Homomorphic Encryption in the Attribute-Based Setting

Michael Clear

A Dissertation submitted to the University of Dublin, Trinity College

in fulfillment of the requirements for the degree of

Doctor of Philosophy (Computer Science)

April 2016

Declaration

I declare that this thesis has not been submitted as an exercise for a degree at this or any other university and it is entirely my own work.

I agree to deposit this thesis in the University's open access institutional repository or allow the library to do so on my behalf, subject to Irish Copyright Legislation and Trinity College Library conditions of use and acknowledgement.

Michael Clear

Dated: April 28, 2016

Summary

The main research question addressed by this thesis is whether we can combine a cryptographic primitive called homomorphic encryption with a cryptographic primitive called attribute based encryption. Firstly, let us define both homomorphic encryption and attribute based encryption.

Homomorphic Encryption (HE) involves the ability to operate on encrypted data without knowing the secret key. Attribute Based Encryption (ABE) provides fine-grained access control over data by allowing an entity to encrypt data with attributes which must be satisfied by a decryptor's access policy in order for decryption to succeed. A special case of ABE is identity-based encryption (IBE) where the access policies are simple equality relations i.e. each access policy is satisfied by a singular attribute.

A natural question is whether we can have both the functionality of ABE and the functionality of HE in the same cryptographic primitive? In this thesis, attribute based homomorphic encryption (ABHE) is introduced which combines the functionality of ABE with the functionality of HE. We say an ABHE scheme is multi-attribute if it supports evaluation on ciphertexts with different attributes. In contrast, a single-attribute scheme only supports evaluation on ciphertexts with the same attribute. Therefore, in sum, there are three facets that describe an ABHE scheme: (1). its supported class of circuits (as in HE); (2). its supported class of access policies (as in ABE); and (3). whether it is single-attribute or multi-attribute.

Suppose one were to maximize each of the above facets. This would give us multi-

attribute attribute-based fully homomorphic encryption (ABFHE) for all polynomial-time access policies. An important research question then is whether a multi-attribute ABFHE scheme for all polynomial time access policies can be constructed under reasonable cryptographic assumptions? In this thesis, we answer this question in the affirmative; we construct and prove the security of the first multi-attribute ABFHE for all polynomial-time access policies.

Beyond this feasibility result, we also explore more "concrete" constructions of ABHE i.e. schemes that are more conducive to practical realization (i.e. are implementable at the current time). On this front, we begin by tackling a flavor of HE called group homomorphic encryption (GHE) from an attribute-based perspective. We define attribute based group homomorphic encryption (ABGHE) and give an instance of an ABGHE scheme. More precisely, this instance is in fact an identity-based scheme that is homomorphic for the XOR operation. We prove the scheme secure under the quadratic residuosity problem in the random oracle model.

We then turn our attention to the second flavor of HE, namely fully homomorphic encryption (FHE). Therefore, we explore attribute based fully homomorphic encryption (ABGHE). Our first result is a compiler to transform any ABHE scheme that can evaluate "shallow" (i.e. polylog depth) circuits into one that can evaluate circuits of arbitrary depth, but with a bounded number of inputs N. We also present an identity-based FHE scheme that can evaluate circuits up to a bounded depth that is specified in advance of generating the public parameters (this is called leveled FHE in the literature) . This scheme allows evaluation on ciphertexts with different identities (hence it is multi-identity"). This scheme can be used to instantiate our previous result to obtain a multi-identity identity-based FHE scheme that can evaluate circuits of arbitrary depth but with a bounded number of inputs.

Acknowledgements

Firstly I would like to thank Hitesh Tewari and Ciaran McGoldrick for supervising me over the past few years. Hitesh, I am thankful for the help you've given me over the years. I'm extremely grateful to Ciaran for the all the hard work he has put in helping me with submitting, correcting my poorly written drafts and giving me plenty of ideas and encouragement - this thesis would not have been finished without you.

I would like to thank Arthur Hughes for meeting with me many times and discussing interesting mathematical topics with me; you gave me great inspiration and support.

Thanks to everyone in DSG for all the interesting chats and nights out throughout the last four years; there are too many people here to name! Thanks to all my friends for helping me find fun distractions from the thesis so many times.

My work has been funded by the Irish Research Council, so I would like to acknowledge their support.

I would like to thank Colm Bhandal, Brendan Cody-Kenny, Paul Duggan, Meriel Huggard, Steven Kelly, Paul Laird and Paul Woods for reading my drafts and taking time to give me feedback.

Finally, I would like to thank my Mam, Dad and brother Aidan for all the support they have given me.

Michael Clear

University of Dublin, Trinity College

April 2016

Publications Related to this Ph.D.

- Clear, M., Hughes, A., and Tewari, H. *Homomorphic Encryption with Access Policies: Characterization and New Constructions.* Africacrypt 2013. [60]
- Clear, M., and McGoldrick, C. *Policy-Based Non-interactive Outsourcing of Computation using multikey FHE and CP-ABE.* SECRYPT 2013. [62]
- Clear, M., Tewari, H., and McGoldrick, C. *Anonymous IBE from Quadratic Residuosity with Improved Performance.* Africacrypt 2014. [65]
- Clear, M., and McGoldrick, C. *Bootstrappable Identity-Based Fully Homomorphic Encryption.* CANS 2014. [63]
- Clear, M., and McGoldrick, C. *Multi-identity and Multi-Key Leveled FHE from Learning with Errors.* CRYPTO 2015. [64]
- Clear, M., and McGoldrick, C. *Attribute-Based Fully Homomorphic Encryption with a Bounded Number of Inputs.* Africacrypt 2016. [61]

