Securely Outsourcing Computations using Homomorphic Encryption

Vom Fachbereich Informatik der
Technischen Universität Darmstadt genehmigte

Dissertation

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

Dipl.-Math. Andreas Peter (M.A.St.)

geboren in Oldenburg

TECHNISCHE
UNIVERSITÄT
DARMSTADT

Referenten:	Prof. Dr. Stefan Katzenbeisser
	Technische Universität Darmstadt
	Prof. Dr. Frederik Armknecht
	Universität Mannheim
Tag der Einreichung:	14. Januar 2013
Tag der mdl. Prüfung:	22. Februar 2013
Hochschulkennziffer:	D 17

Darmstadt 2013

Abstract

Web-servers are a dominant medium of modern communication and thus process vast amounts of highly sensitive, private data. Social networks, online auctions, and all kinds of cloud services constitute popular examples of online services complying with this paradigm. Due to many alarming scandals regarding the misuse of private data, IT security solutions protecting this sensitive data (for example by hiding it through encryption) are indispensable. Ideally, web-servers should be able to compute on private data without seeing it—we refer to this as the "Secure Outsourcing of Computation". Most notably, the cryptographic primitive of Homomorphic Encryption (HE) enables the web-servers to process data in the encrypted domain, which thereby hides/protects all private data as it is never available in the clear. More precisely, HE is a type of encryption that allows for the computation of certain functionalities on encrypted data without knowledge of the decryption key. Depending on the HE scheme, various functionalities can be evaluated in the encrypted domain, ranging from additions or multiplications only (Group Homomorphic Encryption, GHE), to virtually any computable functionality (Fully Homomorphic Encryption, FHE).

We provide a clean foundational framework for HE that permits security characterizations for a large class of HE schemes and shows strong similarities between GHE and FHE. We study these characterizations in order to assess the limits of HE, and show that GHE is obsolete in the quantum world, meaning that any GHE scheme can be broken by a quantum computer. Together with our newly constructed universal blueprint that encompasses virtually any GHE scheme, we know exactly how these schemes must be designed and what the underlying hardness assumptions must look like. To some extent, this rounds off the topic of GHE both in the standard and quantum model of computation.

In the standard model (which is presently assumed to be the most realistic model of computation), GHE is still highly attractive as it provides very efficient solutions to a variety of practical problems. On the basis of our foundational framework, we construct new GHE schemes with unique features and show their employability in practical applications. Most importantly, we use the special features of one of these schemes and develop a novel technique to achieve a practically efficient solution to the problem of securely outsourcing computations. Our construction allows multiple users to send encryptions of their private data under their own keys to a distrusted web-server that can then, on behalf of the users, evaluate any dynamically chosen computable function on these inputs in the encrypted domain (even across encryptions under different public keys) without learning anything at all, and without interacting with the users.

Zusammenfassung

Ein überwiegender Teil des modernen Datenaustauschs geschieht heutzutage über Webserver, welche Unmengen an sensiblen, privaten Daten verarbeiten. Zu den wichtigsten Beispielen zählen Online-Dienste, wie soziale Netzwerke, Internet-Auktionen und alle möglichen Cloud-Dienste. Viele alarmierende Skandale, die den Missbrauch privater Daten betreffen, zeigen die unbedingte Notwendigkeit von Lösungen aus dem Bereich der IT-Sicherheit, welche diese sensiblen Daten schützen (zum Beispiel durch die Benutzung von Verschlüsselungsverfahren). Idealerweise sollten Webserver in der Lage sein mit privaten Daten zu rechnen, ohne diese zu sehen – wir nennen einen solchen Mechanismus eine „sichere Auslagerung von Berechnungen". In diesem Zusammenhang ist die Homomorphe Verschlüsselung (HV) einer der bedeutendsten Bausteine der IT-Sicherheit bzw. der Kryptographie. HV ist ein Verschlüsselungsmechanismus, der es ermöglicht bestimmte Berechnungen auf verschlüsselten Daten auszuführen ohne den Entschlüsselungsschlüssel zu kennen. Abhängig von dem jeweiligen HV-Verfahren können verschiedenste Funktionen ausgewertet werden, angefangen bei simplen Additionen oder Multiplikationen (Gruppen-Homomorphe Verschlüsselung, GHV) bis hin zu nahezu allen berechenbaren Funktionen (Voll-Homomorphe Verschlüsselung, VHV).

In dieser Arbeit beschäftigen wir uns zunächst mit den Grundlagen von HV und geben diverse Äquivalenzbedingungen an die Sicherheit von HV-Verfahren, sowie starke Ähnlichkeiten zwischen GHV- und VHV-Verfahren an. Wir studieren diese Bedingungen im Detail, um die Grenzen von HV zu ermitteln und zeigen, dass GHV in der Quantenwelt obsolet ist, womit gemeint ist, dass jedes solche Verfahren von einem Quantumcomputer gebrochen werden kann. Zusammen mit einem von uns neu entwickelten, universellen Konstruktionsentwurf für GHV, wissen wir dann genau, wie solche Verfahren konstruiert werden müssen und, wie die zugrundeliegenden komplexitätstheoretischen Annahmen aussehen müssen. Zu einem gewissen

Grad rundet dies das Thema GHV sowohl in dem Standard- als auch in dem Quantenberechnungsmodell ab.

Im Standardmodell (welches heutzutage als das realistischste Berechnungsmodell angesehen wird) ist GHV immer noch höchst attraktiv, da es genutzt wird, um sehr effiziente Lösungen für eine Vielzahl von praxisrelevanten Problemen bereitzustellen. Basierend auf unserer Grundlagenforschung zu HV, können wir neue GHV-Verfahren mit einzigartigen Eigenschaften konstruieren, welche wir dann benutzen, um Probleme aus der Praxis zu lösen. Von großer Bedeutung ist in diesem Zusammenhang unsere Erstellung eines besonderen Verfahrens, welches uns erlaubt eine neue Technik vorzustellen, um das Problem des sicheren Auslagerns von Berechnungen in einer sehr effizienten Art und Weise zu lösen. Konkret erlaubt es unsere Technik mehreren Benutzern, Verschlüsselungen ihrer privaten Daten unter ihren eigenen Schlüsseln an einen nicht-vertrauenswürdigen Webserver zu schicken, welcher dann im Auftrag der Benutzer jede beliebige berechenbare Funktion auf diesen verschlüsselten Daten zu berechnen (sogar über Verschlüsselungen unter verschiedenen Schlüsseln hinweg) ohne irgendetwas Sicherheitskritisches zu lernen und ohne mit den Benutzern zu interagieren.

Acknowledgments

Almost four years ago, I embarked on a long journey of adventurous paths in applied cryptographic research. This dissertation marks an important destination on this journey, as it constitutes the end of my PhD studies at the Technische Universität Darmstadt. All along the way, there were many people who accompanied me, gave me constant support, and made this journey a very exciting and enjoyable one. I would like to take this opportunity to express my deep gratitude to all of those.

First and foremost, I want to thank my supervisor Stefan Katzenbeisser who made this journey possible, admitted me a lot of freedom in pursuing my research (while always providing guidance and new food for thought), gave me constant encouragement, showed me the right directions, and supported me in any conceivable way. Working with him was (and still is) a great pleasure and asset. I am sure that there will be many more successful joint research efforts with him. Moreover, being part of the Security Engineering Group always led to very fruitful discussions within the research team: Sami Alsouri, Sebastian Biedermann, Wolfgang Böhmer, Martin Franz, Nikolaos Karvelas, Martin Mink, Sascha Müller, Cuong Nguyen, André Schaller, Heike Schröder, Marius Senftleben, Marion Steiner, and Erik Tews. Thank you for being such good colleagues and friends! Also, I would like to thank Heike Meissner and Andrea Püchner for their organizational support.

In addition, I am heavily indebted to Frederik Armknecht, who introduced me to the topic of homomorphic encryption and with whom I spent hours and hours discussing many different research ideas. Thank you so much for all the time you invested in our joint projects.

Furthermore, I would like to thank the TU Darmstadt and CASED for the great working environment they offered me during my PhD studies. Mostly, I value the diversity of research topics that are covered by the many researchers working there. The numerous discussions I've had around the whiteboard and in the corridors of CASED, always gave me new input and made me see things from a different angle. In particular, I would like to thank: Paul Baecher, Christina Brzuska, Özgür Dagdelen, Pooya Farshim, Marc Fischlin, Tommaso Gagliardoni, Felix Günther, Ünal Koçabas, Mark Manulis, Giorgia Marson, Arno Mittelbach, Cristina Onete, Bertram Poettering, and Dominique Schröder.

Finally, I would like to express my gratitude to my parents Monika and Norbert, my brother Simon, my girlfriend Blanca, and all my friends for their constant encouragement.

Darmstadt, 27 March 2013
Andreas Peter

Contents

1 Introduction

Outsourcing computations on very sensitive data to a distrusted party in a *secure* manner (meaning that the distrusted party learns neither about the underlying sensitive data nor about the result of the computations) is one of the most central topics in modern IT security. In fact, first approaches already appeared with the advent of public-key cryptography [35] in 1976. Nowadays, the topic affects everyone who uses modern online communication being done through central web-servers which process vast amounts of private data. Examples of online services exploiting this paradigm are social networks, online auctions, and cloud services to name just a few. In the recent years, many concerns have been raised regarding data privacy in these scenarios, and serious privacy breaches have occurred [2, 4, 6, 101].

In order to deal with these privacy threats to sensitive data, several different approaches to securely outsource such computations have been proposed, depending on the application scenario. In this work, we focus on solutions that are based on Homomorphic Encryption (HE), which is a type of encryption that allows for the computation of certain functionalities on encrypted data without being able to decrypt. However, current practically feasible HE supports only very limited functionalities (like simple additions) and, hence, does not solve the problem in all its generality. In order to perform more complex operations, decryption steps are needed, which require parties holding a secret key, or a share thereof, to be online and interact in the computations. To realize these steps in a secure way, the concept of Secure Multiparty Computation (SMC) [114] gains increasing importance; here again, our focus lies on solutions based on HE [28, 70]. In this setting, the computation is carried out interactively between several participating parties, in a way that sensitive data is kept hidden (for example encrypted or shared among protocol participants) and only the desired output of the computation is available. Note that, depending on the application, the interactive nature of SMC protocols is not necessarily contradictory to the setting of secure outsourcing (sometimes one only wants to outsource certain parts of the overall computation, and it is acceptable to interact for certain other parts). Overall, this thesis has two main goals:

1. Providing the first thorough foundations on HE (encompassing all existing schemes) to work out the limits of their employability in the secure outsourcing of computations, both in constructive and restrictive ways.

2. Developing the first *efficient* mechanism (on the basis of the first goal), allowing the secure outsourcing of arbitrary computations in the presence of multiple users, but *non-interactively* with these users.

Concretely, the contributions of this thesis with respect to these goals are as follows:

Abstract Security Characterizations of HE (Chapter 4)

We identify and formalize the most basic underlying structure of all existing HE schemes and say that schemes having this structure are of *shift-type*. This particular structure alone, which essentially says that encrypting a message amounts to the addition of random noise, allows us to characterize their IND-CPA security (being the minimal security requirement on HE schemes) in terms of a certain abstract *SubSet Membership Problem* (SSMP). Moreover, for a certain large subclass of HE schemes called *Group Homomorphic Encryption* (GHE) schemes where this abstract problem is considered over groups (which is then called the *SubGroup Membership Problem*, SMP) we can even characterize their IND-CCA1 security (the strongest security requirement on HE schemes) in terms of a variant of the SMP, called the *Splitting Oracle-Assisted Subgroup Membership Problem* (SOAP). GHE schemes work on plaintext groups, such as $\mathbb{Z}_N$ for $N \in \mathbb{N}$, and allow for the evaluation of arbitrarily many group operations in the encrypted domain. In the case of $\mathbb{Z}_N$ such schemes are called *additively homomorphic* and form the basis of many practically important applications, like the outsourcing of computations as we will see later in this work.

Systematic Design Approach to HE: New Constructions (Chapter 5)

Concerning the design of HE schemes it turns out that GHE schemes are, in a certain sense, bound to the shift-type structure. We prove this by deducing the shift-type structure from the IND-CPA security of GHE schemes. Furthermore, we construct a generic GHE scheme that we prove (under certain assumptions) to be employable as a universal blueprint for GHE schemes, meaning that every such scheme is an instance of the generic scheme. The security of the original scheme is provably equivalent to the security of the respective instance of the generic scheme.

Our blueprint allows us to start with an SMP (SOAP, respectively) instance that we plug into the blueprint, which will then output a GHE scheme. If the SMP (SOAP, respectively) is hard, the resulting GHE scheme is IND-CPA (IND-CCA1, respectively) secure. This gives us an easy way to construct new HE schemes with unique properties:

1. We consider the *k-linear problem* [66,103] as a concrete instance of SMP and construct the first GHE scheme whose IND-CPA security is based on it (see Section 5.2). This problem has the progressive property of getting weaker with larger values for k. In addition, we introduce a *new* k-problem (an instantiation of SOAP) that we prove to have the same progressive property as the k-linear problem. This new problem might be of independent interest as it can be used to construct new cryptographic protocols with unique features. For instance, we use it in our blueprint to build the first GHE scheme on bilinear groups [72] that is provably secure in terms of IND-CCA1.

2. We revisit the notion of additively homomorphic *encryption with a double decryption mechanism* (DD-PKE), which has a master decryption procedure that can decrypt all properly formed ciphertexts by using a special master secret. This type of encryption is generally considered as a practical way to enforce access control in the secure outsourcing of computations in hierachical organisations. Up to now, only two additively homomorphic DD-PKE schemes have been proposed: CS-Lite by Cramer and Shoup [29], and a variant called BCP by Bresson, Catalano and Pointcheval [20].

   We argue in Section 5.3 that the two existing schemes only provide partial solutions for hierarchical organisations. Essentially, this is due to the fact that the master authority, being in possession of the master secret, has no control on the validity of given ciphertexts. We say that the master is unable to "detect invalid ciphertexts", which limits the employment of such schemes in practice. Therefore, we propose the first additively homomorphic DD-PKE scheme which allows the master to detect invalid ciphertexts.

Security Analysis of Existing HE and Impossibility Results (Chapter 6)

Our abstract security characterizations (cf. Chapter 4) can be applied to concrete homomorphic schemes by looking at the according instantiations of SMP and SOAP, respectively (see Section 6.1). For example, several results such as the IND-CPA security of ElGamal [107], the IND-CCA1 security of Damgård's ElGamal [30,61, 84,113] and the recently proved IND-CCA1 security of ElGamal [84] can be easily derived from our characterizations. Additionally, we use our IND-CCA1 characterization to approach a long standing open question by showing Paillier's homomorphic encryption scheme [93] provably secure in terms of IND-CCA1.

Furthermore, we derive two impossibility results from our abstract security characterizations (see Section 6.2):

1. We show that no GHE scheme where the ciphertexts form a linear subspace of $\mathbb{F}^n$ for some prime field $\mathbb{F}$, can be IND-CPA secure. This answers the open question [51] whether using linear codes as ciphertext spaces yield more efficient GHE constructions by showing that secure such schemes cannot exist.

2. We prove that once quantum computers reach maturity, the notion of GHE becomes obsolete. By this we mean that for any given GHE scheme, we can construct a quantum adversary that efficiently breaks its IND-CPA security.

Secure and Efficient Outsourcing of Multiparty Computations (Chapter 7)

Finally, we show that our foundational work on HE (Chapters 4–6) has important consequences on the secure and practically efficient outsourcing of arbitrary multiparty computations. For the first time, we propose a solution that tackles the following problem scenario:

1. A set of n mutually distrusting clients $P_1, \ldots, P_n$ (the number n may change dynamically over time), each having its own public and private key pair, encrypt data under their respective public keys and store these encryptions on a server $\mathcal{C}$.

2. Any *dynamically chosen* function (i.e., the function does not need to be specified at the time data is encrypted) should be computed by $\mathcal{C}$ on the clients' data, while all inputs and intermediate results remain private.

3. Due to the fact that clients are not always online in practice, $\mathcal{C}$ needs the ability to compute these functions *without any interaction* of the clients. In particular, this also concerns the clients' retrieval of results.

4. Once online, individual clients can retrieve the result while the server learns nothing at all.

Previous attempts to construct a protocol for this scenario were either inefficient, relied heavily on client interaction, or required the inputs to be encrypted under the *same* public key—drawbacks making the employment in practice very limited.

Our general-purpose construction avoids all these drawbacks: it is efficient, it requires no user interaction whatsoever (except for data up- and download), and it allows evaluating any dynamically chosen function on inputs encrypted under *different* independent public keys. Our solution assumes the existence of two non-colluding but untrusted servers that jointly perform the computation by means of a cryptographic protocol. This protocol is proven to be secure in the semi-honest model, meaning that all protocol participants follow the protocol description, but may try to gather information about other parties' inputs, intermediate results, or overall outputs just by looking at the transcripts. We demonstrate the applicability of our result in two real-world scenarios from different domains: Privacy-Preserving Face Recognition and Private Smart Metering. Finally, we give a performance analysis of our general-purpose construction to highlight its practicability.

1.1. Related Work

Since the introduction of public-key encryption [35], a long-lasting sequence of papers have been (and are still being) published on the topic of securely outsourcing computations. Most importantly, we mention the significant series of works on IND-CPA secure GHE schemes, starting with ElGamal [36] in 1984, followed by Damgård's ElGamal [30], Paillier [93], and Cramer-Shoup [29] to name just a few (our own scheme of Section 5.3 being the most recent). For a rather complete list of GHE schemes, we refer the reader to [41]. Simultaneously to these developments, beginning in 2008, researchers started to look at relaxations of GHE, where only limited group operations are possible to evaluate [88]. Such relaxations are particularly interesting for certain practical applications, when considering additively HE schemes with plaintext group $\mathbb{Z}_N$ for $N \in \mathbb{N}$ [28]. More importantly, additionally requiring such an additively HE scheme to allow for the evaluation of multiplications as well, led to the design of Fully Homomorphic Encryption (FHE) schemes, allowing the evaluation of virtually any function. The first such scheme was presented by Gentry [52] in 2009, preceding a large body of follow-up works [16–19, 53–55, 57, 58, 85, 110].

To analyze the power and the limits of HE schemes, characterizing them in terms of their security and their design has always been in the focus of active research [29, 40, 52, 59, 60, 65]. In contrast to our work (Chapters 4–5), all previous approaches only encompass small, special-purpose classes of HE schemes and only deal with IND-CPA security (excluding IND-CCA1 security). We stress that such restricted characterizations are insufficient in many ways, for instance in order to achieve our general impossibility results on the use of linear codes or post-quantum hardness assumptions in GHE scheme (Chapter 6).

Currently[1], GHE schemes are still the most efficient schemes among all the existing HE schemes and are therefore of great importance in the design of efficient and secure system for the outsourcing of multiparty computations. As we explained before, additively HE schemes can be employed to realize the evaluation of more complex functions (than just addition operations) if one accepts interactivity with the data owners. All solutions (even the ones that are not based on HE) which have been suggested in the past rely heavily on interactivity with the data owners [16, 18, 19, 51, 56, 73], are too inefficient to be practically relevant [85], reveal the result of

[1]January 14, 2013

the computations to *everyone* [64], or require all inputs to be encrypted under the *same* public key [27]. This clearly separates our solution (Chapter 7) from previous constructions.

2 Prerequisites

We recall standard definitions, notation, and basic facts on groups and elliptic curves, which we will need in the course of this work.

2.1. Definitions and Notation

We write $x \longleftarrow X$ if X is a random variable or distribution and x is to be chosen randomly from X according to its distribution. In the case where X is solely a set, $x \overset{U}{\longleftarrow} X$ denotes that x is chosen uniformly at random from X. If we sample an element x from X by using a specific distribution $\mathcal{D}$, we write $x \overset{\mathcal{D}}{\longleftarrow} X$ (or $x \longleftarrow X$ when there is no doubt about the distribution $\mathcal{D}$). For an algorithm $\mathcal{A}$ we write $x \longleftarrow \mathcal{A}(y)$ if $\mathcal{A}$ outputs x on fixed input y according to $\mathcal{A}$'s distribution. Sometimes, we need to specify the randomness of a probabilistic algorithm $\mathcal{A}$ explicitly. To this end, we interpret $\mathcal{A}$ in the usual way as a deterministic algorithm $\mathcal{A}(y; r)$, which has access to values $r \longleftarrow \mathsf{Rnd}$ that are randomly chosen from some randomness space Rnd.

Moreover, two distribution ensembles $X = \{X_\kappa\}_{\kappa\in\mathbb{N}}$ and $Y = \{Y_\kappa\}_{\kappa\in\mathbb{N}}$ taking values in a finite set S_κ (indexed by a parameter κ) are said to be *computationally indistinguishable*, if for all probabilistic polynomial time (PPT) algorithms $\mathcal{A}$ there exists a negligible function negl such that

$$\mathrm{Adv}_{\mathcal{A}}^{X,Y}(\kappa) := \left| \Pr_{x\longleftarrow X_\kappa}[\mathcal{A}(x) = 1] - \Pr_{y\longleftarrow Y_\kappa}[\mathcal{A}(y) = 1] \right| \leq \mathrm{negl}(\kappa).$$

We sometimes call such an algorithm $\mathcal{A}$ a *distinguisher* and say that the distribution ensemble X is *negligibly close* to Y. We denote the latter by $X \overset{c}{=} Y$. Furthermore, we define the *variational distance* as

$$|X - Y|(\kappa) := \sum_{z\in S_\kappa} \left| \Pr_{x\longleftarrow X_\kappa}[x = z] - \Pr_{y\longleftarrow Y_\kappa}[y = z] \right|.$$

We say that X and Y are *statistically indistinguishable* if and only if $|X - Y|$ is negligible in κ. It is clear that if X and Y are statistically indistinguishable, then in particular they are computationally indistinguishable.

Next, we recall some basic facts about *finite groups* [78]. For a group G, we denote the neutral element by 1, and denote the binary operation on G by "$\cdot$", meaning that G is written in *multiplicative notation*. Throughout this work, we assume all groups G to be finite. If the group operation is commutative, we say that the group is *Abelian*. We recall that a subgroup H of a group G (denoted by $H \leq G$) is said to be *normal* if $g \cdot h \cdot g^{-1} \in H$ for all $g \in G, h \in H$. In particular, this means that if G is an Abelian group, then every subgroup H is normal. Even in the non-Abelian case, if H is a non-trivial, proper normal subgroup of G, and $R \subseteq G$ is a fixed system of representatives of G/H, we have the following fact:

Fact 1 *Let τ denote the restriction to R of the canonical surjection $G \longrightarrow G/H$, $g \longmapsto g \cdot H$. Since R is a system of representatives of G/H, every $g \in G$ can be uniquely written as $g = r \cdot h$ with $r \in R$ and $h \in H$. Therefore, τ is a bijection and there is a group structure on R that is inherited from G/H: For $r, r' \in R$, we define $r \odot r' := \tau^{-1}(\tau(r) \cdot \tau(r'))$. We denote the element in R that corresponds to the neutral element in G/H by $\mathbf{1}$. It is easy to verify that with the defined operation $\odot$, R becomes a* group with neutral element $\mathbf{1}$. *In addition, we know that $R \cap H = \mathbf{1}$ as $R \subseteq G$ is a system of representatives of G/H.*

If $f : G \to G'$ is a group homomorphism between two groups G and G', we write $\mathrm{dom}(f) = G$ for the *domain* of f, $\mathrm{im}(f)$ for its *image*, and $\ker(f) := \{g \in G \mid f(g) = 1\}$ for the *kernel* of f. Note that $\mathrm{im}(f)$ is a subgroup of G' and that $\ker(f)$ is a subgroup of G. In addition, we write $f|_H$ for the *restriction* of f to a subgroup $H \subseteq G$, meaning that $f|_H : H \to G'$ with $f|_H(h) := f(h)$ for all $h \in H$. We remark that $f|_H$ is a group homomorphism itself. If f is surjective, we write $f^{-1}(g') := \{g \in G \mid f(g) = g'\}$ for the *preimage* of g' under f for $g' \in G'$. Surjective group homomorphisms are called *group epimorphisms*, while injective homomorphisms are called *group monomorphisms*.

By a *description* of a group G we mean an efficient sampling algorithm (where sampling is denoted by $g \longleftarrow G$), the neutral element $1 \in G$, an efficient algorithm for performing the group operation on G, and one for the inversion of group elements. We abuse notation and write G both for the description and for the group itself. Furthermore, for elements $g_1, \ldots, g_k \in G$, we write $\langle g_1, \ldots, g_k \rangle$ for the subgroup (of G) generated by $g_1, \ldots, g_k$.

2.2. Elliptic Curves over Rings

Let R be a commutative unital ring with R^* denoting its group of units. We say that for $a, b \in R$ the equation

$$E : y^2z = x^3 + axz^2 + bz^3 \tag{2.1}$$

defines an *elliptic curve E over R* if the *discriminant* $\Delta := 16(4a^3+27b^2)$ is a unit in R, so $\Delta \in R^*$. For all triples $(x, y, z) \in R^3$ that satisfy (2.1), we say that (x, y, z) is *equivalent to* (x', y', z') if there exists $\nu \in R^*$ such that $\nu x = x', \nu y = y'$ and $\nu z = z'$. Indeed this defines an equivalence relation (denoted by $\sim$) on all such triples and we denote equivalence classes by $(x : y : z)$. This relation allows us to define the *set of R-valued points* of E (denoted by $E(R)$) as the set of all equivalence classes $(x : y : z)$ with $x, y, z \in R$ satisfying (2.1) such that the *ideal I generated by x, y, z* is R, that is $I = \{rx + sy + tz \mid r, s, t \in R\} = R$.

It can be shown (see [81, Section 3]) that the usual chord and tangent process on elliptic curves over fields (cf. [106, Chapter III]) yields a group law on $E(R)$ with identity element $\mathcal{O}_\infty := (0 : 1 : 0)$ *if* R has the property that every projective R-module of rank one is free. For our work, it suffices to consider this case, since we will only work over finite rings which have this property. Therefore, we will from now on restrict our attention to finite rings R.

It should be noted that there are explicit and efficient formulae to perform the group law on $E(R)$ [42,46,81]). Furthermore, we recall that the Chinese Remainder Theorem on $\mathbb{Z}_N$ (where $N = pq$ is the product of two odd, distinct primes p and q) implies natural reduction maps from $E(\mathbb{Z}_N)$ to $E(\mathbb{Z}_p)$ and $E(\mathbb{Z}_q)$. It follows that $E(\mathbb{Z}_N) \cong E(\mathbb{Z}_p) \times E(\mathbb{Z}_q)$ (see [46]).

There are a few other facts in the case where $R = \mathbb{Z}_{N^2}$ that are of particular interest to us, which follow from the p-adic theory of elliptic curves, and we refer the reader to [46] and [106] for details:

1. $\#E(\mathbb{Z}_{N^2}) = N\#E(\mathbb{Z}_N) = N\#E(\mathbb{Z}_p)\#E(\mathbb{Z}_q)$.
2. $P_i := (Ni : 1 : 0) \in E(\mathbb{Z}_{N^2})$ with $mP_i = P_{mi}$ for all $m \in \mathbb{Z}_N$.
3. $NP_1 = \mathcal{O}_\infty$.

As in the case when working with elliptic curves over fields, we can define bilinear mappings, called pairings [44], on elliptic curves over $\mathbb{Z}_{N^2}$ by taking the pairings over the respective curves over $\mathbb{Z}_p$ and $\mathbb{Z}_q$ (using the Chinese Remainder Theorem) and then clueing their outputs together [48]. If the resulting pairing is efficiently computable given the factorization of N, we call such an elliptic curve over $\mathbb{Z}_{N^2}$ *pairing-friendly*. By the definition of such pairings, we have to evaluate the Chinese Remainder Theorem, which is only possible when the factorization of N is known [48].

2.3. Computationally Hard Problems

We describe computational problems P through experiments $\mathbf{Exp}^{\mathrm{P}}_{\mathcal{A},G}(\kappa)$ for given probabilistic algorithms $\mathcal{A}$ and G that run in time polynomial in a given parameter κ. The output of $\mathbf{Exp}^{\mathrm{P}}_{\mathcal{A},G}(\kappa)$ is always defined to be a single bit. We then say that *problem* P *is hard (relative to G)* if for all probabilistic polynomial time (PPT) algorithms $\mathcal{A}$ there exists a negligible function negl such that

$$\left|\Pr\left[\mathbf{Exp}^{\mathrm{P}}_{\mathcal{A},G}(\kappa) = 1\right] - \frac{1}{2}\right| \leq \mathrm{negl}(\kappa).$$

One of the most famous problems is the *Decisional Diffie-Hellman Problem* (DDH) which is defined as follows: Let GGen be a PPT algorithm (in κ) that outputs a cyclic group G with generator $g \in G$ of order p. We consider the following experiment for GGen and PPT adversary $\mathcal{A}$:

Experiment $\mathbf{Exp}^{\mathrm{DDH}}_{\mathcal{A},\mathsf{GGen}}(\kappa)$:

1. $(G, g) \longleftarrow \mathsf{GGen}(\kappa)$
2. Choose $b \overset{U}{\longleftarrow} \{0,1\}$ and $a, b \overset{U}{\longleftarrow} \mathbb{Z}_p$. If $b = 1$: Compute $z := g^{ab}$. Otherwise: $z := g^c$ for $c \overset{U}{\longleftarrow} \mathbb{Z}_p$
3. $d \longleftarrow \mathcal{A}(G, g, z)$ where $d \in \{0,1\}$
4. The output of the experiment is defined to be 1 if $d = b$ and 0 otherwise.

This experiment defines the Decisional Diffie-Hellman Problem. Similarly, this problem is defined on elliptic curves over $\mathbb{Z}_{N^2}$ for an RSA-modulus N: Given a random point Q of large order k (meaning that k has about the same size as N), points rQ, sQ and tQ $(r, s, t \in \mathbb{Z}_k)$, it is computationally infeasible to decide whether $t = rs \bmod k$ or not. We denote this DDH-variant on such curves by $\mathrm{DDH}_{\mathbb{Z}_{N^2}}$. We state that if the factorization of N is not known, the *Decisional Diffie-Hellman Problem* is believed to be hard for elliptic curves over $\mathbb{Z}_{N^2}$ [46].

3 Homomorphic Encryption (HE)

This chapter gives an introduction to homomorphic encryption (HE) schemes, the fundamental concept of this thesis. Roughly speaking, such encryption schemes allow for the evaluation of certain functionalities over encrypted data without being able to decrypt. Therefore, if Alice $\mathcal{A}$ wants to outsource the computation of a functionality (that is supported by the HE scheme) on $\mathcal{A}$'s sensitive data to Bob $\mathcal{B}$, she can do so by simply encrypting all her data and sending everything to $\mathcal{B}$. $\mathcal{B}$ can then use the homomorphic property of the encryption scheme to perform the computations in such a way that all of $\mathcal{A}$'s data remains hidden (encrypted) from $\mathcal{B}$.

3.1. Group Homomorphic Encryption (GHE)

We recall the notion of public-key *group homomorphic* encryption (cf. [65]), which roughly can be described as usual public-key encryption where the decryption algorithm is a group homomorphism.

Definition 1 (Group Homomorphic Encryption (GHE)) *A public-key encryption scheme* $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *is called* group homomorphic*, if for every output* $(\mathsf{pk}, \mathsf{sk})$ *of* $\mathsf{KeyGen}(\kappa)$*, the plaintext space* $\mathcal{P}$ *and the ciphertext space* $\widehat{\mathcal{C}}$ *are non-trivial (multiplicatively written) groups such that*

- *the* set of all encryptions $\mathcal{C} := \{\mathsf{Enc}_{\mathsf{pk}}(m; r) \mid m \in \mathcal{P}, r \in \mathsf{Rnd}\}$ *is a non-trivial subgroup of* $\widehat{\mathcal{C}}$
- *the decryption* $\mathsf{Dec}_{\mathsf{sk}}$ *is a group homomorphism on* $\mathcal{C}$*, i.e.*

$$\mathsf{Dec}_{\mathsf{sk}}(c \cdot c') = \mathsf{Dec}_{\mathsf{sk}}(c) \cdot \mathsf{Dec}_{\mathsf{sk}}(c'), \text{ for all } c, c' \in \mathcal{C}.$$

There exist some GHE schemes [30] where the decryption algorithm outputs an error $\perp$ on inputs in $\widehat{\mathcal{C}} \setminus \mathcal{C}$.

Definition 2 (GHE Scheme with Decryption Error) *For a* GHE *scheme* $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$*, we say that* $\mathcal{E}$ has a decryption error*, if the decryption algorithm* $\mathsf{Dec}_{\mathsf{sk}}(c)$ *outputs the error symbol* $\perp$ *iff* $c \in \widehat{\mathcal{C}} \setminus \mathcal{C}$*.*

Since GHE schemes are public-key encryption schemes by definition, we have the standard security notions of *indistinguishability under chosen-plaintext attack* (IND-CPA), *indistinguishability under (non-adaptive) chosen-ciphertext attack* (IND-CCA1) and *indistinguishability under adaptive chosen-ciphertext attack* (IND-CCA2). We recall these three notions along the lines of [11, Definition 2.1] and afterwards explain their role in the group homomorphic case.

For $i \in \{1, 2\}$, let $\mathcal{O}_i(\cdot)$ denote an oracle function, where $\mathcal{O}_i(\cdot) = \varepsilon$ means that the oracle always returns the empty string ε on any input. Now, for a GHE scheme $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$, atk $\in \{\text{cpa}, \text{cca1}, \text{cca2}\}$, a given algorithm $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ and security parameter κ, we consider the following experiment:

Experiment $\mathbf{Exp}^{\text{ind-atk}}_{\mathcal{A},\mathsf{KeyGen}}(\kappa)$:

1. $(\mathsf{pk}, \mathsf{sk}) \longleftarrow \mathsf{KeyGen}(\kappa)$
2. $(m_0, m_1, s) \longleftarrow \mathcal{A}_1^{\mathcal{O}_1(\cdot)}(\mathsf{pk})$ where $m_0, m_1 \in \mathcal{P}$ and s is a state of $\mathcal{A}_1$
3. Choose $b \overset{U}{\longleftarrow} \{0, 1\}$ and compute the challenge $c \longleftarrow \mathsf{Enc}_{\mathsf{pk}}(m_b)$
4. $d \longleftarrow \mathcal{A}_2^{\mathcal{O}_2(\cdot)}(m_0, m_1, s, c)$ where $d \in \{0, 1\}$
5. The output of the experiment is defined to be 1 if $d = b$ and 0 otherwise,

$$\text{whereas} \begin{cases} \text{if} & \text{atk} = \text{cpa} & \text{then} & \mathcal{O}_1(\cdot) = \varepsilon & \text{and} & \mathcal{O}_2(\cdot) = \varepsilon \\ \text{if} & \text{atk} = \text{cca1} & \text{then} & \mathcal{O}_1(\cdot) = \mathsf{Dec}_{\mathsf{sk}}(\cdot) & \text{and} & \mathcal{O}_2(\cdot) = \varepsilon \\ \text{if} & \text{atk} = \text{cca2} & \text{then} & \mathcal{O}_1(\cdot) = \mathsf{Dec}_{\mathsf{sk}}(\cdot) & \text{and} & \mathcal{O}_2(\cdot) = \mathsf{Dec}_{\mathsf{sk}}(\cdot). \end{cases}$$

If atk = cca2, we further require that $\mathcal{A}_2$ is not allowed to ask its oracle to decrypt the challenge ciphertext c.

We say that $\mathcal{E}$ is $\mathsf{IND\text{-}ATK}$ *secure* (relative to KeyGen) if the advantage

$$\left| \Pr\left[\mathbf{Exp}^{\text{ind-atk}}_{\mathcal{A},\mathsf{KeyGen}}(\kappa) = 1 \right] - \frac{1}{2} \right| \text{ is negligible for all PPT algorithms } \mathcal{A},$$

where ATK $\in$ {CPA, CCA1, CCA2}. Bellare et al. [11] show that IND-CCA2 is strictly stronger than IND-CCA1, which in turn is strictly stronger than IND-CPA. For reasons of completeness, we prove the following well-known result.

Lemma 1 (No IND-CCA2 Security for GHE Schemes) *Every* GHE *scheme* $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *is insecure in terms of IND-CCA2.*

Proof: On input the public key pk, the adversary $\mathcal{A}_1$ outputs two randomly chosen plaintexts $m_0, m_1 \in \mathcal{P}$ with $m_0 \neq m_1$. The challenger chooses a random bit $b \in \{0,1\}$ and computes the challenge ciphertext $c \longleftarrow \mathsf{Enc}_{\mathsf{pk}}(m_b)$. Upon receiving the challenge, $\mathcal{A}_2$ computes $c_i \longleftarrow c \cdot \mathsf{Enc}_{\mathsf{pk}}(m_i^{-1})$ for $i \in \{0,1\}$, and asks the decryption oracle for the decryptions of c_0 and c_1. By definition, one of these decryptions is 1, and $\mathcal{A}_2$ outputs the index $d \in \{0,1\}$ of the decryption that corresponds to 1. Therefore, the advantage of $\mathcal{A}$ in the IND-CCA2 game is $\frac{1}{2}$, which is non-negligible. □

Due to this Lemma, we know that IND-CCA1 is the strongest of the three security notions for GHE schemes. We remark that there exist three additional, standard security notions: *Non-malleability* with respect to CPA, CCA1 and CCA2. For details on these, we refer to [11] and note that, for obvious reasons, no GHE scheme can be secure in terms of these notions. Therefore, we do not consider these non-malleability notions. Also, we note that non-standard variants, as defined in [22,100], lie outside of the scope of this paper.

Already from the definition of GHE, we can deduce some first facts (see [65] for similar thoughts), but before we do this, we require the following definition.

Definition 3 (The Set of All Encryptions of a given Plaintext) *For a given* GHE *scheme* $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *and message* $m \in \mathcal{P}$*, we define the* set of all (fresh) encryptions of m *as*

$$\mathcal{C}_m = \{c \in \mathcal{C} \mid \mathsf{Dec}_{\mathsf{sk}}(c) = m\}\,.$$

With this definition, the IND-CPA security of a given GHE scheme $\mathcal{E}$ is equivalent to saying that the distribution on $\mathcal{C}_{m_0}$ (induced by the encryption algorithm $\mathsf{Enc}_{\mathsf{pk}}(m_0)$) is negligibly close to the (induced) distribution on $\mathcal{C}_{m_1}$ for any two messages m_0 and m_1, i.e., $\mathcal{C}_{m_0} \overset{c}{=} \mathcal{C}_{m_1}$ [62, Ch. 5.2].

Lemma 2 *For a* GHE *scheme* $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$*, we have*

1. $\mathcal{C}_m = \mathsf{Enc}_{\mathsf{pk}}(m; r) \cdot \mathcal{C}_1$ *for all* $m \in \mathcal{P}$ *and all* $r \in \mathsf{Rnd}$*,*

2. $\mathcal{C}_1$ *is a proper* normal *subgroup of* $\mathcal{C}$ *such that* $|\mathcal{C}_1| = |\mathcal{C}_m|$ *for all* $m \in \mathcal{P}$*.*

It follows that the set $\{\mathsf{Enc}_{\mathsf{pk}}(m; r) \mid m \in \mathcal{P}\}$ *for a fixed* r *is a system of representatives of* $\mathcal{C}/\mathcal{C}_1$*.*

Proof:

1. We fix a random r and $m \in \mathcal{P}$. Let $c \in \mathcal{C}_m$ and set $c_1 := c \cdot \mathsf{Enc}_{\mathsf{pk}}(m, r)^{-1}$. Then, $\mathsf{Dec}_{\mathsf{sk}}(c_1) = m \cdot m^{-1} = 1$, i.e., $c_1 \in \mathcal{C}_1$. Therefore, $c = \mathsf{Enc}_{\mathsf{pk}}(m, r) \cdot c_1 \in \mathsf{Enc}_{\mathsf{pk}}(m, r) \cdot \mathcal{C}_1$. Conversely, let $c_1 \in \mathcal{C}_1$. Then, $\mathsf{Dec}_{\mathsf{sk}}(\mathsf{Enc}_{\mathsf{pk}}(m, r) \cdot c_1) = m \cdot 1 = m$, i.e., $\mathsf{Enc}_{\mathsf{pk}}(m, r) \cdot c_1 \in \mathcal{C}_m$. The first statement of the lemma follows immediately.

