ON THE SECURITY AND EFFICIENCY OF ENCRYPTION

A Thesis
Presented to
The Academic Faculty

by

Charles David Cash

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
College of Computing

Georgia Institute of Technology
December 2009

ON THE SECURITY AND EFFICIENCY OF ENCRYPTION

To Emily. Your love, dedication, and patience give me strength.

You amaze me.

ACKNOWLEDGEMENTS

I first thank my advisor, Sasha, for believing in me as a researcher and guiding me to become one.

I thank Eike and Kaoru for their friendship and collaboration during my summer visits, from which I drew inspiration.

I thank the graduate students and faculty in my department at Georgia Tech, who made my years there a pleasure.

I thank my coauthors, from whom I have learned so much. The work here was done in collaboration with Alexandra Boldyreva, Marc Fischlin, Eike Kiltz, Chris Peikert, Amit Sahai, Victor Shoup, and Bogdan Warinschi.

I thank my parents and my brother, whose love and support have enriched my life beyond words. I love you all dearly.

TABLE OF CONTENTS

SUMMARY

This thesis is concerned with the design and analysis of practical provably-secure encryption schemes. We give several results that include new schemes with attractive tradeoffs between efficiency and security and new techniques for analyzing existing schemes. Our results are divided into three chapters, which we summarize below.

The Twin Diffie-Hellman Problem. We describe techniques for analyzing encryption schemes based on the hardness of Diffie-Hellman-type problems. We apply our techniques to several specific cases of encryption, including identity-based encryption, to design a collection of encryption schemes that offer improved tradeoffs between efficiency and evidence for security over similar schemes. In addition to offering quantitative advantages over prior work in this area, our technique also simplifies security proofs for these types of encryption schemes.

Our main tool in this chapter is the notion of *Twin Diffie-Hellman Problems*, which provide an intermediate step for organizing security reductions and reveal very simple variants of known schemes with correspondingly simple, but non-obvious, analyses.

Non-Malleable Hash Functions. We consider security proofs for encryption that are carried out in the *random oracle model*, where one declares that a scheme's hash functions are "off limits" for an attacker in order to make a proof go through. Such proofs leave some doubt as to the security of the scheme in practice, when attackers are free to exploit weaknesses in the hash functions. A particular concern is that a scheme may be insecure in practice no matter what very strong security properties its real hash functions satisfy.

We address this doubt for an encryption scheme of Bellare and Rogaway by showing that, using appropriately strong hash functions, this scheme's hash functions can be *partially instantiated* in a secure way.

Our primary tool for doing this, which is of interest for other applications, is a new notion of security for hash functions that is useful for both hash function designers and protocol designers. Our notion, called *non-malleability of hash functions*, is similar to related notions of non-malleability in cryptography, but we must address some technical difficulties in transferring this intuitive notion to the hash function setting.

Circular Security. We consider a notion of security, called *circular security*, where an adversary may obtain encryptions of a scheme's secret keys as messages. Standard security notions do not guarantee secrecy under this type of attack.

We give constructions of circular-secure encryption based on hard computational learning problems. Our schemes provide alternatives to the only other known standard-model scheme, which is based on Decision Diffie-Hellman. Our schemes are natural relatives of existing non-circular-secure encryption schemes from the same learning problems, and in fact they are *as efficient* as the other learning-based schemes. This is in contract to the Decision Diffie-Hellman-based scheme, which "pays" an efficiency hit over similar schemes from the same problem.

We also consider foundational questions regarding circular security. We settle the following question: *Once trivialities are addressed, is every standard-secure encryption scheme also circular-secure?* We prove that this is not the case by describing and analyzing a counterexample.

CHAPTER I

INTRODUCTION

This thesis is concerned with the design and analysis of encryption schemes. This chapter provides some context for our work and then summarizes our results, which follow in subsequent chapters.

1.1 Encryption and Provable Security

An *encryption scheme* is a cryptographic tool for exchanging messages secretly over a public channel. The classical, ad-hoc approach for designing secure encryption worked by suggesting schemes that were intuitively secure and then patching any vulnerabilities that were found. This process provided little assurance that a more clever attack would not be found when the scheme was deployed. Moreover, since the search for attacks was not systematic, it was hard to estimate if a scheme would provide secrecy when used in a particular way, leading to problems when encryption schemes were used as components in more complicated protocols.

In this thesis we follow an alternative, and now quite standard, paradigm called *provable security* [48]. This approach provides evidence for a scheme's security by connecting attacks against the scheme to efficient solutions to intractable computational problems. In some sense, one is able to prove that the *only* route towards breaking a scheme involves solving a problem that is believed to be intractable. Furthermore, provable security results guide the usage of schemes by providing precise definitions of the environment in which one can expect security. This approach results in theorems similar to *All semantic information contained in the encrypted message is hidden from any polynomial-time eavesdropping adversary, unless one can factor integers efficiently.*

A provable security result is a mathematical theorem which can be verified like any other theorem. The interpretation and evaluation of such a result, however, involves several orthogonal issues that complicate their use in practice. In order to more effectively compare provably-secure encryption schemes, we fix the following three somewhat informal metrics. Modern research in encryption can pursue a variety of goals, but these metrics seem fairly complete, in that most effort in recent years can be explained as attempts to improve these metrics.

Threat model. A *threat model* is a description of an adversary's capabilities and goals. In the context of provable security, a threat model is a rigorous definition involving algorithms interacting through well-defined interfaces. A threat model describes the type of access an adversary gets to a schemes' components and what type of secrecy is expected in a precise way.

It is desirable to analyze schemes in threat models that reflect real applications as accurately as possible, but analyses in weaker or even stronger models can be fruitful for theoretical understanding and intermediate results.

Security in stronger threat models usually requires some cost in terms of the other metrics. For example, public-key encryption schemes secure against an active attack, where an adversary can request decryptions of ciphertexts, usually require about double the computational resources to encrypt a message as schemes only secure against passive attacks, where an adversary does not get any decryption queries.

Efficiency. Applications prefer that a scheme not use too many computational resources, that its keys not be very large, and that it not introduce too much communication overhead. Depending on the context, we could be interested in the asymptotic behavior of the scheme or instead in more detailed measures.

Reduction quality. All provable security results that we will consider consist of a *reduction*[1] of security to a particular computational problem. That is, they will provide a process by which one can convert an adversary $\mathcal{A}$ that attacks the scheme into an algorithm $\mathcal{B}$ for solving a hard problem. The aim is for the complexity and correctness of $\mathcal{B}$ to be favorably related to the same measures of $\mathcal{A}$.

Evaluating a reduction can be subtle and somewhat qualitative. Below we briefly give an abstract viewpoint to identify the relevant issues.

The intuition for interpreting reductions goes as follows. Suppose the problem we are reducing to is believed to be hard for algorithms with complexity β. Further suppose that we wish to prove that no adversary $\mathcal{A}$ can succeed in breaking our scheme unless it has very high complexity. A security reduction shows how to convert any successful adversary $\mathcal{A}$ with complexity α into a correct algorithm $\mathcal{B}$ with complexity $f(\alpha)$ for some increasing function f. Of course, we must have $f(\alpha) \geq \beta$ because the computational problem is believed to be hard for algorithms with complexity β. Now, using the reduction, we get the security guarantee we sought: If $\mathcal{A}$ is an adversary with complexity α that successfully breaks the scheme, then we have a lower bound on α, namely $f(\alpha) \geq \beta$.

This description reveals one feature of reductions that is usually called *tightness*. The tightness of a reduction is measured, abstractly, by how the function f behaves. If f is, say, defined by $f(x) = x^{100}$, we only get that $\alpha \geq \beta^{1/100}$, a bound which may not rule out the possibility that an efficient adversary could break the scheme. When f is the identity (or very close to it), however, we get a so-called tight reduction where $\alpha \geq \beta$. This gives a better lower bound on α and thus a more reassuring result for security.

[1]This is a reduction in a sense inspired by Turing reductions in complexity theory. We remark that there is an extensively studied area of cryptography dealing with information-theoretic security that does not need the concept of a reduction. The problems in encryption that we consider, however, are provably impossible to resolve with information-theoretic security.

Above we started with the assumption that some problem is not solvable by algorithms running with a certain complexity. If the assumption turns out to be false then the reduction is vacuous and has no bearing on security. Our belief in assumptions is usually based on the failure of experts to find efficient algorithms for the problem. It is not uncommon, however, to give security reductions to problems that have not been studied closely (or at all), but also appear to be hard in the designer's best estimation. In some cases, the problem is quite unnatural and even *interactive*, meaning that the problem involves communicating with some sort of oracle instead of processing an input as usual. Interactive problems do not usually arise in fields like number theory, and it can be very difficult to guess if an interactive problem resists known techniques. Reductions to such problems are better than nothing, but it is clearly preferable to deal with the problems that have received attention from research communities, often for purposes that have nothing to do with cryptography.

A final issue in reductions which will receive attention in this thesis is the use of *ideal components*. In these types of results, one declares a component of the scheme "off limits" for the adversary, and analyzes the scheme in a model where that component behaves in an ideal way. The most common ideal component used is a so-called *random oracle* [11], which idealizes a cryptographic hash function. Clearly, the use of ideal components is undesirable because, in reality, an adversary is free to attack every part of the system. The extent to which we should doubt results that use ideal components is debatable, but results without them are always preferable.

1.2 Contributions of This Thesis

This thesis develops techniques for constructing and analyzing practical provably-secure encryption schemes. A particular emphasis is placed on improving the balance between efficiency, like the running time of the algorithms involved, with the quality of security provided. In the remainder of this chapter, we provide an overview of our results, which are divided into three chapters.

1.2.1 Twin Diffie-Hellman Problems

Some very natural and practical encryption schemes are only known to be secure under contrived, interactive assumptions that have never been studied by research communities outside of cryptography. A prominent example of such a scheme is *hashed ElGamal*, which has been standardized and widely deployed in applications. Known security proofs for hashed ElGamal all depend on interactive variants of assumptions related to computing discrete logarithms in prime order groups. Another example is the basic Boneh-Franklin identity-based encryption scheme, which is in a similar state, but with an even less-studied interactive assumption in pairing groups[2].

In this chapter we show how to modify these, and other, schemes to base security on a new class of interactive assumptions, which we call *twin assumptions*. The advantage of our modified schemes is that the twin assumptions are *equivalent* to standard, relatively well-studied, non-interactive assumptions, and therefore schemes that rely on twin assumptions are actually only relying on standard non-interactive assumptions.

Other techniques are known for modifying schemes like hashed ElGamal to rely on standard assumptions, but our results provide an alternative with some attractive properties. Unlike all previous modifications, our new schemes do not add any extra redundancy or integrity checks to the ciphertexts of the original scheme. Our modifications are also quite simple to implement. Finally, the techniques in the security proofs for our schemes are also relatively simple and portable, and have found further applications.

We now highlight our techniques for the variant of hashed ElGamal in slightly more detail. Our other applications will follow this template and be intuitively very

[2]Here we are referring to chosen-ciphertext security, which is defined in §2. Chosen-ciphertext security is a strong notion of security sufficient for most common applications.

similar. Recall that hashed ElGamal is known to be (chosen-ciphertext) secure, in the random oracle model, under the so-called *strong Diffie-Hellman assumption.* This is an interactive version of the Diffie-Hellman assumption which assumes that the Diffie-Hellman problem is hard, even with the help of a decision oracle for the problem. The decision oracle could, in principle, make the problem easier and thus the assumption stronger and less desirable.

Our variant of hashed ElGamal is called *twin ElGamal.* We prove that twin ElGamal is secure under a new interactive assumption called the *strong twin Diffie-Hellman assumption.* So far, not much has been gained, as the new assumption is seemingly plausible, but it is still interactive and not well studied. The advantage comes when we prove that the strong twin Diffie-Hellman problem is intractable as long as the *ordinary* Diffie-Hellman problem is intractable. It follows that twin ElGamal is secure, in the random oracle model, as long as the ordinary Diffie-Hellman assumption holds. Compared to other ElGamal variants with similar security properties, our scheme is attractive in that it has very short ciphertexts and a very simple and tight security proof. It is also competitive in terms of computational overhead.

Our techniques are portable enough to be applied other Diffie-Hellman-like problems, with similar gains. Applications of these other problems include a new variant of *Cramer-Shoup encryption* [32] with a very simple security proof, without random oracles, under the *hashed* decisional Diffie-Hellman assumption and a new variant of *Boneh-Franklin identity-based encryption* [21], with very short ciphertexts, and a simple and tighter security proof in the random oracle model, assuming the bilinear Diffie-Hellman problem is hard.

Beyond encryption, our techniques also give a new variant of *Diffie and Hellman's non-interactive key exchange protocol* [35], which is secure in the random oracle model assuming the Diffie-Hellman problem is hard and a very simple and efficient method of securing a *password-authenticated key exchange protocol of Abdalla and Pointcheval*

[2] against server compromise, which can be proved secure, in the random oracle model, under the Diffie-Hellman assumption. See [30] for details on key exchange applications.

1.2.2 Non-Malleable Hash Functions and Their Application to Encryption

In this chapter we turn to the issue of random oracles [11] in security reductions. Recall that a reduction is carried out in the random oracle model when all parties have access to an idealized version of a hash function that maps any input to a fresh random string. In reality, however, no such function is available, and one hopes that a cryptographic hash function will approximate the random oracle's behavior.

We are particularly interested in the question *When can a random oracle be implemented securely?* A better understanding of this question will put the random oracle methodology – which we employ in some of the results in Chapter 3 – on a firmer foundation.

Prior work [27] has shown that we cannot *always* securely implement a random oracle. More precisely, it is known that there are protocols which are provably secure, under standard assumptions, in the random oracle model, but which also become insecure when the random oracle is implemented with *any* real hash function. But these protocols are arguably contrived, and they give little insight to question of if common uses of random oracles can be instantiated.

Below we study the possibility of instantiating random oracles in encryption schemes and find that a certain type of random oracle usage *can* be instantiated in secure way. To address this question, we develop a theoretical tool which we hope is of independent interest. Our tool is a new security notion for hash functions called *non-malleability.* Interestingly, this concept turns out to be similar, in spirit, to non-malleability for other cryptographic primitives, but we uncover some fundamental differences. Because of these differences, and because of the potential for wider uses, we suggest that non-malleability of hash functions is interesting in its own rite.

We start with an overview of non-malleability and the need for a useful definition applied to hash functions, and then describe our contributions.

Informally, non-malleability of some function f is a cryptographic property which requires that learning $f(x)$ for some x does not facilitate the task of generating some $f(x^*)$ so that x^* is related to x in a non-trivial way. This notion is especially useful when f is used to build higher-level multi-user protocols where non-malleability of the protocol itself is crucial (e.g., for voting or auctioning). Non-malleability has been rather extensively studied for some cryptographic primitives. For example, both definitions as well as constructions from standard cryptographic assumptions are known for encryption, commitments and zero-knowledge [36, 13, 70, 33, 40, 72, 8, 34, 63, 64, 9]. Non-malleability in the case of other primitives, notably for one-way functions and for hash functions,[3] has only recently surfaced as a crucial property in several works [17, 18, 28, 39], which we discuss below.

Many cryptographic schemes are only known to be secure in the random oracle model [11], where one assumes that a hash function behaves as a truly random function to which every party has access to. It is well-known that such proofs do not strictly guarantee security for instantiations with hash functions whose only design principles are based on one-wayness and/or collision-resistance, because random functions posses multiple properties the proofs may rely on. In fact, some recent results [17] showed that security of the widely deployed RSA-OAEP encryption scheme, provably secure in the random oracle model, can be compromised if the underlying hash function is malleable. A related example is the abstraction used to model hash functions in symbolic (Dolev-Yao) security analysis. In this setting it is *axiomatized* that an adversary can compute some hash only when it knows the underlying

[3]We aggregate both one-way functions and hash functions under the term hash functions for simplicity.

value. Clearly, malleable hash functions do not satisfy this axiom. Therefore, non-malleability for hash functions is necessary in order to ensure that symbolic analysis is (in general) sound with respect to the standard cryptographic model. Otherwise, real attacks that use malleability can not be captured/discovered in the more abstract symbolic model.

In a different vein, and from a more conceptual perspective, higher-level protocols could potentially benefit from non-malleable hash functions as a building block. A recent concrete example is the recommended use of such non-malleable hash functions in a human-computer protocol for protecting local storage [28]. There, access should be linked to the ability to answer human-solvable puzzles (similar to CAPTCHAs), but it should be infeasible for a machine to maul puzzles and redirect them under a different domain to other human beings.

Hence, non-malleability is a useful design principle that designers of new hash functions should keep in mind. At this point, however, it is not even clear what the exact requirements from a theoretical viewpoint are. Therefore, a first necessary step is to find a suitable definition which is (a) achievable, and (b) applicable. The next step would be to design practical hash functions and compression functions which are non-malleable, or which at least satisfy some weaker variant of non-malleability.

In this chapter we initiate the study of non-malleable hash functions and their application to encryption. We start with the design of an appropriate security definition. Our definition follows a standard simulation paradigm of the sort that can be used to define non-malleability for encryption and commitment schemes. It turns out, however, that a careless adjustment of definitions for other primitives yields a definition for non-malleable hash functions that cannot be realized. We therefore motivate and provide a meaningful variation of the definition which ensure that the notion is achievable and may be useful in applications.

Testifying to the difference to other cryptographic primitives, we note that for non-malleable encryption the original simulation-based definition of [36] was later shown to be equivalent to an indistinguishability-based definition [13]. For our case here, finding an equivalent indistinguishability-based definition for non-malleable hash functions appears to be far from trivial, and we leave the question as an interesting open problem.

We note that our definition was shown to be *achievable* under minimal standard assumptions. More precisely, if there exist one-way functions, then there exist hash functions meeting our definition of security. These hash functions are highly impractical and should be seen as a proof-of-concept only. In particular, they employ non-interactive zero-knowledge proofs of knowledge for a general family of languages. This implies that the resulting hash functions are randomized, which we do not know to be necessary. It is still an open problem of finding a practical, deterministic solution from, say, a number-theoretic assumption. Our definition is general enough to allow such constructions.

Next we study of applicability of our definition to encryption. We show that our definition suffices for a *partial instantiation* of a very natural encryption scheme of Bellare and Rogway [11]. Partial instantiations are a subtle type of result, and their interpretation deserves some discussion. In a partial instantiation, one starts with a scheme that uses multiple random oracles, and shows how to realize *some*, but not all, of the random oracles. Thus the "instantiated" scheme is still only proven secure in the random oracle model, and still inherits all of the associated theoretical difficulties. But the gain is that we learn something about the instantiated random oracles. We precisely quantify sufficient properties of the instantiated hash for security to hold. Thus, when implementing the scheme, there is one less source of imprecision in the random oracle methodology.

We show that non-malleability is sufficient to instantiate one of the random oracles in the encryption scheme of Bellare and Rogaway [11]. We briefly recall the scheme here. The scheme is actually a generic and efficient method for encrypting with any trapdoor permutation family. A public key is an instance of the family f, with corresponding secret key set to f^{-1}, the trapdoor for f. Encryption uses two hash functions, G and H. For a message m, it selects r at random from the domain of f and computes

$$y \leftarrow f(r), \quad g \leftarrow G(r) \oplus m, \quad h \leftarrow H(r||m).$$

The ciphertext is (y, g, h). Given a ciphertext $(\hat{y}, \hat{g}, \hat{h})$ and secret key f^{-1}, decryption computes

$$\hat{r} \leftarrow f^{-1}(\hat{y}), \quad \hat{m} \leftarrow \hat{g} \oplus G(\hat{r}).$$

It outputs $\hat{m}$ if $H(\hat{r}\|\hat{m}) = \hat{h}$ holds, and it rejects if not.

This scheme is known to be semantically-secure under chosen ciphertext attacks if f is one-way and if G and H are modeled as random oracles. We show that the scheme is secure if H is instantiated with a non-malleable hash function. At a high level, we are able to show that, as long as an adversary cannot compute $H(\hat{r}||m)$ from $H(r||m)$ for $\hat{r} \neq r$, then then the scheme will not demand any other properties from H, and thus an implementation of this scheme only needs to check for this property.

We remark that the usefulness of the definition was also confirmed in independent results [39] which showed that HMAC is a secure message authentication code, assuming that the compression function is non-malleable.

1.2.3 Circular-Secure Encryption

In the first two chapters we focused primarily on semantic-security under chosen-ciphertext attack (CCA security), which is usually cited as the "de facto gold standard for encryption security." It also well known, however, that for certain natural applications, CCA security is *not* sufficient. The mismatch comes from a subtle observation whose importance is not immediately apparent: The definition only guarantees that it is safe to publish encryptions of messages that *adversaries can compute themselves*. But, in practice, we sometimes *do* encrypt messages that an adversary could not compute itself. In particular, it is sometimes necessary to encrypt messages that *depend on the secret key*.

In this chapter we consider a type of security, called *circular security*, that addresses this mismatch when applications publish encryptions of keys that form "cycles." That is, if (pk_i, sk_i) are public-key/secret-key pairs for $i = 0, \dots, n$, a cycle consists of encryptions of sk_i under $pk_{i+1 \bmod n}$ for each i. Interestingly, each of the ciphertexts alone would be safe to publish, but when considered together, the situation becomes theoretically uncertain.

This issue of key-dependent messages was noticed at the outset of complexity-based cryptography, and such use of encryption was categorically disallowed in practice. In the intervening years, however, practical applications have been found to encrypt such dangerous messages. The most common example is whole-disk encryption, where the secret key used for encryption is usually stored on the disk itself. But the issue arises in other contexts, such as a protocol of Camenisch and Lysyanskaya [25] that forces users to publish a "circular encryption" of keys to prevent them from selectively sharing their credentials. Security under key-dependent messages was also shown to be sufficient for proving soundness results relating formal models (logics) of cryptography with computational results [3].

The concerns over encrypting a key under itself are not a theoretical curiosity. The IEEE P1619 standardization project, which was charged with specifying secure methods for encrypted stored media, actually rejected a mode of operation for AES due to a realistic attack when a particular key-dependent message was encrypted [62].

Black *et al.* [14] and Camenisch and Lysyanskaya [25] independently defined versions of security under key-dependent messages. These works give constructions of symmetric-key and public-key encryption, respectively, but these are only known to be secure using the random oracle model [11]. It was subsequently shown [53] that the scheme of Black *et al.* could be insecure even when the random oracle was instantiated with a hash function meeting a variety of security notions. The first construction of an encryption scheme that is provably circular-secure, without random oracles, was given by Boneh *et al.* [22], based on the decisional Diffie-Hellman problem.

CONSTRUCTIONS FROM LEARNING PROBLEMS. The first set of results in this chapter consists of new constructions of circular-secure encryption. Our core insight is that encryption schemes based on *hard learning problems* (including those related to *worst-case lattice problems*) offer homomorphic properties that are directly applicable to achieving security for key-dependent messages.

As a result, our approach yields very *natural* encryption schemes that have significant efficiency advantages over the prior scheme of Boneh *et al.* [22]. The contrast is clearest when we compare the current "cost" of achieving security for key-dependent messages against the cost of achieving ordinary semantic security, for a given computational intractability assumption. Comparing the scheme of [22] to other semantically secure encryption schemes based on the DDH problem, the cost is dramatic: while standard encryption schemes like ElGamal can encrypt a message of about $k = \log |\mathbb{G}|$ bits (where $\mathbb{G}$ is the underlying group of the DDH problem) using a single exponentiation and one group element of overhead in the ciphertext, the scheme given in [22] requires about k exponentiations and group elements of ciphertext overheard *per bit of key material encrypted.* In contrast, our constructions are essentially *as efficient* as prior semantically secure schemes based on essentially the same hardness assumptions.

Our main results are two standard-model encryption schemes that are provably circular-secure (and more), under standard assumptions concerning the hardness of "noisy learning" problems. Specifically, we obtain:

- A *public-key* scheme based on the *learning with errors* (LWE) problem [69]. The scheme is a variant of Regev's LWE-based scheme and the more-efficient amortized version due to Peikert, Vaikuntanathan, and Waters [66], with several non-trivial modifications to facilitate the proof of security for key-dependent messages. The most efficient version takes only $\tilde{O}(n)$ amortized time per message symbol for both encryption and decryption, and the ciphertext is only a constant factor larger than the plaintext.

- A *symmetric-key* scheme based on the *learning parity with noise* (LPN) problem (c.f. [54]). The scheme is as efficient as a recent proposal for encrypting with LPN [45], which was proved secure only in the standard sense, i.e., without key-dependent messages.

In addition to circular security, both of our schemes actually enjoy *key-dependent message security* with respect to arbitrary affine functions (over the message space) of the secret key, similarly to the scheme of [22].

Informally, the LWE problem (for a dimension n and modulus q) is to recover a secret vector $\mathbf{s} \in \mathbb{Z}_q^n$ given arbitrarily many "noisy random inner products" $(\mathbf{a}_i, b \approx \langle \mathbf{a}_i, \mathbf{s} \rangle) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$ where the $\mathbf{a}_i \in \mathbb{Z}_q^n$ are uniform and independent; LPN is the special case where $q = 2$. These problems have been studied extensively in several works, and much evidence suggests that no efficient algorithm can solve them with better than negligible probability, even using quantum computation. The best known algorithms for these problems require $2^{O(n \log q / \log n)}$ time and space [15]. They are directly related to the famous problem of decoding random linear codes, and the LPN problem also occupies a central position in learning theory: an efficient algorithm for it could be used to learn several important concept classes, including 2-DNF formulas, juntas, and *any* function with a sparse Fourier spectrum [37].

The LWE problem is also especially attractive as the basis for a cryptographic hardness assumption due to its remarkable connection to *worst-case* lattice problems: Regev [69] showed that solving LWE (for certain Gaussian-like error distributions) is as hard as *quantumly* solving a few well-studied lattice problems, such as the approximate shortest vector problem GapSVP. Recently, Peikert [65] also gave a *classical* reduction from GapSVP (and variants) to LWE.

A Counterexample. Prior work leaves open the possibility that every encryption scheme that is semantically secure in a standard sense is also circular-secure, once some trivialities are taken care of. This would imply that our constructions and those of [22] are unnecessary, as one could use simply use any standard scheme for applications requiring circular security. Either proving this equivalence or ruling it out was left as an open question by Boneh et al [22].

We summarize what is known. Here we deal with public-key case. It is a simple matter to give an encryption scheme that is semantically secure (under minimal assumptions), but cannot securely encrypt its own key – that is, given a an encryption of a secret key under its corresponding public key, an adversary can mount a chosen-plaintext attack against the scheme. This counterexample simply implements a "check" in the encryption algorithm that detects of the input message is in fact the secret key. If so, it can misbehave in some way, like outputting the secret key. Standard security is preserved because an adversary is unlikely to submit the secret key in a challenge query during a chosen-plaintext attack.

This counterexample also guides a simple patch that enables any secure encryption scheme to also securely encrypt its own secret key. The idea is for the scheme to check if its input message is its own secret key, and if it is, then encrypt a special symbol instead of the key. When decryption detects this special symbol, it can output the secret key.

This situation does not address more interesting practical cases, however. In most situations we can avoid encrypting a secret key under its own public key, but it seems more difficult to avoid circular encryptions of size say 2. To prevent such a "2-cycle," independent users must coordinate which keys they have encrypted in the past, which may be very difficult or impossible in a complex distributed system. But the counterexample above does not show that this type of cycle is dangerous.

Thus the question remains: *Are encrypted cycles of size 2 or more potentially dangerous?*. If we can prove that, in general, they are not, then our work on circular-secure encryption schemes is rendered obsolete, as one can take any standard scheme and apply the patch to fix "1-cycles."

This question was posed by Boneh *et al.* [22]. They provided a counterexample to show that a *one-way* secure encryption scheme will not always remain one-way after a 2-cycle is published. Finding a counterexample for full semantic security seems more difficult. This is because the keys and ciphertexts involved in an encrypted cycle are all generated with independent randomness, and moreover the scheme must be semantically secure in the standard sense, which rules out the technique of Boneh *et al.*, where each ciphertext leaks a significant amount of information, but not enough to violate one-way security.

We settle this question by showing that not every semantically-secure encryption scheme is also circular secure. We present a simple encryption scheme that is provably semantically secure under chosen plaintext attacks, under a standard assumption called the *symmetric external Diffie-Hellman (SXDH) assumption* [6], but breaks completely when an encrypted two-cycle is published.

Our counterexample works by attaching a tag to the ciphertext of normally secure encryption scheme, so that tags generated while encrypting a cycle will interlock and help an attacker. Without an interlocking tag, however, a given tag is (provably) useless to the adversary.

1.3 Organization and Credits

In Chapter 2 we review some basic notions from probability and cryptography. Our results follow in the remaining three chapters.

The results in Chapter 3 are joint work with Eike Kiltz and Victor Shoup. They were published, along with further results, in *EUROCRYPT 2008* [30] and will appear in *J. Cryptology*. The results in Chapter 4 are joint work with Alexandra Boldyreva, Marc Fischlin and Bogdan Warinschi, and will appear at *ASIACRYPT 2009* [16]. The constructions in Chapter 5 are joint work with Chris Peikert and Amit Sahai, and will appear at *CRYPTO 2009* in a paper including independent results by Benny Applebaum [5]. The counterexamples in Chapter 5 are currently unpublished.

CHAPTER II

PRELIMINARIES

In this chapter we review the basic notation and definitions used in subsequent chapters.

2.1 Notation

Let f, g be positive real-valued functions on $\mathbb{Z}^+$. We say that $f(n) = O(g(n))$ if $\lim_{n\to\infty} \frac{f(n)}{g(n)}$ is a constant, and $f(n) = o(g(n))$ if this limit is 0. We write $f(n) = \Omega(g(n))$ if $g(n) = O(f(n))$, and $f(n) = \omega(g(n))$ if $g(n) = o(f(n))$. A function f is *negligible* if $f(n) = o(n^{-c})$ for every $c > 0$. We say that a sequence of events $\{E_n\}$ occurs with *overwhelming probability* if $\Pr[\neg E_n]$ is negligible.

Let Ψ be a probability distribution. We write Ψ^n for the n-fold product distribution of n i.i.d. copies of Ψ. We write $x \overset{\$}{\leftarrow} \Psi$ to indicate that x is sampled according to Ψ. If X is a set, then we write $x \overset{\$}{\leftarrow} X$ to indicate that x is sampled uniformly at random from X.

For $x \in \{0,1\}^*$, we write $|x|$ to mean the length of x. For equal-length bit strings x, y, we write $x \oplus y$ for the bit-wise exclusive or (XOR) of x and y.

We write $[n]$ for $\{1, \ldots, n\}$.