Notation and Abbreviations

Notation

$\langle x_1, \ldots, x_n \rangle$	Sequence of items $x_1, \ldots, x_n$.
$\vec{\mathbf{v}}$	Vector (always written in boldface)
$\mathbf{M}$	Matrix (always written in boldface)
$\langle \vec{\mathbf{u}}, \vec{\mathbf{v}} \rangle$	Inner (dot) product between vectors $\vec{\mathbf{u}}$ and $\vec{\mathbf{v}}$.
$[n]$	The set of elements $\{1, \ldots, n\}$.
$x \xleftarrow{\$} D$ where D is a distribution	This notation means that x is sampled according to the distribution D.
$x \xleftarrow{\$} S$ where S is a set	This notation means that x is sampled according to the uniform distribution on S.
$\mathsf{supp}(f)$ where f is a function	The set of elements from the domain X of f that map to a non-zero value under f i.e. the set $\{x \in X : f(x) \neq 0\}$.
$y = \mathsf{poly}(n)$	This expression means there exists a fixed polynomial $p(x)$ such that $y = p(n)$.
$y = \mathsf{negl}(n)$	This expression means that $y < 1/p(n)$ for every polynomial $p(x)$.
$D_1 \underset{C}{\approx} D_2$	The distributions D_1 and D_2 are computationally indistinguishable.
$D_1 \underset{S}{\approx} D_2$	The distributions D_1 and D_2 are statistically indistinguishable.

Abbreviations

IND-CPA	Indistinguishability under a chosen plaintext attack.
FHE	Fully Homomorphic Encryption
IBE	Identity Based Encryption
ABE	Attribute Based Encryption
ABHE	Attribute Based Homomorphic Encryption
ABGHE	Attribute Based Group Homomorphic Encryption
ABFHE	Attribute Based Fully Homomorphic Encryption
IBFHE	Identity Based Fully Homomorphic Encryption

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Individuals and organizations often need to outsource computation to a third party, usually to benefit from the computational resources offered. This has taken place since at least the 1950's when jobs were delegated to mainframe computers, remotely accessed via terminals.

Security and privacy are significant issues when considering outsourcing computation to a third party. Trusting a third party such as a "cloud" provider with security might be assessed as high-risk, especially when one considers the fact that the provider is exposed to the public internet. Chow et al. [59] categorize the security concerns in "cloud" computing as: (1). *traditional security*; (2). *availability*; and (3). *third-party data control*. These three principal categories can be described as follows: *traditional security* encompasses network and server intrusions; *availability* encompasses uptime, redundancy, and computational integrity; and *third-party data control* encompasses the control and transparency of data held by the cloud provider.

To highlight these concerns, we take a closer look at some examples in each of the above categories. In regard to network and server intrusions, the cloud provider is susceptible to common types of attack against software and system vulnerabilities. These include buffer overflows, buffer over-reads, cross-site scripting (XSS) and SQL injection.

Vulnerabilities may be found at the virtual machine (VM) level i.e. in the VM monitor (hypervisor), or they may be found at the platform level i.e. in server software such as web servers. Vulnerabilities have been found in several different hypervisors [59]. A vulnerability found in the hypervisor VMWare in 2009 allowed an attacker to escape the guest machine and take control of the host [127]. More common are exploits that arise at the platform level. A case-in-point is the "Heartbleed" [1] buffer over-read vulnerability which was disclosed in April 2014. Heartbleed is a serious security bug in the OpenSSL library, which allows an attacker to expose up to 64 KB of the server's memory (from the heap). OpenSSL is a widely-deployed implementation of the Transport Layer Security (TLS) protocol that is used by many websites to protect the confidentiality and integrity of their users' traffic. It is estimated that approximately 17% of web servers that use TLS were affected by Heartbleed [3]. This naturally includes a large number of cloud services. An attacker can easily exploit the vulnerability to expose sensitive information such as passwords, session cookies and credit card information, and most critically in some cases, the private key associated with the server's certificate. The attacker could accomplish this without a log entry being generated on the server.

It is not fully known whether Heartbleed was known to attackers prior to its public disclosure, although suspicious packets resembling the attack have been found in network logs [5]. If this were indeed the case, then Heartbleed would be an example of a *zero-day vulnerability* (one that is known to attackers but not the public at large). Bilge and Dumitras [31] found 18 zero-day vulnerabilities in data they collected from over 11 million hosts using the worldwide intelligence network environment (WINE); the data was analyzed in retrospect from 2008 to 2011 to determine whether vulnerabilities were used by attackers prior to their official disclosure. The zero-day vulnerabilities they found remained undetected for between 19 days to 30 months, and on average, 312 days. In light of these threats, an organization has to ensure a cloud provider adheres to robust security practices to reduce the risk of attack to their sensitive data.

Availability is another important issue when delegating computation. The robustness

of the provider against denial of service attacks is important to ensure the service remains reachable. Redundancy is essential to protect against data loss and system outages. Also in this category is assurance of computational integrity; in other words, the guarantee that the correct computation was performed by the provider.

Third-party control of data is another factor that weighs heavily on an organization's decision to outsource. The organization must assess the integrity of the cloud provider; it must assess its relationship with domestic and international governments such as its handling of subpoenas; it must assess whether the provider subcontracts its services [59]; and it must assess the threat of insider attacks [123]. The disclosures in 2013 of mass surveillance by intelligence agencies such as the NSA and GCHQ [2] brings many of these concerns sharply into focus. As part of this surveillance, intelligence agencies gathered private customer data from cloud providers either with their co-operation or by some other means, such as via infiltration, eavesdropping or targeted intrusion. These threats to privacy coupled with exposure to a wide array of attack vectors undoubtedly dissuade many from outsourcing their computation.

