv}^{\text{ind-cca1}}_{\mathcal{AE},\boldsymbol{X}}$ is negligible, and thus that $\mathcal{AE}$ is IND-CCA1-secure. We proceed to the constructions and analysis.

The four algorithms we construct are defined in Figure 5.4. Clearly, they all run in polynomial time. Ciphertext-creator adversary $\boldsymbol{C}$ is essentially the same as $\boldsymbol{X}(\mathsf{find},\cdot)$, except that it returns the public key along with M_0, M_1, St. By the assumption that $\mathcal{AE}$ is PA1-secure, there is a successful polynomial-time pa1-extractor $\boldsymbol{C}^*$ for $\boldsymbol{C}$.

A random tape $R[\boldsymbol{C}]\|R[\boldsymbol{C}^*]$ for ind-cpa-adversary $\boldsymbol{Y}$ has two parts, one being a random tape for $\boldsymbol{C}$ (equivalently, for $\boldsymbol{X}$) and the other being a random tape for $\boldsymbol{C}^*$. $\boldsymbol{Y}(\mathsf{find},\cdot)$ initializes and then maintains state for $\boldsymbol{C}^*$. It runs $\boldsymbol{X}(\mathsf{find},\cdot)$, and if the latter makes a query, then $\boldsymbol{Y}(\mathsf{find},\cdot)$ runs $\boldsymbol{C}^*$ to compute a reply, which it returns to $\boldsymbol{X}(\mathsf{find},\cdot)$. When $\boldsymbol{X}(\mathsf{find},\cdot)$ stops, $\boldsymbol{Y}(\mathsf{find},\cdot)$ returns the former's output. $\boldsymbol{Y}(\mathsf{guess},\cdot)$ is identical to $\boldsymbol{X}(\mathsf{guess},\cdot)$.

Consider distinguishers $\boldsymbol{D}_0$ and $\boldsymbol{D}_1$. Intuitively, for $b \in \{0,1\}$, when $\boldsymbol{D}_b$ is run on the output of $\boldsymbol{C}$ after the latter has interacted with the decryption oracle, $\boldsymbol{D}_0$ computes the complement of the outcome of experiment $\mathbf{Exp}^{\text{ind-cca1-0}}_{\mathcal{AE},\boldsymbol{X}}(k)$, and $\boldsymbol{D}_1$ computes the outcome of experiment $\mathbf{Exp}^{\text{ind-cca1-1}}_{\mathcal{AE},\boldsymbol{X}}(k)$.

We claim that Equation (5.1) holds for all $k \in \mathbb{N}$. To prove this, fix $k \in \mathbb{N}$. We state four claims, conclude the proof given them, and then return to prove the claims. The first two claims relate the probability that $\boldsymbol{X}$ guesses the value of challenge bit b, in each of its experiments, to the probability that distinguisher $\boldsymbol{D}_b$ returns 1 when it is run on the output of $\boldsymbol{C}$ after the latter has interacted with the decryption oracle, in experiment $\mathbf{Exp}^{\text{pa1-d}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D}_b}(k)$.

Claim 5.4.6 $\Pr\left[\,\mathbf{Exp}^{\text{ind-cca1-1}}_{\mathcal{AE},\boldsymbol{X}}(k) = 1\,\right] \;=\; \Pr\left[\,\mathbf{Exp}^{\text{pa1-d}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D}_1}(k) = 1\,\right]$. ∎

Claim 5.4.7 $\Pr\left[\,\mathbf{Exp}^{\text{ind-cca1-0}}_{\mathcal{AE},\boldsymbol{X}}(k) = 1\,\right] \;=\; 1 - \Pr\left[\,\mathbf{Exp}^{\text{pa1-d}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D}_0}(k) = 1\,\right]$. ∎

The other claims relate the probability that $\boldsymbol{Y}$ guesses the value of challenge bit b, in each of its experiments, to the probability that distinguisher $\boldsymbol{D}_b$ returns 1 when

Ciphertext creator $\boldsymbol{C}(pk; R[\boldsymbol{C}])$
  Run $\boldsymbol{X}(\mathsf{find}, \cdot)$ on input pk and coins $R[\boldsymbol{C}]$ until it halts, replying to its oracle queries as follows:
  – If $\boldsymbol{X}(\mathsf{find}, \cdot)$ makes query Q then
    Make query Q ; Upon receiving a response M, return M to $\boldsymbol{X}(\mathsf{find}, \cdot)$ as the reply EndIf
  Let (M_0, M_1, St) denote the output of $\boldsymbol{X}(\mathsf{find}, \cdot)$
  Return (M_0, M_1, St, pk)

Adversary $\boldsymbol{Y}(\mathsf{find}, pk; R[\boldsymbol{Y}])$
  Parse $R[\boldsymbol{Y}]$ as $R[\boldsymbol{C}] \| R[\boldsymbol{C}^*]$; $St[\boldsymbol{C}^*] \leftarrow (pk, R[\boldsymbol{C}])$
  Run $\boldsymbol{X}(\mathsf{find}, \cdot)$ on input pk and coins $R[\boldsymbol{C}]$ until it halts, replying to its oracle queries as follows:
  – If $\boldsymbol{X}(\mathsf{find}, \cdot)$ makes query Q then
    $(M, St[\boldsymbol{C}^*]) \leftarrow \boldsymbol{C}^*(Q, St[\boldsymbol{C}^*]; R[\boldsymbol{C}^*])$; Return M to $\boldsymbol{X}(\mathsf{find}, \cdot)$ as the reply EndIf
  Let (M_0, M_1, St) denote the output of $\boldsymbol{X}(\mathsf{find}, \cdot)$
  Return (M_0, M_1, St)

Adversary $\boldsymbol{Y}(\mathsf{guess}, C, St)$
  $d \leftarrow \boldsymbol{X}(\mathsf{guess}, C, St)$
  Return d

Distinguisher $\boldsymbol{D}_0(x)$
  Parse x as (M_0, M_1, St, pk)
  $C \xleftarrow{\$} \mathcal{E}_{pk}(M_0)$
  $d \leftarrow \boldsymbol{X}(\mathsf{guess}, C, St)$
  Return $\bar{d}$

Distinguisher $\boldsymbol{D}_1(x)$
  Parse x as (M_0, M_1, St, pk)
  $C \xleftarrow{\$} \mathcal{E}_{pk}(M_1)$
  $d \leftarrow \boldsymbol{X}(\mathsf{guess}, C, St)$
  Return d

Figure 5.4 Ciphertext-creator adversary $\boldsymbol{C}$, ind-cpa-adversary $\boldsymbol{Y}$, and distinguishers $\boldsymbol{D}_0$, $\boldsymbol{D}_1$ for the proof of Theorem 5.4.1.

---

it is run on the output of $\boldsymbol{C}$ after the latter has interacted with pa2-extractor $\boldsymbol{C}^*$, in experiment $\mathbf{Exp}^{\text{pa1-x}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D}_b,\boldsymbol{C}^*}(k)$.

Claim 5.4.8 $\Pr\left[\mathbf{Exp}^{\text{ind-cpa-1}}_{\mathcal{AE},\boldsymbol{Y}}(k) = 1\right] = \Pr\left[\mathbf{Exp}^{\text{pa1-x}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D}_1,\boldsymbol{C}^*}(k) = 1\right]$. ∎

Claim 5.4.9 $\Pr\left[\mathbf{Exp}^{\text{ind-cpa-0}}_{\mathcal{AE},\boldsymbol{Y}}(k) = 1\right] = 1 - \Pr\left[\mathbf{Exp}^{\text{pa1-x}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D}_0,\boldsymbol{C}^*}(k) = 1\right]$. ∎

Applying these claims, we obtain Equation (5.1) as follows:

$$
\begin{aligned}
&\mathbf{Adv}^{\text{ind-cca1}}_{\mathcal{AE},\boldsymbol{X}}(k) \\
&\quad = \Pr\left[\,\mathbf{Exp}^{\text{ind-cca1-1}}_{\mathcal{AE},\boldsymbol{X}}(k) = 1\,\right] - \Pr\left[\,\mathbf{Exp}^{\text{ind-cca1-0}}_{\mathcal{AE},\boldsymbol{X}}(k) = 1\,\right] \\
&\quad = \Pr\left[\,\mathbf{Exp}^{\text{pa1-d}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D}_1}(k) = 1\,\right] - \left(1 - \Pr\left[\,\mathbf{Exp}^{\text{pa1-d}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D}_0}(k) = 1\,\right]\right) \\
&\quad = \left(\Pr\left[\,\mathbf{Exp}^{\text{pa1-x}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D}_1,\boldsymbol{C}^*}(k) = 1\,\right] + \mathbf{Adv}^{\text{pa1}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D}_1,\boldsymbol{C}^*}(k)\right) - 1 \\
&\qquad + \left(\Pr\left[\,\mathbf{Exp}^{\text{pa1-x}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D}_0,\boldsymbol{C}^*}(k) = 1\,\right] + \mathbf{Adv}^{\text{pa1}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D}_0,\boldsymbol{C}^*}(k)\right) \\
&\quad = \Pr\left[\,\mathbf{Exp}^{\text{pa1-x}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D}_1,\boldsymbol{C}^*}(k) = 1\,\right] - \left(1 - \Pr\left[\,\mathbf{Exp}^{\text{pa1-x}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D}_0,\boldsymbol{C}^*}(k) = 1\,\right]\right) \\
&\qquad + \mathbf{Adv}^{\text{pa1}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D}_0,\boldsymbol{C}^*}(k) + \mathbf{Adv}^{\text{pa1}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D}_1,\boldsymbol{C}^*}(k) \\
&\quad = \Pr\left[\,\mathbf{Exp}^{\text{ind-cpa-1}}_{\mathcal{AE},\boldsymbol{Y}}(k) = 1\,\right] - \Pr\left[\,\mathbf{Exp}^{\text{ind-cpa-0}}_{\mathcal{AE},\boldsymbol{Y}}(k) = 1\,\right] \\
&\qquad + \mathbf{Adv}^{\text{pa1}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D}_0,\boldsymbol{C}^*}(k) + \mathbf{Adv}^{\text{pa1}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D}_1,\boldsymbol{C}^*}(k) \\
&\quad = \mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{AE},\boldsymbol{Y}}(k) + \mathbf{Adv}^{\text{pa1}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D}_0,\boldsymbol{C}^*}(k) + \mathbf{Adv}^{\text{pa1}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D}_1,\boldsymbol{C}^*}(k)\ .
\end{aligned}
$$

It remains to prove the four claims above.

Proof of Claim 5.4.6: Let $\mathcal{S}_{\text{cca1-1}}$ denote the sample space underlying $\mathbf{Exp}^{\text{ind-cca1-1}}_{\mathcal{AE},\boldsymbol{X}}(k)$. A member of this space is a string specifying coin tosses for all algorithms involved, which in this case means the coins of the key-generation algorithm, the random tape of $\boldsymbol{X}$ itself, and the coins used by the encryption algorithm.

A member of the sample space $\mathcal{S}_{\text{pa1-d-1}}$ underlying $\mathbf{Exp}^{\text{pa1-d}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D}_1}(k)$ is a string specifying the coins of the key-generation algorithm, the random tape of $\boldsymbol{C}$, and the random tape of $\boldsymbol{D}_1$. Claim 5.4.6 follows once we observe that by the definitions of $\boldsymbol{C}$ and $\boldsymbol{D}_1$, $\mathcal{S}_{\text{pa1-d-1}}$ is equal to $\mathcal{S}_{\text{cca1-1}}$ (The random tape of $\boldsymbol{C}$ consists of coins for $\boldsymbol{X}$, and the random tape of $\boldsymbol{D}_1$ consists of coins for the encryption algorithm.), and $\mathbf{Exp}^{\text{ind-cca1-1}}_{\mathcal{AE},\boldsymbol{X}}(k) = 1$ if and only if $\mathbf{Exp}^{\text{pa1-d}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D}_1}(k) = 1$. ∎

Proof of Claim 5.4.7: Similarly to the proof of Claim 5.4.6, it is easy to see that the sample space underlying $\mathbf{Exp}^{\text{ind-cca1-0}}_{\mathcal{AE},\boldsymbol{X}}(k)$ is identical to the sample space underlying $\mathbf{Exp}^{\text{pa1-d}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D}_0}(k)$, and $\mathbf{Exp}^{\text{ind-cca1-0}}_{\mathcal{AE},\boldsymbol{X}}(k) = 1$ if and only if $\mathbf{Exp}^{\text{pa1-d}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D}_0}(k) =$

0. ∎

Proof of Claim 5.4.8: Let $\mathcal{S}_{\text{pa1-x-1}}$ denote the sample space underlying $\mathbf{Exp}^{\text{pa1-x}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D}_1,\boldsymbol{C}^*}(k)$. A member of this space is a string specifying the coins of the key-generation algorithm, the random tape of $\boldsymbol{C}$, the random tape of $\boldsymbol{C}^*$, and the random tape of $\boldsymbol{D}_1$. The latter consists of coins for the encryption algorithm.