2.2 Algorithms

When describing an algorithm, we write $Y \leftarrow \mathcal{A}(X)$ to mean that Y is the value output when $\mathcal{A}$ is run on input X. We interpret a randomized algorithm to be deterministic algorithm with an extra input that will represent random choices. We will refer to this extra input as the algorithm's *internal randomness.* If $\mathcal{A}$ is a randomized algorithm, we write $y \overset{\$}{\leftarrow} \mathcal{A}(x)$ to denote that y is a random variable with distribution induced by running $\mathcal{A}$ with input x and a uniformly chosen internal randomness (where uniform here means uniform over $\{0,1\}^{p(|x|)}$ for some appropriate function p).

We will model our cryptographic schemes and adversaries as algorithms in an unspecified reasonable model of computation, like RAM machines. When we are interested in more detailed run-time analysis, we will simply list the dominating operations which can be translated into any model of computation.

2.3 Encryption schemes

In some of our applications we assume that a security parameter, denoted λ, is an implicit input, in unary[1], to every algorithm (including adversaries). When an algorithm takes no other inputs, we will write the security parameter as an explicit input to eliminate ambiguities.

2.3.1 Public-Key Encryption

A *public-key encryption scheme* PKE is composed of three algorithms.

- A probabilistic polynomial-time key-generation algorithm G, which takes $\lambda \in \mathbb{Z}^+$ as input and outputs a secret key/public key pair (sk, pk).
- A probabilistic polynomial-time encryption algorithm E, which takes as input $\lambda \in \mathbb{Z}^+$, a public key pk and a message m, and outputs a ciphertext c.
- A deterministic polynomial-time decryption algorithm D, which takes as input $\lambda \in \mathbb{Z}^+$, a secret key sk and a ciphertext c, and outputs a message m or rejects.

We will distinguish the key inputs to E and D by writing them as subscripts, as in $c \overset{\$}{\leftarrow} \mathsf{E}_{pk}(m)$ and $m \leftarrow \mathsf{D}_{sk}(c)$. We will usually restrict the set of messages allowed to some set $\mathsf{M}^{(\lambda,pk)}$.

[1]The intention is that every algorithm will run in time related to λ itself, not the length of a binary description of λ.

To be useful, any encryption scheme must properly decrypt ciphertexts. We call this requirement *correctness*. More precisely, we require that for all messages $m \in \mathsf{M}^{(\lambda,pk)}$, we have $m = m'$ with overwhelming probability, where pk and m' are computed as

$$(sk, pk) \xleftarrow{\$} \mathsf{G}(1^\lambda); \quad c \xleftarrow{\$} \mathsf{E}_{pk}(m); \quad m' \leftarrow \mathsf{D}_{sk}(c).$$

Note that this probability is over the internal randomness of G and E.

2.3.2 Symmetric-Key Encryption

A *symmetric-key encryption scheme* is a pair of algorithms.

- A probabilistic polynomial-time algorithm E that takes as input $\lambda \in \mathbb{Z}^+$, a key k, and a message m and outputs a ciphertext c.

  Here we assume k and m are a bitstrings of lengths determined by λ. It will sometimes be convenient to allow any $m \in \{0,1\}^*$.

- A deterministic polynomial-time algorithm D that takes as input $\lambda \in \mathbb{Z}^+$, a key k, and a ciphertext c. It outputs a message m.

We will also require symmetric-key encryption schemes to be correct. In this case, we mean for every message m and key k, we require that, with overwhelming probability, $m = m'$, where m' is computed as

$$c \xleftarrow{\$} \mathsf{E}_k(m); \quad m' \leftarrow \mathsf{D}_k(c).$$

This probability is only over the internal randomness of E.

2.4 Security notions for encryption

2.4.1 Security of Public-Key Encryption

We review two of the standard definitions of security for public-key encryption. The first is referred to as *semantic security under chosen ciphertext attack* [68], which is a strong notion sufficient for most common applications.

Definition 1. *Let* PKE *be a public-key encryption scheme. Consider the following game, played between an adversary* $\mathcal{A}$ *and another algorithm, referred to as a challenger. The game is parameterized by the security parameter* λ*.*

1. *The challenger samples* $(pk, sk) \xleftarrow{\$} \mathsf{G}(1^\lambda)$ *and gives* pk *to* $\mathcal{A}$*.*
2. $\mathcal{A}$ *makes a number of* decryption queries *to the challenger, where each such query is a ciphertext* $\hat{c}$*. For each query, the challenger computes* $\hat{m} \leftarrow \mathsf{D}_{sk}(\hat{c})$ *are returns* $\hat{m}$ *to* $\mathcal{A}$*.*
3. $\mathcal{A}$ *makes one* challenge query*, which is a pair of equal-length messages* (m_0, m_1)*. For this query, the challenger chooses* $b \in \{0,1\}$ *at random, computes* $c \xleftarrow{\$} \mathsf{E}_{pk}(m_b)$*, and returns* c *to* $\mathcal{A}$*.*
4. $\mathcal{A}$ *makes more* decryption queries*, just as in step 2, but with the restriction that* $\hat{c} \neq c$*.*
5. $\mathcal{A}$ *outputs* $\hat{b} \in \{0,1\}$*.*

We define the advantage of the adversary, $\mathsf{AdvCCA}_{\mathcal{A},\mathsf{PKE}}(\lambda)$ *to be* $|\Pr[\hat{b} = b] - 1/2|$*.*

In §3 we will consider schemes that do not explicitly use a security parameter. The definition above can also be modified to work without using a security parameter. In particular, key generation produces its output with taking *any* input, and we define advantage as a real number in $[0,1]$ (and not a function of λ).

We will try to bound the quantity $\mathsf{AdvCCA}_{\mathcal{A},\mathsf{PKE}}(\lambda)$ precisely when possible. However, in several contexts it suffices to study the asymptotic behavior of the advantage as a function of λ. In this case, the scheme PKE is said to be *secure against chosen ciphertext attack* if for all efficient adversaries $\mathcal{A}$, this advantage $\mathsf{AdvCCA}_{\mathcal{A},\mathsf{PKE}}(\lambda)$ is negligible (as a function of λ).

In some cases we will be interested in the following weaker type of security called *semantic security under chosen-plaintext attack* [48].

Definition 2. *Chosen-plaintext security is defined* exactly *the same as above, except that steps* 2 *and* 4 *are skipped in the game. We denote the adversary's advantage in this game by* $\mathsf{AdvCPA}_{\mathcal{A},\mathsf{PKE}}(\lambda)$.

Chosen-plaintext security is weaker than chosen-ciphertext security, but it is still sufficient for some applications and typically easier to achieve. In the development of secure encryption, chosen-plaintext security can see be seen as a step towards chosen-ciphertext security.

2.4.2 Security of Symmetric-Key Encryption

Next we consider security definitions for symmetric-key encryption. Some subtleties can arise in moving from public-key encryption, such as how we allow the adversary to request encryptions of messages. This issue has been explored in detail in other works [10]. We will avoid this distraction by achieving a reasonably strong version of security when designing schemes, and demanding a relatively weak version when selecting components for our schemes. We start with the stronger notion, which is very similar to chosen-ciphertext security in the public-key case.

Definition 3. *Let* SE *be a symmetric-key encryption scheme. Consider the following game, played between an adversary* $\mathcal{A}$ *and another algorithm, referred to as a challenger. The game is parameterized by the security parameter* λ.

1. *The challenger samples a random key* k.

2. $\mathcal{A}$ *makes a number of* encryption queries *and* decryption queries *to the challenger. Encryption queries consist of a message* m*; to process these, the challenger computes* $c' \overset{\$}{\leftarrow} \mathsf{E}_k(m)$ *and returns* c'. *Decryption queries consist of a ciphertext* $\hat{C}$*; the challenger processes these by computing* $\hat{m} \leftarrow \mathsf{D}_{sk}(\hat{c})$ *and returning* $\hat{m}$ *to* $\mathcal{A}$.

3. $\mathcal{A}$ *makes one* challenge query*, which is a pair of equal-length messages* (m_0, m_1).

For this query, the challenger chooses $b \in \{0,1\}$ at random, computes $c \xleftarrow{\$} \mathsf{E}_k(m_b)$, and returns c to $\mathcal{A}$.

4. *$\mathcal{A}$ makes more encryption and decryption queries, just as in step 2, but with the restriction that $\hat{c} \neq c$ in decryption queries.*

5. *$\mathcal{A}$ outputs $\hat{b} \in \{0,1\}$.*

We define the advantage of the adversary $\mathsf{AdvCCA}_{\mathcal{A},\mathsf{SE}}(\lambda)$ *to be* $|\Pr[\hat{b} = b] - 1/2|$.

We say that SE is *secure against chosen-ciphertext attack* if $\mathsf{AdvCCA}_{\mathcal{A},\mathsf{SE}}(\lambda)$ is negligible for all polynomial-time adversaries $\mathcal{A}$.

In Chapter 3 we will only require symmetric-key encryption that satisfies a weaker notion of security called *one-time semantic security under chosen-ciphertext attack.* This notion is defined exactly as chosen-ciphertext security for symmetric-key encryption, but with the following differences in the security game. This version completely skips step 2, and in step 4, we allow the adversary to issue decryption queries but no encryption queries. We denote the advantage of an adversary $\mathcal{A}$ against scheme SE by $\mathsf{AdvOTCCA}_{\mathcal{A},\mathsf{SE}}(\lambda)$ in this case. We will sometimes call schemes that satisfy this notion *symmetric ciphers*, in order to acknowledge that one-time CCA schemes may have deterministic encryption algorithms.

2.5 The Random Oracle Model

Here we formalize the idea of the *random oracle model* [11].

To analyze a protocol in the random oracle model, we consider a modified version of the appropriate attack model. We modify the model to allow all algorithms – including the adversary, the challenger, and the protocol's algorithms – to access an oracle that computes a random function. More precisely, before the game starts, a random function $\mathsf{H} : \{0,1\}^* \to \{0,1\}^{n(\lambda)}$ is chosen, and then all algorithms can query H. Alternatively, one can imagine that the game manages the output values of H via *lazy sampling.* This means that the game maintains a list of past input values to H that associates them with previous output values. Whenever an algorithm queries H with an input that has not yet been seen, the game selects a random value in $\{0,1\}^n$, records it in the table and returns that value.

We indicate that we are using the random oracle version of an attack model by writing ro in the notation for advantage, as in $\mathsf{AdvCCA}^{\mathrm{ro}}_{\mathsf{PKE},\mathcal{A}}(\lambda)$.

2.6 Security Proofs Using Games

All of the security reductions in this thesis will be organized into a sequence of games. We will follow an approach similar to the one described by Shoup [74], with some inspiration from the closely related approach of Bellare and Rogaway[12]. The primary goal of this approach is to simultaneously make the proofs easier to read and to verify. An alternative approach, like presenting a monolithic reduction for the entire proof, seems inferior on both counts.

We provide a brief overview of game-based proofs. The references above provide a detailed discussion of the nuances involved. In each of our proofs, we will start with a game determined by the security definition and the scheme at hand. We will then define a sequence of games, where Game i will be a relatively simple modification of Game $i-1$. The idea is that, because we have only changed one portion of the game, the behavior of the adversary can be shown to be very similar in the two games. We continue with games until we arrive at a game that is trivial to analyze. In our proofs, this final game will be a version of the CCA game where the challenger ignores the challenge query messages submitted by the adversary. It is then trivial to see that no adversary can win the final game, and then by collecting the relationships between all of the games, we can conclude that no efficient adversary could win the original game with any significant advantage.

CHAPTER III

THE TWIN DIFFIE-HELLMAN PROBLEM

In this chapter we carry out the plan outlined in §1.2.1. We develop techniques for designing encryption schemes from *Diffie-Hellman problems*. Our technique can be best understood as basing schemes on *a new class of computational problems*, which we call *twin Diffie-Hellman problems*. However, as we will explain below, schemes based on our new problems are just as well relying on the hardness of more basic, well studied problems. We show how to use our technique to give several constructions with attractive features over schemes with similar security properties. As an additional benefit, our technique greatly simplifies security proofs for these types of schemes.

We start with some definitions local to this chapter. In subsequent sections we derive our main result and then show how to apply it to encryption in several contexts.

3.1 Preliminaries

3.1.1 Asymptotic Versus Fixed Parameter Analysis

The definitions of security in §2 used a security parameter to talk about asymptotic versions of security. In this chapter we dispense with the security parameter in order to simplify the discussion. In order to precisely define notions like *intractable*, one needs to extend these results to the asymptotic case in a natural way, such as the *group scheme* approach used by Cramer and Shoup [32]. We stress that extending our results to this setting is quite straightforward.

3.1.2 Target-Collision Resistant Hash Functions

We briefly recall the concept of *target collision-resistant hash functions.* Let $\{\mathsf{T}^{(\lambda)}\}$ be a sequence of probability distributions on functions from $\{0,1\}^*$ to $\{0,1\}^{n(\lambda)}$. We assume that there is a polynomial-time algorithm for sampling from $\mathsf{T}^{(\lambda)}$ and a polynomial-time algorithm for evaluating $T(x)$, given the description of a sample T.

To define target collision-resistance we use the following game, which is played between an adversary and a challenger. It is indexed by $\lambda \in \mathbb{Z}^+$.

1. The adversary gives $x \in \{0,1\}^*$ to the challenger.
2. The challenger runs the sampler to draw a sample T from $\mathsf{T}^{(\lambda)}$. It gives the description of T to the adversary.
3. The adversary gives a second element $x' \in \{0,1\}^*$ to the challenger.

We define the advantage $\mathsf{AdvTCR}_{\mathcal{A},\mathsf{T}}(\lambda)$ of $\mathcal{A}$ as $\Pr[x \neq x' \ \wedge \ T(x) = T(x')]$. We say that $\mathsf{T}^{(\lambda)}$ is *target collision-resistant* if $\mathsf{AdvTCR}_{\mathcal{A},\mathsf{T}}(\lambda)$ is negligible for every poly-time $\mathcal{A}$. In our schemes below, we assume that a sample T from $\mathsf{T}^{(\lambda)}$ is implicitly available in the public parameters and simply write $\mathsf{T}(x)$ when evaluating the hash function.

3.1.3 Identity-Based Encryption

Identity-based encryption [73] is a version of encryption meant to mitigate problems with key distribution. Instead of using one public key per user, it allows one to use any identity (i.e., bit string) as a public key. The encryptor only needs to obtain a master public key instead of a new public key for each user.

An IBE scheme consists of algorithms for master key generation, user key generation, encryption, and decryption. The master key generation algorithm outputs a random private/public master key pair. The user key generation algorithm uses the private master key and outputs a private user key for any identity. To encrypt a message for a user, one inputs the master public key and that user's identity to the encryption algorithm. Decryption then uses the user's private key to recover the message.

The concept of chosen ciphertext security naturally adapts to IBE. For an adversary $\mathcal{A}$ and IBE scheme IBE, the game is as follows:

1. The challenger generates a master public key/secret key pair, and gives the master public key to $\mathcal{A}$.

2. $\mathcal{A}$ makes *user secret key queries* and *decryption queries* to the challenger. Each user secret key query is an identity $\hat{id}$, and the challenger responds by running the user secret key generation on $\hat{id}$ and sending that key to $\mathcal{A}$. Each decryption query is an identity $\hat{id}$ and ciphertext $\hat{c}$, and the challenger responds by decrypting $\hat{c}$ using the secret key for id and sending the result to $\mathcal{A}$.

3. $\mathcal{A}$ makes one *challenge query*, which is an identity id and a pair of equal-length messages (m_0, m_1). The challenger chooses $b \in \{0,1\}$ at random, encrypts m_b for id, and sends the resulting ciphertext c to $\mathcal{A}$. $\mathcal{A}$ is not allowed to choose id after requesting the user private key for id in the previous step.

4. $\mathcal{A}$ makes more *user secret key queries* and *decryption queries*, just as in step 2, but with the restriction that $\hat{id} \neq id$ in user secret key queries and $(\hat{id}, \hat{c}) \neq (id, c)$ in decryption queries.

5. $\mathcal{A}$ outputs $\hat{b} \in \{0,1\}$.

As before, we define the advantage $\mathsf{AdvCCA}_{\mathcal{A},\mathsf{IBE}}$ as $|\Pr[\hat{b} = b] - 1/2|$. When a hash function is modeled as a random oracle, we denote the advantage by $\mathsf{AdvCCA}^{\mathrm{ro}}_{\mathcal{A},\mathsf{IBE}}$.

3.2 Hashed ElGamal Encryption and the Diffie-Hellman Problem

To motivate our results below, we review the "hashed" ElGamal encryption scheme [1], which, along with its variants, has been intensely studied and is widely deployed in practice [41, 61, 75, 31, 7, 1, 32, 55]. Below we will note how a natural issue with proving the security of hashed ElGamal leads to a need for decision oracles in security reductions.

The hashed ElGamal public-key encryption scheme makes use of a group $\mathbb{G}$ of prime order q with generator g. It also uses a hash function H, and a symmetric cipher (E, D). A public key for this scheme is a random group element $X \in \mathbb{G}$, with corresponding secret key $x \in \mathbb{Z}_q$, where $X = g^x$. To encrypt a message m, one chooses a random $y \in \mathbb{Z}_q$, computes

$$Y \leftarrow g^y, \quad Z \leftarrow X^y, \quad k \leftarrow \mathsf{H}(Y, Z), \quad c \overset{\$}{\leftarrow} \mathsf{E}_k(m),$$

and the ciphertext is (Y, c). Decryption works in the obvious way: given the ciphertext (Y, c), and secret key x, one computes

$$Z \leftarrow Y^x, \quad k \leftarrow \mathsf{H}(Y, Z), \quad m \leftarrow \mathsf{D}_k(c).$$

Define the function $\mathrm{dh} : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}$ by

$$\mathrm{dh}(X, Y) = Z, \quad \text{where } X = g^x,\ Y = g^y, \text{ and } Z = g^{xy}. \tag{3.2.1}$$

The *Diffie-Hellman (DH) problem* is to compute $\mathrm{dh}(X, Y)$ for random $X, Y \in \mathbb{G}$. The *DH assumption* asserts that this problem is intractable for polynomial-time algorithms. Clearly, the hashed ElGamal encryption scheme is secure *only if* it is hard to compute $Z = \mathrm{dh}(X, Y)$, given the values X and Y – that is, only if the DH assumption holds. However, this assumption is *not* sufficient to establish the security of hashed ElGamal against a *chosen ciphertext attack*, regardless of what security properties the hash function H may enjoy.

To illustrate the problem, suppose that an adversary selects group elements $\hat{Y}$ and $\hat{Z}$ in some arbitrary way, and computes $\hat{k} \leftarrow \mathsf{H}(\hat{Y}, \hat{Z})$ and $\hat{c} \overset{\$}{\leftarrow} \mathsf{E}_{\hat{k}}(\hat{m})$ for some arbitrary message $\hat{m}$. Further, suppose the adversary gives the ciphertext $(\hat{Y}, \hat{c})$ to a "decryption oracle," obtaining the decryption m. Now, it is very likely that $\hat{m} = m$ if and only if $\hat{Z} = \mathrm{dh}(X, \hat{Y})$. Thus, the decryption oracle can be used by the adversary as an oracle to answer questions of the form "is $\mathrm{dh}(X, \hat{Y}) = \hat{Z}$?" for group elements $\hat{Y}$ and $\hat{Z}$ of the adversary's choosing. In general, the adversary would not be able to efficiently answer such questions on his own, and so the decryption oracle is leaking some information about that secret key x which could conceivably be used to break the encryption scheme.

The Strong DH Assumption. Therefore, to establish the security of hashed ElGamal against chosen ciphertext attack, we need a stronger assumption. For $X, \hat{Y}, \hat{Z} \in \mathbb{G}$, define the predicate

$$\mathrm{dhp}(X, Y, Z) = \begin{cases} 1 & \text{if } \mathrm{dh}(X, Y) = Z \\ 0 & \text{otherwise} \end{cases}$$

At a bare minimum, we need to assume that it is hard to compute $\mathrm{dh}(X, Y)$, given random $X, Y \in \mathbb{G}$, along with access to a *decision oracle* for the predicate $\mathrm{dhp}(X, \cdot, \cdot)$, which on input $(\hat{Y}, \hat{Z})$, returns $\mathrm{dhp}(X, \hat{Y}, \hat{Z})$. This assumption is called the *strong DH assumption* [1].[1] Moreover, it is not hard to prove, if H is modeled as a random oracle, that hashed ElGamal is secure against chosen ciphertext attack under the strong DH assumption, and under the assumption that the underlying symmetric cipher is itself secure against chosen ciphertext attack. This was proved in [1, 55], for a variant scheme in which Y is not included in the hash; including Y in the hash gives a more efficient security reduction (see [32]). Note that the strong DH assumption is different (and weaker) than the so-called *gap DH assumption* [61] where an adversary gets access to a *full* decision oracle for the predicate $\mathrm{dhp}(\cdot, \cdot, \cdot)$, which on input $(\hat{X}, \hat{Y}, \hat{Z})$, returns $\mathrm{dhp}(\hat{X}, \hat{Y}, \hat{Z})$.

[1]We remark that in more recent papers the name strong DH assumption also sometimes refers to a different assumption defined over bilinear maps [20]. We follow the original terminology from [1].

3.3 The Twin Diffie-Hellman Problem

In this section we give our new problem, called the twin Diffie-Hellman problem, and prove that it is no harder than the DH problem, even with the help of a decision oracle.

Again, let $\mathbb{G}$ be a cyclic group with generator g, and of prime order q. Let dh be defined as in (3.2.1). Define the function

$$\begin{aligned} \mathrm{2dh}: \quad & \mathbb{G}^3 \to \mathbb{G}^2 \\ & (X_1, X_2, Y) \mapsto (\mathrm{dh}(X_1, Y), \mathrm{dh}(X_2, Y)). \end{aligned}$$

We call this the *twin DH function.* We also define a corresponding *twin DH predicate*:

$$\mathrm{2dhp}(X_1, X_2, Y, Z_1, Z_2) = \begin{cases} 1 & \text{if } \mathrm{2dh}(X_1, X_2, Y) = (Z_1, Z_2) \\ 0 & \text{otherwise} \end{cases}$$

The *twin DH assumption* states it is hard to compute $\mathrm{2dh}(X_1, X_2, Y)$, given random $X_1, X_2, Y \in \mathbb{G}$. It is clear that the DH assumption implies the twin DH assumption. The *strong twin DH assumption* states that it is hard to compute $\mathrm{2dh}(X_1, X_2, Y)$, given random $X_1, X_2, Y \in \mathbb{G}$, along with access to a *decision oracle* for the predicate $\mathrm{2dhp}(X_1, X_2, \cdot, \cdot, \cdot)$, which on input $(\hat{Y}, \hat{Z}_1, \hat{Z}_2)$, returns $\mathrm{2dhp}(X_1, X_2, \hat{Y}, \hat{Z}_1, \hat{Z}_2)$.

In this section we show that the DH assumption implies the strong twin DH assumption. While this result has direct applications, the basic tool that is used to prove the theorem, which is a kind of "trapdoor test," has found even wider application.

Roughly stated, the trapdoor test works as follows: given a random group element X_1, we can efficiently construct a random group element X_2, together with a secret "trapdoor" τ, such that

- X_1 and X_2 are independent (as random variables), and
- if we are given group elements $\hat{Y}, \hat{Z}_1, \hat{Z}_2$, computed as functions of X_1 and X_2

(but not τ), then using τ, we can efficiently evaluate the predicate

$$2\text{dhp}(X_1, X_2, \hat{Y}, \hat{Z}_1, \hat{Z}_2)$$

while making a mistake with only negligible probability.

The purpose of the trapdoor test will be intuitively clear in the proof of Theorem 2: in order to reduce the strong twin DH assumption to the DH assumption, the DH adversary will have to answer decision oracle queries without knowing the discrete logarithms of the elements of the strong twin DH problem instance. This tool gives us a method for doing so.

Theorem 1 (Trapdoor Test). *Let $\mathbb{G}$ be a cyclic group of prime order q, generated by $g \in \mathbb{G}$. Suppose X_1, r, s are mutually independent random variables, where X_1 takes values in $\mathbb{G}$, and each of r, s is uniformly distributed over $\mathbb{Z}_q$, and define the random variable $X_2 = g^s / X_1^r$. Further, suppose that $\hat{Y}, \hat{Z}_1, \hat{Z}_2$ are random variables taking values in $\mathbb{G}$, each of which is defined as some function of X_1 and X_2. Then we have:*

(i) X_2 is uniformly distributed over $\mathbb{G}$;

(ii) X_1 and X_2 are independent;

(iii) if $X_1 = g^{x_1}$ and $X_2 = g^{x_2}$, then the probability that the truth value of

$$\hat{Z}_1^r \hat{Z}_2 = \hat{Y}^s \tag{3.3.1}$$

does not agree with the truth value of

$$\hat{Z}_1 = \hat{Y}^{x_1} \wedge \ \hat{Z}_2 = \hat{Y}^{x_2} \tag{3.3.2}$$

is at most $1/q$; moreover, if (3.3.2) holds, then (3.3.1) certainly holds.

Proof. Observe that $s = rx_1 + x_2$. It is easy to verify that X_2 is uniformly distributed over $\mathbb{G}$, and that X_1, X_2, r are mutually independent, from which (i) and (ii) follow.

To prove (iii), condition on fixed values of X_1 and X_2. In the resulting conditional probability space, r is uniformly distributed over $\mathbb{Z}_q$, while x_1, x_2, $\hat{Y}$, $\hat{Z}_1$, and $\hat{Z}_2$ are fixed. If (3.3.2) holds, then by substituting the two equations in (3.3.2) into (3.3.1), we see that (3.3.1) certainly holds. Conversely, if (3.3.2) does not hold, we show that (3.3.1) holds with probability at most $1/q$. Observe that (3.3.1) is equivalent to

$$(\hat{Z}_1/\hat{Y}^{x_1})^r = \hat{Y}^{x_2}/\hat{Z}_2. \tag{3.3.3}$$

It is not hard to see that if $\hat{Z}_1 = \hat{Y}^{x_1}$ and $\hat{Z}_2 \neq \hat{Y}^{x_2}$, then (3.3.3) certainly does not hold. This leaves us with the case $\hat{Z}_1 \neq \hat{Y}^{x_1}$. But in this case, the left hand side of (3.3.3) is a random element of $\mathbb{G}$ (since r is uniformly distributed over $\mathbb{Z}_q$), but the right hand side is a fixed element of $\mathbb{G}$. Thus, (3.3.3) holds with probability $1/q$ in this case. $\square$

3.3.1 Main Result

We can easily prove Theorem 2, our main result.

So that we can give a concrete security result, let us define some terms. For a group $\mathbb{G}$ and an adversary $\mathcal{B}$, let us define its *DH advantage*, denoted $\mathsf{AdvDH}_{\mathcal{B},\mathbb{G}}$, to be the probability that $\mathcal{B}$ computes $\mathrm{dh}(X, Y)$, given random $X, Y \in \mathbb{G}$. For an adversary $\mathcal{A}$, let us define his *strong twin DH advantage*, denoted $\mathsf{Adv2DH}_{\mathcal{A},\mathbb{G}}$, to be the probability that $\mathcal{A}$ computes $2\mathrm{dh}(X_1, X_2, Y)$, given random $X_1, X_2, Y \in \mathbb{G}$, along with access to a *decision oracle* for the predicate $2\mathrm{dhp}(X_1, X_2, \cdot, \cdot, \cdot)$, which on input $\hat{Y}, \hat{Z}_1, \hat{Z}_2$, returns $2\mathrm{dhp}(X_1, X_2, \hat{Y}, \hat{Z}_1, \hat{Z}_2)$.

Theorem 2. *Suppose $\mathcal{A}$ is a strong twin DH adversary against the group $\mathbb{G}$ that makes at most Q_{d} queries to its decision oracle, and runs in time at most τ. Then there exists a DH adversary $\mathcal{B}$ with the following properties: $\mathcal{B}$ runs in time at most τ, plus the time to perform $O(Q_{\mathrm{d}} \log q)$ group operations and some minor bookkeeping;*

moreover,

$$\mathsf{Adv2DH}_{\mathcal{A},\mathbb{G}} \leq \mathsf{AdvDH}_{\mathcal{B},\mathbb{G}} + \frac{Q_{\mathrm{d}}}{q}.$$

In addition, if $\mathcal{B}$ does not output "failure," then its output is correct with probability at least $1 - 1/q$.

Proof. Our DH adversary $\mathcal{B}$ works as follows, given a challenge instance (X, Y) of the DH problem. First, $\mathcal{B}$ chooses $r, s \in \mathbb{Z}_q$ at random, sets $X_1 \leftarrow X$ and $X_2 \leftarrow g^s / X_1^r$, and gives $\mathcal{A}$ the challenge instance (X_1, X_2, Y). Second, $\mathcal{B}$ processes each decision query $(\hat{Y}, \hat{Z}_1, \hat{Z}_2)$ by testing if $\hat{Z}_1 \hat{Z}_2^r = \hat{Y}^s$ holds. Finally, if and when $\mathcal{A}$ outputs (Z_1, Z_2), $\mathcal{B}$ tests if this output is correct by testing if $Z_1 Z_2^r = Y^s$ holds; if this does not hold, then $\mathcal{B}$ outputs "failure," and otherwise, $\mathcal{B}$ outputs Z_1. The proof is easily completed using Theorem 1. $\square$

3.4 Twin ElGamal Encryption

We now describe our main application, which is a new public-key encryption scheme called *twin ElGamal*, which we denote by $\mathsf{PKE}_{2\mathrm{dh}}$. This scheme makes use of a hash function H and a symmetric cipher (E, D). It also uses a group $\mathbb{G}$ of prime order q with generator g. A public key for this scheme is a pair of random group elements (X_1, X_2), with corresponding secret key (x_1, x_2), where $X_i = g^{x_i}$ for $i = 1, 2$. To encrypt a message m, one chooses a random $y \in \mathbb{Z}_q$ and computes

$$Y \leftarrow g^y, \quad Z_1 \leftarrow X_1^y, \quad Z_2 \leftarrow X_2^y, \quad k \leftarrow \mathsf{H}(Y, Z_1, Z_2), \quad c \overset{\$}{\leftarrow} \mathsf{E}_k(m).$$

The ciphertext is (Y, c). Decryption works in the obvious way: given the ciphertext (Y, c), and secret key (x_1, x_2), one computes

$$Z_1 \leftarrow Y^{x_1}, \quad Z_2 \leftarrow Y^{x_2}, \quad k \leftarrow \mathsf{H}(Y, Z_1, Z_2), \quad m \leftarrow \mathsf{D}_k(c).$$

We note that the ciphertexts for this scheme are extremely compact — no redundancy is added, as in the Fujisaki-Okamoto transformation [41]. Moreover, the security reduction for our scheme is very tight. We remark that this seems to be the first DH-based encryption scheme with short ciphertexts. All other known constructions either add redundancy to the ciphertext [41, 61, 75, 31, 7] or resort to assumptions stronger than DH [1, 32, 55].