As we have seen, there are many significant issues surrounding data privacy when one considers outsourcing computation to any third party. An individual or organization that sends potentially sensitive data to a remote facility loses some degree of control over that data. They have to trust that the third party is honest, provides state-of-the-art security measures against external intruders, and safeguards against insider attacks (e.g: malicious employees). Furthermore, they have to ensure that there are no legal or regulatory barriers preventing the data being held by the third party.

For these reasons, supplying the data in encrypted form is preferable, since it protects confidentiality in the face of eavesdropping and/or intrusion, and potentially addresses data protection requirements. However, if the data is encrypted, in order for the third party to carry out the desired computation on this data, it needs to be able to operate on encrypted data without being able to decrypt it. This notion is known as *homomorphic encryption*. In brief, homomorphic encryption allows an operation to be performed on

one or more ciphertexts such that a corresponding meaningful operation is performed on the underlying plaintexts, and this can be done without the secret key. Our use of the term homomorphic encryption without further qualification specifically refers to the public-key setting. More precisely, in this setting, an encryptor uses a recipient's public key to encrypt a message, which the recipient can then decrypt with her corresponding private key. Ciphertexts created with the same public key can be operated on *homomorphically*.

If encrypted data is to be stored by a third party so that homomorphic computation can be carried out, then one would expect that users within the delegator's organization be allowed to query and fetch certain portions of it - both ciphertexts corresponding to inputs and ciphertexts corresponding to outputs (i.e. the results of a computation). In the following discussion our mention of "organization" refers to the delegator's organization.

To gain the full benefits of the wider accessibility of information from outside the organization, it is desirable to grant access to this data in a non-interactive manner; that is, without having to mediate access to it through a server within the organization. To achieve this functionality, cryptographic access control is required. Cryptographic access control means that access is granted to a piece of data *non-interactively* through a cryptographic process, as opposed to being enforced by a centralized system, *interactively*, as in traditional access control.

Attribute Based Encryption (ABE) is a cryptographic primitive that realizes the notion of cryptographic access control. ABE owes its roots to a simpler primitive called Identity Based Encryption (IBE), proposed in 1985 by Shamir [167] and first realized in 2001 by Boneh and Franklin [38] and Cocks [66]. IBE is centered around the notion that a user's public key can be efficiently derived from an identity string and a system-wide *master public key*. Another name for the master public key in the literature is the *public parameters*; we adopt this term in this thesis. The identity string may be a person's email address, IP address or staff number, depending on the application. The public

parameters along with a secret trapdoor (master secret key) are generated by a trusted third party referred to as the Trusted Authority (TA). The primary purpose of the TA is to issue a secret key to a user that corresponds to her identity string (we abbreviate this to *identity*) over a secure channel. The means by which the users authenticate to the TA or establish a secure channel are outside the scope of IBE. The TA uses the master secret key to derive the secret keys for identities. It is assumed that all parties have a priori access to the public parameters. For instance, the public parameters may be hard-coded in the software used by the participants, or made available on a public website.

ABE was proposed in 2005 by Sahai and Waters [164]. ABE can be viewed as a generalization of IBE. In ABE, the TA generates secret keys instead for *access policies* (an access policy prescribes the types of data a user is authorized to access). An encryptor Alice can use the public parameters to encrypt data, and embed within the ciphertext a *descriptor* of her choice that suitably describes her data. The descriptor is referred to as an *attribute*. We caution the reader that although the term *attribute* is used here in its singular form, it may in fact incorporate a *collection* of descriptive elements (which we call "subattributes"). To illustrate this, an example of an attribute is {"CS", "CRYPTO"}; it consists of the subattributes "CS" and "CRYPTO". Let us assume that this is the attribute chosen by Alice. Suppose the TA has issued a user Bob a secret key for his access policy. Keeping with the above example, suppose his access policy "accepts" an attribute if it contains **both** the subattributes "CS" and "CRYPTO". It follows that Alice's chosen attribute satisfies Bob's access policy. As such, Bob can use his secret key to decrypt Alice's ciphertext. Notice that IBE is a special case of ABE. One way of looking at an IBE scheme is that each attribute corresponds to a unique identity string such as an email address or phone number. In IBE, there is a one-to-one mapping between attributes and access policies, so Alice is given a secret key for a policy that is singularly satisfied by her identity string (e.g: email address). On the other hand, ABE, supporting a richer class of policies, means that a

user's access policy might be satisfied by many attributes. Conversely, an attribute may satisfy many policies. Formally, an access policy is a predicate over attributes.

To grant a user access to data her credentials allow her access to, the TA (which is hosted by the organization) authenticates the user, determines her access policy (from her credentials), generates a corresponding secret key, and issues the secret key to the user. This is an interactive process that need only take place infrequently*. Issued with a secret key, a user can non-interactively decrypt an unbounded number of ciphertexts that satisfy her policy. ABE delivers fine-grained access control with minimal interaction, and its use has been explored in many applications including distributed file systems [159], social networks [159] and management of personal health records [14, 119, 143].

Suppose an organization chooses to avail of ABE for access control. Can the organization still use it in conjunction with privacy-preserving outsourced computation? An issue here is that one "type" of encryption is needed for privacy-preserving computation (namely, homomorphic encryption) and another "type" of encryption is needed for access control (namely, attribute-based encryption (ABE)). How can one obtain the "best of both worlds"? Thus a natural question is whether these two types can be reconciled.

It turns out that homomorphic encryption in the attribute-based setting is nontrivial to achieve. This thesis is focused on characterizing and realizing *attribute-based homomorphic encryption*. In the next section, we give an informal overview of this notion, and discuss the motivations for studying it with the help of application scenarios.

*The frequency is determined by measures to address revocation; i.e. making the access policies time-limited. There is a trade-off between the number of secret key updates and the window of unauthorized exposure in the event of a key compromise or access revocation.