A member of the sample space $\mathcal{S}_{\text{cpa-1}}$ underlying $\mathbf{Exp}^{\text{ind-cpa-1}}_{\mathcal{AE},\boldsymbol{Y}}(k)$ is a string specifying the coins of the key-generation algorithm, the random tape of $\boldsymbol{Y}$, and the coins used by the encryption algorithm. The random tape of $\boldsymbol{Y}$ consists of coins for $\boldsymbol{C}$ and coins for $\boldsymbol{C}^*$. Hence $\mathcal{S}_{\text{cpa-1}} = \mathcal{S}_{\text{pa1-x-1}}$. We observe that $\mathbf{Exp}^{\text{ind-cpa-1}}_{\mathcal{AE},\boldsymbol{Y}}(k) = 1$ if and only if $\mathbf{Exp}^{\text{pa1-x}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D}_1,\boldsymbol{C}^*}(k) = 1$. Claim 5.4.8 follows. ∎

Proof of Claim 5.4.9: Similarly to the proof of Claim 5.4.8, it is easy to see that the sample space underlying $\mathbf{Exp}^{\text{ind-cpa-0}}_{\mathcal{AE},\boldsymbol{Y}}(k)$ is identical to the sample space underlying $\mathbf{Exp}^{\text{pa1-x}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D}_0,\boldsymbol{C}^*}(k)$, and $\mathbf{Exp}^{\text{ind-cpa-0}}_{\mathcal{AE},\boldsymbol{Y}}(k) = 1$ if and only if $\mathbf{Exp}^{\text{pa1-x}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D}_0,\boldsymbol{C}^*}(k) = 0$. ∎

5.4.2 Proof of Theorem 5.4.2

Assume that $\mathcal{AE}$ is PA2-secure and IND-CPA-secure, and let $\boldsymbol{X}$ be a polynomial-time ind-cca2-adversary attacking $\mathcal{AE}$. We construct a polynomial-time ciphertext creator $\boldsymbol{C}$ for $\mathcal{AE}$, based on $\boldsymbol{X}$, and polynomial-time plaintext creators $\boldsymbol{P}_0$ and $\boldsymbol{P}_1$, and let $\boldsymbol{C}^*$ be a successful polynomial-time pa2-extractor for $\boldsymbol{C}$. Then we construct a polynomial-time ind-cpa-adversary $\boldsymbol{Y}$ for $\mathcal{AE}$, based on $\boldsymbol{X}$ and $\boldsymbol{C}^*$. Finally, we construct polynomial-time distinguishers $\boldsymbol{D}_0$ and $\boldsymbol{D}_1$ for $\boldsymbol{C}$, and prove that for every $k \in \mathbb{N}$:

$$\begin{aligned}&\mathbf{Adv}^{\text{ind-cca2}}_{\mathcal{AE},\boldsymbol{X}}(k)\\ &\quad\leq\ \mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{AE},\boldsymbol{Y}}(k) + \mathbf{Adv}^{\text{pa2}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P}_0,\boldsymbol{D}_0,\boldsymbol{C}^*}(k) + \mathbf{Adv}^{\text{pa2}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P}_1,\boldsymbol{D}_1,\boldsymbol{C}^*}(k)\,. \qquad (5.2)\end{aligned}$$

The assumption that $\mathcal{AE}$ is PA2-secure and IND-CPA-secure implies that the function $\mathbf{Adv}^{\text{ind-cca2}}_{\mathcal{AE},\boldsymbol{X}}$ is negligible, and thus that $\mathcal{AE}$ is IND-CCA2-secure. We proceed

Ciphertext creator $\boldsymbol{C}(pk; R[\boldsymbol{C}])$
  Run $\boldsymbol{X}(\mathsf{find}, \cdot)$ on input pk and coins $R[\boldsymbol{C}]$ until it halts, replying to its oracle queries as follows:
  – If $\boldsymbol{X}(\mathsf{find}, \cdot)$ makes query Q then
    Make query (dec, Q) ; Upon receiving a response M, return M to $\boldsymbol{X}(\mathsf{find}, \cdot)$ as the reply EndIf
  Let (M_0, M_1, St) denote the output of $\boldsymbol{X}(\mathsf{find}, \cdot)$
  Make query $(\mathsf{enc}, (M_0, M_1))$; Upon receiving a response C, run $\boldsymbol{X}(\mathsf{guess}, \cdot)$ on input C, St, until it halts, replying to its oracle queries as follows:
  – If $\boldsymbol{X}(\mathsf{guess}, \cdot)$ makes query Q then
    Make query (dec, Q) ; Upon receiving a response M, return M to $\boldsymbol{X}(\mathsf{guess}, \cdot)$ as the reply EndIf
  Let d denote the output of $\boldsymbol{X}(\mathsf{guess}, \cdot)$
  Return d

Plaintext creator $\boldsymbol{P}_0(Q, St[\boldsymbol{P}]; R[\boldsymbol{P}])$	Plaintext creator $\boldsymbol{P}_1(Q, St[\boldsymbol{P}]; R[\boldsymbol{P}])$
Parse Q as (M_0, M_1)	Parse Q as (M_0, M_1)
Return $(M_0, St[\boldsymbol{P}])$	Return $(M_1, St[\boldsymbol{P}])$

Figure 5.5 Ciphertext-creator adversary $\boldsymbol{C}$ and plaintext-creator adversaries $\boldsymbol{P}_0$, $\boldsymbol{P}_1$ for the proof of Theorem 5.4.2.

---

to the constructions and analysis.

The six algorithms we construct are defined in Figure 5.5 and Figure 5.6. Clearly, they all run in polynomial time. Ciphertext creator $\boldsymbol{C}$ is essentially the same as $\boldsymbol{X}$, except that instead of outputting (M_0, M_1, St), it calls a plaintext creator with argument (M_0, M_1) and, upon receiving a response C, it continues the execution of $\boldsymbol{X}$ by running $\boldsymbol{X}(\mathsf{guess}, \cdot)$ on input C, St. Plaintext creator $\boldsymbol{P}_0$ takes input a pair of messages, and selects the first message. Plaintext creator $\boldsymbol{P}_1$ takes input a pair of messages, and selects the second message. We observe that for $b \in \{0, 1\}$, since in experiment $\mathbf{Exp}^{\text{ind-cca2-}b}_{\mathcal{AE},\boldsymbol{X}}(k)$, $\boldsymbol{X}$ does not query its decryption oracle with ciphertext C, in experiments $\mathbf{Exp}^{\text{pa2-d}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P}_b,\boldsymbol{D}_b}(k)$ and $\mathbf{Exp}^{\text{pa2-x}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P}_b,\boldsymbol{D}_b,\boldsymbol{C}^*}(k)$, $\boldsymbol{C}$ does not make a query (dec, Q) for which $Q \in \textsc{Clist}$. By the assumption that $\mathcal{AE}$ is PA2-secure, there is a successful polynomial-time pa2-extractor $\boldsymbol{C}^*$ for $\boldsymbol{C}$.

Adversary $\boldsymbol{Y}(\mathsf{find}, pk; R[\boldsymbol{Y}])$
  Parse $R[\boldsymbol{Y}]$ as $R[\boldsymbol{C}] \| R[\boldsymbol{C}^*]$; $St[\boldsymbol{C}^*] \leftarrow (pk, R[\boldsymbol{C}])$
  Run $\boldsymbol{X}(\mathsf{find}, \cdot)$ on input pk and coins $R[\boldsymbol{C}]$ until it halts, replying to its oracle queries as follows:
    – If $\boldsymbol{X}(\mathsf{find}, \cdot)$ makes query Q then
      $(M, St[\boldsymbol{C}^*]) \leftarrow \boldsymbol{C}^*(Q, St[\boldsymbol{C}^*]; R[\boldsymbol{C}^*])$
      Return M to $\boldsymbol{X}(\mathsf{find}, \cdot)$ as the reply EndIf
  Let (M_0, M_1, St) denote the output of $\boldsymbol{X}(\mathsf{find}, \cdot)$
  $St' \leftarrow (St, St[\boldsymbol{C}^*], R[\boldsymbol{C}^*])$
  Return (M_0, M_1, St')

Adversary $\boldsymbol{Y}(\mathsf{guess}, C, St')$
  Parse St' as $(St, St[\boldsymbol{C}^*], R[\boldsymbol{C}^*])$
  Run $\boldsymbol{X}(\mathsf{guess}, \cdot)$ on input C, St until it halts, replying to its oracle queries as follows:
    – If $\boldsymbol{X}(\mathsf{guess}, \cdot)$ makes query Q then
      $(M, St[\boldsymbol{C}^*]) \leftarrow \boldsymbol{C}^*(Q, St[\boldsymbol{C}^*]; R[\boldsymbol{C}^*])$
      Return M to $\boldsymbol{X}(\mathsf{guess}, \cdot)$ as the reply EndIf
  Let d denote the output of $\boldsymbol{X}(\mathsf{guess}, \cdot)$
  Return d

Distinguisher $\boldsymbol{D}_0(x)$	Distinguisher $\boldsymbol{D}_1(x)$
Return $\bar{x}$	Return x

Figure 5.6 Ind-cpa-adversary $\boldsymbol{Y}$ and distinguishers $\boldsymbol{D}_0$, $\boldsymbol{D}_1$ for the proof of Theorem 5.4.2.

A random tape $R[\boldsymbol{C}] \| R[\boldsymbol{C}^*]$ for ind-cpa-adversary $\boldsymbol{Y}$ has two parts, one being a random tape for $\boldsymbol{C}$ (equivalently, for $\boldsymbol{X}$) and the other being a random tape for $\boldsymbol{C}^*$. $\boldsymbol{Y}(\mathsf{find}, \cdot)$ initializes and then maintains state for $\boldsymbol{C}^*$. It runs $\boldsymbol{X}(\mathsf{find}, \cdot)$, and if the latter makes a query, then $\boldsymbol{Y}(\mathsf{find}, \cdot)$ runs $\boldsymbol{C}^*$ to compute a reply, which it returns to $\boldsymbol{X}(\mathsf{find}, \cdot)$. When $\boldsymbol{X}(\mathsf{find}, \cdot)$ outputs (M_0, M_1, St) and stops, $\boldsymbol{Y}(\mathsf{find}, \cdot)$ computes some state information St' that is used by $\boldsymbol{Y}(\mathsf{guess}, \cdot)$ and returns (M_0, M_1, St'). $\boldsymbol{Y}(\mathsf{guess}, \cdot)$ runs $\boldsymbol{X}(\mathsf{guess}, \cdot)$, and if the latter makes a query, then $\boldsymbol{Y}(\mathsf{guess}, \cdot)$ runs $\boldsymbol{C}^*$ to compute a reply, which it returns to $\boldsymbol{X}(\mathsf{guess}, \cdot)$. When $\boldsymbol{X}(\mathsf{guess}, \cdot)$ stops, $\boldsymbol{Y}(\mathsf{guess}, \cdot)$ returns the former's output.

Distinguisher $\boldsymbol{D}_0$ returns the bitwise complement of its input and $\boldsymbol{D}_1$ computes the identity function.

We claim that Equation (5.2) holds for all $k \in \mathbb{N}$. To prove this, fix $k \in \mathbb{N}$. We state four claims, conclude the proof given them, and then return to prove the claims. The first two claims relate the probability that $\boldsymbol{X}$ guesses the value of challenge bit b, in each of its experiments, to the probability that distinguisher $\boldsymbol{D}_b$ returns 1 when it is run on the output of $\boldsymbol{C}$ after the latter has interacted with the decryption oracle, in experiment $\mathbf{Exp}^{\text{pa2-d}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P}_b,\boldsymbol{D}_b}(k)$.

Claim 5.4.10 $\Pr\left[\,\mathbf{Exp}^{\text{ind-cca2-1}}_{\mathcal{AE},\boldsymbol{X}}(k) = 1\,\right] \;=\; \Pr\left[\,\mathbf{Exp}^{\text{pa2-d}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P}_1,\boldsymbol{D}_1}(k) = 1\,\right]$. ∎

Claim 5.4.11 $\Pr\left[\,\mathbf{Exp}^{\text{ind-cca2-0}}_{\mathcal{AE},\boldsymbol{X}}(k) = 1\,\right] \;=\; 1 - \Pr\left[\,\mathbf{Exp}^{\text{pa2-d}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P}_0,\boldsymbol{D}_0}(k) = 1\,\right]$. ∎

The other claims relate the probability that $\boldsymbol{Y}$ guesses the value of challenge bit b, in each of its experiments, to the probability that distinguisher $\boldsymbol{D}_b$ returns 1 when it is run on the output of $\boldsymbol{C}$ after the latter has interacted with pa2-extractor $\boldsymbol{C}^*$, in experiment $\mathbf{Exp}^{\text{pa2-x}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P}_b,\boldsymbol{D}_b,\boldsymbol{C}^*}(k)$.

Claim 5.4.12 $\Pr\left[\,\mathbf{Exp}^{\text{ind-cpa-1}}_{\mathcal{AE},\boldsymbol{Y}}(k) = 1\,\right] \;=\; \Pr\left[\,\mathbf{Exp}^{\text{pa2-x}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P}_1,\boldsymbol{D}_1,\boldsymbol{C}^*}(k) = 1\,\right]$. ∎

Claim 5.4.13 $\Pr\left[\,\mathbf{Exp}^{\text{ind-cpa-0}}_{\mathcal{AE},\boldsymbol{Y}}(k) = 1\,\right] = 1 - \Pr\left[\,\mathbf{Exp}^{\text{pa2-x}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P}_0,\boldsymbol{D}_0,\boldsymbol{C}^*}(k) = 1\,\right]$. ∎

Applying these claims, we obtain Equation (5.2) as follows:

$$
\begin{aligned}
&\mathbf{Adv}^{\text{ind-cca2}}_{\mathcal{AE},\boldsymbol{X}}(k) \\
&\quad= \Pr\left[\,\mathbf{Exp}^{\text{ind-cca2-1}}_{\mathcal{AE},\boldsymbol{X}}(k) = 1\,\right] - \Pr\left[\,\mathbf{Exp}^{\text{ind-cca2-0}}_{\mathcal{AE},\boldsymbol{X}}(k) = 1\,\right] \\
&\quad= \Pr\left[\,\mathbf{Exp}^{\text{pa2-d}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P}_1,\boldsymbol{D}_1}(k) = 1\,\right] - \left(1 - \Pr\left[\,\mathbf{Exp}^{\text{pa2-d}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P}_0,\boldsymbol{D}_0}(k) = 1\,\right]\right) \\
&\quad= \left(\Pr\left[\,\mathbf{Exp}^{\text{pa2-x}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P}_1,\boldsymbol{D}_1,\boldsymbol{C}^*}(k) = 1\,\right] + \mathbf{Adv}^{\text{pa2}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P}_1,\boldsymbol{D}_1,\boldsymbol{C}^*}(k)\right) - 1 \\
&\qquad + \left(\Pr\left[\,\mathbf{Exp}^{\text{pa2-x}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P}_0,\boldsymbol{D}_0,\boldsymbol{C}^*}(k) = 1\,\right] + \mathbf{Adv}^{\text{pa2}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P}_0,\boldsymbol{D}_0,\boldsymbol{C}^*}(k)\right) \\
&\quad= \Pr\left[\,\mathbf{Exp}^{\text{pa2-x}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P}_1,\boldsymbol{D}_1,\boldsymbol{C}^*}(k) = 1\,\right] - \left(1 - \Pr\left[\,\mathbf{Exp}^{\text{pa2-x}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P}_0,\boldsymbol{D}_0,\boldsymbol{C}^*}(k) = 1\,\right]\right) \\
&\qquad + \mathbf{Adv}^{\text{pa2}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P}_0,\boldsymbol{D}_0,\boldsymbol{C}^*}(k) + \mathbf{Adv}^{\text{pa2}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P}_1,\boldsymbol{D}_1,\boldsymbol{C}^*}(k) \\
&\quad= \Pr\left[\,\mathbf{Exp}^{\text{ind-cpa-1}}_{\mathcal{AE},\boldsymbol{Y}}(k) = 1\,\right] - \Pr\left[\,\mathbf{Exp}^{\text{ind-cpa-0}}_{\mathcal{AE},\boldsymbol{Y}}(k) = 1\,\right]
\end{aligned}
$$

$$
\begin{aligned}
&+ \mathbf{Adv}^{\text{pa2}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P}_0,\boldsymbol{D}_0,\boldsymbol{C}^*}(k) + \mathbf{Adv}^{\text{pa2}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P}_1,\boldsymbol{D}_1,\boldsymbol{C}^*}(k) \\
= \;& \mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{AE},\boldsymbol{Y}}(k) + \mathbf{Adv}^{\text{pa2}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P}_0,\boldsymbol{D}_0,\boldsymbol{C}^*}(k) + \mathbf{Adv}^{\text{pa2}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P}_1,\boldsymbol{D}_1,\boldsymbol{C}^*}(k) \ .
\end{aligned}
$$

It remains to prove the four claims above.

Proof of Claim 5.4.10: Let $\mathcal{S}_{\text{cca2-1}}$ denote the sample space underlying $\mathbf{Exp}^{\text{ind-cca2-1}}_{\mathcal{AE},\boldsymbol{X}}(k)$. A member of this space is a string specifying the coins of the key-generation algorithm, the random tape of $\boldsymbol{X}$, and the coins used by the encryption algorithm.