We now analyze the security of the twin ElGamal encryption scheme. The security will be based on the strong twin DH assumption, of course, and this allows us to borrow the "oracle patching" technique from previous analyzes of hashed ElGamal encryption based on the strong DH assumption [32]. We stress, however, that unlike previous applications of this technique, the end result is a scheme based on the original DH assumption.

Theorem 3. *Suppose* H *is modeled as a random oracle and that the DH assumption holds. Then* $\mathsf{PKE}_{\rm 2dh}$ *is secure against chosen ciphertext attack.*

In particular, suppose $\mathcal{A}$ *is an adversary that carries out a chosen ciphertext attack against* $\mathsf{PKE}_{\rm 2dh}$ *in the random oracle model, and that* $\mathcal{A}$ *runs in time* τ*, and makes at most* $Q_{\rm h}$ *hash queries and* $Q_{\rm d}$ *decryption queries. Then there exists a DH adversary* $\mathcal{B}_{\rm dh}$ *and an adversary* $\mathcal{B}_{\rm sym}$ *that carries out a chosen ciphertext attack against* SE*, such that both* $\mathcal{B}_{\rm dh}$ *and* $\mathcal{B}_{\rm sym}$ *run in time at most* τ*, plus the time to perform* $O((Q_{\rm h}+Q_{\rm d})\log q)$ *group operations; moreover,*

$$\mathsf{AdvCCA}^{\rm ro}_{\mathcal{A},\mathsf{PKE}_{\rm 2dh}} \le \mathsf{AdvDH}_{\mathcal{B}_{\rm dh},\mathbb{G}} + \mathsf{AdvCCA}_{\mathcal{B}_{\rm sym},\mathsf{SE}} + \frac{Q_{\rm h}}{q}.$$

Proof. In light of Theorem 2, the proof is fairly standard. We proceed with a sequence of games.

Game 0. Let Game 0 be the original chosen ciphertext attack game, and let S_0 be the event that $\hat{b} = b$ in this game.

In this game, the challenger generates the secret key (x_1, x_2) and computes the corresponding public key (X_1, X_2). We have to describe how the random oracle is implemented by the challenger. This is done in a special way to facilitate the proof.

The challenger implements the random oracle using an associative array L, indexed by elements of $\mathbb{G}^3$, where each element in the array takes an initial, default value of $\perp$, indicating that it is undefined. In addition, the challenger prepares some values in advance, to be used later as part of the ciphertext generated in response to the adversary's challenge query. Namely, the challenger chooses a random symmetric key k, and a random $y \in \mathbb{Z}_q$, sets $Y \leftarrow g^y$, $Z_1 \leftarrow X_1^y$, and $Z_2 \leftarrow X_2^y$. The challenger also sets $L[Y, Z_1, Z_2] \leftarrow k$, which intuitively represents the fact that $\mathsf{H}(Y, Z_1, Z_2) = k$.

Now, the challenger sends the public key to the adversary. Whenever the adversary makes a random oracle query, the challenger sends the corresponding entry in L to the adversary, initializing it, if necessary, to a random symmetric key if it is currently $\perp$.

To process decryption queries in step 2 of the chosen ciphertext attack game, suppose the ciphertext is $(\hat{Y}, \hat{c})$. If $\hat{Y} = Y$, then the challenger simply responds with $\mathsf{D}_k(\hat{c})$. Otherwise, the challenger decrypts as usual, using the secret key (x_1, x_2), and processing its own random oracle queries using L, just as above.

To process the challenge query in step 3, the challenger uses the values Y, Z_1, Z_2, k generated in the initialization step, and computes $c \xleftarrow{\$} \mathsf{E}_k(m_b)$. The ciphertext (Y, c) is given to the adversary. Decryption queries in step 4 are processed just as in step 2.

That finishes the description of Game 0. Despite the syntactic differences, it is clear that

$$\mathsf{AdvCCA}^{\mathrm{ro}}_{\mathcal{A},\mathsf{PKE}_{\mathsf{2dh}}} = |\Pr[S_0] - 1/2|. \tag{3.4.1}$$

Game 1. We now describe Game 1, which is the same as Game 0, but with the following difference: in the initialization step, the challenger does not initialize $L[Y, Z_1, Z_2]$. Everything else remains *exactly* the same.

Let S_1 be the event that $\hat{b} = b$ in Game 1. Let F be the event that the adversary queries the random oracle at (Y, Z_1, Z_2) in Game 1. Note that the challenger never queries the random oracle at this point, due to the special way that decryption and challenge queries are processed. Since both Games 0 and 1 proceed identically unless F occurs, we have

$$|\Pr[S_1] - \Pr[S_0]| \leq \Pr[F]. \tag{3.4.2}$$

We claim that

$$\Pr[F] \leq \mathsf{Adv2DH}_{\mathcal{B}_{\mathrm{2dh}},\mathbb{G}}, \tag{3.4.3}$$

where $\mathcal{B}_{\mathrm{2dh}}$ is an efficient strong twin DH adversary that makes at most Q_{h} decision oracle queries. We sketch at a very high level how $\mathcal{B}_{\mathrm{2dh}}$ works. Basically, $\mathcal{B}_{\mathrm{2dh}}$ runs just like the challenger in Game 1, but for every random oracle query $(\hat{Y}, \hat{Z}_1, \hat{Z}_2)$, $\mathcal{B}_{\mathrm{2dh}}$ sends this triple to its own decision oracle, and marks it "good" or "bad" accordingly (this is the only time $\mathcal{B}_{\mathrm{2dh}}$ uses its decision oracle). Using this information, $\mathcal{B}_{\mathrm{2dh}}$ can easily process decryption requests without using the secret key: given a ciphertext $(\hat{Y}, \hat{c})$ with $\hat{Y} \neq Y$, it checks if it has already seen a "good" triple of the form $(\hat{Y}, \cdot, \cdot)$ among the random oracle queries; if so, it uses the key associated with that triple; if not, it generates a random key, and it will stay on the lookout for a "good" triple of the form $(\hat{Y}, \cdot, \cdot)$ in future random oracle queries, associating this key with that triple to keep things consistent. At the end of the game, $\mathcal{B}_{\mathrm{2dh}}$ checks if it has seen a "good" triple of the form $(Y, \cdot, \cdot)$; if so, it outputs the last two components.

Of course, Theorem 1 gives us an efficient DH adversary $\mathcal{B}_{\mathrm{dh}}$ with

$$\mathsf{Adv2DH}_{\mathcal{B}_{\mathrm{2dh}},\mathbb{G}} \leq \mathsf{AdvDH}_{\mathcal{B}_{\mathrm{dh}},\mathbb{G}} + \frac{Q_{\mathrm{h}}}{q}. \tag{3.4.4}$$

Finally, it is easy to see that in Game 1, the adversary is essentially playing the chosen ciphertext attack game against SE. Thus, there is an efficient adversary $\mathcal{B}_{\mathrm{sym}}$ such that

$$|\Pr[S_1] - 1/2| = \mathsf{AdvOTCCA}_{\mathcal{B}_{\mathrm{sym}},\mathsf{SE}}. \tag{3.4.5}$$

The theorem now follows by combining (3.4.1)–(3.4.5). $\square$

Instantiating $\mathsf{PKE}_{\text{2dh}}$ with a length-preserving one-time CCA secure symmetric encryption scheme (see Section 2.4.2 and [67, 50, 49, 51] for constructions), we obtain a DH-based chosen-ciphertext secure encryption scheme with the following properties.

Optimal ciphertext overhead. The ciphertext overhead, i.e. ciphertext size minus plaintext size, is exactly one group element, which is optimal for Diffie-Hellman based schemes.

Encryption/decryption efficiency. Encryption needs three exponentiations in $\mathbb{G}$, one of which is to the fixed-base g (that can be shared among many public-keys). Decryption only needs one sequential exponentiation in $\mathbb{G}$ to compute Y^{x_1} and Y^{x_2} simultaneously, which is nearly as efficient as one single exponentiation (see, e.g., [57]).

3.5 A Variant of the Cramer-Shoup Encryption Scheme

In this section we show how to apply our trapdoor test to construct public-key encryption schemes with security proofs without random oracles. We give a new assumption based on the *decisional* Diffie-Hellman problem and describe several schemes with varying efficiency and security properties.

3.5.1 The DDH and Twin DDH Assumptions

Let $\mathbb{G}$ be a group of order q and let g be a generator of $\mathbb{G}$. Distinguishing the two distributions $(X, Y, \text{dh}(X, Y))$ and (X, Y, Z) for random $X, Y, Z \in \mathbb{G}$ is the *decision Diffie-Hellman* (DDH) problem. For an adversary $\mathcal{B}$, let us define his *DDH advantage*, denoted $\mathsf{AdvDDH}_{\mathcal{B},\mathbb{G}}$, by

$$\mathsf{AdvDDH}_{\mathcal{B},\mathbb{G}} = \Pr[\mathcal{B}(X, Y, \text{dh}(X, Y)) = 1] - \Pr[\mathcal{B}(X, Y, Z) = 1], \tag{3.5.1}$$

where X, Y, Z are uniform random variables on $\mathbb{G}$. The *DDH assumption* states that the DDH problem is hard.

We consider a natural decision variant of the twin DH problem, the *twin DDH problem* is distinguishing the two distributions $(X_1, X_2, Y, \mathrm{dh}(X_1, Y))$ and (X_1, X_2, Y, Z) for random $X_1, X_2, Y, Z \in \mathbb{G}$. The *strong twin DDH assumption* states that the twin DDH problem is hard, even given access to a decision oracle for the predicate for $2\mathrm{dhp}(X_1, X_2, \cdot, \cdot, \cdot)$, which on input $(\hat{Y}, \hat{Z}_1, \hat{Z}_2)$ returns $2\mathrm{dhp}(X_1, X_2, \hat{Y}, \hat{Z}_1, \hat{Z}_2)$. (Note the value $\mathrm{dh}(X_2, Y)$ is never provided as input to the distinguisher since otherwise the strong twin DDH assumption could be trivially broken using the 2dhp oracle.) For an adversary $\mathcal{B}$, we define its *strong twin DDH advantage*, denoted $\mathsf{Adv2DDH}_{\mathcal{B},\mathbb{G}}$, by

$$\mathsf{Adv2DDH}_{\mathcal{B},\mathbb{G}} = \Pr[\mathcal{B}(X_1, X_2, Y, \mathrm{dh}(X_1, Y)) = 1] - \Pr[\mathcal{B}(X_1, X_2, Y, Z_1) = 1], \quad (3.5.2)$$

where X_1, X_2, Y, Z_1 are uniform random variables on $\mathbb{G}$, and $\mathcal{B}$ has access to an oracle for $2\mathrm{dhp}(X_1, X_2, \cdot, \cdot, \cdot)$.

We also consider potentially weaker "hashed" variants of the above two assumptions. For a hash function $\mathsf{H} : \mathbb{G} \to \{0,1\}^\kappa$, the *hashed DDH* problem is to distinguish the two distributions $(X, Y, \mathsf{H}(\mathrm{dh}(X, Y))$ and (X, Y, k), for random $X, Y \in \mathbb{G}$ and $k \in \{0,1\}^\kappa$. The *hashed DDH assumption* states that the hashed DDH problem is hard. Finally, the *strong twin hashed DDH assumption* states that it is hard to distinguish the distributions $(X_1, X_2, Y, \mathsf{H}(\mathrm{dh}(X, Y))$ and (X_1, X_2, Y, k), even with access to an oracle computing $2\mathrm{dhp}(X_1, X_2, \cdot, \cdot, \cdot)$, where $X_1, X_2, Y \in \mathbb{G}$ and $k \in \{0,1\}^\kappa$ are random.

We note that the (strong twin) hashed DDH assumption simplifies to the (strong twin) DDH assumption if the range of the hash function is $\mathbb{G}$ instead of $\{0,1\}^\kappa$ and H is the identity (i.e., it maps $Z \in \mathbb{G}$ to $Z \in \mathbb{G}$). Furthermore, there are natural groups (such as non-prime-order groups like $\mathbb{Z}_p$) where the DDH problem is known to be easy yet the hashed DDH problem can still plausibly assumed to be hard for a reasonable choice of the hash function [43]. If H is modeled as random oracle then the hashed DDH and the DH assumptions become equivalent.

Using the trapdoor test in Theorem 1, we can prove an analogue of Theorem 2.

Theorem 4. *The (hashed) DDH assumption holds if and only if the strong twin (hashed) DDH assumption holds.*

In particular, suppose $\mathcal{A}$ is a strong twin (hashed) DDH adversary that makes at most $Q_{\rm d}$ queries to its decision oracle, and runs in time at most τ. Then there exists a (hashed) DDH adversary $\mathcal{B}$ with the following properties: $\mathcal{B}$ runs in time at most τ, plus the time to perform $O(Q_{\rm d} \log q)$ group operations and some minor bookkeeping; moreover,

$$\mathsf{Adv2DDH}_{\mathcal{A},\mathbb{G}} \leq \mathsf{AdvDDH}_{\mathcal{B},\mathbb{G}} + \frac{Q_{\rm d}}{q}.$$

3.5.2 A Variant of the Cramer-Shoup Scheme

We now can consider the following encryption scheme which we call $\mathsf{PKE}_{\widetilde{\mathrm{cs}}}$. This scheme makes use of a symmetric cipher (E, D) and a hash function $\mathsf{T} : \mathbb{G} \rightarrow \mathbb{Z}_q$ which we assume to be target collision-resistant (see §3.1.2).

This scheme uses a group $\mathbb{G}$ of prime order q with generator g. A public key for this scheme is a tuple of random group elements $(X_1, \tilde{X}_1, X_2, \tilde{X}_2) \in \mathbb{G}^4$, with corresponding secret key $(x_1, \tilde{x}_1, x_2, \tilde{x}_2)$, where $X_i = g^{x_i}$ and $\tilde{X}_i = g^{\tilde{x}_i}$ for $i = 1, 2$. To encrypt a message m, one chooses a random $y \in \mathbb{Z}_q$, computes

$$Y \leftarrow g^y, \quad t \leftarrow \mathsf{T}(Y), \quad Z_1 \leftarrow (X_1^t \tilde{X}_1)^y, \quad Z_2 \leftarrow (X_2^t \tilde{X}_2)^y, \quad k \leftarrow \mathsf{H}(X_1^y), \quad c \overset{\$}{\leftarrow} \mathsf{E}_k(m),$$

and the ciphertext is (Y, Z_1, Z_2, c). Decryption works as follows: given the ciphertext (Y, Z_1, Z_2, c) and secret key $(x_1, \tilde{x}_1, x_2, \tilde{x}_2)$, one computes $t \leftarrow \mathsf{T}(Y)$ and checks if

$$Y^{x_1 t + \tilde{x}_1} = Z_1 \text{ and } Y^{x_2 t + \tilde{x}_2} = Z_2. \tag{3.5.3}$$

If not, then one rejects the ciphertext. (In this case, we say the ciphertext is *not consistent*). Otherwise, compute

$$k \leftarrow \mathsf{H}(Y^{x_1}), \quad m \leftarrow \mathsf{D}_k(c).$$

We remark that since $|\mathbb{G}| = |\mathbb{Z}_q| = q$, hash function T could be a bijection, and hence target collision-resistant unconditionally. See [23] for efficient constructions for certain groups $\mathbb{G}$.

Relation to Cramer-Shoup. Our scheme is very similar to the one by Cramer and Shoup [32]. Syntactically, the difference is that in Cramer-Shoup the value Z_1 is computed as $Z_1 = X_3^y$ (where X_3 is another random group element in the public key) and t is computed as $t = \mathsf{T}(Y, Z_1)$. However, our variant allows for a *simple security proof* based on the *hashed* DDH assumption whereas for the Cramer-Shoup scheme only proofs based on the DDH assumption are known (and the known proofs do not seem to extend to the hashed case because the reductions all apply algebraic operations to the challenge input).

We now show that, using the trapdoor test, $\mathsf{PKE}_{\widetilde{\mathrm{cs}}}$ allows for a very elementary proof under the hashed DDH assumption. We stress that are security proof is not in the random oracle model.

Theorem 5. *Suppose T is a target collision resistant hash function. Further, suppose the hashed DDH assumption holds, and that the symmetric cipher $\mathsf{SE} = (\mathsf{E}, \mathsf{D})$ is secure against chosen ciphertext attack. Then $\mathsf{PKE}_{\widetilde{\mathrm{cs}}}$ is secure against chosen ciphertext attack.*

In particular, suppose $\mathcal{A}$ is an adversary that carries out a chosen ciphertext attack against $\mathsf{PKE}_{\widetilde{\mathrm{cs}}}$ and that $\mathcal{A}$ runs in time τ, and makes at most Q_{d} decryption queries. Then there exists a hashed DDH adversary $\mathcal{B}_{\mathrm{ddh}}$, an adversary $\mathcal{B}_{\mathrm{sym}}$ that carries out a chosen ciphertext attack against SE, and a TCR adversary $\mathcal{B}_{\mathrm{tcr}}$ such that both $\mathcal{B}_{\mathrm{ddh}}$, $\mathcal{B}_{\mathrm{sym}}$ and $\mathcal{B}_{\mathrm{tcr}}$ run in time at most τ, plus the time to perform $O(Q_{\mathrm{d}} \log q)$ group operations; moreover,

$$\mathsf{AdvCCA}_{\mathcal{A},\mathsf{PKE}_{\widetilde{\mathrm{cs}}}} \leq \mathsf{AdvDDH}_{\mathcal{B}_{\mathrm{ddh}},\mathbb{G},\mathsf{H}} + \mathsf{AdvCCA}_{\mathcal{B}_{\mathrm{sym}},\mathsf{SE}} + \mathsf{AdvTCR}_{\mathcal{B}_{\mathrm{tcr}},\mathsf{T}} + \frac{Q_{\mathrm{d}}}{q}.$$

Proof. We proceed with a sequence of games.

Game 0. Let Game 0 be the original chosen ciphertext attack game, and let S_0 be the event that $\hat{b} = b$ in this game. It is apparent that

$$\mathsf{AdvCCA}_{\mathcal{A},\mathsf{PKE}_{\widetilde{\mathrm{cs}}}} = |\Pr[S_0] - 1/2|. \tag{3.5.4}$$

Game 1. Let Game 1 be like Game 0, but with the following difference. Game 1 aborts if the adversary, at any time, makes a decryption query containing a $\hat{Y}$ such that $\hat{Y} \neq Y$ and $\mathsf{T}(\hat{Y}) = \mathsf{T}(Y)$ where Y comes from the challenge ciphertext. Call this event F. It is clear that

$$|\Pr[S_1] - \Pr[S_0]| = \Pr[F], \tag{3.5.5}$$

Using a standard argument from [32] it is easy to show that

$$\Pr[F] \leq \mathsf{AdvTCR}_{\mathcal{B}_{\mathrm{tcr}},\mathsf{T}}, \tag{3.5.6}$$

where $\mathcal{B}_{\mathrm{tcr}}$ is an adversary as specified in the theorem. We outline how to prove this inequality. In the TCR game, the adversary $\mathcal{B}_{\mathrm{tcr}}$ first selects a random Y from $\mathbb{G}$ and gives it to the TCR challenger, which returns a sampled hash function T. $\mathcal{B}_{\mathrm{tcr}}$ proceeds to simulate the entire chosen ciphertext game for $\mathcal{A}$ using T. When generating the challenge ciphertext, $\mathcal{B}_{\mathrm{tcr}}$ uses its sample Y. If $\mathcal{A}$ ever submits a decryption query that includes $\hat{Y}$ such that $T(\hat{Y}) = T(Y)$, then $\mathcal{B}_{\mathrm{tcr}}$ returns $\hat{Y}$ to the TCR challenger. If $\mathcal{A}$ never submits such a query, then $\mathcal{B}_{\mathrm{tcr}}$ chooses an arbitrary second output. It is apparent that the game simulated by $\mathcal{B}_{\mathrm{tcr}}$ is exactly like Game 1, and that whenever the event F occurs, $\mathcal{B}_{\mathrm{tcr}}$ wins the TCR game. The inequality follows.

Game 2. Let Game 2 be as Game 1 with the following differences. For computing the public-key the experiment picks $x_1, x_2, y, a_1, a_2 \in \mathbb{Z}_q$ at random and computes $X_1 = g^{x_1}$, $X_2 = g^{x_2}$, and $Y = g^y$. Next, it computes $t \leftarrow \mathsf{T}(Y)$ and

$$\tilde{X}_1 \leftarrow X_1^{-t} g^{a_1}, \quad \tilde{X}_2 \leftarrow X_2^{-t} g^{a_2}.$$

Note that the way the public-key is setup uses a technique to prove selective-ID security for IBE schemes [19].

The challenge ciphertext (Y, Z_1, Z_2, c) for message m_b is computed as

$$t \leftarrow \mathsf{T}(Y), \quad Z_1 \leftarrow Y^{a_1}, \quad Z_2 \leftarrow Y^{a_2}, \quad k \leftarrow \mathsf{H}(X_1^y), \quad c \xleftarrow{\$} \mathsf{E}_k(m_b). \tag{3.5.7}$$

This is a correctly distributed ciphertext for m_b and randomness $y = \log_g(Y)$ since, for $i = 1, 2$, $(X_i^t \tilde{X}_i)^y = (X_i^{t-t} g^{a_i})^y = (g^{a_i})^y = Y^{a_i} = Z_i$. We can assume (Y, Z_1, Z_2, k) to be computed in the beginning of the experiment since they are independent of m_0, m_1.

A decryption query for ciphertext $(\hat{Y}, \hat{Z}_1, \hat{Z}_2, \hat{c})$ is answered as follows. Compute $\hat{t} = \mathsf{T}(\hat{Y})$. If $t = \hat{t}$ then verify consistency by checking if $Z_1 = \hat{Z}_1$ and $Z_2 = \hat{Z}_2$. If the ciphertext is consistent then use the challenge key k defined in (3.5.7) to decrypt $\hat{c}$. If $t \neq \hat{t}$ then proceed as follows. For $i = 1, 2$, compute $\bar{Z}_i = (\hat{Z}_i / \hat{Y}^{a_i})^{1/(\hat{t}-t)}$. Consistency of the ciphertext is verified by checking if

$$\hat{Y}^{x_1} = \bar{Z}_1 \text{ and } \hat{Y}^{x_2} = \bar{Z}_2. \tag{3.5.8}$$

Let $\hat{y} = \log_g \hat{Y}$. The value $\hat{Z}_i$ was correctly generated iff $\hat{Z}_i = (X_i^{\hat{t}} \tilde{X}_i)^{\hat{y}} = (X_i^{\hat{t}-t} g^{a_i})^{\hat{y}} = (\hat{Y}^{x_i})^{\hat{t}-t} \cdot \hat{Y}^{a_i}$ which is equivalent to $\bar{Z}_i = \hat{Y}^{x_i}$. Hence, (3.5.8) is equivalent to the test from the original scheme (3.5.3). If the ciphertext is consistent then one can use the symmetric key $\hat{k} = \mathsf{H}(\bar{Z}_1) = \mathsf{H}(\hat{Y}^{x_1})$ to decrypt $\hat{c}$ and return $\hat{m} = \mathsf{D}_{\hat{k}}(\hat{c})$.

Let S_2 be the event that $\hat{b} = b$ in this game. As we have seen,

$$\Pr[S_2] = \Pr[S_1]. \tag{3.5.9}$$

Game 3. Let Game 3 be as Game 2 except that the value k used in the challenge ciphertext is now chosen at random. We claim that

$$|\Pr[S_3] - \Pr[S_2]| \leq \mathsf{Adv2DDH}_{\mathcal{B}_{2\text{ddh}}, \mathbb{G}, \mathsf{H}}, \tag{3.5.10}$$

where $\mathcal{B}_{2\text{ddh}}$ is an efficient strong twin hashed DDH adversary that makes at most Q_{d} queries to the decision oracle. $\mathcal{B}_{2\text{ddh}}$ is defined as follows. Using the values

(X_1, X_2, Y, k) from its challenge (where either $k = \mathsf{H}(\mathrm{dh}(X_1, Y))$ or k is random), adversary $\mathcal{B}_{\mathrm{2ddh}}$ runs (without knowing x_1, x_2, y) the experiment as described in Game 2 using k as the challenge key in (3.5.7) to encrypt m_b. Note that the only point where Games 2 and 3 make use of x_1 and x_2 is the consistency check (3.5.8) which $\mathcal{B}_{\mathrm{2ddh}}$ equivalently implements using the 2dhp oracle, i.e. by checking if

$$2\mathrm{dhp}(X_1, X_2, \hat{Y}, \bar{Z}_1, \bar{Z}_2)$$

holds. We have that if $k = \mathsf{H}(\mathrm{dh}(X_1, Y)) \in \{0,1\}^\kappa$, this perfectly simulates Game 2, whereas if $k \in \{0,1\}^\kappa$ is random this perfectly simulates Game 3. This proves (3.5.10).

Finally, it is easy to see that in Game 3, the adversary is essentially playing the chosen ciphertext attack game against SE. Thus, there is an efficient adversary $\mathcal{B}_{\mathrm{sym}}$ such that

$$|\Pr[S_3] - 1/2| = \mathsf{AdvCCA}_{\mathcal{B}_{\mathrm{sym}},\mathsf{SE}}. \tag{3.5.11}$$

The theorem now follows by combining (3.5.4)–(3.5.11) with Theorem 4. □

3.5.3 A Variant with Shorter Ciphertexts

Consider a ciphertext in the above scheme that is of the form $(Y = g^r, Z_1 = (X_1^t \tilde{X}_1)^{r'}, Z_2, c)$ with $r \neq r'$. Such a ciphertext is inconsistent and should therefore be rejected by Equation (3.5.3) in the decryption algorithm. Essentially, the trapdoor test says that in the view of the adversary, the unique value Z_2 that leads the simulation (as described in the proof of Theorem 5) to *falsely accept* such ciphertexts is a uniformly distributed group element. Therefore, the adversary can never guess this "bad Z_2" and, with high probability, the simulation of the CCA experiment is correct.

With this intuition it is easy to see that one can as well replace $Z_2 \in \mathbb{G}$ in the ciphertext by $Z_2' = \mathsf{KDF}(Z_2) \in \{0,1\}^k$, where $\mathsf{KDF} : \mathbb{G} \to \{0,1\}^k$ is a secure key-derivation function. (For uniform $X \in \mathbb{G}$, $\mathsf{KDF}(X)$ is computationally indistinguishable from an uniform bitstring in $\{0,1\}^k$.) Accordingly, decryption is modified to check $Z_2' = \mathsf{KDF}(Y^{x_2 t + \tilde{x}_2})$. This variant shortens the ciphertexts by replacing a group element by a bitstring in $\{0,1\}^k$.

Yet another variant uses the value Z_2 directly as a source for an integrity check of the symmetric cipher. Here we assume that symmetric encryption satisfies the stronger notion of (one-time) authenticated encryption [52]. Such a ciphertext can, for example, be obtained by combining a one-time pad with a message authenticated code (MAC). The idea is to move the value Z_2 from the ciphertext into the symmetric key which we re-define as $k = \mathsf{H}(X_1^y \cdot Z_2) = \mathsf{H}(X_1^y \cdot (X_2^t \tilde{X}_2)^y)$. Now, if (Y, Z_1) is inconsistent (in the above sense that $r \neq r'$) then the value for Z_2 used in the simulation is random and will make the symmetric key k essentially look random (from the adversary's view). Consequently, the authenticity property of the symmetric cipher makes the simulated decryption algorithm reject this ciphertext. After applying one more simplification (defining $Z_2 = X_2^y$) we get the following scheme which we call $\mathsf{PKE}_{\widetilde{\mathrm{kd}}}$.

Public and secret keys as the same as in $\mathsf{PKE}_{\widetilde{\mathrm{cs}}}$ with the difference that the element $\tilde{X}_2$ is no longer needed in the public-key. To encrypt a message m, one chooses a random $y \in \mathbb{Z}_q$, computes

$$Y \leftarrow g^y, \quad t \leftarrow \mathsf{T}(Y), \quad Z_1 \leftarrow (X_1^t \tilde{X}_1)^y, \quad k \leftarrow \mathsf{H}(X_2^y), \quad c \overset{\$}{\leftarrow} \mathsf{E}_k(m),$$

and the ciphertext is (Y, Z_1, c). Decryption works as follows: given the ciphertext (Y, Z_1, c), and secret key $(x_1, \tilde{x}_1, x_2)$, one computes $t \leftarrow \mathsf{T}(Y)$ and checks if

$$Y^{x_1 t + \tilde{x}_1} = Z_1.$$

If not, reject; otherwise, compute

$$k \leftarrow \mathsf{H}(Y^{x_2}), \quad m \leftarrow \mathsf{D}_k(c).$$

This scheme is essentially the public-key encryption scheme presented in [52]. Here, using the trapdoor test we offer a different and maybe simpler interpretation of its security.

Theorem 6. *Suppose* T *is a target collision resistant hash function. Further, suppose the hashed DDH assumption holds, and that the symmetric cipher* $\mathsf{SE} = (\mathsf{E}, \mathsf{D})$ *is secure in the sense of authenticated encryption. Then* $\mathsf{PKE}_{\widetilde{\mathrm{kd}}}$ *is secure against chosen ciphertext attack.*

A proof and a more precise security statement can be looked up in [52] or can alternatively be obtained by modifying the proof of Theorem 5 as described above. We remark that even though it is not explicitly mentioned in [52] their original proof already implies security of the $\mathsf{PKE}_{\widetilde{\mathrm{kd}}}$ scheme based on the *hashed* DDH assumption.

3.5.4 A Variant with Security From the DH Assumption

We now consider an extension of $\mathsf{PKE}_{\widetilde{\mathrm{cs}}}$ that achieves security based on the (computational) DH assumption. The idea is to first extend the public keys and ciphertexts to have several X_i and Z_i terms, respectively, and then use the Goldreich-Levin hard-core function [47, 46] as the hash function to extract symmetric key bits. Because security depends on the reduction to the hard-coreness of the function, the reduction is not very tight, and so we carry out the analysis in asymptotic terms. Below, we denote by f_{gl} the Goldreich-Levin hard-core function for $\mathrm{dh'}(X, Y, R) = (\mathrm{dh}(X, Y), R)$.