2. With respect to the second claim, we show by contradiction that $\mathcal{C}_1 \neq \mathcal{C}$. Therefore, assume that $\mathcal{C}_1 = \mathcal{C}$. Since the decryption $\mathsf{Dec}^*_{\mathsf{sk}}$ is surjective, this means that $\mathcal{P}$ is a trivial group, which contradicts the definition of a homomorphic scheme. Now, by looking at the definition of $\mathcal{C}_1$, we see that $\mathcal{C}_1 = \ker(\mathsf{Dec}^*_{\mathsf{sk}})$. Therefore, $\mathcal{C}_1$ is a *normal* subgroup of $\mathcal{C}$ (see [79, p. 13]). The last claim is an immediate consequence of the equality $\mathcal{C}_m = \mathsf{Enc}_{\mathsf{pk}}(m, r) \cdot \mathcal{C}_1$.

□

As a final remark of this section, we want to point out that a GHE scheme is called *additively homomorphic*, if the plaintext group is $\mathbb{Z}_N$ for a given natural number $N \in \mathbb{N}$.

3.2. Somewhat Homomorphic Encryption (SWHE) and Fully Homomorphic Encryption (FHE)

3.2.1. Definitions

Recall that the notion of GHE requires an encryption scheme to be able to evaluate an *unbounded* number of group operations. This requirement can be relaxed as described in the next definition (following [69, Definition 5] and [52, Definition 1]).

Definition 4 (Somewhat Homomorphic Encryption (SWHE)) *A public-key encryption scheme $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ is called* homomorphic for a set of circuits $\mathbb{C} = \mathbb{C}[\kappa]$ *(that depends on the security parameter κ), if there exists a PPT algorithm* Eval *(that outputs a ciphertext and takes as input public keys* pk *from the output of* KeyGen*, circuits $C \in \mathbb{C}(\kappa)$ and ciphertexts $(c_1, \ldots, c_r)$ with $c_i \longleftarrow \mathsf{Enc}_{\mathsf{pk}}(m_i)$ for some $m_i \in \mathcal{P}$, $i = 1, \ldots, r$) such that for every output $(\mathsf{pk}, \mathsf{sk})$ of $\mathsf{KeyGen}(\kappa)$ it holds that (*correctness condition*)*

$$\mathsf{Dec}_{\mathsf{sk}}(\mathsf{Eval}_{\mathsf{pk}}(C, c_1, \ldots, c_r)) = C(m_1, \ldots, m_r),$$

except with negligible probability (in κ) over the random coins in Eval*. We also call such schemes* Somewhat Homomorphic Encryption (SWHE) *schemes.*

The security notions for SWHE schemes are exactly the same as for GHE schemes, while IND-CPA is the minimal security requirement and IND-CCA1 is the strongest.

Sometimes it is required that, even if the secret key is known, the output of Eval does not reveal any information about the circuit that it evaluates, except for the output value of that circuit. This requirement is captured in the following definition.

Definition 5 (Circuit Privacy) *An* SWHE *scheme $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$, that is homomorphic for a set $\mathbb{C} = \mathbb{C}[\kappa]$ of circuits, is said to be* (computationally) circuit-private on $\mathbb{C}$, *if for every keypair* $(\mathsf{pk}, \mathsf{sk}) \longleftarrow \mathsf{KeyGen}(\kappa)$*, any circuit* $C \in \mathbb{C}$*, and any fixed tuple of fresh encryptions* $(c_1, \ldots, c_r)$ *with* $c_i \longleftarrow \mathsf{Enc}_{\mathsf{pk}}(m_i)$ *for plaintexts* $m_i \in \mathcal{P}$ *and* $i = 1, \ldots, r$*, the following distributions (over the random coins in* Enc *and* Eval*) are (computationally) indistinguishable:*

$$\mathsf{Enc}_{\mathsf{pk}}(C(m_1, \ldots, m_r)) \overset{c}{=} \mathsf{Eval}_{\mathsf{pk}}(C, c_1, \ldots, c_r).$$

Now, informally, an SWHE scheme that is homomorphic for *all* circuits is called a *Fully Homomorphic Encryption* (FHE) scheme. But to rule out trivial FHE schemes $\mathcal{E}$, e.g., where Eval simply outputs its input circuit C together with its input ciphertexts and Dec takes circuits C as input as well and simply outputs the evaluation of C on the decryptions of the plugged-in ciphertexts, we require the additional property of *compactness* (cf. [51, Definition 2.1.2]). Informally this means that the size of the output of Eval does not depend on the size of the circuit it evaluates.

Definition 6 (Compactness) *Let* $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ *be an encryption scheme that is homomorphic for a set of circuits* $\mathbb{C} = \mathbb{C}[\lambda]$*.* $\mathcal{E}$ *is called* compact*, if* Dec *can be expressed as a circuit of size at most* $p(\kappa)$ *for some polynomial* p*.*

With this definition in mind, we can formalize the notion of FHE:

Definition 7 (Fully Homomorphic Encryption (FHE)) *An encryption scheme* $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ *that is homomorphic for all circuits and compact is called* fully homomorphic*.*

Moreover, an FHE *scheme is called* circuit-private*, if it is circuit-private for* all *circuits.*

In between SWHE schemes and FHE schemes, there are schemes that are homomorphic for *all circuits up to a certain depth.* We call such schemes *Leveled* FHE schemes.

Definition 8 (Leveled FHE [52]) *We call a family* $\{\mathcal{E}^{(d)} \mid d \in \mathbb{Z}^+\}$ *of schemes* leveled fully homomorphic*, if they all use the same decryption circuit,* $\mathcal{E}^{(d)}$ *is homomorphic for all circuits of depth at most* d*, and the computational complexity of* $\mathcal{E}^{(d)}$*'s algorithms is polynomial in* κ*,* d*, while* $\mathsf{Eval}^{(d)}$*'s runtime additionally is polynomial in the size of its input circuit* C*.*

3.2.2. Gentry's Bootstrapping Technique

We briefly want to recall some facts about Gentry's bootstrapping technique [52] on how to construct FHE schemes. Roughly speaking, Gentry constructs a homomorphic encryption scheme for circuits of any depth from an underlying encryption scheme that is homomorphic for "just a little more than its own decryption circuit". We formalize the term in double quotes momentarily (see also [52, Definition 4]), but first need to do some more definitional work.

Definition 9 *Let $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ be an encryption scheme in which* Dec *is implemented by a circuit that does only depend on the security parameter κ. For every output $(\mathsf{pk}, \mathsf{sk})$ of $\mathsf{KeyGen}(\kappa)$, we let Γ be a set of gates with inputs and output in plaintext space $\mathcal{P}$ including the identity gate (input and output are the same). For gate $g \in \Gamma$, the g-*augmented decryption circuit *consists of a g-gate connecting multiple copies of* Dec *(the number of copies equals the number of inputs to g), where* Dec *takes the secret key* sk *and a ciphertext as input formatted as elements of $\mathcal{P}^{\ell(\kappa)}$, where $\ell(\kappa)$ is some polynomial in κ. We denote the set of all g-augmented decryption circuits, $g \in \Gamma$, by $\mathsf{Dec}[\kappa](\Gamma)$.*[1]

The most important property of an encryption scheme to be of any use in Gentry's approach is that of *bootstrappability.*

Definition 10 *Let $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ be a homomorphic encryption scheme for a set of circuits $\mathbb{C} = \mathbb{C}[\kappa]$. $\mathcal{E}$ is called* bootstrappable for a set of gates Γ*, if* $\mathsf{Dec}[\kappa](\Gamma) \subseteq \mathbb{C}[\kappa]$ *for all security parameters κ.*

A boostrappable SWHE scheme is already sufficient to construct FHE schemes as the following result of [52] shows:

Theorem 1 *There is an efficient and explicit transformation that for any given bootstrappable scheme $\mathcal{E}$ for a set of gates Γ and parameter $d = d(\kappa)$ outputs another encryption scheme $\mathcal{E}^{(d)}$ that is*

1. *compact and whose decryption circuit is identical to that of $\mathcal{E}$*

2. *homomorphic for all circuits with gates in Γ of depth at most d.*

In the transformation of Theorem 1, there are some particular facts about the resulting scheme $\mathcal{E}^{(d)}$ that will become very important in the course of this work (see [52] for more details).

Fact 2 1. *The plaintext space $\mathcal{P}$ of $\mathcal{E}^{(d)}$ is the same as that of $\mathcal{E}$.*

2. *The key generation algorithm of $\mathcal{E}^{(d)}$ uses the key generator* KeyGen *of $\mathcal{E}$ $(d{+}1)$-times to produce $d+1$ public and secret key pairs $(\mathsf{pk}_i, \mathsf{sk}_i)$, $i = 0, \ldots, d$. Let $\mathsf{sk}_{i1}, \ldots, \mathsf{sk}_{i\ell}$ be the representation of sk_i as elements of $\mathcal{P}$ with $\ell = \ell(\kappa)$ as in Definition 9. The key generator of $\mathcal{E}^{(d)}$ then computes $\overline{\mathsf{sk}_{ij}} \longleftarrow \mathsf{Enc}_{\mathsf{pk}_{i-1}}(sk_{ij})$ for $i = 1, \ldots, d$ and $j = 1, \ldots, \ell$, and outputs the secret key $\mathsf{sk}^{(d)} := sk_0$, and public key*
$$\mathsf{pk}^{(d)} := \left((\mathsf{pk}_i)_{i=1,\ldots,d}, (\overline{\mathsf{sk}_{ij}})_{\substack{i=1,\ldots,d \\ j=1,\ldots,\ell}} \right).$$

[1]We sometimes make Dec's dependency on κ obvious by writing $\mathsf{Dec}[\kappa]$ instead of just Dec.

3. *Encryption of a message $m \in \mathcal{P}$ in $\mathcal{E}^{(d)}$ is done by computing a ciphertext $c \longleftarrow \mathsf{Enc}_{\mathsf{pk}_d}(m)$, i.e., an encryption of m under pk_d by using the encryption algorithm Enc of $\mathcal{E}$.*

Concerning the security of the resulting scheme $\mathcal{E}^{(d)}$, Gentry proves the following theorem.

Theorem 2 *Let $\mathcal{E}$ be a bootstrappable scheme for a set of gates Γ. For all parameters $d = d(\kappa)$, we have that the output $\mathcal{E}^{(d)}$ of the transformation from Theorem 1 applied to $\mathcal{E}$ and d is IND-CPA secure if $\mathcal{E}$ is.*

Unfortunately, the resulting scheme $\mathcal{E}^{(d)}$ after applying the transformation is not yet a “pure” FHE scheme as it is only homomorphic for all circuits with gates in Γ of depth at most d (i.e., it is leveled fully homomorphic). However, in [52], Gentry shows how to modify the previously described technique to get “pure” FHE schemes (see [51, Section 4.3] for details).

Theorem 3 *There is an efficient and explicit transformation that for any given bootstrappable scheme $\mathcal{E}$ for a* universal[2] *set of gates Γ outputs another encryption scheme $\mathcal{E}^*$ that is fully homomorphic and whose decryption circuit is identical to that of $\mathcal{E}$.*

Proof: In the key generation step of Theorem 1 (this is step 2), $\mathcal{E}^*$ uses the key generator KeyGen of $\mathcal{E}$ only once (instead of $(d+1)$-times) to compute a key pair $(\mathsf{pk}, \mathsf{sk})$ and outputs the secret key $\mathsf{sk}^* := \mathsf{sk}$ and public key $\mathsf{pk}^* := (\mathsf{pk}, \overline{\mathsf{sk}_1}, \dots, \overline{\mathsf{sk}_\ell})$ where $\overline{\mathsf{sk}_i} \longleftarrow \mathsf{Enc}_{\mathsf{pk}}(\mathsf{sk}_i)$ and $\mathsf{sk}_1, \dots, \mathsf{sk}_\ell$ is the representation of sk as elements of $\mathcal{P}$. This is the only modification and the rest works exactly as in the transformation of Theorem 1. $\square$

There is a similar proof of security for the resulting scheme $\mathcal{E}^*$ as in the leveled FHE case. However, in the “pure” FHE case, there is the additional assumption that the underlying bootstrappable scheme $\mathcal{E}$ is *circular secure*:

Definition 11 (Circular Security [52]) *For a bootstrappable encryption schemes $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$, we consider the following experiment for a given algorithm $\mathcal{A}$ and security parameter κ:*

Experiment $\mathbf{Exp}^{\text{circular}}_{\mathcal{A},\mathsf{KeyGen}}(\kappa)$:

1. *Compute* $(\mathsf{pk}, \mathsf{sk}) \longleftarrow \mathsf{KeyGen}(\kappa)$

[2]This is a set of gates by which any circuit can be expressed, e.g., a NAND-gate when considering boolean circuits.

2. *Choose* $b \overset{U}{\longleftarrow} \{0,1\}$. *If* $b = 0$, *then compute* $\overline{\mathsf{sk}_j} \longleftarrow \mathsf{Enc}_{\mathsf{pk}}(\mathsf{sk}_j)$ *for all* $j = 1,\ldots,\ell$ *where* $\mathsf{sk}_1,\ldots,\mathsf{sk}_\ell$ *is the representation of* sk *as elements of* $\mathcal{P}$ *with* $\ell = \ell(\lambda)$ *as in Definition 9. If* $b = 1$, *then compute* $\overline{\mathsf{sk}_j}$ *as encryptions of some fixed element* $\mathbf{0} \in \mathcal{P}$, *unrelated to* pk, *for all* $j = 1,\ldots,\ell$

3. $d \longleftarrow \mathcal{A}\left(\mathsf{pk}, \overline{\mathsf{sk}_1}, \ldots, \overline{\mathsf{sk}_\ell}\right)$ *where* $d \in \{0,1\}$

4. *The output of the experiment is defined to be 1 if* $d = b$ *and 0 else.*

This experiment defines circular security *for bootstrappable encryption schemes* $\mathcal{E}$.

Gentry [52] proves the following theorem about the security of the resulting FHE scheme of Theorem 3:

Theorem 4 *Let* $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ *be a bootstrappable scheme for a universal set of gates* Γ. *We have that the resulting scheme* $\mathcal{E}^*$ *of the transformation from Theorem 3 applied to* $\mathcal{E}$ *is IND-CPA secure, if* $\mathcal{E}$ *is circular secure.*

3.3. Secure Two-Party Computation based on Additively Homomorphic Encryption

Homomorphic encryption, as defined in the previous two sections, allows for the secure outsourcing of computations of certain functionalities (or all functionalities in the case of FHE) from a *single* party to another party. Several applications, however, require the ability to securely outsource computations between *multiple* parties. We say that we want to securely outsource *multiparty computations.* To this end, it is often useful to consider the topic of *Secure Multiparty Computation* (SMC). SMC enables a set of parties to jointly evaluate any kind of functionality on their respective inputs while keeping these inputs hidden throughout the computations. In general, SMC can be realized by using garbled circuits [115], secret sharing [26], or homomorphic encryption [28]. In this work, we will only deal with SMC based on homomorphic encryption and refer the interested reader for the other approaches to the relevant literature in the field [26, 115]. Moreover, for the purposes of this thesis it suffices to restrict our attention to the following two-party setting (while pointing to [28] for a more general treatment):

Alice $\mathcal{A}$ holds a public and secret key pair $(\mathsf{pk}, \mathsf{sk})$ of some IND-CPA secure additively homomorphic encryption scheme. Furthermore, $\mathcal{B}$ knows two encryptions $c_a = \mathsf{Enc}_{\mathsf{pk}}(a), c_b = \mathsf{Enc}_{\mathsf{pk}}(b)$ (of unknown plaintexts a and b) and is to compute the evaluation on a and b of an arbitrary 2-input function f.[3] Here, we assume that the function f is represented as an arithmetic circuit over the plaintext space

[3]The generalization to more than two inputs is straightforward and we refer to [28] for details.

of the additively homomorphic encryption scheme. Obviously, $\mathcal{B}$ cannot compute the function f on his own if f involves multiplication gates, since the encryption scheme is additively homomorphic and only allows for the evaluation of addition gates. Therefore, whenever a multiplication gate needs to be evaluated, $\mathcal{B}$ requires the help of $\mathcal{A}$ in an interactive protocol: First, $\mathcal{B}$ chooses two random plaintexts r_a and r_b, computes $c_a \cdot \mathsf{Enc}_{\mathsf{pk}}(-r_a)$ and $c_b \cdot \mathsf{Enc}_{\mathsf{pk}}(-r_b)$, and sends this to $\mathcal{A}$. $\mathcal{A}$ knows the secret key sk and decrypts the received information. Since the original plaintexts are blinded with the random values r_a and r_b, $\mathcal{A}$ does not see what $\mathcal{B}$ wants to compute. Then, $\mathcal{A}$ performs the multiplication in the clear, encrypts the result (denoted by c_d), and sends it back to $\mathcal{B}$. $\mathcal{B}$, in turn, computes $c_{-r_a r_b} = \mathsf{Enc}_{\mathsf{pk}}(-r_a r_b)$ and

$$c = c_d \cdot c_a^{r_b} \cdot c_b^{r_a} \cdot c_{-r_a r_b}.$$

This is an encryption of $a \cdot b$ as the decryption of c shows:

$$\mathsf{Dec}_{\mathsf{sk}}(c) = (a - r_a) \cdot (b - r_b) + r_b a + r_a b - r_a r_b = ab.$$

Concerning the security of the protocol, it can be shown [28] that neither $\mathcal{A}$ nor $\mathcal{B}$ learn anything about the original inputs a and b, when they behave according to the protocol description. More formally, the security of the overall protocol is defined in the "real-vs.-ideal" framework [62, Ch. 7]: there is an ideal model where all computations are performed via an additional trusted party and it is then shown that all adversarial behavior in the real model (where there is no trusted party) can be simulated in the ideal model. We say that the protocol is *secure in the semi-honest model.* Due to the Composition Theorem for the semi-honest model [62, Theorem 7.3.3], we can deal with the addition and multiplication protocols individually. All of this is standard material in any introductory course on cryptography and we refer to [62] for details.

4 Characterization of Homomorphic Encryption

Homomorphic encryption schemes form one of the most important cryptographic primitives for the secure outsourcing of computations. Understanding their inherent structures is therefore a major research goal in the analysis of protocols for the secure outsourcing of computations. We provide the first universal framework for group homomorphic encryption schemes that allows us to characterize their IND-CPA and IND-CCA1 security through certain abstract subgroup problems. Later in this thesis, we will see how our characterizations can be used to derive impossibility results and to analyze the security of existing schemes. Moreover, we show that our framework also applies to the current approach on the design of FHE schemes.

4.1. Subgroup and Subset Problems

4.1.1. The Splitting Problem and the Subgroup Membership Problem

In [59], Gjøsteen introduces a computational problem, called *Splitting Problem*, together with a related decisional problem, called *Subgroup Membership Problem*. We recall these two problems, while extending Gjøsteen's original definitions to a more general setting.

Let $\widehat{G}$ be a finite (not necessarily Abelian) group, G a non-trivial subgroup of $\widehat{G}$, H a non-trivial, proper normal subgroup of G, and $R \subseteq G$ a fixed system of representatives of G/H. Additionally to the standard sampling algorithms contained in the description of a group, we assume here that there is an algorithm that allows for the efficient sampling from $G \setminus H$.

We recall that every $g \in G$ can be uniquely written as $g = r \cdot h$ with $r \in R$ and $h \in H$ and that there is a natural group structure on R that is inherited from G/H (cf. Fact 1). Moreover, we notice that the following map is a bijection:

$$R \times H \longrightarrow G \text{ given by } (r, h) \longmapsto r \cdot h.$$

We denote its *inverse* by σ and call σ the *splitting map* for (G, H, R).

Informally, the *Splitting Problem* (SP) for (G, H, R) is to compute $\sigma(g)$ for a randomly given $g \in G$. Before we give a formal definition of SP, we note that our definition extends Gjøsteen's in that it considers a system of representatives that need not be a subgroup of G, while Gjøsteen always assumes it to be a subgroup. In addition, we allow G to be a non-Abelian group, while Gjøsteen only considers the Abelian case. Now let GGen be a PPT algorithm that takes a security parameter κ as input and outputs (G, H, R) where G, H and R are descriptions of the respective groups defined above. Consider the following experiment for given algorithms GGen, $\mathcal{A}$ and parameter κ:

Experiment $\mathbf{Exp}^{\mathrm{SP}}_{\mathcal{A},\mathsf{GGen}}(\kappa)$:

1. $(G, H, R) \longleftarrow \mathsf{GGen}(\kappa)$
2. $(r, h) \longleftarrow \mathcal{A}(G, H, R, g)$ where $r \in R, h \in H$ and $g \overset{U}{\longleftarrow} G$
3. The output of the experiment is defined to be 1 if $z = r \cdot h$ and 0 otherwise.

This experiment defines the *Splitting Problem*, SP, *(relative to* GGen*)*.

Next, we recall the *Subgroup Membership Problem* (SMP). Let GGen' be a PPT algorithm that takes a security parameter κ as input and outputs descriptions (G, H) of a non-trivial, proper subgroup H of a (not necessarily Abelian) finite group G. Consider the following experiment for a given algorithm GGen', algorithm $\mathcal{A}$ and parameter κ:

Experiment $\mathbf{Exp}^{\mathrm{SMP}}_{\mathcal{A},\mathsf{GGen}'}(\kappa)$:

1. $(G, H) \longleftarrow \mathsf{GGen}'(\kappa)$
2. Choose $b \overset{U}{\longleftarrow} \{0, 1\}$. If $b = 1$: $g \longleftarrow G \setminus H$. Otherwise: $g \longleftarrow H$.
3. $d \longleftarrow \mathcal{A}(G, H, g)$ where $d \in \{0, 1\}$
4. The output of the experiment is defined to be 1 if $d = b$ and 0 otherwise.

This experiment defines the *Subgroup Membership Problem*, SMP, *(relative to* GGen'*)* which, informally, states that given (G, H, g) where $g \longleftarrow G$, one has to decide whether $g \in H$ or not.

It is easy to see that if one can efficiently solve the Splitting Problem for (G, H, R), one can also solve the Subgroup Membership Problem for (G, H): Let $g \in G$ be the challenge of the SMP for (G, H). By using the SP-solver, we can compute $\sigma(g) = (r, h)$ and we have the relation that $g \in H$ *if and only if* $r = \mathbf{1}$. So deciding whether $g \in H$ amounts to deciding whether $r = \mathbf{1}$ which is easy since the neutral element $\mathbf{1}$ of R is always included in the description of R (cf. Section 2.1).

4.1.2. The Splitting Oracle-Assisted Subgroup Membership Problem

We introduce the *Splitting Oracle-Assisted Subgroup Membership Problem* (SOAP), which is situated in the same setting as the Splitting Problem (recall the groups $\widehat{G}, G, H, R$) and consists of two phases. In the first phase the adversary is given access to an oracle $\mathcal{O}_{\mathrm{SP}}^{\widehat{G},G,H,R}(\cdot)$ that either solves the Splitting Problem for (G, H, R) or outputs the special symbol $\perp$ if the input was not an element of G. In the second/challenge phase, the adversary has to solve the Subgroup Membership Problem for (G, H). Formally, we let GGen be a PPT algorithm that takes a security parameter κ as input and outputs descriptions $(\widehat{G}, G, H, R)$ of a non-trivial, proper normal subgroup H of a group G that is itself a subgroup of a finite group $\widehat{G}$ and a system of representatives $R \subseteq G$ of G/H. As before, we assume that the descriptions also contain an efficient algorithms for sampling from $G \setminus H$. We consider the following experiment for a given algorithm GGen, algorithm $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ and parameter κ:

Experiment $\mathbf{Exp}_{\mathcal{A},\mathsf{GGen}}^{\mathrm{SOAP}}(\kappa)$:

1. $(\widehat{G}, G, H, R) \longleftarrow \mathsf{GGen}(\kappa)$
2. $s \longleftarrow \mathcal{A}_1^{\mathcal{O}_{\mathrm{SP}}^{\widehat{G},G,H,R}(\cdot)}(\widehat{G}, G, H, R)$ where s is a state of $\mathcal{A}_1$
3. Choose $b \overset{U}{\longleftarrow} \{0, 1\}$. If $b = 1$: $g \longleftarrow G \setminus H$. Otherwise: $g \longleftarrow H$
4. $d \longleftarrow \mathcal{A}_2(\widehat{G}, G, H, R, s, g)$ where $d \in \{0, 1\}$
5. The output of the experiment is defined to be 1 if $d = b$ and 0 otherwise.

This experiment defines the *Splitting Oracle-Assisted Subgroup Membership Problem (relative to* GGen*)*, denoted by SOAP. We note that the splitting oracle $\mathcal{O}_{\mathrm{SP}}^{\widehat{G},G,H,R}(\cdot)$ does not solve a random instance of SP, but rather solves the Splitting Problem for (G, H, R) which are the parameters of the corresponding SMP the adversary has to solve in the challenge phase. Therefore, we say that the splitting oracle solves the *static* Splitting Problem (SSP), while "static" in this context refers to the SMP instance the adversary has to solve in the SOAP game. In fact, a very special case of SOAP occured in [84], where Lipmaa introduces a dedicated notation. We follow his approach and sometimes denote SOAP by $\mathrm{SMP}^{\mathrm{SSP}}$.

4.1.3. The Subset Membership Problem

The *Subset Membership Problem* (SSMP) was introduced by Cramer and Shoup in [29] and is a generalization of the SMP. Let SGen be a PPT algorithm that takes a security parameter κ as input and outputs descriptions $(\mathcal{S}, \mathcal{N})$ where $\mathcal{N}$ is a non-trivial, proper subset of a finite set $\mathcal{S}$. As in the setting of SMP, we assume that the descriptions contain an efficient algorithm for sampling from the set $\mathcal{S} \setminus \mathcal{N}$. Consider the following experiment for a given algorithm SGen, algorithm $\mathcal{A}$ and parameter κ:

Experiment $\mathbf{Exp}^{\mathrm{SSMP}}_{\mathcal{A},\mathsf{SGen}}(\kappa)$:

1. $(\mathcal{S},\mathcal{N}) \longleftarrow \mathsf{SGen}(\kappa)$
2. Choose $b \overset{U}{\longleftarrow} \{0,1\}$. If $b=1$: $z \longleftarrow \mathcal{S} \setminus \mathcal{N}$. Otherwise: $z \longleftarrow \mathcal{N}$.
3. $d \longleftarrow \mathcal{A}(\mathcal{S},\mathcal{N},z)$ where $d \in \{0,1\}$
4. The output of the experiment is defined to be 1 if $d=b$ and 0 otherwise.

This experiment defines the *Subset Membership Problem* SSMP *(relative to* SGen*)* which, informally, states that given $(\mathcal{S},\mathcal{N},z)$ where $z \longleftarrow \mathcal{S}$, one has to decide whether $z \in \mathcal{N}$ or not.

4.2. Characterizing IND-CPA Security

4.2.1. Group Homomorphic Encryption

Let $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ be a GHE scheme with the group $\mathcal{C}$ of all encryptions and the subgroup $\mathcal{C}_1$ of all encryptions of the neutral element 1. We want to show a necessary condition for $\mathcal{E}$ to be IND-CPA secure. More precisely, we show that if $\mathcal{E}$ is IND-CPA secure, then the SMP is hard for $(\mathcal{C},\mathcal{C}_1)$ (i.e., relative to KeyGen). The sampling algorithms for the groups $\mathcal{C}$ and $\mathcal{C}_1$ are the ones inherited from the encryption algorithm of $\mathcal{E}$. In particular, sampling an element c from $\mathcal{C}\setminus\mathcal{C}_1$ is done by choosing a random message $m \in \mathcal{P}$ with $m \neq 1$ and then computing c as $\mathsf{Enc}_{\mathsf{pk}}(m;r)$ for $r \longleftarrow \mathsf{Rnd}$. First, we prove the following helpful result.

Lemma 3 *If* $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *is a* GHE *scheme, we have for any two messages* $m, m' \in \mathcal{P}$ *and* $\rho \in \mathsf{Rnd}$*:*

$$\mathcal{C}_{mm'} = \mathsf{Enc}_{\mathsf{pk}}(m;\rho) \cdot \mathcal{C}_{m'} \text{ (as sets)} \tag{4.1}$$

Proof: **"$\supset$":** Recall that $\mathcal{C}_{mm'} = \{c \in \mathcal{C} \mid \mathsf{Dec}_{\mathsf{sk}}(c) = mm'\}$ by definition. Now, for $c' \in \mathcal{C}_{m'}$, we have that $\mathsf{Dec}_{\mathsf{sk}}(\mathsf{Enc}_{\mathsf{pk}}(m;\rho)\cdot c') = mm'$ since $\mathcal{E}$ is group homomorphic. This implies that $\mathsf{Enc}_{\mathsf{pk}}(m;\rho)\cdot c' \in \mathcal{C}_{mm'}$, which shows the first inclusion.

"=": By the first inclusion, we know that $\mathcal{C}_{mm'} \supset \mathsf{Enc}_{\mathsf{pk}}(m;\rho)\cdot\mathcal{C}_{m'}$. But Lemma 2 tells us that $|\mathcal{C}_{mm'}| = |\mathcal{C}_{m'}|$ which equals $|\mathsf{Enc}_{\mathsf{pk}}(m;\rho)\cdot\mathcal{C}_{m'}|$. This, however, yields the set equality $\mathcal{C}_{mm'} = \mathsf{Enc}_{\mathsf{pk}}(m;\rho)\cdot\mathcal{C}_{m'}$.

□

The next Theorem shows the aforementioned necessary condition for a GHE scheme to be IND-CPA secure.

Theorem 5 (Necessary Condition on IND-CPA Security) *For a* GHE *scheme* $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *we have:*

$$\mathcal{E} \textit{ is IND-CPA secure } \Longrightarrow \text{ SMP } \textit{is hard (relative to } \mathsf{KeyGen}\textit{)}.$$

Proof: We prove the Theorem by contradition and show that if we have a PPT algorithm $\mathcal{A}$ that breaks the hardness of SMP with non-negligible advantage $\Gamma(\kappa)$, we can construct (in PPT) an algorithm $\mathcal{B}$ that breaks the IND-CPA security with non-negligible advantage. To this end, we fix an SMP-adversary $\mathcal{A}$ and construct an IND-CPA-adversary $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$.

We start by letting $\mathcal{B}_1$ choose $m_0 = 1 \in \mathcal{P}$ and a random message $m_1 \longleftarrow \mathcal{P}$ with $m \neq 1$. Next, $\mathcal{B}_1$ sends the two messages m_0, m_1 to the IND-CPA-challenger. The challenger chooses a random bit $b \in \{0, 1\}$ and returns the ciphertext $c \longleftarrow \mathsf{Enc}_{\mathsf{pk}}(m_b)$. Then, $\mathcal{B}_2$ simply relays the ciphertext c to the SMP-adversary $\mathcal{A}$ who will output a bit $d \in \{0, 1\}$, which in turn is forwarded by $\mathcal{B}_2$ to the IND-CPA-challenger.

It remains to be shown that $d = b$ with a non-negligible advantage, i.e., that

$$\left| \Pr\left[\mathbf{Exp}^{\text{ind-cpa}}_{\mathcal{B},\mathsf{KeyGen}}(\kappa) = 1\right] - \frac{1}{2} \right| \quad \text{is non-negligible.}$$

By the assumption on $\mathcal{A}$, we know that $\mathcal{A}$'s advantage is non-negligible, namely $\Gamma(\kappa)$. Moreover, the ciphertext c is formatted as in SMP so $\mathcal{A}$ behaves as in the SMP-game (it is either a fresh encryption of 1 or of a random message different from 1), meaning that $d = b$ with advantage $\left| \Pr\left[\mathbf{Exp}^{\text{smp}}_{\mathcal{A},\mathsf{KeyGen}}(\kappa) = 1\right] - \frac{1}{2} \right|$, i.e.,

$$\left| \Pr\left[\mathbf{Exp}^{\text{ind-cpa}}_{\mathcal{B},\mathsf{KeyGen}}(\kappa) = 1\right] - \frac{1}{2} \right| = \left| \Pr\left[\mathbf{Exp}^{\text{smp}}_{\mathcal{A},\mathsf{KeyGen}}(\kappa) = 1\right] - \frac{1}{2} \right| = \Gamma(\kappa).$$

This concludes the proof of the Theorem. □

The converse of this Theorem, however, does not hold in general as the following example shows.

Example (SMP Hard $\not\Rightarrow$ IND-CPA Security) We construct a GHE scheme that is *not* IND-CPA secure, but whose corresponding SMP is hard. To this end, let $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ be an IND-CPA secure GHE scheme with a plaintext group $\mathcal{P}$ that is exponentially large in the security parameter such that the sampling algorithm contained in the description of $\mathcal{P}$ that samples according to the uniform distribution—for instance, this property is satisfied by the ElGamal cryptosystem [36]. By Theorem 5, we know that the SMP relative to KeyGen is hard. Now, the idea is to slightly modify $\mathcal{E}$ such that the corresponding SMP remains hard but the IND-CPA security can be easily broken. Therefore, we fix a public value $r^* \in \mathsf{Rnd}$, a public message $m^* \in \mathcal{P}$, and construct a scheme $\mathcal{E}^*$ which is exactly the *same* as $\mathcal{E}$, except for the encryption algorithm. We denote the encryption algorithm of $\mathcal{E}^*$ by Enc^* and define it as follows:

Encryption. Enc^* takes the public key pk, a message m, and a random value $r \in \mathsf{Rnd}$ as input. The output is defined as follows:

$$\mathsf{Enc}^*_{\mathsf{pk}}(m;r) := \begin{cases} \mathsf{Enc}_{\mathsf{pk}}(m;r^*) & \text{, if } m = m^* \\ \mathsf{Enc}_{\mathsf{pk}}(m;r) & \text{, otherwise.} \end{cases}$$

Recall that $r^* \in \mathsf{Rnd}$ and $m^* \in \mathcal{P}$ are fixed and public values corresponding to $\mathcal{E}^*$.

Our new scheme $\mathcal{E}^*$ certainly is *not* IND-CPA secure: Assume an adversary chooses two messages $m_0, m_1 \in \mathcal{P}$ where $m_0 = m^*$. Upon the retrieval of an encryption c of either of the two messages, the adversary checks whether $c = \mathsf{Enc}_{\mathsf{pk}}(m;r^*)$. If so, she knows that m_0 was encrypted, while otherwise, c encrypts the message m_1.

On the other hand, we see that the SMP corresponding to $\mathcal{E}^*$ is still hard: Recall that in the SMP game, the challenger flips a coin $b \in \{0,1\}$. If $b = 1$, the challenger samples a randomly chosen message $m \overset{U}{\longleftarrow} \mathcal{P}$ with $m \neq 1$ (recall that sampling from $\mathcal{P}$ is done according to the uniform distribution) and sends $c = \mathsf{Enc}^*_{\mathsf{pk}}(m)$ to an SMP-adversary. If $b = 0$, the challenger simply sends $c = \mathsf{Enc}^*_{\mathsf{pk}}(1)$ to the adversary. It is obvious that this SMP instance (using Enc^*) behaves exactly in the same way as our orginial SMP game (with Enc) corresponding to $\mathcal{E}$ if $b = 0$. But also if $b = 1$, it is clear that the advantage of an adversary in the SMP with Enc^* is negligibly close to the advantage of an adversary in the SMP with Enc. This is due to the fact that the plaintext space is exponentially large in the security parameter and the particular message m^* will only be chosen with a negligible probability. Therefore, the two games SMP with Enc^* and SMP with Enc are computationally indistinguishable, and so our SMP corresponding to $\mathcal{E}^*$ is hard.

4.2.2. Shift-Type Group Homomorphic Encryption

We have seen in the previous section that, in general, we cannot achieve an equivalence between the IND-CPA security of a given GHE scheme and the hardness of SMP. Therefore, we restrict our attention to a certain large subclass of GHE schemes where we are actually able to prove such an equivalence.

Definition 12 (Shift-Type GHE) *A* GHE *scheme* $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *is said to be of* shift-type*, if the first component of the output of* KeyGen *(i.e., any public key* pk*) contains a description of an efficient injective homomorphism* $\varphi : \mathcal{P} \longrightarrow \widehat{\mathcal{C}}$ *such that for all plaintexts* $m \in \mathcal{P}$*, we have:*

$$\mathsf{Enc}_{\mathsf{pk}}(m;r) = \varphi(m) \cdot \mathsf{Enc}_{\mathsf{pk}}(1;r).$$

Moreover, we require that an efficient description of the inverse of φ *on* $im(\varphi)$ *is also contained in the public key* pk*, while we denote this inverse on* $im(\varphi)$ *by* φ^{-1}*.*

Theorem 5 tells us that *if* a shift-type GHE scheme is IND-CPA secure, then the corresponding SMP is hard. We show that the converse holds as well.

Theorem 6 (Characterization of IND-CPA Security) *For a shift-type* GHE *scheme* $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *we have:*

$$\mathcal{E} \textit{ is IND-CPA secure} \iff \text{SMP} \textit{ is hard (relative to } \mathsf{KeyGen}\text{)}.$$

Proof: "$\Leftarrow$": Assume that $\mathcal{E}$ is not IND-CPA secure, i.e., there exists a PPT algorithm $\mathcal{A}^{\text{cpa}} = (\mathcal{A}_1^{\text{cpa}}, \mathcal{A}_2^{\text{cpa}})$ that breaks the security with non-negligible advantage $f(\kappa)$. We derive a contradiction by constructing a PPT algorithm $\mathcal{A}^{\text{smp}}$ that successfully solves SMP with advantage $\frac{1}{2}f(\kappa)$.

Since SMP and IND-CPA are both considered relative to KeyGen, $\mathcal{A}^{\text{smp}}$ can simply forward the public key pk of the output of $\mathsf{KeyGen}(\kappa)$ to $\mathcal{A}_1^{\text{cpa}}$. Next, $\mathcal{A}_1^{\text{cpa}}$ outputs two messages $m_0, m_1 \in \mathcal{P}$ and sends them to $\mathcal{A}^{\text{smp}}$. The SMP challenger chooses a bit $b \xleftarrow{U} \{0,1\}$ and sends the challenge $c \in \mathcal{C}$ to $\mathcal{A}^{\text{smp}}$, who then chooses a bit $d \xleftarrow{U} \{0,1\}$ and sends the challenge $c_d := \varphi(m_d) \cdot c$ to $\mathcal{A}_2^{\text{cpa}}$. Now, $\mathcal{A}_2^{\text{cpa}}$ outputs a bit d' and sends it back to $\mathcal{A}^{\text{smp}}$ which sends $b' := d \oplus d'$ to the SMP challenger.

We have the following relations: If $b = 0$, then $c \in \mathcal{C}_1$ and $c_d \in \mathcal{C}_{m_d}$ (a fresh encryption of m_d) by definition. Hence, $\mathcal{A}_2^{\text{cpa}}$ makes the right guess with advantage $f(\kappa)$, i.e., $\Pr[b' = b | b = 0] \geq \frac{1}{2} + f(\kappa)$. If $b = 1$, then $c \in \mathcal{C} \setminus \mathcal{C}_1$, meaning that it is a fresh encryption of some random message $m \neq 1$ (by definition of the set $\mathcal{C}$). But φ is a homomorphism and so c_d is a fresh encryption of (the random message) $m_d \cdot m$. Hence, $\mathcal{A}_2^{\text{cpa}}$ guesses d with no advantage, i.e. $\Pr[b' = b | b = 1] = \frac{1}{2}$. We have shown:

$$\begin{aligned}\Pr\left[\mathbf{Exp}^{\text{SMP}}_{\mathcal{A}^{\text{smp}},\mathsf{KeyGen}}(\kappa) = 1\right] &= \sum_{\beta \in \{0,1\}} \Pr\left[b' = b | b = \beta\right] \cdot \Pr\left[b = \beta\right] \\ &\geq \frac{1}{2} \cdot \left(\frac{1}{2} + f(\kappa) + \frac{1}{2}\right) = \frac{1}{2} + \frac{1}{2} f(\kappa).\end{aligned}$$

"$\Rightarrow$": This direction was proven in Theorem 5. $\square$

4.2.3. Somewhat and Fully Homomorphic Encryption

Similarly to the group homomorphic case, we define the notion of *shift-type* for SWHE schemes and can actually show a similar IND-CPA security characterization.