1.1 Motivation

1.1.1 Overview of the Problem Domain

A coarse-grained view of homomorphic encryption (HE) is that it comes in two flavors. One flavor allows a single operation such as addition or multiplication to be evaluated on encrypted data; this is referred to as *partial homomorphic encryption*. A more powerful flavor is *fully homomorphic encryption* (FHE) that allows arbitrary computation to be performed on encrypted data. FHE was first constructed in 2009 in a breakthrough work by Gentry [93]. We discuss HE in more detail later in this chapter in Section 1.3.

In standard (i.e. public-key) HE, there is only a single target recipient. This may be ill-suited to the needs of a large organization such as a university. Consider a scenario where university staff have restricted access to data based on their department and position. The university has opted to avail of the computational resources of a third party such as a cloud provider for the purpose of delegating sizable computational tasks. We call this third party the *evaluator*. Each sender of data acts independently since they are potentially unaware of each other's participation.

To comply with the organization's privacy regulations, each sender must encrypt her data with an appropriate *attribute* that describes the data in question, and which is used to ensure only staff members with qualifying credentials can access the data (or any derivative thereof). Recall that an *attribute* may contain subattributes. We assume an appropriate attribute can be feasibly determined from the data source and context.

The computation to be performed, and the inputs to be used, may be decided at a later stage by a subset of the senders, or indeed another party entirely, including the evaluator itself.

The results of the computation are then stored by the evaluator so that they can be queried by staff members. The results should *only* be decryptable by a given staff member if her access policy is satisfied by the attributes associated with *all* the inputs used.

One approach to satisfy these requirements is to use public-key HE together with a trusted access control system (ACS), which holds the private key for the HE scheme. The role of the ACS is to grant users (i.e. staff members in the above scenario) access to a plaintext after verifying that their access policy gives them permission to recover the plaintext [†] Access control of this form facilitates expressive policies.

This approach however suffers from a number of drawbacks:

- All parties interested in a result are required to contact the ACS, which must remain online and exhibit high availability in order to guarantee satisfactory responsiveness. The ACS may therefore act as a bottleneck, especially under high load scenarios.

- Adhering to the principle of least privilege, the organization may wish to limit the capabilities of the ACS. In particular, it may have reservations about the ACS being compromised, and potentially providing an attacker access to all results returned from the cloud.

- Remote users with valid credentials cannot directly query the cloud for data and decrypt non-interactively. All requests must be routed via the organization's ACS.

Another approach is to use ABE in conjunction with HE. In more detail, a sender, Alice, generates a fresh public key and private key $(\mathsf{pk}, \mathsf{sk})$ for the HE scheme. Then she encrypts her data using the HE scheme with the public key pk; call this ciphertext c_{HE}. She then encrypts sk using the ABE scheme with her chosen attribute; call this ciphertext c_{ABE}. Finally she sends the ciphertext $(c_{\mathsf{HE}}, c_{\mathsf{ABE}})$. This solution is adequate if there is only *one* sender who contributes data to a homomorphic computation. Another user, Bob, who follows the same steps as Alice will generate a HE ciphertext with a

[†] A non-interactive zero-knowledge (NIZK) proof system [32] is needed so that an encryptor can *bind* an attribute to a ciphertext, and so the evaluator can *bind* to the output ciphertext the attributes associated with the input ciphertexts. This is needed because otherwise a corrupt staff member could change the attribute associated with a ciphertext to one that satisfied his access policy.

different public key. As such his ciphertext cannot be evaluated together with Alice's ciphertext. Since our goal is to support evaluation with data contributed by multiple independent senders, who may not even be aware of each other, this approach does not offer a solution.

This motivates an alternative solution where a homomorphic encryption scheme *natively* offers access control. Such a scheme accommodates multiple independent senders without requiring any interaction. Another salient feature is that a sender and an evaluator can non-interactively perform their tasks with the public parameters alone. Additionally, a decryptor need only interact with the TA *once* to obtain a secret key for a particular access policy. He can thereafter decrypt an unbounded number of ciphertexts whose attributes satisfy this access policy, without further interaction. Such a scheme is an instance of attribute based homomorphic encryption (ABHE), which comes in two primary flavors - attribute based group homomorphic encryption (ABGHE) and attribute based fully homomorphic encryption (ABFHE). Like their public-key counterparts, ABGHE informally means that a single (group) operation can be performed on the data, whereas ABFHE means that any computation can be performed.

We say an ABHE scheme is *multi-attribute* if it allows evaluation on ciphertexts with different attributes. In contrast, a *single-attribute* scheme only allows evaluation on ciphertexts with the same attribute. ABHE is discussed in more detail in Section 1.3.

Remark One of the assumptions we make is that the evaluator is *semi-honest*. This means that it is assumed to correctly follow the protocol but it may attempt to learn as much information as possible about the data that is encrypted by examining all protocol communication - this adversarial model is also known as "honest-but-curious". This adversarial model can be justified if the stakes are too high for the evaluator to be caught cheating [27] by being *malicious*. A malicious adversary can act arbitrarily i.e. engage in an active attack by deviating from the protocol. So in our case, it might evaluate a different function to the one that is desired, or manipulate the inputs/outputs. Verifying

that the evaluator performs the desired computation exactly as prescribed by the protocol is beyond the scope of this thesis, although there has been much research along this line, most particularly is the primitive Verifiable Computing [89] and protocols based on succinct non-interactive arguments of knowledge (SNARKs) [28]. Such approaches can be used in tandem with our schemes to achieve verifiability.

1.2 Motivating Scenarios

In order to facilitate ready access to, and build understanding of, the concepts, techniques and contributions within this thesis, a selection of use-case scenarios will be developed and extended throughout the work.