Let κ be the security parameter, and for simplicity we assume that it is also the length (in bits) of the symmetric keys for (E, D). Let $\nu = O(\log \kappa)$ be some integer that divides κ, and let $\ell = \kappa/\nu$. In this scheme, the secret key now consists of $2(\ell+1)$ random elements of $\mathbb{Z}_p$, denoted $x_i, \tilde{x}_i$ for $i = 1, \ldots, \ell+1$. The public key contains the $2(\ell+1)$ corresponding group elements $X_i = g^{x_i}, \tilde{X}_i = g^{\tilde{x}_i}$, for $i = 1, \ldots, \ell+1$, along with a random bit string R of length long enough to evaluate a hard-core function with ν output bits ($u = 2 \log |\mathbb{G}|$ bits are sufficient). To encrypt a message m, one chooses a random $y \in \mathbb{Z}_q$ and computes

$$Y \leftarrow g^y, \quad t \leftarrow \mathsf{T}(Y), \quad Z_i \leftarrow (X_i^t \tilde{X}_i)^y \text{ for } i = 1, \ldots, \ell + 1.$$

Then one sets $k_i \leftarrow f_{\mathrm{gl}}(X_i^y, R) \in \{0,1\}^\nu$ $(i = 1, \ldots, \ell)$. Note that $X_{\ell+1}, \tilde{X}_{\ell+1}$, and $Z_{\ell+1}$ are not used for key derivation. Finally, a concatenation of all k_i yields a symmetric key $k \in \{0,1\}^\kappa$ that is used to encrypt m as $c \xleftarrow{\$} \mathsf{E}_k(m)$. The ciphertext is $(Y, Z_1, \ldots, Z_{\ell+1}, c)$. Decryption first verifies the consistency of $(Y, Z_1, \ldots, Z_{\ell+1}, c)$ by checking if $Y^{x_i t + \tilde{x}_i} = Z_i$ for all $i = 1, \ldots, \ell + 1$. Then the key k is reconstructed as the concatenation of $k_i = f_{\mathrm{gl}}(Y^{x_i}, R)$ for $i = 1, \ldots, \ell$, and finally m is recovered by computing $m \leftarrow \mathsf{D}_k(c)$.

In order to analyze this scheme we will need the following version of the Goldreich-Levin theorem.

Theorem 7. *Suppose that $\mathcal{A}_{\mathrm{gl}}$ is a probabilistic poly-time algorithm such that, given (X, Y, R, k) as input, $\mathcal{A}_{\mathrm{gl}}$ distinguishes $k = f_{\mathrm{gl}}(\mathrm{dh}(X, Y), R)$ from a uniform string with non-negligible advantage, for random $X, Y \in \mathbb{G}$ and random $R \in \{0,1\}^u$. Then there exists a probabilistic poly-time algorithm $\mathcal{A}_{\mathrm{dh}}$ that computes $\mathrm{dh}(X, Y)$ with non-negligible probability for random X, Y.*

Using our techniques, it also not hard to show that this theorem still holds if we assume that $\mathcal{A}_{\mathrm{gl}}$ additionally gets as input a random $X' \in \mathbb{G}$ and has access to an oracle computing $\mathrm{2dhp}(X, X', \cdot, \cdot, \cdot)$. We will use this augmented version in our analysis below.

Theorem 8. *Suppose T is a target collision resistant hash function. Further, suppose the DH assumption holds, and that the symmetric cipher $\mathsf{SE} = (\mathsf{E}, \mathsf{D})$ is secure against chosen ciphertext attack. Then $\mathsf{PKE}_{\mathrm{dh}}$ is secure against chosen ciphertext attack.*

Proof. We proceed with a sequence of games. For each i, let S_i be the event that $\hat{b} = b$ in Game i.

Game 0. We define Game 0 to be the original CCA game that $\mathcal{A}$ plays against $\mathsf{PKE}_{\mathrm{dh}}$. By definition,

$$|\Pr[S_0] - 1/2| = \mathsf{AdvCCA}_{\mathcal{A},\mathsf{PKE}_{\mathrm{dh}}} \tag{3.5.12}$$

Game 1. Game 1 is the same as Game 0, except now if the adversary asks for a decryption of a ciphertext containing $\hat{Y} \neq Y$ but $\mathsf{T}(\hat{Y}) = \mathsf{T}(Y)$, where Y is from the challenge ciphertext, then the game aborts. By the target-collision resistance of T, we have that

$$|\Pr[S_1] - \Pr[S_0]| \leq \mathrm{negl}(\kappa). \tag{3.5.13}$$

The proof of this inequality is almost exactly like transition between Games 0 and 1 in the proof of Theorem 5, and is omitted.

Game 2. Game 2 is the same as Game 1, except that k is set to a uniform and independent bit string. We claim that

$$|\Pr[S_2] - \Pr[S_1]| \leq \mathrm{negl}(\kappa). \tag{3.5.14}$$

We will prove this by a hybrid argument. For $j = 0, \ldots, \ell$ we define the hybrid games H_j and H'_j. Intuitively, in H_j and H'_j, $k_1, \ldots, k_j$ will be uniformly random strings, while $k_{j+1}, \ldots, k_\ell$ will be computed normally. The hybrids will be defined so that H_j and H'_j have exactly the same output distribution, but are run in a slightly different way to facilitate the proof. In addition, H'_0 will have the same output distribution as Game 1, and H_ℓ will have that of Game 2. (The hybrid games H_0 and H'_ℓ will not be defined.) In the analysis, we will show that the games H_j and H'_j induce the same output distribution and that the games H'_{j-1} and H_j are computationally indistinguishable.

We now describe the hybrid games. Fix some $j \in \{0, \ldots, \ell\}$. We start with H_j and show how to define H'_j afterwards. In H_j, the public key is generated as follows.

It samples $y \xleftarrow{\$} \mathbb{Z}_q$ and sets $Y \leftarrow g^y$, $t \leftarrow \mathsf{T}(Y)$. For $i = 1, \dots \ell,\ i \neq j,\ X_i, \widetilde{X}_i$ are generated normally. The game then samples $x_j, x_{\ell+1}, a_j, a_{\ell+1} \xleftarrow{\$} \mathbb{Z}_q$ and computes

$$X_j \leftarrow g^{x_j}, \quad X_{\ell+1} \leftarrow g^{x_{\ell+1}}, \quad \widetilde{X}_j \leftarrow X_j^{-t} g^{a_j}, \quad \widetilde{X}_{\ell+1} \leftarrow X_{\ell+1}^{-t} g^{a_{\ell+1}}. \tag{3.5.15}$$

Finally, it samples $R \xleftarrow{\$} \{0,1\}^u$ and sets the public key to $(X_1, \widetilde{X}_1, \dots, X_{\ell+1}, \widetilde{X}_{\ell+1}, R)$.

To compute the challenge ciphertext, H_j does the following. It uses the Y that it computed at the start of the game, and sets $Z_i \leftarrow Y^{x_i}$ for $i \neq j, \ell+1$. It sets $Z_j \leftarrow Y^{a_j}$ and $Z_{\ell+1} \leftarrow Y^{a_{\ell+1}}$. For $i = 1, \dots, j$, it sets $k_i \xleftarrow{\$} \{0,1\}^\nu$. For $i = j+1, \dots, \ell$ it computes $k_i \leftarrow f_{\mathrm{gl}}(Y^{x_i}, R)$. It uses $k \leftarrow k_1 \dots k_\ell$ and computes $c \leftarrow \mathsf{E}_k(m_b)$, and the challenge ciphertext is $(Y, Z_1, \dots, Z_{\ell+1}, c)$.

To respond to a decryption query for the ciphertext $(\hat{Y}, \hat{Z}_1, \dots, \hat{Z}_{\ell+1}, \hat{c})$, H_j first computes $\hat{t} \leftarrow \mathsf{T}(\hat{Y})$. If $\hat{t} = t$, it checks if $\hat{Z}_i = Z_i$ for all i. If this holds it decrypts $\hat{c}$ using k. If $\hat{t} \neq t$, it verifies the consistency of Z_i for $i \neq j, \ell+1$, normally. It then computes

$$\bar{Z}_j \leftarrow (\hat{Z}_j / \hat{Y}_j^{a_j})^{1/(t-\hat{t})}, \qquad \bar{Z}_{\ell+1} \leftarrow (\hat{Z}_{\ell+1} / \hat{Y}_{\ell+1}^{a_{\ell+1}})^{1/(t-\hat{t})}$$

and tests if

$$\hat{Y}^{x_j} = \bar{Z}_j \ \text{ and } \ \hat{Y}^{x_{\ell+1}} = \bar{Z}_{\ell+1}. \tag{3.5.16}$$

If this holds, it computes $\hat{k}_j \leftarrow f_{\mathrm{gl}}(\bar{Z}_j, R)$ and then computes the rest of the $\hat{k}_i$ normally (as $f_{\mathrm{gl}}(Y^{x_i}, R)$) and decrypts $\hat{c}$ using $\hat{k} \leftarrow \hat{k}_1 \dots \hat{k}_\ell$. This completes the decryption of H_j.

We let the hybrid game H'_j be *exactly* like H_{j+1}, except that k_{j+1} is computed as $k_{j+1} \leftarrow f_{\mathrm{gl}}(Y^{x_{j+1}}, R)$ instead of being set to a random string. Note that in both H_j and H'_j, $k_1, \dots, k_j$ are set to random strings and $k_{j+1}, \dots, k_\ell$ are computed normally. The only difference between H_j and H'_j is the way in which the games are "managed," but the output distributions are exactly the same. The change between H_j and H'_j is essentially like the change between Games 1 and 2 in the proof of Theorem 5, and the same argument there can be applied here. The essential difference between H_j

and H'_j is which elements in the key are "trapdoor elements": in H_j they are $X_j, \widetilde{X}_j$ while in H'_j they are $X_{j+1}, \widetilde{X}_{j+1}$.

We are now ready to describe our adversary that breaks the hardcore-ness of f_{gl}. Let $\mathcal{B}_{\mathrm{gl}}$ be an adversary that gets (X, X', Y, R, s) as input, where either $s = f_{\mathrm{gl}}(\mathrm{dh}(X, Y), R)$ or s is a random string. In addition, $\mathcal{B}_{\mathrm{gl}}$ has access to an oracle computing $\mathrm{2dhp}(X, X', \hat{Y}, \hat{Z}, \hat{Z}')$.

$\mathcal{B}_{\mathrm{gl}}$ does the following. It selects $j \overset{\$}{\leftarrow} \{1, \ldots, \ell\}$, sets $X_j \leftarrow X$, and $X_{\ell+1} \leftarrow X'$. It proceeds to simulate H'_{j-1} for $\mathcal{A}$, except that it sets $k_j \leftarrow s$. The only point where x_j and $x_{\ell+1}$ are used is in the consistency check in (3.5.16), which $\mathcal{B}_{\mathrm{gl}}$ can perform by using an oracle query as in the proof of Theorem 5. When $\mathcal{A}$ outputs $\hat{b}$, $\mathcal{B}_{\mathrm{gl}}$ checks if $\hat{b} = b$, and outputs 1 if this holds and 0 otherwise.

It is not hard to check that, conditioned on $s = f_{\mathrm{gl}}(\mathrm{dh}(X, Y), R)$, $\mathcal{B}_{\mathrm{gl}}$ simulates H'_{j-1} for $\mathcal{A}$, and conditioned on the event that s was random, $\mathcal{B}_{\mathrm{gl}}$ simulates H_j. Then the following standard hybrid argument applies.

$$\begin{aligned}
&\Pr\left[\mathcal{B}_{\mathrm{gl}}(X, Y, R, s) = 1 \,|\, s = f_{\mathrm{gl}}(\mathrm{dh}(X, Y), R)\right] - \Pr\left[\mathcal{B}_{\mathrm{gl}}(X, Y, R, s) = 1 \,|\, s \text{ is random}\right] \\
&= \frac{1}{\ell}\sum_{j=1}^{\ell} \Pr\left[\hat{b} = b \text{ in } \mathsf{H}'_{j-1}\right] - \frac{1}{\ell}\sum_{j=1}^{\ell} \Pr\left[\hat{b} = b \text{ in } \mathsf{H}_j\right] \\
&= \frac{1}{\ell}\sum_{j=1}^{\ell} \left(\Pr\left[\hat{b} = b \text{ in } \mathsf{H}'_{j-1}\right] - \Pr\left[\hat{b} = b \text{ in } \mathsf{H}_j\right]\right) \\
&= \frac{1}{\ell}\left(\Pr\left[\hat{b} = b \text{ in } \mathsf{H}'_0\right] - \Pr\left[\hat{b} = b \text{ in } \mathsf{H}_\ell\right]\right) \qquad (3.5.17) \\
&= \frac{1}{\ell}\left(\Pr[S_1] - \Pr[S_0]\right)
\end{aligned}$$

We get (3.5.17) by recalling that $\Pr\left[\hat{b} = b \text{ in } \mathsf{H}_j\right] = \Pr\left[\hat{b} = b \text{ in } \mathsf{H}'_j\right]$ for $j = 1, \ldots, \ell - 1$.

Finally, if the advantage of $\mathcal{B}_{\mathrm{gl}}$ is non-negligible, then by Theorem 7 (augmented with our trapdoor test), we get an adversary $\mathcal{B}_{\mathrm{dh}}$ that solves the DH problem with non-negligible advantage. Then, by the DH assumption, (3.5.14) follows.

Returning to the proof of the theorem, in Game 2 the adversary is mounting a chosen-ciphertext attack against the symmetric encryption scheme. Thus, by the CCA security of SE,

$$\Pr[S_2] = \text{negl}(k). \tag{3.5.18}$$

The proof is completed by combining (3.5.12), (3.5.13), (3.5.14), and (3.5.18). $\square$

3.6 Identity-Based Encryption

In this section we show how to apply the trapdoor test in Theorem 1 to identity-based encryption. We give a bilinear version of the strong twin DH problem and show that it can be reduced to the standard bilinear DH problem. We then use this assumption to construct a new IBE scheme that we call twin Boneh-Franklin. While our scheme is not as computationally efficient as some other CCA secure schemes, it only incures one group element of overhead in the ciphertexts and has tighter reduction in the random oracle model to the BDH assumption than the original (CPA) scheme on which it is based.

3.6.1 The BDH and Twin BDH Assumptions

In group schemes equipped with a pairing $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, we can define the following function. Let g be a generator of $\mathbb{G}$.

$$\text{bdh}(X, Y, W) = Z, \quad \text{where } X = g^x,\ Y = g^y,\ W = g^w, \text{ and } Z = \hat{e}(g, g)^{wxy}.$$

Computing $\text{bdh}(X, Y, W)$ for random $X, Y, W \in \mathbb{G}$ is the *bilinear DH (or BDH) problem* [21]. For an adversary $\mathcal{B}$, let us define his *BDH advantage*, denoted $\mathsf{AdvBDH}_{\mathcal{B},\mathbb{G}}$, as the probability that $\mathcal{B}$ computes $\text{bdh}(X, Y, W)$ for random $X, Y, W \in \mathbb{G}$. The *BDH assumption* states that solving the BDH problem is hard, that is, that all probabilistic polynomial-time adversaries have negligible BDH advantage. Next we define a predicate

$$\mathrm{bdhp}(X, \hat{Y}, \hat{W}, \hat{Z}) = \begin{cases} 1 & \text{if } \mathrm{bdh}(X, Y, W) = Z \\ 0 & \text{otherwise} \end{cases} .$$

We can also consider the BDH problem where, in addition to random (X, Y, W), one is also given access to an oracle that on input $(\hat{Y}, \hat{W}, \hat{Z})$ returns $\mathrm{bdhp}(X, \hat{Y}, \hat{W}, \hat{Z})$. The *strong BDH assumption* [56] states that the BDH problem remains hard even with the help of the oracle.

For reasons similar to the issue with hashed ElGamal encryption, the strong BDH assumption seems necessary to prove the CCA security of the basic version [56] of the original Boneh-Franklin IBE [21]. We can repeat the "twinning" idea and define the *twin BDH problem*, where one must compute $2\mathrm{bdh}(X_1, X_2, Y, W)$ for random X_1, X_2, Y, W, where we define

$$2\mathrm{bdh}(X_1, X_2, Y, W) = (\mathrm{bdh}(X_1, Y, W), \mathrm{bdh}(X_2, Y, W)).$$

The *strong twin BDH problem* is the same as the twin BDH problem, but the adversary has access to an oracle computing the predicate

$$2\mathrm{bdhp}(X_1, X_2, \hat{Y}, \hat{W}, \hat{Z}_1, \hat{Z}_2) = \begin{cases} 1 & \text{if } 2\mathrm{bdh}(X_1, X_2, \hat{Y}, \hat{W}) = (\hat{Z}_1, \hat{Z}_2) \\ 0 & \text{otherwise} \end{cases}$$

for $\hat{Y}, \hat{W}, \hat{Z}_1, \hat{Z}_2$ of its choice. For an adversary $\mathcal{B}$, define his *strong twin BDH advantage*, denoted $\mathsf{Adv2BDH}_{\mathcal{B},\mathbb{G}}$, to be the probability that $\mathcal{B}$ computes $\mathrm{bdh}(X, Y, W)$ when given random $X, Y, W \in \mathbb{G}$ along with access to an oracle for the predicate $2\mathrm{bdhp}(X_1, X_2, \cdot, \cdot, \cdot, \cdot)$, which on input $\hat{Y}, \hat{W}, \hat{Z}_1, \hat{Z}_2$ returns $2\mathrm{bdhp}(X_1, X_2, \hat{Y}, \hat{W}, \hat{Z}_1, \hat{Z}_2)$. The *strong twin BDH assumption* states that the BDH problem is still hard, even with access to the decision oracle.

We will need a slight generalization of the trapdoor test in Theorem 1 to prove the following theorem. It is easy to check that Theorem 1 is still true if the elements $\hat{Z}_1, \hat{Z}_2$ are in a *different* cyclic group of the same order (we will take them in the range group of the pairing), and we replace $\hat{Y}$ with $\hat{e}(\hat{Y}, \hat{W})$. With this observation, the proof of the following theorem is almost identical to proof of Theorem 2 and is omitted.

Theorem 9. *Suppose $\mathcal{B}_{\text{2bdh}}$ is a strong twin BDH adversary that makes at most Q_{d} queries to its decision oracle, and runs in time at most τ. Then there exists a BDH adversary $\mathcal{B}_{\text{bdh}}$ with the following properties: $\mathcal{B}_{\text{bdh}}$ runs in time at most τ, plus the time to perform $O(Q_{\text{d}} \log q)$ group operations and some minor bookkeeping; moreover,*

$$\mathsf{Adv2BDH}_{\mathcal{B}_{\text{2bdh}},\mathbb{G}} \leq \mathsf{AdvBDH}_{\mathcal{B}_{\text{bdh}},\mathbb{G}} + \frac{Q_{\text{d}}}{q}.$$

In addition, if $\mathcal{B}_{\text{bdh}}$ does not output "failure," then its output is correct with probability at least $1 - 1/q$.

3.6.2 Twin Boneh-Franklin

Theorem 9 admits a simple analysis of the following IBE scheme, which we call the *twin Boneh-Franklin IBE scheme*. This scheme will use a bilinear pairing scheme $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ where each of the groups is of prime order q, and g is a generator of $\mathbb{G}$. It will also use two hash functions, H (which outputs symmetric keys) and G (which outputs group elements), and a symmetric cipher (E, D). A master public key is a pair of group elements (X_1, X_2), where $X_i = g^{x_i}$ for $i = 1, 2$. The master private key is (x_1, x_2), which are selected at random from $\mathbb{Z}_q$ by the setup algorithm. The secret key for an identity $id \in \{0,1\}^*$ is $(S_1, S_2) = (\mathsf{G}(id)^{x_1}, \mathsf{G}(id)^{x_2})$. To encrypt a message m for identity id, one chooses $y \in \mathbb{Z}_q$ at random and sets

$$Y \leftarrow g^y, \quad Z_1 \leftarrow \hat{e}(\mathsf{G}(id), X_1)^y, \quad Z_2 \leftarrow \hat{e}(\mathsf{G}(id), X_2)^y,$$
$$k \leftarrow \mathsf{H}(id, Y, Z_1, Z_2), \quad c \xleftarrow{\$} \mathsf{E}_k(m).$$

The ciphertext is (Y, c). To decrypt using the secret key (S_1, S_2) for id, one computes

$$Z_1 \leftarrow \hat{e}(S_1, Y), \quad Z_2 \leftarrow \hat{e}(S_2, Y), \quad k \leftarrow \mathsf{H}(id, Y, Z_1, Z_2), \quad m \leftarrow \mathsf{D}_k(c).$$

We shall denote this scheme $\mathsf{IBE}_{\mathrm{2bdh}}$. Now we can essentially borrow the analysis of the original Boneh-Franklin scheme under the strong BDH assumption [56], except now we get that the scheme is secure against chosen ciphertext attacks under the strong twin BDH assumption. By Theorem 9, we get that the above IBE scheme is CCA secure under the BDH assumption if the symmetric cipher is secure and the hash functions are treated as random oracles. This is captured in the following theorem.

Theorem 10. *Suppose H and G are modeled as random oracles. Further, suppose the BDH assumption holds in $\mathbb{G}$, and that the symmetric cipher $\mathsf{SE} = (\mathsf{E}, \mathsf{D})$ is secure against chosen ciphertext attack. Then $\mathsf{IBE}_{\mathrm{2bdh}}$ is secure against chosen ciphertext attack.*

In particular, suppose $\mathcal{A}$ is an adversary that carries out a chosen ciphertext attack against $\mathsf{IBE}_{\mathrm{2bdh}}$ in the random oracle model, and that $\mathcal{A}$ runs in time τ, and makes at most Q_{h} hash queries, Q_{d} decryption queries, and Q_{id} user secret key queries. Then there exists a BDH adversary $\mathcal{B}_{\mathrm{bdh}}$ and an adversary $\mathcal{B}_{\mathrm{sym}}$ that carries out a chosen ciphertext attack against SE, such that both $\mathcal{B}_{\mathrm{bdh}}$ and $\mathcal{B}_{\mathrm{sym}}$ run in time at most τ, plus the time to perform $O((Q_{\mathrm{id}} + Q_{\mathrm{h}} + Q_{\mathrm{d}}) \log q)$ group operations; moreover,

$$\begin{aligned}&\mathsf{AdvCCA}^{\mathrm{ro}}_{\mathcal{A},\mathsf{IBE}_{\mathrm{2bdh}}} \leq \\ &\qquad e \cdot (Q_{\mathrm{id}} + 1) \cdot \left(\frac{2Q_{\mathrm{h}} + Q_{\mathrm{d}}}{q} + \mathsf{AdvBDH}_{\mathcal{B}_{\mathrm{bdh}},\mathbb{G}} + \mathsf{AdvCCA}_{\mathcal{B}_{\mathrm{sym}},\mathsf{SE}} \right).\end{aligned}$$

Proof. As with our other proofs, we proceed with a sequence of games.

Game 0. Let Game 0 be the original IBE chosen ciphertext attack game, and let S_0 be the event that $\hat{b} = b$ in this game.

The challenger chooses the master private key (x_1, x_2) and gives the adversary the corresponding master public key (X_1, X_2) as normal. To track random oracle responses, the challenger uses two associative arrays L and K. L will store responses for G and K will store responses for H, and both will initially have all entries set to $\perp$.

When processing a random oracle response, the adversary returns the corresponding entry if it is defined, and otherwise initializes it with an appropriate random value and returns that. Apart from this bookkeeping, the challenger runs Game 0 exactly as specified in the definition, and we have

$$\mathsf{AdvCCA}^{\mathrm{ro}}_{\mathcal{A},\mathsf{IBE}_{2\mathrm{bdh}}} = |\Pr[S_0] - 1/2|. \tag{3.6.1}$$

Game 1. Game 1 will be like Game 0, but now we change how the challenger processes queries to G. Now, in addition to inserting oracle responses into L, the challenger also "marks" some entries in the L array used to store G responses. On query $\mathsf{G}(\hat{id})$, in addition to the normal processing, with probability δ the challenger marks $L[\hat{id}]$. The challenger completely hides the marks from the adversary.

At the end of the game, the challenger looks at L and decides if it should abort the game. For each user secret key query that the adversary issued during the game, the challenger checks if the entry in L for that identity is marked. If any of them are marked, the challenger aborts the game. Finally it checks the entry $L[id]$, where id is the identity from the challenge query. If that entry is *not* marked, then the challenger aborts. Otherwise it proceeds normally.

Let S_1 be the event that $\hat{b} = b$ in Game 1 and F_1 be the event that the challenger aborts. Since the coins that determine F_1 are independent of the rest of the game, it follows that

$$|\Pr[S_1] - \Pr[S_0]| = \Pr[F_1] \leq \delta \cdot (1-\delta)^{Q_{\mathrm{id}}},$$

and if we set $\delta = 1/(1+Q_{\mathrm{id}})$,

$$|\Pr[S_1] - \Pr[S_0]| \leq (e(1+Q_{\mathrm{id}}))^{-1}. \tag{3.6.2}$$

Game 2. Game 2 will be like Game 1, except that now the challenger sets up some of the challenge ciphertext in advance. Before starting the game, it chooses a random symmetric key k, random $y \in \mathbb{Z}_q$ and random $W \in \mathbb{G}$, sets $Y \leftarrow g^y$, $Z_1 \leftarrow \hat{e}(W, X_1)^y$

and $Z_2 \leftarrow \hat{e}(W, X_2)^y$.

Now the challenger uses these values in the rest of the game. When creating the challenge ciphertext, the challenger sets $K[id, Y, Z_1, Z_2] \leftarrow k$ (overwriting the entry if it is already defined), computes $c \xleftarrow{\$} \mathsf{E}(k, m_b)$, and returns (Y, c).

For decryption queries, when the adversary asks for the decryption of $(\hat{Y}, \hat{c})$ under identity $\hat{id}$, if $\hat{id} = id$, $L[id]$ is marked, and $\hat{Y} = Y$, then the challenger uses k to decrypt $\hat{c}$. Otherwise, the challenger decrypts normally.

For the challenge query, the challenger uses k to compute $c \xleftarrow{\$} \mathsf{E}(k, m_b)$ and returns (Y, c).

Let S_2 be the event that $\hat{b} = b$ in Game 2. Since Game 2 and Game 1 only differ when the adversary manages to query $\mathsf{H}(id, Y, Z_1, Z_2)$ *before* the challenge query, and this event only happens if the adversary can guess Y, an independently chosen group element. Thus

$$|\Pr[S_2] - \Pr[S_1]| \leq Q_{\mathrm{H}}/q. \tag{3.6.3}$$

Game 3. Game 3 will include one simple change from Game 2: it no longer immediately stores the value k in K as described in Game 2. Instead, it leaves that entry unchanged, but still uses the k, Y, Z_1, Z_2 generated at the beginning of the game to generate the challenge ciphertext.

Let S_3 be the event that $\hat{b} = b$ in Game 3. Let F_{2bdh} be the event that the adversary queries H at (id, Y, Z_1, Z_2), where id is the identity used in the challenge ciphertext. Since Game 2 and Game 3 are exactly the same when F_{2bdh} does not occur, it follows that

$$|\Pr[S_3] - \Pr[S_2]| \leq \Pr[F_{\mathrm{2bdh}}]. \tag{3.6.4}$$

We claim that

$$\Pr[F_{\mathrm{2bdh}}] \leq \mathsf{Adv2BDH}_{\mathcal{B}_{\mathrm{2bdh}},\mathbb{G}}, \tag{3.6.5}$$

for an efficient strong twin BDH adversary $\mathcal{B}_{\text{2bdh}}$ that makes $Q_{\text{h}} + Q_{\text{d}}$ decision oracle queries. We give a high level description of $\mathcal{B}_{\text{2bdh}}$. $\mathcal{B}_{\text{2bdh}}$ gets (X_1, X_2, Y, W) as input and begins to run Game 3, acting as the challenger for the adversary. Of course, it sets the master public key to (X_1, X_2) and uses (Y, c) as challenge ciphertext, where $c \overset{\$}{\leftarrow} \mathsf{E}_k(m_b)$, as in Game 3.

We need to describe how $\mathcal{B}_{\text{2bdh}}$ answers queries for the random oracles and user secret keys. When the adversary requests $\mathsf{G}(\hat{id})$, if that entry gets marked, $\mathcal{B}_{\text{2bdh}}$ chooses a new random $r \in \mathbb{Z}_q$, sets $L[\hat{id}] \leftarrow Wg^r$, and gives Wg^r to the adversary. If the entry does not get marked, $\mathcal{B}_{\text{2bdh}}$ returns g^r instead. (Note that $\mathcal{B}_{\text{2bdh}}$ can respond with the corresponding user secret key for unmarked identities.) In either case, r is remembered for later.

When the adversary requests the user secret key for an unmarked identity $\hat{id}$, $\mathcal{B}_{\text{2bdh}}$ retrieves the r used to generate the entry g^r in $L[\hat{id}]$, and returns (X_1^r, X_2^r). If the adversary requests the user secret key for a marked identity, $\mathcal{B}_{\text{2bdh}}$ immediately aborts.

For H queries, $\mathcal{B}_{\text{2bdh}}$ implements the same oracle patching idea used in the proof of Theorem 3. On query $\mathsf{H}(\hat{id}, \hat{Y}, \hat{Z}_1, \hat{Z}_2)$, $\mathcal{B}_{\text{2bdh}}$ looks up $\hat{W}$ stored at $L[\hat{id}]$ and queries its decision oracle with $(\hat{Y}, \hat{W}, \hat{Z}_1, \hat{Z}_2)$, and marks the tuple as "good" or "bad" depending on the answer. If it finds a good tuple, it uses the corresponding key to decrypt ciphertexts with $\hat{Y}$. Otherwise, it generates a random symmetric key to use with those ciphertexts, and watches for a good tuple to come up as a hash query. When it sees one, it "patches" that query by returning the symmetric key generated earlier.

After the game ends, $\mathcal{B}_{\text{2bdh}}$ checks that the identity from the test query was unmarked. If not, $\mathcal{B}_{\text{2bdh}}$ aborts. Otherwise, it examines K and looks for a good entry of the form $K[id, Y, Z_1, Z_2]$ (where id and Y are from the test query). If it finds one, it looks up the $\hat{W} = Wg^r$ and corresponding r and outputs $(Z_1/\hat{e}(X_1, Y)^r, Z_2/\hat{e}(X_2, Y)^r)$.

It is straightforward to check that $\mathcal{B}_{\text{2bdh}}$ solves the strong twin BDH problem whenever the event F_{2bdh} would happen in Game 3.

Finally, in Game 3 the adversary is essentially playing the chosen ciphertext game against SE. Thus there is an adversary $\mathcal{B}_{\text{sym}}$ such that

$$|\Pr[S_1] - 1/2| = \mathsf{AdvCCA}_{\mathcal{B}_{\text{sym}},\mathsf{SE}}. \tag{3.6.6}$$

The theorem follows by combining (3.6.1)–(3.6.6). □

We remark that our ideas can also be applied to the IBE scheme from Sakai-Kasahara [71]. The resulting IBE scheme is more efficient, but its security can only be proved based on the (computational) q-BDHI assumption [19].

CHAPTER IV

NON-MALLEABLE HASH FUNCTIONS AND THEIR APPLICATION TO ENCRYPTION

In this chapter we develop the notion of *non-malleability for hash functions* and show how to analyze an encryption scheme of Bellare and Rogaway [11] using our notion. We start with several definitions from prior work that are used below.