A member of the sample space $\mathcal{S}_{\text{pa2-d-1}}$ underlying $\mathbf{Exp}^{\text{pa2-d}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P}_1,\boldsymbol{D}_1}(k)$ is a string specifying the coins of the key-generation algorithm, the random tape of $\boldsymbol{C}$, the random tape of $\boldsymbol{P}_1$, the coins used by the encryption algorithm across its invocations, and the random tape of $\boldsymbol{D}_1$. Claim 5.4.10 follows once we observe that by the definitions of $\boldsymbol{C}$, $\boldsymbol{P}_1$ and $\boldsymbol{D}_1$, $\mathcal{S}_{\text{pa2-d-1}}$ is equal to $\mathcal{S}_{\text{cca2-1}}$ (The random tape of $\boldsymbol{C}$ consists of coins for $\boldsymbol{X}$; $\boldsymbol{P}_1$ and $\boldsymbol{D}_1$ are deterministic, so their random tapes have length 0; and in $\mathbf{Exp}^{\text{pa2-d}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P}_1,\boldsymbol{D}_1}(k)$, the encryption algorithm is invoked once.), and $\mathbf{Exp}^{\text{ind-cca2-1}}_{\mathcal{AE},\boldsymbol{X}}(k) = 1$ if and only if $\mathbf{Exp}^{\text{pa2-d}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P}_1,\boldsymbol{D}_1}(k) = 1$. ▮

Proof of Claim 5.4.11: Similarly to the proof of Claim 5.4.10, it is easy to see that the sample space underlying $\mathbf{Exp}^{\text{ind-cca2-0}}_{\mathcal{AE},\boldsymbol{X}}(k)$ is identical to the sample space underlying $\mathbf{Exp}^{\text{pa2-d}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P}_0,\boldsymbol{D}_0}(k)$, and $\mathbf{Exp}^{\text{ind-cca2-0}}_{\mathcal{AE},\boldsymbol{X}}(k) = 1$ if and only if $\mathbf{Exp}^{\text{pa2-d}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P}_0,\boldsymbol{D}_0}(k) = 0$. ▮

Proof of Claim 5.4.12: Let $\mathcal{S}_{\text{pa2-x-1}}$ denote the sample space underlying $\mathbf{Exp}^{\text{pa2-x}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P}_1,\boldsymbol{D}_1,\boldsymbol{C}^*}(k)$. A member of this space is a string specifying the coins of the key-generation algorithm, the random tape of $\boldsymbol{C}$, the random tape of $\boldsymbol{P}_1$, the random tape of $\boldsymbol{C}^*$, the coins used by the encryption algorithm across its invocations, and the random tape of $\boldsymbol{D}_1$. We observe that $\boldsymbol{P}_1$ and $\boldsymbol{D}_1$ are deterministic, so their random tapes have length 0, and that in $\mathbf{Exp}^{\text{pa2-x}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P}_1,\boldsymbol{D}_1,\boldsymbol{C}^*}(k)$, the encryption algorithm is invoked once.

A member of the sample space $\mathcal{S}_{\text{cpa-1}}$ underlying $\mathbf{Exp}^{\text{ind-cpa-1}}_{\mathcal{AE},\boldsymbol{Y}}(k)$ is a string specify-

ing the coins of the key-generation algorithm, the random tape of $\boldsymbol{Y}$, and the coins used by the encryption algorithm. The random tape of $\boldsymbol{Y}$ consists of coins for $\boldsymbol{C}$ and coins for $\boldsymbol{C}^*$. Hence $\mathcal{S}_{\text{cpa-1}} = \mathcal{S}_{\text{pa2-x-1}}$. We observe that $\mathbf{Exp}^{\text{ind-cpa-1}}_{\mathcal{AE},\boldsymbol{Y}}(k) = 1$ if and only if $\mathbf{Exp}^{\text{pa2-x}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P}_1,\boldsymbol{D}_1,\boldsymbol{C}^*}(k) = 1$. Claim 5.4.12 follows. ▌

Proof of Claim 5.4.13: Similarly to the proof of Claim 5.4.12, it is easy to see that the sample space underlying $\mathbf{Exp}^{\text{ind-cpa-0}}_{\mathcal{AE},\boldsymbol{Y}}(k)$ is identical to the sample space underlying $\mathbf{Exp}^{\text{pa2-x}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P}_0,\boldsymbol{D}_0,\boldsymbol{C}^*}(k)$, and $\mathbf{Exp}^{\text{ind-cpa-0}}_{\mathcal{AE},\boldsymbol{Y}}(k) = 1$ if and only if $\mathbf{Exp}^{\text{pa2-x}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P}_0,\boldsymbol{D}_0,\boldsymbol{C}^*}(k) = 0$. ▌

5.4.3 Proof of Theorem 5.4.3

Let $\mathcal{AE}' = (\mathcal{K}', \mathcal{E}', \mathcal{D}')$ be an IND-CCA2-secure asymmetric encryption scheme. We construct an IND-CCA2-secure asymmetric encryption scheme $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ that is not PA0-secure. Let $f : \{0,1\}^* \to \{0,1\}^*$ be a length preserving one-way function. (This exists assuming IND-CCA2-secure asymmetric encryption schemes exist.) The algorithms constituting $\mathcal{AE}$ are defined as follows:

Algorithm $\mathcal{K}(1^k)$
  $(pk', sk') \xleftarrow{\$} \mathcal{K}'(1^k)$
  $u \xleftarrow{\$} \{0,1\}^k$; $U \leftarrow f(u)$
  $pk \leftarrow (pk', U)$
  $sk \leftarrow (sk', u)$
  Return (pk, sk)

Algorithm $\mathcal{E}_{pk}(x)$
  Parse pk as (pk', U)
  Return $(0, \mathcal{E}'_{pk'}(x))$

Algorithm $\mathcal{D}_{sk}(y)$
  Parse sk as (sk', u)
  Parse y as (v, y')
  If $v = 0$ then
  return $\mathcal{D}'_{sk'}(y')$ EndIf
  If $v = 1$ then
    If $y' = f(u)$ then
      return u
    else return $\perp$ EndIf
  EndIf

To prove that $\mathcal{AE}$ is not PA0-secure, we proceed by contradiction. Assume that $\mathcal{AE}$ is PA0-secure and consider the pa0 ciphertext creator $\boldsymbol{C}$ depicted in Figure 5.7. Notice that $\boldsymbol{C}$ is deterministic and it runs in polynomial time. Let $\boldsymbol{C}^*$ be a successful polynomial-time pa0-extractor for it. We define a polynomial-time distinguisher $\boldsymbol{D}$ for $\boldsymbol{C}$ and a polynomial-time inverter $\boldsymbol{I}$ for function f as shown

Ciphertext creator $\boldsymbol{C}(pk; R[\boldsymbol{C}])$
Parse pk as (pk', U) ; Make query $(1, U)$; Upon receiving a response u', Return (pk', U, u')

Distinguisher $\boldsymbol{D}(x)$
Parse x as (pk', U, u') ; If $f(u') = U$ then return 1 else return 0 EndIf

---

Inverter $\boldsymbol{I}(U)$
$k \leftarrow |U|$; $(pk', sk') \xleftarrow{\$} \mathcal{K}'(1^k)$; $pk \leftarrow (pk', U)$
Choose coins $R[\boldsymbol{C}^*]$ for $\boldsymbol{C}^*$; $St[\boldsymbol{C}^*] \leftarrow (pk, \varepsilon)$
$(u', St[\boldsymbol{C}^*]) \leftarrow \boldsymbol{C}^*((1, U), St[\boldsymbol{C}^*]; R[\boldsymbol{C}^*])$
Return u'

Figure 5.7 Ciphertext-creator adversary $\boldsymbol{C}$, distinguisher $\boldsymbol{D}$, and inverter $\boldsymbol{I}$ for the proof of Theorem 5.4.3.

---

in Figure 5.7. Fix $k \in \mathbb{N}$. The probability that $\boldsymbol{I}$ is successful can be computed as follows.

$$\begin{aligned}
&\Pr\left[\, U \leftarrow \{0,1\}^k \,;\, u' \xleftarrow{\$} \boldsymbol{I}(U) \;\;:\;\; f(u') = U \,\right] \\
&\quad = \;\Pr\left[\, \mathbf{Exp}^{\text{pa1-x}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D},\boldsymbol{C}^*}(k) = 1 \,\right] \\
&\quad = \;\Pr\left[\, \mathbf{Exp}^{\text{pa1-d}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D}}(k) = 1 \,\right] - \mathbf{Adv}^{\text{pa1}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D},\boldsymbol{C}^*}(k) \\
&\quad = \;1 - \mathbf{Adv}^{\text{pa1}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D},\boldsymbol{C}^*}(k) \;.
\end{aligned}$$

Since $\boldsymbol{C}^*$ is a successful pa0-extractor, the function $\mathbf{Adv}^{\text{pa1}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D},\boldsymbol{C}^*}$ is negligible and hence the probability of success of $\boldsymbol{I}$ is not negligible. This contradicts the one-wayness of f, as desired.

We proceed to prove that $\mathcal{AE}$ is IND-CCA2-secure. Let $\boldsymbol{X}$ be an ind-cca2-adversary attacking $\mathcal{AE}$. We define an ind-cca2-adversary $\boldsymbol{X}'$ attacking $\mathcal{AE}'$ as depicted in Figure 5.8. A random tape $u \| R[\boldsymbol{X}]$ for adversary $\boldsymbol{X}'$ has two parts. The first part is a k-bit string that $\boldsymbol{X}'$ uses to reply to $\boldsymbol{X}$'s queries of the form $(1, f(u))$. (The answer to such a query in experiment $\mathbf{Exp}^{\text{ind-cca2-}b}_{\mathcal{AE},\boldsymbol{X}}(k)$ would be the randomly chosen k-bit string u that corresponds to the second component of the secret key sk.) The second part is a random tape for $\boldsymbol{X}$. $\boldsymbol{X}'$ runs $\boldsymbol{X}$ and

Adversary $\boldsymbol{X}'(\mathsf{find}, pk'; R[\boldsymbol{X}'])$
Parse $R[\boldsymbol{X}']$ as $u \| R[\boldsymbol{X}]$, where $u \in \{0,1\}^k$; $U \leftarrow f(u)$; $pk \leftarrow (pk', U)$
Run $\boldsymbol{X}(\mathsf{find}, \cdot)$ on input pk and coins $R[\boldsymbol{X}]$ until it halts, replying to its oracle queries as follows:
– If $\boldsymbol{X}(\mathsf{find}, \cdot)$ makes query (v, y') then
If $v = 0$ then
Make query y' ; Upon receiving a response M, return M to $\boldsymbol{X}(\mathsf{find}, \cdot)$ as the reply EndIf
If $v = 1$ then
If $y' = U$ then return u to $\boldsymbol{X}(\mathsf{find}, \cdot)$ as the reply
else return $\perp$ to $\boldsymbol{X}(\mathsf{find}, \cdot)$ as the reply EndIf EndIf EndIf
Let (M_0, M_1, St) denote the output of $\boldsymbol{X}(\mathsf{find}, \cdot)$
$St' \leftarrow (St, u, U)$
Return (M_0, M_1, St')

Adversary $\boldsymbol{X}'(\mathsf{guess}, C', St')$
Parse St' as (St, u, U) ; $C \leftarrow (0, C')$
Run $\boldsymbol{X}(\mathsf{guess}, \cdot)$ on input C, St until it halts, replying to its oracle queries as follows:
– If $\boldsymbol{X}(\mathsf{guess}, \cdot)$ makes query (v, y') then
If $v = 0$ then
Make query y' ; Upon receiving a response M, return M to $\boldsymbol{X}(\mathsf{guess}, \cdot)$ as the reply EndIf
If $v = 1$ then
If $y' = U$ then return u to $\boldsymbol{X}(\mathsf{guess}, \cdot)$ as the reply
else return $\perp$ to $\boldsymbol{X}(\mathsf{guess}, \cdot)$ as the reply EndIf EndIf EndIf
Let d denote the output of $\boldsymbol{X}(\mathsf{guess}, \cdot)$
Return d

Figure 5.8 Ind-cca2-adversary $\boldsymbol{X}'$ for the proof of Theorem 5.4.3.

---

uses its decryption oracle and the value u to reply to the oracle queries of the latter. Clearly, $\boldsymbol{X}'$ runs in polynomial time. Furthermore, for $b \in \{0,1\}$ and $k \in \mathbb{N}$, the replies that experiment $\mathbf{Exp}^{\text{ind-cca2-}b}_{\mathcal{AE}', \boldsymbol{X}'}(k)$ computes to $\boldsymbol{X}'$'s queries allow this adversary to respond to $\boldsymbol{X}$'s queries exactly as experiment $\mathbf{Exp}^{\text{ind-cca2-}b}_{\mathcal{AE}, \boldsymbol{X}}(k)$ does. Therefore, $\mathbf{Adv}^{\text{ind-cca2}}_{\mathcal{AE}', \boldsymbol{X}'}(k) = \mathbf{Adv}^{\text{ind-cca2}}_{\mathcal{AE}, \boldsymbol{X}}(k)$. The assumption that $\mathcal{AE}'$ is IND-CCA2-secure implies that the function $\mathbf{Adv}^{\text{ind-cca2}}_{\mathcal{AE}', \boldsymbol{X}'}$ is negligible, and thus the

function $\mathbf{Adv}^{\text{ind-cca2}}_{\mathcal{AE},\boldsymbol{X}}$ is negligible. Hence $\mathcal{AE}$ is IND-CCA2-secure.

5.4.4 Proof of Theorem 5.4.4

Let $\mathcal{AE}' = (\mathcal{K}, \mathcal{E}', \mathcal{D}')$ be a PA1-secure and IND-CPA-secure asymmetric encryption scheme. We construct a PA1-secure and IND-CPA-secure asymmetric encryption scheme $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ that is not IND-CCA2-secure. Notice that the key-generation algorithm is the same. The encryption and decryption algorithms are defined as follows:

Algorithm $\mathcal{E}_{pk}(x)$	Algorithm $\mathcal{D}_{sk}(y)$
$r \xleftarrow{\$} \{0,1\}$	Parse y as (r, y') ; $x \leftarrow \mathcal{D}'_{sk}(y')$
Return $(r, \mathcal{E}'_{pk}(x))$	Return x

To prove that $\mathcal{AE}$ is not IND-CCA2-secure, we define an ind-cca2-adversary $\boldsymbol{X}$ attacking $\mathcal{AE}$ as shown in Figure 5.9. Clearly, $\boldsymbol{X}$ runs in polynomial time and $\mathbf{Adv}^{\text{ind-cca2}}_{\mathcal{AE},\boldsymbol{X}}(k) = 1$ for every $k \in \mathbb{N}$.