1.2.0.1 Computations on Medical Records

Consider a hospital H that avails of the computational facilities of a cloud provider E. Data protection legislation requires the hospital encrypt all sensitive data stored on third party servers. The hospital deploys attribute-based encryption to manage access to potentially sensitive data. Therefore it manages a "trusted authority" that issues secret keys for access policies to staff in accordance with their roles / credentials. Beyond deploying standard attribute-based encryption, H elects to adopt multi-attribute ABFHE because this allows computation to be performed on encrypted data stored at a third party facility such as E.

Parties such as outside researchers, medical practitioners and internal staff in H are able to encrypt sensitive data with appropriate attributes in order to limit access to authorized staff. For example, a doctor in the cardiology unit might encrypt medical data with the attribute "CARDIOLOGY" and a researcher in the maternity unit might encrypt his data with the attribute "MATERNITY". Suppose both encrypted data sets are sent to the cloud provider E to carry out computational processing on the data (while remaining encrypted). A multi-attribute ABFHE allows E to perform the desired

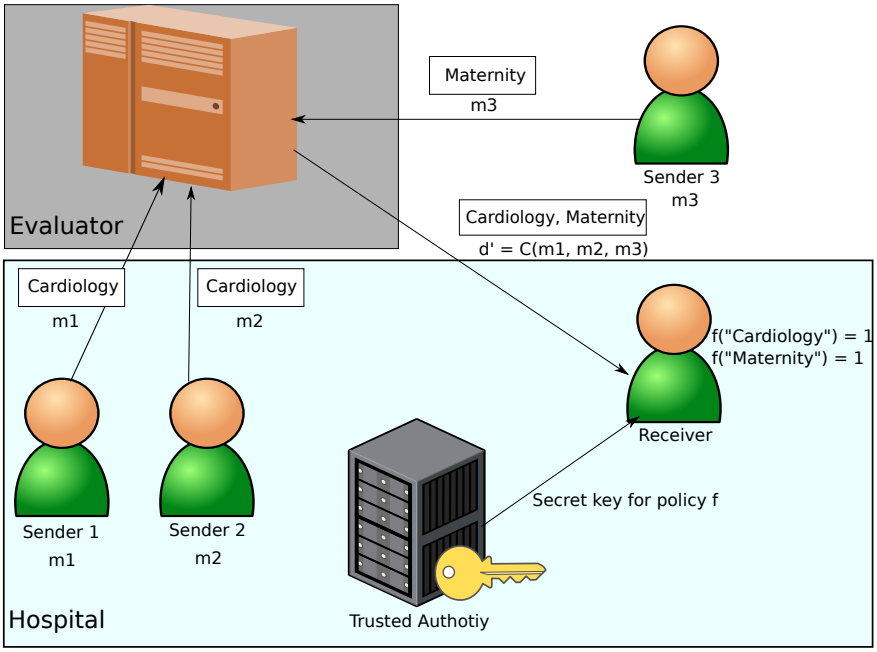

Fig. 1.1: Diagram of Medical Records Scenario

computation homomorphically on both data sets together irrespective of the fact that the data sets were encrypted with different attributes.

Suppose a doctor in the cardiology unit, which we call Sender 1, encrypts her data set of patient records m_1 with attribute "CARDIOLOGY"; we call her ciphertext(s) $E(m_1)$. Likewise a researcher in the maternity, which we call Sender 2, encrypts his data m_2 with attribute "MATERNITY"; we call his ciphertext(s) $E(m_2)$. Furthermore, suppose a research collaborator from outside the hospital, which we call Sender 3, sends encrypted cardiology data to E i.e. she encrypts her data m_3 under attribute "CARDIOLOGY"; we call her ciphertext(s) $E(m_3)$. Figure 1.1 illustrates this scenario.

Let C be the desired computation that needs to be performed on the data sets m_1, m_2 and m_3. For example: C involves calculation of the number of heart attacks experienced by mothers within 3 months of birth. Let $m' = C(m_1, m_2, m_3)$ denote the result of this computation. The goal in question is to offload this computation C to the evaluator E

while retaining input and output privacy of the data sets. Figure 1.1 shows that the evaluator obtains the result m' in encrypted form, as desired. Furthermore, the figure shows that an entity, which we call the "receiver", obtains the result in encrypted form, denoted by $E(m')$. The figure also shows that $E(m')$ is associated with both attributes "CARDIOLOGY" and "MATERNITY", as expected. The receiver's policy f, for which a secret key is obtained from the trusted authority, is satisfied by both attributes. An example of such an f is as follows:

$$f(x) \triangleq x = \text{“MATERNITY”} \text{ OR } x = \text{“CARDIOLOGY”}.$$

It follows that the receiver is able to decrypt the result of the computation. This matches our intuition because her policy permits her access to both the data sets used in the computation. However, a member of staff whose access policy permits access to either "MATERNITY" or "CARDIOLOGY" (but not both) should not be able to decrypt the result.

1.2.0.2 Aggregation in Wireless Sensor Networks

There have been numerous approaches in recent years to apply IBE to Wireless Sensor Networks (WSNs). Notable contributions in this regard include [134, 150, 151, 172]. One prevalent paradigm of a WSN involves a source node that collects sensor measurements in some environment, and forwards these measurements along an established route to a base station. Security becomes an issue in a hostile environment where malicious nodes may intercept the transmitted data. Since the autonomous sensor nodes are heavily resource-constrained, it is imperative to conserve energy where possible to prolong the lifetime and effectiveness of the network.