4.1 Preliminaries

4.1.1 Hash Functions

We start with a definition of a *hash function*. In order to study our new security properties, it will be necessary to use a definition that is slightly more general than usual. In particular, we will use a definition that allows for randomized hash functions, which can produce several possible hashes for fixed input. Unlike our treatment of hash functions in §3, we will make the concept of a hash key explicit, since how and when the key is used will become important below.

A *hash function* consists of three algorithms.

- A probabilistic polynomial-time algorithm HK, which on input a security parameter λ (in unary), outputs a hash key K. We assume that K implicitly defines a domain D_K of messages for the hash function.

- A probabilistic polynomial-time algorithm H for inputs K and $x \in D_K$ returns a value $y \in \{0,1\}^*$.

- A polynomial-time algorithm HVf that, on inputs K, x, y, returns a bit.

We will only consider hash functions that satisfy a basic correctness requirement.

Specifically, let K be any key that is possibly output by $\mathsf{HK}(1^\lambda)$, and let $x \in D_K$. We require that $b = 1$ with probability 1, where b is computed as

$$y \xleftarrow{\$} \mathsf{H}(K, x), \quad b \leftarrow \mathsf{HVf}(K, x, y).$$

We remark that, for deterministic hash functions, verification can simply recompute the hash value can compare it to the given value.

4.1.2 Collision Resistance

We recall the standard definition of *collision resistance* for hash functions, adapted to our general version of hash functions. Let $(\mathsf{HK}, \mathsf{H}, \mathsf{HVf})$ be a hash function. For an adversary $\mathcal{A}$, we consider the following game, played with challenger.

1. The challenger runs $K \xleftarrow{\$} \mathsf{HK}(1^\lambda)$, and gives K to the adversary.

2. The adversary outputs bitstrings x_0, x_1, y.

We define the term $\mathsf{AdvCR}_{\mathsf{H},\mathcal{A}}(\lambda)$ to be the probability that

$$x_0 \neq x_1 \quad \wedge \quad \mathsf{HVf}(K, x_0, y) = 1 \quad \wedge \quad \mathsf{HVf}(K, x_1, y) = 1$$

holds. We say that the hash function H is *collision resistant* if $\mathsf{AdvCR}_{\mathsf{H},\mathcal{A}}(\lambda)$ is negligible for all probabilistic polynomial-time adversary $\mathcal{A}$.

4.1.3 Perfect One-wayness

We recall the notion of *perfect one-wayness* [26], which some adaptations to our setting.

Definition 4. *Let* H *be a hash function. We say that* H *is* perfectly one-way with respect to predicate P and side information function hint *if, for every probabilistic polynomial-time adversary* $\mathcal{B}$*, the following two distributions are indistinguishable.*

The distributions are indexed by a bit b*. They are produced by the following procedure.*

1. *A key $K \xleftarrow{\$} \mathsf{HK}(1^\lambda)$ is sampled and given to the $\mathcal{B}$.*

2. *$\mathcal{B}$ returns a description of a message distribution $\mathcal{X}$.*

3. *The challenger selects $x \xleftarrow{\$} \mathcal{X}$, $\sigma \xleftarrow{\$} \mathsf{hint}(K, x)$. Then if $b = 0$, it computes $y \xleftarrow{\$} \mathsf{H}(K, x)$. If $b = 1$, it samples $x' \xleftarrow{\$} \mathcal{X}$ and sets $y \leftarrow \mathsf{H}(K, x')$. It gives y, σ to $\mathcal{B}$.*

4. *$\mathcal{B}$ outputs a bit d.*

5. *The procedure checks if $P(\mathcal{X}) = 1$. If so, it outputs (K, x, d). Otherwise it outputs $\perp$.*

As pointed out in [26, 29] the definition only makes sense if hint is an uninvertible function of the input (such that finding the pre-image x from $\mathsf{hint}(x)$ is infeasible) and $\mathcal{B}$ only outputs descriptions of well- spread distributions (i.e., those with super-logarithmic min-entropy). Otherwise the notion is impossible to achieve.

Perfectly one-way hash functions (in the sense above) can be constructed from any one-way permutation [29, 38] (for the uniform input distribution), any regular collision-resistant hash function [29] (for any distribution with fixed, super-logarithmic min-entropy), or under the decisional Diffie-Hellman assumption [26] (for the uniform distribution). Usually, these general constructions are not known to be secure assuming arbitrary functions hint, but for the particular function hint required by the application, they can often be adapted accordingly. A concrete example is given below, in our discussion of the Bellare-Rogaway encryption scheme.

4.1.4 Trapdoor Permutations and Partial One-wayness

We review the standard concept of *trapdoor permutations* [76]. Formally, we define a family of trapdoor permutations $\mathcal{F}$ as a triple of three algorithms.

- A probabilistic polynomial-time algorithm TDPGen that, on input λ (in unary) outputs a description of a function f and a trapdoor τ. We assume that f is a permutation on $\{0,1\}^\lambda$.

- A polynomial-time algorithm, that, on input f and x in the domain of f, outputs $f(x)$.
- A polynomial-time algorithm that on input y in the range of f and a trapdoor τ, computes $f^{-1}(y)$.

We will be interested in trapdoor permutation families that are *partial one-way.* We say that a permutation family is partial one-way if, for $(f,\tau) \overset{\$}{\leftarrow} \mathsf{TDPGen}(1^\lambda)$ and a random x in $\{0,1\}^\lambda$, any probabilistic polynomial-time adversary $\mathcal{A}$ that is given $f, f(x)$ outputs the $\lambda/2$ most significant bits of x with only negligible probability.

We remark that this is still a mild hardness requirement for a trapdoor function. For example, the RSA trapdoor function is known to be partial one-way under the RSA assumption [42]. Moreover, these exist simple generic constructions from any one-way trapdoor permutation.

4.2 Non-Malleability of Hash Functions

Our definition for hash functions follows the classical (simulation-based) approach for defining non-malleability [36]. Informally, our definition requires that the success probability of an adversary, given a hash value h, finding another value h^*, such that the pre-images are related, is negligibly close to the one of a simulator which does not see h.

In the adversary's attack we consider a three-stage process. The adversary first selects a distribution $\mathcal{X}$ from which an input x is then sampled. In the second stage the algorithm sees a hash value h of this input x, and the adversary's goal is to create another hash value h^* (different from h). In the third stage the adversary is given x and now has to output a pre-image x^* to h^* which is "related" to x. The simulator may also pick a distribution $\mathcal{X}$ according to which x is sampled, but then the simulator needs to specify x^* directly from the key of the hash function only.

In the second stage the adversary (and consequently the simulator) also gets as input a "hint" σ about the original pre-image x, to represent some side a-priori information potentially gathered from other executions of other protocols in which x is used. As in the case of non-malleable commitments and encryption, related pre-images are define via a relation $R(x, x^*)$ which also depends on the distribution $\mathcal{X}$ to catch significantly diverging choices of the adversary and the simulator and to possibly restrict the choices for $\mathcal{X}$, say, to require a certain min-entropy. However, unlike for these primitives, we do not measure the success of the adversary and the simulator for arbitrary relations R between x and x^*, but instead restrict R to a class $\mathcal{R}$ of admissible relations. We discuss the necessity of this restriction and other subtleties after the definition:

Definition 5 (Non-Malleable Hash functions). *For some hash function* H*, we define two games used in the definition. These games are parameterized by* $\lambda \in \mathbb{Z}^+$*, a relation* R*, and a side-information function* hint*.*

The first is the real game *between an adversary* $\mathcal{A}$ *an a challenger, and second is the* simulated game *between a simulator* $\mathcal{S}$ *and a different challenger. We say that the hash function is* non-malleable with respect to R *if, for every probabilistic polynomial-time adversary* $\mathcal{A}$*, there exists a probabilistic polynomial-time simulator* $\mathcal{S}$ *such that*

$$\left|\Pr[\mathcal{A} \textit{ wins the real game}] - \Pr[\mathcal{S} \textit{ wins the simulated game}]\right| \tag{4.2.1}$$

is negligible (as a function of λ*), where we define these events as follows.*

We now define the real game.

1. *The challenger runs* $\mathsf{HK}(1^\lambda)$ *to generate a hash key* K*, and gives it to* $\mathcal{A}$*.*

2. *$\mathcal{A}$ responds with a circuit describing a message sampler, denoted $\mathcal{X}$.*

3. *The challenger runs $\mathcal{X}$ on a random input to sample a message x. It then computes $\sigma \xleftarrow{\$} \mathsf{hint}(K, x)$ and $h \xleftarrow{\$} \mathsf{H}(K, x)$. It returns h, σ to $\mathcal{A}$.*

4. *$\mathcal{A}$ outputs a hash h^*.*

5. *The challenger gives x to $\mathcal{A}$, and finally $\mathcal{A}$ returns some message x^**

We say that $\mathcal{A}$ wins the real game *if the following hold:*

$$R(\mathcal{X}, x, x^*) = 1, \quad (x, h) \neq (x^*, h^*), \quad \mathsf{HVf}_K(x^*, h^*) = 1. \tag{4.2.2}$$

Now we define the simulated game as follows.

1. *The challenger runs $\mathsf{HK}(1^\lambda)$ to generate a hash key K, and gives it to $\mathcal{S}$.*

2. *$\mathcal{S}$ responds with a circuit describing a message sampler, denoted $\mathcal{X}$.*

3. *The challenger runs $\mathcal{X}$ on a random input to sample a message x. It then computes $\sigma \xleftarrow{\$} \mathsf{hint}(K, x)$. It returns σ to $\mathcal{S}$.*

4. *$\mathcal{S}$ outputs a message x^*.*

We say that the simulator $\mathcal{S}$ wins the simulated game *if*

$$R(\mathcal{X}, x, x^*) = 1. \tag{4.2.3}$$

We extend the definition to a class of relations $\mathcal{R}$ in the natural way: A hash function is non-malleable with respect to $\mathcal{R}$ *if it is non-malleable with respect to every $R \in \mathcal{R}$.*

Our definition only fixes a relation R or class of relations $\mathcal{R}$. This is because, for some relations, the definition is simply not achievable, as in the case when the relation involves the hash of x instead of x itself. For example, consider the relation $R(\mathcal{X}, x, x^*)$

which parses x^* as K, h and outputs $\mathsf{HVf}(K, x, h)$. Then, an adversary, on input h, σ, may output $h^* \xleftarrow{\$} \mathsf{H}(K, (K, h))$ and then, given x, returns $x^* = (K, h)$. This adversary succeeds in the real game with probability 1. In contrast, any simulator is likely to fail, as long as the hash function does not have "weak" keys, i.e., keys for which the distribution of generated images is not well-spread (such that the simulator can guess h with sufficiently high probability).

We avoid this difficulty by only requiring the definition to hold for a subset $\mathcal{R}$ of all relations. It is, of course, desirable to seek secure constructions with respect to very broad classes of relations, which are more useful. At the same time, certain scenarios may only require non-malleability with respect to a small set of relations, as in the case of encryption below. For virtually all "interesting" functions H and relation classes $\mathcal{R}$ the definition is achievable only for adversaries and simulators that output descriptions of well-spread distributions $\mathcal{X}$ in the first stage, and "uninvertible" functions hint.

In the case of non-malleable encryption, the original simulation-based definition of [36] was later shown to be equivalent to an indistinguishability-based definition [13]. The superficial similarity between our definition of non-malleable hash functions and the one of non-malleable encryption suggests that this may be possible here as well. Surprisingly, straightforward attempts to define non-malleability of hash functions through indistinguishability do not seem to yield an equivalent definition. We leave it as an interesting open problem to find a suitable indistinguishability-based definition for non-malleable hash functions.

4.3 *Application to Encryption*

In this section we confirm the usefulness of our definition for cryptographic applications.

We recall the encryption scheme proposed by Bellare and Rogaway in [11], which we will denote $\mathsf{PKE}_{\mathrm{br}}$. Let $\mathcal{F}$ be a familiy of trapdoor permutations and G, H be random oracles. The message space of the scheme is the range of G. The key generation algorithm, on input $\lambda \in \mathbb{Z}^+$, samples (f, τ) from the trapdoor permutation family and outputs f as the public key and τ as the secret key. The encryption algorithm, on inputs $pk = f$ and message m, picks random r in the domain of f and computes

$$y \leftarrow f(r), \quad g \leftarrow G(r) \oplus m, \quad h \leftarrow H(r \| m).$$

Finally, encryption computes $c \leftarrow (y, g, h)$ and outputs c as the ciphertext. The decryption algorithm, on inputs $sk = \tau, \hat{c} = (\hat{y}, \hat{g}, \hat{h})$, computes

$$\hat{r} \leftarrow f^{-1}(\hat{y}), \quad \hat{m} \leftarrow \hat{g} \oplus G(\hat{r}).$$

It outputs $\hat{m}$ if $H(\hat{r} \| \hat{m}) = \hat{h}$ holds, and it rejects if not. This completes the description of the scheme $\mathsf{PKE}_{\mathrm{br}}$.

The scheme $\mathsf{PKE}_{\mathrm{br}}$ was proven to be CCA secure, in the random oracle model, assuming that $\mathcal{F}$ is one-way.

We study *partial H-instantiations* of the scheme. That is, we consider the possibility of analyzing the scheme when H is instantiated with a function family $\mathcal{H} = (\mathsf{HK}, \mathsf{H}, \mathsf{HVf})$.

More precisely, we consider a modified version of the scheme where the public and secret keys also contains a key $K \overset{\$}{\leftarrow} \mathsf{HK}(1^\lambda)$ specifying a function, and encryption computes $\mathsf{H}(K, r \| m)$ instead of $H(r \| m)$. Decryption computes $\mathsf{HVf}(K, r \| m, h)$ instead of checking whether $H(r \| m) = h$.

Before stating the theorem, we must address one technicality. It will become necessary to assume that the (non-malleable) hash function used below *includes a random sample from a trapdoor permutation family in the key.* That it is, in addition to using a hash key K, it will also sample f, which will not be used anywhere in the hash, and set the key to $K' = (K, f)$. One could remove this assumption by allowing the various algorithms involved to take an additional input, but this assumption seems is sufficient and quite reasonable. In particular, the known construction from one-way functions achieves the definition with this modification.

We fix some notation for the parameters used in the theorem. Let

- $\mathrm{msb} : \{0,1\}^\lambda \to \{0,1\}^{\lambda/2}$ output the $\lambda/2$ most significant bits of its input.
- $\mathsf{hint}_{\mathrm{br}}$ that takes as input a key (K, f) and message x, and outputs $f(\mathrm{msb}(x))$.
- Predicate $P_{\mathrm{br}}(\mathcal{X})$ output 1 if and only if $\mathcal{X}$ is a canonical distribution that samples uniform $r \overset{\$}{\leftarrow} \{0,1\}^{\lambda/2}$, and then samples an arbitrary $m \in \{0,1\}^{\lambda/2}$, and outputs $r \| m$.
- The relation $R_{\mathrm{br}}(\mathcal{X}, x, \hat{x})$ which outputs 1 if and only if the following hold: (1) $P(\mathcal{X}) = 1$, and (2) $\mathrm{msb}(x) = \mathrm{msb}(\hat{x})$.

Theorem 11. *Let $\mathcal{F}$ be a family of trapdoor permutations and let $\mathcal{H} = (\mathsf{HK}, \mathsf{G}, \mathsf{HVf})$ be a hash function. Then $\mathsf{PKE}_{\mathrm{br}}$ is CCA secure, in the random oracle model, if $\mathcal{H}$ is*

1. *collision resistant,*
2. *non-malleable with respect to the relation R_{br} and side information $\mathsf{hint}_{\mathrm{br}}$,*
3. *perfectly one-way with respect to the predicate P_{br} and side information $\mathsf{hint}_{\mathrm{br}}$.*

Such non-malleable hash functions were shown to exist, assuming that one-way functions exist [16]. Perfectly one-way hash functions meeting our requirements were shown to exist under reasonable assumptions [29].

We note that, assuming only conditions (1) and (2), we can show that any successful CCA adversary can be turned into a successful CPA adversary against the scheme. The third condition is only needed to show that CPA security holds.

We also note that the security assumption on $\mathcal{F}$ is actually implicit due to the assumptions of $\mathcal{H}$. This is because the non-malleability of $\mathcal{H}$ with respect to our particular side information implicitly requires $\mathcal{F}$ to be partial one-way.

The proof appears in the next section, but before moving on we provide some intuition. Consider an adversary $\mathcal{A}$ that breaks CCA security of $\mathsf{PKE}_{\mathrm{br}}$. After issuing a challenge query m_0, m_1, it is given the challenge ciphertext of the form $(y, g, h) = (f(r), G(r) \oplus m_b, \mathsf{H}(K, r \| m_b))$ for a random string r and bit b, and tries to predict b. We first claim that the scheme is CPA secure, meaning that without decryption queries $\mathcal{A}$ cannot break security. This follows from condition (3) above, the perfect one-wayness. That is, if $\mathcal{B}$ has non-negligible advantage in determining b without making any decryption queries, then one can break adaptive perfect one-wayness of $\mathcal{H}$.

Next we show that decryption queries do not help $\mathcal{A}$. Assume that $\mathcal{A}$ makes decryption queries of the form $\hat{c} = (\hat{y}, \hat{g}, \hat{h})$. If $\mathcal{A}$ has queried oracle G about $\hat{r} = f^{-1}(\hat{y})$ before then we can easily find this entry in the list of G-queries and simulate the additional decryption steps. Else, consider the case that $\mathcal{A}$ has not made such a query to G but tries to succeed by mauling the challenge ciphertext $c = (y, g, h)$ to $\hat{c} = (\hat{y}, \hat{g}, \hat{h})$. Then it follows from the non-malleability of $\mathcal{H}$ that this ciphertext is likely to be invalid. The collision-resistance additionally prevents that $\mathcal{A}$ creates any other valid ciphertext without querying G about $\hat{r}$ beforehand.

4.3.1 Proof of Proposition 11 – Security of Bellare-Rogaway Encryption

We proceed with a sequence of games, where the first game is the original CCA game just described, and the final game is simple to analyze. We show that for each two adjacent games Game i and Game $i+1$, the adversary's advantage can change only negligibly. In the final game, we complete the proof by showing that the adversary's advantage is negligible.

Game 0. Let Game 0 be the original chosen-ciphertext attack game, and let S_0 be the event that the adversary wins this game. Then we have, by definition,

$$\Pr[S_0] = \mathsf{AdvCCA}^{\mathrm{ro}}_{\mathcal{A},\mathsf{PKE}_{\mathrm{br}}}(\lambda). \tag{4.3.1}$$

Below, we will denote by $c = (y, g, h)$ the challenge ciphertext submitted by $\mathcal{A}$.

Game 1. This is like Game 0, except that on decryption query $\hat{c} = (\hat{y}, \hat{g}, \hat{h})$, if $\hat{y} \neq y$ (where y is from the challenge ciphertext) and $\mathcal{B}$ has not queried an $\hat{r}$ to G such that $\hat{y} = f(\hat{r})$, Game 1 rejects the ciphertext. Everything else remains unchanged.

Let S_1 be the event that $\mathcal{A}$ wins Game 1, and let F_1 be the event that $\mathcal{B}$ submits a ciphertext in Game 1 that is rejected, but would not have been rejected in Game 0. It is clear that when F_1 does not occur, $\mathcal{B}$'s outputs in Game 0 and Game 1 have the same distribution. More precisely, we have

$$|\Pr[S_1] - \Pr[S_0]| \leq \Pr[F_1]. \tag{4.3.2}$$

We claim that $\Pr[F_1]$ is negligible, by the collision resistance of $\mathcal{H}$. We will show that there exists an adversary $\mathcal{B}_{\mathrm{cr}}$, which runs in about the same time as $\mathcal{A}$, such that

$$\mathsf{AdvCR}_{\mathcal{B}_{\mathrm{cr}},\mathcal{H}}(\lambda) \geq \big(\Pr[F_1]\big)^3/8. \tag{4.3.3}$$

The adversary $\mathcal{B}_{\mathrm{cr}}$ works as follows. It takes as input a hash key K and uses it in the public key in Game 1. It generates the rest of the public key/secret key pair itself and answers random oracle queries for $\mathcal{A}$, having full control over the random oracle G (i.e. answering each new query with a new random string and storing all queries and the relies in an array called G-list, so the repeated queries could be answered consistently). It also creates the challenge ciphertext as described by the protocol.

$\mathcal{B}_{\mathrm{cr}}$ handles decryption queries exactly as in Game 0 by using the secret key, except with the following difference. If $\mathcal{A}$ submits a decryption query $(\hat{y}, \hat{g}, \hat{h})$ that would get rejected in Game 1, but not Game 0, then $\mathcal{B}_{\mathrm{cr}}$ halts the game. $\mathcal{B}_{\mathrm{cr}}$ computes

$\hat{r} \leftarrow f^{-1}(\hat{y})$, $\hat{m} \leftarrow G(\hat{r}) \oplus \hat{g}$, generating the value $G(\hat{r})$ from scratch. Finally, $\mathcal{B}_{\mathrm{cr}}$ selects a new random message m, and outputs

$$(\hat{r}\|\hat{m},\ \hat{r}\|m), \quad \hat{h}$$

as its messages and hash for the collision resistance game. This completes the description of $\mathcal{B}_{\mathrm{cr}}$.

We now analyze the advantage of $\mathcal{B}_{\mathrm{cr}}$. Since the ciphertext $(\hat{y}, \hat{g}, \hat{h})$ would not be rejected in Game 0, we know that $\mathsf{HVf}(K, \hat{h}, \hat{r}\|\hat{m}) = 1$. Observe that when $\mathcal{B}_{\mathrm{cr}}$ computed $\hat{m} = G(\hat{r}) \oplus \hat{g}$, it was actually *setting $\hat{m}$ to a random message*. This is true because $\mathcal{B}$ had never queried $G(\hat{r})$, so $\mathcal{B}_{\mathrm{cr}}$ selected a random string for it after the game was over.

Let $\varepsilon = \Pr[F_1]$. Now condition on all of the random choices carried out by $\mathcal{B}_{\mathrm{cr}}$ until it selects $\hat{m}$ at random. It then follows by a standard averaging argument that, with probability $\varepsilon/2$ over of the choices made by $\mathcal{B}_{\mathrm{cr}}$, we have that the conditional probability of F_1 is at least $\varepsilon/2$ (over the choice of $G(r)$). This argument applies equally well when $\mathcal{B}_{\mathrm{cr}}$ selects a second message m after halting the game. Thus there is at least an $\varepsilon^2/4$ chance that m will satisfy $\mathsf{HVf}(K, h, \hat{r}\|m) = 1$. Thus the chance that $\mathcal{B}_{\mathrm{cr}}$ finds a collision is at least $\varepsilon^3/8$, which proves (4.3.3).

Game 2. This game is like Game 1, except now the decryption oracle applies the following rejection rules, in addition to all of the processing in Game 1. Now, for a decryption query $\hat{c} = (\hat{y}, \hat{g}, \hat{h})$, where $\hat{y} = y$ and $\hat{h} = h$ (and thus $\hat{g} \neq g$), such that the adversary has not queried $G(\hat{r})$ such that $\hat{y} = f(\hat{r})$, are rejected. Let S_2 be the event that $\mathcal{A}$ wins Game 2.

It is clear that until this reject rule is used, Games 1 and 2 are the same. Let F_2 be the event that a query is rejected because of this rule that would not have been rejected in Game 1. Then we have

$$|\Pr[S_2] - \Pr[S_1]| \leq \Pr[F_2]. \tag{4.3.4}$$

We claim that there exists an adversary $\mathcal{C}_{\mathrm{cr}}$ that runs in about the same time as $\mathcal{A}$, such that

$$\mathsf{AdvCR}_{\mathcal{C}_{\mathrm{cr}},\mathcal{H}}(\lambda) \geq \Pr[F_2]. \tag{4.3.5}$$

$\mathcal{C}_{\mathrm{cr}}$ that takes as input a hash key K and attempts to find a collision in $\mathcal{H}$. $\mathcal{C}_{\mathrm{cr}}$ uses K in the public key and runs Game 2 for $\mathcal{B}$ until the reject rule in Game 2 is used (again, controlling G and generating a valid challenge ciphertext).

Let $\hat{C} = (\hat{y}, \hat{g}, \hat{h})$ be the ciphertext that triggered the reject rule, so $\hat{y} = y$ and $\hat{h} = h$. $\mathcal{C}_{\mathrm{cr}}$ computes $\hat{r} \leftarrow f^{-1}(\hat{y})$ and $\hat{m} \leftarrow G(r) \oplus g$. Then it recalls that m_b was the message used in the challenge ciphertext, and then outputs

$$(\hat{r} \| m_b,\ \hat{r} \| \hat{m}), \quad \hat{h}$$

as its messages and hash for the collision resistance game.

We claim that $\mathcal{C}_{\mathrm{cr}}$ finds a collision with probability $\Pr[F_2]$. This is because if the event F_2 occured, then the ciphertext $\hat{C}$ must have been valid, i.e., $\mathsf{HVf}(K, \hat{r} \| (G(\hat{r}) \oplus g)) = 1$. By assumption $g \neq \hat{g}$, hence $\hat{m} = G(r) \oplus g$ is different from $m_b = G(r) \oplus \hat{g}$ but still verifies (prepended by $r = \hat{r}$) for the same hash value $h = \hat{h}$. This establishes (4.3.5).

Game 3. Game 3 is like Game 2, except now the decryption oracle rejects *all* ciphertexts $\hat{C} = (\hat{y}, \hat{g}, \hat{h})$ such that $\mathcal{A}$ did not query $G(\hat{r})$ such that $\hat{y} = f(\hat{r})$. (This amounts to adding a new rule to reject ciphertexts with $\hat{y} = y$, $\hat{h} \neq h$, and this property.) Let S_3 be the event that $\mathcal{A}$ wins Game 3.

Games 2 and 3 are clearly the same until the new rule is applied. Let F_3 be the event that a ciphertext is queried that triggers this reject rule in Game 3 but would not have been rejected in Game 2. Then we have

$$\big|\Pr[S_3] - \Pr[S_2]\big| \leq \Pr[F_3]. \tag{4.3.6}$$

We claim that

$$\Pr[F_3] \leq \operatorname{negl}(\lambda), \tag{4.3.7}$$

by the non-malleability condition on $\mathcal{H}$ in the theorem.

We prove this as follows. Using $\mathcal{A}$, we construct an adversary $\mathcal{B}_{\text{nm}}$ that wins the real non-malleability game with the prescribed parameters with probability related to $\Pr[F_3]$. We then show that any simulator can win the simulated game with only negligible probability. By the non-malleability condition, it follows that $\Pr[F_3]$ must be negligible.

We now describe $\mathcal{B}_{\text{nm}}$. In the first step of the non-malleability game it gets a hash key $K' = (K, f)$ as input. It simulates Game 3 for $\mathcal{A}$ using K and f in the public key. To simulate the random oracle G properly, $\mathcal{B}_{\text{nm}}$ stores all the random oracle queries $\mathcal{A}$ makes and the corresponding replies in an associative array, the G-list. $\mathcal{B}_{\text{nm}}$ answers decryption queries (y, g, h) as described in Game 2 by first locating a previously made query g in the G-list such that $y = f(r)$, and rejecting all other ciphertexts. $\mathcal{B}_{\text{nm}}$ never needs to compute f^{-1} to maintain this simulation.

$\mathcal{B}_{\text{nm}}$ continues the simulation until $\mathcal{A}$ issues its challenge query (m_0, m_1). Then $\mathcal{B}_{\text{nm}}$ chooses a random bit b and outputs the canonical encoding of the distribution $\mathcal{X}$ that selects $r \xleftarrow{\$} \{0,1\}^{\lambda/2}$ and outputs $r \| m_b$. This concludes the description of $\mathcal{B}_{\text{nm}}$ through the first two steps of the non-malleability game.

In the next step, $\mathcal{B}_{\text{nm}}$ receives h, σ from the challenger, where h was computed as $h \xleftarrow{\$} \mathsf{H}(K, x)$ and $x \xleftarrow{\$} \mathcal{X}$, and σ was computed as $y \leftarrow f(\operatorname{msb}(x))$. It returns to the simulation with $\mathcal{A}$. To compute the challenge ciphertext, $\mathcal{B}_{\text{nm}}$ sets g to a random string, defines $G(r) = g \oplus m_b$ and returns $c = (y, g, h)$ as the challenge ciphertext for $\mathcal{A}$. From this point, $\mathcal{B}_{\text{nm}}$ handles random oracles queries slightly differently. For each of $\mathcal{A}$'s queries $\hat{r}$ to the random oracle G, $\mathcal{B}_{\mathrm{nm}}$ checks if $f(\hat{r}) = y$, and if so, returns $G(r)$. $\mathcal{A}_y$ continues to answer all other queries as before, until $\mathcal{A}$ halts.

After $\mathcal{A}$ halts, $\mathcal{B}_{\mathrm{nm}}$ examines all of the rejected decryption queries $\hat{c} = (\hat{y}, \hat{g}, \hat{h})$ issued by $\mathcal{A}$ that had $\hat{y} = y$ and $\hat{g} \neq g$ (noting that queries $\hat{y} = y$, $\hat{g} = g$ and $\hat{h} \neq h$ are clearly invalid). Out of these queries, $\mathcal{B}_{\mathrm{nm}}$ selects one at random and returns $\hat{h}$ as its new hash in the non-malleability game.

In the final step of the non-malleability game, $\mathcal{B}_{\mathrm{nm}}$ gets $x = r \| m_b$. To compute x^*, it selects m^* at random, and outputs $x^* = r \| m^*$. This completes the description of the non-malleability adversary $\mathcal{B}_{\mathrm{nm}}$.

Now if the event F_3 occurred, then there is some ciphertext that would have been rejected in Game 3 but not Game 2, and $\mathcal{B}_{\mathrm{nm}}$ will have at least a $1/Q_{\mathrm{d}}$ chance at picking that ciphertext, where Q_{d} is the number of decryption queries issued by $\mathcal{A}$ in the simulated game. It is simple to check that if F_3 occurs and $\mathcal{B}_{\mathrm{nm}}$ selects such a ciphertext, then it wins the non-malleability game. This gives $\mathcal{B}_{\mathrm{nm}}$ a $\Pr[F_3]/Q_{\mathrm{d}}$ chance at winning.

We now analyze the performance of any simulator $\mathcal{S}$. $\mathcal{S}$ receives $K' = (K, f)$ as input, chooses a distribution $\mathcal{X}$, and gets only $\sigma = f(r)$ in the third step of the simulated game. In particular, it cannot predict the leading $\lambda/2$ bits of r because by the partial one-wayness of $\mathcal{F}$. Therefore, the probability of the simulator satisfying the relation R_{br} is negligible. But then we have that $\Pr[F_3]/Q_{\mathrm{d}}$ is also negligible, which establishes (4.3.7).