To prove that $\mathcal{AE}$ is PA1-secure, let $\boldsymbol{C}$ be a polynomial-time ciphertext creator attacking $\mathcal{AE}$. We define a ciphertext creator $\boldsymbol{C}'$ attacking $\mathcal{AE}'$ as shown in Figure 5.9. Clearly, $\boldsymbol{C}'$ runs in polynomial time. By the assumption that $\mathcal{AE}'$ is PA1-secure, there is a successful polynomial-time pa1-extractor $\boldsymbol{C}'^{*}$ for $\boldsymbol{C}'$. We construct a pa1-extractor $\boldsymbol{C}^{*}$ for $\boldsymbol{C}$, based on $\boldsymbol{C}'^{*}$ as shown in Figure 5.9. It is clear that $\boldsymbol{C}^{*}$ runs in polynomial time. Let $\boldsymbol{D}$ be a polynomial-time distinguisher for $\boldsymbol{C}$, and fix $k \in \mathbb{N}$. It is easy to see that

$$\begin{aligned} \Pr\left[\mathbf{Exp}^{\text{pa1-d}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D}}(k) = 1\right] &= \Pr\left[\mathbf{Exp}^{\text{pa1-d}}_{\mathcal{AE}',\boldsymbol{C}',\boldsymbol{D}}(k) = 1\right] \quad \text{and} \\ \Pr\left[\mathbf{Exp}^{\text{pa1-x}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D},\boldsymbol{C}^{*}}(k) = 1\right] &= \Pr\left[\mathbf{Exp}^{\text{pa1-x}}_{\mathcal{AE}',\boldsymbol{C}',\boldsymbol{D},\boldsymbol{C}'^{*}}(k) = 1\right]. \end{aligned}$$

Therefore,

$$\mathbf{Adv}^{\text{pa1}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D},\boldsymbol{C}^{*}}(k) = \mathbf{Adv}^{\text{pa1}}_{\mathcal{AE}',\boldsymbol{C}',\boldsymbol{D},\boldsymbol{C}'^{*}}(k).$$

Since $\boldsymbol{C}'^{*}$ is a successful pa1-extractor for $\boldsymbol{C}'$, for every polynomial-time distinguisher $\boldsymbol{D}$, the function $\mathbf{Adv}^{\text{pa1}}_{\mathcal{AE}',\boldsymbol{C}',\boldsymbol{D},\boldsymbol{C}'^{*}}$ is negligible and hence for every

Adversary $\boldsymbol{X}(\mathsf{find}, pk; R[\boldsymbol{X}])$	Adversary $\boldsymbol{X}(\mathsf{guess}, C, St)$
Return $(0, 1, \varepsilon)$	Parse C as (r, C') Make query $(\bar{r}, C')$ and let M denote the response If $M = 0$ then $d \leftarrow 0$ else $d \leftarrow 1$ EndIf Return d

Ciphertext creator $\boldsymbol{C}'(pk; R[\boldsymbol{C}'])$
Run $\boldsymbol{C}$ on input pk and coins $R[\boldsymbol{C}']$ until it halts, replying to its oracle queries as follows:
– If $\boldsymbol{C}$ makes query (r, y') then
Make query y'; Upon receiving a response M, return M to $\boldsymbol{C}$ as the reply EndIf
Let x denote the output of $\boldsymbol{C}$; Return x

Pa1-extractor $\boldsymbol{C}^*(Q, St[\boldsymbol{C}^*]; R[\boldsymbol{C}^*])$
Parse Q as (r, y'); $(M, St[\boldsymbol{C}'^*]) \leftarrow \boldsymbol{C}'^*(y', St[\boldsymbol{C}^*]; R[\boldsymbol{C}^*])$
Return $(M, St[\boldsymbol{C}'^*])$

Adversary $\boldsymbol{Y}'(\mathsf{find}, pk; R[\boldsymbol{Y}'])$	Adversary $\boldsymbol{Y}'(\mathsf{guess}, C', St')$
Parse $R[\boldsymbol{Y}']$ as $r \| R[\boldsymbol{Y}]$, where $r \in \{0,1\}$ $(M_0, M_1, St) \xleftarrow{\$} \boldsymbol{Y}(\mathsf{find}, pk; R[\boldsymbol{Y}])$ $St' \leftarrow (St, r)$ Return (M_0, M_1, St')	Parse St' as (St, r) $C \leftarrow (r, C')$ $d \leftarrow \boldsymbol{Y}(\mathsf{guess}, C, St)$ Return d

Figure 5.9 Ind-cca2-adversary $\boldsymbol{X}$, ciphertext-creator adversary $\boldsymbol{C}'$, pa1-extractor $\boldsymbol{C}^*$, and ind-cpa-adversary $\boldsymbol{Y}'$ for the proof of Theorem 5.4.4.

polynomial-time distinguisher $\boldsymbol{D}$, the function $\mathbf{Adv}^{\mathrm{pa1}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D},\boldsymbol{C}^*}$ is negligible. Thus $\boldsymbol{C}^*$ is a successful pa1-extractor for $\boldsymbol{C}$, and $\mathcal{AE}$ is PA1-secure.

To prove that $\mathcal{AE}$ is IND-CPA-secure, let $\boldsymbol{Y}$ be an ind-cpa-adversary attacking $\mathcal{AE}$. Consider the ind-cpa-adversary $\boldsymbol{Y}'$ attacking $\mathcal{AE}'$ depicted in Figure 5.9. A random tape $r \| R[\boldsymbol{Y}]$ for adversary $\boldsymbol{Y}'$ has two parts. The first part is a bit that $\boldsymbol{Y}'$ uses to compute the challenge ciphertext C for $\boldsymbol{Y}$. The second part is a random tape for $\boldsymbol{Y}$. $\boldsymbol{Y}'$ runs $\boldsymbol{Y}$ and returns the output of the latter. Clearly, $\boldsymbol{Y}'$ runs in polynomial time and $\mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{AE}',\boldsymbol{Y}'}(k) = \mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{AE},\boldsymbol{Y}}(k)$ for every $k \in \mathbb{N}$. The

assumption that $\mathcal{AE}'$ is IND-CPA-secure implies that the ind-cpa-advantage of $\boldsymbol{Y}'$ is negligible, and hence it follows that the ind-cpa-advantage of $\boldsymbol{Y}$ is negligible. Thus $\mathcal{AE}$ is IND-CPA-secure.

5.4.5 Proof of Theorem 5.4.5

Let $\mathcal{AE}' = (\mathcal{K}, \mathcal{E}', \mathcal{D}')$ be a PA0-secure and IND-CPA-secure asymmetric encryption scheme. We construct a PA0-secure and IND-CPA-secure asymmetric encryption scheme $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ that is not IND-CCA1-secure. Its constituent algorithms are defined as follows:

Algorithm $\mathcal{K}(1^k)$	Algorithm $\mathcal{E}_{pk}(x)$	Algorithm $\mathcal{D}_{sk}(y)$
$(pk', sk') \xleftarrow{\$} \mathcal{K}'(1^k)$	Return $(0, \mathcal{E}'_{pk}(x))$	Parse sk as (sk', u)
$u \xleftarrow{\$} \{0,1\}^{\lvert sk' \rvert}$		Parse y as (v, y')
$sk \leftarrow (sk', u)$		If $v = 0$ then
Return (pk', sk)		return $\mathcal{D}'_{sk'}(y')$ EndIf
		If $v = 1$ then
		If $y' = 0$ then
		return u
		else return $u \oplus sk'$ EndIf
		EndIf

To prove that $\mathcal{AE}$ is not IND-CCA1-secure, we define an ind-cca1-adversary $\boldsymbol{X}$ attacking $\mathcal{AE}$ as shown in Figure 5.10. Clearly, $\boldsymbol{X}$ runs in polynomial time and $\mathbf{Adv}^{\text{ind-cca1}}_{\mathcal{AE},\boldsymbol{X}}(k) = 1$ for every $k \in \mathbb{N}$.

To prove that $\mathcal{AE}$ is PA0-secure, let $\boldsymbol{C}$ be a polynomial-time pa0 ciphertext creator attacking $\mathcal{AE}$. We define a pa0 ciphertext creator $\boldsymbol{C}'$ attacking $\mathcal{AE}'$ as shown in Figure 5.10. Clearly, $\boldsymbol{C}'$ runs in polynomial time. By the assumption that $\mathcal{AE}'$ is PA0-secure, there is a successful polynomial-time pa0-extractor $\boldsymbol{C}'^*$ for $\boldsymbol{C}'$. We construct a pa0-extractor $\boldsymbol{C}^*$ for $\boldsymbol{C}$, based on $\boldsymbol{C}'^*$ as shown in Figure 5.10. It is clear that $\boldsymbol{C}^*$ runs in polynomial time. Let $\boldsymbol{D}$ be a polynomial-time distinguisher for $\boldsymbol{C}$, and fix $k \in \mathbb{N}$. It is easy to see that

$$\begin{aligned} \Pr\left[\mathbf{Exp}^{\text{pa1-d}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D}}(k) = 1\right] &= \Pr\left[\mathbf{Exp}^{\text{pa1-d}}_{\mathcal{AE}',\boldsymbol{C}',\boldsymbol{D}}(k) = 1\right] \quad \text{and} \\ \Pr\left[\mathbf{Exp}^{\text{pa1-x}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D},\boldsymbol{C}^*}(k) = 1\right] &= \Pr\left[\mathbf{Exp}^{\text{pa1-x}}_{\mathcal{AE}',\boldsymbol{C}',\boldsymbol{D},\boldsymbol{C}'^*}(k) = 1\right]. \end{aligned}$$

Adversary $\boldsymbol{X}(\mathsf{find}, pk; R[\boldsymbol{X}])$
Make query $(1,0)$ and let M_0 denote the response
Make query $(1,1)$ and let M_1 denote the response
$sk' \leftarrow M_0 \oplus M_1$; Return $(0,1,sk')$

Adversary $\boldsymbol{X}(\mathsf{guess}, C, St)$
Parse C as $(0, C')$
If $\mathcal{D}'_{St}(C') = 0$ then $d \leftarrow 0$
else $d \leftarrow 1$ EndIf
Return d

Ciphertext creator $\boldsymbol{C}'(pk; R[\boldsymbol{C}'])$
Parse $R[\boldsymbol{C}']$ as $u \| R[\boldsymbol{C}]$
Run $\boldsymbol{C}$ on input pk and coins $R[\boldsymbol{C}]$ until it halts, replying to its oracle query as follows:
– When $\boldsymbol{C}$ makes query (v, y') do
Make query y' ; Let M denote the response
If $v = 1$ then $M \leftarrow u$ EndIf
Return M to $\boldsymbol{C}$ as the reply
Let x denote the output of $\boldsymbol{C}$; Return x

Pa0-extractor $\boldsymbol{C}^*(Q, St[\boldsymbol{C}^*]; R[\boldsymbol{C}^*])$
Parse $St[\boldsymbol{C}^*]$ as $(pk, R[\boldsymbol{C}])$; Parse $R[\boldsymbol{C}^*]$ as $u \| R[\boldsymbol{C}'^*]$; Parse Q as (v, y')
$St[\boldsymbol{C}'^*] \leftarrow (pk, u \| R[\boldsymbol{C}])$; $(M, St[\boldsymbol{C}'^*]) \leftarrow \boldsymbol{C}'^*(y', St[\boldsymbol{C}'^*]; R[\boldsymbol{C}'^*])$
If $v = 1$ then $M \leftarrow u$ EndIf
Return $(M, St[\boldsymbol{C}'^*])$

Adversary $\boldsymbol{Y}'(\mathsf{find}, pk; R[\boldsymbol{Y}'])$
$(M_0, M_1, St) \xleftarrow{\$} \boldsymbol{Y}(\mathsf{find}, pk; R[\boldsymbol{Y}'])$
Return (M_0, M_1, St)

Adversary $\boldsymbol{Y}'(\mathsf{guess}, C', St)$
$C \leftarrow (0, C')$; $d \leftarrow \boldsymbol{Y}(\mathsf{guess}, C, St)$
Return d

Figure 5.10 Ind-cca1-adversary $\boldsymbol{X}$, pa0 ciphertext-creator adversary $\boldsymbol{C}'$, pa0-extractor $\boldsymbol{C}^*$, and ind-cpa-adversary $\boldsymbol{Y}'$ for the proof of Theorem 5.4.5.

Therefore,

$$\mathbf{Adv}^{\mathrm{pa1}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D},\boldsymbol{C}^*}(k) \;=\; \mathbf{Adv}^{\mathrm{pa1}}_{\mathcal{AE}',\boldsymbol{C}',\boldsymbol{D},\boldsymbol{C}'^*}(k)\ .$$

Since $\boldsymbol{C}'^*$ is a successful pa0-extractor for $\boldsymbol{C}'$, for every polynomial-time distinguisher $\boldsymbol{D}$, the function $\mathbf{Adv}^{\mathrm{pa1}}_{\mathcal{AE}',\boldsymbol{C}',\boldsymbol{D},\boldsymbol{C}'^*}$ is negligible and hence for every polynomial-time distinguisher $\boldsymbol{D}$, the function $\mathbf{Adv}^{\mathrm{pa1}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D},\boldsymbol{C}^*}$ is negligible. Thus $\boldsymbol{C}^*$ is a successful pa0-extractor for $\boldsymbol{C}$, and $\mathcal{AE}$ is PA0-secure.

To prove that $\mathcal{AE}$ is IND-CPA-secure, let $\boldsymbol{Y}$ be an ind-cpa-adversary attacking $\mathcal{AE}$. Consider the ind-cpa-adversary $\boldsymbol{Y}'$ attacking $\mathcal{AE}'$ depicted in Figure 5.10. Clearly, $\boldsymbol{Y}'$ runs in polynomial time and $\mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{AE}',\boldsymbol{Y}'}(k) = \mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{AE},\boldsymbol{Y}}(k)$ for every $k \in \mathbb{N}$. The assumption that $\mathcal{AE}'$ is IND-CPA-secure implies that the ind-cpa-advantage of $\boldsymbol{Y}'$ is negligible, and hence it follows that the ind-cpa-advantage of $\boldsymbol{Y}$ is negligible. Thus $\mathcal{AE}$ is IND-CPA-secure.

5.5 Constructions

5.5.1 Approaches

Before presenting our results, we discuss some possible approaches to designing PA encryption schemes. One natural approach is based on the use of non-interactive zero-knowledge proofs of knowledge (NIZK-POKs) [78]. In particular, a candidate construction is the following. Let the public key have the form (pk, R) where pk is the public key of some IND-CPA-secure (or even IND-CCA2-secure) "base" encryption scheme and R is a random reference string. Encryption consists of providing an encryption of the message under pk via the base scheme, together with a NIZK-POK of the message relative to reference string R. However, this type of approach fails to yield even the weakest form of PA. The problem is that the PA extractor must work with the given public key of the ciphertext creator, and hence a given reference string, while the NIZK-POK extractor that one would hope to use to define the PA extractor, creates a simulated reference string with accompanying trapdoor.

This might lead one to ask why our definition of PA is not relaxed to allow the extractor to choose or simulate the public key rather than having to work with the given one. Besides the fact that the intuition captured is quite different, it is not clear how to make such a relaxation while preserving the PA1 + IND-CPA $\rightarrow$ IND-CCA1 and PA2 + IND-CPA $\rightarrow$ IND-CCA2 implications of Theorems 5.4.1

and 5.4.2. (In particular, if we allow the extractor to simply choose a public key, it can choose one whose corresponding secret key it knows, making the notion trivial to achieve and making the implications fail.)