Definition 13 (Shift-Type Homomorphic Encryption) *A* SWHE *scheme* $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ *is of* shift-type*, if for every output* $(\mathsf{pk}, \mathsf{sk})$ *of* $\mathsf{KeyGen}(\kappa)$*, the plaintext space* $\mathcal{P}$ *and the ciphertext space* $\widehat{\mathcal{C}}$ *are (multiplicatively written) non-trivial groups such that the public key* pk *contains a description of a subset* $\mathcal{N} \subseteq \widehat{\mathcal{C}}$

and an efficient injective homomorphism $\varphi : \mathcal{P} \longrightarrow \widehat{\mathcal{C}}$ *so that for all plaintexts* $m \in \mathcal{P}$,

$$\mathsf{Enc}_{\mathsf{pk}}(m) \text{ outputs } \varphi(m) \cdot n, \text{ where } n \longleftarrow \mathcal{N}.$$

Moreover, we require that an efficient description of the inverse of φ *on* $im(\varphi)$ *is also contained in the public key* pk*, while we denote this inverse on* $im(\varphi)$ *by* φ^{-1}.

As for shift-type GHE schemes, we denote the *set of all encryptions* by

$$\mathcal{C} := \{\mathsf{Enc}_{\mathsf{pk}}(m) \mid m \in \mathcal{P}\} \subseteq \widehat{\mathcal{C}}$$

and sometimes call its elements *fresh* ciphertexts/encryptions. Since φ is a homomorphism, we know that $\mathcal{N}$ is actually a subset of $\mathcal{C}$. Moreover, we use the notation

$$\mathcal{C}_m := \{c \in \mathcal{C} \mid \mathsf{Dec}_{\mathsf{sk}}(c) = m\}$$

to denote the set of fresh ciphertexts decrypting to $m \in \mathcal{P}$. In particular, we have $\mathcal{N} = \mathcal{C}_1$ in this notation. We are now in a position to prove a characterization of IND-CPA security of shift-type SWHE schemes. More precisely, we show that if a shift-type SWHE scheme $\mathcal{E}$ is IND-CPA secure, then the SMP is hard for $(\mathcal{C}, \mathcal{C}_1)$ (i.e., relative to KeyGen), and vice versa. The sampling algorithms for the sets $\mathcal{C}$ and $\mathcal{C}_1$ are the ones inherited from the encryption algorithm of $\mathcal{E}$. In particular, sampling an element c from $\mathcal{C} \setminus \mathcal{C}_1$ is done by choosing a random message $m \in \mathcal{P}$ with $m \neq 1$ and then computing c as $\mathsf{Enc}_{\mathsf{pk}}(m; r)$ for $r \longleftarrow \mathsf{Rnd}$.

Theorem 7 (IND-CPA Security of Shift-Type Schemes) *For a shift-type homomorphic encryption scheme* $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ *we have:*

$$\mathcal{E} \text{ is IND-CPA (rel. to } \mathsf{KeyGen}) \iff \text{SSMP is hard (rel. to } \mathsf{KeyGen})$$

Proof: "$\Leftarrow$": Assume that $\mathcal{E}$ is not IND-CPA secure, i.e. there exists a PPT algorithm $\mathcal{A}^{\text{cpa}} = (\mathcal{A}_1^{\text{cpa}}, \mathcal{A}_2^{\text{cpa}})$ that breaks the security with non-negligible advantage $f(\kappa)$. We derive a contradiction by constructing a PPT algorithm $\mathcal{A}^{\text{ssmp}}$ that successfully solves SSMP with advantage $\frac{1}{2}f(\kappa)$.

Since SSMP and IND-CPA are both considered relative to KeyGen, $\mathcal{A}^{\text{ssmp}}$ can simply forward the public key pk of the output of $\mathsf{KeyGen}(\kappa)$ to $\mathcal{A}_1^{\text{cpa}}$. Next, $\mathcal{A}_1^{\text{cpa}}$ outputs two messages $m_0, m_1 \in \mathcal{P}$ to $\mathcal{A}^{\text{ssmp}}$. The SSMP challenger chooses a bit $b \overset{U}{\longleftarrow} \{0,1\}$ and sends the challenge $c \in \mathcal{C}$ to $\mathcal{A}^{\text{ssmp}}$, who then chooses a bit $d \overset{U}{\longleftarrow} \{0,1\}$ and sends the challenge $c_d := \varphi(m_d) \cdot c$ to $\mathcal{A}_2^{\text{cpa}}$. Now, $\mathcal{A}_2^{\text{cpa}}$ outputs a bit d' and sends it back to $\mathcal{A}^{\text{ssmp}}$ which sends $b' := d \oplus d'$ to the SSMP challenger.

We have the following relations: If $b = 0$, then $c \in \mathcal{N} = \mathcal{C}_1$ and $c_d \in \mathcal{C}_{m_d}$ (a fresh encryption of m_d) by definition. Hence, $\mathcal{A}_2^{\text{cpa}}$ makes the right guess with advantage $f(\kappa)$, i.e., $\Pr[b' = b | b = 0] \geq \frac{1}{2} + f(\kappa)$. If $b = 1$, then $c \in \mathcal{C}$, meaning that it is a fresh encryption (by definition of the set $\mathcal{C}$) of some random message m. But φ is a homomorphism and so c_d is a fresh encryption of (the random message) $m_d \cdot m$. Hence, $\mathcal{A}_2^{\text{cpa}}$ guesses d with no advantage, i.e. $\Pr[b' = b | b = 1] = \frac{1}{2}$. We have shown:

$$\begin{aligned}\Pr\left[\mathbf{Exp}^{\text{SSMP}}_{\mathcal{A}^{\text{ssmp}},\mathsf{KeyGen}}(\kappa) = 1\right] &= \sum_{\beta\in\{0,1\}} \Pr\left[b' = b | b = \beta\right] \cdot \Pr\left[b = \beta\right] \\ &\geq \frac{1}{2} \cdot \left(\frac{1}{2} + f(\kappa) + \frac{1}{2}\right) = \frac{1}{2} + \frac{1}{2} f(\kappa).\end{aligned}$$

"$\Rightarrow$": For the converse, we assume that there is a PPT algorithm $\mathcal{A}^{\text{ssmp}}$ that solves SSMP with advantage $f(\kappa)$. Similarly to what we have done above, we construct a PPT algorithm $\mathcal{A}^{\text{cpa}} = (\mathcal{A}_1^{\text{cpa}}, \mathcal{A}_2^{\text{cpa}})$ that successfully breaks the IND-CPA security with advantage $f(\kappa)$.

Again as above, $\mathcal{A}_1^{\text{cpa}}$ forwards the output of $\mathsf{KeyGen}(\kappa)$ to $\mathcal{A}^{\text{ssmp}}$. Next, $\mathcal{A}_1^{\text{cpa}}$ outputs two random messages $m_0, m_1 \in \mathcal{P}$. The IND-CPA challenger chooses a bit $b \xleftarrow{U} \{0,1\}$ and sends the challenge $c_b \longleftarrow \mathsf{Enc}_{\mathsf{pk}}(m_b)$ to $\mathcal{A}_2^{\text{cpa}}$, who then computes $c := \varphi(m_0^{-1}) \cdot c_b \in \mathcal{C}$ and sends the challenge c to $\mathcal{A}^{\text{ssmp}}$. Now, $\mathcal{A}^{\text{ssmp}}$ returns a bit d' to $\mathcal{A}_2^{\text{cpa}}$ that then outputs $b' := d'$ to the IND-CPA challenger.

We have the following relations: If $b = 0$, then $c \in \mathcal{C}_1 = \mathcal{N}$ and $\mathcal{A}^{\text{ssmp}}$ guesses b with advantage $f(\kappa)$, i.e., $\Pr[b' = b | b = 0] \geq \frac{1}{2} + f(\kappa)$. If $b = 1$, then c is a random element in $\mathcal{C}$ and $\mathcal{A}^{\text{ssmp}}$ guesses b again with advantage $f(\kappa)$, i.e., $\Pr[b' = b | b = 1] \geq \frac{1}{2} + f(\kappa)$. Therefore, we have shown:

$$\begin{aligned}\Pr\left[\mathbf{Exp}^{\text{ind-cpa}}_{\mathcal{A}^{\text{cpa}},\mathsf{KeyGen}}(\kappa) = 1\right] &= \sum_{\beta\in\{0,1\}} \Pr\left[b' = b | b = \beta\right] \cdot \Pr\left[b = \beta\right] \\ &\geq \frac{1}{2} \cdot (1 + 2f(\kappa)) = \frac{1}{2} + f(\kappa).\end{aligned}$$

□

Recall that leveled FHE schemes are based on bootstrappable SWHE schemes (cf. Section 3.2). The previous Theorem characterizes all shift-type SWHE schemes, meaning that Theorem 2 shows that if the underlying bootstrapple scheme is of shift-type and the corresponding SSMP is hard, then the resulting leveled FHE scheme is IND-CPA secure. In fact, we show that the converse holds as well.

Theorem 8 *Let* $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ *be a bootstrappable scheme for a set of gates* Γ. *For parameter* $d = d(\lambda)$, *let* $\mathcal{E}^{(d)}$ *denote the output of the transformation from Theorem 1 applied to* $\mathcal{E}$ *and* d. *For all parameters* d, *it holds:*

$$\mathcal{E}^{(d)} \textit{ is IND-CPA secure} \iff \mathcal{E} \textit{ is IND-CPA secure.}$$

In particular, if $\mathcal{E}$ *is of shift-type, we have:*

$$\mathcal{E}^{(d)} \textit{ is IND-CPA secure} \iff \text{SSMP} \textit{ is hard (rel. to } \mathsf{KeyGen}\textit{).}$$

Proof: For the first part of the Theorem, we have:
"$\Leftarrow$": This is Theorem 2.
"$\Rightarrow$": If $\mathcal{A}$ is a PPT adversary that successfully breaks the IND-CPA security of $\mathcal{E}$, then $\mathcal{A}$ can also be used to break the IND-CPA security of $\mathcal{E}^{(d)}$. By looking at the details of the tranformation of Theorem 1 (see Fact 2), we know that in the IND-CPA security game for $\mathcal{E}^{(d)}$, $\mathcal{A}$ receives the public key pk_d, outputs two messages $m_0, m_1 \in \mathcal{P}$ and gets the ciphertext $c \longleftarrow \mathsf{Enc}_{\mathsf{pk}_d}(m_b)$ as the challenge ciphertext, where $b \overset{U}{\longleftarrow} \{0,1\}$. Due to the initial assumption on $\mathcal{A}$, $\mathcal{A}$ can guess the bit b with non-negligible advantage.
The second part of the Theorem follows immediately by the first part together with Theorem 7. $\square$

For "pure" FHE schemes, the situation is a bit more involved due to the circular security that the underlying bootstrappable scheme must satisfy. At least we can show the following:

Theorem 9 *Let $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ be a bootstrappable scheme for a universal set of gates Γ. If the resulting scheme $\mathcal{E}^*$ of the transformation in Theorem 3 is circuit-private, it holds that*

$$\mathcal{E}^* \textit{ is IND-CPA secure} \iff \mathcal{E} \textit{ is } \text{circular} \textit{ secure.}$$

Proof: "$\Leftarrow$": This is Theorem 4.
"$\Rightarrow$": We assume that $\mathcal{E}$ is not circular secure, i.e., there exists a PPT algorithm $\mathcal{A}^{\text{circular}}$ that breaks the security of $\mathcal{E}$ with non-negligible advantage $f(\kappa)$. We derive a contradiction by constructing a PPT algorithm $\mathcal{A}^{\text{cpa}} = (\mathcal{A}_1^{\text{cpa}}, \mathcal{A}_2^{\text{cpa}})$ that successfully breaks the IND-CPA security of $\mathcal{E}^*$ with advantage $f(\kappa)$.

First, the adversary $\mathcal{A}_1^{\text{cpa}}$ receives the public key $\mathsf{pk}^* = (\mathsf{pk}, \overline{\mathsf{sk}_1}, \dots, \overline{\mathsf{sk}_\ell})$ where $\overline{\mathsf{sk}_i} \longleftarrow \mathsf{Enc}_{\mathsf{pk}}(\mathsf{sk}_i)$ and $\mathsf{sk}_1, \dots, \mathsf{sk}_\ell$ is the representation of the secret key sk as elements of $\mathcal{P}$. Then, $\mathcal{A}_1^{\text{cpa}}$ chooses messages $\mathbf{0} \neq m_0 \in \mathcal{P}$ and $m_1 := \mathbf{0}$ together with circuits C_i such that $C_i(m_0, \mathsf{sk}_i) = \mathsf{sk}_i$ and $C_i(m_1, \mathsf{sk}_i) = m_1$ for all $i = 1, \dots, \ell$. For instance, if we consider all boolean circuits and assume that $\mathcal{P} = \{0,1\}$, $\mathcal{A}_1^{\text{cpa}}$ could simply choose $m_0 = 1, m_1 = 0$ and C_i as a single AND-gate for all $i = 1, \dots, \ell$. Now, the IND-CPA challenger chooses a random bit $b \overset{U}{\longleftarrow} \{0,1\}$ and sends the challenge $c \longleftarrow \mathsf{Enc}_{\mathsf{pk}}(m_b)$ to $\mathcal{A}_2^{\text{cpa}}$. Since $\mathcal{E}^*$ is fully homomorphic, $\mathcal{A}_2^{\text{cpa}}$ can compute $\overline{\sigma_i} \longleftarrow \mathsf{Eval}_{\mathsf{pk}}(C_i, c, \overline{\mathsf{sk}_i})$ for all $i = 1, \dots, \ell$. Due to the correctness condition on $\mathcal{E}^*$, this means for all $i = 1, \dots, \ell$:

$$\sigma_i := \mathsf{Dec}_{\mathsf{sk}}(\overline{\sigma_i}) = C_i(m_b, \mathsf{sk}_i). \tag{4.2}$$

Next, $\mathcal{A}_2^{\text{cpa}}$ sends $(\mathsf{pk}, \overline{\sigma_1}, \dots, \overline{\sigma_\ell})$ to $\mathcal{A}^{\text{circular}}$ that returns a bit $d \in \{0,1\}$ which in turn is the output b' of $\mathcal{A}_2^{\text{cpa}}$, i.e., $b' = d$.

We have the following relations: If $b = 0$, then $\overline{\sigma_i}$ is computationally indistinguishable (since $\mathcal{E}^*$ was assumed to be circuit-private) from a fresh encryption of sk_i, meaning in particular that $\sigma_i = \mathsf{sk}_i$ for all $i = 1, \dots, \ell$ due to equation (4.2). Hence, $\mathcal{A}^{\text{circular}}$ makes the right guess on b with advantage $f(\kappa)$, i.e., $\Pr[b' = b | b = 0] \geq \frac{1}{2} + f(\kappa)$. If $b = 1$, then $\overline{\sigma_i}$ is computationally indistinguishable from a fresh encryption of $\mathbf{0}$, unrelated to pk, for all $i = 1, \dots, \ell$. Hence, $\mathcal{A}^{\text{circular}}$ again guesses b with advantage $f(\kappa)$, i.e., $\Pr[b' = b | b = 1] \geq \frac{1}{2} + f(\kappa)$. We have shown:

$$\begin{aligned}\Pr\left[\mathbf{Exp}^{\text{ind-cpa}}_{\mathcal{A}^{\text{cpa}},\mathsf{KeyGen}}(\kappa) = 1\right] &= \sum_{\beta \in \{0,1\}} \Pr\left[b' = b | b = \beta\right] \cdot \Pr\left[b = \beta\right] \\ &\geq \frac{1}{2} \cdot (1 + 2f(\kappa)) = \frac{1}{2} + f(\kappa).\end{aligned}$$

□

Remark 1 *We would like to stress that Theorem 9 actually holds in a more general context as well. Looking at the proof, one notices that there is no need for $\mathcal{E}^*$ to be circuit-private for all circuits. The circuit privacy is only needed for the special circuits C_i used in the proof. In particular, in the case when only boolean circuits are considered and the plaintext space is $\mathcal{P} = \{0, 1\}$, the circuits C_i are all the same, namely a aingle AND-gate.*

When dealing with the construction of “pure” FHE schemes, one usually assumes that the IND-CPA security of the underlying bootstrappable SWHE scheme already implies the circular security. Making this assumption allows us to characterize shift-type, circuit-pricate FHE schemes by their corresponding SSMP.

Corollary 1 *Let $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ be a shift-type bootstrappable scheme for a universal set of gates Γ. If the resulting scheme $\mathcal{E}^*$ of the transformation in Theorem 3 is circuit-private and assuming that the IND-CPA security of $\mathcal{E}$ is equivalent to its circular security, then it holds that*

$$\mathcal{E}^* \textit{ is IND-CPA secure} \iff \text{SSMP } \textit{is hard (rel. to } \mathsf{KeyGen}\textit{).}$$

Proof: This is an immediate consequence of Theorems 7 and 9. □

4.3. Characterizing IND-CCA1 Security

We have seen in Section 3.1 that homomorphic encryption schemes cannot be IND-CCA2 secure, meaning that IND-CCA1 is the strongest security notion for such schemes. In this section, we want to analyse the IND-CCA1 security notion of GHE schemes in detail. In fact, we will give a characterization of the IND-CCA1 security in terms of the splitting-oracle assisted subgroup membership problem, SOAP (cf. Section 4.1.2) in the case of shift-type GHE with decryption error (see Definition 2). We note that similar characterizations for general GHE schemes as well as for SWHE schemes are not known but also unlikely to exist. For the latter, this is due to the missing group theoretic structure in SWHE schemes, which would lead to a non-meaningful characterization. For non-shift-type GHE schemes, the problem is that, in general, we have no efficient splitting of the group $\mathcal{C}$ of all encryptions into $\mathcal{C}_1$ and some other group R (i.e., we cannot efficiently compute a system of representatives of $\mathcal{C}/\mathcal{C}_1$).

Recall that for a given shift-type GHE scheme $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$, a public key pk includes $(\widehat{\mathcal{C}}, \mathcal{C}, \mathcal{C}_1, \mathcal{P}, \varphi, \varphi^{-1})$. But for any plaintext $m \in \mathcal{P}$, we have that $\varphi(m) \in \mathcal{C}_m$, i.e., $\mathrm{im}(\varphi)$ is a system of representatives of $\mathcal{C}/\mathcal{C}_1$ by Lemma 2. Therefore, the public key pk (with $R = \mathrm{im}(\varphi)$) is a valid input to the SOAP experiment and we can consider SOAP relative to KeyGen in this way.

Theorem 10 (Characterization of IND-CCA1 Security) *For every shift-type* GHE *scheme* $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *with decryption error, we have:*

$$\mathcal{E} \textit{ is IND-CCA1 secure} \iff \text{SOAP } \textit{is hard (relative to } \mathsf{KeyGen})$$

Proof: "$\Leftarrow$": We assume that $\mathcal{E}$ is not IND-CCA1 secure, i.e., there exists a PPT algorithm $\mathcal{A}^{\mathrm{cca1}} = (\mathcal{A}_1^{\mathrm{cca1}}, \mathcal{A}_2^{\mathrm{cca1}})$ that breaks the security with non-negligible advantage $f(\kappa)$. We derive a contradiction by constructing a PPT algorithm $\mathcal{A}^{\mathrm{soap}} = (\mathcal{A}_1^{\mathrm{soap}}, \mathcal{A}_2^{\mathrm{soap}})$ that successfully solves SOAP with advantage $\frac{1}{2}f(\kappa)$.

Since SOAP and IND-CCA1 are both considered relative to KeyGen, $\mathcal{A}_1^{\mathrm{soap}}$ can simply forward the public key pk of the output of $\mathsf{KeyGen}(\kappa)$ to $\mathcal{A}_1^{\mathrm{cca1}}$. If $\mathcal{A}_1^{\mathrm{cca1}}$ queries the decryption oracle for a decryption of some ciphertext $c \in \widehat{\mathcal{C}}$, $\mathcal{A}_1^{\mathrm{soap}}$ asks the oracle $\mathcal{O}_{\mathrm{SP}}^{\widehat{\mathcal{C}},\mathcal{C},\mathcal{C}_1,R}(c)$ on input c (with $R = \mathrm{im}(\varphi)$) which outputs the element $\sigma(c) = (r, n) \in R \times \mathcal{C}_1$ if $c \in \mathcal{C}$ and $\perp$ otherwise. In the former case, it is readily seen that $m := \varphi^{-1}(r)$ is the correct decryption of c, since $\mathcal{E}$ is of shift-type. $\mathcal{A}_1^{\mathrm{soap}}$ forwards this plaintext m to $\mathcal{A}_1^{\mathrm{cca1}}$. In the latter case, $\mathcal{A}_1^{\mathrm{soap}}$ simply forwards $\perp$ to $\mathcal{A}_1^{\mathrm{cca1}}$.

After the query phase of $\mathcal{A}_1^{\mathrm{cca1}}$ is over, $\mathcal{A}_1^{\mathrm{cca1}}$ outputs two messages $m_0, m_1 \in \mathcal{P}$ to $\mathcal{A}_2^{\mathrm{soap}}$. The SOAP challenger chooses a bit $b \xleftarrow{U} \{0,1\}$ and sends the challenge $c \in \mathcal{C}$ to $\mathcal{A}^{\mathrm{soap}}$, who then chooses a bit $d \xleftarrow{U} \{0,1\}$ and sends the challenge $c_d := \varphi(m_d) \cdot c$ to $\mathcal{A}_2^{\mathrm{cca1}}$. Now, $\mathcal{A}_2^{\mathrm{cca1}}$ outputs a bit d' and sends it back to $\mathcal{A}_2^{\mathrm{soap}}$ which sends $b' := d \oplus d'$ to the SOAP challenger.

We have the following relations: If $b = 0$, then $c \in \mathcal{C}_1$ and c_d is a correct encryption of the message m_d. Hence, $\mathcal{A}_2^{\mathrm{cca1}}$ makes the right guess with advantage $f(\kappa)$, i.e. $\Pr[b' = b | b = 0] \geq \frac{1}{2} + f(\kappa)$. If $b = 1$, then $c \in \mathcal{C} \setminus \mathcal{C}_1$ and c_d is an encryption of a random message. Hence, $\mathcal{A}_2^{\text{cca1}}$ guesses d with no advantage, i.e. $\Pr[b' = b | b = 1] = \frac{1}{2}$. We have shown:

$$\begin{aligned}\Pr\left[\mathbf{Exp}^{\text{SOAP}}_{\mathcal{A}^{\text{soap}},\mathsf{KeyGen}}(\kappa) = 1\right] &= \sum_{\beta \in \{0,1\}} \Pr\left[b' = b | b = \beta\right] \cdot \Pr[b = \beta] \\ &\geq \frac{1}{2} \cdot \left(\frac{1}{2} + f(\kappa) + \frac{1}{2}\right) = \frac{1}{2} + \frac{1}{2} f(\kappa).\end{aligned}$$

"$\Rightarrow$": For the converse, we assume that there is a PPT algorithm $\mathcal{A}^{\text{soap}} = (\mathcal{A}_1^{\text{soap}}, \mathcal{A}_2^{\text{soap}})$ that solves SOAP with advantage $f(\kappa)$. Similarly to what we have done above, we construct a PPT algorithm $\mathcal{A}^{\text{cca1}} = (\mathcal{A}_1^{\text{cca1}}, \mathcal{A}_2^{\text{cca1}})$ that successfully breaks the IND-CCA1 security with advantage $f(\kappa)$.

First, $\mathcal{A}_1^{\text{cca1}}$ forwards the part $(\widehat{\mathcal{C}}, \mathcal{C}, \mathcal{C}_1, R = \text{im}(\varphi))$ of the output of $\mathsf{KeyGen}(\kappa)$ to $\mathcal{A}_1^{\text{soap}}$. If $\mathcal{A}_1^{\text{soap}}$ queries the oracle $\mathcal{O}_{\text{SP}}^{\widehat{\mathcal{C}},\mathcal{C},\mathcal{N},R}(c)$ on input $c \in \widehat{\mathcal{C}}$, $\mathcal{A}_1^{\text{cca1}}$ asks the decryption oracle for a decryption of c that outputs the plaintext $m := \mathsf{Dec}_{\mathsf{sk}}(c)$ if $c \in \mathcal{C}$ and $\perp$ otherwise. In the former case, we notice that $\varphi(m) \in R$ and so $\mathcal{A}_1^{\text{cca1}}$ sends the correct Splitting Problem solution $(\varphi(m), \varphi(m) \cdot c^{-1})$ to $\mathcal{A}_1^{\text{soap}}$. In the latter case, $\mathcal{A}_1^{\text{cca1}}$ simply forwards $\perp$ to $\mathcal{A}_1^{\text{soap}}$. After the query phase of $\mathcal{A}_1^{\text{soap}}$ is over, $\mathcal{A}_1^{\text{cca1}}$ outputs two randomly chosen messages $m_0, m_1 \in \mathcal{P}$. The IND-CCA1 challenger chooses a bit $b \xleftarrow{U} \{0,1\}$ and sends the challenge $c_b \longleftarrow \mathsf{Enc}_{\mathsf{pk}}(m_b)$ to $\mathcal{A}_2^{\text{cca1}}$, who then computes $c := c_b \cdot \varphi(m_0)^{-1} \in \mathcal{C}$ and sends the challenge c to $\mathcal{A}_2^{\text{soap}}$. Now, $\mathcal{A}_2^{\text{soap}}$ returns a bit d' to $\mathcal{A}_2^{\text{cca1}}$ that then outputs $b' := d'$ to the IND-CCA1 challenger.

We have the following relations: If $b = 0$, then $c \in \mathcal{C}_1$ and $\mathcal{A}_2^{\text{soap}}$ guesses b with advantage $f(\kappa)$, i.e. $\Pr[b' = b | b = 0] \geq \frac{1}{2} + f(\kappa)$. If $b = 1$, then $c \in \mathcal{C} \setminus \mathcal{C}_1$ and $\mathcal{A}_2^{\text{soap}}$ guesses b again with advantage $f(\kappa)$, i.e. $\Pr[b' = b | b = 1] \geq \frac{1}{2} + f(\kappa)$. Therefore, we have shown:

$$\begin{aligned}\Pr\left[\mathbf{Exp}^{\text{ind-cca1}}_{\mathcal{A}^{\text{cca1}},\mathsf{KeyGen}}(\kappa) = 1\right] &= \sum_{\beta \in \{0,1\}} \Pr\left[b' = b | b = \beta\right] \cdot \Pr[b = \beta] \\ &\geq \frac{1}{2} \cdot (1 + 2f(\kappa)) = \frac{1}{2} + f(\kappa).\end{aligned}$$

$\square$

5 Construction of Homomorphic Encryption

While the previous chapter was concerned with the characterization of the security of homomorphic encryption schemes, we next want to characterize such schemes in terms of their design and to give new constructions. Hence, we start this chapter by providing a universal blueprint for shift-type GHE schemes with decryption error. This means that any such scheme is an instatiation of our blueprint and so we know exactly how to construct a new scheme—we just have to put the proper parameters into our blueprint. Essentially, our result says that we can take any hard instance of SMP (or SOAP, respectively) and our blueprint will output an IND-CPA (or IND-CCA1, respectively) secure GHE scheme. Later on, we will use this to construct new GHE schemes with unique properties that turn out to be very beneficial in outsourcing computations as we will see in Chapter 7.

5.1. Universal Blueprint for Shift-Type Group Homomorphic Encryption with Decryption Error

We give a universal blueprint for the design of shift-type GHE schemes with decryption error. More precisely, we first define an abstract public-key encryption scheme that we show to be shift-type group homomorphic. Then, we prove that any given shift-type GHE scheme with decryption error is an instance of our abstract scheme. Due to its generality, we call our abstract scheme the Generic shIFt-Type (GIFT) scheme.

Definition 14 (GIFT **Scheme)** GIFT *is defined as a public-key encryption scheme* $\mathcal{E}_{\mathrm{GIFT}} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *with*

Key Generation: KeyGen *takes a security parameter* κ *as input and outputs a tuple* $(\mathsf{pk}, \mathsf{sk})$ *where* pk *is the public key that contains descriptions of*

- *a non-trivial group $\mathcal{P}$ of plaintexts and a non-trivial group $\widehat{\mathcal{C}}$ of ciphertexts together with a non-trivial subgroup $\mathcal{C} \leq \widehat{\mathcal{C}}$ that will act as the set of encryptions*
- *a non-trivial, proper normal subgroup $\mathcal{N}$ of $\mathcal{C}$ such that $|\mathcal{C}/\mathcal{N}| = |\mathcal{P}|$*
- *an efficient isomorphism $\varphi : \mathcal{P} \longrightarrow R$ where $R \subseteq \mathcal{C}$ (not necessarily a subgroup but certainly a group, cf. Fact 1) is a system of representatives of $\mathcal{C}/\mathcal{N}$ (φ^{-1} is efficiently computable as well),*

and sk *is the secret key that contains*

- *an efficient description of the epimorphism $\nu : \mathcal{C} \longrightarrow R$ such that $\nu(c)$ is the unique representative $r \in R$ with $c = r \cdot n$ for some $n \in \mathcal{N}$.*
- *an efficient function $\delta : \widehat{\mathcal{C}} \longrightarrow \{0,1\}$ such that $\delta(c) = 1 \iff c \in \mathcal{C}$.*

Encryption: Enc *takes the public key* pk *and a message $m \in \mathcal{P}$ as input and outputs the ciphertext $c := \varphi(m) \cdot n \in \mathcal{C}$ where $n \longleftarrow \mathcal{N}$.*

Decryption: Dec *takes the secret key* sk *and a ciphertext $c \in \widehat{\mathcal{C}}$ as input. If $\delta(c) = 0$, it outputs $\perp$, otherwise it outputs the plaintext $\varphi^{-1}(\nu(c)) \in \mathcal{P}$.*

Remark 2 *In* GIFT *we know that $\mathbf{1} \in \mathcal{N}$,*[1] *so*

$$\begin{aligned} \mathcal{C}_1 &= \{c \in \mathcal{C} \mid \varphi^{-1}(\nu(c)) = 1\} = \{c \in \mathcal{C} \mid \nu(c) = \mathbf{1}\} \\ &= \{c \in \mathcal{C} \mid c^{-1} \cdot \mathbf{1} \in \mathcal{N}\} = \mathcal{N}, \end{aligned}$$

i.e., $\mathcal{N}$ is the group of all encryptions of 1.

Next, we prove that GIFT indeed is a shift-type GHE scheme with decryption error, and that every such scheme can be described in terms of GIFT.

Theorem 11 (Generality) *Any shift-type* GHE *scheme with decryption error can be described in terms of* GIFT*, and vice versa.*

Proof: We start by proving that the GIFT scheme $\mathcal{E}_{\mathrm{GIFT}} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ is a shift-type GHE scheme with decryption error. By the definition of $\mathcal{E}_{\mathrm{GIFT}}$, it is easy to see that $\mathsf{Enc}_{\mathsf{pk}}$ has the shift-type structure and that the scheme has a decryption error (as there is a decision function δ). Therefore, it suffices to show the correctness of the scheme and that $\mathsf{Dec}_{\mathsf{sk}}$ is a group epimorphism on $\mathcal{C}$.

The correctness can be readily seen, since we know by definition that $\nu(r) = r$ for all $r \in R$, which implies that $\nu(\varphi(m)) = \varphi(m)$ and $\nu(n) = \mathbf{1}$ for all $m \in \mathcal{P}$ and all $n \in \mathcal{N}$. Using that ν and φ are homomorphisms, this yields for all $m \in \mathcal{P}$:

$$\varphi^{-1}(\nu(\varphi(m) \cdot n)) = \varphi^{-1}(\nu(\varphi(m)) \cdot \nu(\mathbf{1})) = \varphi^{-1}(\varphi(m) \cdot \mathbf{1}) = m.$$

[1]Recall that we denoted the representative in R of $1 \cdot \mathcal{N}$ by $\mathbf{1}$ (see Fact 1).

Clearly, $\mathsf{Dec}_{\mathsf{sk}}|_{\mathcal{C}} = \varphi^{-1} \circ \nu$ is an epimorphism since it is the composition of two epimorphisms with $\mathrm{im}(\nu) = \mathrm{dom}(\varphi^{-1})$.

Conversely, let $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ be an arbitrary shift-type group homomorphic scheme and let $(\mathsf{pk}, \mathsf{sk})$ be an output of $\mathsf{KeyGen}(\kappa)$. We define $\mathcal{N} := \mathcal{C}_1$, which is a proper normal subgroup of $\mathcal{C}$ by Lemma 2 and we set $R := \mathrm{im}(\varphi)$. Now, by Lemma 2, we know that R is a system of representatives of $\mathcal{C}/\mathcal{N}$. Then, we also know that $|\mathcal{P}| = |R| = |\mathcal{C}/\mathcal{N}|$. Since $\mathcal{N}$ is defined as $\mathcal{C}_1$, sampling from $\mathcal{N}$ is done by computing $\mathsf{Enc}_{\mathsf{pk}}(1; r)$ for some random $r \in \mathsf{Rnd}$. Therefore, the shift-type property of $\mathcal{E}$ ensures that the encryption algorithm Enc is a form as required in our GIFT scheme.

All remaining compontents of GIFT are given as follows: By considering $\nu : \mathcal{C} \to R$ as $\nu := \varphi \circ \mathsf{Dec}_{\mathsf{sk}}|_{\mathcal{C}}$, one easily sees that $\mathsf{Dec}_{\mathsf{sk}}(c) = \varphi^{-1}(\nu(c))$, if $c \in \mathcal{C}$. Otherwise, i.e., if $\delta(c) = 0$, we have $\mathsf{Dec}_{\mathsf{sk}}(c) = \perp$. In particular, when defining ν in this way, ν is an epimorphism. Hence, we have successfully described $\mathcal{E}$ in terms of GIFT. □

We have the following immediate consequence:

Corollary 2 *If we denote the* GIFT *instantiation (according to Theorem 11) of a given shift-type* GHE *scheme $\mathcal{E}$ with decryption error by* GIFT$(\mathcal{E})$*, we have that*

1. *$\mathcal{E}$ is IND-CPA secure $\iff$* GIFT$(\mathcal{E})$ *is IND-CPA secure,*
2. *$\mathcal{E}$ is IND-CCA1 secure $\iff$* GIFT$(\mathcal{E})$ *is IND-CCA1 secure.*

Proof: The first part is Theorem 6, while the second is Theorem 10. □

In total, we have shown that, concerning shift-type GHE schemes with decryption error, we can always restrict our attention to their GIFT instantiations.

5.2. Weakening the Underlying Assumption: The k-Linear Problem

We use our universal blueprint in order to construct the first IND-CPA secure GHE scheme that is provably secure under the *decisional k-linear* (DLIN_k) assumption, for $k \in \mathbb{N}$. Basing cryptographic tools on this assumption is of great value for a couple of reasons [66, 103]:

1. It is a natural generalization of DDH (in particular: $\mathrm{DLIN}_1 = \mathrm{DDH}$),
2. DLIN_k is provably hard in the generic group model, and
3. even if DLIN_k is easy, DLIN_{k+1} remains provably hard in the generic group model.

The final point means that the k-linear assumption is getting weaker with larger values for k. So by choosing the parameter k for the GHE scheme we construct in this section, we can weaken the underlying assumption for the scheme's security. In other words, we can make the scheme more secure in this sense.

Moreover, we introduce a new assumption with provably the same characteristics in the generic group model as DLIN_k, under which we show the IND-CCA1 security of our proposed scheme. We start by recalling some basics about DLIN_k.

The k-Linear Problem

Fix $k \in \mathbb{N}$. Let $\widehat{\mathcal{C}} := \mathcal{C} := G^{k+1}$ where G is a cyclic group of prime order p, generated by g. Furthermore, we choose $a_i \overset{U}{\longleftarrow} \mathbb{Z}_p^*$ for $i = 1, \ldots, k$ and set $\mathcal{N} := \{(g^{a_1 r_1}, \ldots, g^{a_k r_k}, g^{\sum_{i=1}^k r_i}) \mid \forall i = 1, \ldots, k : r_i \in \mathbb{Z}_p\}$ and $R := \langle 1 \rangle^k \times G$. Clearly, $|\mathcal{N}| = p^k$, $|R| = p$ and $\mathcal{N} \cap R = \{(1, \ldots, 1)\}$. Therefore, R is a system of representatives of $\mathcal{C}/\mathcal{N}$ (the isomorphism is given by $(1, \ldots, 1, g^r) \mapsto (1, \ldots, 1, g^r) \cdot \mathcal{N}$). The splitting map $\sigma : \mathcal{C} \to R \times \mathcal{N}$ for $(\mathcal{C}, \mathcal{N}, R)$ is given by

$$(c_1, \ldots, c_{k+1}) \mapsto \left(\left(1, \ldots, 1, c_{k+1} \cdot \left(\prod_{i=1}^{k} c_i^{a_i^{-1}} \right)^{-1} \right), \left(c_1, \ldots, c_k, \prod_{i=1}^{k} c_i^{a_i^{-1}} \right) \right).$$

Now, the *computational k-linear problem* (CLIN_k) is defined as the SP for $(\mathcal{C}, \mathcal{N}, R)$ while the *decisional k-linear problem* (DLIN_k) is defined as the SMP for $(\mathcal{C}, \mathcal{N})$. As a new problem, we define $\mathrm{DLIN}_k^{\mathrm{SCLIN}_k}$ as the SOAP for $(\widehat{\mathcal{C}}, \mathcal{C}, \mathcal{N}, R)$ where SCLIN_k is the *static*-CLIN_k, i.e. it is defined with respect to the public parameters of the underlying DLIN_k problem in $\mathrm{DLIN}_k^{\mathrm{SCLIN}_k}$.

Theorem 12 (On the Hardness of $\mathrm{DLIN}_k^{\mathrm{SCLIN}_k}$) *We have:*

1. *If* $\mathrm{DLIN}_{k+1}^{\mathrm{SCLIN}_{k+1}}$ *is easy, then so is* $\mathrm{DLIN}_k^{\mathrm{SCLIN}_k}$*, and*
2. $\mathrm{DLIN}_k^{\mathrm{SCLIN}_k}$ *is hard in the generic group model.*

Proof: The first statement is trivial, while the second holds since $\mathrm{DLIN}_1^{\mathrm{SCLIN}_1}$ is hard in the generic group model [84], and so $\mathrm{DLIN}_k^{\mathrm{SCLIN}_k}$ is hard in the generic group model by using the first statement. □

Additionally, we have the following result.

Theorem 13 ($\mathrm{DLIN}_k^{\mathrm{SCLIN}_k}$ in the Generic Group Model) *In the generic group model, we have the following* Progressive Property*:*

If $\mathrm{DLIN}_k^{\mathrm{SCLIN}_k}$ *is easy, then* $\mathrm{DLIN}_{k+1}^{\mathrm{SCLIN}_{k+1}}$ *is still hard.*

Proof: Let G be a cyclic group of prime order p. Similarly to Shacham's proof [103] of the progressive property of DLIN_k, we prove an even stronger result than Theorem 13 by using multilinear maps [15]. We call an efficient map $e_k : G^k \to G_T$ *k-multilinear*, if $e_k(z_1^{r_1}, \dots, z_k^{r_k}) = e_k(z_1, \dots, z_k)^{\prod_{i=1}^k r_i}$ for all $z_1, \dots, z_k \in G$ and $r_1, \dots, r_k \in \mathbb{Z}_p$.