Game 4. Game 4 is like Game 3, except that now the challenge encryption query ignores the adversary's messages and encrypts a random message.

Let S_4 be the event that it wins Game 4. We claim that

$$|\Pr[S_3] - \Pr[S_4]| \leq \mathrm{negl}(\lambda), \tag{4.3.8}$$

by the perfect one-wayness condition in the theorem. To prove this, we construct an adversary $\mathcal{B}_{\text{pow}}$ that wins the POW game with probability $\varepsilon = |\Pr[S_3] - |\Pr[S_4]|$. $\mathcal{B}_{\text{pow}}$ gets a hash key $K' = (K, f)$ as input, and runs Game 4 while using (K, f) in the public key. $\mathcal{B}_{\text{pow}}$ answers decryption queries itself, as described in Game 3, without f^{-1}. When $\mathcal{A}$ issues its challenge query (m_0, m_1), $\mathcal{B}_{\text{pow}}$ selects a random bit b and outputs a canonical distribution $\mathcal{X}$ that samples $r \xleftarrow{\$} \{0,1\}^{\lambda/2}$ and a random m' from the message space, and outputs $r\|m_b$ with probability 1/2 and $r\|m'$ otherwise.

In the third step of the perfect one-wayness game, $\mathcal{B}_{\text{pow}}$ gets as input h and side information $y = f(r)$. $\mathcal{B}_{\text{pow}}$ first chooses a random string g, which implicitly defines $G(r) = g \oplus m$, where m is either m_b or a random message chosen by the game. $\mathcal{B}_{\text{pow}}$ continues to simulate the game for $\mathcal{A}$ with challenge ciphertext $C = (y, g, h)$ until either (1) $\mathcal{A}$ queries $r = f^{-1}(y)$ to the random oracle, or (2) $\mathcal{A}$ halts with a bit d as output.

In case (1), $\mathcal{B}_{\text{pow}}$ aborts the simulation and can immediately win the game (with overwhelming probability) by checking if $\mathsf{HVf}(K, r\|m_b, \hat{h}) = 1$. In case (2), $\mathcal{B}_{\text{pow}}$ tests if $\hat{b} = b$, and if so, it outputs 1, and otherwise it outputs 0. If the hash oracle returned $\hat{h} = \mathsf{G}(K, r\|m_b)$, then $\mathcal{B}_{\text{pow}}$ perfectly simulated Game 3 for the adversary. Otherwise, $\mathcal{B}_{\text{pow}}$ perfectly simulated Game 4. A standard argument gives that $\mathcal{B}_{\text{pow}}$ has advantage at least $\Pr[F_4]/4$, finishing the proof of the claim.

Finally, it is obvious that

$$\Pr[S_4] = 1/2 \tag{4.3.9}$$

because the bit b is never used in Game 4. Collecting (4.3.1)-(4.3.9) we get that $\mathcal{A}$ must have had negligible advantage in the original CCA game.

CHAPTER V

CIRCULAR SECURITY

In this chapter we give two complementary sets of results concerning circular security. First, we show how to construct very natural and efficient circular-secure encryption schemes using hard learning problems. In the second part, we prove that our schemes are *non-trivial*, in that not all semantically-secure encryption schemes are necessarily also circular secure. We prove this by giving a counterexample that is provably secure under a Diffie-Hellman-like assumption, but is easily breakable when a circular encryption of size 2 is published.

5.1 Preliminaries

For our learning-based schemes, we will denote the security parameter by n instead of λ. This change is due to prior work in lattice-based cryptography, where the n (which will be identified below in the definition of the LWE problem) is the most natural parameter to adjust for hardness.

Let Ber_ε denote the Bernoulli distribution over $\{0,1\}$ that is 1 with probability ε and 0 with probability $1-\varepsilon$.

5.1.1 Key-Dependent Message Security

We follow the presentation of Boneh *et al.* [22], which generalizes the definition of Black *et al.* [14]. In this definition, an adversary interacts with a challenger that answers encryption queries for functions of the users' secret keys. The adversary is restricted to functions from a certain family, which we will denote $\mathcal{F} \subset \{f \mid f : \mathsf{K}^\ell \to \mathsf{M}\}$. Strictly speaking, $\mathcal{F}$ is a family of sets of functions parameterized by the security parameter n and the number of users ℓ.

Definition 6. *Let* $(\mathsf{G}, \mathsf{E}, \mathsf{D})$ *be a public-key encryption scheme, and let* $\mathcal{A}$ *be an adversary. The game proceeds as follows:*

1. *The challenger chooses a bit* $b \leftarrow \{0,1\}$. *It also samples* $(pk_i, sk_i) \xleftarrow{\$} \mathsf{G}(1^n)$ *for* $i = 1, \ldots, \ell$. *It gives* $pk_1, \ldots, pk_\ell$ *to* $\mathcal{A}$.

2. $\mathcal{A}$ *makes encryption queries of the form* (j, f), *where* $j \in [\ell]$ *and* $f \in \mathcal{F}$. *To process a query, if* $b = 0$, *the challenger computes* $m \leftarrow f(sk_1, \ldots, sk_\ell)$ *and* $c \xleftarrow{\$} \mathsf{E}_{pk_j}(m)$. *If* $b = 1$ *it instead sets* $c \xleftarrow{\$} \mathsf{E}_{pk_j}(0^{|m|})$. *It returns* c *to* $\mathcal{A}$.

3. $\mathcal{A}$ *ends the game and outputs* $\hat{b} \in \{0,1\}$.

We define the advantage of $\mathcal{A}$ with respect to $\mathcal{F}$, *denoted* $\mathsf{AdvKDM}_{\mathcal{A},\mathsf{PKE},\mathcal{F}}(\lambda)$, *to be* $|\Pr[b = \hat{b}] - 1/2|$. *If* $\mathsf{AdvKDM}_{\mathcal{A},\mathsf{PKE},\mathcal{F}}(\lambda)$ *is negligible for all probabilistic polynomial-time adversaries* $\mathcal{A}$, *then we say that* PKE *is* KDM-secure with respect to $\mathcal{F}$.

For a symmetric-key encryption scheme SKE, *we define all these terms in exactly the same way, except the adversary is not given the public keys, and encryption naturally uses the secret key to encrypt.*

If all constant functions (that is, functions f_m such that $f_m(sk_1, \ldots, sk_\ell) = m$ for some message m) are contained in $\mathcal{F}$, then security with respect to $\mathcal{F}$ can be shown to imply standard CPA security. If the projection functions, f_j such that $f_j(sk_1, \ldots, sk_\ell) = sk_j$ for some j are also contained in $\mathcal{F}$, then security with respect to $\mathcal{F}$ implies circular security.

We remark that these definitions can be extended to allow the adversary to issue decryption queries [24]. We do not pursue this direction here.

5.1.2 Noisy Learning Problems

We recall the learning with error (LWE), due to Regev [69], with the learning parity with noise (LPN) as a special case.

For positive integers n and $q \geq 2$, a vector $\mathbf{s} \in \mathbb{Z}_q^n$, and a probability distribution χ on $\mathbb{Z}_q$, define $A_{\mathbf{s},\chi}$ to be the distribution over $\mathbb{Z}_q^n \times \mathbb{Z}_q$ induced by choosing a vector $\mathbf{a} \in \mathbb{Z}_q^n$ uniformly at random, an error term $x \overset{\$}{\leftarrow} \chi$, and outputting $(\mathbf{a}, \langle \mathbf{a}, \mathbf{s} \rangle + x)$.

Definition 7. *For an integer function $q = q(n)$ and an error distribution χ on $\mathbb{Z}_q$, the* learning with errors *problem $\mathsf{LWE}_{q,\chi}$ in n dimensions is to find $\mathbf{s} \in \mathbb{Z}_q^n$ (with non-negligible probability) given oracle access to the distribution $A_{\mathbf{s},\chi}$ for uniformly random $\mathbf{s}$.*

The learning parity with noise *problem LPN_ε is the special case of $\mathsf{LWE}_{q,\chi}$ for $q = 2$ and $\chi = \mathsf{Ber}_\varepsilon$.*

Using standard self-reduction and amplification techniques, an algorithm that solves the LWE problem with non-negligible probability for uniform $\mathbf{s}$ may be efficiently transformed into one that solves LWE with overwhelming probability for arbitrary $\mathbf{s}$.

We will be interested in error distributions χ over $\mathbb{Z}_q$ that are derived from Gaussians. For any $r > 0$, define the one-dimensional Gaussian probability distribution by its density function $D_r(x) = \exp(-\pi(x/r)^2)/r$. For $\alpha > 0$, define $\bar{\Psi}_\alpha$ to be the distribution on $\mathbb{Z}_q$ obtained by drawing $y \leftarrow D_\alpha$ and outputting $\lfloor q \cdot y \rceil \bmod q$.

Regev [69] demonstrated strong evidence for the hardness of the LWE problem (with Gaussian error distribution), by giving a *quantum* reduction from approximating well-studied lattice problems to within $\tilde{O}(n/\alpha)$ factors in the *worst case* to solving $\mathsf{LWE}_{q,\bar{\Psi}_\alpha}$, when $\alpha \cdot q \geq 2\sqrt{n}$. Recently, Peikert [65] gave a related *classical* reduction for similar parameters.

Regev showed [69] that, when $q = \text{poly}(n)$ is *prime* and χ is a Gaussian error distribution, the distribution $A_{\mathbf{s},\chi}$ is pseudorandom, assuming only that the search version of LWE is hard. Here we extend the argument to show that for *prime power* moduli and the same error distribution, the LWE distribution $A_{\mathbf{s},\chi}$ (for uniform $\mathbf{s}$) is also pseudorandom (i.e., computationally indistinguishable from uniform), assuming that the LWE problem is hard.

Lemma 12. *Let $q = p^e$ be a prime power with $p = \mathrm{poly}(n)$, and let χ be a distribution over $\mathbb{Z}_q$ that produces an element in $\{-\frac{p-1}{2}, \ldots, \frac{p-1}{2}\} \subset \mathbb{Z}_q$ with overwhelming probability. There is a probabilistic polynomial-time reduction from $\mathsf{LWE}_{q,\chi}$ to distinguishing between $A_{\mathbf{s},\chi}$ for uniformly random $\mathbf{s} \in \mathbb{Z}_q^n$ and the uniform distribution on $\mathbb{Z}_q^n \times \mathbb{Z}_q$).*

Proof. The proof is a simple extension of prior ones for prime moduli (see, e.g., [69, Lemma 4.2]), therefore we emphasize only the new elements. The idea is to use a distinguisher to recover the least significant base-p digits of $\mathbf{s}$, at which point the distribution is "narrow" enough to solve for the remainder of $\mathbf{s}$ via rounding and standard linear algebra.

For $i = 0, \ldots, e$, define the hybrid distribution $A^i_{\mathbf{s},\chi}$ that is obtained by drawing a sample $(\mathbf{a}, b)$ from $A_{\mathbf{s},\chi}$ and outputting $(\mathbf{a}, b + p^i \cdot r) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$ for a uniformly random $r \in \mathbb{Z}_q$ (freshly chosen for each sample). Note that $A^e_{\mathbf{s},\chi}$ is identical to $A_{\mathbf{s},\chi}$, while $A^0_{\mathbf{s},\chi} = U$. Therefore, an algorithm D that distinguishes between $A_{\mathbf{s},\chi}$ and U with non-negligible advantage must distinguish between $A^{j-1}_{\mathbf{s},\chi}$ and $A^j_{\mathbf{s},\chi}$ with non-negligible advantage, for some $j \in \{1, \ldots, e\}$. By standard self-reduction and amplification techniques, we can assume that D accepts with *overwhelming* probability given $A^{j-1}_{\mathbf{s},\chi}$ for *any* $\mathbf{s}$, and rejects given $A^j_{\mathbf{s},\chi}$ (with overwhelming probability, for any $\mathbf{s}$).

Below we show how to use D to find $\mathbf{s}' = \mathbf{s} \bmod p$, i.e., the least significant digit (in base p) of each entry of $\mathbf{s}$. Having done so, we can then transform $A_{\mathbf{s},\chi}$ into $A_{p\cdot\mathbf{t},\chi}$, where $p \cdot \mathbf{t} = \mathbf{s} - \mathbf{s}'$. A sample from the latter distribution is of the form $(\mathbf{a}, b = p \cdot \langle \mathbf{a}, \mathbf{t} \rangle + x)$ for $x \leftarrow \chi$; because $x \in \{-\frac{p-1}{2}, \ldots, \frac{p-1}{2}\}$ with overwhelming probability, we may round b to the nearest multiple of p and learn the value of $\langle \mathbf{a}, \mathbf{t} \rangle \bmod p$. By drawing n samples, we may then solve for $\mathbf{t}$ by linear algebra.

We find the entries of $\mathbf{s}'$ one by one in the following way. To test whether entry $s_i' = 0 \in \mathbb{Z}_p$, apply a transformation to $A_{\mathbf{s},\chi}$ mapping $(\mathbf{a}, b)$ to $(\mathbf{a} - l \cdot p^{j-1} \cdot \mathbf{e}_i, b + p^j \cdot r)$ for uniform and independent $l, r \in \mathbb{Z}_p$ (where $\mathbf{e}_i$ is the ith standard basis vector). Because p is prime, it may be verified that if $s_i' = 0$, then the resulting distribution is exactly $A^j_{\mathbf{s},\chi}$, otherwise it is $A^{j-1}_{\mathbf{s},\chi}$, therefore D reveals which is the case. Testing whether $\mathbf{s}_i' = k$ for each $k \in \mathbb{Z}_p$ is done similarly; because $p = \text{poly}(n)$, the reduction is polynomial-time. □

5.2 Public-Key Encryption Scheme Based on LWE

In this section we design a public-key encryption scheme based on the $\mathsf{LWE}_{q,\chi}$ problem, where as usual, the error distribution $\chi = \bar{\Psi}_\alpha$ for some $\alpha = \alpha(n) \in (0, 1)$, chosen with the modulus q to satisfy various constraints.

Our techniques. Our LWE-based construction involves a few techniques that may be of independent interest and application.

In the LWE problem and the encryption schemes of [69, 66], the secret key is a vector $\mathbf{s} \in \mathbb{Z}_q^n$ chosen uniformly at random (where the modulus q is related to the security parameter n), while the message space is $\mathbb{Z}_p$ for some $p \ll q$. An important idea in the work of Boneh *et al.* [22] is the ability to generate, given only a public key, a ciphertext that decrypts to a message related to $\mathbf{s}$. Because decryption in the LWE-based schemes of [69, 66] is essentially a linear operation, it is possible to generate ciphertexts that are somehow related to $\mathbf{s}$. However, because the entries of $\mathbf{s}$ are taken modulo $q \gg p$, it is unclear how to "fit" the entries into the message space.

In our scheme, we address this issue by instead drawing the entries of the secret key $\mathbf{s}$ from the very same (Gaussian) *error distribution* as in the underlying LWE problem. For a sufficiently "narrow" error distribution, each entry of $\mathbf{s}$ can take on at most p different values (with overwhelming probability), allowing the entire entry to fit unambiguously into the message space. Moreover, this change *does not affect the* *hardness of the* LWE *problem*: we show a simple, tight reduction from the standard LWE problem to the variant just described. Abstractly, the reduction may be viewed as putting the LWE distribution into *Hermite normal form* (HNF); interestingly, the HNF was also used by Micciancio [58] and Micciancio and Regev [60] as a way to improve the *efficiency* of lattice-based encryption schemes.

The second important technique relates to the faithful simulation of key-dependent messages. In Regev's scheme, for instance, ciphertexts are produced from two components: a uniformly random "syndrome" $\mathbf{u} \in \mathbb{Z}_q^n$, and a pseudorandom "mask" $v \in \mathbb{Z}_q$ (for hiding the message) that is concentrated around the inner product $\langle \mathbf{u}, \mathbf{s} \rangle \in \mathbb{Z}_q$. However, the exact distribution of $d = v - \langle \mathbf{u}, \mathbf{s} \rangle \in \mathbb{Z}_q$ is difficult to characterize, and may differ for each syndrome $\mathbf{u}$. We modify the encryption algorithm so that d has the same "nice" (Gaussian) distribution for *every* $\mathbf{u}$, which ensures that our simulated key-dependent ciphertexts are properly distributed. This step applies analytical techniques used in Regev's reduction from worst-case lattice problems to LWE [69, Section 3.2.1]; we also generalize the techniques to deal with the amortized encryption scheme of Peikert *et al.* [66], where ciphertexts contain several masks v_i for a single syndrome $\mathbf{u}$.

A final interesting technique concerns key cycles or cliques, where every user's secret key may be encrypted under every user's public key. Simulating such a scenario seems to require knowing a relation between every pair of (unknown and independent) secret keys. In the work of Boneh *et al.* [22], the simulator creates the relations itself by "masking" one unknown (binary) secret key with random strings, and generating the related public keys via homomorphisms. In our setting with Gaussian secret keys, it is unclear how to generate a properly distributed public key by manipulating a given one. Fortunately, the HNF-style LWE transformation described above also happens to produce the required linear relations as a side-effect! Briefly, this is because the transformation may be applied several times to the LWE source distribution, and each resulting Gaussian secret is linked to the original secret $\mathbf{s}$ via a known linear relation.

5.2.1 A Generic Transformation

We start with a useful transformation that reduces the LWE problem to one in which the *secret itself* is chosen from the error distribution χ, essentially putting the LWE distribution into "Hermite normal form." The transformation applies for any q and χ.

Lemma 13. *There is a deterministic polynomial-time transformation T that, for arbitrary $\mathbf{s} \in \mathbb{Z}_q^n$ and error distribution χ, maps $A_{\mathbf{s},\chi}$ to $A_{\bar{\mathbf{x}},\chi}$ where $\bar{\mathbf{x}} \leftarrow \chi^n$, and maps uniform distribution on $\mathbb{Z}_q^n \times \mathbb{Z}_q$ to itself. The transformation also produces an invertible square matrix $\bar{\mathbf{A}} \in \mathbb{Z}_q^{n\times n}$ and $\bar{\mathbf{b}} \in \mathbb{Z}_q^n$ that, when mapping $A_{\mathbf{s},\chi}$ to $A_{\bar{\mathbf{x}},\chi}$, satisfy $\bar{\mathbf{x}} = -\bar{\mathbf{A}}^T\mathbf{s} + \bar{\mathbf{b}}$.*

Proof. The transformation T is given access to some distribution D over $\mathbb{Z}_q^n \times \mathbb{Z}_q$ (where D may be either $A_{\mathbf{s},\chi}$ or uniform over $\mathbb{Z}_q^n \times \mathbb{Z}_q$), and proceeds in two stages.

In the first stage, T performs some initial processing to obtain $\bar{\mathbf{A}}, \bar{\mathbf{b}}$. It does this by drawing several pairs $(\mathbf{a}, b)$ from D, and keeping certain pairs until it has accumulated a set of n samples $\{(\bar{\mathbf{a}}_i, \bar{b}_i)\}$ that will make up $\bar{\mathbf{A}}, \bar{\mathbf{b}}$ in the natural way. With each new sample $(\mathbf{a}, b)$, T checks whether $\mathbf{a}$ is linearly independent modulo q of all those $\bar{\mathbf{a}}_i$ that have been kept so far; if so, $(\mathbf{a}, b)$ is kept, otherwise it is discarded. Note that the probability of keeping a particular sample is at least $\varphi(q)/q \geq 1/2$ (where φ denotes the Euler totient function), so with high probability, T accumulates the required n samples after drawing $O(n^2)$ samples from D. Now by construction, $\bar{\mathbf{A}}$ is invertible modulo q. Also observe that each sample is kept or discarded based only on its $\mathbf{a}$ component, so when $D = A_{\mathbf{s},\chi}$, we have $\bar{\mathbf{b}} = \bar{\mathbf{A}}^T\mathbf{s} + \bar{\mathbf{x}}$ where $\bar{\mathbf{x}}$ is drawn from χ^n.

The second stage actually transforms (fresh) samples from D into samples from a possibly different distribution. Given a draw $(\mathbf{a}, b) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$ from D, T outputs

$(\mathbf{a}', b') \in \mathbb{Z}_q^n \times \mathbb{Z}_q$, where

$$\mathbf{a}' = -\bar{\mathbf{A}}^{-1}\mathbf{a} \quad \text{and} \quad b' = b + \langle \mathbf{a}', \bar{\mathbf{b}} \rangle.$$

Observe that because $\bar{\mathbf{A}}$ is invertible modulo q and $\mathbf{a} \in \mathbb{Z}_q^n$ is uniform, $\mathbf{a}' \in \mathbb{Z}_q^n$ is uniform as well. We now consider the two cases for D. If $D = U$, then $(\mathbf{a}', b')$ is also distributed according to U, because $b \in \mathbb{Z}_q$ is uniform and independent of $\mathbf{a}$. If $D = A_{\mathbf{s},\chi}$, then $b = \langle \mathbf{a}, \mathbf{s} \rangle + x$ for some $x \leftarrow \chi$, so we have

$$b' = \langle \mathbf{a}, \mathbf{s} \rangle + x - \langle \bar{\mathbf{A}}^{-1}\mathbf{a}, \bar{\mathbf{A}}^T\mathbf{s} \rangle + \langle \mathbf{a}', \bar{\mathbf{x}} \rangle = \langle \mathbf{a}', \bar{\mathbf{x}} \rangle + x.$$

Therefore, $(\mathbf{a}', b')$ is distributed according to $A_{\bar{\mathbf{x}},\chi}$, as desired. □

5.2.2 Public-Key Scheme

We now define a KDM-secure encryption scheme based on the LWE problem. For technical reasons, our construction uses a *prime power* modulus $q = p^2$ of a certain size, with messages taken over $\mathbb{Z}_p$. (Other choices of $q = p^e$ are possible, but $q = p^2$ seems to correspond to the mildest underlying assumption.) Note that any element $v \in \mathbb{Z}_q$ may be written as $v = (v_1, v_0) \in \mathbb{Z}_p \times \mathbb{Z}_p$, where v_1 and v_0 are the most and least significant digits in the base-p representation of v, respectively, with the digits chosen from the set of residues $\{-\frac{p-1}{2}, \dots, \frac{p-1}{2}\}$. Recall that by Lemma 12, the LWE distribution $A_{\mathbf{s},\chi}$ (for uniform $\mathbf{s} \in \mathbb{Z}_q^n$) remains pseudorandom if the search problem $\mathsf{LWE}_{q,\chi}$ is hard, and if the error distribution χ is concentrated on $\{0\} \times \mathbb{Z}_p$ (which will be the case in our system, by explicit design).

For simplicity, we start with a scheme that encrypts a single element of $\mathbb{Z}_p$ at a time, later extending it to an amortized version in Section 5.2.4. Our scheme is similar to the construction of Regev [69], with two main differences. First, the entries of the secret key $\mathbf{s} \in \mathbb{Z}_q^n$ are chosen from the (narrow) *error distribution*, so that they may be represented unambiguously as elements of the message space $\mathbb{Z}_p$ (see Lemma 18); this is safe due to Lemma 13. Second, we modify the encryption algorithm so that it induces a “nice” distribution over ciphertexts (see Lemma 19).

The scheme involves a number of parameters. The main parameters are the prime p and error parameter α, which we instantiate below. Let $\chi = \bar{\Psi}_\alpha$.

- *Key generation:* The secret key is $\mathbf{s} \leftarrow \chi^n$. The public key is $(\mathbf{A}, \mathbf{b}) \in \mathbb{Z}_q^{n \times m} \times \mathbb{Z}_q^m$, which is made up of $m \geq 2(n+1)\lg q$ draws $(\mathbf{a}_i, b_i)$ from $A_{\mathbf{s},\chi}$. I.e., $\mathbf{b} = \mathbf{A}^T\mathbf{s} + \mathbf{x}$ for independent $\mathbf{A} \overset{\$}{\leftarrow} \mathbb{Z}_q^{n \times m}$ and $\mathbf{x} \overset{\$}{\leftarrow} \chi^m$.

- *Encryption:* Before specifying the encryption algorithm, we define a distribution $E_{\mathbf{A},\mathbf{b}}$ over $\mathbb{Z}_q^n \times \mathbb{Z}_q$, which has parameters $r = \omega(\sqrt{\log m})$ and $r' = r \cdot \sqrt{m} \cdot (\alpha + \frac{1}{2q})$. The distribution is obtained by choosing $\mathbf{r} \overset{\$}{\leftarrow} D_{\mathbb{Z}^m, r}$ and $e \overset{\$}{\leftarrow} \bar{\Psi}_{r'}$ and outputting

  $$(\mathbf{A}\mathbf{r}, \langle \mathbf{r}, \mathbf{b} \rangle + e) \in \mathbb{Z}_q^n \times \mathbb{Z}_q.$$

  To encrypt a message $z \in \mathbb{Z}_p$ given the public key $(\mathbf{A}, \mathbf{b})$, draw a sample $(\mathbf{u}, v)$ from $E_{\mathbf{A},\mathbf{b}}$ and output the ciphertext $(\mathbf{u}, c = v + z \cdot p) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$.

- *Decryption:* To decrypt a ciphertext $(\mathbf{u}, c)$ given the secret key $\mathbf{s}$, output the $z \in \mathbb{Z}_p$ such that $z \cdot p$ is closest to $c - \langle \mathbf{u}, \mathbf{s} \rangle$ modulo q.

The main conflicting constraints on the parameters are given by the correctness requirement (α cannot be too large) and the hardness requirement (α should be large enough to invoke the worst-case lattice connections of [69, 65]). These constraints are captured by the following inequalities to be satisfied:

$$\frac{2\sqrt{n}}{q} = \frac{2\sqrt{n}}{p^2} \leq \alpha \leq \frac{1}{p \cdot \sqrt{m} \cdot \omega(\log n)} \tag{5.2.1}$$

It is possible to satisfy these constraints simultaneously for $m = O(n \log n)$, $p = \tilde{O}(\sqrt{mn})$, and $\alpha = 1/\tilde{O}(m \cdot \sqrt{n})$. This yields an underlying worst-case approximation factor of $\tilde{O}(n/\alpha) = \tilde{O}(n^{2.5})$ for lattice problems such as GapSVP.

We fix some notation used in the theorem below. For $i \in [\ell]$, $\mathbf{t} \in \mathbb{Z}_p^n$, and $w \in \mathbb{Z}_p$, define the function

$$\begin{aligned} f_{i,\mathbf{t},w} : \quad & (\mathbb{Z}_p^n)^\ell \to \mathbb{Z}_p \\ & (\mathbf{s}_1, \ldots, \mathbf{s}_\ell) \mapsto \langle \mathbf{t}, \mathbf{s}_i \rangle + w \end{aligned} .$$

Let $\mathcal{F}_{\text{aff}} = \{f_{i,\mathbf{t},w} : i \in [\ell], \mathbf{t} \in \mathbb{Z}_p^m, w \in \mathbb{Z}_p\}$.

Theorem 14. *Let n, m, p, q, α satisfy (5.2.1). Then the above scheme is KDM-secure with respect to $\mathcal{F}_{\text{aff}}$, assuming that $\mathsf{LWE}_{q,\chi}$ is hard.*

We remark that one can extend the message space of this scheme from $\mathbb{Z}_p$ to $\mathbb{Z}_p^n$, without hurting KDM security, by simply concatenating ciphertexts. This gives security against a suitably extended family of affine functions, which implies circular security.

5.2.3 Proof of Security

5.2.3.1 Overview

The proof of Theorem 14 has the following structure. First we show completeness, including correct decryption of key-dependent messages. Next we prove KDM security in two main steps.

The first step is to show that the view of the adversary in the real attack game may be generated faithfully, up to negligible statistical distance, via an alternate game: starting from the distribution $A_{\mathbf{s},\chi}$ (for uniformly random $\mathbf{s} \in \mathbb{Z}_q^n$), the game invokes the transformation from Lemma 13 several times to produce independent distributions $A_{\mathbf{s}_1,\chi}, A_{\mathbf{s}_2,\chi}, \ldots$ for each user (where each $\mathbf{s}_i \leftarrow \chi^n$), then generates the public keys from these distributions in the natural way. The transformation additionally outputs an invertible linear relation modulo q (hence modulo p as well) between each $\mathbf{s}_i$ and $\mathbf{s}$, thus linking every pair $\mathbf{s}_i, \mathbf{s}_j$ in a known way. The game answers the adversary's (key-dependent) message queries using these relations and the linear homomorphisms of the scheme (this is where we use the fact that the system has a "nice" ciphertext distribution). The crucial property of this game is that, aside from oracle access to $A_{\mathbf{s},\chi}$, the game works without needing to know any of the secret vectors $\mathbf{s}, \mathbf{s}_1, \mathbf{s}_2, \ldots$.

The second (and final) step is to consider a game that proceeds in exactly the same way as above, except that the original distribution $A_{\mathbf{s},\chi}$ is replaced by the *uniform* distribution over $\mathbb{Z}_q^n \times \mathbb{Z}_q$. Because the game uses only oracle access to its given distribution, the two games are computationally indistinguishable under the LWE assumption. Moreover, all the public keys in this game are uniform and independent, which implies that all the simulated ciphertexts are as well (up to negligible statistical distance). It follows that the adversary has negligible advantage in this game, and the scheme is secure.

5.2.3.2 Technical Lemmas

We will need the following (now standard) technical lemmas, which were proved in prior works, below. We will refer to the following standard tail bound: a Gaussian random variable with variance σ is within distance $t \cdot \sigma$ of its mean, except with negligible (in t) probability.

Below we will refer to a particular lattice defined by a matrix $\mathbf{A}$, denoted $\Lambda^{\perp}(\mathbf{A})$, which was defined by Ajtai [4]. We do not need the exact definition of $\Lambda^{\perp}(\mathbf{A})$ in our constructions or analysis, but the machinery we use will refer to it. We will also make similar use of the *smoothing parameter* of a lattice, denoted $\eta_\epsilon(\Lambda)$ [59] for some $\epsilon > 0$.

The following lemma [44, Corollary 5.4], shows that the distribution of the vector $\mathbf{u}$ in our scheme's ciphertexts is statistically close uniformly.

Lemma 15. *Let n and q be integers and let $m \geq 2n \lg q$. Then for all but a negligible (in n) fraction of $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, and for any $r \geq \omega(\sqrt{\log m})$, the distribution of $\mathbf{u} = \mathbf{Ar}$ is within negligible statistical distance of uniform over $\mathbb{Z}_q^n$, where $\mathbf{r} \overset{\$}{\leftarrow} D_{\mathbb{Z}^m,r}$.*

We also need the following lemma, which is a special case of [69, Corollary 3.10]. We note the original lemma actually dealt with e drawn from a continuous Gaussian, not its discretized version. But since we are adding e to a value that is always integral, rounding e first is equivalent to rounding the resulting sum.