As such approaches are considered and discarded, it becomes apparent that achieving even the weaker forms of PA under standard assumptions may be difficult. We have been able to make progress, however, under some strong assumptions, as we now describe.

5.5.2 Prime-order groups

If p, q are primes such that $p = 2q + 1$, then we let G_q denote the subgroup of quadratic residues of $\mathbb{Z}_p^*$. Recall this is a cyclic subgroup of order q. If g is a generator of G_q then $\mathrm{dlog}_{q,g}(X)$ denotes the discrete logarithm of $X \in G_q$ to base g. A *prime-order-group generator* is a polynomial-time algorithm G that on input 1^k returns a triple (p, q, g) such that p, q are primes with $p = 2q + 1$, g is a generator of G_q, and $2^{k-1} < p < 2^k$ (p is k bits long).

5.5.3 The DHK assumptions

Let G be a prime-order-group generator, and suppose $(p, q, g) \in [\mathsf{G}(1^k)]$. We say that (A, B, W) is a *DH-triple* if there exist $a, b \in Z_q$ such that $A = g^a \bmod p$, $B = g^b \bmod p$ and $W = g^{ab} \bmod p$. We say that (B, W) is a *DH-pair relative to* A if (A, B, W) is a DH-triple. One way for an adversary $\boldsymbol{H}$ taking input p, q, g, A to output a DH-pair (B, W) relative to A is to pick —and thus "know"— some $b \in \mathbb{Z}_q$, set $B = g^b \bmod p$ and $W = A^b \bmod p$, and output (B, W). Damgård [36] makes an assumption which, informally, says that this is the "only" way that a polynomial-time adversary $\boldsymbol{H}$ can output a DH-pair relative to A. His framework to capture this requires that there exist a suitable extractor $\boldsymbol{H}^*$ that can compute $\mathrm{dlog}_{q,g}(B)$ whenever $\boldsymbol{H}$ outputs some DH-pair (B, W) relative to A.

We provide a formalization of this assumption that we refer to as the DHK0

Experiment $\mathbf{Exp}^{\text{dhk1}}_{\mathsf{G},\boldsymbol{H},\boldsymbol{H}^*}(k)$
$(p,q,g) \xleftarrow{\$} \mathsf{G}(1^k)$; $a \xleftarrow{\$} \mathbb{Z}_q$; $A \leftarrow g^a \bmod p$
Choose coins $R[\boldsymbol{H}], R[\boldsymbol{H}^*]$ for $\boldsymbol{H}, \boldsymbol{H}^*$, respectively
$St[\boldsymbol{H}^*] \leftarrow ((p,q,g,A), R[\boldsymbol{H}])$
Run $\boldsymbol{H}$ on input p,q,g,A and coins $R[\boldsymbol{H}]$ until it halts, replying to its oracle queries as follows:
– If $\boldsymbol{H}$ makes query (B,W) then
$(b, St[\boldsymbol{H}^*]) \leftarrow \boldsymbol{H}^*((B,W), St[\boldsymbol{H}^*]; R[\boldsymbol{H}^*])$
If $W \equiv B^a \pmod p$ and $B \not\equiv g^b \pmod p$ then return 1
else return b to $\boldsymbol{H}$ as the reply EndIf EndIf
Return 0

Figure 5.11 Experiment used to define DHK1 and DHK0.

(DHK stands for Diffie-Hellman Knowledge) assumption. We also present a natural extension of this assumption that we refer to as DHK1. Here the adversary $\boldsymbol{H}$, given p,q,g,A, interacts with the extractor, querying it adaptively. The extractor is required to be able to return $\text{dlog}_{q,g}(B)$ for each DH-pair (B,W) relative to A that is queried to it. Below we first present the DHK1 assumption, and then define the DHK0 assumption via this.

Assumption 5.5.1 [**DHK1**] Let G be a prime-order-group generator. Let $\boldsymbol{H}$ be an algorithm that has access to an oracle, takes two primes and two group elements, and returns nothing. Let $\boldsymbol{H}^*$ be an algorithm that takes a pair of group elements and some state information, and returns an exponent and a new state. We call $\boldsymbol{H}$ a dhk1-*adversary* and $\boldsymbol{H}^*$ a dhk1-*extractor*. For $k \in \mathbb{N}$ we define the experiment shown in Figure 5.11. The dhk1-*advantage* of $\boldsymbol{H}$ relative to $\boldsymbol{H}^*$ is

$$\mathbf{Adv}^{\text{dhk1}}_{\mathsf{G},\boldsymbol{H},\boldsymbol{H}^*}(k) \;=\; \Pr\left[\,\mathbf{Exp}^{\text{dhk1}}_{\mathsf{G},\boldsymbol{H},\boldsymbol{H}^*}(k) = 1\,\right].$$

We say that G satisfies the DHK1 assumption if for every polynomial-time dhk1-adversary $\boldsymbol{H}$ there exists a polynomial-time dhk1-extractor $\boldsymbol{H}^*$ such that the function $\mathbf{Adv}^{\text{dhk1}}_{\mathsf{G},\boldsymbol{H},\boldsymbol{H}^*}$ is negligible. ▌

Assumption 5.5.2 [**DHK0**] Let G be a prime-order-group generator. We call a dhk1-adversary that makes *exactly one* oracle query a dhk0-*adversary*. We call a dhk1-extractor for a dhk0-adversary a dhk0-*extractor*. We say that G satisfies the Diffie-Hellman Knowledge (DHK0) assumption if for every polynomial-time dhk0-adversary $\boldsymbol{H}$ there exists a polynomial-time dhk0-extractor $\boldsymbol{H}^*$ such that the function $\mathbf{Adv}^{\mathrm{dhk1}}_{\mathsf{G},\boldsymbol{H},\boldsymbol{H}^*}$ is negligible. ▮

We observe that DHK1 implies DHK0 in the sense that if a prime-order-group generator satisfies the former assumption then it also satisfies the latter assumption.

5.5.4 Constructions

We would like to build an asymmetric encryption scheme that is PA0-secure (and IND-CPA-secure) under the DHK0 assumption. An obvious idea is to use ElGamal encryption. Here the public key is $X = g^x$, where x is the secret key, and an encryption of message $M \in G_q$ has the form (Y, U), where $Y = g^y \bmod p$ and $U = X^y \cdot M \bmod p = g^{xy} \cdot M \bmod p$. However, we do not know whether this scheme is PA0-secure. (We can show that it is not sPA0-secure unless the discrete-logarithm problem is easy, but whether or not it is PA0-secure remains open.)

We consider a modification of the ElGamal scheme that was proposed by Damgård [36]. We call this scheme *Damgård ElGamal* or DEG. It is parameterized by a prime-order group generator G, and its components are depicted in Figure 5.12. The proof of the following is in Section 5.5.6:

Theorem 5.5.3 Let G be a prime-order-group generator and let DEG $= (\mathcal{K}, \mathcal{E}, \mathcal{D}, \mathsf{MsgSp})$ be the associated Damgård ElGamal asymmetric encryption scheme defined in Figure 5.12. If G satisfies the DHK0 and DDH assumptions then DEG is PA0 + IND-CPA-secure. If G satisfies the DHK1 and DDH assumptions then DEG is PA1 + IND-CPA-secure. ▮

Algorithm $\mathcal{K}(1^k)$ $(p, q, g) \xleftarrow{\$} \mathsf{G}(1^k)$ $x_1 \xleftarrow{\$} \mathbb{Z}_q$; $X_1 \leftarrow g^{x_1} \bmod p$ $x_2 \xleftarrow{\$} \mathbb{Z}_q$; $X_2 \leftarrow g^{x_2} \bmod p$ Return $((p, q, g, X_1, X_2), (p, q, g, x_1, x_2))$	Algorithm $\mathcal{E}_{(p,q,g,X_1,X_2)}(M)$ $y \xleftarrow{\$} \mathbb{Z}_q$; $Y \leftarrow g^y \bmod p$ $W \leftarrow X_1^y \bmod p$; $V \leftarrow X_2^y \bmod p$ $U \leftarrow V \cdot M \bmod p$ Return (Y, W, U)
Algorithm $\mathcal{D}_{(p,q,g,x_1,x_2)}((Y, W, U))$ If $W \not\equiv Y^{x_1} \pmod p$ then return $\perp$ else $M \leftarrow U \cdot Y^{-x_2} \bmod p$; Return M EndIf	$\mathsf{MsgSp}((p, q, g, X_1, X_2)) = G_q$

Figure 5.12 **Damgård ElGamal (DEG) encryption scheme**. DEG = $(\mathcal{K}, \mathcal{E}, \mathcal{D}, \mathsf{MsgSp})$ is based on prime-order-group generator G.

As a consequence of the above and Theorem 5.4.1, DEG is IND-CCA1-secure under the DHK1 and DDH assumptions. DEG is in fact the most efficient known IND-CCA1-secure scheme with some proof of security in the standard model.

Next we consider the "lite" version of the Cramer-Shoup asymmetric encryption scheme [35]. The scheme, denoted CSL, is parameterized by a prime-order group generator G, and its components are depicted in Figure 5.13. This scheme is known to be IND-CCA1-secure under the DDH assumption [35]. We are able to show the following. (The proof can be found in Section 5.5.7.)

Theorem 5.5.4 Let G be a prime-order-group generator, and let CSL = $(\mathcal{K}, \mathcal{E}, \mathcal{D}, \mathsf{MsgSp})$ be the associated Cramer-Shoup Lite asymmetric encryption scheme defined in Figure 5.13. If G satisfies the DHK0 and DDH assumptions then CSL is PA0 + IND-CPA-secure. If G satisfies the DHK1 and DDH assumptions then CSL is PA1 + IND-CPA-secure. ▮

Again, the above and Theorem 5.4.1 imply that CSL is IND-CCA1-secure under the DHK1 and DDH assumptions. This however is not news, since we already know that DDH alone suffices to prove it IND-CCA1-secure [35]. However, it does perhaps provide a new perspective on why the scheme is IND-CCA1-secure,

Algorithm $\mathcal{K}(1^k)$ $(p, q, g_1) \xleftarrow{\$} \mathsf{G}(1^k)$; $g_2 \xleftarrow{\$} G_q \setminus \{1\}$ $x_1 \xleftarrow{\$} \mathbb{Z}_q$; $x_2 \xleftarrow{\$} \mathbb{Z}_q$; $z \xleftarrow{\$} \mathbb{Z}_q$ $X \leftarrow g_1^{x_1} \cdot g_2^{x_2} \bmod p$; $Z \leftarrow g_1^z \bmod p$ Return $((p, q, g_1, g_2, X, Z), (p, q, g_1, g_2, x_1, x_2, z))$	Algorithm $\mathcal{E}_{(p,q,g_1,g_2,X,Z)}(M)$ $r \xleftarrow{\$} \mathbb{Z}_q$ $R_1 \leftarrow g_1^r \bmod p$ $R_2 \leftarrow g_2^r \bmod p$ $E \leftarrow Z^r \cdot M \bmod p$ $V \leftarrow X^r \bmod p$ Return (R_1, R_2, E, V)
Algorithm $\mathcal{D}_{(p,q,g_1,g_2,x_1,x_2,z)}((R_1, R_2, E, V))$ If $V \not\equiv R_1^{x_1} \cdot R_2^{x_2} \pmod p$ then return $\perp$ else $M \leftarrow E \cdot R_1^{-z} \bmod p$; Return M EndIf	$\mathsf{MsgSp}((p, q, g_1, g_2, X, Z)) = G_q$

Figure 5.13 **Cramer-Shoup Lite (**CSL**) encryption scheme**. CSL $= (\mathcal{K}, \mathcal{E}, \mathcal{D}, \mathsf{MsgSp})$ is based on prime-order-group generator G.

namely that this is due to its possessing some form of plaintext awareness.

In summary, we have been able to show that plaintext awareness without ROs is efficiently achievable, even though under very strong and non-standard assumptions.

5.5.5 A lemma

We first state and prove a lemma that will be used in the proofs of the theorems stated above.

Lemma 5.5.5 Let $\mathcal{AE}$ be an asymmetric encryption scheme. Let $\boldsymbol{C}$ be a polynomial-time ciphertext creator attacking $\mathcal{AE}$, $\boldsymbol{D}$ a polynomial-time distinguisher, and $\boldsymbol{C}^*$ a polynomial-time pa1-extractor. Let DecOK denote the event that all $\boldsymbol{C}^*$'s answers to $\boldsymbol{C}$'s queries are correct in experiment $\mathbf{Exp}^{\text{pa1-x}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D},\boldsymbol{C}^*}(k)$. Then,

$$\Pr\left[\mathbf{Exp}^{\text{pa1-x}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D},\boldsymbol{C}^*}(k) = 1\right] \;\geq\; \Pr\left[\mathbf{Exp}^{\text{pa1-d}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D}}(k) = 1\right] - \Pr\left[\overline{\text{DecOK}}\right]. \quad \blacksquare$$

Proof: We observe that if experiment $\mathbf{Exp}^{\text{pa1-x}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D},\boldsymbol{C}^*}(k)$ returns 0 and event DecOK occurs, then experiment $\mathbf{Exp}^{\text{pa1-d}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D}}(k)$ also returns 0. Therefore,

$$\begin{aligned}
&1 - \Pr\left[\,\mathbf{Exp}^{\text{pa1-x}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D},\boldsymbol{C}^*}(k) = 1\,\right] \\
&\quad = \Pr\left[\,\mathbf{Exp}^{\text{pa1-x}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D},\boldsymbol{C}^*}(k) = 0\,\right] \\
&\quad = \Pr\left[\,\mathbf{Exp}^{\text{pa1-x}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D},\boldsymbol{C}^*}(k) = 0 \wedge \text{DecOK}\,\right] + \\
&\qquad \Pr\left[\,\mathbf{Exp}^{\text{pa1-x}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D},\boldsymbol{C}^*}(k) = 0 \wedge \overline{\text{DecOK}}\,\right] \\
&\quad \leq \Pr\left[\,\mathbf{Exp}^{\text{pa1-d}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D}}(k) = 0\,\right] + \Pr\left[\,\overline{\text{DecOK}}\,\right] \\
&\quad = 1 - \Pr\left[\,\mathbf{Exp}^{\text{pa1-d}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D}}(k) = 1\,\right] + \Pr\left[\,\overline{\text{DecOK}}\,\right]
\end{aligned}$$

Transposing terms and simplifying completes the proof of the lemma. ∎

5.5.6 Proof of Theorem 5.5.3

We first show that the DHK1 assumption implies DEG is PA1-secure, and then that the DDH assumption implies it is IND-CPA-secure. Finally we briefly indicate how to show that the DHK0 assumption implies DEG is PA0-secure.

Let $\boldsymbol{C}$ be a polynomial-time ciphertext creator attacking DEG. We build a polynomial-time pa1-extractor $\boldsymbol{C}^*$ for it. To do so, we first define a polynomial-time dhk1-adversary $\boldsymbol{H}$ attacking prime-order-group generator G. By the DHK1 assumption, $\boldsymbol{H}$ has a polynomial-time dhk1-extractor $\boldsymbol{H}^*$. We then use $\boldsymbol{H}^*$ to build $\boldsymbol{C}^*$. The descriptions of $\boldsymbol{H}$ and $\boldsymbol{C}^*$ are in Figure 5.14.