IBE is a natural choice for this application because nodes deployed in the field neither have to store sensitive secret keys (for symmetric encryption) nor expensively fetch, store and validate public keys for particular base stations (traditional PKI). Instead, since all nodes are identified with a unique network address, it is possible to establish well-defined

identity strings. In addition, all nodes can be pre-loaded with the public parameters of the IBE scheme prior to deployment. Accordingly, in order for a node to transmit to a particular base station B with address a_B, it can derive the public key for B from a_B and the public parameters.

The most costly activity for nodes in a WSN is radio usage. Thus, it is essential to minimize the number of transmissions necessary to accomplish the network's goals. As such, a widely-used optimization strategy is aggregation of data along the path from the source to the sink (the base station). There may be a multitude of sources transmitting independent data along a particular path towards a sink. An intermediate node on the path acting as a relay, or router, may coalesce a collection of data it receives from multiple sources by performing some applicable aggregation function. An example would be to take the mean of the incoming measurements, and forward this mean to the base station. But how can this be accomplished if the data emerging from the sources is encrypted with the identity (i.e. network address in this case) of the ultimate destination, namely that of a base station? A solution to this problem is identity-based homomorphic encryption. To calculate the mean, an additive homomorphism is sufficient. If the aggregation function is more complex, then we need a scheme that can homomorphically evaluate more complex functions. However since the nodes are heavily constrained, in practice, one would expect the aggregation function to not be very complex; hence an additive homomorphism might well suffice.

While identity-based homomorphic encryption is advantageous to WSNs, even greater flexibility is afforded in terms of more fine-grained access control if attribute-based homomorphic encryption (ABHE) is employed. Consider the following scenario. A WSN is deployed in an area in which sensors measure moisture and temperature. The area is divided up into N regions, labeled $R_1, \ldots, R_N$. Each of these regions contains one or more base stations. Suppose it is sufficient for the base stations to determine the aggregate moisture and/or aggregate temperature measured in their region. Note that the mean can be easily derived from these quantities if needed, since we assume the total

number of measurements is also reported. Furthermore, we assume sensor nodes have the capability (such as via GPS) to determine which region they are in. To cut down on communication, aggregator nodes are employed to aggregate reported measurements that are sent by the sensor nodes as they are transmitted en-route to a base station. To minimize data exposure in the presence of adversarial nodes, an ABHE scheme is deployed within the WSN. The ABHE scheme supports an additive homomorphism to satisfy the needs of aggregation as described. Every node, prior to its deployment, is pre-loaded with the public parameters of the scheme. The WSN administrator operates the TA offline, unconnected to the WSN.

A *plaintext* in the system is an integer from the set $\mathcal{P} \triangleq \{0, \ldots, M\}$; sensor readings are assumed to take on values in the range $0, \ldots, M$ for some M. An *attribute* in the system is of the form $(\mathsf{type}, \mathsf{region})$ where $\mathsf{type} \in \{\mathsf{MOISTURE}, \mathsf{TEMPERATURE}\}$ and $\mathsf{region} \in \{R_1, \ldots, R_N\}$. Let $\mathbb{A}$ be the set of attributes. Let $\mathbb{F}$ be a class of access policies modeled as predicates (i.e. Boolean-valued functions), where every policy $f : \mathbb{A} \to \{0, 1\} \in \mathbb{F}$ maps an attribute to $\{0, 1\}$ (denoting false and true respectively).

Adhering to the principle of least privilege, a base station B in region R_1, whose purpose is to monitor moisture content in that region, is issued a secret key for the following policy, denoted f:

$$f(a := (\mathsf{type}, \mathsf{region})) \triangleq (\mathsf{type} = \mathsf{MOISTURE}) \wedge (\mathsf{region} = R_1).$$

Another base station B' whose purpose is to monitor both moisture and temperature in the regions R_1 and R_2 is issued a secret key for the following policy, denoted f':

$$f'(a := (\mathsf{type}, \mathsf{region})) \triangleq (\mathsf{type} = \mathsf{MOISTURE} \ \vee \ \mathsf{type} = \mathsf{TEMPERATURE}) \\ \wedge \ (\mathsf{region} = R_1 \ \vee \ \mathsf{region} = R_2).$$

Suppose an aggregator node near B' receives encrypted readings from two different sensor nodes. The first reading originated in R_1 and has the attribute $a_1 := (\mathsf{type} := \mathsf{MOISTURE}, \mathsf{region} := R_1)$ while the second reading originated in R_2 and has the attribute $a_2 := (\mathsf{type} := \mathsf{MOISTURE}, \mathsf{region} := R_2)$. If the ABHE scheme is *multi-attribute*,

then the aggregator can add the two encrypted readings homomorphically irrespective of the fact that they have *different* attributes. Suppose it subsequently forwards the encrypted result to B'. Intuitively, B' should be able to recover the plaintext because its policy f' authorizes both attributes; that is, we have $f'(a_1) = f'(a_2) = 1$. In contrast, if the base station B gets hold of the ciphertext, it should not be able to recover the plaintext because its policy f is satisfied by only one of the attributes, namely a_1. On the other hand, if the scheme is *single-attribute*, then homomorphic addition is only supported on ciphertexts with the *same* attribute. So in the context of our scenario, this would mean that readings from different regions cannot be aggregated together. This limitation is probably more pronounced in contexts with more extensive and varied attribute families.

1.2.0.3 Participatory Sensing

In participatory sensing, users with personal mobile devices, such as phones that are equipped with sensors, share data acquired from these sensors with a network. We refer to these entities as mobile nodes. Other entities, called queriers, subscribe to receive certain types of data.