Lemma 16. *Let $\Lambda \subset \mathbb{R}^m$ be a lattice. Let $\mathbf{x} \in \mathbb{Z}^m$ be a vector and $r, r' > 0$ be real. Assume that $\left(1/r^2 + (\|\mathbf{x}\|/r')^2\right)^{-1/2} \geq \eta_\epsilon(\Lambda)$ for some negligible $\epsilon = \epsilon(m)$. Then there* *exists $\beta \in \mathbb{R}$, $\sqrt{2}r' \geq \beta > 0$, such that the distribution of $\langle \mathbf{r}, \mathbf{x} \rangle + e$, where $\mathbf{r} \overset{\$}{\leftarrow} D_{\Lambda,r}$ and $e \overset{\$}{\leftarrow} \bar{\Psi}_{r'}$, is within negligible statistical distance of $\bar{\Psi}_{\beta}$.*

The following lemma [44, Lemma 5.2] allows us to argue about the conditional distribution of the randomness $\mathbf{r}$ used in encryption, given the first value $\mathbf{u}$ in the ciphertext.

Lemma 17. *Let $\mathbf{u} \in \mathbb{Z}_q^n$ and $\mathbf{t} \in \mathbb{Z}^m$ be an arbitrary solution to $\mathbf{At} = \mathbf{u} \mod q$. Then there exists a lattice $\Lambda^{\perp} = \Lambda^{\perp}(\mathbf{A})$ such that for any $r \geq \eta_{\epsilon}(\Lambda^{\perp})$, the conditional distribution of $\mathbf{r} \overset{\$}{\leftarrow} D_{\mathbb{Z}^m,r}$, given $\mathbf{Ar} = \mathbf{u}$ is exactly $\mathbf{t} + D_{\Lambda^{\perp},r,-\mathbf{t}}$. Moreover, with overwhelming probability over the choice of $\mathbf{A}$, the lattice $\Lambda^{\perp}$ satisfies $\eta_{\epsilon}(\Lambda^{\perp}) \leq \omega(\sqrt{\log m})$ for some negligible $\epsilon(m)$.*

5.2.3.3 Abstract Properties

Here we derive a few technical facts about the distributions of keys and ciphertexts in the public-key scheme. The proof of security relies only on these abstract properties, which will be shown via the technical lemmas above.

The first fact is that the entries of the secret key may be represented unambiguously in the message space $\mathbb{Z}_p$. For convenience in dealing with key-dependent messages, from now on we view the secret key $\mathbf{s}$ as an element of $\mathbb{Z}_p^n \subset \mathbb{Z}_q^n$.

Lemma 18. *An $s \leftarrow \chi$ is of the form $s = (0, s_0) \in \mathbb{Z}_p \times \mathbb{Z}_p$ with overwhelming probability.*

Proof. This follows directly from the upper bound on α from Equation Equation (5.2.1) and the exponential tail bound on the Gaussian distribution. □

The following lemmas characterize the ciphertext distribution, which is needed for showing correctness, and (more importantly) for producing proper key-dependent ciphertexts using the scheme's homomorphisms.

Lemma 19. *With overwhelming probability over the choice of the secret key* $\mathbf{s}$ *and corresponding public key* $(\mathbf{A}, \mathbf{b})$*, the distribution* $E_{\mathbf{A},\mathbf{b}}$ *is within negligible statistical distance of* $A_{\mathbf{s},\bar{\Psi}_\beta}$ *for some* $\beta \leq \sqrt{2}r'$*.*

Proof. In this proof we omit routine calculations with parameters, which can easily be checked. For $(\mathbf{u}, v)$ drawn from $E_{\mathbf{A},\mathbf{b}}$ where $\mathbf{b} = \mathbf{A}^T\mathbf{s} + \mathbf{x}$, we have

$$(\mathbf{u}, v) = (\mathbf{u}, \langle \mathbf{r}, \mathbf{b} \rangle + e) = (\mathbf{u}, \mathbf{r}^T\mathbf{A}^T\mathbf{s} + \langle \mathbf{r}, \mathbf{x} \rangle + e) = (\mathbf{u}, \langle \mathbf{u}, \mathbf{s} \rangle + \langle \mathbf{r}, \mathbf{x} \rangle + e).$$

First, by Lemma 15 we have that the marginal distribution of $\mathbf{u}$ induced by $E_{\mathbf{A},\mathbf{b}}$ is within negligible statistical distance of uniform, with overwhelming probability over the choice of $\mathbf{A}$. Next, fix an arbitrary $\mathbf{u} \in \mathbb{Z}_q^n$ and consider the distribution of $\mathbf{r}$ conditioned on the event that $\mathbf{Ar} = \mathbf{u}$. By Lemma 17, this conditional distribution is a discrete Gaussian over (a coset of) a particular lattice $\Lambda^\perp$ with the same parameter r. Finally, for fixed $\mathbf{x}$, we can apply Lemma 16 with $\Lambda^\perp$ and conclude that the distribution of $\langle \mathbf{r}, \mathbf{x} \rangle + e$ is within negligible statistical distance of $\bar{\Psi}_\beta$. We note that $\mathbf{x}$, drawn as specified in our scheme, will satisfy the condition in Lemma 16 with overwhelming probability. This concludes the proof. □

Lemma 20. *Let* $\mathbf{t} \in \mathbb{Z}_p^n$ *and* $y \in \mathbb{Z}_p$ *be arbitrary. With overwhelming probability over the choice of the public key* $(\mathbf{A}, \mathbf{b})$ *for arbitrary secret key* $\mathbf{s} \in \mathbb{Z}_p^n$*, the following holds: for* $(\mathbf{u}, v) \xleftarrow{\$} E_{\mathbf{A},\mathbf{b}}$*, the distribution of*

$$(\mathbf{u} - \mathbf{t} \cdot p, v + w \cdot p) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$$

is within negligible statistical distance of a (properly generated) encryption of the message $\langle \mathbf{t}, \mathbf{s} \rangle + w \in \mathbb{Z}_p$*.*

Proof. Consider the distribution of an encryption of $\langle \mathbf{t}, \mathbf{s} \rangle + w$ for fixed $(\mathbf{A}, \mathbf{b})$. By Lemma 19 above, the ciphertext is distributed (up to negligible statistical distance) as

$$(\mathbf{u}', \langle \mathbf{u}', \mathbf{s} \rangle + x + (\langle \mathbf{t}, \mathbf{s} \rangle + w) \cdot p) = (\mathbf{u}', \langle \mathbf{u}' + p \cdot \mathbf{t}, \mathbf{s} \rangle + x + w \cdot p),$$

where $\mathbf{u}' \in \mathbb{Z}_q^n$ is uniform and $x \leftarrow \bar{\Psi}_\beta$ for some fixed $\beta \leq \sqrt{2}r'$. Rewriting $\mathbf{u}' = \mathbf{u} - p \cdot \mathbf{t}$ for uniformly random $\mathbf{u} \in \mathbb{Z}_q^n$, the ciphertext is negligibly close to

$$(\mathbf{u} - \mathbf{t} \cdot p, \langle \mathbf{u}, \mathbf{s} \rangle + x + w \cdot p) = (\mathbf{u}, v + w \cdot p),$$

where $(\mathbf{u}, v)$ is drawn from $E_{\mathbf{A},\mathbf{b}}$ (again by Lemma 19). □

Finally, the next lemma is used for showing statistical security in the final hybrid game.

Lemma 21. *With overwhelming probability over the choice of a "malformed" public key* $(\mathbf{A}, \mathbf{b})$ *from the* uniform *distribution over* $\mathbb{Z}_q^{n \times m} \times \mathbb{Z}_q^m$*, the distribution* $E_{\mathbf{A},\mathbf{b}}$ *is within negligible statistical distance of the uniform distribution over* $\mathbb{Z}_q^n \times \mathbb{Z}_q$*.*

Proof. The proof is immediate from Lemma 15. Note that we apply the lemma to conclude that *all* of the $m + 1$ inner products in $E_{\mathbf{A},\mathbf{b}}$ are uniform. □

5.2.3.4 Proof Details

We first establish correctness. By Lemma 19, the noise in the c component of a ciphertext is distributed according to $\bar{\Psi}_\beta$ for some

$$\beta \leq \sqrt{2}r' \leq 4\alpha\sqrt{m} \cdot \omega(\sqrt{\log n}) \leq \frac{1}{p \cdot \omega(\sqrt{\log n})},$$

by Equation Equation (5.2.1). By the exponential tail inequality for Gaussians and the definition of $\bar{\Psi}_\beta$, the noise term does not exceed $q/2p = p/2$, except with negligible probability.

Next we establish security. We proceed with a sequences of games, as usual.

Game 0. We define Game 0 to be the original KDM security game. Let S_0 be the event that $\mathcal{A}$ wins this game. Then we trivially have

$$\mathsf{AdvKDM}_{\mathcal{A},\mathsf{PKE},\mathcal{F}_{\text{aff}}}(n) = \Pr[S_0]. \tag{5.2.2}$$

Game 1. We now describe an alternate game that faithfully simulates the true KDM attack game, up to negligible statistical distance. The game starts with access to the distribution $A_{\mathbf{s},\chi}$ for uniformly random $\mathbf{s} \in \mathbb{Z}_q^n$. For each user i, it applies the transformation described in Lemma 13 (using fresh draws from $A_{\mathbf{s},\chi}$) to produce the distribution $A_{\mathbf{s}_i,\chi}$, where $\mathbf{s}_i$ is distributed according to χ^n. As a side-effect, the transformation also outputs invertible square matrices $\bar{\mathbf{A}}_i \in \mathbb{Z}_q^{n\times n}$ and vectors $\bar{\mathbf{b}}_i \in \mathbb{Z}_q^n$ such that for all i,

$$\mathbf{s} = \bar{\mathbf{A}}_i^{-T}(\bar{\mathbf{b}}_i - \mathbf{s}_i) \bmod q.$$

Note that by setting the right-hand sides equal for any i, j and reducing modulo p, we have

$$\bar{\mathbf{A}}_i^{-T}(\mathbf{s}_i - \bar{\mathbf{b}}_i) = \bar{\mathbf{A}}_j^{-T}(\mathbf{s}_j - \bar{\mathbf{b}}_j) \bmod p \iff \mathbf{s}_i = \bar{\mathbf{A}}_{i,j}^T \cdot \mathbf{s}_j + \bar{\mathbf{b}}_{i,j} \bmod p, \tag{5.2.3}$$

where $\bar{\mathbf{A}}_{i,j} = \bar{\mathbf{A}}_j^{-1}\bar{\mathbf{A}}_i$ and $\bar{\mathbf{b}}_{i,j} = \bar{\mathbf{b}}_i - \bar{\mathbf{A}}_{i,j}^T \cdot \bar{\mathbf{b}}_j$. The game then generates a public key $(\mathbf{A}_i, \mathbf{b}_i)$ for each user i in the usual way by drawing m samples from $A_{\mathbf{s}_i,\chi}$.

We now describe how the game answers (key-dependent) message queries. Suppose the adversary requests an encryption, under the jth user's public key $(\mathbf{A}_j, \mathbf{b}_j)$, of the function $f_{\mathbf{t},w}(\mathbf{s}_i) = \langle \mathbf{t}, \mathbf{s}_i \rangle + w \in \mathbb{Z}_p$ (for some $\mathbf{t} \in \mathbb{Z}_p^n, w \in \mathbb{Z}_p$) applied to the ith user's secret key $\mathbf{s}_i$. Observe that

$$f_{\mathbf{t},w}(\mathbf{s}_i) = \langle \mathbf{t}, \mathbf{s}_i \rangle + w = (\underbrace{\bar{\mathbf{A}}_{i,j} \cdot \mathbf{t}}_{\mathbf{t}' \in \mathbb{Z}_p^n})^T \cdot \mathbf{s}_i + \underbrace{\langle \mathbf{t}, \bar{\mathbf{b}}_{i,j} \rangle + w}_{w' \in \mathbb{Z}_p}.$$

The game therefore draws a sample $(\mathbf{u}, v) \leftarrow E_{\mathbf{A}_j,\mathbf{b}_j}$ and outputs

$$(\mathbf{u} - \mathbf{t}' \cdot p, v + w' \cdot p) \in \mathbb{Z}_q^n \times \mathbb{Z}_q.$$

This completes the description of the game.

Let S_1 be the event that $\mathcal{A}$ wins Game 1. By the above description and Lemmas 13 and 20, the following claim is apparent.

Claim 22. *The distributions of the public keys and ciphertexts generated by the challenger in Game 1 are within negligible statistical distance of the distributions in Game 0.*

This claim gives

$$|\Pr[S_1] - \Pr[S_0]| \leq \text{negl}(n). \tag{5.2.4}$$

Game 2. The last hybrid game proceeds exactly as the one above, except that the initial distribution $A_{\mathbf{s},\chi}$ is replaced with the uniform distribution over $\mathbb{Z}_q^n \times \mathbb{Z}_q$). Note that the game above only treats $A_{\mathbf{s},\chi}$ as an oracle (it never uses $\mathbf{s}$ directly), so $A_{\mathbf{s},\chi}$ may be replaced in this way.

Now by Lemma 13, all the public keys $(\mathbf{A}_i, \mathbf{b}_i)$ generated by the game are uniform and independent. Moreover, by Lemma 21, all the (key-dependent) message queries are answered by ciphertexts that are uniform and independent of the message. The next claim follows, and the proof of Theorem 14 is complete.

Claim 23. *Assuming that* $\mathsf{LWE}_{q,\chi}$ *is hard, the distributions of Games 1 and 2 are computationally indistinguishable, assuming that the* $\mathsf{LWE}_{q,\chi}$ *problem is hard.*

Let S_2 be the event that $\mathcal{A}$ wins Game 2. Then the claim gives

$$|\Pr[S_2] - \Pr[S_1]| \leq \text{negl}(n). \tag{5.2.5}$$

But in Game 2, the bit b chosen by the challenger is never used. It is apparent that

$$\Pr[S_2] = 1/2. \tag{5.2.6}$$

The theorem is completed by collecting (5.2.2), (5.2.4), (5.2.5) and (5.2.6).

5.2.4 Amortized Extension

The system described in Section 5.2 encrypts only a single element $z \in \mathbb{Z}_p$ per syndrome $\mathbf{u} \in \mathbb{Z}_q^n$, so the ciphertext is a factor at least n larger than the message, and the encryption algorithm performs at least $n \cdot m$ operations per message element. Peikert, Vaikuntanathan, and Waters [66] proposed a significantly more efficient "amortized" version of the scheme, which can encrypt $\ell = O(n)$ symbols using only about twice the time and space. Applying our techniques, we can show that a variant of that system is also KDM-secure. The changes to the system in Section 5.2.2 are as follows.

- *Key generation:* The secret key is $\mathbf{S} \overset{\$}{\leftarrow} \chi^{n \times \ell}$. The public key is $(\mathbf{A}, \mathbf{B}) \in \mathbb{Z}_q^{n \times m} \times \mathbb{Z}_q^{m \times \ell}$ for $m \geq 2(n + \ell) \lg q$, where $\mathbf{B} = \mathbf{A}^T\mathbf{S} + \mathbf{X}$ for independent $\mathbf{A} \overset{\$}{\leftarrow} \mathbb{Z}_q^{n \times m}$ and $\mathbf{X} \overset{\$}{\leftarrow} \chi^{m \times \ell}$.

- *Encryption:* a distribution $E_{\mathbf{A},\mathbf{B}}$ over $\mathbb{Z}_q^n \times \mathbb{Z}_q^\ell$ is defined similarly. The distribution is obtained by outputting $(\mathbf{u}, \mathbf{v}) = (\mathbf{A}\mathbf{r}, \mathbf{B}^T\mathbf{r} + \mathbf{e})$ for $\mathbf{e} \overset{\$}{\leftarrow} \bar{\Psi}_{r'}^{\ell}$, *where the parameter r' is a $\sqrt{\ell}$ factor larger than before.* A message $\mathbf{z} \in \mathbb{Z}_p^\ell$ is encrypted as $(\mathbf{u}, \mathbf{c} = \mathbf{v} + \mathbf{z} \cdot p)$.

- *Decryption:* output the $\mathbf{z} \in \mathbb{Z}_p^\ell$ such that $\mathbf{z} \cdot p$ is closest to $\mathbf{c} - \mathbf{S}^T\mathbf{u}$ modulo q.

The proof of security extends to this system in a straightforward way, except in generalizing Lemma 19, which characterizes the ciphertext distribution and allows the simulator to encrypt the secret key (without knowing it, of course). In the analysis below, we give the details only for this portion, as the rest of the security proof is a straightforward adaptation of the basic scheme's analysis.

As in the basic scheme, we will start by showing that the first ciphertext component $\mathbf{u}$ is statistically close to uniform. Then we examine the conditional distribution of the noise in the second ciphertext component, given $\mathbf{u}$. It turns out that this conditional distribution is essentially the same for any $\mathbf{u}$, which implies that a simulator can produce simulated ciphertexts by manipulating the value $\mathbf{u}$ in a ciphertext and leave the noise in the second component untouched.

We start with a technical lemma that shows we can decompose a spherical Gaussian into two parts. For a matrix $\mathbf{X}$ with columns $\mathbf{x}_i$, let $\|\mathbf{X}\| = \max\|\mathbf{x}_i\|$, i.e., the

Euclidean length of its longest column.

Lemma 24. *Let* $\mathbf{X} \in \mathbb{Z}_q^{m\times\ell}$ *such that* $\|\mathbf{X}\| \leq L$*, where* $L = \left(\alpha + \frac{1}{2q}\right) \cdot \sqrt{m}$*. Let* $r' = \sqrt{\ell} \cdot L \cdot r$ *and* $\mathbf{e} \leftarrow \Psi_{r'}^{\ell}$*. Then there exist random variables* $\mathbf{e}_0, \mathbf{e}_1$ *such that* $\mathbf{e} = \mathbf{X}^T\mathbf{e}_0 + \mathbf{e}_1$*,* $\mathbf{e}_0 \leftarrow \Psi_r^m$ *and* $\mathbf{e}_1$ *is distributed as a non-spherical Gaussian on* $\mathbb{Z}^\ell$ *that is independent of* $\mathbf{e}_0$*.*

Proof. Let $\mathbf{e}_0$ have distribution Ψ_r^m, and define $\mathbf{e}_1 = \mathbf{e} - \mathbf{X}^T\mathbf{e}_0$. We will show that the constraints on the covariance matrix of $\mathbf{e}_1$ can be satisfied by a Gaussian, which is sufficient to prove the lemma. In particular, we will show that all of the eigenvalues of $\mathrm{Cov}(\mathbf{e}_1)$ are positive. First we have

$$\mathrm{Cov}(\mathbf{X}^T\mathbf{e}_0 + \mathbf{e}_1) = \mathrm{Cov}(\mathbf{X}^T\mathbf{e}_0) + \mathrm{Cov}(\mathbf{e}_1) = r^2\mathbf{X}^T\mathbf{X} + \mathrm{Cov}(\mathbf{e}_1).$$

Since each column of $\mathbf{X}$ has length at most L, every eigenvalue of $r^2\mathbf{X}^T\mathbf{X}$ is at most $r^2L^2\ell$ by the Cauchy-Shwarz inequality. We also have that $\mathrm{Cov}(\mathbf{e}) = (r')^2 \cdot \mathbf{I}$, so all of the eigenvalues of $\mathrm{Cov}(\mathbf{e})$ are equal to $(r')^2$.

Returning to $\mathbf{e}_1$, we can see that

$$\mathrm{Cov}(\mathbf{e}_1) = \mathrm{Cov}(\mathbf{e}) - \mathrm{Cov}(\mathbf{X}^T\mathbf{e}_0) = (r')^2\mathbf{I} - r^2\mathbf{X}^T\mathbf{X},$$

from which it is easily checked that the eigenvalues of $\mathrm{Cov}(\mathbf{e}_1)$ are at least $(r')^2 - \lambda_{\max}(r^2\mathbf{X}^T\mathbf{X})$, which is positive when r' satisfies

$$(r')^2 > r^2L^2\ell \geq \lambda_{\max}(r^2\mathbf{X}^T\mathbf{X})$$

(So $r' \geq r \cdot (\alpha \cdot \omega(\sqrt{\log m}) + 1/2q) \cdot \sqrt{\ell m}$.) Since all of the eigenvalues of $\mathrm{Cov}(\mathbf{e}_1)$ are positive, there exists a (non-spherical) Gaussian with this covariance matrix. □

Now we can prove a an analogous version of Lemma 19 for our amortized scheme.

Lemma 25. *Let* $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m}, \mathbf{X} \xleftarrow{\$} \chi, \mathbf{r} \xleftarrow{\$} D_{\mathbb{Z}^m,r}, \mathbf{e} \xleftarrow{\$} \bar{\Psi}_{r'}^{\ell}$*. Then the distribution of* $(\mathbf{A}\mathbf{r}, \mathbf{B}^T\mathbf{r} + \mathbf{e})$ *is within negligible statistical distance of* $(\mathbf{u}, \mathbf{S}^T\mathbf{u} + \mathbf{v})$*, where* $\mathbf{v}$ *is distributed as a multivariate (non-spherical) Gaussian independent of* $\mathbf{S}$ *and* $\mathbf{u}$*.*

Proof. By Lemma 15, the marginal distribution of $\mathbf{u} = \mathbf{Ar} \mod q$ is negligibly far from uniform. And by Lemma 17, the conditional distribution of $\mathbf{r}$, given $\mathbf{Ar} = \mathbf{u}$, is Gaussian over a particular shifted lattice. The lemma would follow immediately if this distribution were almost the same for every $\mathbf{u}$, but this is not the case. Instead we appeal to a result of Regev along with Lemma 24 to show the structure of $\mathbf{r}$ that depends on $\mathbf{u}$ is effectively destroyed by the additional noise $\mathbf{e}$.

In this line of reasoning, we first have that

$$\mathbf{B}^T\mathbf{r} + \mathbf{e} = \left(\mathbf{A}^T\mathbf{S} + \mathbf{X}\right)^T\mathbf{r} + \mathbf{e} = \mathbf{S}^T\mathbf{u} + \mathbf{X}^T\mathbf{r} + \mathbf{e}.$$

By observing that $\|\mathbf{X}\| \leq L$ with overwhelming probability, Lemma 24 applies and we can write the error term $\mathbf{X}^T\mathbf{r} + \mathbf{e} = \mathbf{X}^T(\mathbf{r} + \mathbf{e}_0) + \mathbf{e}_1$. By [69, Lemma 3.9], we have that the distribution of $\mathbf{r}' = \mathbf{r} + \mathbf{e}_0$ is negligibly far from a continuous spherical Gaussian distribution. The key property of the distribution of $\mathbf{r}'$ is that does not depend on $\mathbf{u}$. Once we observe that $\mathbf{e}_1$ does not depend on $\mathbf{u}$, the lemma follows. □

Lemma 26. *Let $\mathbf{t} \in \mathbb{Z}_p^n$ and $y \in \mathbb{Z}_p$ be arbitrary. With overwhelming probability over the choice of the public key $(\mathbf{A}, \mathbf{B})$ for arbitrary secret key $\mathbf{S} \in \mathbb{Z}_p^{n \times \ell}$, the following holds: for $(\mathbf{u}, \mathbf{c}) = (\mathbf{Ar}, \mathbf{B}^T\mathbf{r} + \mathbf{e})$, the distribution of*

$$(\mathbf{u} - \mathbf{t} \cdot p, \mathbf{c} + \mathbf{w} \cdot p) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$$

is within negligible statistical distance of a (properly generated) encryption of the message $\mathbf{S}^T\mathbf{t} + \mathbf{w} \in \mathbb{Z}_p$.

Proof. Exactly the same as Lemma 20. This should be deleted, but I'm leaving it for now in case it is needed. A proper encryption of $\mathbf{S}^T\mathbf{t} + \mathbf{w}$ is distributed as (up to negligible error)

$$(\mathbf{u}', \mathbf{S}^T\mathbf{u}' + \mathbf{v} + (\mathbf{S}^T\mathbf{t} + \mathbf{w}) \cdot p) = (\mathbf{u}', \mathbf{S}^T(\mathbf{u}' + \mathbf{t} \cdot p) + \mathbf{v} + \mathbf{w} \cdot p)$$

by Lemma 25. Rewriting $\mathbf{u}' = \mathbf{u} - p \cdot \mathbf{t}$ for a uniformly chosen $\mathbf{u} \in \mathbb{Z}_q^n$,

$$(\mathbf{u} - \mathbf{t} \cdot p, \mathbf{S}^T\mathbf{u} + \mathbf{v} + \mathbf{w} \cdot p)$$

$\square$

5.3 *Symmetric-Key Encryption Scheme Based on* LPN

In this section we describe and analyze our KDM-secure encryption scheme based on the LPN problem. Our construction is very simple and efficient, involving only bit arithmetic. It can encrypt multi-bit messages, including its entire key in one ciphertext. The KDM security analysis is also very simple compared to the analysis of [22] and our LWE-based scheme. In the analysis we exploit the natural homomorphic and "self referential" properties of the underlying learning problem, except now the arithmetic is over $\mathbb{Z}_2$ and there is no accumulation of error terms when encrypting, which eliminates many technical difficulties.

Our construction is similar to the scheme of Gilbert *et al.* [45], which is also based on LPN, except that our ciphertexts include a matrix $\mathbf{U}$ where they only need one vector $\mathbf{u}$. In particular, the computation for encryption and decryption per plaintext bit in our scheme is the same as in [45], as is the ratio of ciphertext length to plaintext length.

5.3.1 The Scheme

Fix $\epsilon > 0$. Let $\mathbf{C} \in \{0,1\}^{m\times n}$ be the generating matrix of an efficiently decodable linear code that can correct up to $d > (\epsilon + \alpha)m$ errors, for some $\alpha > 0$.

We will denote our scheme by $\mathsf{SKE}_{\text{lpn}}$. The key space and message space of the scheme are both $\{0,1\}^{n\times k}$, where n is determined by the LPN_ϵ parameters and k is free. In particular, if $k = 1$ the secret keys will be very short, but a larger k improves the ciphertext overhead.

- *Key generation*: The secret key chosen as $\mathbf{S} \overset{\$}{\leftarrow} \{0,1\}^{n\times k}$.

- *Encryption*: To encrypt a message $\mathbf{M} \in \{0,1\}^{n \times k}$ given the secret key $\mathbf{S}$, compute
$$\mathbf{U} \xleftarrow{\$} \{0,1\}^{m \times n}, \quad \mathbf{X} \xleftarrow{\$} \mathsf{Ber}_{\varepsilon}^{m \times k}, \quad \mathbf{B} \leftarrow \mathbf{CM} + \mathbf{US} + \mathbf{X}.$$
The ciphertext is $(\mathbf{U}, \mathbf{B})$.

- *Decryption*: To decrypt a ciphertext $(\mathbf{U}, \mathbf{B}) \in \{0,1\}^{m \times n} \times \{0,1\}^{m \times k}$ given the secret key $\mathbf{S}$, compute
$$\mathbf{W} \leftarrow \mathbf{B} + \mathbf{US}.$$
Then output $\mathbf{M} \in \{0,1\}^{n \times k}$, where each column of $\mathbf{M}$ is computed by decoding the corresponding column of $\mathbf{W}$.

We briefly compare our scheme to the recently proposed scheme of Gilbert *et al.* [45]. The ciphertexts in their scheme consist of a vector $\mathbf{u} \in \{0,1\}^n$, and the pad for the message is computed as $\mathbf{S}^t\mathbf{u}$. A ciphertext from our scheme can be seen as an entangled set of m ciphertexts from their scheme, where this entangling seems necessary to prove KDM security. While the ratio of ciphertext length to plaintext length of our scheme is the same (if both schemes are instantiated with the same size keys), our scheme is restricted in that it must send a full matrix $\mathbf{U}$ with each ciphertext, even if the message $\mathbf{M}$ does not require such a large pad.

5.3.2 Analysis

We observe that our scheme correctly decrypts with overwhelming probability and then analyze its KDM security.

Theorem 27 (Correctness). *Decryption errors occur with negligible probability. In particular, we have, for all* $\mathbf{M} \in \{0,1\}^{n \times k}$ *and all* $\mathbf{S} \in \{0,1\}^{n \times k}$*,*
$$\Pr\left[\mathsf{D}(\mathbf{S}, \mathsf{E}_{\mathbf{S}}(\mathbf{M})) \neq \mathbf{M}\right]$$
is negligible, where the probability is taken over the choice of $\mathbf{U}$ *and* $\mathbf{X}$*.*

Proof. A decryption error only occurs when a column of $\mathbf{X}$ has hamming weight greater than $(\epsilon + \alpha)m$. The probability that is happens for a fixed column is exponentially small in m when ϵ, α are constants by a simple Chernoff bound. We can then take a union bound over the columns of $\mathbf{X}$ to finish the proof. $\square$

The security analysis uses the following lemma, implicit in [45]. It can be proven by a simple hybrid argument.

Lemma 28. *Let $A_{\mathbf{S},\epsilon}$ be the distribution induced by computing*

$$\mathbf{U} \xleftarrow{\$} \{0,1\}^{m\times n}, \quad \mathbf{X} \xleftarrow{\$} \mathsf{Ber}_{\varepsilon}^{m\times k}, \quad \mathbf{R} \leftarrow \mathbf{U}\mathbf{S} + \mathbf{X}$$

and outputting $(\mathbf{U}, \mathbf{R}) \in \{0,1\}^{m\times n} \times \{0,1\}^{m\times k}$.

Then an oracle that returns samples from $A_{\mathbf{S},\epsilon}$ for a random $\mathbf{S} \in \{0,1\}^{n\times k}$ is indistinguishable from an oracle returning uniform samples, assuming that the LPN_ϵ problem is hard.

Below we use the following simple observations about the schemes homomorphic properties.

1. Let $\mathbf{D} \in \{0,1\}^{n\times n}$, and $(\mathbf{U}, \mathbf{R}) \xleftarrow{\$} A_{\mathbf{S},\epsilon}$. Then the distribution of

$$(\mathbf{U} + \mathbf{C}\mathbf{D}, \mathbf{R})$$

is exactly the same as the distribution of an encryption of the message $\mathbf{D}\mathbf{S}$ under key $\mathbf{S}$ – that is, the same distribution as $\mathsf{E}_{\mathbf{S}}(\mathbf{D}\mathbf{S})$. To see this, write $\mathbf{U}' = \mathbf{U} + \mathbf{C}\mathbf{D}$, and then we have that

$$(\mathbf{U} + \mathbf{C}\mathbf{D}, \mathbf{R}) = (\mathbf{U} + \mathbf{C}\mathbf{D}, \mathbf{U}\mathbf{S} + \mathbf{X}) = (\mathbf{U}', \mathbf{U}'\mathbf{S} + \mathbf{C}\,(\mathbf{D}\mathbf{S}) + \mathbf{X}),$$

where $\mathbf{X} \xleftarrow{\$} \mathsf{Ber}_{\varepsilon}^{m\times k}$.