The random tape $X_2 \| R[\boldsymbol{C}]$ of $\boldsymbol{H}$ consists of a choice X_2 of an element in the group G_q together with a random tape $R[\boldsymbol{C}]$ for $\boldsymbol{C}$. The random tape of pa1-extractor $\boldsymbol{C}^*$ consists of a random tape for dhk1-extractor $\boldsymbol{H}^*$. Observe that extractor $\boldsymbol{C}^*$ gets input the random tape $R[\boldsymbol{C}]$ of $\boldsymbol{C}$, while extractor $\boldsymbol{H}^*$ must get as input the random tape $R[\boldsymbol{H}] = X_2 \| R[\boldsymbol{C}]$ of $\boldsymbol{H}$. Clearly, $\boldsymbol{H}$ and $\boldsymbol{C}^*$ are polynomial time. We claim that $\boldsymbol{C}^*$ is a successful pa1-extractor for $\boldsymbol{C}$. To prove this, let $\boldsymbol{D}$ be a polynomial-time distinguisher for $\boldsymbol{C}$, and fix $k \in \mathbb{N}$. We state a

Dhk1-adversary $\boldsymbol{H}(p, q, g, A; R[\boldsymbol{H}])$
Parse $R[\boldsymbol{H}]$ as $X_2 \| R[\boldsymbol{C}]$ where $X_2 \in G_q$
Run $\boldsymbol{C}$ on input (p, q, g, A, X_2) and coins $R[\boldsymbol{C}]$ until it halts, replying to its oracle queries as follows:
– If $\boldsymbol{C}$ makes query (Y, W, U) then
Make query (Y, W); Let b denote the response
If $\big(Y \not\equiv g^b \pmod p$ or $W \not\equiv A^b \pmod p\big)$ then $M \leftarrow \perp$
else $M \leftarrow U \cdot X_2^{-b} \bmod p$ EndIf
Return M to $\boldsymbol{C}$ as the reply EndIf
Halt

Pa1-extractor $\boldsymbol{C}^*(Q, St[\boldsymbol{C}^*]; R[\boldsymbol{C}^*])$
If $St[\boldsymbol{C}^*]$ is the initial state then
Parse $St[\boldsymbol{C}^*]$ as $((p, q, g, A, X_2), R[\boldsymbol{C}])$; $St[\boldsymbol{H}^*] \leftarrow ((p, q, g, A), X_2 \| R[\boldsymbol{C}])$
else Parse $St[\boldsymbol{C}^*]$ as $((p, q, g, A, X_2), St[\boldsymbol{H}^*])$ EndIf
Parse Q as (Y, W, U); $(b, St[\boldsymbol{H}^*]) \leftarrow \boldsymbol{H}^*((Y, W), St[\boldsymbol{H}^*]; R[\boldsymbol{C}^*])$
If $\big(Y \not\equiv g^b \pmod p$ or $W \not\equiv A^b \pmod p\big)$ then $M \leftarrow \perp$
else $M \leftarrow U \cdot X_2^{-b} \bmod p$ EndIf
$St[\boldsymbol{C}^*] \leftarrow ((p, q, g, A, X_2), St[\boldsymbol{H}^*])$
Return $(M, St[\boldsymbol{C}^*])$

Adversary $\boldsymbol{Y}'(\mathsf{find}, (p, q, g, X); R[\boldsymbol{Y}'])$
Parse $R[\boldsymbol{Y}']$ as $x_1 \| R[\boldsymbol{Y}]$, where $x_1 \in \mathbb{Z}_q$; $X_1 \leftarrow g^{x_1} \bmod p$
$(M_0, M_1, St) \xleftarrow{\$} \boldsymbol{Y}(\mathsf{find}, (p, q, g, X_1, X); R[\boldsymbol{Y}])$; $St' \leftarrow (St, x_1)$
Return (M_0, M_1, St')

Adversary $\boldsymbol{Y}'(\mathsf{guess}, C', St')$
Parse St' as (St, x_1); Parse C' as (Y, U); $W \leftarrow Y^{x_1} \bmod p$; $C \leftarrow (Y, W, U)$
$d \leftarrow \boldsymbol{Y}(\mathsf{guess}, C, St)$; Return d

Figure 5.14 Dhk1-adversary $\boldsymbol{H}$, pa1-extractor $\boldsymbol{C}^*$, and ind-cpa-adversary $\boldsymbol{Y}'$ for the proof of Theorem 5.5.3.

---

claim, conclude the proof given this claim and Lemma 5.5.5, and then return to prove the claim.

Claim 5.5.6 Let DecOK denote the event that all $\boldsymbol{C}^*$'s answers to $\boldsymbol{C}$'s queries are correct in experiment $\mathbf{Exp}^{\text{pa1-x}}_{\text{DEG},\boldsymbol{C},\boldsymbol{D},\boldsymbol{C}^*}(k)$. Then, $\Pr\left[\overline{\text{DecOK}}\right] \leq \mathbf{Adv}^{\text{dhk1}}_{\mathsf{G},\boldsymbol{H},\boldsymbol{H}^*}(k)$. ∎

Applying Lemma 5.5.5 and Claim 5.5.6, we obtain the desired result as follows.

$$\begin{aligned}
&\mathbf{Adv}^{\mathrm{pa1}}_{\mathrm{DEG},\boldsymbol{C},\boldsymbol{D},\boldsymbol{C}^*}(k) \\
&\quad = \Pr\left[\mathbf{Exp}^{\mathrm{pa1\text{-}d}}_{\mathrm{DEG},\boldsymbol{C},\boldsymbol{D}}(k)=1\right] - \Pr\left[\mathbf{Exp}^{\mathrm{pa1\text{-}x}}_{\mathrm{DEG},\boldsymbol{C},\boldsymbol{D},\boldsymbol{C}^*}(k)=1\right] \\
&\quad \leq \Pr\left[\mathbf{Exp}^{\mathrm{pa1\text{-}d}}_{\mathrm{DEG},\boldsymbol{C},\boldsymbol{D}}(k)=1\right] - \Pr\left[\mathbf{Exp}^{\mathrm{pa1\text{-}d}}_{\mathrm{DEG},\boldsymbol{C},\boldsymbol{D}}(k)=1\right] + \Pr\left[\overline{\textsc{DecOK}}\right] \\
&\quad \leq \mathbf{Adv}^{\mathrm{dhk1}}_{\mathsf{G},\boldsymbol{H},\boldsymbol{H}^*}(k)
\end{aligned}$$

By the DHK1 assumption, the function $\mathbf{Adv}^{\mathrm{dhk1}}_{\mathsf{G},\boldsymbol{H},\boldsymbol{H}^*}$ is negligible and hence for every polynomial-time distinguisher $\boldsymbol{D}$, the function $\mathbf{Adv}^{\mathrm{pa1}}_{\mathrm{DEG},\boldsymbol{C},\boldsymbol{D},\boldsymbol{C}^*}$ is negligible. Thus $\boldsymbol{C}^*$ is a successful pa1-extractor for $\boldsymbol{C}$, and DEG is PA1-secure. It remains to prove the claim above.

Proof of Claim 5.5.6: We observe that by the definition of pa1-extractor $\boldsymbol{C}^*$ and DEG's decryption algorithm $\mathcal{D}$, if $\boldsymbol{C}^*$'s response M to a query (Y,W,U) made by $\boldsymbol{C}$ is such that $M \neq \perp$ then $\mathcal{D}_{(p,q,g,x_1,x_2)}((Y,W,U)) \neq \perp$ and $M = \mathcal{D}_{(p,q,g,x_1,x_2)}((Y,W,U))$. Therefore,

$$\begin{aligned}
\Pr\left[\overline{\textsc{DecOK}}\right] &= \Pr[\boldsymbol{C} \text{ makes a query } (Y,W,U) \text{ for which } \boldsymbol{C}^*\text{'s response } M \\
&\qquad \text{is such that } M \neq \mathcal{D}_{(p,q,g,x_1,x_2)}((Y,W,U))] \\
&\leq \Pr[\boldsymbol{C} \text{ makes a query } (Y,W,U) \text{ for which } \boldsymbol{C}^*\text{'s response } M \\
&\qquad \text{is such that } M = \perp \wedge \mathcal{D}_{(p,q,g,x_1,x_2)}((Y,W,U)) \neq \perp] \\
&\leq \Pr[\boldsymbol{C} \text{ makes a query } (Y,W,U) \text{ for which } \boldsymbol{C}^*\text{'s response } M \\
&\qquad \text{is such that } Y \not\equiv g^b \pmod{p} \wedge W \equiv Y^{\mathrm{dlog}_g A} \pmod{p}] \\
&\leq \mathbf{Adv}^{\mathrm{dhk1}}_{\mathsf{G},\boldsymbol{H},\boldsymbol{H}^*}(k)
\end{aligned}$$

The last inequality follows from the definition of dhk1-adversary $\boldsymbol{H}$. ∎

To prove that DEG is IND-CPA-secure under the DDH assumption, we use the fact that if this assumption holds, then the ElGamal scheme EG is IND-CPA-secure. Let $\boldsymbol{Y}$ be an ind-cpa-adversary attacking DEG. Consider the ind-cpa-adversary $\boldsymbol{Y}'$ attacking EG depicted in Figure 5.14. A random tape $x_1 \| R[\boldsymbol{Y}]$

for adversary $\boldsymbol{Y}'$ has two parts. The first part is a choice x_1 of an exponent in $\mathbb{Z}_q$ that $\boldsymbol{Y}'$ uses to compute the public key (p, q, g, X_1, X) and the challenge ciphertext C for $\boldsymbol{Y}$. The second part is a random tape for $\boldsymbol{Y}$. $\boldsymbol{Y}'$ runs $\boldsymbol{Y}$ and returns the output of the latter. Clearly, $\boldsymbol{Y}'$ runs in polynomial time and $\mathbf{Adv}^{\text{ind-cpa}}_{\text{EG},\boldsymbol{Y}'}(k) = \mathbf{Adv}^{\text{ind-cpa}}_{\text{DEG},\boldsymbol{Y}}(k)$, for every $k \in \mathbb{N}$. Since EG is IND-CPA-secure, the ind-cpa-advantage of $\boldsymbol{Y}'$ is negligible, and hence it follows that the ind-cpa-advantage of $\boldsymbol{Y}$ is negligible. Thus DEG is IND-CPA-secure.

The proof that the DHK0 assumption implies DEG is PA0-secure is analogous to the proof that the DHK1 assumption implies DEG is PA1-secure. The difference is that the given ciphertext creator $\boldsymbol{C}$ attacking DEG makes a single oracle query, and thus the dhk1-adversary $\boldsymbol{H}$ attacking prime-order-group generator G is a dhk0-adversary. The DHK0 assumption then implies the existence of a polynomial-time dhk0-extractor $\boldsymbol{H}^*$ for $\boldsymbol{H}$. The extractor $\boldsymbol{C}^*$ defined in Figure 5.14 is then a pa0-extractor. The proof that $\boldsymbol{C}^*$ is a successful polynomial-time pa0-extractor for $\boldsymbol{C}$ is exactly as before.

5.5.7 Proof of Theorem 5.5.4

CSL is known to be IND-CCA1-secure (and hence IND-CPA-secure) under the DDH assumption (cf. [35]). Therefore, it is sufficient to prove that it is PA1-secure under the DHK1 assumption and PA0-secure under the DHK0 assumption.

We begin with the former. Let $\boldsymbol{C}$ be a polynomial-time ciphertext creator attacking CSL.

We build a polynomial-time pa1-extractor $\boldsymbol{C}^*$ for it. First, we construct a polynomial-time dhk1-adversary $\boldsymbol{H}$ attacking prime-order-group generator G. By the DHK1 assumption, $\boldsymbol{H}$ has a polynomial-time dhk1-extractor $\boldsymbol{H}^*$. We then use $\boldsymbol{H}^*$ to build $\boldsymbol{C}^*$. Algorithms $\boldsymbol{H}$ and $\boldsymbol{C}^*$ are defined in Figure 5.15.

The random tape $g_2 \| x_2 \| Z \| R[\boldsymbol{C}]$ of $\boldsymbol{H}$ consists of a choice g_2 of an element in the group G_q, a choice x_2 of an exponent in $\mathbb{Z}_q$, a choice Z of an element in

Dhk1-adversary $\boldsymbol{H}(p, q, g, A; R[\boldsymbol{H}])$
Parse $R[\boldsymbol{H}]$ as $g_2 \| x_2 \| Z \| R[\boldsymbol{C}]$ where $g_2 \in G_q$, $x_2 \in \mathbb{Z}_q$, and $Z \in G_q$
$X \leftarrow A \cdot g_2^{x_2} \bmod p$
Run $\boldsymbol{C}$ on input (p, q, g_1, g_2, X, Z) and coins $R[\boldsymbol{C}]$ until it halts, replying to its oracle queries as follows:
– If $\boldsymbol{C}$ makes query (R_1, R_2, E, V) then
$W \leftarrow V \cdot R_2^{-x_2} \bmod p$; Make query (R_1, W) and let b denote the response
If $\big(R_1 \not\equiv g_1^b \pmod p$ or $R_2 \not\equiv g_2^b \pmod p$ or $V \not\equiv X^b \pmod p\big)$ then
$M \leftarrow \perp$
else $M \leftarrow E \cdot Z^{-b} \bmod p$ EndIf
Return M to $\boldsymbol{C}$ as the reply EndIf
Halt

Pa1-extractor $\boldsymbol{C}^*(Q, St[\boldsymbol{C}^*]; R[\boldsymbol{C}^*])$
If $St[\boldsymbol{C}^*]$ is the initial state then
Parse $St[\boldsymbol{C}^*]$ as $((p, q, g_1, g_2, X, Z), R[\boldsymbol{C}])$
Parse $R[\boldsymbol{C}^*]$ as $x_2 \| R[\boldsymbol{H}^*]$ where $x_2 \in \mathbb{Z}_q$
$A \leftarrow X \cdot g_2^{-x_2} \bmod p$; $St[\boldsymbol{H}^*] \leftarrow ((p, q, g, A), g_2 \| x_2 \| Z \| R[\boldsymbol{C}])$
else Parse $St[\boldsymbol{C}^*]$ as $((p, q, g_1, g_2, X, Z), St[\boldsymbol{H}^*], R[\boldsymbol{H}^*])$ EndIf
Parse Q as (R_1, R_2, E, V)
$W \leftarrow V \cdot R_2^{-x_2} \bmod p$; $(b, St[\boldsymbol{H}^*]) \leftarrow \boldsymbol{H}^*((R_1, W), St[\boldsymbol{H}^*]; R[\boldsymbol{H}^*])$
If $\big(R_1 \not\equiv g_1^b \pmod p$ or $R_2 \not\equiv g_2^b \pmod p$ or $V \not\equiv X^b \pmod p\big)$ then $M \leftarrow \perp$
else $M \leftarrow E \cdot Z^{-b} \bmod p$ EndIf
$St[\boldsymbol{C}^*] \leftarrow ((p, q, g_1, g_2, X, Z), St[\boldsymbol{H}^*], R[\boldsymbol{H}^*])$
Return $(M, St[\boldsymbol{C}^*])$

Figure 5.15 Dhk1-adversary $\boldsymbol{H}$ and pa1-extractor $\boldsymbol{C}^*$ for the proof of Theorem 5.5.4.