De Cristofaro and Soriente [71, 72, 76] presented a model for participatory sensing with privacy-enhanced capabilities using provably-secure cryptographic primitives. Their model, called PEPSI, is described as follows. In PEPSI, the main entities are (1). *mobile nodes* that sense and produce data such as noise level, temperature etc; (2). *queriers* that consume data that is sensed - such nodes subscribe to receive certain types of data matching particular descriptions; and (3). a *service provider* (SP) that acts as an intermediate between mobile nodes and queriers, passing data received from the former on to the latter in accordance with their subscriptions. In addition, PEPSI involves a trusted authority, called a registration authority (RA) whose purpose is node registration - authenticating mobile nodes and queriers and granting them appropriate credentials.

Günther et al. [114] improved the security of PEPSI by making it resistant to collusion between mobile nodes and queriers. Their modified model, called PEPSIco, achieves data privacy, query privacy and node unlinkability (not being able to link two reports sent by the same mobile node). We refer to [114] for more detailed discussion on PEPSI and PEPSIco. An interesting feature that Günther et al. incorporate in PEPSIco is support for data aggregation, which they argue is useful to reduce the amount of information to be sent to queriers, cutting down on communication cost. Günther et al. give a realization of PEPSIco with data aggregation based on additively homomorphic IBE. This highlights the utility of homomorphic IBE - the more homomorphic capacity a scheme has, the more complex aggregation can be performed, thus saving on bandwidth in a participatory sensing context. In Chapter 4, we discuss additively homomorphic IBE in more detail.

The participatory-sensing schemes in [72, 76] and [114] rely on an IBE scheme $\mathcal{E}$. As shown in [114], data aggregation can be added to the system if $\mathcal{E}$ is homomorphic. So this provides us with a practical application of identity-based homomorphic encryption. Günther et al. [114] only consider an additive homomorphism, but clearly, if $\mathcal{E}$ were homomorphic for more complex functions, then more complex aggregation could be performed. While outside the scope of this thesis, an interesting goal for future work is to improve the capabilities of the PEPSIco model by considering attribute-based encryption instead. In such a context, one would require attribute-based homomorphic encryption to do data aggregation as before.

1.2.0.4 Personal Data Management

Another way of looking at attribute-based cryptography is by considering an individual as the trusted authority. Therefore, the public parameters correspond to a user's public key. Alice can grant access privileges to various parties by asserting their credentials and issuing a secret key for an appropriate access policy. At the present time, data is frequently associated with "tags" that serve to describe data. Popular sites such as

the micro-blogging site Twitter [6] encourage users to mark their posts with descriptive tags (known as "hashtags" in the context of Twitter). The applicability of ABE in this context has been investigated [73].

Suppose Alice generates public parameters PP_A along with master secret key MSK_A for an attribute-based homomorphic encryption (ABHE) scheme $\mathcal{E}$. She publishes PP_A as her public key. Suppose Alice works as a journalist with the newspaper "P" and sometimes her work overlaps with her other interests. Alice primarily uses two independent "cloud-based" sites including a "social networking" site SN and a "professional networking" site PN. An attribute in the ABHE scheme $\mathcal{E}$ is modelled as a set A of subattributes. She issues SN with a secret key for the following policy:

$$f_{\mathsf{SN}}(A) \triangleq ((\mathsf{social} \in A) \vee (\mathsf{writing} \in A)) \wedge (\text{“P”} \notin A).$$

The policy f_{SN} allows SN access to messages tagged with "social" or messages tagged with "writing" provided the subattribute "P" is not present.

Similarly Alice grants PN a policy that is inversely related to f_{SN} i.e. it is defined as

$$f_{\mathsf{PN}}(A) \triangleq ((\text{“P”} \in A) \vee (\mathsf{writing} \in A)) \wedge (\mathsf{social} \notin A).$$

Suppose Bob posts an encrypted message to SN, whose plaintext is denoted by M_B, tagged with both writing and "P". Furthermore, the message is encrypted with the attribute $A_B := \{\text{"writing"}, \text{"P"}\}$. Carol also posts an encrypted message, whose plaintext is denoted by M_C, to SN that is encrypted under the attribute $A_C := \{\text{"writing"}, \text{"music"}\}$. One of the facilities provided by SN is the capability to calculate the statistical closeness of the two messages. Let us call this function R. Alice runs a thin client with limited computing power and would like to obtain the value $R(M_B, M_C)$. Note that SN cannot decrypt either Bob or Carol's ciphertext, but it can perform the computation on both homomorphically, and return the encrypted result to Alice. Observe that this can be achieved if $\mathcal{E}$ has the homomorphic capacity to evaluate R. Furthermore, $\mathcal{E}$ must be multi-attribute i.e it must support evaluation on *distinct* attributes. A decryptor must

have an access policy that is satisfied by both A_B and A_C. An example of such an access policy is the following:

$$f_T(A) \triangleq (\mathsf{writing} \in A) \vee (\mathsf{music} \in A).$$

Imagine that Alice entrusts a third party T with any content tagged with writing or tagged with music. For example, T might be a device within Alice's personal area network such as a laptop. Alice equips T with a secret key for f_T. As a result, T can decrypt Bob's ciphertext, Carol's ciphertext and the result of the homomorphic computation of R on both ciphertexts.

As the scenario above illustrates, ABHE enables privacy-preserving outsourcing of computation in addition to the fine-grained access control offered by ABE. Alice, who is the TA in this scenario, chooses what types of data various services get access to. As we have seen, even if a service cannot decrypt encrypted data, it can still act on it (homomorphically) and return the result (in encrypted form).

1.3 Research Question and Contributions

Before examining attribute-based homomorphic encryption, we need to say a little more about what constitutes a homomorphic encryption (HE) scheme and what constitutes an attribute based encryption (ABE) scheme. Our intention is to keep the discussion informal for the moment; we will elaborate later on the precise definitions of these primitives and their related security notions.