2. Let $(\mathbf{U}, \mathbf{R}) \xleftarrow{\$} \mathsf{E}_{\mathbf{S}}(\mathbf{M})$. Then the distribution of

$$(\mathbf{U}, \mathbf{R} + \mathbf{C}\mathbf{W})$$

is exactly the same as the distribution of $\mathsf{E}_{\mathbf{S}}(\mathbf{M}+\mathbf{W})$. We also have that

$$(\mathbf{U}, \mathbf{R}+\mathbf{U}\mathbf{T})$$

has the same distribution as $\mathsf{E}_{\mathbf{S}+\mathbf{T}}(\mathbf{M})$. These can be verified through straightforward calculation.

In the following theorem, we denote by $f_{j,\mathbf{D},\mathbf{E}}$ the function $f_{j,\mathbf{D},\mathbf{E}}(\mathbf{S}_1, \ldots, \mathbf{S}_\ell) = \mathbf{D}\mathbf{S}_j + \mathbf{E}$, and let

$$\mathcal{F}_{\text{aff}} = \{f_{j,\mathbf{D},\mathbf{E}} \ : \ j \in [\ell], \mathbf{D} \in \{0,1\}^{n\times n}, \mathbf{E} \in \{0,1\}^{n\times k}\}.$$

Theorem 29 (Security). *The scheme* $\mathsf{SKE}_{\text{lpn}}$ *is KDM-CPA secure with respect to* $\mathcal{F}_{\text{aff}}$*, assuming that the* LPN_ϵ *problem is hard.*

Proof. We proceed with a sequence of games. Let $\mathcal{A}$ be an adversary attacking $\mathsf{SKE}_{\text{lpn}}$ is the KDM-CPA game with functions $\mathcal{F}_{\text{aff}}$.

Game 0. Game 0 is defined to be the original attack game. We have

$$\Pr[S_0] = \mathsf{AdvKDM}_{\mathcal{A},\mathsf{SKE}_{\text{lpn}},\mathcal{F}_{\text{aff}}}(n). \tag{5.3.1}$$

Game 1. Game 1 is like Game 0, except we change how the game is run by the challenger. These changes do not affect the distribution of the ciphertexts generated for the adversary, however.

Let $\mathbf{S}_i$ be the key for user i. In Game 0 each $\mathbf{S}_i$ is selected independently at random, but Game 1 samples them as follows. It selects $\mathbf{S} \overset{\$}{\leftarrow} \{0,1\}^{n\times k}$, and then for each user i, it computes

$$\mathbf{T}_i \overset{\$}{\leftarrow} \{0,1\}^{n\times k}, \quad \mathbf{S}_i \leftarrow \mathbf{S} + \mathbf{T}_i.$$

We first describe how to process an encryption query $(i, f_{j,\mathbf{D},\mathbf{E}})$, as follows.

1. It samples $(\mathbf{U}^{(1)}, \mathbf{R}^{(1)}) \overset{\$}{\leftarrow} A_{\mathbf{S},\epsilon}$. This sample has the same distribution as $\mathsf{E}_{\mathbf{S}}(\mathbf{0})$.

2. It computes

$$(\mathbf{U}^{(2)}, \mathbf{R}^{(2)}) \leftarrow (\mathbf{U}^{(1)} + \mathbf{CD}, \mathbf{R}^{(1)}).$$

   $(\mathbf{U}^{(2)}, \mathbf{R}^{(2)})$ now has the same distribution as $\mathsf{E}_{\mathbf{S}}(\mathbf{DS})$.

3. The challenger then computes

$$(\mathbf{U}^{(3)}, \mathbf{R}^{(3)}) \leftarrow (\mathbf{U}^{(2)}, \mathbf{R}^{(2)} + \mathbf{UT}_i).$$

   $(\mathbf{U}^{(3)}, \mathbf{R}^{(3)})$ has the same distribution as $\mathsf{E}_{\mathbf{S}+\mathbf{T}_i}(\mathbf{DS}) \equiv \mathsf{E}_{\mathbf{S}_i}(\mathbf{DS})$.

4. Finally, the challenger computes

$$(\mathbf{U}^{(4)}, \mathbf{R}^{(4)}) \leftarrow (\mathbf{U}^{(3)}, \mathbf{R}^{(3)} + \mathbf{CDT}_j + \mathbf{CE}).$$

   The distribution of $(\mathbf{U}^{(4)}, \mathbf{R}^{(4)})$ is exactly the same as that of

$$\mathsf{E}_{\mathbf{S}_i}(\mathbf{D}(\mathbf{S} + \mathbf{T}_j) + \mathbf{E}) \equiv \mathsf{E}_{\mathbf{S}_i}(\mathbf{DS}_j + \mathbf{E}) \equiv \mathsf{E}_{\mathbf{S}_i}(f_{j,\mathbf{D},\mathbf{E}}(\mathbf{S}_1, \ldots, \mathbf{S}_\ell)).$$

The challenger returns the ciphertext $(\mathbf{U}^{(4)}, \mathbf{R}^{(4)})$ to the adversary.

Now, when running the game, if $b = 0$ the challenger processes the query as described. If $b = 1$, the adversary instead uses the above procedure with the function f that is identically zero.

Since we have only changed the way the challenger runs the game, and not the distribution of ciphertexts returned, we have

$$\Pr[S_1] = \Pr[S_0]. \tag{5.3.2}$$

Game 2. Game 2 is exactly like Game 1, except with the following change: When processing encryption queries, in step 1, the challenger sets $(\mathbf{U}^{(1)}, \mathbf{R}^{(1)})$ a uniformly random sample. Everything else remains exactly the same.

We claim that

$$\big|\Pr[S_2] - \Pr[S_1]\big| \leq \text{negl}(n), \tag{5.3.3}$$

assuming that the LPN_ϵ problem is hard. To see this, we construct an adversary that solves the LPN_ϵ problem in the straightforward way: It selects $b \overset{\$}{\leftarrow} \{0,1\}$, and then responds to encryption queries are prescribed in Game 1, except that in step 1 it queries its own oracle, which either samples $A_{\mathbf{S},\epsilon}$ or uniform bits. It does not need $\mathbf{S}$ anywhere else in the simulation. It runs until $\mathcal{A}$ halts, and if $\mathcal{A}$ wins it guesses "real", and "random" otherwise.

We observe that, depending on the oracle, our adversary will simulate either Game 1 or Game 2. If there is a non-negligible difference in $\Pr[S_1]$ and $\Pr[S_2]$, this will give our adversary non-negligible advantage in distinguishing the oracles. Then by Lemma 28, our adversary can be used to solve the LPN_ϵ problem, which contradicts the assumed hardness.

Finally, it is apparent that

$$\Pr[S_2] = 1/2, \tag{5.3.4}$$

as the challenger's responses are independent of the bit b. The theorem is completed by collection (5.3.1),(5.3.2), (5.3.3) and (5.3.4).

□

5.4 Separating Standard Security and Circular Security

In this section we provide a simple example of a public-key encryption scheme that semantically secure under a standard assumption, but is not circular secure.

External Diffie-Hellman. Below we will use the so-called *symmetric external Diffie-Hellman (SXDH) assumption.* [6] To define the SXDH assumption, first recall the decisional Diffie-Hellman (DDH) problem, defined in §3.5. Let $\mathbb{G}$ be a group with prime order q, and let g be a generator of $\mathbb{G}$. The DDH problem is to distinguish the uniform distribution on $\mathbb{G}^3$ from the distribution induced by $(X, Y, \mathrm{dh}(X, Y))$ on $\mathbb{G}^3$, where X, Y are uniform random variables on $\mathbb{G}$, and dh is defined by $\mathrm{dh}(g^x, g^y) = g^{xy}$.

The SXDH assumption is defined with respect to groups that support an *asymmetric* pairing. That is, we consider triples of groups $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$, such that an efficient computable bilinear map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is available.

The SXDH assumption is then: *The DDH problem is intractable in both* $\mathbb{G}_1$ *and* $\mathbb{G}_2$. Here, intractable can mean that any probabilistic polynomial-time algorithm will be able to distinguish the distributions involved with at most negligible advantage. Of course, one must take an appropriate asymptotic version of the groups involved to define this precisely.

5.4.1 The Counterexample Scheme

We now describe our encryption scheme, denoted $\mathsf{PKE}_{\text{sxdh}}$, that is semantically-secure against chosen-plaintext attacks, but breaks completely when a two-cycle is published. Let $\text{en}_1 : \mathbb{Z}_q \to \mathbb{G}_1$ and $\text{en}_2 : \mathbb{Z}_q \to \mathbb{G}_2$ be efficiently invertible encoding functions.

Fix an asymmetric pairing $\hat{e} : \mathbb{G}_1, \mathbb{G}_2 \to \mathbb{G}_T$, and let g_1, g_2 be generators for $\mathbb{G}_1, \mathbb{G}_2$, respectively. A secret key in this scheme is a random $(x_1, x_2) \in \mathbb{Z}_q \times \mathbb{Z}_q$, and the corresponding public key is $(X_1, X_2) = (g_1^{x_1}, g_2^{x_2}) \in \mathbb{G}_1 \times \mathbb{G}_2$.

Encryption takes as input a public key (X_1, X_2) and a message $(m_1, m_2) \in \mathbb{Z}_q \times \mathbb{Z}_q$, where $m_i \neq 0$ for $i = 1, 2$. It chooses $(y_1, y_2) \xleftarrow{\$} \mathbb{Z}_q \times \mathbb{Z}_q, (r_1, r_2) \xleftarrow{\$} \mathbb{Z}_q \times \mathbb{Z}_q$ and, for $i = 1, 2$, computes

$$Y_i \leftarrow g_i^{y_i}, \quad Z_i \leftarrow X_i^{y_i}, \quad C_i \leftarrow \text{en}_i(m_i) \cdot Z_i.$$

It then computes

$$R_1 \leftarrow g_1^{r_1}, \quad R_2 \leftarrow g_2^{r_2}, \quad S_1 \leftarrow (X_1)^{r_1}, \quad S_2 \leftarrow (X_2)^{r_2}$$
$$\tau_1 \leftarrow S_1^{1/m_2}, \quad \tau_2 \leftarrow (S_2)^{1/m_1}.$$

A ciphertext is composed of $(Y_i, C_i, R_i, \tau_i) \in \mathbb{G}_i^4$ for $i = 1, 2$. We draw the reader's attention to the fact that τ_1 is computed using m_2, while τ_2 is computed using m_1.

Decryption gets as input a ciphertext

$$(\hat{Y}_1, \hat{C}_1, \hat{R}_1, \hat{\tau}_1),\ (\hat{Y}_2, \hat{C}_2, \hat{R}_2, \hat{\tau}_2)$$

ignores $\hat{R}_1, \hat{R}_2, \hat{\tau}_1, \hat{\tau}_2$ and uses only $\hat{Y}_1, \hat{Y}_2, \hat{C}_1, \hat{C}_2$, in addition to the secret key (x_1, x_2). For $i = 1, 2$, it computes

$$\hat{M}_i \leftarrow \hat{C}_i/(\hat{Y}_i^{x_i}), \quad \hat{m}_i \leftarrow \mathrm{en}^{-1}(\hat{M}_i).$$

It outputs $(\hat{m}_1, \hat{m}_2)$ as the decrypted message.

Circular insecurity. We first verify that the scheme is not circular-secure. More precisely, we will show that the circular-encryption of two secret keys sk and sk' is distinguishable from a pair of ciphertexts that is not an encrypted cycle. In this attack, the adversary is given either ciphertexts (C, C') generated according to

$$C \overset{\$}{\leftarrow} \mathsf{E}_{pk}(sk'), \quad C' \overset{\$}{\leftarrow} \mathsf{E}_{pk'}(sk) \tag{5.4.1}$$

or as

$$C \overset{\$}{\leftarrow} \mathsf{E}_{pk}(m'), \quad C' \overset{\$}{\leftarrow} \mathsf{E}_{pk'}(m) \tag{5.4.2}$$

for some arbitrary messages $(m, m') \neq (sk, sk')$. These message can be random, fixed, or whatever is appropriate for the exact definition we wish to separate.

The attack proceeds as follows. Let $sk = (x_1, x_2), sk' = (x_1', x_2')$, and write

$$C = \big((Y_1, C_1, R_1, \tau_1),\ (Y_2, C_2, R_2, \tau_2)\big)$$

and

$$C' = \big((Y_1', C_1', R_1', \tau_1'),\ (Y_2', C_2', R_2', \tau_2')\big)$$

The attack simply tests if

$$\hat{e}(\tau_1, \tau_2') = \hat{e}(R_1, R_2'). \tag{5.4.3}$$

We now analyze the test in (5.4.3). Suppose (C, C') was generated according to (5.4.1). Then we have

$$\begin{aligned}\hat{e}(\tau_1, \tau_2') &= \hat{e}\big((X_1)^{r_1/x_2'}, (X_2')^{r_2'/x_1}\big) = \hat{e}\big(g_1^{r_1 \cdot x_1/x_2'}, g_2^{r_2' \cdot x_2'/x_1}\big) \\ &= \hat{e}\big(g_1^{r_1}, g_2^{r_2}\big)^{(x_1/x_2')\cdot(x_2'/x_1)} \\ &= \hat{e}(R_1, R_2'),\end{aligned}$$

and (5.4.3) holds with probability 1.

Now consider the case where (C, C') was generated according to (5.4.2). Write $m = (m_1, m_2)$ and $m' = (m_1', m_2')$. Then we have

$$\begin{aligned}\hat{e}(\tau_1, \tau_2') &= \hat{e}\big((X_1)^{r_1/m_2'}, (X_2')^{r_2'/m_1}\big) = \hat{e}\big(g_1^{r_1 \cdot x_1/m_2'}, g_2^{r_2' \cdot x_2'/m_1}\big) \\ &= \hat{e}\big(g_1^{r_1}, g_2^{r_2}\big)^{(x_1/m_2')\cdot(x_2'/m_1)} \\ &= \hat{e}(R_1, R_2')^{(x_1/m_2')\cdot(x_2'/m_1)}.\end{aligned}$$

From this we can conclude that when

$$(x_1/m_2') \cdot (x_2'/m_1) \neq 1 \ \in \mathbb{Z}_q,$$

the test in (5.4.3) will not hold. If m_1, m_2' are random elements, or some fixed message like $\text{en}_i(0)$, this equality will hold with negligible probability.

Thus, the test correctly distinguishes a circular encryption from a non-circular encryption while erring with only negligible probability. This concludes the attack on circular security.

Standard security. In this section we verify that the scheme is semantically secure, assuming that the SXDH problem is hard. This proof is quite standard.

Theorem 30. *The public-key encryption scheme* $\mathsf{PKE}_{\text{sxdh}}$ *is semantically-secure against chosen plaintext attacks, assuming the DDH problem is hard in* $\mathbb{G}_1$ *and* $\mathbb{G}_2$.

Proof. We proceed with a sequence of games. Let $\mathcal{A}$ be an adversary attacking $\mathsf{PKE}_{\text{sxdh}}$, and let S_i be the event that $\mathcal{A}$ wins Game i.

Game 0. Game 0 is defined to be the chosen plaintext attack game. The following is then apparent.

$$\Pr[S_0] = \mathsf{AdvCPA}_{\mathcal{A},\mathsf{PKE}_{\mathrm{sxdh}}}. \tag{5.4.4}$$

Game 1. Game 1 is exactly like Game 0, except with the following differences. When generating the challenge ciphertext, instead of computing Z_1 as prescribed, the challenger selects it at random from the appropriate groups. A standard argument shows that there is an SXDH adversary $\mathcal{B}_{\mathrm{ddh1}}$, running in about the same time as $\mathcal{A}$, such that

$$\big|\Pr[S_1] - \Pr[S_0]\big| \leq \mathsf{AdvDDH}_{\mathcal{B}_{\mathrm{ddh1}},\mathbb{G}_1}. \tag{5.4.5}$$

We outline the standard argument. The adversary $\mathcal{B}_{\mathrm{ddh1}}$ gets $X_1, Y_1, Z_1 \in \mathbb{G}_1$ as input in the DDH game, where Z_1 is set to either $\mathrm{dh}(X_1, Y_1)$ or to a random element of $\mathbb{G}_1$. It uses X_1 in the public key, and selects X_2 as normal. For the challenge ciphertext, it uses Y_1 and Z_1 in place of normal computations. It runs $\mathcal{A}$ until it halts, and then checks if it wins the simulated game. If so, $\mathcal{B}_{\mathrm{ddh1}}$ guesses "real" and otherwise it guesses "random." It is not hard to verify that, depending on how Z_1 was set in the DDH game, $\mathcal{B}_{\mathrm{ddh1}}$ will simulate either Game 0 or Game 1. The claim follows.

Game 2. Game 2 is like Game 1, except now the values Z_2, S_1, S_2 are selected at random instead of being computed normally. By repeating an argument that is nearly identical to the above one, we have

$$\big|\Pr[S_2] - \Pr[S_1]\big| \leq \mathsf{AdvDDH}_{\mathcal{B}_{\mathrm{ddh1}},\mathbb{G}_1} + 2 \cdot \mathsf{AdvDDH}_{\mathcal{B}_{\mathrm{ddh2}},\mathbb{G}_2} \tag{5.4.6}$$

for adversaries $\mathcal{B}_{\mathrm{ddh1}}, \mathcal{B}_{\mathrm{ddh2}}$ that run in about the same time as $\mathcal{A}$. We note that to replace Z_2 and S_2 with random elements we must appeal to the DDH assumption in $\mathbb{G}_2$.

Game 3. In Game 3, the challenger ignores the messages submitted by $\mathcal{A}$ in the challenge query, and instead encrypts a random message. It is clear that this does not change the distribution of the ciphertext given to $\mathcal{A}$, and so we have

$$\Pr[S_2] = \Pr[S_1]. \tag{5.4.7}$$

But it is also clear that

$$\Pr[S_2] = 1/2. \tag{5.4.8}$$

The proof is completed by collecting (5.4.4),(5.4.5), (5.4.6), (5.4.7) and (5.4.8). $\square$

REFERENCES

[1] Abdalla, M., Bellare, M., and Rogaway, P., "The oracle diffie-hellman assumptions and an analysis of dhies," in *CT-RSA*, pp. 143–158, 2001.

[2] Abdalla, M. and Pointcheval, D., "Simple password-based encrypted key exchange protocols," in *CT-RSA*, pp. 191–208, 2005.

[3] Adão, P., Bana, G., Herzog, J., and Scedrov, A., "Soundness of formal encryption in the presence of key-cycles," in *ESORICS*, pp. 374–396, 2005.

[4] Ajtai, M., "Generating hard instances of lattice problems," *Quaderni di Matematica*, vol. 13, pp. 1–32, 2004. Preliminary version in STOC 1996.

[5] Applebaum, B., "Fast cryptographic primitives based on the hardness of decoding random linear code," *Princeton University Department of Computer Science Technical Report TR-845-08*, December 2008.

[6] Ateniese, G., Camenisch, J., Hohenberger, S., and de Medeiros, B., "Practical group signatures without random oracles." Cryptology ePrint Archive, Report 2005/385, 2005. `http://eprint.iacr.org/`.

[7] Baek, J., Lee, B., and Kim, K., "Secure length-saving elgamal encryption under the computational diffie-hellman assumption," in *ACISP*, pp. 49–58, 2000.

[8] Barak, B., "Constant-round coin-tossing with a man in the middle or realizing the shared random string model," in *FOCS*, pp. 345–355, 2002.

[9] Barak, B., Prabhakaran, M., and Sahai, A., "Concurrent non-malleable zero knowledge," in *FOCS*, pp. 345–354, 2006.

[10] Bellare, M., Desai, A., Jokipii, E., and Rogaway, P., "A concrete security treatment of symmetric encryption," in *FOCS*, pp. 394–403, 1997.

[11] Bellare, M. and Rogaway, P., "Random oracles are practical: A paradigm for designing efficient protocols," in *ACM Conference on Computer and Communications Security*, pp. 62–73, 1993.

[12] Bellare, M. and Rogaway, P., "Code-based game-playing proofs and the security of triple encryption." Cryptology ePrint Archive, Report 2004/331, 2004. `http://eprint.iacr.org/`.

[13] Bellare, M. and Sahai, A., "Non-malleable encryption: Equivalence between two notions, and an indistinguishability-based characterization," in *CRYPTO*, pp. 519–536, 1999.

[14] Black, J., Rogaway, P., and Shrimpton, T., “Encryption-scheme security in the presence of key-dependent messages,” in *Selected Areas in Cryptography*, pp. 62–75, 2002.

[15] Blum, A., Kalai, A., and Wasserman, H., “Noise-tolerant learning, the parity problem, and the statistical query model,” *J. ACM*, vol. 50, no. 4, pp. 506–519, 2003.

[16] Boldyreva, A., Cash, D., Fischlin, M., and Warinschi, B., “Foundations of non-malleable hash and one-way functions,” in *ASIACRYPT*, 2009. To appear.

[17] Boldyreva, A. and Fischlin, M., “Analysis of random oracle instantiation scenarios for oaep and other practical schemes,” in *CRYPTO*, pp. 412–429, 2005.

[18] Boldyreva, A. and Fischlin, M., “On the security of oaep,” in *ASIACRYPT*, pp. 210–225, 2006.

[19] Boneh, D. and Boyen, X., “Efficient selective-id secure identity-based encryption without random oracles,” in *EUROCRYPT*, pp. 223–238, 2004.

[20] Boneh, D. and Boyen, X., “Short signatures without random oracles and the sdh assumption in bilinear groups,” *J. Cryptology*, vol. 21, no. 2, pp. 149–177, 2008.

[21] Boneh, D. and Franklin, M. K., “Identity-based encryption from the weil pairing,” in *CRYPTO*, pp. 213–229, 2001.

[22] Boneh, D., Halevi, S., Hamburg, M., and Ostrovsky, R., “Circular-secure encryption from decision Diffie-Hellman,” in *CRYPTO*, pp. 108–125, 2008.

[23] Boyen, X., Mei, Q., and Waters, B., “Direct chosen ciphertext security from identity-based techniques,” in *ACM Conference on Computer and Communications Security*, pp. 320–329, 2005.

[24] Camenisch, J., Chandran, N., and Shoup, V., “A public key encryption scheme secure against key dependent chosen plaintext and adaptive chosen ciphertext attacks,” in *EUROCRYPT*, pp. 351–368, 2009.

[25] Camenisch, J. and Lysyanskaya, A., “An efficient system for non-transferable anonymous credentials with optional anonymity revocation,” in *EUROCRYPT*, pp. 93–118, 2001.

[26] Canetti, R., “Towards realizing random oracles: Hash functions that hide all partial information,” in *CRYPTO*, pp. 455–469, 1997.

[27] Canetti, R., Goldreich, O., and Halevi, S., “The random oracle methodology, revisited,” *J. ACM*, vol. 51, no. 4, pp. 557–594, 2004.

[28] Canetti, R., Halevi, S., and Steiner, M., “Mitigating dictionary attacks on password-protected local storage,” in *CRYPTO*, pp. 160–179, 2006.

[29] Canetti, R., Micciancio, D., and Reingold, O., "Perfectly one-way probabilistic hash functions (preliminary version)," in *STOC*, pp. 131–140, 1998.

[30] Cash, D., Kiltz, E., and Shoup, V., "The twin diffie-hellman problem and applications," in *EUROCRYPT*, pp. 127–145, 2008.

[31] Coron, J.-S., Handschuh, H., Joye, M., Paillier, P., Pointcheval, D., and Tymen, C., "Gem: A generic chosen-ciphertext secure encryption method," in *CT-RSA*, pp. 263–276, 2002.

[32] Cramer, R. and Shoup, V., "Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack," *SIAM J. on Comput.*, vol. 33, pp. 167–226, 2003.

[33] Crescenzo, G. D., Ishai, Y., and Ostrovsky, R., "Non-interactive and non-malleable commitment," in *STOC*, pp. 141–150, 1998.

[34] Damgård, I. and Groth, J., "Non-interactive and reusable non-malleable commitment schemes," in *STOC*, pp. 426–437, 2003.

[35] Diffie, W. and Hellman, M. E., "New directions in cryptography," *IEEE Transactions on Information Theory*, vol. IT-22, no. 6, pp. 644–654, 1976.

[36] Dolev, D., Dwork, C., and Naor, M., "Nonmalleable cryptography," *SIAM J. Comput.*, vol. 30, no. 2, pp. 391–437, 2000.

[37] Feldman, V., Gopalan, P., Khot, S., and Ponnuswami, A. K., "New results for learning noisy parities and halfspaces," in *FOCS*, pp. 563–574, 2006.

[38] Fischlin, M., "Pseudorandom function tribe ensembles based on one-way permutations: Improvements and applications," in *EUROCRYPT*, pp. 432–445, 1999.

[39] Fischlin, M., "Security of nmac and hmac based on non-malleability," in *CT-RSA*, pp. 138–154, 2008.

[40] Fischlin, M. and Fischlin, R., "Efficient non-malleable commitment schemes," in *CRYPTO*, pp. 413–431, 2000.

[41] Fujisaki, E. and Okamoto, T., "How to enhance the security of public-key encryption at minimum cost," in *Public Key Cryptography*, pp. 53–68, 1999.

[42] Fujisaki, E., Okamoto, T., Pointcheval, D., and Stern, J., "Rsa-oaep is secure under the rsa assumption," in *CRYPTO*, pp. 260–274, 2001.

[43] Gennaro, R., Krawczyk, H., and Rabin, T., "Secure hashed diffie-hellman over non-ddh groups," in *EUROCRYPT*, pp. 361–381, 2004.

[44] Gentry, C., Peikert, C., and Vaikuntanathan, V., "Trapdoors for hard lattices and new cryptographic constructions," in *STOC*, pp. 197–206, 2008.

[45] Gilbert, H., Robshaw, M. J. B., and Seurin, Y., "How to encrypt with the LPN problem," in *ICALP (2)*, pp. 679–690, 2008.

[46] Goldreich, O., *Foundations of Cryptography: Basic Tools.* New York, NY, USA: Cambridge University Press, 2000.

[47] Goldreich, O. and Levin, L. A., "A hard-core predicate for all one-way functions," in *STOC*, pp. 25–32, 1989.

[48] Goldwasser, S. and Micali, S., "Probabilistic encryption," *J. Comput. Syst. Sci.*, vol. 28, no. 2, pp. 270–299, 1984.

[49] Halevi, S., "Eme*: Extending eme to handle arbitrary-length messages with associated data," in *INDOCRYPT*, pp. 315–327, 2004.

[50] Halevi, S. and Rogaway, P., "A tweakable enciphering mode," in *CRYPTO*, pp. 482–499, 2003.

[51] Halevi, S. and Rogaway, P., "A parallelizable enciphering mode," in *CT-RSA*, pp. 292–304, 2004.

[52] Hofheinz, D. and Kiltz, E., "Secure hybrid encryption from weakened key encapsulation," in *CRYPTO*, pp. 553–571, 2007.

[53] Hofheinz, D. and Unruh, D., "Towards key-dependent message security in the standard model," in *EUROCRYPT*, pp. 108–126, 2008.

[54] Katz, J., "Efficient cryptographic protocols based on the hardness of learning parity with noise," in *IMA Int. Conf.*, pp. 1–15, 2007.

[55] Kurosawa, K. and Matsuo, T., "How to remove mac from dhies," in *ACISP*, pp. 236–247, 2004.

[56] Libert, B. and Quisquater, J.-J., "Identity based encryption without redundancy," in *ACNS*, pp. 285–300, 2005.

[57] Menezes, A. J., Vanstone, S. A., and Oorschot, P. C. V., *Handbook of Applied Cryptography.* Boca Raton, FL, USA: CRC Press, Inc., 1996.

[58] Micciancio, D., "Improving lattice based cryptosystems using the Hermite normal form," in *CaLC*, pp. 126–145, 2001.

[59] Micciancio, D. and Regev, O., "Worst-case to average-case reductions based on Gaussian measures.," *SIAM J. Comput.*, vol. 37, no. 1, pp. 267–302, 2007. Preliminary version in FOCS 2004.

[60] Micciancio, D. and Regev, O., "Lattice-based cryptography," in *Post Quantum Cryptography* (Bernstein, D. J., Buchmann, J., and Dahmen, E., eds.), pp. 147–191, Springer, February 2009.

[61] Okamoto, T. and Pointcheval, D., "The gap-problems: A new class of problems for the security of cryptographic schemes," in *Public Key Cryptography*, pp. 104–118, 2001.

[62] P1619, I., "Standard for cryptographic protection of data on block-oriented storage devices," 2007. https://siswg.net/.

[63] Pass, R. and Rosen, A., "Concurrent non-malleable commitments," in *FOCS*, pp. 563–572, 2005.

[64] Pass, R. and Rosen, A., "New and improved constructions of non-malleable cryptographic protocols," in *STOC*, pp. 533–542, 2005.

[65] Peikert, C., "Public-key cryptosystems from the worst-case shortest vector problem: extended abstract," in *STOC*, pp. 333–342, 2009.

[66] Peikert, C., Vaikuntanathan, V., and Waters, B., "A framework for efficient and composable oblivious transfer," in *CRYPTO*, pp. 554–571, 2008.

[67] Phan, D. H. and Pointcheval, D., "About the security of ciphers (semantic security and pseudo-random permutations)," in *Selected Areas in Cryptography*, pp. 182–197, 2004.

[68] Rackoff, C. and Simon, D. R., "Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack," in *CRYPTO*, pp. 433–444, 1991.

[69] Regev, O., "On lattices, learning with errors, random linear codes, and cryptography," in *STOC*, pp. 84–93, 2005.

[70] Sahai, A., "Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security," in *FOCS*, pp. 543–553, 1999.

[71] Sakai, R. and Kasahara, M., "Id based cryptosystems with pairing on elliptic curve." Cryptology ePrint Archive, Report 2003/054, 2003. `http://eprint.iacr.org/`.

[72] Santis, A. D., Crescenzo, G. D., Ostrovsky, R., Persiano, G., and Sahai, A., "Robust non-interactive zero knowledge," in *CRYPTO*, pp. 566–598, 2001.

[73] Shamir, A., "Identity-based cryptosystems and signature schemes," in *CRYPTO*, pp. 47–53, 1984.

[74] Shoup, V., "Sequences of games: A tool for taming complexity in security proofs," 2004. Available at http://www.shoup.net/papers/.

[75] Steinfeld, R., Baek, J., and Zheng, Y., "On the necessity of strong assumptions for the security of a class of asymmetric encryption schemes," in *ACISP*, pp. 241–256, 2002.

[76] Yao, A. C.-C., "Theory and applications of trapdoor functions (extended abstract)," in *FOCS*, pp. 80–91, 1982.