G_q, and a random tape $R[\boldsymbol{C}]$ for $\boldsymbol{C}$. The random tape $x_2 \| R[\boldsymbol{H}^*]$ of pa1-extractor $\boldsymbol{C}^*$ consists of a choice x_2 of an exponent in $\mathbb{Z}_q$ and a random tape for dhk1-extractor $\boldsymbol{H}^*$. $\boldsymbol{C}^*$ uses x_2 to compute value A and a random tape $g_2 \| x_2 \| Z \| R[\boldsymbol{C}]$ corresponding to $\boldsymbol{H}$, for $\boldsymbol{H}^*$. While extractor $\boldsymbol{C}^*$ gets input the random tape $R[\boldsymbol{C}]$ of $\boldsymbol{C}$, extractor $\boldsymbol{H}^*$ must be given input the random tape $R[\boldsymbol{H}] = g_2 \| x_2 \| Z \| R[\boldsymbol{C}]$ of $\boldsymbol{H}$. Clearly, $\boldsymbol{H}$ and $\boldsymbol{C}^*$ are polynomial time. We claim that $\boldsymbol{C}^*$ is a successful pa1-extractor for $\boldsymbol{C}$. To prove this, let $\boldsymbol{D}$ be a polynomial-time distinguisher for

$\boldsymbol{C}$, and fix $k \in \mathbb{N}$. We state a claim, conclude the proof given this claim and Lemma 5.5.5, and then return to prove the claim.

Claim 5.5.7 Let DecOK denote the event that all $\boldsymbol{C}^*$'s answers to $\boldsymbol{C}$'s queries are correct in experiment $\mathbf{Exp}^{\text{pa1-x}}_{\text{CSL},\boldsymbol{C},\boldsymbol{D},\boldsymbol{C}^*}(k)$. Then there exists a negligible function $\nu_{\boldsymbol{D}}$ such that $\Pr\left[\,\overline{\text{DecOK}}\,\right] \leq \mathbf{Adv}^{\text{dhk1}}_{\mathsf{G},\boldsymbol{H},\boldsymbol{H}^*}(k) + \nu_{\boldsymbol{D}}(k)$. ∎

Analogously to the proof of Theorem 5.5.3, Lemma 5.5.5 and Claim 5.5.7 imply that

$$\mathbf{Adv}^{\text{pa1}}_{\text{CSL},\boldsymbol{C},\boldsymbol{D},\boldsymbol{C}^*}(k) \;\leq\; \mathbf{Adv}^{\text{dhk1}}_{\mathsf{G},\boldsymbol{H},\boldsymbol{H}^*}(k) + \nu_{\boldsymbol{D}}(k)\,.$$

By the DHK1 assumption, the function $\mathbf{Adv}^{\text{dhk1}}_{\mathsf{G},\boldsymbol{H},\boldsymbol{H}^*}$ is negligible and hence for every polynomial-time distinguisher $\boldsymbol{D}$, the function $\mathbf{Adv}^{\text{pa1}}_{\text{CSL},\boldsymbol{C},\boldsymbol{D},\boldsymbol{C}^*}$ is negligible. Thus $\boldsymbol{C}^*$ is a successful pa1-extractor for $\boldsymbol{C}$, and CSL is PA1-secure. It remains to prove Claim 5.5.7.

Sketch of Proof of Claim 5.5.7: We call $(R_1, R_2, E, V) \in G_q^4$ a *valid ciphertext* with respect to public key (p, q, g_1, g_2, X, Z) if $\text{dlog}_{g_1} R_1 = \text{dlog}_{g_2} R_2$, and an *invalid ciphertext* otherwise. Cramer and Shoup [35] proved that the decryption algorithm of their IND-CCA2-secure scheme rejects all invalid ciphertexts generated by an adversary with all but negligible probability. By slightly modifying their proof, we can show that the decryption algorithm of CSL rejects all invalid ciphertexts generated by an adversary with all but negligible probability. Using this fact, we can prove that

$$\mathbf{Adv}^{\text{pa1}}_{\text{CSL},\boldsymbol{C},\boldsymbol{D},\boldsymbol{C}^*}(k) \;\leq\; \mathbf{Adv}^{\text{dhk1}}_{\mathsf{G},\boldsymbol{H},\boldsymbol{H}^*}(k) + \nu_{\boldsymbol{D}}(k)\,,$$

for a negligible function $\nu_{\boldsymbol{D}}$. Details are omitted. ∎

The proof that the DHK0 assumption implies CSL is PA0-secure is analogous to the proof that the DHK1 assumption implies CSL is PA1-secure. The difference is that the given ciphertext creator $\boldsymbol{C}$ attacking CSL makes a single oracle query, and thus the dhk1-adversary $\boldsymbol{H}$ attacking prime-order-group generator G is a dhk0-adversary. The DHK0 assumption then implies the existence of a polynomial-time

dhk0-extractor $\boldsymbol{H}^*$ for $\boldsymbol{H}$. The extractor $\boldsymbol{C}^*$ defined in Figure 5.15 is then a pa0-extractor. The proof that $\boldsymbol{C}^*$ is a successful polynomial-time pa0-extractor for $\boldsymbol{C}$ is exactly as before.

5.6 Damgård's arguments about DEG's security

We first review Damgård's security notions and then his proof.

5.6.1 RPR-security

Let $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D}, \mathsf{MsgSp})$ be an encryption scheme. Damgård [36] considers security against recovery of a random plaintext under a non-adaptive chosen-ciphertext attack. Namely, let us say that $\mathcal{AE}$ is RPR-CCA1-secure if for every polynomial time $\boldsymbol{R}$, the probability that the following experiment returns 1 is negligible as a function of k:

$(pk, sk) \xleftarrow{\$} \mathcal{K}(1^k)$
$St \xleftarrow{\$} \boldsymbol{R}^{\mathcal{D}_{sk}(\cdot)}(\mathsf{find}, pk)\,;\ M \xleftarrow{\$} \mathsf{MsgSp}(pk)\,;\ C \xleftarrow{\$} \mathcal{E}_{pk}(M)\,;\ M' \leftarrow \boldsymbol{R}(\mathsf{guess}, C, St)$
If $M = M'$ then return 1 else return 0

One can show that IND-CCA1 $\rightarrow$ RPR-CCA1 and RPR-CCA1 $\not\rightarrow$ IND-CCA1, meaning this notion of security is weaker than IND-CCA1-security. One can define RPR-CPA-security by not giving $\boldsymbol{R}$ the decryption oracle in the first stage above.

5.6.2 Claim and proof approach

Damgård [36, Theorem 2] claims that DEG is RPR-CCA1-secure assuming DHK0 and the security of the ElGamal encryption scheme under RPR-CPA. He first shows that if ElGamal is RPR-CPA-secure then so is DEG [36, Lemma 1]. His proof of his Theorem 2 [36, Page 453] claims to turn a given rpr-cca1-adversary into an rpr-cpa-adversary. Applying his Lemma 1, he can conclude. The issue is how an rpr-cca1-adversary $\boldsymbol{R}$ is turned into an rpr-cpa-adversary. Quoting from

the proof of [36, Page 453], with some minor changes for consistency with our notation:

> Let $C_1, C_2, \ldots$ be the sequence of ciphertexts whose decryptions $\boldsymbol{R}$ requests from its oracle. Let $\boldsymbol{H}_i$ be the algorithm that simulates $\boldsymbol{R}$ until the output of C_i and then stops. We can now show by induction that for all i, $\boldsymbol{H}_i$ can be simulated without access to a decryption oracle. $\boldsymbol{H}_1$ is clear. To do $\boldsymbol{H}_{i+1}$, observe that by induction, $\boldsymbol{H}_i$ can be simulated without the oracle. Then the DHK0 assumption guarantees us the existence of an algorithm $\boldsymbol{H}_i^*$ that outputs y where $C_i = (Y, W, U)$ and $Y = g^y$, whenever C_i produces a non-null output from the decryption. Knowledge of y suffices to decrypt C_i, and therefore we can simulate also the last steps of $\boldsymbol{R}_{i+1}$. From $\boldsymbol{R}$ we can therefore *build an algorithm that breaks the system* under an RPR-CPA attack, and we are done by Lemma 1.

The problem is the emphasized text at the end of the quoted proof above. An *algorithm* is by definition a *finite* object. We could view it as a program, or, more formally, as a Turing machine, but it must have a finite description of size independent of the size of the input. However, the algorithm resulting from the above proof contains descriptions of the extractors $\boldsymbol{H}_1^*, \boldsymbol{H}_2^*, \ldots$ which it must run as subroutines. We claim this list is infinite, so that the constructed "algorithm" is actually an object having an infinite description, and not an algorithm at all.

Why is the list of extractors infinite? The list is finite for any given value of the security parameter. But suppose $\boldsymbol{R}$ makes $q(k) = k$ oracle queries. The constructed algorithm must work for any value of k. So it must include the list of extractors corresponding to all values of k, and this list is unbounded.

The easiest fix to the above is to use the DHK1 assumption instead. This guarantees a single extractor that can interactively take inputs and extract the appropriate quantities from them. In that case, Damgård's proof goes through to show that DEG is RPR-CCA1-secure assuming DHK1 and the RPR-CPA-security of ElGamal. (Note we show something somewhat stronger, namely IND-CCA1-security, and we also show PA0, PA1.)

We remark that a strategy similar to Damgård's is used by [7] in their proof that PA-BDPR + IND-CPA implies IND-CCA2 in the RO model. They can avoid having their algorithm remember infinitely many extractors because in their definition of PA-BDPR the extractor does not depend on the adversary.

This chapter, in full, is a reprint of the material as it appears in M. Bellare and A. Palacio, "Towards Plaintext-Aware Public-Key Encryption without Random Oracles," *Advances in Cryptology - Asiacrypt 2004 Proceedings*, Lecture Notes in Computer Science Vol. 3329, P. J. Lee ed., Springer-Verlag, 2004.

Bibliography

[1] M. Abdalla, J. An, M. Bellare, and C. Namprempre. "From identification to signatures via the Fiat-Shamir Transform: Minimizing assumptions for security and forward-security". In *Advances in Cryptology – EUROCRYPT 2002 Proceedings*. Lecture Notes in Computer Science, Vol. 2332, L. Knudsen ed., Springer-Verlag, 2002.

[2] M. Backes, B. Pfitzmann, and M. Waidner. "A composable cryptographic library with nested operations". In *Proceedings of the 10th Annual Conference on Computer and Communications Security*. ACM, 2003.

[3] B. Barak. "How to go beyond the black-box simulation barrier". In *Proceedings of the 42nd Symposium on Foundations of Computer Science*. IEEE, 2001.

[4] N. Barić and B. Pfitzmann. "Collision-free accumulators and fail-stop signature schemes without trees". In *Advances in Cryptology – EUROCRYPT 1997 Proceedings*. Lecture Notes in Computer Science, Vol. 1233, W. Fumy ed., Springer-Verlag, 1997.

[5] M. Bellare. "A note on negligible functions". *Journal of Cryptology*, 15(4):271–284, 2002.

[6] M. Bellare, A. Boldyreva, and A. Palacio. "An un-instantiable random oracle model scheme for a hybrid encryption problem". In *Advances in Cryptology – EUROCRYPT 2004 Proceedings*. Lecture Notes in Computer Science, Vol. 3027, C. Cachin and J. Camenisch ed., Springer-Verlag, 2004.

[7] M. Bellare, A. Desai, D. Pointcheval, and P. Rogaway. "Relations among notions of security for public-key encryption schemes". In *Advances in Cryptology – CRYPTO 1998 Proceedings*. Lecture Notes in Computer Science, Vol. 1462, H. Krawczyk ed., Springer-Verlag, 1998.

[8] M. Bellare, M. Fischlin, S. Goldwasser, and S. Micali. "Identification protocols secure against reset attacks". In *Advances in Cryptology – EUROCRYPT 2001 Proceedings*. Lecture Notes in Computer Science, Vol. 2045, B. Pfitzmann ed., Springer-Verlag, 2001.

[9] M. Bellare and O. Goldreich. "On defining proofs of knowledge". In *Advances in Cryptology – CRYPTO 1992 Proceedings*. Lecture Notes in Computer Science, Vol. 740, E. Brickell ed., Springer-Verlag, 1992.

[10] M. Bellare, R. Guérin, and P. Rogaway. "XOR MACs: New methods for message authentication using finite pseudorandom functions". In *Advances in Cryptology – CRYPTO 1995 Proceedings*. Lecture Notes in Computer Science, Vol. 963, D. Coppersmith ed., Springer-Verlag, 1995.

[11] M. Bellare, M. Jakobsson, and M. Yung. "Round-optimal zero-knowledge arguments based on any one-way function". In *Advances in Cryptology – EUROCRYPT 1997 Proceedings*. Lecture Notes in Computer Science, Vol. 1233, W. Fumy ed., Springer-Verlag, 1997.

[12] M. Bellare, J. Kilian, and P. Rogaway. "The security of cipher block chaining'. In *Advances in Cryptology – CRYPTO 1994 Proceedings*. Lecture Notes in Computer Science, Vol. 839, Y. Desmedt ed., Springer-Verlag, 1994.

[13] M. Bellare, S. Micali, and R. Ostrovsky. "The (true) complexity of statistical zero-knowledge". In *Proceedings of the 22nd Annual Symposium on the Theory of Computing*. ACM, 1990.

[14] M. Bellare and S. Miner. "A forward-secure digital signature scheme". In *Advances in Cryptology – CRYPT0 1999 Proceedings*. Lecture Notes in Computer Science, Vol. 1666, M. Weiner ed., Springer-Verlag, 1999.

[15] M. Bellare, C. Namprempre, and G. Neven. "Security proofs for identity-based identification and signature schemes". In *Advances in Cryptology – EUROCRYPT 2004 Proceedings*. Lecture Notes in Computer Science, Vol. 3027, C. Cachin and J. Camenisch eds., Springer-Verlag, 2004.

[16] M. Bellare, C. Namprempre, D. Pointcheval, and M. Semanko. "The one-more-RSA-inversion problems and the security of Chaum's blind signature scheme". *Journal of Cryptology*, 16(3):185–215, 2003. Preliminary version, entitled "The power of RSA inversion oracles and the security of Chaum's RSA-based blind signature scheme," in *Financial Cryptography 2001 Proceedings*, Lecture Notes in Computer Science, Vol. 2339, P. Syverson ed., Springer-Verlag, 2001.

[17] M. Bellare and G. Neven. "Transitive signatures: New schemes and proofs". *IEEE Transactions on Information Theory*, 51(6):2133–2151, 2005. Preliminary version, entitled "Transitive signatures based on factoring and RSA," in *Advances in Cryptology – ASIACRYPT 2002 Proceedings*, Lecture Notes in Computer Science, Vol. 2501, Y. Zheng ed., Springer-Verlag, 2001.