In what follows, we show that in generic groups featuring a $(k+1)$-multilinear map $\mathrm{DLIN}_k^{\mathrm{SCLIN}_k}$ is easy, but $\mathrm{DLIN}_{k+1}^{\mathrm{SCLIN}_{k+1}}$ is hard. This result implies Theorem 13.

We make extensive use of Shacham's paper [103], starting with a trivial consequence of one of his results. In Lemma B.1 of [103] it is shown that when given a $(k+1)$-multilinear map, there is an efficient algorithm for deciding DLIN_k. Immediately, this yields:

Corollary 3 *Given a $(k+1)$-multilinear map, there is an efficient algorithm for solving* $\mathrm{DLIN}_k^{\mathrm{SCLIN}_k}$.

Next, we give an upper bound on the success probability of an $\mathrm{DLIN}_k^{\mathrm{SCLIN}_k}$-adversary in the presence of a k-multilinear map. We proof this results along the lines of [103] (wherein a similar results is proven for DLIN_k).

Lemma 4 *If a q-step ($q \geq 2k$) adversary $\mathcal{A}$ solves $\mathrm{DLIN}_k^{\mathrm{SCLIN}_k}$ in the generic group model (featuring a k-multilinear map), then its success probability is at most $\frac{q\cdot(q+2k+4)^2}{2p}$.*

Proof: First, we stress that the *computational* k-linear problems are all equivalent to each other [103], and we can therefore restrict our attention to the problem $\mathrm{DLIN}_k^{\mathrm{SCDH}}$. Now, let g_0 be a generator of G, and $a_1, \dots, a_k, y \overset{U}{\longleftarrow} \mathbb{Z}_p$. We set $g_i := g_0^{a_i}$ for $i \in \{1, \dots, k\}$ and $g := g_0^y$. Furthermore, let $r_1, \dots, r_k, s \overset{U}{\longleftarrow} \mathbb{Z}_p$ and $d \overset{U}{\longleftarrow} \{0,1\}$, and set $T_d := g_0^{y\sum_{i=1}^k r_i}$ and $T_{1-d} := g_0^s$. The adversary $\mathcal{A}$ is first given access to an SCDH oracle and then receives the opaque representations for the elements

$$g_0, g_0^{a_1}, \dots, g_0^{a_k}, g_0^y, g_0^{a_1 r_1}, \dots, g_0^{a_k r_k}, T_0, T_1. \tag{5.1}$$

Upon reception, $\mathcal{A}$ outputs a bit d' and wins, if $d' = d$.

Let $Q \leq q$ be the number of queries made by the adversary to the SCDH oracle. In the generic group model, the SCDH oracle is equivalent to the multiplication with the element y (cf., [84]). So in the challenge phase, the adversary $\mathcal{A}$ does not only get the opaque representations for the elements in (5.1), but also representations of $g_0^{y^2}, \dots, g_0^{y^{Q+1}}$. As usual in the generic group model [87], we have an algorithm $\mathcal{B}$ that, internally, keeps track of elements handled by $\mathcal{A}$ as polynomials in the ring $\mathbb{Z}_p[A_1, \dots, A_k, Y, R_1, \dots, R_k, S]$ and, externally, describes these as arbitrary opaque strings in some sufficiently large domain. It maintains these two representations in two lists $\{(F_i, \xi_i)\}$ and $\{(F_{T,i}, \xi_{T,i})\}$ for elements of G and G_T, respectively. We assume that the domain for external representations is large enough so that, except with negligible probability, $\mathcal{A}$ can only query for elements it previously obtained from $\mathcal{B}$, and $\mathcal{B}$ never outputs the same opaque representation for two different elements.

Now, in the challenge phase, $\mathcal{A}$ is provided with elements that $\mathcal{B}$ internally represents by the following polynomials:

$$g_0 : F = 1,\ g_1 : F = A_1,\ \ldots,\ g_k : F = A_k,\ g : F = Y,\ \ldots,\ g^{y^{Q+1}} : F = Y^{Q+1}$$
$$\text{and } g_1^{r_1} : F = A_1 R_1,\ \ldots,\ g_k^{r_k} : F = A_k R_k,\ T_0 : F = T_0,\ T_1 : F = T_1.$$

On these elements to which $\mathcal{A}$ is given opaque representations, $\mathcal{A}$ can perform the following operations by using $\mathcal{B}$:

- *Group Action:* On input two elements of G, internally represented as F_1 and F_2, $\mathcal{B}$ adds $F' := F_1 + F_2$ to the representation list of G (if not already there), and outputs with the corresponding external representation. The group action for G_T is handled analogously.

- *Inversion:* On input an element of G, internally represented as F, $\mathcal{B}$ adds $F' := -F$ to the representation list of G (if not already there), and outputs with the corresponding external representation. The inversion for G_T is handled analogously.

- *Multilinear Map:* On input k elements of G (internally represented as $F_1, \ldots, F_k$) $\mathcal{B}$ adds $F' := \prod_{i=1}^{k} F_i$ to the representation list of G_T (if not already there), and outputs with the corresponding external representation.

We see that for all F on the representation list for G, we have $\deg(F) \leq q$, while for all F_T on the representation list for G_T, we have $\deg(F_T) \leq 2k$. After placing the remaining $q - Q$ queries (recall that $\mathcal{A}$ is allowed to make q steps in total) to these operations, it outputs its guess d' for d.

Now, $\mathcal{B}$ chooses $a_1, \ldots, a_k, y, r_1, \ldots, r_k, s \overset{U}{\longleftarrow} \mathbb{Z}_p$. If we set

$$A_1 := a_1,\ \ldots,\ A_k := a_k,\ Y := y,\ R_1 := r_1,\ \ldots,\ R_k := r_k, \tag{5.2}$$
$$T_d := y \cdot \textstyle\sum_{i=1}^{k} r_i,\ T_{1-d} := s, \tag{5.3}$$

the simulation engineered by algorithm $\mathcal{B}$ is consistent with these values unless there are two distinct polynomials F_1 and F_2 on the representation list for G or two distinct polynomials $F_{T,1}$ and $F_{T,2}$ on the representation list for G_T that take on the same value under the assignment above. It remains to show that $\mathcal{A}$ cannot construct such a collision independently of the choice of the random values and that the probability that the choice of random values produces a collision is bounded. We recall that $\mathcal{A}$ additionally has the opaque representations of $y^2, \ldots, y^{Q+1}$ due to the SCDH oracle.

The probability that there are at least two equal values among $y, y^2, \dots, y^{Q+1}$ is negligible in p, and since all the random values are independent of each other, except for the value of $T_d = y \cdot \sum_{i=1}^{k} r_i$, the adversary $\mathcal{A}$ must produce a multiple of $Y \cdot \sum_{i=1}^{k} R_i$, say $F = XY \sum_{i=1}^{k} R_i$ for some non-zero X, only by using the terms in (5.2) and (5.3). Clearly, any monomial that can be produced from $A_1, \dots, A_k, Y, \dots, Y^{Q+1}, A_1R_1, \dots, A_kR_k, T_d, T_{1-d}$ by using the above described operations is divisible by A_i if it is divisible by R_i for each $i = 1, \dots, k$. Furthermore, for every i, each monomial in the expansion of XYR_i in $F = XY \sum_{j=1}^{k} R_j$ must be divisible by A_i, hence $A_i \mid X$ (a formal proof of this fact is given in [103]). Therefore, F is divisible by the $k+2$ monomials $A_1, \dots, A_k, Y$ and R_i for some i. Since $\mathcal{A}$ only knows Y and its powers $Y^2, \dots, Y^{Q+1}$, and since no term A_aA_b is known to $\mathcal{A}$ for any a, b, forming F would require taking the product of at least $k+1$ of the polynomials available to the adversary. But the multilinear map only allows for forming the product of at most k terms. Thus, $\mathcal{A}$ cannot produce F and is hence unable to cause a collision.

Finally, we give an upper bound for the probability that a random choice of the values $a_1, \dots, a_k, y, r_1, \dots, r_k, s$ causes the same value on two distinct polynomials. Since the degrees of the polynomials in the representation list of G are upper bounded by q, the probability that two such polynomials have the same evaluation for some random values is at most $\frac{q}{p}$ (over the choice of values) (cf., [105, Lemma 1]). Analogously, this probability is at most $\frac{2k}{p}$ for polynomials in the representation list of G_T since the degrees of these are upper bounded by $2k$. In the challenge phase, the two representation lists consist together of $2k+Q+4$ values. When the adversary $\mathcal{A}$ does its remaining $q-Q$ queries, the lists contain at most $q+2k+4$ values, and the success probability of $\mathcal{A}$ is bounded by

$$\binom{q+2k+4}{2}\frac{q}{p} \leq \frac{q \cdot (q+2k+4)^2}{2p}.$$

In particular, constant success probability requires $q = \Omega(\sqrt[3]{p})$ steps. $\square$

Therefore, we have proven Theorem 13 by taking Corollary 3 and Lemma 4 together. $\square$

The Cryptosystem and Its Security

Let $\widehat{\mathcal{C}}$, $\mathcal{C}$, $\mathcal{N}$, R, g and the a_i's be as in the previous section, and let $\delta : \widehat{\mathcal{C}} \longrightarrow \{0,1\}$ be defined as $\delta(c) = 1$ for all $c \in \widehat{\mathcal{C}} = \mathcal{C}$. Furthermore, we set $\mathcal{P} := G$. We have the isomorphism $\varphi : \mathcal{P} \to R$ given by $m \mapsto (1, \dots, 1, m)$ and the epimorphism $\nu : \mathcal{C} \to R$ given by $(c_1, \dots, c_{k+1}) \mapsto \left(1, \dots, 1, c_{k+1} \cdot \prod_{i=1}^{k} c_i^{-a_i^{-1}}\right)$. We have successfully defined all the ingredients for GIFT for a fixed $k \in \mathbb{N}$. The resulting cryptosystem can be summarized as follows:

Key Generation: *Input.* Security parameter κ. *Output.* $\mathsf{sk} = (a_1, \dots, a_k)$ and $\mathsf{pk} = (p, g, g_1 := g^{a_1}, \dots, g_k := g^{a_k})$ where $a_i \overset{U}{\longleftarrow} \mathbb{Z}_p^*$ for $i = 1, \dots, k$ and g is a generator of a cyclic group G of prime order p such that κ is the length of the binary representation of p.

Encryption: *Input.* Public key pk and plaintext $m \in G$. *Output.* Ciphertext c with

$$c := (g_1^{r_1}, \dots, g_k^{r_k}, m \cdot g^{\sum_{i=1}^{k} r_i}) \text{ where } r_i \overset{U}{\longleftarrow} \mathbb{Z}_p \text{ for } i = 1, \dots, k.$$

Decryption: *Input.* Secret key sk and ciphertext $c = (c_1, \dots, c_{k+1}) \in G^{k+1}$. *Output.* Plaintext $m := c_{k+1} \cdot \prod_{i=1}^{k} c_i^{-a_i^{-1}}$.

When instantiated with $k = 1$ the above cryptosystem is ElGamal (cf. Seciont 6.1.1), while for $k = 2$ it is the *linear encryption scheme* introduced in [13]. For the security of the introduced cryptosystem, Theorems 6 and 10 yield:

Corollary 4 *The above cryptosystem is IND-CPA secure (resp. IND-CCA1 secure) if and only if* DLIN_k *(resp.* $\mathrm{DLIN}_k^{\mathrm{SCLIN}_k}$*) is hard.*

5.3. Dedicated Homomorphic Encryption: The Double Decryption Mechanism

There are certain application scenarios, where a dedicated type of homomorphic encryption is needed. Here, we consider a concrete example taken from practice that involves a company having many employees (e.g., an insurance company) with a certain hierarchy among them, and in particular with some master authority (e.g., the head of the company) that sits at the top of this hierarchy. Most of the company's data is stored on some central servers where hierachical access control is enforced by using encryption. But it happens occasionally that some employees leave the company or new people are being employed, and so every employee should get her own public and corresponding private keys. In this scenario, the company should be concerned with the following challenges:

- To avoid expensive key management, employees should be able to generate their own key pairs without getting in touch with the master authority.
- If an employee leaves the company or loses her keys (this concerns both the public and the private key), the master authority still wants to be able to recover all data. Hence, the master authority needs some master secret (independent of the employees' individual private keys) that allows to decrypt *any* data stored on the company's servers. Moreover, the master authority should be able to check whether a ciphertext has been encrypted under a

given employee's public key. This is relevant, for instance, in the following case: Assume an unavailable employee (for whatever reason, maybe due to quitting) left some important data on the server, e.g., an encryption of an important decision (1 for 'yes' and 0 for 'no'). The master authority needs to know this decision, but at the same time needs to verify whether it was encrypted by the respective employee, i.e., under her public key. In fact, an encryption under the wrong employee's public key might lead the master to a wrong decision.

- Additionally, in practice there is often the requirement that the used cryptosystem has a certain malleability property or is even homomorphic.

The just described scenario is a typical application of so-called additively homomorphic *encryption schemes with a double decryption mechanism* (DD-PKE) [49] which combine all the above properties in just one cryptosystem. Roughly speaking, such schemes have two independent, additively homomorphic decryption procedures. Currently[2], there exist two homomorphic DD-PKE schemes in the literature: CS-Lite by Cramer and Shoup [29] and a variant called *BCP* by Bresson, Catalano and Pointcheval [20]. Looking at these two schemes in detail, one notices two major weaknesses:

1. In the BCP cryptosystem, in order for the master authority to decrypt a given ciphertext, it has to know the employee's public key under which it was created. This fact contradicts to the requirement that the company does not want to do any complex key management (and in fact simply does not see the public keys in general).

2. Furthermore, both cryptosystems have the drawback that the master authority is unable to check whether a given ciphertext was encrypted under a given public key. This also contradicts the requirements of the above scenario. We note here that the authors of [20] left such "ciphertext validity checks" of the master authority as an open question.

In this section, we propose the *first* additively homomorphic DD-PKE scheme that avoids both just mentioned drawbacks: It is *User-Independent* (i.e., the master decryption procedure is independent of the public keys of the employees/users) and it allows the master to *detect invalid ciphertexts* (i.e., given a ciphertext and a user's public key the master can check whether the ciphertext was encrypted under the given public key).

Our solution is based on elliptic curves over rings $\mathbb{Z}_{N^2}$ where $N = pq$ is some RSA-modulus, and we prove its semantic security under a Decisional Diffie-Hellman

[2]April 6, 2013

(DDH) related assumption on such curves. We refer the reader to Section 2.2 for the necessary background on elliptic curves over rings. Finally, we discuss different possible choices of elliptic curves in the setup of our cryptosystem. Since these choices might have an effect on the security of our scheme, we also consider randomly chosen curves (which we require to have an order with at least two large prime factors). For this, one has to rely on a conjecture by Galbraith and McKee [47] about the likelyhood of hitting on such curves. Therefore, we made a substantial number of experiments to get an idea on the efficiency of our setup algorithm for randomly chosen curves. Since there are only a few experimental results on this matter in the literatue, our results might be of independent interest.

Related Work

Since the first efficient, additively homomorphic encryption scheme was proposed by Paillier [93] a lot of follow-up papers appeared in this area (see [41] for a survey). In particular, there were many approaches to construct such schemes by using elliptic curves (see Galbraith's elliptic-curve-based Paillier scheme [46] and the references therein). While we are only interested in additively homomorphic encryption (i.e., it is possible to evaluate the addition of plaintexts over their encryptions without knowledge of the private key), much attention is recently being devoted to the topic of fully homomorphic encryption [18, 52], which allows for the evaluation of any circuit over encrypted data without being able to decrypt.

Besides the great many of works on homomorphic encryption, there are several constructions of (non-homomorphic) DD-PKE schemes [49, 116]. In this regard, we note that although identity-based encryption [14, 104] is related to DD-PKE, therein the master secret is essential in order to generate the users' private keys (in DD-PKE only some publicly known master information is needed, so there is no interaction between the users and the master).

Finally, we mention the only two existing schemes [29] and [20] which are *both* additively homomorphic *and* have a double decryption mechanism.

5.3.1. The Notion of (User-Independent) Double Decryption

We start by recalling what it means for an encryption scheme to have a double decryption mechanism. We do this along the lines of Galindo and Herranz's work [49].

Definition 15 *A public key encryption scheme* with a double decryption mechanism (DD-PKE) *is a tuple* $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{mDec})$ *of PPT algorithms such that*

Setup: $\mathsf{Setup}(\kappa)$ *takes a security parameter* κ *as input and outputs a tuple* $(\mathsf{PP}, \mathsf{MK})$ *where* PP *contains the public system parameters (particularly includes descrip-*

tions of the plaintext space $\mathcal{P}$ and the ciphertext space $\widehat{\mathcal{C}}$), and MK *is the master secret key which is only known to the master entity.*

Key Generation: $\mathsf{KeyGen}(\mathsf{PP})$ *takes the system's public parameters* PP *as input and outputs a pair of public/private keys* $(\mathsf{pk}, \mathsf{sk})$ *to a user.*

Encryption: $\mathsf{Enc}_{(\mathsf{PP},\mathsf{pk})}(m)$ *takes the public parameters* PP*, a user's public key* pk *and a message $m \in \mathcal{P}$ as input and outputs a ciphertext $c \in \mathcal{C}$.*

User Decryption: $\mathsf{Dec}_{(\mathsf{PP},\mathsf{sk})}(c)$ *takes the public parameters* PP*, a user's secret key* sk *and a ciphertext $c \in \widehat{\mathcal{C}}$ as input and outputs either a plaintext $m \in \mathcal{P}$ or the special symbol $\perp$.*

Master Decryption: $\mathsf{mDec}_{(\mathsf{PP},\mathsf{MK},\mathsf{pk})}(c)$ *takes the public parameters* PP*, the master secret key* MK*, a user's public key* pk *and a ciphertext $c \in \widehat{\mathcal{C}}$ as input and outputs either a plaintext $m \in \mathcal{P}$ or the symbol $\perp$.*

For such schemes, we require the usual correctness condition in public key encryption schemes both for the user decryption and the master decryption. It should be noted that by combining the system's public parameters in the user's public keys, we can think of a DD-PKE scheme as being a usual encryption scheme that additionally has a master decryption procedure (that uses the master secret key). Also, we stress that the notion of semantic security is exactly the same as that for usual public-key encryption schemes. Furthermore, it is noteworthy that the key generation algorithm KeyGen does *not* get the master secret MK as input.

Next, we introduce the notion of *User Independence* in the context of such DD-PKE schemes, which basically means that the master entity can decrypt any given ciphertext even without knowing the corresponding receiver (i.e., the user's public key under which it has been encrypted). In other words this means that the master decryption is independent of the users.

Definition 16 *A* DD-PKE *scheme is* user-independent (UI-DD-PKE) *if the master decryption does* not *get the user's public key as an input, i.e., it only gets the system's public parameters, the master secret and a ciphertext as input.*

5.3.2. Our Construction

We introduce a new public key cryptosystem with a simple structure that combines a couple of unique properties in a single scheme. Due to its many properties, we will restrict our attention to the scheme's formal definition and proof of correctness in this section, and deal with its properties in the next section. In order to formally define our cryptosystem, we need the following two facts:

Proposition 1 *If $N = pq$ is some RSA-modulus, i.e., p and q are primes of about the same bit length κ, then there is an efficient construction of elliptic curves $E : y^2z = x^3 + axz^2 + bz^3$ over $\mathbb{Z}_{N^2}$ such that $M := \operatorname{lcm}(\#E(\mathbb{Z}_p), \#E(\mathbb{Z}_q))$ has at least two large (of about the same size as p and q) prime factors.*

Proof: There are three different methods to construct such elliptic curves, which have direct influence on the system's efficiency and applicability. We therefore put the proof of this proposition in a section on its own (see Section 5.3.5). $\square$

Lemma 5 *As in Proposition 1, let $M \in \mathbb{N}$ have at least two large prime factors (of about κ bits). If $\pi(M)$ denotes the product of all small prime factors (including multiples) of M, then*

$$\Pr_{s \xleftarrow{U} \Pi(M)} [\gcd(s, M) \neq 1] \text{ is negligible in } \kappa,$$

where $\Pi(M) := \{s \in \mathbb{Z}_{N^2} \setminus \{0\} \mid \gcd(s, \pi(M)) = 1\}$.

Proof: Let $L(M) = \prod_{i=1}^{r} p_i^{\nu_i}$ be the product ($r \geq 2$) of all large (of about κ bits) prime factors in M, i.e., $M = \pi(M) \cdot L(M)$. By definition, we have that $\Pr[\gcd(s, M) \neq 1 \text{ for } s \in \Pi(M)] = \Pr\left[s \in \mathbb{Z}_{L(M)} \setminus \mathbb{Z}^*_{L(M)}\right]$. But if φ denotes Euler's totient function, we have

$$\#\mathbb{Z}^*_{L(M)} = \varphi(L(M)) = \prod_{i=1}^{r} (p_i - 1)p_i^{\nu_i - 1}, \text{ and hence } \frac{\#\mathbb{Z}^*_{L(M)}}{\#\mathbb{Z}_{L(M)}} = \prod_{i=1}^{r} \left(1 - \frac{1}{p_i}\right).$$

However, the fractions $\frac{1}{p_i}$ are negligible in κ, and so the product of all these is negligibly close to 1. Therefore, we have

$$\Pr\left[s \in \mathbb{Z}_{L(M)} \setminus \mathbb{Z}^*_{L(M)}\right] = 1 - \frac{\#\mathbb{Z}^*_{L(M)}}{\#\mathbb{Z}_{L(M)}} = 1 - (1 - \operatorname{negl}(\kappa)) = \operatorname{negl}(\kappa),$$

where $\operatorname{negl}(\kappa)$ denotes a negligible function in κ. $\square$

Definition 17 (Our Cryptosystem) *Our cryptosystem is defined as follows:*

Setup: $\mathsf{Setup}(\kappa)$ *computes an RSA-modulus $N = pq$ where p and q are primes of about the same bit length κ and constructs an elliptic curve $E : y^2z = x^3 + axz^2 + bz^3$ over $\mathbb{Z}_{N^2}$ such that E has the properties as described in Proposition 1. Furthermore, it chooses a point $Q = (x : y : z) \in E(\mathbb{Z}_{N^2})$ whose order divides $M = \operatorname{lcm}(\#E(\mathbb{Z}_p), \#E(\mathbb{Z}_q))$.*[3]

It outputs the public parameters $\mathsf{PP} := (N, \pi(M), a, b, Q)$ and the master secret key $\mathsf{MK} := M$. The plaintext space is $\mathcal{P} = \mathbb{Z}_N$ and the ciphertext space is $\widehat{\mathcal{C}} = \langle Q \rangle \times \langle Q, P_1 \rangle$.

[3]This can be done by taking a random point $Q' = (x' : y' : z') \in E(\mathbb{Z}_{N^2})$ and setting $Q := NQ'$. See also Section 5.3.5.

Key Generation: $\mathsf{KeyGen}(\mathsf{PP})$ *chooses* $s \in \mathbb{Z}_M^*$ *at random and computes* $R := sQ$. *This can be done by sampling* $s \in \Pi(M)$ *(which is possible as* $\pi(M)$ *is included in* PP*), since then* $s \in \mathbb{Z}_M^*$ *holds with overwhelming probability by Lemma 5.*[4] *It outputs the user's public key* $\mathsf{pk} := R$ *and secret key* $\mathsf{sk} := s$.

Encryption: $\mathsf{Enc}_{(\mathsf{PP},\mathsf{pk})}(m)$ *chooses a random value* $r \in \mathbb{Z}_{N^2}$ *and computes the ciphertext* (A, B) *as*

$$A := rQ \text{ and } B := rR + P_m.^5$$

User Decryption: $\mathsf{Dec}_{(\mathsf{PP},\mathsf{sk})}(A, B)$ *outputs*

$$m = \frac{x(B - sA)}{N}.$$

Master Decryption: $\mathsf{mDec}_{(\mathsf{PP},\mathsf{MK})}(A, B)$ *outputs*

$$m = \frac{x(MB)}{N} M^{-1} \text{ mod } N.$$

Concerning the *correctness* of both decryption procedures, we see that

$$\mathsf{Dec}_{(\mathsf{PP},\mathsf{sk})}(\mathsf{Enc}_{(\mathsf{PP},\mathsf{pk})}(m)) = \frac{x(rR + P_m - srQ)}{N} = m$$

and

$$\mathsf{mDec}_{(\mathsf{PP},\mathsf{MK})}(\mathsf{Enc}_{(\mathsf{PP},\mathsf{pk})}(m)) = \frac{x(M(rR + P_m))}{N} M^{-1} \bmod N = m$$

by using the fact that $\operatorname{ord} Q$ divides M, so $MR = sMQ = \mathcal{O}_\infty$.

Remark 3 *1. We stress that the knowledge of* M *is polynomial-time equivalent to the knowledge of the factorization of* N *(cf. [92, Theorem 10]). Therefore, it is computationally infeasible to compute the master secret key* MK *from the public parameters* PP.

2. It is also computationally infeasible to compute the user's secret key from its public key under the assumption that the Discrete Logarithm Problem is hard in $E(\mathbb{Z}_{N^2})$.

3. Without knowledge of the factorization of N *it is computationally infeasible to find a point* Q' *on the curve (that differs from linear combinations of the publicly known points* Q, R *and* P_1*), because one would need to solve polynomial equations in* $\mathbb{Z}_{N^2}$.

[4]We note that by using Hasse's bound on $\#E(\mathbb{Z}_p)$ and $\#E(\mathbb{Z}_q)$, we have $M \leq \#E(\mathbb{Z}_p)\#E(\mathbb{Z}_q) \leq N^2$.

[5]We note that if we forget about the first component A of our ciphertexts, then the encryption looks very similar to Galbraith's elliptic-curve-based Paillier scheme [46].

4. *We notice that the users' public keys are not needed in the master decryption algorithm, and so we have successfully defined a* UI-DD-PKE *scheme.*

5. *Finally, we note that the master decryption never fails on a given ciphertext $c \in \widehat{\mathcal{C}}$, and so it always outputs a message $m \in \mathcal{P}$. This is different for the user decryption. It will output $\perp$ if $x(B - sA)$ is not divisible by N. We will show in the next section (Property 2) that this happens if and only if the given ciphertext is invalid, which users can efficiently detect.*

5.3.3. Properties of the Cryptosystem

We start with two properties of the cryptosystem that are independent of the choice of elliptic curves in the setup algorithm as long as these curves satisfy the properties of Proposition 1.

Property 1 *The cryptosystem is additively homomorphic, i.e.,*

$$\mathsf{Dec}_{(\mathsf{PP},sk)}(\mathsf{Enc}_{(\mathsf{PP},\mathsf{pk})}(m_1) + \mathsf{Enc}_{(\mathsf{PP},\mathsf{pk})}(m_2)) = m_1 + m_2.$$

Together with item 4 of Remark 3 this means that the scheme is an additively homomorphic UI-DD-PKE *scheme.*

Proof: Let $m_1, m_2 \in \mathbb{Z}_N$ be two plaintexts encrypted as (A_1, B_1) and (A_2, B_2), respectively. Then $(A, B) := (A_1, B_1) + (A_2, B_2)$ is a ciphertext of $m := m_1 + m_2$ since

$$\begin{aligned}\frac{x(B - sA)}{N} &= \frac{x(r_1R + P_{m_1} + r_2R + P_{m_2} - sr_1Q - sr_2Q)}{N} \\ &= \frac{x(P_{m_1+m_2})}{N} = \frac{(m_1 + m_2)N}{N} = m_1 + m_2.\end{aligned}$$

□

Property 2 *Users can detect invalid ciphertexts.*

Proof: By definition (see also item 3 of Remark 3), a ciphertext c is of the form $(A, B) = (rQ, tQ + P_m) \in \langle Q \rangle \times \langle Q, P_1 \rangle$ (recall that $mP_1 = P_m$ and $\operatorname{ord} P_1 \mid N$; cf. Section 2.2). If s denotes a user's private key, we know that a ciphertext c is valid if and only if $t = rs \bmod \operatorname{ord} Q$, which in turn is equivalent to saying that $B - sA = P_m$ (recall that $P_m \notin \langle Q \rangle$ for all $0 \neq m \in \mathbb{Z}_N$). □

There are a couple of interesting properties of the cryptosystem that depend on the actual choice of the elliptic curve in the setup algorithm. We start with the detection of invalid ciphertexts for the master entity.

Property 3 *If* $\mathrm{DDH}_{\mathbb{Z}_{N^2}}$ *is hard in* $E(\mathbb{Z}_{N^2})$ *(even when the factorization of* N *is known), then the master entity, when given a user's public key,* cannot *decide whether a given ciphertext is a valid encryption under this public key or not.*

Proof: Assume that the master can detect invalid ciphertexts which are, by definition, of the form $(A, B) = (rQ, tQ+P_m) \in \langle Q \rangle \times \langle Q, P_1 \rangle$. Then we can use this detection algorithm to solve $\mathrm{DDH}_{\mathbb{Z}_{N^2}}$ as follows: Given a $\mathrm{DDH}_{\mathbb{Z}_{N^2}}$-tuple (Q, rQ, sQ, tQ), we just check the ciphertext $(A, B) = (rQ, tQ)$ for validity under the public key sQ. Clearly, we have:

$$(A, B) \text{ is valid} \iff (Q, rQ, sQ, tQ) \text{ is a valid } \mathrm{DDH}_{\mathbb{Z}_{N^2}}\text{-tuple.}$$

□

There are applications where the master entity should be able to check ciphertexts for validity as well. This is where pairings come into play. We will see that our cryptosystem is actually a nice application of *hidden pairings* – a notion introduced by Dent and Galbraith [34]. Therein, they present an identification scheme as a cryptographic application which was the only interesting application known until now. Unfortunately, since our scheme uses elliptic curves with certain properties in a non-black-box way, we cannot use the construction of a "hidden pairing"-group of [34] directly, but need to construct our own. Our construction is given in the following result that we prove in Section 5.3.5:

Lemma 6 *There is an efficient construction of an elliptic curve* E *over* $\mathbb{Z}_{N^2}$ *with properties as in Proposition 1 together with a point* $Q \in E(\mathbb{Z}_{N^2})$ *of large order dividing* M *such that*

1. *if* Q_1 *and* Q_2 *denote the natural reductions of* Q *to* $E(\mathbb{Z}_p)$ *and* $E(\mathbb{Z}_q)$*, respectively, we have that* $\operatorname{ord} Q = \operatorname{lcm}(\operatorname{ord} Q_1, \operatorname{ord} Q_2)$
2. *we can efficiently compute the 'reduced' Tate pairings* τ_p *and* τ_q *on* E *over* $\mathbb{Z}_p$ *and* $\mathbb{Z}_q$*, respectively.*

Since the Tate pairings τ_p and τ_q can only be computed if the factorization of N is known, they are called *hidden pairings.* Concerning the security of elliptic curves with properties as in the Lemma, we refer the reader to [48] and [34]. This Lemma has an interesting consequence on our cryptosystem:

Property 4 *Let* Q *be a point on an elliptic curve* E *over* $\mathbb{Z}_{N^2}$ *as in Lemma 6. If our cryptosystem uses* E *and* Q *in its public parameters, then the master entity can detect invalid ciphertexts under a given user's public key.*

Proof: Let $R = sQ$ be a user's public key and let $(A, B) = (rQ, tQ + P_m) \in \langle Q \rangle \times \langle Q, P_1 \rangle$ be a ciphertext. In order to check the validity of (A, B) under R, the master entity first uses the master secret M to compute the plaintext m (by using mDec). Since the master knows the factorization of N, it can now compute the reductions modulo p of Q, R, A and $T := B - P_m = tQ$ in $E(\mathbb{Z}_p)$ which we denote by Q_1, R_1, A_1 and T_1, respectively. Additionally, let $Q_2, R_2, A_2, T_2 \in E(\mathbb{Z}_q)$ be the respective reductions modulo q. Since the master can efficiently compute the 'reduced' Tate pairings τ_p and τ_q, respectively, it can check whether (Q_1, R_1, A_1, T_1) and (Q_2, R_2, A_2, T_2) are valid DDH-tuples in $E(\mathbb{Z}_p)$ and $E(\mathbb{Z}_q)$, respectively, in the usual way (see [45] and [89]). We have the relation that (Q_1, R_1, A_1, T_1) is a valid DDH-tuple in $E(\mathbb{Z}_p)$ if and only if $t = sr \bmod \operatorname{ord} Q_1$. An analogous relation holds for the prime q. Together, the Chinese Remainder Theorem over $\mathbb{Z}_{\operatorname{ord} Q}$ yields that (Q_1, R_1, A_1, T_1) and (Q_2, R_2, A_2, T_2) are valid DDH-tuples over their respective prime fields if and only if $t = rs \bmod \operatorname{ord} Q$ which in turn holds if and only if (A, B) is a valid ciphertext under R. $\square$

5.3.4. IND-CPA Security

Considering the fact that our cryptosystem is an additive variant of the ElGamal cryptosystem, it is rather obvious that it is IND-CPA secure under the $\mathrm{DDH}_{\mathbb{Z}_{N^2}}$-assumption. Proving the IND-CPA security of additively homomorphic cryptosystems boils down to proving that a random encryption is computationally indistinguishable from an encryption of 0 (cf. Section 4.2.2). In our cryptosystem, a random encryption has the form $(rQ, rR + P_m)$ with randomness $r \in \mathbb{Z}_{N^2}$ and random message $m \in \mathbb{Z}_N$. An encryption of 0, on the other hand, has the form (rQ, rR) for randomness $r \in \mathbb{Z}_{N^2}$. Now, if we write $X = rQ$ and $S = rR + P_m$ for randomness $r \in \mathbb{Z}_{N^2}$ and random message $m \in \mathbb{Z}_N$, we see that the IND-CPA security states: Given points $X, R \in \langle Q \rangle$ and given a random point S, decide whether $\log_Q(S) = \log_Q(X) \log_Q(R)$. This problem is the $\mathrm{DDH}_{\mathbb{Z}_{N^2}}$-problem, except that S is chosen from a larger group (and not only from $\langle Q \rangle$). However, $\mathrm{DDH}_{\mathbb{Z}_{N^2}}$ reduces to this more general problem.

In practice, from an adversary's point of view the situation is even worse, since without knowledge of the factorization of N it is extremely hard to find a point Q' on the curve at all (that differs from linear combinations of the publicly known points Q, R and P_1), because one would need to solve polynomial equations in $\mathbb{Z}_{N^2}$. It should be mentioned though that the security highly depends on the order of the point Q. Therefore, one should always take great care in the setup of the cryptosystem that the point Q really has large order (of about the same size as the prime factors of N).

Finally, we note that concerning the size of the security parameter of our scheme, we need to ensure that the bit length of the primes p and q is roughly 512 (at least). This yields a 1024 bit RSA-modulus and so we can assume that factoring such a large number is indeed hard in practice. Since solving discrete logarithms on elliptic curves over prime fields is assumed hard if the bit length of the order of the underlying prime field is about 180, having 512 bits here makes it reasonable to assume that the DLP is indeed hard on our chosen curves. Such a parameter setting is similar to the settings of [46] and [34], where it is argued that one can assume a high level of security while having efficient group operations on the curve at the same time.

5.3.5. Concrete Setup of the System's Parameters

The basic goal of this section is to prove Proposition 1 and Lemma 6, i.e., to give efficient constructions of elliptic curves with the properties as described in the respective claim. Since curves satisfying Lemma 6 will also satisfy Proposition 1, we start with the latter (Method 1) and then look at which of these curve additionally satisfy the Lemma (Methods 2 and 3).

Method 1: Random Curves.

Given a security parameter κ (which in practice will be of size 512), the fundamental idea is to choose two distinct, random primes p and q of about κ bits (so $N = pq$ is our RSA-modulus) together with two random elliptic curves E_1 and E_2 over $\mathbb{Z}_p$ and $\mathbb{Z}_q$, respectively. We require that both $E_1(\mathbb{Z}_p)$ and $E_2(\mathbb{Z}_q)$ have at least one large prime factor (of about κ bits) – so we discard all curves not having this property and repeat choosing random curves until we find two suitable elliptic curves. Then, by using standard techniques (i.e., considering E_1 and E_2 over $\mathbb{Z}_{p^2}$ and $\mathbb{Z}_{q^2}$, respectively (cf. Lemma 7), and then using the Chinese Remainder Theorem), we construct an elliptic curve E over $\mathbb{Z}_{N^2}$ such that $M := \mathrm{lcm}(\#E(\mathbb{Z}_p), \#E(\mathbb{Z}_q))$ has at least two large prime factors.

We remark that concerning the security of our cryptosystem, this way of constructing the elliptic curves prevents an attacker to exploit any particular structure of the used elliptic curve.

Likelyhood of hitting on such curves. One problem with this approach concerns the likelyhood of hitting on such curves by random sampling given a prime p. Since there is no final answer to this question in theory, we have to rely on a conjecture by Galbraith and McKee [47]: First, let us only consider elliptic curves E with prime order. It is conjectured that

$$\Pr\left[\#E(\mathbb{Z}_p) \text{ is prime}\right] \text{ is asymptotic to } c_p \frac{1}{\log p} \text{ as } p \to \infty,$$

where

$$c_p = \frac{2}{3} \prod_{l>2} \left(1 - \frac{1}{(l-1)^2}\right) \prod_{2<l|p-1} \left(1 + \frac{1}{(l+1)(l-2)}\right)$$

and the probability is over all random primes p and $(a,b) \xleftarrow{U} \mathbb{Z}_p^2 \setminus \{(a,b) \in \mathbb{Z}_p^2 \mid 4a^3+27b^2 = 0\}$. We ran some numerical tests ourselves (see Table 5.1) which confirm the conjecture in practice.

Bit length of p	64	128	192	256
$\Pr\left[\#E(\mathbb{Z}_p) \text{ is prime}\right]$	1.17 %	0.58 %	0.38 %	0.27 %

Table 5.1.: Numerical probability of hitting on a curve with prime order

As we have discussed before, we actually do not need the curve to have a large prime order, but only a nearly prime order. Therefore, we can optimize our search for elliptic curves by using a result by Lenstra [80] that small prime factors appear with a high probability. The idea is to fix a set S of small primes and allow $\#E(\mathbb{Z}_p)$ to be divisible by powers of $s \in S$. This increases the probability to hit on a curve with a large prime dividing the order by a huge factor (e.g., for orders of the form 2^k·prime this factor is about 3, while for orders of the form $2^k \cdot 3^l$·prime the factor is about 5.5 in our numerical results). This was also conjectured by Galbraith and McKee in [47] and our numerical tests give evidence for this conjecture (cf. Table 5.2).

Bit length of p	64	128	192	256
$\Pr\left[\#E(\mathbb{Z}_p) = 2^k \cdot \text{prime}\right]$	3.61 %	1.78 %	1.28 %	0.90 %
$\Pr\left[\#E(\mathbb{Z}_p) = 2^k \cdot 3^l \cdot \text{prime}\right]$	6.58 %	3.06 %	2.23 %	1.45 %

Table 5.2.: Numerical probability of hitting on a curve with nearly prime order

Concerning the efficiency of constructing curves as in Proposition 1, our experiments show that for an RSA-modulus of 512 bits (i.e., two primes of about 256 bits) it takes roughly 15 minutes using MAGMA on a single core of an Intel Xeon running at 2.5 GHz. For an 1024 bit RSA-modulus, it takes approximately 13 hours per curve. Allowing primes of up to three dividing the group order, we were able to generate five pairs of elliptic curves in approximately two days, while allowing prime factors of up to 13, this time halves (cf. Table 5.3). Since the Setup algorithm of our cryptosystem needs to be run only once, such an efficiency is reasonable in practice.