[18] M. Bellare and A. Palacio. "The knowledge of exponent assumptions and 3-round zero-knowledge protocols". In *Advances in Cryptology – CRYPTO 2004 Proceedings*. Lecture Notes in Computer Science, Vol. 3152, M. Franklin ed., Springer-Verlag, 2004.

[19] M. Bellare, D. Pointcheval, and P. Rogaway. "Authenticated key exchange secure against dictionary attacks". In *Advances in Cryptology – EUROCRYPT 2000 Proceedings*. Lecture Notes in Computer Science, Vol. 1807, B. Preneel ed., Springer-Verlag, 2000.

[20] M. Bellare and P. Rogaway. "Entity authentication and key distribution". In *Advances in Cryptology – CRYPTO 1993 Proceedings*. Lecture Notes in Computer Science, Vol. 773, D. Stinson ed., Springer-Verlag, 1993.

[21] M. Bellare and P. Rogaway. "Random oracles are practical: A paradigm for designing efficient protocols". In *Proceedings of the 1st Annual Conference on Computer and Communications Security*. ACM, 1993.

[22] M. Bellare and P. Rogaway. "Optimal asymmetric encryption". In *Advances in Cryptology – EUROCRYPT 1994 Proceedings*. Lecture Notes in Computer Science, Vol. 950, A. De Santis ed., Springer-Verlag, 1994.

[23] M. Blum. "How to prove a theorem so no one else can claim it". In *Proceedings of the International Congress of Mathematicians*, pages 1444–1451, 1986.

[24] M. Blum, P. Feldman, and S. Micali. "Non-interactive zero-knowledge and its applications". In *Proceedings of the 20th Annual Symposium on the Theory of Computing*. ACM, 1988.

[25] M. Blum, P. Feldman, and S. Micali. "P roving security against chosen ciphertext attacks". In *Advances in Cryptology – CRYPTO 1988 Proceedings*. Lecture Notes in Computer Science, Vol. 403, S. Goldwasser ed., Springer-Verlag, 1988.

[26] D. Boneh. "Simplified OAEP for the RSA and Rabin functions". In *Advances in Cryptology – CRYPTO 2001 Proceedings*. Lecture Notes in Computer Science, Vol. 2139, J. Kilian ed., Springer-Verlag, 2001.

[27] N. Borisov, I. Goldberg, and D. Wagner. "Intercepting mobile communications: The insecurity of 802.11". In *Proceedings of the ACM SIGMOBILE 7th Annual International Conference on Mobile Computing and Networking, MOBICOM 2001*. ACM, 2001.

[28] G. Brassard, D. Chaum, and C. Crépeau. "Minimum disclosure proofs of knowledge". *Journal of Computer and System Sciences*, 37(2):156–189, 1988.

[29] R. Canetti, O. Goldreich, and S. Halevi. "The random oracle methodology, revisited". In *Proceedings of the 30th Annual Symposium on the Theory of Computing*. ACM, 1998.

[30] R. Canetti, S. Goldwasser, O. Goldreich, and S. Micali. "Resettable zero-knowledge". In *Proceedings of the 32nd Annual Symposium on the Theory of Computing*. ACM, 2000.

[31] R. Canetti and H. Krawczyk. "Universally composable notions of key-exchange and secure channels". In *Advances in Cryptology – EUROCRYPT 2002 Proceedings*. Lecture Notes in Computer Science, Vol. 2332, L. Knudsen ed., Springer-Verlag, 2002.

[32] D. Chaum. "Blind signatures for untraceable payments". In *Advances in Cryptology – CRYPTO 1982 Proceedings*. Lecture Notes in Computer Science, Plenum Press, New York and London, 1983, 1982.

[33] R. Cramer and V. Shoup. "A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack". In *Advances in Cryptology – CRYPTO 1998 Proceedings*. Lecture Notes in Computer Science, Vol. 1462, H. Krawczyk ed., Springer-Verlag, 1998.

[34] R. Cramer and V. Shoup. "Signature schemes based on the strong RSA assumption". *ACM Transactions on Information and System Security*, 3(3):161–185, 2000. Preliminary version in *Proceedings of the 6th Annual Conference on Computer and Communications Security*, ACM, 1999.

[35] R. Cramer and V. Shoup. "Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack". *SIAM Journal on Computing*, 33(1):167–226, 2003.

[36] I. Damgård. "Towards practical public key systems secure against chosen ciphertext attacks". In *Advances in Cryptology – CRYPTO 1991 Proceedings*. Lecture Notes in Computer Science, Vol. 576, J. Feigenbaum ed., Springer-Verlag, 1991.

[37] A. W. Dent. "The Cramer-Shoup encryption scheme is plaintext aware in the standard model". In *Advances in Cryptology – EUROCRYPT 2006 Proceedings*. Lecture Notes in Computer Science, Vol. 4004, S. Vaudenay ed., Springer-Verlag, 2006.

[38] A. W. Dent. "The hardness of the DHK problem in the generic group model". *Cryptology ePrint Archive: Report 2006/156*, 2006. `http://eprint.iacr.org/2006/156/`.

[39] W. Diffie and M.E. Hellman. "New directions in cryptography'. *IEEE Transactions on Information Theory*, IT-22(6):644–654, 1976.

[40] D. Dolev, C. Dwork, and M. Naor. "Non-malleable cryptography". *SIAM Journal on Computing*, 30(2):391–437, 2000.

[41] D. Dolev and A. Yao. "On the security of public-key protocols". *IEEE Transactions on Information Theory*, 29(2):198–208, 1983.

[42] T. ElGamal. "A public key cryptosystem and signature scheme based on discrete logarithms". *IEEE Transactions of Information Theory*, IT-31(4):469–472, 1985.

[43] U. Feige, A. Fiat, and A. Shamir. "Zero knowledge proofs of identity". *Journal of Cryptology*, 1(2):77–94, 1988.

[44] U. Feige and A. Shamir. "Witness indistinguishable and witness hiding protocols". In *Proceedings of the 22nd Annual Symposium on the Theory of Computing*. ACM, 1990.

[45] A. Fiat and A. Shamir. "How to prove yourself: Practical solutions to identification and signature problems". In *Advances in Cryptology – CRYPTO 1986 Proceedings*. Lecture Notes in Computer Science, Vol. 263, A. Odlyzko ed., Springer-Verlag, 1986.

[46] S. Fluhrer, I .Mantin, and A. Shamir. "Weaknesses in the key scheduling algorithm of RC4". In *Selected Areas in Cryptography: 8th Annual International Workshop Proceedings*. Lecture Notes in Computer Science, Vol. 2259, S. Vaudenay and A. M. Youssef ed., Springer-Verlag, 2001.

[47] E. Fujisaki and T. Okamoto. "Statistical zero knowledge protocols to prove modular polynomial relations". In *Advances in Cryptology – CRYPTO 1997 Proceedings*. Lecture Notes in Computer Science, Vol. 1294, B. Kaliski ed., Springer-Verlag, 1997.

[48] E. Fujisaki, T. Okamoto, D. Pointcheval, and J. Stern. "RSA-OAEP is secure under the RSA assumption". In *Advances in Cryptology – CRYPTO 2001 Proceedings*. Lecture Notes in Computer Science, Vol. 2139, J. Kilian ed., Springer-Verlag, 2001.

[49] R. Gennaro, S. Halevi, and T. Rabin. "Secure hash-and-sign signatures without the random oracle". In *Advances in Cryptology – EUROCRYPT 1999 Proceedings*. Lecture Notes in Computer Science, Vol. 1592, J. Stern ed., Springer-Verlag, 1999.

[50] O. Goldreich. "A uniform-complexity treatment of encryption and zero-knowledge". *Journal of Cryptology*, 6(1):21–53, 1993.

[51] O. Goldreich. *"Foundations of Cryptography"*, volume Basic Tools. Cambridge University Press, June 2001.

[52] O. Goldreich and H. Krawczyk. "An the composition of zero-knowledge proof systems". *SIAM Journal on Computing*, 25(1):169–192, 1996.

[53] O. Goldreich, S. Micali, and A. Wigderson. "Proofs that yield nothing but their validity or all languages in $\mathbb{NP}$ have zero-knowledge proof systems". *Journal of the ACM*, 38(1):691–729, 1991. Preliminary version in *Proceedings of the 27th Symposium on Foundations of Computer Science*, IEEE, 1986.

[54] O. Goldreich and Y. Oren. "Definitions and properties of zero-knowledge proof systems". *Journal of Cryptology*, 7(1):1–32, 1994.

[55] S. Goldwasser and S. Micali. "Probabilistic encryption". *Journal of Computer and System Sciences*, 28(2):270–299, 1984.

[56] S. Goldwasser, S. Micali, and C. Rackoff. "The knowledge complexity of interactive proof systems". *SIAM Journal on Computing*, 18(1):186–208, 1989.

[57] S. Goldwasser and Y. Taumann. "On the (in)security of the Fiat-Shamir paradigm". In *Proceedings of the 44th Symposium on Foundations of Computer Science*. IEEE, 2003.

[58] L. Guillou and J. J. Quisquater. "A "paradoxical" identity-based signature scheme resulting from zero-knowledge". In *Advances in Cryptology – CRYPTO 1988 Proceedings*. Lecture Notes in Computer Science, Vol. 403, S. Goldwasser ed., Springer-Verlag, 1988.

[59] S. Hada and T. Tanaka. "On the existence of 3-round zero-knowledge protocols". In *Advances in Cryptology – CRYPTO 1998 Proceedings*. Lecture Notes in Computer Science, Vol. 1462, H. Krawczyk ed., Springer-Verlag, 1998. [Preliminary version of [60]].

[60] S. Hada and T. Tanaka. "On the existence of 3-round zero-knowledge protocols". *Cryptology ePrint Archive: Report 1999/009*, 1999. [Final version of [59]].

[61] J. Herzog, M. Liskov, and S. Micali. "Plaintext awareness via key registration. In *Advances in Cryptology – CRYPTO 2003 Proceedings*. Lecture Notes in Computer Science, Vol. 2729, D. Boneh ed., Springer-Verlag, 2003.

[62] J. Kilian. "A note on efficient zero-knowledge proofs and arguments". In *Proceedings of the 24th Annual Symposium on the Theory of Computing*. ACM, 1992.

[63] T. Kohno, A. Stubblefield, A. D. Rubin, and D. S. Wallach. "Analysis of an electronic voting system". In *Proceedings of the Symposium on Security and Privacy*. IEEE, 2004.

[64] M. Lepinski. "On the existence of 3-round zero-knowledge proof systems". *MS Thesis*, June 2002. `http://theory.lcs.mit.edu/~cis/theses/lepinski-masters.ps`.

[65] M. Lepinski and S. Micali. "On the existence of 3-round zero-knowledge proof systems". *MIT LCS Technical Memo. 616*, April 2001. `http://www.lcs.mit.edu/publications/pubs/pdf/MIT-LCS-TM-616.pdf`.

[66] S. Micali, C. Rackoff, and B. Sloan. "The notion of security for probabilistic cryptosystems". *SIAM Journal on Computing*, 17(2):412–426, 1988.

[67] S. Micali and R. Rivest. "Transitive signature schemes". In *Topics in Cryptology – CT-RSA 2002 Proceedings*. Lecture Notes in Computer Science, Vol. 2271, B. Preneel ed., Springer-Verlag, 2002.

[68] M. Naor. "On cryptographic assumptions and challenges". Invited paper and talk. In *Advances in Cryptology – CRYPTO 2003 Proceedings*. Lecture Notes in Computer Science, Vol. 2729, D. Boneh ed., Springer-Verlag, 2003.

[69] M. Naor and M. Yung. "Public-key cryptosystems provably secure against chosen ciphertext attacks". In *Proceedings of the 22nd Annual Symposium on the Theory of Computing*. ACM, 1990.

[70] J. B. Nielsen. "Separating random oracle proofs from complexity theoretic proofs: The non-committing encryption case". In *Advances in Cryptology – CRYPTO 2002 Proceedings*. Lecture Notes in Computer Science, Vol. 2242, M. Yung ed., Springer-Verlag, 2002.

[71] T. Okamoto. "Provably secure and practical identification schemes and corresponding signature schemes". In *Advances in Cryptology – CRYPTO 1992 Proceedings*. Lecture Notes in Computer Science, Vol. 740, E. Brickell ed., Springer-Verlag, 1992.

[72] H. Ong and C. P. Schnorr. "Fast signature generation with a Fiat Shamir–like scheme". In *Advances in Cryptology – EUROCRYPT 1990 Proceedings*. Lecture Notes in Computer Science, Vol. 473, I. Damgård ed., Springer-Verlag, 1990.

[73] D. Pointcheval. "New public key cryptosystems based on the dependent-RSA problems". In *Advances in Cryptology – EUROCRYPT 1999 Proceedings*. Lecture Notes in Computer Science, Vol. 1592, J. Stern ed., Springer-Verlag, 1999.

[74] D. Pointcheval and J. Stern. "Security arguments for digital signatures and blind signatures". *Journal of Cryptology*, 13(3):361–396, 2000.

[75] C. Rackoff and D. Simon. "Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack". In *Advances in Cryptology – CRYPTO 1991 Proceedings*. Lecture Notes in Computer Science, Vol. 576, J. Feigenbaum ed., Springer-Verlag, 1991.

[76] R. Rivest, A. Shamir, and L. Adleman. "A method for obtaining digital signatures and public-key cryptosystems". *Communications of the ACM*, 21(2):120–126, 1978.

[77] K. Sakurai and T. Itoh. "On the discrepancy between serial and parallel of zero-knowledge protocols". In *Advances in Cryptology – CRYPTO 1992 Proceedings*. Lecture Notes in Computer Science, Vol. 740, E. Brickell ed., Springer-Verlag, 1992.

[78] A. De Santis and G. Persiano. "Zero-knowledge proofs of knowledge without interaction". In *Proceedings of the 33rd Symposium on Foundations of Computer Science*. IEEE, 1992.

[79] C. P. Schnorr. "Efficient signature generation by smart cards". *Journal of Cryptology*, 4(3):161–174, 1991.

[80] C. P. Schnorr. "Security of the 2^t-root identification and signatures". In *Advances in Cryptology – CRYPTO 1996 Proceedings*. Lecture Notes in Computer Science, Vol. 1109, N. Koblitz ed., Springer-Verlag, 1996.

[81] V. Shoup. "Lower bounds for discrete logarithms and related problems". In *Advances in Cryptology – EUROCRYPT 1997 Proceedings*. Lecture Notes in Computer Science, Vol. 1233, W. Fumy ed., Springer-Verlag, 1997.

[82] V. Shoup. "On formal models for secure key exchange (version 4)". *Cryptology ePrint Archive: Report 1999/012*, 1999. `http://eprint.iacr.org/1999/012/`.

[83] V. Shoup. "On the security of a practical identification scheme". *Journal of Cryptology*, 12(4):247–260, 1999.

[84] V. Shoup. "OAEP reconsidered". *Journal of Cryptology*, 15(4):223–249, 2002.

[85] A. Stubblefield, J. Ioannidis, and A. D. Rubin. "Using the Fluhrer, Mantin, and Shamir attack to break WEP". In *Proceedings of the Network and Distributed System Security Symposium, NDSS 2002*. The Internet Society, 2002.