	$S = \{2, 3\}$	$S = \{2, 3, 5\}$	$S = \{2, 3, 5, 7, 11, 13\}$
Time	2d 6h 4m 20s	23h 15m 10s	1d 4h 43m 17s
Tested Curves	1298	552	687

Table 5.3.: Numerical results for the runtime of the Setup algorithm for $\log p = 512$ and 5 keys generated, where $S := \{\text{prime } \mathfrak{p} \mid \mathfrak{p} \text{ is allowed to divide } \#E(\mathbb{Z}_p)\}$

Performing our cryptosystem's setup. Recall that for a high level of security it is not enough to find suitable elliptic curves, we should also choose the point $Q \in E(\mathbb{Z}_{N^2})$ in the setup to be of large order dividing $M = \operatorname{lcm}(\#E(\mathbb{Z}_p), \#E(\mathbb{Z}_q))$ (cf. Section 5.3.4). The following two lemmata can be used in order to do this:

Lemma 7 *For a prime $p > 3$ and an elliptic curve E over $\mathbb{Z}_p$, we can efficiently construct an elliptic curve E' over $\mathbb{Z}_{p^2}$ such that $E'(\mathbb{Z}_{p^2})$ has order $\#E(\mathbb{Z}_p) \cdot p$ and the reduction from $\mathbb{Z}_{p^2}$ to $\mathbb{Z}_p$ induces a group homomorphism from $E'(\mathbb{Z}_{p^2})$ to $E(\mathbb{Z}_p)$.*

Proof: Let E be given by the short Weierstrass equation $y^2z = x^3 + axz^2 + bz^3$ over $\mathbb{Z}_p$. The existence of E' such that it reduces to E is simple, because it is sufficient to define E' by the same Weierstrass equation. Since the discriminant of E is invertible modulo p it also is modulo p^2, thus E' is an elliptic curve. Due to the geometric definition of the elliptic curve group law, the existence of the induced group homomorphism is obvious. It is left to be proven that this homomorphism is surjective.

Fix any finite point $P = (x_0 : y_0 : 1) \in E(\mathbb{Z}_p)$, then y_0 is a solution to the polynomial equation $0 = y^2 - (x_0^3 + ax_0 + b)$. In the case $y_0 \not\equiv 0 \pmod{p}$ there is a unique integer $0 \leq k < p$ such that $(y_0 + kp)^2 - (x_0^3 + ax_0 + b) \equiv 0 \pmod{p^2}$ by Hensel's lifting lemma. The new point $(x_0 : y_0 + kp : 1)$ obviously reduces to the initial point. In the case $y_0 \equiv 0 \pmod{p}$ we know that x_0 has been a solution for $0 = x^3 + ax + b \pmod{p}$. Since this polynomial cannot have any double roots we can find a solution $x_0 + kp$ for the same equation modulo p^2. This proves surjectivity.

To compute the order of $E'(\mathbb{Z}_{p^2})$ it is sufficient to compute the kernel of the reduction to $E(\mathbb{Z}_p)$. Obviously there are exactly p points $(kp : 1 : 0)$ for $0 \leq k < p$ on $E'(\mathbb{Z}_{p^2})$ that reduce to the point at infinity on $E(\mathbb{Z}_p)$. Thus $\#E'(\mathbb{Z}_{p^2}) = \#E(\mathbb{Z}_p) \cdot p$ holds due to the homomorphism theorem. □

Lemma 8 *Let $p > 3$ be a prime, E be an elliptic curve defined over $\mathbb{Z}_p$ and $P \in E(\mathbb{Z}_p)$ a point with $\gcd(\operatorname{ord} P, p) = 1$. Then the curve $E'(\mathbb{Z}_{p^2})$ constructed as in Lemma 7 contains a point P' of order $\operatorname{ord} P$.*

Proof: Let Q' be any preimage of P under the reduction map (Q' can be constructed following the proof of Lemma 7). By the homomorphism theorem, we have $\operatorname{ord} P \mid \operatorname{ord} Q'$. Multiplying both points with p permutes the subgroup generated by P on $E(\mathbb{Z}_p)$ and $P' := pQ'$ has order $\operatorname{ord} P' = \operatorname{ord} pP = \operatorname{ord} P$ since the order of P is coprime to p. □

Now, the construction of a point $Q \in E(\mathbb{Z}_{N^2})$ with large order dividing M, where E is a random curve such that M has at least two large prime factors (as in Proposition 1), works as follows:

1. Choose a random RSA-modulus $N = pq$ and random elliptic curves $E_1(\mathbb{Z}_p)$, $E_2(\mathbb{Z}_p)$ with nearly prime order as described before.
2. Pick points $P_1 \in E_1(\mathbb{Z}_p)$ and $P_2 \in E_2(\mathbb{Z}_q)$ of high order coprime to p and q.
3. Apply Lemma 7 and 8 to construct elliptic curves E_1' and E_2' with points $P_1' \in E_1'(\mathbb{Z}_{p^2})$ and $P_2' \in E_2'(\mathbb{Z}_{q^2})$ of high order.
4. Use the Chinese Remainder Theorem to merge E_1' and E_2' to a single curve E defined over the ring $\mathbb{Z}_{N^2}$. The lift Q of P_1' and P_2' has order $\operatorname{lcm}(\operatorname{ord} P_1, \operatorname{ord} P_2)$ and is the point used for the public parameters PP.

Method 2: Supersingular Curves.

A more efficient way to construct elliptic curves that satisfy Proposition 1 is by using supersingular curves E and particular RSA-moduli $N = pq$. For such curves it is known that over a prime field $\mathbb{Z}_p$ we have $\#E(\mathbb{Z}_p) = p + 1$. We note that the following discussion can be done for arbitrary supersingular elliptic curves, however, we restrict our attention to the following family of curves:

Lemma 9 (see [76]) *Let p be an odd prime with $p \equiv 2 \pmod 3$ and let $0 \neq b \in \mathbb{Z}_p$. Consider the elliptic curve $E : y^2 = x^3 + b$. Then, $E(\mathbb{Z}_p)$ is cyclic and $\#E(\mathbb{Z}_p) = p + 1$.*

So if we start with a *strong* prime p (i.e., $p + 1$ is not smooth) with $p \equiv 2 \pmod 3$ and setting E to be the curve given by the equation $y^2 = x^3 + b$ for some $0 \neq b \in \mathbb{Z}_p$, we ensure a large factor in $\#E(\mathbb{Z}_p) = p + 1$. To construct a strong prime p fulfilling the congruence condition it is possible to take a prime p' of the desired bit length κ such that $p := 6p' - 1$ is also prime.

Now, by using Lemmas 7 and 8, we can construct an elliptic curve together with a point Q of high order suitable for our cryptosystem in exactly the same way as we did in Method 1 (items 2 – 4 in the construction therein). We remark that constructing the elliptic curves in this way gives a very fast and easy setup of our system.

Additional property: Hidden pairing. Since supersingular elliptic curves have an embedding degree of at most $k = 6$, they allow for an efficient evaluation of the 'reduced' Tate pairing, which can then be used to solve DDH-challenges [45, 89]. Therefore, our just constructed elliptic curve E over $\mathbb{Z}_{N^2}$ has a hidden pairing [34], and we can efficiently solve DDH if the factorization of N is known. This proves Lemma 6.

Method 3: Complex Multiplication.

The CM method [10] allows us to construct elliptic curves E together with primes p and q such that E satisfies Proposition 1. Even more, by using extended algorithms [44], it is possible to construct E over $\mathbb{Z}_{N^2}$ such that it has a small embedding degree over the prime fields $\mathbb{Z}_p$ and $\mathbb{Z}_q$. This yields another construction satisfying Lemma 6.

5.3.6. Comparison to CS-Lite and BCP

We recall the two existing group homomorphic DD-PKE schemes CS-Lite [29] and BCP [20] and compare them with our cryptosystem presented in Section 5.3.2.

Definition 18 (CS-Lite) *CS-Lite is defined as follows:*

Setup: $\mathsf{Setup}(\kappa)$ *computes an RSA-modulus* $N = pq$ *in the* safe prime setting, *i.e., p and q are primes such that $p = 2p' + 1$ and $q = 2q' + 1$ where p' and q' are also prime (the bit length of p is κ). Furthermore, it chooses a random element $\mu \in \mathbb{Z}^*_{N^2}$, and computes $g := -\mu^{2N}$ mod N^2. It outputs the public parameters* $\mathsf{PP} := (N, g)$ *and the master secret key* $\mathsf{MK} := (p, q)$. *The plaintext space is $\mathcal{P} = \mathbb{Z}_N$ and the ciphertext space is $\widehat{\mathcal{C}} = \langle g \rangle \times \langle g \rangle$.*

Key Generation: $\mathsf{KeyGen}(\mathsf{PP})$ *chooses $a \in \mathbb{Z}_{\operatorname{ord} N^2/2}$ at random and computes $h := g^a$ mod N^2. It outputs the user's public key* $\mathsf{pk} := h$ *and secret key* $\mathsf{sk} := a$.

Encryption: $\mathsf{Enc}_{(\mathsf{PP},\mathsf{pk})}(m)$ *chooses a random value $r \in \mathbb{Z}_{N/4}$ and computes the ciphertext (A, B) as*

$$A = g^r \text{ mod } N^2 \text{ and } B = h^r(1 + mN) \text{ mod } N^2.$$

User Decryption: $\mathsf{Dec}_{(\mathsf{PP},\mathsf{sk})}(A, B)$ *outputs*

$$m = \frac{B/(A^a) - 1 \text{ mod } N^2}{N}.$$

Master Decryption: $\mathsf{mDec}_{(\mathsf{PP},\mathsf{MK},\mathsf{pk})}(A, B)$ *computes*

$$m = \frac{B^{\mathrm{cmf}(N)} - 1 \text{ mod } N^2}{N} \cdot \pi \text{ mod } N,$$

where π denotes the inverse of $\mathrm{cmf}(N)$ *modulo N.*[6]

The correctness of the scheme can be easily verified and it is semantically secure under the DDH Assumption in $\mathbb{Z}^*_{N^2}$ (see [29] for details).

Definition 19 (BCP) *BCP is defined as follows:*

Setup: $\mathsf{Setup}(\kappa)$ *computes an RSA-modulus $N = pq$ in the* safe prime setting*, i.e., p and q are primes such that $p = 2p' + 1$ and $q = 2q' + 1$ where p' and q' are also prime (the bit length of p is κ). Furthermore, it chooses a random element $g \in \mathbb{Z}^*_{N^2}$ of order $pp'qq'$ such that $g^{p'q'}$ mod $N^2 = 1+kN$ for $k \in \{1, \ldots, N-1\}$. It outputs the public parameters* $\mathsf{PP} := (N, k, g)$ *and the master secret key* $\mathsf{MK} := (p', q')$*. The plaintext space is $\mathcal{P} = \mathbb{Z}_N$ and the ciphertext space is $\widehat{\mathcal{C}} = \langle g \rangle \times \langle g \rangle$.*

Key Generation: $\mathsf{KeyGen}(\mathsf{PP})$ *chooses $0 \neq a \in \mathbb{Z}_{N^2}$ at random and computes $h := g^a$ mod N^2. It outputs the user's public key* $\mathsf{pk} := h$ *and secret key* $\mathsf{sk} := a$*.*

Encryption: $\mathsf{Enc}_{(\mathsf{PP},\mathsf{pk})}(m)$ *chooses a random value $r \in \mathbb{Z}_{N^2}$ and computes the ciphertext (A, B) as*

$$A = g^r \text{ mod } N^2 \text{ and } B = h^r(1 + mN) \text{ mod } N^2.$$

User Decryption: $\mathsf{Dec}_{(\mathsf{PP},\mathsf{sk})}(A, B)$ *outputs*

$$m = \frac{B/(A^a) - 1 \text{ mod } N^2}{N}.$$

Master Decryption: $\mathsf{mDec}_{(\mathsf{PP},\mathsf{MK},\mathsf{pk})}(A, B)$ *computes the partial discrete logarithms a mod N and r mod N (see Theorem 4 of [20] for details), and $\gamma := ar$ mod N. Furthermore, it computes*

$$D := \left(\frac{B}{g^\gamma}\right)^{\mathrm{cmf}(N)} = 1 + m\mathrm{cmf}(N)N \text{ mod } N^2,$$

and outputs

$$m = \frac{D - 1 \text{ mod } N^2}{N\mathrm{cmf}(N)} \text{ mod } N.$$

The correctness of the scheme is an immediate consequence of the correctness of CS-Lite, and the scheme is also semantically secure under the DDH Assumption in $\mathbb{Z}^*_{N^2}$ (cf. [20, Section 6.1]). The following remark compares the two schemes with ours:

[6] $\mathrm{cmf}(N)$ denotes the *Carmichael Function* of N. If $N = \prod_{i=1}^r p_i^{v_i}$ is the prime factorization of an odd number N, $\mathrm{cmf}(N)$ equals $\mathrm{lcm}(p_1^{v_1-1}(p_1 - 1), \ldots, p_r^{v_r-1}(p_r - 1))$.

Remark 4 *Comparison of the properties of CS-Lite, BCP and our scheme:*

1. *"Users can detect invalid ciphertexts"*

   CS-Lite: *Yes (see [20, Remark 7]).*

   BCP: *Yes (see [20, Remark 7]).*

   Our Scheme: *Yes (Property 2).*

2. *"The master entity can detect invalid ciphertexts"*

   CS-Lite: *No. This is only shown by giving a counterexample though (see [20, Remark 7]).*

   BCP: *No. Same example as for CS-Lite.*

   Our Scheme: *Either yes or (provably) no. The party who runs the cryptosystem's setup algorithm can decide: If a pairing-friendly elliptic curve is used, then the master* can *detect invalid ciphertexts, not only for a small class (Property 4). Otherwise, we* proved *(under* $\mathrm{DDH}_{\mathbb{Z}_{N^2}}$*) that the master* cannot *detect invalid ciphertexts (Property 3).*

3. *"User-Independent Double Decryption"*

   CS-Lite: *Yes. This is easily seen in Definition 18.*

   BCP: *No. The master decryption uses the user's public key h in an essential way as it first computes the partial discrete logarithm $a \bmod N$ of h.*

   Our Scheme: *Yes (Item 4 of Remark 3).*

6 Security Analysis of Homomorphic Encryption and Impossibility Results

6.1. Analyzing Existing Schemes

Currently[1], there does not exist a single IND-CPA secure homomorphic encryption scheme that deviates from the shift-type structure as defined in Definition 13. Therefore, our results have a strong impact on the security analysis of all existing schemes. In order to show how our results can be applied to existing schemes, we want to give a few examples taken from the class of GHE schemes and of FHE schemes.

6.1.1. ElGamal Encryption

In GIFT (cf. Section 5.1), we let $\widehat{\mathcal{C}} = \mathcal{C} = G \times G$ be the direct product of a cyclic group G (multiplicatively written) of prime order p with generator g. Since $\widehat{\mathcal{C}} = \mathcal{C}$, there is a trivial decision function $\delta : \widehat{\mathcal{C}} \to \{0, 1\}$ that always outputs 1. We set $\mathcal{P} := G$ and let $\mathcal{N} = \langle (g, h) \rangle$ be a subgroup of $\mathcal{C}$ generated by $(g, h) \in \mathcal{C}$ where $h := g^a$ for a secret $a \overset{U}{\longleftarrow} \mathbb{Z}_p$. Since $\mathcal{N} \cap R = \{(1, 1)\}$ where $R := \langle (1, g) \rangle \leq \mathcal{C}$ with $|R| = p$, we know that R is a system of representatives of $\mathcal{C}/\mathcal{N}$ (the isomorphism is given by $(1, g^r) \mapsto (1, g^r) \cdot \mathcal{N}$). Trivially, we have the efficient isomorphism $\varphi : \mathcal{P} \to R$ given by $g^r \mapsto (1, g^r)$. Also, we define an efficient epimorphism $\nu : \mathcal{C} \to R$ given by $(g^r, g^s) \mapsto (1, g^s \cdot g^{-ar})$. We have successfully defined the ingredients of the public key pk and the secret key sk as required in GIFT. Clearly, this instantiation of GIFT is ElGamal [36].

Next, we look at the three subgroup problems for this particular instantiation. First, recall that a triple of elements $(g_1, g_2, g_3) = (g^a, g^b, g^\gamma) \in G^3$ is called a Diffie-Hellman triple if $\gamma = a \cdot b$. Furthermore, one can easily check that $(g_2, g_3) \in \mathcal{N}$ if and only if (h, g_2, g_3) is a Diffie-Hellman triple. The Splitting Problem for $(\mathcal{C}, \mathcal{N}, R)$ is the computational Diffie-Hellman (CDH) problem for (h, c_1), since the splitting

[1] January 14, 2013

map $\sigma : \mathcal{C} \to R \times \mathcal{N}$ is given by $(c_1, c_2) \mapsto ((1, c_2 \cdot c_1^{-a}), (c_1, c_1^a))$. The Subgroup Membership Problem for $(\mathcal{C}, \mathcal{N})$ is the decisional Diffie-Hellman (DDH) problem for (h, c_1, c_2), and SOAP for $(\widehat{\mathcal{C}}, \mathcal{C}, \mathcal{N}, R, \delta)$ is the problem $\mathrm{DDH}^{\mathrm{SCDH}}$ where SCDH denotes the static computational Diffie-Hellman problem (cf. [84]).

In the ElGamal instantiation, we see that Theorem 6 states that ElGamal is IND-CPA secure if and only if DDH is hard, while Theorem 10 states that it is IND-CCA1 secure if and only if $\mathrm{DDH}^{\mathrm{SCDH}}$ is hard. The former characterization was proven in [107], while the latter was proven in [84], which are both trivial consequences of our general results.

6.1.2. Damgård's ElGamal Encryption

Again, we look at a concrete instantiation of GIFT. Here, we let $\widehat{\mathcal{C}} = G^3$ be the direct product of a prime ordered cyclic group G with generator g, and set $\mathcal{P} := G$. Furthermore, we choose random secrets $a, b \overset{U}{\longleftarrow} \mathbb{Z}_p$, compute the values $h := g^a, s := g^s$ and set $\mathcal{C} := \langle (g, h) \rangle \times G$. For a ciphertext $c = (c_1, c_2, c_3) \in \widehat{\mathcal{C}}$ we see that $c \in \mathcal{C} \iff c_2 = c_1^a$. Therefore, we have found an efficient decision function $\delta : \widehat{\mathcal{C}} \to \{0, 1\}$. Next, we set $\mathcal{N} := \langle (g, h, s) \rangle$ and $R := \langle (1, 1, g) \rangle$. Since $\mathcal{N} \cap R = \{(1, 1, 1)\}$ and $|R| = p$, we see that R is a system of representatives of $\mathcal{C}/\mathcal{N}$ (the isomorphism is given by $(1, 1, g^r) \mapsto (1, 1, g^r) \cdot \mathcal{N}$). We immediately derive an efficient isomorphism $\varphi : \mathcal{P} \to R$ given by $g^r \mapsto (1, 1, g^r)$ and define the map $\nu : \mathcal{C} \to R$ by $(g^r, h^r, g^t) \mapsto (1, 1, g^t \cdot g^{-br})$. We have successfully defined the ingredients of the public key pk and the secret key sk as required in GIFT and easily see that this instantiation is Damgård's ElGamal [30].

By considering the Splitting Problem for $(\mathcal{C}, \mathcal{N}, R)$ in this particular instantiation, we see that the splitting map $\sigma : \mathcal{C} \to R \times \mathcal{N}$ is given by $(c_1, c_2, c_3) \mapsto ((1, 1, c_3 \cdot c_1^{-b}), (c_1, c_2, c_1^b))$. Therefore, this Splitting Problem coincides with the CDH problem with parameters (g, s, g^r) for random $r \overset{U}{\longleftarrow} \mathbb{Z}_p$. In [84], this problem is denoted by CDEG. The Subgroup Membership Problem for $(\mathcal{C}, \mathcal{N})$ is the DDH problem with parameters (g, s, g^r, g^t) for random $r \overset{U}{\longleftarrow} \mathbb{Z}_p$ and $t \in \mathbb{Z}_p$; In [84], this problem is denoted by DDEG. Finally, SOAP for $(\widehat{\mathcal{C}}, \mathcal{C}, \mathcal{N}, R, \delta)$ is the problem $\mathsf{DDEG}^{\mathsf{SCDEG}}$ where SCDEG is the static CDEG (cf. [84]).

For this instantiation, i.e. for Damgård's ElGamal, Theorem 6 states that it is IND-CPA secure if and only if DDEG is hard, while Theorem 10 states that it is IND-CCA secure if and only if $\mathsf{DDEG}^{\mathsf{SCDEG}}$ is hard. The former characterization was proven in [30], while the latter was very recently proven in [84]. Again, we see that both results are immediate given our general characterizations.

6.1.3. Paillier Encryption

We briefly recall Paillier's homomorphic encryption scheme [93] by plugging the appropriate parameters into GIFT. Therefore, let $N = pq$ be an RSA-modulus and set $\widehat{\mathcal{C}} := \mathcal{C} := \mathbb{Z}_{N^2}^*$, $\mathcal{P} := \mathbb{Z}_n$ and $\mathcal{N} := \{r^N \bmod N^2 \mid r \in \mathbb{Z}_N^*\}$. Recall the following homomorphism

$$\mathcal{E}_g : \mathbb{Z}_N \times \mathbb{Z}_N^* \longrightarrow \mathbb{Z}_{N^2}^* \text{ with } \mathcal{E}_g(x, y) := g^x \cdot y^N \bmod N^2$$

for an element $g \in \mathbb{Z}_{N^2}^*$. It is known that $\mathcal{E}_g$ is an isomorphism if $g = 1 + N$ [23] or, more generally, if g is a multiple of N [93]. In these cases, there is a unique tuple $(x, y) \in \mathbb{Z}_N \times \mathbb{Z}_N^*$ for each $\omega \in \mathbb{Z}_{N^2}^*$ with $\mathcal{E}_g(x, y) = \omega$. The value x is called the *N-th residuosity class of ω (with respect to g)*, denoted by $[\![\omega]\!]_g$. The problem of computing $[\![\omega]\!]_g$ for given $\omega \in \mathbb{Z}_{N^2}^*$ and g is called the *Computational Composite Residuosity* (CCR) problem. Paillier showed that when the factorization of N is known, it is easy to compute $[\![\omega]\!]_g$ given ω and g. The problem of deciding whether $x = [\![\omega]\!]_g$, given ω, g and x, is called *Decisional Composite Residuosity* (DCR) problem.

In the following, we fix $g \in \mathbb{Z}_{N^2}^*$ such that $\mathcal{E}_g$ is an isomorphism and consider the subgroup $R := \langle h \rangle$ of $\mathcal{C}$ generated by $h := 1 + N$. In [29, Section 8.2.1], it is shown that $R = \{1 + aN \bmod N^2 \mid a \in \mathbb{Z}_N\}$ with $|R| = N = |\mathcal{C}/\mathcal{N}|$ (in particular, we can efficiently solve discrete logarithms in R due to this simple structure). In fact, R is a system of representatives of $\mathcal{C}/\mathcal{N}$:

Lemma 10 *Let $\pi : \mathcal{C} \to \mathcal{C}/\mathcal{N}$ be the canonical epimorphism, i.e. $\pi(c) := c \cdot \mathcal{N}$. Then, the map $\rho := \pi|_R : R \to \mathcal{C}/\mathcal{N}$ is an isomorphism, i.e. R is a system of representatives of $\mathcal{C}/\mathcal{N}$.*

Proof: Since ρ, as the restriction of π, is a homomorphism and $|R| = |\mathcal{C}/\mathcal{N}|$, it suffices to show that ρ is injective. Therefore, let $h^a \bmod N^2 \in \ker(\rho) = \mathcal{N} \cap R$ for some $a \in \mathbb{Z}_N$, i.e. there exists $z \in \mathbb{Z}_N^*$ such that $h^a \equiv z^N \pmod{N^2}$. But $\mathcal{N}$ is a group and so there exists an element $y \in \mathbb{Z}_N^*$ such that $y^N \cdot z^N \equiv 1 \pmod{N^2}$, i.e. $h^a \cdot y^N \equiv 1 \pmod{N^2}$. This in turn implies that $\mathcal{E}_h(a, y) \equiv 1 \pmod{N^2}$. But $\mathcal{E}_h$ is an isomorphism, i.e. $(a, y) = (0, 1) \in \mathbb{Z}_N \times \mathbb{Z}_N^*$ which implies $h^a \bmod N^2 = 1 \bmod N^2$ and so ρ is injective. $\square$

Trivially, we have the isomorphism $\varphi : \mathcal{P} \to R$ given by $m \mapsto 1 + mN \bmod N^2$. By [93, Lemma 1 and Lemma 2], we know that the "class function" $[\![\cdot]\!]_g : \mathbb{Z}_{N^2}^* \to \mathbb{Z}_N$ is a group epimorphism and so the mapping $\nu : \mathcal{C} \to R$ given by $c \mapsto h^{[\![c]\!]_g \bmod N} \bmod N^2$ is a group epimorphism. It can be efficiently computed when the factorization of N is known [93, Theorem 1]. Since we can solve discrete logarithms in R very efficiently, computing $\nu(c)$ is equivalent to computing $[\![c]\!]_g$.

We have successfully defined the public key $\mathsf{pk} = (N, g)$ and the secret key $\mathsf{sk} = (p, q)$ in GIFT. The resulting scheme is Paillier's homomorphic encryption scheme [93]. Observe that the splitting map $\sigma : \mathcal{C} \to R \times \mathcal{N}$ is given by $\omega \mapsto (\llbracket\omega\rrbracket_g, \omega \cdot g^{-\llbracket\omega\rrbracket_g})$. We immediately see that the SP in this instantiation is the CCR problem. Furthermore, $\mathcal{N}$ contains by definition all elements $r^N \bmod N^2$ for $r \in \mathbb{Z}_N^*$. Therefore, the SMP for $(\mathcal{C}, \mathcal{N})$ is the DCR problem. As a consequence of Theorems 10 and 6, we get the following characterizations of the security of Paillier's scheme:

Theorem 14 (Security Characterization of Paillier) *Paillier's scheme is IND-CCA1 (resp. IND-CPA) secure if and only if* $\mathrm{DCR}^{\mathsf{SCCR}}$ *(resp.* DCR*) is hard.*

We note that $\mathrm{DCR}^{\mathsf{SCCR}}$ is a new (though naturally arising) problem and so a thorough analysis of its hardness is advisable. However, assuming its hardness seems reasonable and gives the first proof of the IND-CCA1 security of Paillier's scheme.

Damgård and Jurik proposed an extension of Paillier's scheme to a generalized group structure [33]. We stress that we can achieve a similiar characterization of the IND-CCA1 security of their scheme by applying similar thoughts as the above.

6.1.4. vDGHV Encryption

We briefly recall the SWHE scheme by van Dijk et al. [110] and show that it is shift-type homomorphic, meaning that we can apply our IND-CPA security characterization of Theorem 7, which then also extends to its FHE transformation by Theorem 8 (resp. Corollary 1). To get rid of a very voluminous and confusing introduction of parameters, we will fix a particular setup of parameters in the key generation phase and note that all of the following can be done in a more general fashion (see [110]). Also, we will focus here on the encryption algorithm only and refer the reader to [110] for details on the remaining algorithms for decryption and evaluation. For the security parameter κ, we fix:

$$\rho := \kappa, \rho' := 2\kappa, \eta \in \rho' \cdot \Theta(\kappa \log^2 \kappa), \gamma \in \omega(\eta^2 \log \kappa) \text{ and } \tau := \gamma + \kappa.$$

The secret key sk of the scheme is $p \overset{U}{\longleftarrow} (2\mathbb{Z} + 1) \cap [2^{\eta-1}, 2^\eta)$ and we define the following efficiently sampleable distribution

$$\mathcal{D}_{\gamma,\rho}(p) := \left\{ x = pq + r \mid q \overset{U}{\longleftarrow} \mathbb{Z} \cap [0, 2^\gamma/p), r \overset{U}{\longleftarrow} \mathbb{Z} \cap (-2^\rho, 2^\rho) \right\}.$$

With this notation, we let $\mathsf{pk} = (x_0, \ldots, x_\tau)$ be the public key with $x_i \overset{U}{\longleftarrow} \mathcal{D}_{\gamma,\rho}(p)$ for all $i = 0, \ldots, \tau$ whereas the x_i's are relabeled such that x_0 is the largest (if x_0 is even or $x_0 \bmod p$ is odd, then restart). The plaintext space is $\{0, 1\}$.

The encryption algorithm takes the public key pk and a plaintext $m \in \{0, 1\}$ as input and outputs a ciphertext $c := [(m + 2r + 2\sum_{i \in S} x_i) \bmod x_0]$ whereas S is a random subset of $\{1, \ldots, \tau\}$ and $r \overset{U}{\longleftarrow} \mathbb{Z} \cap (-2^{\rho'}, 2^{\rho'})$. In the notation of the shift-type homomorphic definition (see Definition 13), we have that $\widehat{\mathcal{C}}$ is the ring $\mathbb{Z}_{x_0}$ and

$$\mathcal{N} = \left\{ 2(r + \sum_{i \in S} x_i) \bmod x_0 \mid r \in \mathbb{Z} \cap (-2^{\rho'}, 2^{\rho'}), S \subseteq \{1, \ldots, \tau\} \right\}.$$

The injective homomorphism φ is given by $m \mapsto m \bmod x_0$, which is even a ring homomorphism. Encryption is then given by $\varphi(m) + n$ where $n \in \mathcal{N}$. Concerning the homomorphic property in Definition 13, we need to make more effort:

It is shown in [110, Lemma 3.3] that the scheme is homomorphic for Boolean circuits with the property that for any $\alpha \geq 1$ and any set of integer inputs all less than $2^{\alpha(\rho'+2)}$ in absolute value, it must hold that the output of the *generalized circuit* (same circuit where the ADD- and MULT-gates are applied to integers instead of bits) has absolute value at most $2^{\alpha(\eta-4)}$. Furthermore, it is shown in [110, Lemma 3.5] that if $f(x_1, \ldots, x_t)$ is the multivariate polynomial of degree d computed by the generalized circuit of a given boolean circuit C with t inputs, then the scheme is homomorphic for C if $|\bar{f}| \cdot (2^{\rho'+2})^d \leq 2^{\eta-4}$, where $|\bar{f}|$ is the l_1 norm of the coefficient vector of f. In respect of the homomorphic property of Definition 13, it suffices to show that the scheme is homomorphic for the boolean circuit C_{ADD} that consists of a single ADD-gate only. Clearly, the multivariate polynomial that is computed by the generalized circuit of C_{ADD} is $f(x_1, x_2) = x_1 + x_2$ and has degree $d = 1$ with $|\bar{f}| = 2$. Therefore, the scheme is homomorphic for this circuit if we have

$$2^{\rho'+3} \leq 2^{\eta-4}, \text{ which in turn is fulfilled if } \eta \geq \rho' + 7.$$

This final condition holds as $\eta \in \rho' \cdot \Theta(\kappa \log^2 \kappa)$. In total we have shown that the above scheme indeed is shift-type homomorphic.

6.2. Impossibility Results on Group Homomorphic Encryption

Let GGen be a PPT algorithm that takes a security parameter κ as input and outputs descriptions $(G, \mathcal{N})$ where $\mathcal{N}$ is a non-trivial, proper subgroup of a finite group G with an additional algorithm for the efficient sampling from $G \backslash \mathcal{N}$ (cf. Section 4.1.1). Now, assume that for any such algorithm GGen, we can construct an algorithm $\mathcal{A}$ that breaks the hardness of SMP relative to GGen. In particular, for a given group homomorphic encryption scheme $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ this means that we have an algorithm $\mathcal{A}$ that breaks the hardness of SMP relative to KeyGen. However, by Theorem 5, this implies that we can construct an algorithm that breaks the IND-CPA security of $\mathcal{E}$. Since we had no restriction on the encryption scheme $\mathcal{E}$, this would imply that *any* group homomorphic encryption scheme $\mathcal{E}$ is insecure in terms of IND-CPA.

In this section, we first show that if a GHE scheme is instantiated with a linear code as the ciphertext group, it is insecure in terms of IND-CPA. We show this by giving an attack on the corresponding SMP. Secondly, we show that *any* GHE scheme is insecure in the quantum world, at least for the Abelian case. More precisely, we construct a quantum algorithm that breaks any instance of SMP, if G is Abelian.

In the following, we consider outputs $(G, \mathcal{N})$ of GGen in the case where G is Abelian. Before we prove our impossibility results, we need some results on the sampling of a set of generators of $\mathcal{N}$.

6.2.1. Sampling Group Generators Uniformly at Random

We start this section by assuming that the sampling algorithm for $\mathcal{N}$ samples *uniformly at random.* In this case, we can obtain a set of generators for $\mathcal{N}$ by sampling polynomially (in the base-2 logarithm of the order of $\mathcal{N}$) many times from $\mathcal{N}$. In order to show this, we need the following technical Lemma.

Lemma 11 *For any $k > 0$ ($k \in \mathbb{N}$) we have:*

$$\left(1 - \prod_{j=0}^{k-1}\left(1 - 2^{j-k}\right)\right)^{k+1} \leq \frac{1}{4}.$$

Proof: We set $f(k) := 1 - \prod_{j=0}^{k-1}\left(1 - 2^{j-k}\right)$. Now, it is obvious that if $k = 1$, then $f(k)^{k+1} = \frac{1}{4}$. We prove the Lemma by showing that the function $f(k)^{k+1}$ is strictly monotonically decreasing. We do so by showing that the first derivative of this function is strictly smaller than 0. Therefore, we start by looking at the derivative of $f(k)$. A sequential application of the product rule yields:

$$\frac{d(f(k))}{dk} = -\ln 2 \cdot \sum_{j=0}^{k-1}\left(2^{j-k}\prod_{\substack{i=0\\ i\neq j}}^{k-1}\left(1 - 2^{i-k}\right)\right) < 0.$$

While using the chain rule on $f(k)^{k+1}$ yields

$$\frac{d(f(k)^{k+1})}{dk} = f(k)^k \cdot \frac{d(f(k))}{dk} \cdot (k+1) + \ln(f(k)) \cdot f(k)^{k+1}.$$

But, since $0 < f(k) < 1$, we have that the second summand is < 0, while the first summand is < 0 as well, since $\frac{d(f(k))}{dk} < 0$. In total, this shows that $f(k)^{k+1}$ indeed is monotonically decreasing, which proves the Lemma. $\square$

We use Lemma 11 to compute the probability of sampling a small set of generators for the group $\mathbb{Z}_2^k$:

Theorem 15 *For any $k \in \mathbb{N}$ we have:*

$$\Pr_{x_1,\ldots,x_{k+1} \xleftarrow{U} \mathbb{Z}_2^k} \left[\langle x_1, \ldots, x_{k+1} \rangle = \mathbb{Z}_2^k \right] \geq \frac{3}{4}.$$

Proof: Notice that a generating set for $\mathbb{Z}_2^k$ must contain at least k elements. If we sample k elements at random, they generate $\mathbb{Z}_2^k$ if and only if they are linearly independent as vectors over GF(2). We do the following easy combinatorial thought:

- for $k = 1$ we only have two elements (0 and 1), so the success probability (i.e., the probability of generating $\mathbb{Z}_2^1$ by sampling just one element) is $\frac{1}{2}$;
- for $k = 2$ we succeed in spanning all the group if and only if the first element is nonzero (probability $1 - \frac{1}{4}$), and the second element is not linearly dependent to the first, i.e., it does not belong to the subgroup spanned by the first element (which has $2^{k-1} = 2$ elements over a total of $2^k = 4$ elements in the group, so the probability of this happening is $\frac{1}{2} = 1 - \frac{1}{2}$ and hence the total success probability is $\left(1 - \frac{1}{4}\right)\left(1 - \frac{1}{2}\right)$);
- analogously, for $k = 3$ the probability of generating all $\mathbb{Z}_2^3$ by sampling at random 3 elements is $\left(1 - \frac{1}{8}\right)\left(1 - \frac{1}{4}\right)\left(1 - \frac{1}{2}\right)$.

In general then, the probability of *not* spanning all $\mathbb{Z}_2^k$ with k random elements is:

$$1 - \prod_{j=0}^{k-1} \left(1 - 2^{j-k}\right).$$

But if we sample $k+1$ elements, instead, there are $\binom{k+1}{k} = k+1$ possible different subsets of k elements. The probability that all $k+1$ elements do not span the whole group is then the probability that *none* of these subsets spans the group, that is:

$$\left(1 - \prod_{j=0}^{k-1} \left(1 - 2^{j-k}\right)\right)^{k+1},$$

and the Theorem follows from Lemma 11. □

This Theorem immediately gives us a lower bound on the probability of sampling a generating set of an arbitrary group.

Corollary 5 *For an Abelian group G with $k = \lceil \log_2(|G|) \rceil$, we have:*

$$\Pr_{x_1,\ldots,x_{k+1} \xleftarrow{U} G} \left[\langle x_1, \ldots, x_{k+1} \rangle = G \right] \geq \frac{3}{4}.$$

Proof: We prove this along the lines of [94, Theorem 2.1] and consider the following random process: Pick a uniform group element $x_1 \in G$. If $H_1 = \langle x_1 \rangle \neq G$, pick a uniform group elenwnt $x_2 \in G$. If $H_2 = \langle x_1, x_2 \rangle \neq G$, pick another group element, etc. Clearly, for a given i, we have that

$$\Pr_{x_{i+1} \xleftarrow{U} G} [H_i \neq H_{i+1}] = 1 - \frac{|H_i|}{|G|}.$$

In this case, it holds that $|H_{i+1}|/|H_i| \geq 2$, with an equality when $G \cong \mathbb{Z}_2^k$. Therefore, if (in the above random process) τ denotes the first time we generate the whole group G, this shows that

$$\Pr_{x_1,\ldots,x_\tau \xleftarrow{U} G} [\langle x_1, \ldots, x_\tau \rangle = G] \text{ is minimized if } G = \mathbb{Z}_2^k.$$

But by Theorem 15, we know that this probability is $\geq 3/4$ if $G = \mathbb{Z}_2^k$ and $\tau \geq k+1$, finishing the proof. $\square$

We stress that it is well-known that polynomially many samples suffice to get a generating set of $\mathbb{Z}_2^k$. Our lower bound on the probability, however, improves all previously known bounds [94].

6.2.2. Breaking the SMP with Uniform Sampling

As in the previous section, we assume that the sampling algorithm for $\mathcal{N}$ samples with the uniform distribution.

Linear Codes

Let $\mathbb{F}$ be a prime field. Recall that a *linear code* of length n and rank k is a linear subspace $C \subseteq \mathbb{F}^n$ of the vector space $\mathbb{F}^n$ such that $\dim(C) = k$. We want to show that if G is a linear code, then the SMP (with uniform sampling) is always easy to break.

Theorem 16 *Let $(G, \mathcal{N})$ be an output of* GGen *such that the sampling algorithm of $\mathcal{N}$ samples uniformly at random and $\mathcal{N}$ is a k-dimensional linear subspace of $\mathbb{F}^n$. Then, there exists a PPT algorithm $\mathcal{A}$ that breaks the* SMP *for $(G, \mathcal{N})$.*

Proof: We construct a generating set $(h_1, \ldots, h_s)$ of $\mathcal{N}$ as follows: We sample $s := k+1$ times uniformly at random from $\mathcal{N}$. If $(h_1, \ldots, h_s)$ is linearly dependent, we sample again until we get a linearly independent tuple. This process terminates after polynomially many steps by Corollary 5. These vectors $h_1, \ldots, h_s$ of $\mathcal{N}$ are vectors in $\mathbb{F}^n$. Therefore, when given an arbitrary element $g \in G$, we can efficiently compute the rank r of the matrix $(g, h_1, \ldots, h_s)$. If $r = k$, we know that $g \in \mathcal{N}$, otherwise $g \notin \mathcal{N}$. $\square$

This Theorem proves that GHE schemes with a linear code as a ciphertext group and where the output distribution of the encryption algorithm is the uniform distribution, cannot be IND-CPA secure.

Corollary 6 *Let $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ be a* GHE *scheme such that the set of encryptions $\mathcal{C}$ is a k-dimensional linear subspace of $\mathbb{F}^n$ and such that the output distribution of the encryption algorithm is the uniform distribution. Then, $\mathcal{E}$ is insecure in terms of IND-CPA (relative to* KeyGen*).*

In particular this holds if $\mathcal{C}$ (or the ciphertext space $\widehat{\mathcal{C}}$)[2] *is a linear code.*

Proof: By Theorem 5, it suffices to break the SMP relative to KeyGen. But this is exactly what Corollary 6 does. □

We remark that the same attack also works in the following settings, making the impossibility result more general:

1. If $\mathcal{E}$ is also homomorphic with respect to the scalar multiplication in $V = \mathbb{F}^n$ (i.e. decryption is $\mathbb{F}$-linear), we do not need the restriction that $\mathbb{F}$ is a prime field.

2. Corollary 6 also holds for arbitrary n-dimensional $\mathbb{F}$-vector spaces V, if there is a (publicly known) efficiently computable isomorphism from V to $\mathbb{F}^n$ (the inversion must be efficiently computable as well). We note that this is not always the case, as is seen by considering the ElGamal encryption scheme (see Section 6.1.1):

   Certainly, the ciphertext group $\mathcal{C} = G \times G$ of ElGamal is a 2-dimensional $\mathbb{F}_p$-vector space, where G is an additive cyclic group of prime order p. In addition, it is easily seen that the group $\mathcal{C}_0$ of all encryptions of 0 is in fact an $\mathbb{F}_p$-subspace of $\mathcal{C}$. So, if there would be a publicly known and efficiently computable isomorphism $F : \mathcal{C} \to \mathbb{F}_p^2$, Corollary 6 would break ElGamal. Fortunately, we can prove that no such isomorphism can exist:

   Claim. If there exists an efficient isomorphism $F : \mathcal{C} \to \mathbb{F}_p^2$, we can efficiently solve discrete logarithms in G (which is supposed to be hard in the setting of ElGamal).

   Proof: Assume that $F : \mathcal{C} \to \mathbb{F}_p^2$ is an efficiently computable isomorphism. Let $0 \neq g \in G$ be an arbitrary element of G, i.e., $G = \langle g \rangle$. Now, for a given $h \in G$, we can compute $\log_g(h)$ by computing $\log_{F(g,g)}(F(h,h))$. This works since F

[2] $\mathbb{F}$ is a prime field and so the notion of subgroups coincides with the notion of $\mathbb{F}$-subspaces (see [78, Theorem 2.1.8(b)]). Since we assume $\mathcal{C}$ to be a subgroup of $\widehat{\mathcal{C}}$, it follows that if $\widehat{\mathcal{C}}$ is a linear code, then $\mathcal{C}$ is a linear code as well.

is $\mathbb{F}_p$-linear (i.e., $F(h,h) = \log_g(h) \cdot F(g,g)$ and so $\log_{F(g,g)}(F(h,h)) = \log_g(h)$) and solving discrete logarithms in the additive group $\mathbb{F}_p^2$ is easy. $\square$

The Quantum World

It is well-known that Watrous' modification [112] of the famous order-finding quantum algorithm can efficiently find the order of an Abelian group, given that we have its description by a set of generators.

Theorem 17 (Quantum Order-Finding Algorithm with Generators [112]) *Let G be a finite Abelian group with $k = \lceil \log_2(|G|) \rceil$. Then, there exists a quantum algorithm which, given a generating set of G and an error probability ε as an input, outputs the order of G with probability at least $1-\varepsilon$ in time $o(\mathtt{poly}(k + \log_2(1/\varepsilon)))$.*

This Theorem already is sufficient to break the hardness of SMP (relative to GGen), *if* the description of $\mathcal{N}$ contains a set of generators, as the next Theorem shows.

Lemma 12 (Quantum Attack on SMP with Generators) *Let $(G, \mathcal{N})$ be the output of $\mathsf{GGen}(\kappa)$, for some security parameter κ, such that $\mathcal{N}$ contains a set of generators $g_1, \ldots, g_r$. Since GGen is a PPT algorithm, this implies that $k = k(\kappa) = \lceil \log_2(|\mathcal{N}|) \rceil$ is a polynomial in κ. There exists a quantum algorithm which, given $g_1, \ldots, g_r$ (i.e., the description of $\mathcal{N}$), breaks the hardness of* SMP *with probability at least $(1-\varepsilon)^2$ in time $o(\mathtt{poly}(k + \log_2(1/\varepsilon)))$.*

Proof: Let z denote the challenge in the SMP game, i.e., $z \in G \setminus \mathcal{N}$ if $b = 1$, and $z \in \mathcal{N}$ otherwise. Since $\mathcal{N}$ contains a set of generators $g_1, \ldots, g_r$, we can run the quantum algorithm in Theorem 17 twice: the first time on the generating set and the second time on the generating set plus the element z. Provided that both runs succeed, we have that $z \in \mathcal{N}$ (i.e., $b = 0$) if and only if the two subgroup orders, obtained from the two algorithm runs, are the same. But both runs succeed with probability $(1-\varepsilon)^2$. This proves the Lemma. $\square$

Together with Corollary 5, this Lemma gives us a way to break the SMP if the sampling algorithm for $\mathcal{N}$ samples uniformly at random.

Theorem 18 *Let $(G, \mathcal{N})$ be the output of $\mathsf{GGen}(\kappa)$ with $k = \lceil \log_2(|\mathcal{N}|) \rceil$, for some security parameter κ, such that the sampling algorithm in the description of $\mathcal{N}$ samples* uniformly at random *from $\mathcal{N}$. Then, there exists a quantum algorithm which breaks the hardness of* SMP *with probability at least $\frac{3}{4}(1-\varepsilon)^2$ in time $o(\mathtt{poly}(k + \log_2(1/\varepsilon)))$, and by sampling only $k+1$ times from $\mathcal{N}$.*

Proof: This is Lemma 12 together with Corollary 5. $\square$

This gives the following impossibility result for GHE schemes.

Corollary 7 *Let $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ be an IND-CPA secure group homomorphic encryption scheme with Abelian plaintext and ciphertext groups, such that the distribution on $\mathcal{C}_1$ is the uniform distribution. Then, there exists a quantum PPT algorithm that breaks the security of $\mathcal{E}$ with non-negligible probability.*

Proof: This is implied by Theorem 18 and Theorem 5. □

6.2.3. Breaking the SMP with Arbitrary Sampling

We show an extension of the previous section to the general case, where the description of $\mathcal{N}$ only contains a sampling algorithm for $\mathcal{N}$ with *unknown/arbitrary distribution $\mathcal{D}$*. The main observation to make this extension work (we prove this in Lemma 13) is that in order to break the SMP for $(G, \mathcal{N})$ we actually do not have to sample a generating set for the whole group $\mathcal{N}$, but rather for the *"computationally observable"* subgroup $\mathcal{N}_{\text{comp}}$ of $\mathcal{N}$. This subgroup is the *largest* subgroup of $\mathcal{N}$ such that when sampling polynomially many elements from $\mathcal{N}$ according to $\mathcal{D}$, these elements generate the subgroup $\mathcal{N}_{\text{comp}}$ with high probability.

Definition 20 (Computationally Observable Subgroup) *For a finite Abelian group $\mathcal{N}$ (generated in PPT, with parameter κ, and sampling algorithm with distribution $\mathcal{D}$), we define the* computationally observable subgroup $\mathcal{N}_{\text{comp}}$ of $\mathcal{N}$ *as the largest subgroup J of $\mathcal{N}$ such that:*

$$\exists\ p = \mathtt{poly}(\kappa) : \Pr_{x_1,\ldots,x_p \xleftarrow{\mathcal{D}} \mathcal{N}} [\langle x_1, \ldots, x_p \rangle = J] \ \textit{ is non-negligible in } \kappa.$$

We say that $\mathcal{N}$ is computationally observable *iff $\mathcal{N} = \mathcal{N}_{\text{comp}}$.*

Note that, for the SMP to be hard, the order of $\mathcal{N}_{\text{comp}}$ is required to be exponentially large in κ. We show that if we modify the original SMP game by sampling from $\mathcal{N}_{\text{comp}}$ instead of the whole group $\mathcal{N}$, the difference between the winning probability of the resulting game and of the original game is upper bounded by some negligible function in κ. In order to prove this, we define a new distribution $\mathcal{D}_{\text{comp}}$ obtained by $\mathcal{D}$ by 'cutting away' any occurrence of elements which are in $\mathcal{N}$ but not in $\mathcal{N}_{\text{comp}}$ (by defining the probability on these elements as zero).

Definition 21 (Computationally Observable Distribution) *Let $\mathcal{D} = (\pi_x)_{x \in \mathcal{N}}$ be an arbitrary distribution on $\mathcal{N}$, where $\pi_x = \Pr_{z \xleftarrow{\mathcal{D}} \mathcal{N}} [z = x]$. Then we define the* computationally observable distribution $\mathcal{D}_{\text{comp}}$ *as $\mathcal{D}_{\text{comp}} := (\pi'_x)_{x \in \mathcal{N}}$, where*

$$\pi'_x := \begin{cases} 0 & \textit{if } x \in \mathcal{N} \setminus \mathcal{N}_{\text{comp}}, \\ \pi_x + \frac{\sigma_{\mathcal{D}}}{|\mathcal{N}_{\text{comp}}|} & \textit{if } x \in \mathcal{N}_{\text{comp}}, \end{cases}$$

with $\sigma_{\mathcal{D}} := \sum_{y \in \mathcal{N} \setminus \mathcal{N}_{\text{comp}}} \pi_y$.

Then we show that the new distribution is statistically indistinguishable from the original one:

Lemma 13 *The variational distance between $\mathcal{D}$ and $\mathcal{D}_{\text{comp}}$ is negligible.*

Proof: First we compute the variational distance between the two distributions, and then we show that this distance is negligible in κ:

$$|\mathcal{D} - \mathcal{D}_{\text{comp}}| = \sum_{x \in \mathcal{N}} |\pi_x - \pi'_x| = \sum_{y \in \mathcal{N} \setminus \mathcal{N}_{\text{comp}}} \pi_y + \sum_{x \in \mathcal{N}_{\text{comp}}} \frac{\sigma_{\mathcal{D}}}{|\mathcal{N}_{\text{comp}}|} = 2\sigma_{\mathcal{D}}.$$

Now, assume $\sigma_{\mathcal{D}}$ is non-negligible in κ and sample p elements in $\mathcal{N}$ from distribution $\mathcal{D}$ (where p is a polynomial from Definition 20). By definition of $\mathcal{N}_{\text{comp}}$ there exists a non-negligible function μ such that:

$$\Pr_{x_1,\ldots,x_p \overset{\mathcal{D}}{\leftarrow} \mathcal{N}} [\langle x_1, \ldots, x_p \rangle = \mathcal{N}_{\text{comp}}] \geq \mu. \tag{6.1}$$

At this point we sample an additional element x_{p+1} from $\mathcal{D}$, and by definition of $\sigma_{\mathcal{D}}$ we have:

$$\Pr_{x_{p+1} \overset{\mathcal{D}}{\leftarrow} \mathcal{N}} [x_{p+1} \in \mathcal{N} \setminus \mathcal{N}_{\text{comp}}] \geq \sigma_{\mathcal{D}}, \tag{6.2}$$

Then, equations (6.1) and (6.2) imply that

$$\Pr_{x_1,\ldots,x_{p+1} \overset{\mathcal{D}}{\leftarrow} \mathcal{N}} [\langle x_1, \ldots, x_{p+1} \rangle = T \text{ with } T \gneqq \mathcal{N}_{\text{comp}}] \geq \mu\sigma_{\mathcal{D}}.$$

which is non-negligible. This is a contradiction, since $\mathcal{N}_{\text{comp}}$ was defined as the *largest* subgroup of $\mathcal{N}$ with such a property. □

Now, by definition of $\mathcal{N}_{\text{comp}}$, we find a generating set of $\mathcal{N}_{\text{comp}}$ with non-negligible probability in a polynomial number of calls to the sampling algorithm by definition, hence in probabilistic polynomial time.

Linear Codes

As a trivial Corollary of Lemma 13 and Theorem 16, we have:

Corollary 8 *Let $(G, \mathcal{N})$ be an output of* GGen *such that $\mathcal{N}$ is a k-dimensional linear subspace of $\mathbb{F}^n$. Then, there exists a PPT algorithm $\mathcal{A}$ that breaks the* SMP *for $(G, \mathcal{N})$.*

This in turn immediately implies our general impossibility result on the use of linear codes as ciphertext spaces.

Theorem 19 *Let $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ be a* GHE *scheme such that the set of encryptions $\mathcal{C}$ is a k-dimensional linear subspace of $\mathbb{F}^n$. Then, $\mathcal{E}$ is insecure in terms of IND-CPA (relative to* KeyGen*). In particular this holds if $\mathcal{C}$ (or the ciphertext space $\widehat{\mathcal{C}}$) is a linear code.*

The Quantum World

Additionally, we can also extend our quantum results. An easy consequence from Lemma 13 and Theorem 12 is the following result.

Corollary 9 (Quantum Attack on SMP with Arbitary Sampling) *Let* $(G, \mathcal{N})$ *be the output of* $\mathsf{GGen}(\kappa)$ *with* $k = \lceil \log_2(|\mathcal{N}|) \rceil$*, for some security parameter* κ*. We denote the distribution of the sampling algorithm contained in the description of* $\mathcal{N}$ *by* $\mathcal{D}$*. Then, there exists a polynomial* $p(\kappa)$ *and a quantum algorithm which breaks the hardness of* SMP *with probability at least* $\frac{3}{4}(1-\varepsilon)^2$ *in time* $o(\mathtt{poly}(k + \log_2(1/\varepsilon)))$*, and by sampling only* $p(\kappa)$ *times from* $\mathcal{N}$*.*

Finally, this Theorem (together with Theorem 5) implies our main result.

Theorem 20 (General Impossibility of GHE in the Quantum World) *Let* $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *be an IND-CPA secure* GHE *scheme with Abelian plaintext and ciphertext groups. Then, there exists a quantum PPT algorithm that breaks the security of* $\mathcal{E}$ *with non-negligible probability.*

We point out that Theorem 17 actually holds for general solvable groups [112], which is why we believe that also Theorem 20 can be extended to this more general case as well. However, we do not see any practical relevance of such an extension since one usually requires some form of commutativity in practical applications.

7 Secure and Efficient Outsourcing of Multiparty Computations

In this chapter, we consider the following scenario:

1. A set of n mutually distrusting clients $P_1, \ldots, P_n$ (the number n may change dynamically over time), each having its own public and private key pair, encrypt data under their respective public keys and store these encryptions on a server $\mathcal{C}$.

2. Any *dynamically chosen* function (i.e., the function does not need to be specified at the time data is encrypted) should be computed by $\mathcal{C}$ on the clients' data, while all inputs and intermediate results remain private.

3. Due to the fact that clients are not always online in practice, $\mathcal{C}$ needs the ability to compute these functions *without any interaction* of the clients. In particular, this also concerns the clients' retrieval of results.

4. Once online, individual clients can retrieve the result while the server learns nothing at all.

For the first time, we present a simple and efficient solution to this scenario, meaning that we give a general-purpose cryptographic protocol (with no client interaction) that allows outsourcing of any multi-party computation of inputs encrypted under multiple unrelated public keys. Our solution employs two non-colluding, semi-honest but untrusted servers $\mathcal{C}$ and $\mathcal{S}$. All steps in our protocol rely on *no interaction* with the clients whatsoever. These clients only initially store their encrypted inputs on the (main) server $\mathcal{C}$ who in turn can compute *any dynamically chosen* n-input function on n given (encrypted) inputs in an SMC protocol together with the second server $\mathcal{S}$. We show our protocols secure in the semi-honest model, meaning that all protocol participants follow the protocol description, but may try to gather information about other parties' inputs, intermediate results, or overall outputs just by looking at the transcripts. For performance reasons, this is the predominant security model used in practical implementations of SMC [12,24,37,38,50,82,99], which particularly makes sense in our setting where the servers (performing the computations) are business driven parties in practice, and cheating would cause negative publicity and harm their reputation.

Assuming the existence of two non-colluding servers in order to perform the secure computations is very common both in the theoretical (e.g., [27,32]) and the practical community (e.g., [24, 38]). In fact, according to [111], a completely non-interactive solution in the *single* server setting can be proven to be impossible to realize (drawing on the impossibility of program obfuscation). Hence, if complete non-interaction of the clients is required (as in our case), we need at least two servers and so our solution is the best we can hope for in this regard. Moreover, several real-world applications relied on the assumption of multiple non-colluding servers in the past [12, 24]. For instance when considering cloud computing scenarios, the multiple servers could be different cloud providers [86].

We make extensive use of the BCP cryptosystem by Bresson, Catalano and Pointcheval [20] which is both additively homomorphic (i.e., it allows addition of plaintexts in the encrypted domain) and offers two independent decryption mechanisms (cf. Section 5.3). The successful usage of the second decryption mechanism depends on a master secret key that is stored on the second server $\mathcal{S}$ in our proposal. With this in mind, the basic idea of our construction consists of three steps:

1. After collecting the individually encrypted inputs, the main server $\mathcal{C}$ runs an SMC protocol with $\mathcal{S}$ in order to transform the given inputs (encrypted under the clients' public keys) into encryptions under the *product* of all involved public keys (ensuring that *all* clients need to participate in a successful decryption) without changing the underlying plaintexts.

2. With these transformed ciphertexts (under the *same* public key), we can run traditional addition and multiplication SMC protocols by using the additively homomorphic property of the underlying cryptosystem, allowing to compute any function represented by an arithmetic circuit.

3. Once the result (encrypted under the product of all keys) is ready, $\mathcal{C}$ runs a final SMC protocol with $\mathcal{S}$ in order to transform this result back into encryptions under the clients' respective public keys.

Finally, we show the applicability of our framework to two application scenarios taken from different domains. Recently, SMC protocols have been used to construct privacy-preserving protocols for face recognition [37,102]. We show how our general-purpose protocols can be leveraged here in order to get rid of interaction with the users of such systems. To highlight the real-world applicability of our construction, we elaborate on several scenarios in the area of social networks where face recognition is used for image tagging services [5] or in order to help law enforcement agencies to prosecute suspected persons [3]. Since social network users usually access their profiles via resource-constrained mobile devices that are not always online, a completely non-interactive solution is crucial for such systems to work in practice. We adapt the privacy-preserving face recognition protocol presented in [37] to our construction and show its practicability by giving an implementation together with a detailed performance analysis. Similarly, privately collecting sensor readings from smart meters has been an important application of SMC protocols recently [50, 77, 82]. We show that certain variants of our original proposal can be used to efficiently enhance privacy in smart grids. For instance, we design a protocol to privately aggregate sensor readings with very low computational costs at the smart meters due to our non-interactive construction. Additionally, we can realize privacy-preserving billing under complex policies.

Related Work

Previous papers on SMC protocols were concerned with *interactive* solutions where all parties are actively involved in computing an arbitrary function on their respective inputs in a privacy-preserving manner [25, 63, 83, 99, 114]. Since we strive for a non-interactive solution, these constructions are not applicable in our scenario. To reduce computational costs at the clients' side, SMC has been considered in the client/server model (as we do) [12, 32, 39, 68, 73, 91], but again with interaction of the clients during the computations. Furthermore, Choi et al. [27] give a solution with minimal interaction of the clients, while relying on two non-colluding servers (as in our construction). Their solution, however, is mostly of theoretical interest both for efficiency reasons and because clients are bound to encrypt their private inputs under a *single* public key that is shared between the two servers (so clients do not have individual private keys). Therefore, in order to efficiently deal with modern star-like communication patterns where clients store their data on a central server (encrypted under their own associated public keys), Halevi et al. [64] proposed a solution in which, although being non-interactive, the server is entitled to learn the result of the computation which contradicts our setting where only clients are allowed to learn the output.

In [51], Gentry proposes to leverage Fully Homomorphic Encryption (FHE) to solve the problem statement in our setting. Unfortunately, besides the lack of efficiency of recent FHE schemes [16, 18, 19, 56], the main drawback is that all clients need to run an interactive setup and an interactive decryption phase after the server computed the result.

Very recently, López-Alt et al. [85] introduced the notion of *On-the-Fly* SMC as the first solution to a similar scenario as ours, yet still relying on an interactive decryption phase. They implement this by using a novel primitive called *Multikey* FHE that allows computation on data encrypted under multiple unrelated public keys, having similar efficiency shortcomings as all the other FHE schemes.

In a nutshell, all existing solutions are not applicable in our setting where SMC is to be performed on data encrypted under multiple unrelated keys, except for one [85] which lacks in efficiency and relies on an interactive user decryption.

7.1. Our Construction

From now on, any occurance of encryption is with respect to the BCP scheme as defined in Definition 19. If the context is clear, we omit the public parameters PP in the algorithms of BCP, e.g., we write $\mathsf{Enc}_{\mathsf{pk}}(m)$ instead of $\mathsf{Enc}_{(\mathsf{PP},\mathsf{pk})}(m)$ for a given message m.

Recall that we are considering a scenario with n (mutually distrusting) clients, denoted by $P_1, \ldots, P_n$, each having its own pair of public and private keys $(\mathsf{pk}_i, \mathsf{sk}_i)$, $i = 1, \ldots, n$.[1] Each client P_i ($i = 1, \ldots, n$) stores private data m_i encrypted under its respective public key pk_i on an untrusted server $\mathcal{C}$.

Now, $\mathcal{C}$ is assigned to compute an arbitrary n-input function f on the clients' inputs, while keeping the inputs and intermediate results private. We represent such functions f by means of arithmetic circuits, i.e., the computation of a given function amounts to the evaluation of addition and multiplication gates over encrypted inputs. More details on this and other means of representing a function f can be found in [75].

Our basic idea to realize this functionality can be summarized as follows (illustrated in Figure 7.1):

1. We assume the existence of a second untrusted server $\mathcal{S}$ that acts semi-honestly and that does not collude with any of the other parties.

2. Initially, this second server $\mathcal{S}$ runs a setup Init that sets up the system and distributes the system's public parameters.

3. After this initial setup, clients can use the cryptosystem's KeyGen (independently of any further party) to generate their respective pair of public and private keys, and to upload encryptions of their private data to the first server $\mathcal{C}$.

[1] Fixing the number n initially is for reasons of readability only. In fact, in our construction, this number is allowed to change over time. More importantly, clients are able to generate their own pair of public and private keys without communicating to some trusted third party. Therefore, participation in the system is a dynamic process.

4. Once an (arbitrary) function f is to be evaluated on the, say, n inputs $m_1, \ldots, m_n$ of clients $P_1, \ldots, P_n$, the server $\mathcal{C}$ runs a cryptographic protocol with the second server $\mathcal{S}$ that consists of only four building blocks: $\mathsf{KeyProd}, \mathsf{Add}, \mathsf{Mult}$ and $\mathsf{TransDec}$. $\mathsf{KeyProd}$ tranforms all ciphertexts to encryptions under a single public key (whose corresponding secret key is unknown), Add and Mult evaluate addition and multiplication gates on encrypted inputs, respectively, and $\mathsf{TransDec}$ transforms the encrypted result $f(m_1, \ldots, m_n)$ back to n encryptions each under a different client's public key.

   The overall protocol is run with no interaction of the clients whatsoever. Recall that the inputs $m_1, \ldots, m_n$ are encrypted under *different* public keys.

5. After all computations are done, each client retrieves the encrypted output of the server $\mathcal{C}$ which it decrypts locally with its respective private key in order to get the result $f(m_1, \ldots, m_n)$.

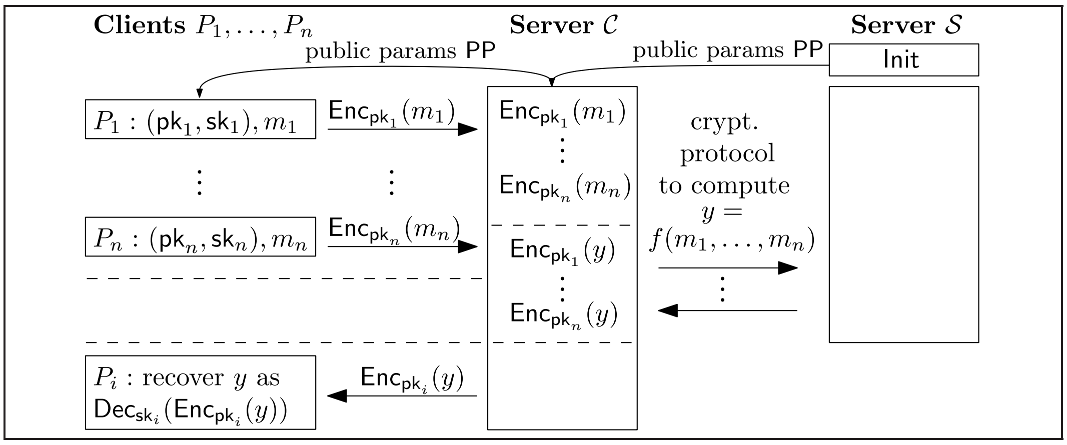

Clients $P_1, \ldots, P_n$
Server $\mathcal{C}$
Server $\mathcal{S}$
public params PP
public params PP
Init
$P_1 : (\mathsf{pk}_1, \mathsf{sk}_1), m_1$
$\mathsf{Enc}_{\mathsf{pk}_1}(m_1)$
$\mathsf{Enc}_{\mathsf{pk}_1}(m_1)$
crypt. protocol to compute $y = f(m_1, \ldots, m_n)$
$\mathsf{Enc}_{\mathsf{pk}_n}(m_n)$
$P_n : (\mathsf{pk}_n, \mathsf{sk}_n), m_n$
$\mathsf{Enc}_{\mathsf{pk}_n}(m_n)$
$\mathsf{Enc}_{\mathsf{pk}_1}(y)$
$\mathsf{Enc}_{\mathsf{pk}_n}(y)$
P_i : recover y as $\mathsf{Dec}_{\mathsf{sk}_i}(\mathsf{Enc}_{\mathsf{pk}_i}(y))$
$\mathsf{Enc}_{\mathsf{pk}_i}(y)$

Figure 7.1.: The basic concept of our construction.

In the following, we explain the individual steps of our protocols:

Initialization. Initially, a setup process initializes the BCP cryptosystem and distributes the system's public parameters. This setup is run by the second server $\mathcal{S}$ (since $\mathcal{S}$ is semi-honest and needs the master secret). We denote this algorithm by Init, which simply runs the algorithm Setup of the BCP cryptosystem and sends its public parameters $\mathsf{PP} = (N, k, g)$ to the server $\mathcal{C}$ (see Figure 7.2).

Data Upload. In order to upload private data to the server $\mathcal{C}$, a client P first needs to receive the system's public parameters $\mathsf{PP} = (N, k, g)$ to be able to generate its own pair of public and private keys. After these keys are generated, the client P

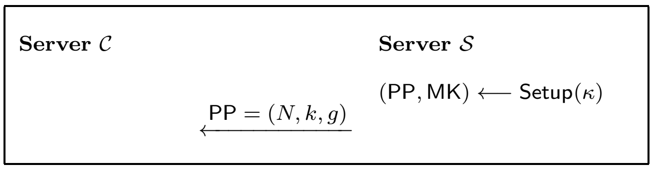

Server $\mathcal{C}$
Server $\mathcal{S}$
$(\mathsf{PP}, \mathsf{MK}) \longleftarrow \mathsf{Setup}(\kappa)$
$\xleftarrow{\mathsf{PP} = (N, k, g)}$

Figure 7.2.: $\mathsf{Init}(\kappa)$. The initial setup algorithm, where κ is the security parameter for the BCP scheme.

can encrypt its private data using Enc and upload it together with its public key to the server $\mathcal{C}$. The details of this procedure can be found in Figure 7.3.

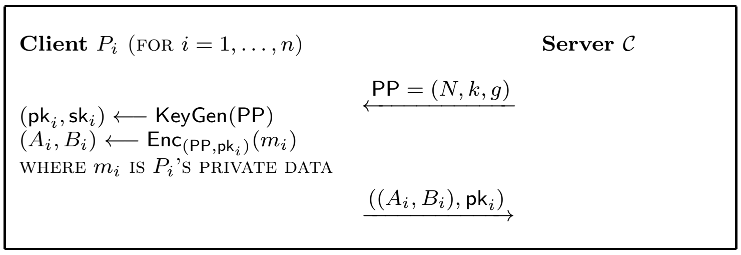

Client P_i (for $i = 1, \dots, n$)
Server $\mathcal{C}$
$\xleftarrow{\mathsf{PP} = (N, k, g)}$
$(\mathsf{pk}_i, \mathsf{sk}_i) \longleftarrow \mathsf{KeyGen}(\mathsf{PP})$
$(A_i, B_i) \longleftarrow \mathsf{Enc}_{(\mathsf{PP}, \mathsf{pk}_i)}(m_i)$
where m_i is P_i's private data
$\xrightarrow{((A_i, B_i), \mathsf{pk}_i)}$

Figure 7.3.: Steps that have to be taken by a client P_i (for $i = 1, \dots, n$) which wants to participate in computations made on the server $\mathcal{C}$.

Cryptographic Protocol between Servers $\mathcal{C}$ and $\mathcal{S}$. Assume that the server $\mathcal{C}$ wants to compute an encryption of $f(m_1, \dots, m_n)$ for an n-input function f where $m_1, \dots, m_n$ are the private inputs of the clients $P_1, \dots, P_n$. Recall that during the data upload phase, $\mathcal{C}$ retrieved only encryptions of the inputs $m_1, \dots, m_n$. $\mathcal{C}$ does its computations by means of a cryptographic protocol between $\mathcal{C}$ and $\mathcal{S}$ consisting of the 4 subprotocols: $\mathsf{KeyProd}$, Add, Mult and $\mathsf{TransDec}$.

Recall that we represent the function f by an arithmetic circuit, meaning that we have to be able to securely evaluate addition and multiplication gates. Addition gates seem to be easy to deal with since the underlying cryptosystem is additively homomorphic. Unfortunately though, the clients' inputs are encrypted under *different* public keys and the additive property of the BCP cryptosystem only works for encryptions under the *same* public key. Therefore, the server $\mathcal{C}$ first runs the algorithm $\mathsf{KeyProd}$ which transforms all involved ciphertexts to encryptions under a *single* key. This single key is the product of all involved public keys and so $\mathcal{C}$ remains unable to decrypt the ciphertexts as it does not know the corresponding secret key. In fact, the secret key needed to decrypt encryptions under the product of all clients' public keys is the sum of all clients' secret keys. Of course, decryption still works by using the master secret which is only known to the second server $\mathcal{S}$ and *not* to $\mathcal{C}$. We stress that $\mathcal{S}$ never gets to see encryptions of the clients' original inputs but only blinded versions of them, so it does not learn these inputs although having the master secret.

After this key-transformation of ciphertexts, the additive property of the underlying cryptosystem can be exploited to securely evaluate addition gates. This step is denoted by Add in our construction. Multiplication gates can be securely evaluated by (an adapted version of) the well-known protocol of [28], essentially relying on "blinding-the-plaintext" techniques. This is done by our protocol Mult.

Finally, once the complete arithmetic circuit representing the function f is successfully evaluated by using Add and Mult, $\mathcal{C}$ runs the protocol $\mathsf{TransDec}$ in order to transform the results (encrypted under the product of all public keys) back to encryptions of the individual clients' public keys without changing the underlying plaintext.

In the following, we describe the individual building blocks for this protocol between $\mathcal{C}$ and $\mathcal{S}$.

The Subprotocol $\mathsf{KeyProd}$**.** The purpose of this protocol is to transform the encryptions of all participating clients $P_1, \ldots, P_n$ into encryptions under a single public key, namely the product $\mathsf{Prod.pk} := \prod_{i=1}^{n} \mathsf{pk}_i \bmod N^2$ of all involved public keys without changing the underlying plaintexts. Using the product key here ensures that it is not a key of $\mathcal{C}$'s choosing. In fact, the corresponding secret key required to successfully decrypt an encryption under $\mathsf{Prod.pk}$ is the sum $\sum_{i=1}^{n} \mathsf{sk}_i$ of all clients' secret keys. This subprotocol needs to be run only *once* per fixed set of encrypted inputs and does not depend on the actual function $\mathcal{C}$ wants to evaluate. This means that after the execution of $\mathsf{KeyProd}$, any function can be computed on the transformed ciphertexts.

For a given ciphertext (A_i, B_i) encrypted under the public key pk_i of the client P_i $(i = 1, \ldots, n)$, $\mathcal{C}$ blinds the ciphertext with a random message τ_i and sends it to $\mathcal{S}$. Since $\mathcal{S}$ knows the master secret MK, it uses it to decrypt this blinded ciphertext and re-encrypt it under the product of all clients' public keys. The result of this is then sent back to $\mathcal{C}$ who can remove the blinding τ_i again, achieving an encryption under the product key without changing the underlying plaintext. A detailed description of these steps can be seen in Figure 7.4.

The Subprotocols Add **and** Mult**.** Recall that the server $\mathcal{C}$ wants to compute the function f, which we consider to be represented as an arithmetic circuit over the ring $\mathbb{Z}_N$ (note that hitting on a value in $\mathbb{Z}_N$ which is not invertible modulo N happens with negligible probability only). Therefore, we have to deal with addition- and multiplication-gates in $\mathbb{Z}_N$. Without loss of generality, we consider these as 2-input-1-output gates. The algorithm Add deals with an addition-gate, while the subprotocol Mult deals with a multiplication-gate. We start with the former and stress that it is a non-interactive protocol which does not need the server $\mathcal{S}$. This is

```
Server $\mathcal{C}$, STORES $((A_i, B_i), \mathsf{pk}_i), i = 1, \ldots, n$                    Server $\mathcal{S}$, STORES $\mathsf{MK} = (p', q')$

ON INPUT CIPHERTEXTS $(A_1, B_1), \ldots, (A_n, B_n)$:
$\mathsf{Prod.pk} \longleftarrow \prod_{i=1}^{n} \mathsf{pk}_i \mod N^2$
FOR ALL $i = 1, \ldots, n$:
    PICK RANDOM $\tau_i \overset{U}{\longleftarrow} \mathbb{Z}_N$
    $(C_i, D_i) \longleftarrow \mathsf{Add}((A_i, B_i), \mathsf{Enc}_{\mathsf{pk}_i}(\tau_i))$
                                   $\xrightarrow[(\mathsf{pk}_i, (C_i, D_i))]{\forall i = 1, \ldots, n :}$
                                                            $\mathsf{Prod.pk} \longleftarrow \prod_{i=1}^{n} \mathsf{pk}_i \mod N^2$
                                                            FOR ALL $i = 1, \ldots, n$:
                                                                $z_i \longleftarrow \mathsf{mDec}_{(\mathsf{pk}_i, \mathsf{MK})}(C_i, D_i)$
                                                                $(W_i, Z_i) \longleftarrow \mathsf{Enc}_{\mathsf{Prod.pk}}(z_i)$
                                   $\xleftarrow[(W_i, Z_i)]{\forall i = 1, \ldots, n :}$
FOR ALL $i = 1, \ldots, n$:
    $(\overline{A_i}, \overline{B_i}) \longleftarrow \mathsf{Add}((W_i, Z_i), \mathsf{Enc}_{\mathsf{Prod.pk}}(-\tau))$
OUTPUT THE CIPHERTEXTS $(\overline{A_1}, \overline{B_1}), \ldots, (\overline{A_n}, \overline{B_n})$
```

Figure 7.4.: KeyProd. Transforms encryptions under $\mathsf{pk}_1, \ldots, \mathsf{pk}_n$ into encryptions under $\mathsf{Prod.pk} = \prod_{i=1}^{n} \mathsf{pk}_i \mod N^2$ without changing the underlying plaintexts.

due to the fact that the underlying BCP cryptosystem is additively homomorphic for encryptions under the *same* public key. We recall that this is exactly what the subprotocol KeyProd achieved by computing encryptions under the product Prod.pk, so all clients' private inputs $m_1, \ldots, m_n$ are now encrypted under the same public key. The algorithm to perform the addition is depicted in Figure 7.5.

```
Server $\mathcal{C}$ STORES ENCRYPTIONS
    OF THE FORM $(A, B)$ ENCRYPTED UNDER $\mathsf{Prod.pk}$

ON INPUT TWO CIPHERTEXTS $(A, B)$ AND $(A', B')$:
    $(\overline{A}, \overline{B}) \longleftarrow (A \cdot A' \mod N^2, B \cdot B' \mod N^2)$
OUTPUT THE CIPHERTEXT $(\overline{A}, \overline{B})$
```

Figure 7.5.: Add. Given two ciphertexts (A, B) and (A', B') encrypted under Prod.pk, it computes an encryption of the sum of the underlying plaintexts.

A multiplication-gate, however, has to be computed interactively with the server $\mathcal{S}$. In fact, the protocol we use is an adaptation of the well-known multiplication protocol of [28] which sends blinded version of the original ciphertexts (that are to be multiplied) to $\mathcal{S}$ that in turn uses the master secret to decrypt. Then, $\mathcal{S}$ performs the multiplication in the clear and re-encrypts. The (encrypted) result is sent back to $\mathcal{C}$, which can remove the blinding again. Details are given in Figure 7.6.

The Subprotocol TransDec. Finally, the task of subprotocol TransDec is to take the encrypted result of $f(m_1, \ldots, m_n)$, encrypted under Prod.pk, and to transform it back to n encryptions of the same plaintext $f(m_1, \ldots, m_n)$, each under a different client's public key $\mathsf{pk}_1, \ldots, \mathsf{pk}_n$, respectively.

Server $\mathcal{C}$, STORES $\mathsf{Prod.pk}$ AND $\mathsf{pk}_1, \ldots, \mathsf{pk}_n$		**Server** $\mathcal{S}$, STORES $\mathsf{MK}, \mathsf{Prod.pk}$, AND $\mathsf{pk}_1, \ldots, \mathsf{pk}_n$
ON INPUT CIPHERTEXTS (A, B) AND (A', B') UNDER $\mathsf{Prod.pk}$:		
PICK RANDOM $\sigma, \sigma' \xleftarrow{U} \mathbb{Z}_N$		
$(C, D) \longleftarrow \mathsf{Add}((A, B), \mathsf{Enc}_{\mathsf{Prod.pk}}(-\sigma))$		
$(C', D') \longleftarrow \mathsf{Add}((A', B'), \mathsf{Enc}_{\mathsf{Prod.pk}}(-\sigma'))$	$\xrightarrow{(C, D), (C', D')}$	
		$z \longleftarrow \mathsf{mDec}_{(\mathsf{Prod.pk}, \mathsf{MK})}(C, D)$
		$z' \longleftarrow \mathsf{mDec}_{(\mathsf{Prod.pk}, \mathsf{MK})}(C', D')$
	$\xleftarrow{(Z_1, Z_2)}$	$(Z_1, Z_2) \longleftarrow \mathsf{Enc}_{\mathsf{Prod.pk}}(z \cdot z')$
$(T_1, T_2) \longleftarrow \mathsf{Enc}_{\mathsf{Prod.pk}}(-\sigma\sigma')$		
$(\overline{A}, \overline{B}) \longleftarrow (Z_1 A^{\sigma'} A'^{\sigma} T_1 \mod N^2, Z_2 B^{\sigma'} B'^{\sigma} T_2 \mod N^2)$		
OUTPUT CIPHERTEXT $(\overline{A}, \overline{B})$		

Figure 7.6.: Mult. Given two ciphertexts (A, B) and (A', B') encrypted under $\mathsf{Prod.pk}$, it computes an encryption of the multiplication of the underlying plaintexts.

Again, the idea is to blind the original ciphertext and send it to $\mathcal{S}$, which in turn decrypts using the master secret and then creates n encryptions for each client's public key. The created ciphertexts are returned to $\mathcal{C}$ which removes the blindings. The precise steps of this protocol are summarized in Figure 7.7.

Server $\mathcal{C}$, STORES $\mathsf{Prod.pk}$ AND $\mathsf{pk}_1, \ldots, \mathsf{pk}_n$		**Server** $\mathcal{S}$, STORES $\mathsf{MK}, \mathsf{Prod.pk}$ AND $\mathsf{pk}_1, \ldots, \mathsf{pk}_n$
ON INPUT A CIPHERTEXT (A, B) UNDER $\mathsf{Prod.pk}$:		
PICK RANDOM $\tau \xleftarrow{U} \mathbb{Z}_N$		
$(C, D) \longleftarrow \mathsf{Add}((A, B), \mathsf{Enc}_{\mathsf{Prod.pk}}(\tau))$	$\xrightarrow{(C, D)}$	
		$z \longleftarrow \mathsf{mDec}_{(\mathsf{Prod.pk}, \mathsf{MK})}(C, D)$
		FOR ALL $i = 1, \ldots, n$:
	$\xleftarrow{\forall i = 1, \ldots, n : (W_i, Z_i)}$	$(W_i, Z_i) \longleftarrow \mathsf{Enc}_{\mathsf{pk}_i}(z)$
FOR ALL $i = 1, \ldots, n$:		
$(\overline{A_i}, \overline{B_i}) \longleftarrow \mathsf{Add}((W_i, Z_i), \mathsf{Enc}_{\mathsf{pk}_i}(-\tau))$		
OUTPUT THE CIPHERTEXTS $(\overline{A_1}, \overline{B_1}), \ldots, (\overline{A_n}, \overline{B_n})$		

Figure 7.7.: $\mathsf{TransDec}$. Given a ciphertext (A, B), it computes n ciphertexts $(\overline{A_1}, \overline{B_1}), \ldots, (\overline{A_1}, \overline{B_1})$ of the same plaintext, encrypted under $\mathsf{pk}_1, \ldots, \mathsf{pk}_n$, respectively.

Data Retrieval. Each client P_i, $i = 1, \ldots, n$, can get the result of the computation by first retrieving (from $\mathcal{C}$) the encryption of $f(m_1, \ldots, m_n)$ under its public key pk_i that has been computed during the subprotocol $\mathsf{TransDec}$, and then decrypting this ciphertext by using its corresponding private key sk_i.

7.1.1. Correctness

We assume that all participants follow our protocol decriptions. Furthermore, we assume that the initial setup Init has been performed as decribed in Figure 7.2, and that all clients (wishing to participate) sent their encrypted private data to the server $\mathcal{C}$ as depicted in Figure 7.3. Under these assumptions, we show that the remaining protocols $\mathsf{KeyProd}, \mathsf{Add}, \mathsf{Mult}$ and $\mathsf{TransDec}$ produce the desired outputs correctly.

$\mathsf{KeyProd}$: Observe that for $i = 1, \ldots, n$, the values (C_i, D_i) are just blinded versions (using τ_i) of the original input ciphertexts (A_i, B_i), which are then decrypted (using the master secret MK) and re-encrypted under the product key $\mathsf{Prod.pk}$. The resulting values (W_i, Z_i) can then be transformed back to encryptions of the original underlying plaintexts by using the additively homomorphic property of the BCP scheme, and substracting the blinding value τ_i again.

Add: The correctness of this algorithm follows immediately from the additively homomorphic property and we refer to [20] for details.

Mult: We show that the output ciphertext $(\overline{A}, \overline{B})$ of algorithm Mult is indeed an encryption (under $\mathsf{Prod.pk}$) of the multiplication of the underlying plaintexts m, m' of the two input ciphertexts $(A, B), (A', B')$ (encrypted under $\mathsf{Prod.pk}$), respectively. Observe that the decryption of $(\overline{A}, \overline{B})$ under the secret key corresponding to $\mathsf{Prod.pk}$ (as mention before, if $\mathsf{Prod.pk} = \prod_{i=1}^{n} \mathsf{pk}_i$ the corresponding secret key is $\sum_{i=1}^{n} \mathsf{sk}_i$) yields $z \cdot z' + \sigma' m + \sigma m' + (-\sigma\sigma')$ where $z = m - \sigma$, $z' = m' - \sigma'$, and σ, σ' are random blinding values (cf. Figure 7.6). The expansion of this term gives $mm' - \sigma' m - \sigma m' + \sigma\sigma' + \sigma' m + \sigma m' + (-\sigma\sigma') = mm'$, which is the desired result.

$\mathsf{TransDec}$: Recall that this protocol takes an encryption (A, B) under $\mathsf{Prod.pk}$ of a message m as input and outputs ciphertexts $(\overline{A_i}, \overline{B_i})$, which are encryptions of the same message m but under the different public keys pk_i, for all $i = 1, \ldots, n$. In this protocol, essentially, $\mathcal{C}$ blinds the ciphertext (A, B) by adding a random message τ to the underlying plaintext m, which $\mathcal{S}$ decrypts and encrypts again under pk_i, for all $i = 1, \ldots, n$. It is obvious that once $\mathcal{C}$ subtracted τ again, the resulting ciphertext will be an encryption of m under pk_i, for all $i = 1, \ldots, n$.

Since $\mathsf{KeyProd}$ is independent of the actual function f that is to be computed, we see that once the underlying message $f(m_1, \ldots, m_n)$ has been computed in the encrypted domain (using Add and Mult), the correctness of $\mathsf{TransDec}$ yields that each client P_i can retrieve its dedicated encryption of $f(m_1, \ldots, m_n)$, which it can successfully decrypt by using its corresponding private key sk_i, $i = 1, \ldots, n$.

7.2. Security Analysis

The following security analysis considers the semi-honest model only, meaning that all parties follow the protocol description but try to gather information about other parties' inputs, intermediate results, or overall outputs just by looking at the protocol's transcripts. As usual, security in this model is proven in the "real-vs.-ideal" framework [62, Ch. 7]: there is an ideal model where all computations are performed via an additional trusted party and it is then shown that all adversarial behavior in the real model (where there is no trusted party) can be simulated in the ideal model. We deal with each subprotocol individually which is possible due to the Composition Theorem for the semi-honest model [62, Theorem 7.3.3]. Note that the security of all our protocols is essentially based on the well-known concept of "blinding" the plaintext: Given an encryption of a message, we use the additively homomorphic property of the cryptosystem to add a random message to it, which blinds the original plaintext.

Recall that before the actual computations (i.e., the cryptographic protocol) are performed between servers $\mathcal{C}$ and $\mathcal{S}$, there is only the initial setup of the BCP cryptosystem and the step where clients $P_1, \ldots, P_n$ store their encrypted private inputs $m_1, \ldots, m_n$ on the server $\mathcal{C}$ – for these two steps, the security follows from the semantic security of the BCP cryptosystem. Since we assume no collusion at all between any of the participating parties, it remains to show that neither $\mathcal{C}$ nor $\mathcal{S}$ learn anything from the cryptographic protocol (consisting of $\mathsf{KeyProd}, \mathsf{Add}, \mathsf{Mult}$ and $\mathsf{TransDec}$) computing the n encryptions of $f(m_1, \ldots, m_n)$ under the clients' public keys $\mathsf{pk}_1, \ldots, \mathsf{pk}_n$, respectively.

$\mathsf{KeyProd}$: The only data sent is fresh ciphertexts. Due to the blinding values τ_i, $i = 1, \ldots, n$, and the semantic security of the BCP cryptosystem, these encryptions are indistinguishable from random ciphertexts and are therefore easily simulatable by encryption of, say, 1. Hence, both servers $\mathcal{C}$ and $\mathcal{S}$ do not learn anything at all from these ciphertexts.

Add: This algorithm is non-interactive and does not involve the server $\mathcal{S}$ at all, so we are only concerned about $\mathcal{C}$. For $\mathcal{C}$, however, no information leakage is assured by the semantic security of the BCP scheme since Add just uses the additively homomorphic property of the cryptosystem.

Mult: Again, the only data sent is fresh ciphertexts of blinded messages and due to the semantic security of the underlying cryptosystem, we can simulate these by random encryptions.

$\mathsf{TransDec}$: Basically, the security argument here is the same as for the protocol $\mathsf{KeyProd}$. The first step of $\mathsf{TransDec}$ is for $\mathcal{C}$ to blind the underlying message of the ciphertext (A, B) (which is encrypted under $\mathsf{Prod.pk}$) with the random message τ. This ensures that $\mathcal{S}$ does not learn any information about the original plaintext when receiving the "blinded" ciphertext (C, D). On the other hand, since $\mathcal{C}$ receives *fresh* encryptions under the public keys of the clients, he gets no information about the underlying plaintext whatsoever. In the language of simulations, a formal proof would amount to simulating the views which again is possible by using random encryptions due to the semantic security of the BCP scheme.

7.3. Variants

Recall that the main goal of our construction was to get rid of any interaction with the users. At the same time, our solution is very efficient compared to other existing work in similar scenarios. There are certain application scenarios, however, where the interaction with users is explicitly wanted, e.g., when the server is allowed to learn the result upon the approval of *all* participating users. In this section, we give a few variants of our original proposal that allow leveraging our efficient solution to such applications as well:

1. *Intermediate Key Aggregation.* If the (encrypted) private data from a client Q is not sent to the server $\mathcal{C}$ directly but goes through a chain of intermediate clients, this variant allows for the secure aggregation of intermediate public keys to Q's encrypted data and hence reduces work that otherwise needs to be done by the protocol $\mathsf{KeyProd}$.
2. *Disclosure by Clients' Approval.* This variant achieves a solution to the following scenario: Assume that clients are not supposed to see the result of $\mathcal{C}$'s computation. However, if *all* participating clients give their approval, $\mathcal{C}$ is able to read the result (while clients stay oblivious to this result).
3. *Interactive Decryption by all Clients.* If clients should learn the result only if *all* participating clients get together (and not each client independently as in our original construction), this variant can be run instead of protocol $\mathsf{TransDec}$ in order to make the decryption interactive between all clients. Decryption is successful if and only if all clients participate in this interaction (again in the semi-honest model).

Intermediate Key Aggregation. Assume that the (encrypted) private data of a client Q participating in our protocol is not sent to the server $\mathcal{C}$ directly (as depicted in Figure 7.3), but goes through other clients $Q_1, \ldots, Q_\ell$ (a subset of all clients $P_1, \ldots, P_n$). The protocol presented here optimizes $\mathsf{KeyProd}$ in this scenario: Recall, that $\mathsf{KeyProd}$ takes all the participating clients' ciphertexts (A_i, B_i) as input and transforms these to encryptions under the product $\mathsf{Prod.pk}$ of all participating public keys. The optimization we aim for does the aggregation of the public keys of the clients $Q, Q_1, \ldots, Q_\ell$ *before* the ciphertexts end up at the server $\mathcal{C}$. More precisely, let (A, B) be a ciphertext of a message m encrypted under a public key pk which arrives at client Q_i, $i = 1, \ldots, \ell$. This client can then use its own public key pk_i in order generate an encryption of the same message m but encrypted under the product $\mathsf{pk} \cdot \mathsf{pk}_i \bmod N^2$. This "key aggregation" is one step that otherwise had to be done by the second server $\mathcal{S}$ in the original algorithm $\mathsf{KeyProd}$. Since Q_i knows its own private key, it can use the first component A and raise it to the power of its private key. The result of this can then be multiplied with the second component B which transforms the ciphertext (A, B) to an encryption under $\mathsf{pk} \cdot \mathsf{pk}_i \bmod N^2$. Details of this are described in Figure 7.8.

```
Client $Q_i$ HAS PRIVATE KEY $\mathsf{sk}_i$

UPON RETRIEVAL OF CIPHERTEXT $(A, B)$ (ENCRYPTED UNDER $\mathsf{pk}$):
  $\overline{B} \longleftarrow A^{\mathsf{sk}_i} \cdot B \mod N^2$
OUTPUT CIPHERTEXT $(A, \overline{B})$
```

Figure 7.8.: Upon retrieval of a ciphertext (A, B) of m under pk, client Q_i produces an encryption of m under $\mathsf{pk} \cdot \mathsf{pk}_i \bmod N^2$, $i = 1, \ldots, \ell$.

The correctness of this algorithm is immediately seen, since when $(A, B) = (g^r \bmod N^2, \mathsf{pk}^r(1 + mN) \bmod N^2)$, we have that $\overline{B} = A^{\mathsf{sk}_i + \mathsf{sk}}(1 + mN) \bmod N^2$ since $\mathsf{pk}^r = g^{r\mathsf{sk}} = A^{\mathsf{sk}} \bmod N^2$.

The security of this variant is implied by the security of the BCP cryptosystem. This is because the first client Q in the chain of clients is simply giving away a fresh encryption under its own public key.

Disclosure by Clients' Approval. Assume that the clients are not allowed to see the actual result $f(m_1, \ldots, m_n)$ of the computation done by the server $\mathcal{C}$, but if *all* clients approve it, $\mathcal{C}$ should be able to retrieve the result (while all clients stay oblivious). More precisely, let (A, B) denote the encrypted result of the computation done by the server $\mathcal{C}$ before applying algorithm $\mathsf{TransDec}$ to it (so (A, B) is an encryption under the public key $\mathsf{Prod.pk}$). We assume that the participating clients $P_1, \ldots, P_n$ are not supposed to see the (encrypted) result, but on their approval (from *all* of them), the server $\mathcal{C}$ should be able to decrypt (A, B) in order to see the result of the computation. To achieve this, we can run protocol $\mathsf{ODApproval}$ of Figure 7.9 *instead* of $\mathsf{TransDec}$. The basic idea of this protocol is to run the "key aggregation" in reversed order, meaning that $\mathcal{C}$ blinds the first component A of the encrypted result and sends it to the clients. By using their respective private keys, each client returns a modified version of A. Now, recall that the result is encrypted under the product $\mathsf{Prod.pk}$ of all clients' public keys. Therefore, these modified versions of A can be used by $\mathcal{C}$ to "divide out" the public keys of each individual client seperately, ending up with a encryption under the public key 1 which simply is the plaintext itself.

The correctness of this protocol is shown by proving that if (A, B) is an encryption of m' under the public key $\mathsf{Prod.pk}$, then the output m of $\mathsf{ODApproval}$ equals this message m'. If (A, B) was encrypted by using randomness r, then

$$(\overline{A}, \overline{B}) = (g^{\tau} \mod N^2, \mathsf{Prod.pk}^{\tau}(1 + m'N) \mod N^2), \tag{7.1}$$

Clients $P_1, \ldots, P_n$ WITH SECRET KEYS $\mathsf{sk}_1, \ldots, \mathsf{sk}_n$		**Server** $\mathcal{C}$, STORES $\mathsf{Prod.pk}$ AND $\mathsf{pk}_1, \ldots, \mathsf{pk}_n$
		ON INPUT CIPHERTEXT (A, B) (ENCRYPTED UNDER $\mathsf{Prod.pk}$): PICK RANDOM $\rho \overset{U}{\longleftarrow} \mathbb{Z}_N$; $(\overline{A}, \overline{B}) \longleftarrow (A \cdot g^{\rho} \mod N^2, B \cdot \mathsf{Prod.pk}^{\rho} \mod N^2)$
	$\xleftarrow{\overline{A} \text{ TO } \textit{all} \text{ CLIENTS}}$	
CLIENT P_i $(i = 1, \ldots, n)$: $X_i \longleftarrow \overline{A}^{\mathsf{sk}_i} \mod N^2$	$\xrightarrow{X_i}$	
		UPON RETRIEVAL OF $X_1, \ldots, X_n$: $K \longleftarrow \prod_{i=1}^{n} X_i \mod N^2$; $m \longleftarrow \frac{\overline{B}/K - 1 \mod N^2}{N}$; OUTPUT PLAINTEXT m

Figure 7.9.: ODApproval: Server $\mathcal{C}$ asks all participating clients for their approval to disclose the encrypted result to $\mathcal{C}$ while all clients stay oblivious to this result.

where $\tau = r + \rho$. But $K = \prod_{i=1}^{n} \overline{A}^{\mathsf{sk}_i} \mod N^2 = \mathsf{Prod.pk}^{\tau} \mod N^2$ and so

$$m = \frac{(1 + m'N) - 1 \mod N^2}{N} = m'. \tag{7.2}$$

As for the security, we note that the clients do not learn anything at all since the plaintext is encoded in the second component of the ciphertext (A, B) which is never sent to any of the clients. So the plaintext remains information theoretically secure.

Concerning the server $\mathcal{C}$, we recall that due to the semantic security of the BCP cryptosystem, $\mathcal{C}$ is not able to compute the randomness used to encrypt (A, B) and so the clients' messages X_i are indistinguishable from random elements in $\langle g \rangle$. This also implies that as long as one of the messages X_i is missing, $\mathcal{C}$ is not able to decrypt: Assume that the final message X_n is missing and $\mathcal{C}$ already computed $K' = \prod_{i=1}^{n-1} X_i \mod N^2$. Trying to decrypt $(\overline{A}, \overline{B}) = (g^r \mod N^2, \mathsf{Prod.pk}^r(1 + mN) \mod N^2)$ using K' results in the ciphertext $(g^r \mod N^2, \mathsf{pk}_n^r(1+mN) \mod N^2)$, i.e., an encryption under the public key of P_n which is the client from whom the message X_n is still missing.

Interactive Decryption by all Clients. Assume that we want all participating clients to decrypt the result of $\mathcal{C}$'s computation together (in an interactive protocol) so that neither the server $\mathcal{C}$ nor the server $\mathcal{S}$ learn the result, but all clients do (if and only if all clients participate). Essentially, this can be achieved by using the protocol ODApproval (cf. Figure 7.9). Server $\mathcal{C}$ sends the re-randomized encryption $(\overline{A}, \overline{B})$ to all clients and then, instead of sending the values $X_1, \ldots, X_n$ to the server $\mathcal{C}$, for each $i = 1, \ldots, n$, client P_i broadcasts its respective value X_i to all other clients. Upon retrieval of all these values $X_1, \ldots, X_n$, each client is now able to decrypt (in the same way as $\mathcal{C}$ does in Figure 7.9) the ciphertext $(\overline{A}, \overline{B})$ to reveal the underlying result of $\mathcal{C}$'s computation. The correctness of this protocol follows from the correctness of ODApproval.

The same argument as for the previous variant shows that all messages $X_1, \ldots, X_n$ need to be received in order to decrypt the ciphertext $(\overline{A}, \overline{B})$. So decryption is not possible until all clients broadcasted their respective value X_i. Since the clients never see the (encrypted) private inputs of other clients, they cannot learn more than the decryption of $(\overline{A}, \overline{B})$ which is the result of the computation done by the server $\mathcal{C}$.

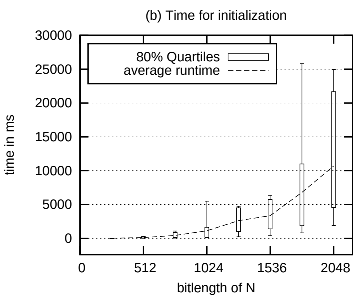
7.4. Performance

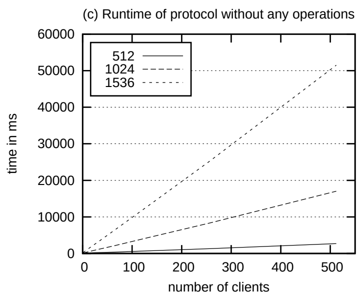
The performance of our contribution depends on the security parameter κ, the number of clients n, the number of additions and multiplications performed, and the network performance (bandwidth and latency). All algorithms, except Init are solely based on arithmetic operations (addition, subtraction, multiplication, exponentiation, inversion, division, comparison) modulo N or N^2 and random number generation. The size of N grows linearly with κ. A full overview of the complexity of each protocol step is given in Figure 7.10 (a). To show that our system can be used in practice, we implemented a proof-of-concept version of all protocols in python. Because the speed of the implementation depends mainly on the speed of the bignum library, we used the GMP library though pythons *gmpy* module. The GMP library is known to be asymptotically faster than pythons native bignum library, when it comes to processing bigger numbers. GMP offers all basic operations we need, except for the generation of safe primes numbers. Therefore, we also used the OpenSSL library through a custom C-binding, but solely for generating safe prime numbers. We used a TCP network connection over the loopback interface for transferring data between the two servers.

The runtime of our code is heavily influenced by the size of N. We recommend a size of N of at least 1024 bit, while 1536 or 2048 bit are better choices for security reasons. Therefore, we performed all benchmarks with these three security parameters. We performed all tests on a *Lenovo Thinkpad T410s* with an *Intel Core i5 M560* running at 2.67 GHz with *Debian Sid*, *Python 2.7.3rc2* and *gmpy 1.15-1*. We ran all clients and both servers on the same host, so that network latency can be ignored. Algorithm Init has the worst time complexity due to the generation of two safe primes. Its runtime is expected to vary, because the number of instructions it needs to execute in order to find two suitable primes depends on the random numbers generated by the algorithm. Figure 7.10 (b) shows the runtime distribution of Init, depending on κ. 20 tests were performed for each choice of κ, and it is clearly visible that the runtime varies a lot. In the next step, we determined how long it takes to encrypt all client's inputs, transcode them at the server and hand the result back to the clients (omitting the initialization step). Figure 7.10 (c) shows that

(a) Complexity of the protocol

Algorithm	Time	Traffic in bits	Round trips
Init	$O(\kappa^5/\log(\kappa)^2)$ on $\mathcal{S}$	4κ	0
Data Upload	$O(\kappa^3)$ on each client $O(n\kappa^2)$ on $\mathcal{C}$	$6n\kappa$	0.5
KeyEval	$O(n\kappa^3)$ on $\mathcal{S}$ $O(n\kappa^2)$ on $\mathcal{C}$	8κ	1
Add	$O(\kappa^2)$ on $\mathcal{C}$	0	0
Mult	$O(\kappa^3)$ on $\mathcal{C}$ $O(\kappa^3)$ on $\mathcal{S}$	12κ	1
TransDec	$O(n\kappa^3)$ on $\mathcal{C}$ $O(n\kappa^3)$ on $\mathcal{S}$	$4(n+1)\kappa$	1
Data Retrieval	$O(\kappa^3)$ on each client	$4n\kappa$	0.5





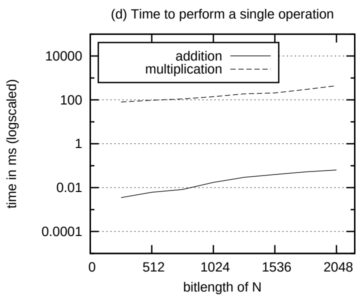


Figure 7.10.: Complexity and runtime analysis of the protocol.

our implementation scales linearly with the number of clients. Without any arithmetic operations, we can perform a full protocol run with 16 clients in 1.7 seconds, using a 1536 bit modulus. Finally, we were interested in the runtime of Add and Mult. Figure 7.10 (d) shows that an addition is much faster than a multiplication. With a 1536 bit modulus, about 25,000 additions, but only 5 multiplications can be performed per second. The results show that our scheme is really useable in practice.

7.5. Applications

Finally, we present two applications of our general-purpose construction, namely privacy-preserving face recognition and private smart metering, which benefit from the non-interactive nature of our protocols. In order to demonstrate the practicability of our approach, we decided to implement the much more complex application of these two, namely privacy-preserving face recognition.

Privacy-Preserving Face Recognition

Face recognition is a widely-used tool in many areas of modern everyday life, among which social networks probably is one of the most important. Many social networks use face recognition tools for things like automatic photo tagging [5] or in order to help law enforcement agencies to prosecute suspected persons [3] (to name just a few). This is usually being done without the explicit consent of users, raising important privacy concerns, as convincingly demonstrated in [37] and [102]. Therefore, Erkin et al. [37] proposed the first privacy-preserving protocol for face recognition which, however, heavily relies on user interaction. Unfortunately, their solution generally does not work in the social network setting where users access their profiles by using resource-constrained mobile devices which are not always online. Therefore, having a completely non-interactive (with the users) solution is crucial in such scenarios. Additionally, users typically want to use their own public keys to encrypt their private images (which, again, is not possible in previous solutions).

Similarly to Erkin et al. [37], the database of "known images" (i.e., when given a face image, we want to find out whether it is in the database or not) is unencrypted in our setting. For instance, in the scenario of helping with the prosecution of suspected persons, the database of known images consists of the suspected persons and is therefore known by the social network provider in plaintext (unencrypted). On the other hand, it is desirable to have all users' profile pictures (or images of users' holidays etc.) encrypted, so that the social network provider is only able to recognize faces that are found in the database while all other faces remain hidden.[2] To achieve this goal, we adapt the privacy-preserving protocol by Erkin et al. [37] to our framework and hence rely on the standard eigenfaces recognition algorithm [108, 109], which can be summarized as follows:

Enrolment. This is the first phase, in which a "face space" is determined on the basis of a set of M training images $\Theta_1, \ldots, \Theta_M$ (represented as a vector of length L). Later on, in the recognition phase, face images will be projected onto this "face space" and matched against the training images. Creating the "face space" is done by

1. computing the "average face" $\Psi = \frac{1}{M} \sum_{i=1}^{M} \Theta_i$,
2. computing the "difference vectors" $\Phi_i = \Theta_i - \Psi$ for all $i = 1, \ldots, M$,
3. applying the Principal Component Analysis (PCA) to the covariance matrix $C = \frac{1}{M} AA^T$ where A is the matrix $[\Theta_1 \ \Theta_2 \ \ldots \ \Theta_M]$ (see [109] for details).[3] This yields orthonormal eigenvectors with associated eigenvalues of C,
4. selecting $K \ll M$ eigenvectors $u_1, \ldots, u_K$ (the "eigenfaces") associated to the K largest eigenvalues,

[2]Note that the correct recognition is only assured with a certain probability, since the tool of face recognition is error-prone. In the implementation that we use (cf. [37]), the correct classification rate is approximately 96%.

[3]Actually, PCA is applied to the much smaller matrix $A^T A$ with appropriate post-processing [109].

5. and projecting $\Theta_1, \ldots, \Theta_M$ to the "face space" spanned by $u_1, \ldots, u_K$ to get their "feature vectors" $\Omega_1, \ldots, \Omega_M$ where $\Omega_i = (\omega_{i1}, \ldots, \omega_{iK})^T$ with $\omega_{ij} = u_j^T(\Theta_i - \Psi)$ for all $j = 1, \ldots, K$ and $i = 1, \ldots, M$.

Recognition. Given a face image Γ, recognition is done by

1. ("Projection" step) projecting Γ onto the face space by computing its "feature vector" $V = (v_1, \ldots, v_K)^T$ with $v_j = u_j^T(\Gamma - \Psi)$ for all $j = 1, \ldots, K$,
2. ("Distance" step) computing the squares $D_1, \ldots, D_M$ of the Euclidean distances between V and $\Omega_1, \ldots, \Omega_M$ as $D_j = \|V - \Omega_j\|^2$ for $j = 1, \ldots, M$,
3. ("Minimum" step) and reporting a match if the smallest distance $D_{\min} = \min\{D_1, \ldots, D_M\}$ is smaller than a given threshold τ.

We stress that since the database of "known images" is known by the social network provider, all steps of the enrolment phase can be done in the clear, without using any form of encryption. Hence, we only have to transform the steps of the recognition phase into the encrypted domain. We do this by a rather straight-forward adaption of the protocols in [37] to our framework:

Encrypting Images. Assume that a user U wants to upload an encryption of a private face image Γ under its respective public key pk to the server $\mathcal{C}$ (here, $\mathcal{C}$ is the social network provider). Recall that we represent images as vectors of length L, i.e., $\Gamma = (\Gamma_1, \ldots, \Gamma_L)^T$. User U encrypts the image Γ component-wise, which we denote by $\mathsf{EncImg}_{\mathsf{pk}}(\Gamma)$, i.e., $\mathsf{EncImg}_{\mathsf{pk}}(\Gamma) = (\mathsf{Enc}_{\mathsf{pk}}(\Gamma_1), \ldots, \mathsf{Enc}_{\mathsf{pk}}(\Gamma_L))^T$.

Projection. Given an image encryption $\mathsf{EncImg}_{\mathsf{pk}}(\Gamma) = (c_1, \ldots, c_L)^T$ of a face image Γ under a user's public key pk, server $\mathcal{C}$ computes for all $i = 1, \ldots, K$:

$$\overline{v_i} := \mathsf{Enc}_{\mathsf{pk}}\left(-\sum_{j=1}^{L} u_{ij}\Psi_j\right) \cdot \prod_{j=1}^{L} c_j^{u_{ij}} \bmod N^2,$$

where u_{ij} denotes the j-th component of the i-th eigenface u_i and Ψ_j denotes the j-th component of the average face Ψ. This way, $\mathcal{C}$ obtains the encrypted feature vector $\overline{V} = (\overline{v_1}, \ldots, \overline{v_K})^T$ corresponding to Γ.

Distance. Given the projected face vector $\overline{V} = (\overline{v_1}, \ldots, \overline{v_K})^T$ from the "projection" step, $\mathcal{C}$ computes the Euclidean distances to the feature vectors $\Omega_i = (\omega_{i1}, \ldots, \omega_{iK})^T$ for $i = 1, \ldots, M$, with the help of server $\mathcal{S}$ as follows: First, $\mathcal{C}$ computes for all $i = 1, \ldots, M$: $\sigma_{i1} := \mathsf{Enc}_{\mathsf{pk}}\left(\sum_{j=1}^{K} \omega_{ij}^2\right)$ and $\sigma_{i2} := \prod_{j=1}^{K} \overline{v_j}^{(-2\omega_{ij})}$. Note that $\mathcal{C}$ can perform these computations without interacting with $\mathcal{S}$. Second, $\mathcal{C}$ computes $\sigma_3 := \prod_{i=1}^{M} \mathsf{Mult}(\overline{v_i}, \overline{v_i})$. Note that Mult runs interactively with server $\mathcal{S}$. Finally, for all $i = 1, \ldots, M$, $\mathcal{C}$ computes the results $\delta_i := \sigma_{i1} \cdot \sigma_{i2} \cdot \sigma_3$, which are encryptions of the squares of the Euclidean distances of V with $\Omega_1, \ldots, \Omega_M$.

Minimum. In this final step, $\mathcal{C}$ performs a joint computation with $\mathcal{S}$ in order to determine the minimum of the encrypted distances $\delta_1, \ldots, \delta_M$ together with an encryption of its index $\mathsf{id} \in \{1, \ldots, M\}$ and to check whether this minimum is smaller than a given threshold τ. This is done by encrypting τ (under pk) and treating it as an additional (encrypted) distance with the special index 0 (which we encrypt under pk as well). The distances as well as the indices have to be stored encrypted since only the user U should be able to see the final result of this minimum-finding. The protocol to perform this step in a privacy-preserving manner is exactly as in [37] and we refer the reader to this reference for details since the protocol is rather complex and would go beyond the scope of this paper. We stress that one simply has to replace the parties Alice and Bob in [37] by servers $\mathcal{S}$ and $\mathcal{C}$, respectively, and the Paillier cryptosystem [93] by the BCP cryptosystem [20]. Also, we note that in our implementation we solely used the BCP cryptosystem, while [37] switches to the homomorphic encryption scheme DGK by Damgård, Geisler, and Krøigaard [31] for efficiency reasons.

We stress that we deliberately implemented all protocols only by using our general-purpose construction as is, *without* any tricks to optimize the performance, such as switching to the much more efficient DGK cryptosystem for computing the minimal distance, or treating the fact that most computations can be done in an offline pre-computation phase. Note, however, that all such tricks can be applied to our protocols in exactly the same way as it is done in [37]. We did not implement these tricks in order to show the performance of our general-purpose construction in its plain as-is state without tweaking it to specific application scenarios.

Performance Analysis. We implemented the complete protocol for privacy-preserving face recognition, as described above, in python while basing it on our implementation of our general-purpose construction described in Section 7.4. This means that we used the GMP and OpenSSL libraries, while all tests were performed on a *Lenovo Thinkpad T410s* with an *Intel Core i5 M560* running at 2.67 GHz, running *Debian Sid* with *Python 2.7.3rc2* and *gmpy 1.15-1.* The security parameter of our general-purpose construction was fixed to $\kappa = 1024$, i.e., the size of the RSA-modulus N is 1024 bits.

In order to avoid confusion, we used the same set of parameters for the privacy-preserving face recognition protocol as [37]. This also ensures the same reliability results as in [37] (achieving a correct classification rate of approximately 96%). In particular, we used the "ORL Database of Faces" from AT&T Laboratories Cambridge [1] containing 10 images of 40 different subjects, yielding a total amount of 400 images.[4] These images are of size 92×112 pixels, which we represented as

[4]We actually use a subset of these which is determined through the size M of the database of "known images".

Database size M	Runtime in sec.	Traffic $\mathcal{C}$ to $\mathcal{S}$ in MB	Traffic $\mathcal{S}$ to $\mathcal{C}$ in MB	Runtime w/ remote $\mathcal{S}$ in sec.
10	106	0.6	0.7	100
50	297	4	4.2	294
100	533	8	8.5	506
150	761	12	13	744
200	1004	16	17	970

Table 7.1.: Complexity of performing privacy-preserving face recognition with our general-purpose construction.

vectors of length $L = 92 \cdot 112 = 10304$. It is demonstrated in [37] that $K > 12$ eigenfaces in the enrolment phase do not significantly increase the correct classification rate, which is why we used their suggestion of $K = 12$ (for $M = 10$, we used $K = 5$ instead). Table 7.1 depicts the results of our performance analysis of the complete privacy-preserving face recognition protocol (containing both $\mathcal{C}$'s and $\mathcal{S}$'s costs). The first column shows the amount M of feature vectors stored in the database, while the second column shows the runtime (wall clock time in seconds) of the full protocol per invocation with one face image (that is to be matched with the database). The runtime shown in the second column contains the enrolment phase which in each case took less than 1% time of the shown overall runtime. We therefore included the enrolment phase here, since it does not significantly change the overall performance. The communication complexity (measured with iptables) is summarized in the other two columns: the third column shows the amount of data (in megabytes) sent from server $\mathcal{C}$ to server $\mathcal{S}$, while the fourth column shows the total traffic from $\mathcal{S}$ back to $\mathcal{C}$.

Finally, the fifth column shows the runtime (in seconds) of the full protocol (per single query) when using a remote server $\mathcal{S}$ over the Internet. For this remote setup, we connected our Lenovo Thinkpad T410s (server $\mathcal{C}$) with a 20 Mbit/s consumers cable connection to the Internet and started our server $\mathcal{S}$ implementation on a remote system equipped with a 3.2 GHz *AMD Phenom II X6 1090T* processor in a datacenter connected to the Internet with a Gigabit Ethernet adapter. The average round trip time to that system was 32 ms, measured with the linux "ping" utility. Observe that the overall runtime when running server $\mathcal{S}$ remotely is less than when running $\mathcal{S}$ locally. This is due to the fact that the remote system that we used has higher performance than the local one.

In summary, for a database of size $M = 200$, a full protocol run requires approximately 16 minutes (both when running $\mathcal{S}$ locally or remotely) for checking whether a given encrypted image is contained in the database or not. This demonstrates that our general-purpose construction can really be used in practice for performing facial recognition in a privacy-preserving manner in the social network setting, even *without* any efficiency optimizations. Again, we stress that we can apply the same performance tweaks to our protocol as Erkin et al. [37], which would yield a significant efficiency boost. Also, we note that we did not use explicit parallelism in our implementation, so that the performance can moreover be improved by using multiple CPU cores. We leave the elaboration of such possibilities as interesting future work.

Private Smart Metering

Another interesting application domain, actively promoted by many governments, concerns the deployment of smart grids for modernizing the distribution networks of electricity, gas or water. For such services, smart meters record the consumption of individual consumers on a fine-grained basis and send all collected data to a central authority (the *supplier*), which in turn uses these inputs to compute overall consumptions, bills (by using dynamic pricing schemes), usage patterns, or to detect fraud or leakage in other utilities. Due to the collection of massive amount of sensitive data, privacy concerns have been raised in the past years, and various solutions protecting the consumers' privacy have been proposed for different computing tasks of the supplier. Current solutions either only consider eavesdropping outsiders (and not the supplier) [82], require interaction and high computational overhead at the smart meters [50, 77] or rely on trusted components alongside each smart meter [71, 90].

Our construction of Section 7.1 offers a non-interactive protocol with very low computational costs at the smart meters. The smart meters would act as the clients in our protocol description of Section 7.1, sending their encrypted private data to the supplier $\mathcal{C}$. With our cryptographic protocol between $\mathcal{C}$ and a second server $\mathcal{S}$, the supplier $\mathcal{C}$ can essentially compute any function on the consumers' inputs in a privacy-preserving way. Furthermore, concerning private information aggregation, we can extend the distributed incremental data aggregation approach of [82] in order to protect the consumers' privacy from the supplier $\mathcal{C}$ as well. In [82], the idea is to encrypt each consumer's private data with an additively homomorphic encryption scheme under the public key of the supplier $\mathcal{C}$ and then send this encrypted information through other households in the neighbourhood (this is due to the use of short-range communication networks) in order to finally reach the supplier. Intermediate households aggregate their private data to the one they receive by using the additive property of the cryptosystem. Once the supplier received the data from all these chains of households, it aggregates these encryptions together to get the aggregation of all consumers' inputs. In [82], only eavesdropping adversaries (such as the intermediate households) are considered. By employing our construction variant “Intermediate Key Aggregation”, we can protect the consumers’ privacy from intermediate households as well as from the supplier: Consumer P_1 encrypts its private data by using its own public key pk_1 and sends it to the next consumer P_2 which in turn uses the “Intermediate Key Aggregation” algorithm to transform the encryption into an encryption under the product $\mathsf{pk}_1\mathsf{pk}_2 \bmod N^2$ of the two consumers’ public keys. Furthermore, P_2 encrypts its own private input under this product $\mathsf{pk}_1\mathsf{pk}_2 \bmod N^2$ as well and uses the additively homomorphic property of the BCP cryptosystem to aggregate its encrypted input to the encrypted input of the first consumer P_1. These steps continue through the whole chain of consumers until the supplier $\mathcal{C}$ is reached. $\mathcal{C}$ can now aggregate all the remaining consumers’ encrypted inputs (similar to [82]) but still is unable to see any of the underlying plaintexts. Depending on the application, we can continue in three ways:

1. $\mathcal{C}$ decrypts the result jointly with the second server $\mathcal{S}$ (recall that $\mathcal{S}$ still has the master secret).
2. We use the variant “Disclosure by Clients’ Approval” which means that $\mathcal{C}$ can only decrypt if all consumers approve it.
3. $\mathcal{C}$ sends the encrypted result of the aggregation (or any other evaluation on the basis of this aggregation) back to the consumers for local decryption.

We stress that due to the high efficiency of our solution, it is particularly interesting in the smart grid scenario as it minimizes computational overhead at the smart meters.

8 Conclusion

The contributions of this dissertation have a strong impact on both the theoretical understanding and the practical implementation of homomorphic encryption schemes. From a theoretical perspective, our universal characterization results (Chapters 4–6) round off the topic of GHE and identifies similarities with the relaxed notion of SWHE, paving the way for the study on FHE schemes. By definition, GHE schemes are less powerful (in terms of operations that can be performed in the encrypted domain) than most SWHE and FHE schemes, but compared to current instantiations, they are still more efficient. To make up for this less powerfulness, our result on the secure outsourcing of multiparty computation (Chapter 7) shows that by relying on interactivity, GHE schemes can nevertheless be used to tackle important problems that previously could only be solved using FHE. From a practical perspective, this gives rise to efficient solutions for real-world problems, such as the secure outsourcing of private data to a cloud provider, bridging our strong foundational work with practice.

In the following, we summarize the major contributions of this dissertation with respect to the impact on their respective research areas and conclude each summary with directions for future work.

Security Foundations of GHE and Links to FHE

The encryption algorithms of all existing IND-CPA secure HE schemes (GHE, SWHE, and FHE) follow the shift-type approach: first, the plaintext is embedded into the ciphertext space and then randomly shifted inside of this space (e.g., “plaintext” + “noise”). For these shift-type schemes, we show in Chapter 4 that we can characterize their IND-CPA security via SMP. For GHE schemes, we also achieved a similar characterization for their IND-CCA1 security (via SOAP).

As it turns out, once an HE scheme is plugged into our framework, our IND-CPA (IND-CCA1, resp.) characterization outputs the corresponding underlying hardness assumption in terms of an instance of SMP (SOAP, resp.). This helps in assessing the security of a given HE scheme and allows to show the Paillier cryptosystem provably secure in terms of IND-CCA1 under a reasonable assumption.

We see the development of a similar IND-CCA1 characterization for more general SWHE schemes as an interesting future work. In particular, it would be worthwhile studying the possibilities of constructing an IND-CCA1 secure FHE scheme. Such schemes do not exist yet. This is due to the fact that the bootstrapping technique is the only way to construct FHE schemes that we are currently aware of. Unfortunately, boostrapping is not designed to give you IND-CCA1 security. In fact, it is trivial to break the IND-CCA1 security of FHE scheme by simply decrypting the encrypted secret key bits in the decryption phase.

New Constructions with Unique Features

Additionally to our security characterizations, we give a characterization of shift-type GHE schemes in terms of their design. More precisely, Chapter 5 gives a universal blueprint (a generic scheme) for such cryptosystems, meaning that every scheme is an instance of the generic scheme. With this result, we are able to construct new GHE scheme with unique features. As the main contribution, we give the first GHE scheme that is provably IND-CCA1 secure in the generic group model and instantiable in bilinear groups. Moreover, we construct the first additively HE scheme on elliptic curves that has two independent decryption mechanisms. Such schemes have interesting applications in the secure outsourcing of computations to a cloud provider as we show in Chapter 7.

While our additively HE scheme is based on elliptic curves over rings, it is still an open question whether we can construct an additively HE scheme (with plaintext space exponentially large in the security parameter) that is based on elliptic curves over fields. Such a scheme would be very desirable for implementations on resource-constrained devices such as smart cards for which elliptic curves over fields are very suitable.

Impossibility Results

The main consequence of our IND-CPA security characterization is the general (in the Abelian case) impossibility of GHE in the quantum world (Chapter 6). Furthermore, we deduce the impossibility of using a linear code as the ciphertext space of an IND-CPA secure GHE scheme. Our result in the quantum world is quite strong and literally brings the topic of GHE to an end once efficient quantum computers exist. It additionally has the corollary that every new encryption scheme whose security is based on a quantum-resistant hardness assumption should be checked for the group homomorphic property. This is because our impossibility result shows that if the new scheme is group homomorphic it cannot be secure in the quantum world and so, the underlying hardness assumption cannot be quantum-resistant.

We consider the extension of our quantum impossibility result to general solvable groups (non-Abelian) as interesting future work. Although, this does not seem to be practically relevant (as some form of commutativity is always needed in practical applications), it is at least of great theoretical interest.

Efficient and Secure Outsourcing of Multiparty Computations

In Chapter 7, we show that additively HE schemes with a double decryption mechanism (such as the one we propose in Chapter 5) have a strong impact on the secure outsourcing of arbitrary computations. In fact, we give the first construction that allows multiple clients to send encryptions of their private data under their own public key to a distrusted server that can then, on behalf of the clients, compute any dynamically chosen function on these inputs in the encrypted domain (even across encryptions under different public keys) without learning anything at all, and non-interactively with the clients.

The main contribution of this result is in practical applications where interaction with clients should be minimal, as in privacy-preserving face recognition or private smart metering. Our general-purpose solution runs very efficiently in such settings, but is only secure against semi-honest adversaries. The extension to malicious adversaries is treated as future work. We further want to evaluate the performance of our solution in other application scenarios as well, such as electronic healthcare scenarios.

Bibliography

[1] The database of faces, (formerly 'the orl database of faces'). AT&T Laboratories Cambridge. http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html.

[2] Google admits weekend privacy breach. Fox News, March 10, 2009. http://www.foxnews.com/story/0,2933,508324,00.html.

[3] Guess what? facebook monitors postings and chats. Daily News, July 16, 2012. http://articles.nydailynews.com/2012-07-16/news/32701753_1_facebook-employees-conversations-software.

[4] The facebook privacy scandal. ABC News, October 19, 2010. http://abcnews.go.com/GMA/Consumer/facebook-privacy-scandal-facebooks-watergate/story?id=11912201.

[5] Camurati: Recognizing the dangers behind facial recognition. Newsday, July 27, 2012. http://www.newsday.com/opinion/viewsday-1.3683911/camurati-recognizing-the-dangers-behind-facial-recognition-1.3865294.

[6] A face is exposed for AOL searcher no. 4417749. The New York Times, August 9, 2006. http://www.nytimes.com/2006/08/09/technology/09aol.html.

[7] Frederik Armknecht, Stefan Katzenbeisser, and Andreas Peter. Group homomorphic encryption: characterizations, impossibility results, and applications. *Designs, Codes and Cryptography*, pages 1–24, 2012.

[8] Frederik Armknecht, Stefan Katzenbeisser, and Andreas Peter. Shift-type homomorphic encryption and its application to fully homomorphic encryption. In Aikaterini Mitrokotsa and Serge Vaudenay, editors, *AFRICACRYPT 12:*

5th International Conference on Cryptology in Africa, volume 7374 of *Lecture Notes in Computer Science*, pages 234–251, Ifrance, Morocco, July 10–12, 2012. Springer, Berlin, Germany.

[9] Frederik Armknecht, Andreas Peter, and Stefan Katzenbeisser. A cleaner view on ind-cca1 secure homomorphic encryption using soap. *IACR Cryptology ePrint Archive*, 2010, 2010.

[10] A. O. L. Atkin and F. Morain. Elliptic curves and primality proving. *Math. Comp*, 61:29–68, 1993.

[11] Mihir Bellare, Anand Desai, David Pointcheval, and Phillip Rogaway. Relations among notions of security for public-key encryption schemes. In Hugo Krawczyk, editor, *Advances in Cryptology – CRYPTO'98*, volume 1462 of *Lecture Notes in Computer Science*, pages 26–45, Santa Barbara, CA, USA, August 23–27, 1998. Springer, Berlin, Germany.

[12] Peter Bogetoft, Dan Lund Christensen, Ivan Damgård, Martin Geisler, Thomas Jakobsen, Mikkel Krøigaard, Janus Dam Nielsen, Jesper Buus Nielsen, Kurt Nielsen, Jakob Pagter, Michael I. Schwartzbach, and Tomas Toft. Secure multiparty computation goes live. In Roger Dingledine and Philippe Golle, editors, *FC 2009: 13th International Conference on Financial Cryptography and Data Security*, volume 5628 of *Lecture Notes in Computer Science*, pages 325–343, Accra Beach, Barbados, February 23–26, 2009. Springer, Berlin, Germany.

[13] Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. In Matthew Franklin, editor, *Advances in Cryptology – CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 41–55, Santa Barbara, CA, USA, August 15–19, 2004. Springer, Berlin, Germany.

[14] Dan Boneh and Matthew K. Franklin. Identity based encryption from the Weil pairing. *SIAM Journal on Computing*, 32(3):586–615, 2003.

[15] Dan Boneh and Alice Silverberg. Applications of multilinear forms to cryptography. *Contemporary Mathematics*, 324:71–90, 2002.

[16] Zvika Brakerski. Fully homomorphic encryption without modulus switching from classical GapSVP. In Reihaneh Safavi-Naini and Ran Canetti, editors, *Advances in Cryptology – CRYPTO 2012*, volume 7417 of *Lecture Notes in Computer Science*, pages 868–886, Santa Barbara, CA, USA, August 19–23, 2012. Springer, Berlin, Germany.

[17] Zvika Brakerski and Gil Segev. Better security for deterministic public-key encryption: The auxiliary-input setting. In Phillip Rogaway, editor, *Advances in Cryptology – CRYPTO 2011*, volume 6841 of *Lecture Notes in Computer Science*, pages 543–560, Santa Barbara, CA, USA, August 14–18, 2011. Springer, Berlin, Germany.

[18] Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. In Rafail Ostrovsky, editor, *52nd Annual Symposium on Foundations of Computer Science*, pages 97–106, Palm Springs, California, USA, October 22–25, 2011. IEEE Computer Society Press.

[19] Zvika Brakerski and Vinod Vaikuntanathan. Fully homomorphic encryption from ring-LWE and security for key dependent messages. In Phillip Rogaway, editor, *Advances in Cryptology – CRYPTO 2011*, volume 6841 of *Lecture Notes in Computer Science*, pages 505–524, Santa Barbara, CA, USA, August 14–18, 2011. Springer, Berlin, Germany.

[20] Emmanuel Bresson, Dario Catalano, and David Pointcheval. A simple public-key cryptosystem with a double trapdoor decryption mechanism and its applications. In Chi-Sung Laih, editor, *Advances in Cryptology – ASIACRYPT 2003*, volume 2894 of *Lecture Notes in Computer Science*, pages 37–54, Taipei, Taiwan, November 30 – December 4, 2003. Springer, Berlin, Germany.

[21] Christina Brzuska, Heike Busch, Özgür Dagdelen, Marc Fischlin, Martin Franz, Stefan Katzenbeisser, Mark Manulis, Cristina Onete, Andreas Peter, Bertram Poettering, and Dominique Schröder. Redactable signatures for tree-structured data: Definitions and constructions. In Jianying Zhou and Moti Yung, editors, *ACNS 10: 8th International Conference on Applied Cryptography and Network Security*, volume 6123 of *Lecture Notes in Computer Science*, pages 87–104, Beijing, China, June 22–25, 2010. Springer, Berlin, Germany.

[22] Ran Canetti, Hugo Krawczyk, and Jesper Buus Nielsen. Relaxing chosen-ciphertext security. In Dan Boneh, editor, *Advances in Cryptology – CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 565–582, Santa Barbara, CA, USA, August 17–21, 2003. Springer, Berlin, Germany.

[23] Dario Catalano, Rosario Gennaro, Nick Howgrave-Graham, and Phong Q. Nguyen. Paillier's cryptosystem revisited. In *ACM CCS 01: 8th Conference on Computer and Communications Security*, pages 206–214, Philadelphia, PA, USA, November 5–8, 2001. ACM Press.

[24] Octavian Catrina and Florian Kerschbaum. Fostering the uptake of secure multiparty computation in e-commerce. In *Proceedings of the The Third International Conference on Availability, Reliability and Security, ARES 2008, March 4-7, 2008, Technical University of Catalonia, Barcelona , Spain*, pages 693–700. IEEE Computer Society, 2008.

[25] David Chaum, Claude Crépeau, and Ivan Damgård. Multiparty unconditionally secure protocols. In *20th Annual ACM Symposium on Theory of Computing*, pages 11–19, Chicago, Illinois, USA, May 2–4, 1988. ACM Press.

[26] Sen-Ching S. Cheung and Thinh P. Nguyen. Secure multiparty computation between distrusted networks terminals. *EURASIP J. Information Security*, 2007, 2007.

[27] Seung Geol Choi, Ariel Elbaz, Ari Juels, Tal Malkin, and Moti Yung. Two-party computing with encrypted data. In Kaoru Kurosawa, editor, *Advances in Cryptology – ASIACRYPT 2007*, volume 4833 of *Lecture Notes in Computer Science*, pages 298–314, Kuching, Malaysia, December 2–6, 2007. Springer, Berlin, Germany.

[28] Ronald Cramer, Ivan Damgård, and Jesper Buus Nielsen. Multiparty computation from threshold homomorphic encryption. In Birgit Pfitzmann, editor, *Advances in Cryptology – EUROCRYPT 2001*, volume 2045 of *Lecture Notes in Computer Science*, pages 280–299, Innsbruck, Austria, May 6–10, 2001. Springer, Berlin, Germany.

[29] Ronald Cramer and Victor Shoup. Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In Lars R. Knudsen, editor, *Advances in Cryptology – EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 45–64, Amsterdam, The Netherlands, April 28 – May 2, 2002. Springer, Berlin, Germany.

[30] Ivan Damgård. Towards practical public key systems secure against chosen ciphertext attacks. In Joan Feigenbaum, editor, *Advances in Cryptology – CRYPTO'91*, volume 576 of *Lecture Notes in Computer Science*, pages 445–456, Santa Barbara, CA, USA, August 11–15, 1992. Springer, Berlin, Germany.

[31] Ivan Damgård, Martin Geisler, and Mikkel Krøigaard. Efficient and secure comparison for on-line auctions. In Josef Pieprzyk, Hossein Ghodosi, and Ed Dawson, editors, *ACISP 07: 12th Australasian Conference on Information Security and Privacy*, volume 4586 of *Lecture Notes in Computer Science*, pages 416–430, Townsville, Australia, July 2–4, 2007. Springer, Berlin, Germany.

[32] Ivan Damgård and Yuval Ishai. Constant-round multiparty computation using a black-box pseudorandom generator. In Victor Shoup, editor, *Advances in Cryptology – CRYPTO 2005*, volume 3621 of *Lecture Notes in Computer Science*, pages 378–394, Santa Barbara, CA, USA, August 14–18, 2005. Springer, Berlin, Germany.

[33] Ivan Damgård and Mats Jurik. A generalisation, a simplification and some applications of Paillier's probabilistic public-key system. In Kwangjo Kim, editor, *PKC 2001: 4th International Workshop on Theory and Practice in Public Key Cryptography*, volume 1992 of *Lecture Notes in Computer Science*, pages 119–136, Cheju Island, South Korea, February 13–15, 2001. Springer, Berlin, Germany.

[34] Alexander W. Dent and Steven D. Galbraith. Hidden pairings and trapdoor ddh groups. In *ANTS-VII*, volume 4076 of *LNCS*, pages 436–451. Springer, 2006.

[35] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.

[36] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In G. R. Blakley and David Chaum, editors, *Advances in Cryptology – CRYPTO'84*, volume 196 of *Lecture Notes in Computer Science*, pages 10–18, Santa Barbara, CA, USA, August 19–23, 1985. Springer, Berlin, Germany.

[37] Zekeriya Erkin, Martin Franz, Jorge Guajardo, Stefan Katzenbeisser, Inald Lagendijk, and Tomas Toft. Privacy-preserving face recognition. In *Privacy Enhancing Technologies, 9th International Symposium, PETS 2009, Seattle, WA, USA, August 5-7, 2009. Proceedings*, volume 5672 of *Lecture Notes in Computer Science*, pages 235–253. Springer, 2009.

[38] Zekeriya Erkin, Thijs Veugen, Tomas Toft, and Reginald L. Lagendijk. Generating private recommendations efficiently using homomorphic encryption and data packing. *IEEE Transactions on Information Forensics and Security*, 7(3):1053–1066, 2012.

[39] Uriel Feige, Joe Kilian, and Moni Naor. A minimal model for secure computation (extended abstract). In *26th Annual ACM Symposium on Theory of Computing*, pages 554–563, Montréal, Québec, Canada, May 23–25, 1994. ACM Press.

[40] Michael Fellows and Neal Koblitz. Combinatorial cryptosystems galore! In *Finite Fields: Theory, Applications, and Algorithms*, volume 168 of *Contemporary Mathematics*, pages 51–61. American Mathematical Society, 1993.

[41] Caroline Fontaine and Fabien Galand. A survey of homomorphic encryption for nonspecialists. *EURASIP J. Inf. Secur.*, 2007:15:1–15:15, January 2007.

[42] Felix Fontein. Elliptic curves over rings with a point of view on cryptography and factoring. Diploma Thesis, 2005. `http://user.math.uzh.ch/fontein/diplom-fontein.pdf`.

[43] Martin Franz, Peter Williams, Bogdan Carbunar, Stefan Katzenbeisser, Andreas Peter, Radu Sion, and Miroslava Sotáková. Oblivious outsourced storage with delegation. In George Danezis, editor, *FC 2011: 15th International Conference on Financial Cryptography and Data Security*, volume 7035 of *Lecture Notes in Computer Science*, pages 127–140, Gros Islet, St. Lucia, February 28 – March 4, 2011. Springer, Berlin, Germany.

[44] David Freeman, Michael Scott, and Edlyn Teske. A taxonomy of pairing-friendly elliptic curves. *Journal of Cryptology*, 23(2):224–280, April 2010.

[45] Gerhard Frey and Hans-Georg Rück. A remark concerning m-divisibility and the discrete logarithm in the divisor class group of curves. *Math. Comput.*, 62:865–874, April 1994.

[46] Steven D. Galbraith. Elliptic curve Paillier schemes. *Journal of Cryptology*, 15(2):129–138, 2002.

[47] Steven D. Galbraith and James McKee. The probability that the number of points on an elliptic curve over a finite field is prime. *Journal of the LMS*, 62(03):671–684, 2000.

[48] Steven D. Galbraith and James F. McKee. Pairings on elliptic curves over finite commutative rings. In Nigel P. Smart, editor, *10th IMA International Conference on Cryptography and Coding*, volume 3796 of *Lecture Notes in Computer Science*, pages 392–409, Cirencester, UK, December 19–21, 2005. Springer, Berlin, Germany.

[49] David Galindo and Javier Herranz. On the security of public key cryptosystems with a double decryption mechanism. *Information Processing Letters*, 108(5):279–283, 2008.

[50] Flavio D. Garcia and Bart Jacobs. Privacy-friendly energy-metering via homomorphic encryption. In *Security and Trust Management - 6th International Workshop, STM 2010, Athens, Greece, September 23-24, 2010, Revised*

Selected Papers, volume 6710 of *Lecture Notes in Computer Science*, pages 226–238. Springer, 2010.

[51] C. Gentry. *A fully homomorphic encryption scheme*. PhD thesis, Stanford University, 2009.

[52] Craig Gentry. Fully homomorphic encryption using ideal lattices. In Michael Mitzenmacher, editor, *41st Annual ACM Symposium on Theory of Computing*, pages 169–178, Bethesda, Maryland, USA, May 31 – June 2, 2009. ACM Press.

[53] Craig Gentry and Shai Halevi. Fully homomorphic encryption without squashing using depth-3 arithmetic circuits. In Rafail Ostrovsky, editor, *52nd Annual Symposium on Foundations of Computer Science*, pages 107–109, Palm Springs, California, USA, October 22–25, 2011. IEEE Computer Society Press.

[54] Craig Gentry and Shai Halevi. Implementing Gentry's fully-homomorphic encryption scheme. In Kenneth G. Paterson, editor, *Advances in Cryptology – EUROCRYPT 2011*, volume 6632 of *Lecture Notes in Computer Science*, pages 129–148, Tallinn, Estonia, May 15–19, 2011. Springer, Berlin, Germany.

[55] Craig Gentry, Shai Halevi, and Nigel P. Smart. Better bootstrapping in fully homomorphic encryption. In Marc Fischlin, Johannes Buchmann, and Mark Manulis, editors, *PKC 2012: 15th International Workshop on Theory and Practice in Public Key Cryptography*, volume 7293 of *Lecture Notes in Computer Science*, pages 1–16, Darmstadt, Germany, May 21–23, 2012. Springer, Berlin, Germany.

[56] Craig Gentry, Shai Halevi, and Nigel P. Smart. Fully homomorphic encryption with polylog overhead. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology – EUROCRYPT 2012*, volume 7237 of *Lecture Notes in Computer Science*, pages 465–482, Cambridge, UK, April 15–19, 2012. Springer, Berlin, Germany.

[57] Craig Gentry, Shai Halevi, and Nigel P. Smart. Homomorphic evaluation of the AES circuit. In Reihaneh Safavi-Naini and Ran Canetti, editors, *Advances in Cryptology – CRYPTO 2012*, volume 7417 of *Lecture Notes in Computer Science*, pages 850–867, Santa Barbara, CA, USA, August 19–23, 2012. Springer, Berlin, Germany.

[58] Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. i-Hop homomorphic encryption and rerandomizable Yao circuits. In Tal Rabin, editor, *Advances in Cryptology – CRYPTO 2010*, volume 6223 of *Lecture Notes in Computer Science*, pages 155–172, Santa Barbara, CA, USA, August 15–19, 2010. Springer, Berlin, Germany.

[59] Kristian Gjøsteen. Homomorphic cryptosystems based on subgroup membership problems. In *Mycrypt*, volume 3715 of *LNCS*, pages 314–327. Springer, 2005.

[60] Kristian Gjøsteen. Symmetric subgroup membership problems. In Serge Vaudenay, editor, *PKC 2005: 8th International Workshop on Theory and Practice in Public Key Cryptography*, volume 3386 of *Lecture Notes in Computer Science*, pages 104–119, Les Diablerets, Switzerland, January 23–26, 2005. Springer, Berlin, Germany.

[61] Kristian Gjøsteen. A new security proof for damgård's ElGamal. In David Pointcheval, editor, *Topics in Cryptology – CT-RSA 2006*, volume 3860 of *Lecture Notes in Computer Science*, pages 150–158, San Jose, CA, USA, February 13–17, 2006. Springer, Berlin, Germany.

[62] Oded Goldreich. *Foundations of Cryptography: Basic Applications*, volume 2. Cambridge University Press, Cambridge, UK, 2004.

[63] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game, or a completeness theorem for protocols with honest majority. In Alfred Aho, editor, *19th Annual ACM Symposium on Theory of Computing*, pages 218–229, New York City,, New York, USA, May 25–27, 1987. ACM Press.

[64] Shai Halevi, Yehuda Lindell, and Benny Pinkas. Secure computation on the web: Computing without simultaneous interaction. In Phillip Rogaway, editor, *Advances in Cryptology – CRYPTO 2011*, volume 6841 of *Lecture Notes in Computer Science*, pages 132–150, Santa Barbara, CA, USA, August 14–18, 2011. Springer, Berlin, Germany.

[65] Brett Hemenway and Rafail Ostrovsky. On homomorphic encryption and chosen-ciphertext security. In Marc Fischlin, Johannes Buchmann, and Mark Manulis, editors, *PKC 2012: 15th International Workshop on Theory and Practice in Public Key Cryptography*, volume 7293 of *Lecture Notes in Computer Science*, pages 52–65, Darmstadt, Germany, May 21–23, 2012. Springer, Berlin, Germany.

[66] Dennis Hofheinz and Eike Kiltz. Secure hybrid encryption from weakened key encapsulation. In Alfred Menezes, editor, *Advances in Cryptology – CRYPTO 2007*, volume 4622 of *Lecture Notes in Computer Science*, pages 553–571, Santa Barbara, CA, USA, August 19–23, 2007. Springer, Berlin, Germany.

[67] Andreas Hülsing, J. Buchmann, Stefan G. Weber, Albrecht Petzoldt, Enrico Thomae, Juliane Krämer, Pierre-Louis Cayrel, Paulo Barreto, Annelie Heuser,

and u.a. *Tagungsband des 13. Kryptotags. Workshop der Fachgruppe "Angewandte Kryptographie" der "Gesellschaft für Informatik e.V."*. Security Engineering Group (TU Darmstadt), Darmstadt, December 2010.

[68] Yuval Ishai and Eyal Kushilevitz. Private simultaneous messages protocols with applications. In *ISTCS*, pages 174–184, 1997.

[69] Yuval Ishai and Anat Paskin. Evaluating branching programs on encrypted data. In Salil P. Vadhan, editor, *TCC 2007: 4th Theory of Cryptography Conference*, volume 4392 of *Lecture Notes in Computer Science*, pages 575–594, Amsterdam, The Netherlands, February 21–24, 2007. Springer, Berlin, Germany.

[70] Stanislaw Jarecki and Vitaly Shmatikov. Efficient two-party secure computation on committed inputs. In Moni Naor, editor, *Advances in Cryptology – EUROCRYPT 2007*, volume 4515 of *Lecture Notes in Computer Science*, pages 97–114, Barcelona, Spain, May 20–24, 2007. Springer, Berlin, Germany.

[71] Marek Jawurek, Martin Johns, and Florian Kerschbaum. Plug-in privacy for smart metering billing. In *PETS*, volume 6794 of *LNCS*, pages 192–210. Springer, 2011.

[72] Antoine Joux and Kim Nguyen. Separating decision Diffie-Hellman from computational Diffie-Hellman in cryptographic groups. *Journal of Cryptology*, 16(4):239–247, September 2003.

[73] Seny Kamara, Payman Mohassel, and Mariana Raykova. Outsourcing multiparty computation. Cryptology ePrint Archive, Report 2011/272, 2011. `http://eprint.iacr.org/`.

[74] Ünal Koçabas, Andreas Peter, Stefan Katzenbeisser, and Ahmad-Reza Sadeghi. Converse puf-based authentication. In *Trust and Trustworthy Computing - 5th International Conference, TRUST 2012, Vienna, Austria, June 13-15, 2012. Proceedings*, volume 7344 of *Lecture Notes in Computer Science*, pages 142–158. Springer, 2012.

[75] Vladimir Kolesnikov, Ahmad-Reza Sadeghi, and Thomas Schneider. From dust to dawn: Practically efficient two-party secure function evaluation protocols and their modular design. *IACR Cryptology ePrint Archive*, 2010:79, 2010.

[76] Kenji Koyama, Ueli M. Maurer, Tatsuaki Okamoto, and Scott A. Vanstone. New public-key schemes based on elliptic curves over the ring z_n. In Joan Feigenbaum, editor, *Advances in Cryptology – CRYPTO'91*, volume 576 of

Lecture Notes in Computer Science, pages 252–266, Santa Barbara, CA, USA, August 11–15, 1992. Springer, Berlin, Germany.

[77] Klaus Kursawe, George Danezis, and Markulf Kohlweiss. Privacy-friendly aggregation for the smart-grid. In *Privacy Enhancing Technologies - 11th International Symposium, PETS 2011, Waterloo, ON, Canada, July 27-29, 2011. Proceedings*, volume 6794 of *Lecture Notes in Computer Science*, pages 175–191. Springer, 2011.

[78] Hans Kurzweil and Bernd Stellmacher. *The Theory of Finite Groups: An Introduction*. Springer, 2004.

[79] Serge Lang. *Algebra*, volume 211 of *Graduate Texts in Mathematics*. Springer-Verlag, 2002.

[80] H.W̃. Lenstra. Factoring integers with elliptic curves. *Annals of mathematics*, pages 649–673, 1987.

[81] H.W̃. Lenstra. Elliptic curves and number theoretic algorithms. In *Proceedings of the International Congress of Mathematicians*, pages 99–120, 1988.

[82] Fengjun Li, Bo Luo, and Peng Liu. Secure and privacy-preserving information aggregation for smart grids. *IJSN*, 6(1):28–39, 2011.

[83] Yehuda Lindell and Benny Pinkas. An efficient protocol for secure two-party computation in the presence of malicious adversaries. In Moni Naor, editor, *Advances in Cryptology – EUROCRYPT 2007*, volume 4515 of *Lecture Notes in Computer Science*, pages 52–78, Barcelona, Spain, May 20–24, 2007. Springer, Berlin, Germany.

[84] Helger Lipmaa. On the cca1-security of elgamal and damgård's elgamal. In *Inscrypt*, volume 6584 of *LNCS*, pages 18–35. Springer, 2010.

[85] Adriana López-Alt, Eran Tromer, and Vinod Vaikuntanathan. On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption. In Howard J. Karloff and Toniann Pitassi, editors, *44th Annual ACM Symposium on Theory of Computing*, pages 1219–1234, New York, NY, USA, May 19–22, 2012. ACM Press.

[86] Steve Lu and Rafail Ostrovsky. Distributed oblivious ram for secure two-party computation. Cryptology ePrint Archive, Report 2011/384, 2011. `http://eprint.iacr.org/`.

[87] Ueli M. Maurer. Abstract models of computation in cryptography (invited paper). In Nigel P. Smart, editor, *10th IMA International Conference on*

Cryptography and Coding, volume 3796 of *Lecture Notes in Computer Science*, pages 1–12, Cirencester, UK, December 19–21, 2005. Springer, Berlin, Germany.

[88] Carlos Aguilar Melchor, Guilhem Castagnos, and Philippe Gaborit. Lattice-based homomorphic encryption of vector spaces. In *IEEE International Symposium on Information Theory, ISIT 2008, Toronto, ON, Canada, July 6-11, 2008*, pages 1858–1862. IEEE, 2008.

[89] Alfred Menezes, Tatsuaki Okamoto, and Scott A. Vanstone. Reducing elliptic curve logarithms to logarithms in a finite field. *IEEE Trans Inf Theory*, 39(5):1639–1646, 1993.

[90] Andres Molina-Markham, Prashant Shenoy, Kevin Fu, Emmanuel Cecchet, and David Irwin. Private memoirs of a smart meter. In *BuildSys*, 2010.

[91] Moni Naor, Benny Pinkas, and Reuban Sumner. Privacy preserving auctions and mechanism design. In *ACM Conference on Electronic Commerce*, pages 129–139, 1999.

[92] Tatsuaki Okamoto and Shigenori Uchiyama. Security of an identity-based cryptosystem and the related reductions. In Kaisa Nyberg, editor, *Advances in Cryptology – EUROCRYPT'98*, volume 1403 of *Lecture Notes in Computer Science*, pages 546–560, Espoo, Finland, May 31 – June 4, 1998. Springer, Berlin, Germany.

[93] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In Jacques Stern, editor, *Advances in Cryptology – EUROCRYPT'99*, volume 1592 of *Lecture Notes in Computer Science*, pages 223–238, Prague, Czech Republic, May 2–6, 1999. Springer, Berlin, Germany.

[94] Igor Pak and Sergey Bratus. On sampling generating sets of finite groups and product replacement algorithm (extended abstract). In *ISSAC*, pages 91–96. ACM, 1999.

[95] Andreas Peter. The arithmetic of elliptic and hyperelliptic curves with applications to pairing-based cryptography. Diploma thesis, University of Oldenburg, Germany, March 24, 2009.

[96] Andreas Peter, Thomas Hartmann, Sascha Müller, and Stefan Katzenbeisser. Privacy-preserving architecture for forensic image recognition. In *Information Forensics and Security (WIFS), 2012 IEEE International Workshop on*, 2012.

[97] Andreas Peter, Max Kronberg, Wilke Trei, and Stefan Katzenbeisser. Additively homomorphic encryption with a double decryption mechanism, revisited. In Dieter Gollmann and Felix C. Freiling, editors, *ISC 2012: 15th International Conference on Information Security*, volume 7483 of *Lecture Notes in Computer Science*, pages 242–257, Passau, Germany, September 19–21, 2012. Springer, Berlin, Germany.

[98] Andreas Peter, Erik Tews, and Stefan Katzenbeisser. Efficiently outsourcing multiparty computation under multiple keys. Cryptology ePrint Archive, Report 2013/013, 2013. `http://eprint.iacr.org/`.

[99] Benny Pinkas, Thomas Schneider, Nigel P. Smart, and Stephen C. Williams. Secure two-party computation is practical. In Mitsuru Matsui, editor, *Advances in Cryptology – ASIACRYPT 2009*, volume 5912 of *Lecture Notes in Computer Science*, pages 250–267, Tokyo, Japan, December 6–10, 2009. Springer, Berlin, Germany.

[100] Manoj Prabhakaran and Mike Rosulek. Homomorphic encryption with CCA security. In Luca Aceto, Ivan Damgård, Leslie Ann Goldberg, Magnús M. Halldórsson, Anna Ingólfsdóttir, and Igor Walukiewicz, editors, *ICALP 2008: 35th International Colloquium on Automata, Languages and Programming, Part II*, volume 5126 of *Lecture Notes in Computer Science*, pages 667–678, Reykjavik, Iceland, July 7–11, 2008. Springer, Berlin, Germany.

[101] Thomas Ristenpart, Eran Tromer, Hovav Shacham, and Stefan Savage. Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds. In Ehab Al-Shaer, Somesh Jha, and Angelos D. Keromytis, editors, *ACM CCS 09: 16th Conference on Computer and Communications Security*, pages 199–212, Chicago, Illinois, USA, November 9–13, 2009. ACM Press.

[102] Ahmad-Reza Sadeghi, Thomas Schneider, and Immo Wehrenberg. Efficient privacy-preserving face recognition. In Donghoon Lee and Seokhie Hong, editors, *ICISC 09: 12th International Conference on Information Security and Cryptology*, volume 5984 of *Lecture Notes in Computer Science*, pages 229–244, Seoul, Korea, December 2–4, 2009. Springer, Berlin, Germany.

[103] Hovav Shacham. A cramer-shoup encryption scheme from the linear assumption and from progressively weaker linear variants. Cryptology ePrint Archive, Report 2007/074, 2007. `http://eprint.iacr.org/`.

[104] Adi Shamir. Identity-based cryptosystems and signature schemes. In G. R. Blakley and David Chaum, editors, *Advances in Cryptology – CRYPTO'84*,

volume 196 of *Lecture Notes in Computer Science*, pages 47–53, Santa Barbara, CA, USA, August 19–23, 1985. Springer, Berlin, Germany.

[105] Victor Shoup. Lower bounds for discrete logarithms and related problems. In Walter Fumy, editor, *Advances in Cryptology – EUROCRYPT'97*, volume 1233 of *Lecture Notes in Computer Science*, pages 256–266, Konstanz, Germany, May 11–15, 1997. Springer, Berlin, Germany.

[106] J.H̃. Silverman. *The Arithmetic of Elliptic Curves.* GTM 106. Springer, 1986.

[107] Yiannis Tsiounis and Moti Yung. On the security of ElGamal based encryption. In Hideki Imai and Yuliang Zheng, editors, *PKC'98: 1st International Workshop on Theory and Practice in Public Key Cryptography*, volume 1431 of *Lecture Notes in Computer Science*, pages 117–134, Pacifico Yokohama, Japan, February 5–6, 1998. Springer, Berlin, Germany.

[108] M. A. Turk and A. P. Pentland. Face recognition using eigenfaces. In *CVPR*, pages 586–591. IEEE Comput. Sco. Press, 1991.

[109] Matthew Turk and Alex Pentland. Eigenfaces for recognition. *J. Cognitive Neuroscience*, 3(1):71–86, January 1991.

[110] Marten van Dijk, Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. Fully homomorphic encryption over the integers. In Henri Gilbert, editor, *Advances in Cryptology – EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 24–43, French Riviera, May 30 – June 3, 2010. Springer, Berlin, Germany.

[111] Marten Van Dijk and Ari Juels. On the impossibility of cryptography alone for privacy-preserving cloud computing. In *Proceedings of the 5th USENIX conference on Hot topics in security*, HotSec'10, pages 1–8. USENIX Association, 2010.

[112] John Watrous. Quantum algorithms for solvable groups. In *33rd Annual ACM Symposium on Theory of Computing*, pages 60–67, Crete, Greece, July 6–8, 2001. ACM Press.

[113] J. Wu and D.R. Stinson. On the security of the elgamal encryption scheme and damgard's variant. Cryptology ePrint Archive, Report 2008/200, 2008. `http://eprint.iacr.org/`.

[114] Andrew C. Yao. Protocols for secure computations. In *23rd Annual Symposium on Foundations of Computer Science*, pages 160–164, Chicago, Illinois, November 3–5, 1982. IEEE Computer Society Press.

[115] Andrew Chi-Chih Yao. How to generate and exchange secrets (extended abstract). In *27th Annual Symposium on Foundations of Computer Science, Toronto, Canada, 27-29 October 1986*, pages 162–167. IEEE Computer Society, 1986.

[116] Taek-Young Youn, Young-Ho Park, Chang Han Kim, and Jongin Lim. An efficient public key cryptosystem with a privacy enhanced double decryption mechanism. In Bart Preneel and Stafford Tavares, editors, *SAC 2005: 12th Annual International Workshop on Selected Areas in Cryptography*, volume 3897 of *Lecture Notes in Computer Science*, pages 144–158, Kingston, Ontario, Canada, August 11–12, 2005. Springer, Berlin, Germany.

A Publication Record

Proceedings Volumes

[67] A. Peter and S. Katzenbeisser, **Proceedings of 13th Cryptoday (13. Kryptotags)**. In *tuprints - Dokumenten- und Publikationsservice der TU Darmstadt: 2371*, January 5, 2011.

Journal Articles

[7] F. Armknecht, S. Katzenbeisser, and A. Peter, **Group Homomorphic Encryption: Characterizations, Impossibility Results, and Applications**. In *Designs, Codes and Cryptography (DESI)*, February 2, 2012. DOI: 10.1007/s10623-011-9601-2

($\longrightarrow$ Chapters 4–6)

Conferences and Workshops with Proceedings

[21] C. Brzuska, H. Busch, Ö. Dagdelen, M. Fischlin, M. Franz, S. Katzenbeisser, M. Manulis, C. Onete, A. Peter, B. Poettering, and D. Schröder, **Redactable Signatures for Tree-Structured Data: Definitions and Constructions**. In *8th International Conference on Applied Cryptography and Network Security (ACNS 2010)*, vol. 6123 of LNCS, pp. 87-104, Springer, June 22-25, 2010

[43] M. Franz, P. Williams, B. Carbunar, S. Katzenbeisser, A. Peter, R. Sion, and M. Sotakova, **Oblivious Outsourced Storage with Delegation**. In *15th International Conference on Financial Cryptography and Data Security (FC 2011)*, vol. 7035 of LNCS, pp. 127-140, Springer, February 28-March 4, 2012

[74] Ü. Kocabas, A. Peter, S. Katzenbeisser, and A.-R. Sadeghi, **Converse PUF-based Authentication**. In *5th International Conference on Trust and Trustworthy Computing (TRUST 2012)*, vol. 7344 of LNCS, pp. 142-158, Springer, June 13-15, 2012

[8] F. Armknecht, S. Katzenbeisser, and A. Peter, **Shift-Type Homomorphic Encryption and its Application to Fully Homomorphic Encryption**. In *5th International Conference on Cryptology in Africa (AfricaCrypt 2012)*, vol. 7374 of LNCS, pp. 234-251, Springer, July 10-12, 2012

($\longrightarrow$ Chapter 4)

[97] A. Peter, M. Kronberg, W. Trei, and S. Katzenbeisser, **Additively Homomorphic Encryption with a Double Decryption Mechanism, Revisited**. In *15th International Conference on Information Security (ISC 2012)*, vol. 7483 of LNCS, pp. 242-257, Springer, September 19-21, 2012

($\longrightarrow$ Chapter 5)

[96] A. Peter, T. Hartmann, S. Müller, and S. Katzenbeisser, **Privacy-Preserving Architecture for Forensic Image Recognition**. In *4th IEEE International Workshop on Information Forensics and Security (WIFS 2012)*, IEEE Press, December 2-5, 2012

Technical Reports

[9] F. Armknecht, S. Katzenbeisser, and A. Peter, **A Cleaner View on IND-CCA1 Secure Homomorphic Encryption using SOAP**. *Cryptology ePrint Archive: Report 2010/501*, September 29, 2010

[98] A. Peter, E. Tews, and S. Katzenbeisser, **Efficiently Outsourcing Multiparty Computation under Multiple Keys**. *Cryptology ePrint Archive: Report 2013/013*, January 12, 2013

($\longrightarrow$ Chapter 7)

Theses

[95] A. Peter, **The Arithmetic of Elliptic and Hyperelliptic Curves with Applications to Pairing-Based Cryptography**. Diploma Thesis, University of Oldenburg, Germany, March 24, 2009

Workshops without Proceedings

- A. Peter, **Homomorphic Encryption with a Double Decryption Mechanism based on Elliptic Curves over Rings**. In *15. Workshop der Fachgruppe Kryptographie in der Gesellschaft für Informatik (Kryptotag)*, vol. WSI-2011-12, December 1-2, 2011

In Submission

- M. Franz, S. Katzenbeisser, A. Peter, R. Sion, and P. Williams, **Oblivious Databases without Central Management**. In Submission.
- F. Armknecht, T. Gagliardoni, S. Katzenbeisser, and A. Peter, **General Impossibility of Group Homomorphic Encryption in the Quantum World**. In Submission.

  ($\longrightarrow$ **Chapter 6**)

Wissenschaftlicher Werdegang

Oktober 2003 – März 2009

Studium der Mathematik (Diplom) an der Universität Oldenburg

Oktober 2007 – Juni 2008

Studium der Mathematik (Master of Advanced Study) an der University of Cambridge, England

Mai 2009 – Februar 2013

Promotion an der TU Darmstadt unter Leitung von Prof. Dr. Stefan Katzenbeisser

Mai 2009 – Februar 2013

Wissenschaftlicher Mitarbeiter am Fachgebiet “Security Engineering Group” an der TU Darmstadt