A HE scheme can perform computation over its ciphertexts. The generalized definition of HE given by Gentry [92] uses the circuit[‡] model of computation to represent such computations. Furthermore Gentry's definition requires compact evaluation of a circuit over the ciphertexts; that is, the size of the resulting ciphertext is independent

[‡]A circuit with n inputs over some domain $\mathcal{P}$ is a directed acyclic graph in which every node is either one of the n input values, or an operation (known as a *gate*) from some finite set of gates, each of which maps a number of elements of $\mathcal{P}$ to another element of $\mathcal{P}$.

of the size of the circuit (i.e. the number of gates in the circuit). We distinguish HE schemes based on the class of circuits they support. Two primary subclasses of HE are group homomorphic encryption (GHE) and fully homomorphic encryption (FHE). The former allows a group operation such as modular addition or modular multiplication to be compactly evaluated and this operation can be applied an unbounded number of times. The algebraic structure of a group gives rise to some interesting properties, and as such, GHE is often used as a building block in protocols. Furthermore, there are applications where a single operation such as addition or XOR is sufficient.

On the other hand, FHE facilitates arbitrary computation on the ciphertexts i.e. any computable function can be compactly evaluated homomorphically. In other words, FHE supports a class of circuits with a universal set of gates (such as $\{\text{AND}, \text{XOR}\}$ in the Boolean case), which is known to be Turing-equivalent. The idea of FHE was first proposed in 1978 by Rivest, Adleman and Dertouzos [162]. Considered the "holy grail" of cryptography [140, 173], many in the cryptography research community believed it to be impossible [140]. In 2009, Gentry [93] presented the first FHE scheme, triggering a burst of research into the topic.

Where the "power" of a HE scheme is given by the complexity of its supported class of circuits, the "power" of an ABE scheme is given by the complexity of its supported class of access policies. There has been much research into achieving ABE with expressive access policies. The simplest meaningful class of access policies corresponds to equality checking; that is, for every attribute a, there is an access policy f_a that is satisfied by a and only a. This corresponds to the special case of IBE. When pursuing research goals in the attribute-based setting, especially feasibility results, the special case of IBE is often the first "port of call" because it represents the simplest meaningful class of access policies.

An important question about HE in the attribute-based setting is whether there is support for homomorphic evaluation over ciphertexts with different attributes. An evaluation may involve *composition* with different attributes; we term the number of

such attributes the *degree of composition*. We say an ABHE scheme is *multi-attribute* if it allows evaluation on ciphertexts with different attributes. A parameter representing the number of distinct attributes (i.e. degree of composition) to tolerate is specified in advance of generating the public parameters. In contrast, a *single-attribute* scheme only allows evaluation on ciphertexts with the same attribute; in other words, the degree of composition is 1. Putting all the pieces together, we can characterize *attribute-based homomorphic encryption* (ABHE) by three principal facets: (1). class of circuits; (2). class of access policies; and (3). composition (i.e. single-attribute vs. multi-attribute). One of the fundamental results of this thesis is the following:

- (Informal) Under reasonable cryptographic assumptions, there exists a secure *multi-attribute* Attribute-Based Fully Homomorphic Encryption (ABFHE) scheme supporting all polynomial-time access policies.

This serves as a feasibility result. Furthermore, our proof is constructive - we give a construction of both a single-attribute and multi-attribute ABFHE. This is a surprising result because the techniques used to achieve FHE in the public-key setting appeared to be incompatible with the attribute-based setting. It also solves an open problem first mentioned by Naccache at his talk at CHES/Crypto 2010 [146], namely "identity-based fully homomorphic encryption", which follows as a corollary of our result. However, our constructions underlying this feasibility result are far from practical, and rely on the machinery of indistinguishability obfuscation. For the moment it is sufficient for the reader to understand that indistinguishability obfuscation is a cryptographic primitive for which candidate constructions [87,96,155] of have been recently proposed, but which is computationally expensive. In particular, its use in our construction is highly computationally expensive. This renders our feasibility results impractical at the present time. This stimulates inquiry into more "concrete" and efficient constructions, and it prompts some natural research questions, which we will now elaborate on.

For many applications, the full capabilities of multi-attribute ABFHE are not needed.

In terms of functionality, relaxations can be made on all three facets above: supported circuits, supported access policies and supported composition. Can we achieve more efficient constructions by making one or more such relaxations? To put this into perspective, let us revisit the primary subclasses of HE, namely GHE and FHE.

GHE has been applied in many applications (see Section 2.1.1.1, Section 2.3.0.1 and the surveys [83, 173]). However, GHE has not been explored in the attribute-based setting, to the best of our knowledge. So our first avenue is to investigate *attribute based group homomorphic encryption* (ABGHE). Additive (group) homomorphisms are particularly useful for real-world applications. However there are no known attribute-based additively[§] homomorphic schemes. One of the aims of this thesis is to formulate the notion of attribute-based group homomorphic encryption (ABGHE) and construct an attribute-based additively homomorphic scheme with expressive access policies. We present a formal syntax for ABGHE and succeed in constructing an identity-based additively-homomorphic scheme for finite modular groups of *small* order m. Specifically m is required to be polynomially sized (in the security parameter) since the ciphertext size grows with m. The special case of $m = 2$ gives an XOR homomorphism, which is valuable for many applications (see Section 2.3.0.1). Achieving an additive homomorphism for small m with more compact ciphertexts remains an open problem as does achieving an additively homomorphic scheme for large m (i.e. superpolynomial) as we have in the public-key setting [154]. Furthermore, our result is limited to the identity-based case; it is open to construct such a scheme for more complex access policies.

Now let us refocus our attention on FHE, and we ask, can we construct more efficient attribute-based FHE (ABFHE) schemes than our feasibility result? Recall that the latter relied on the computationally expensive machinery of indistinguishability obfuscation, but as noted above, this was found to be the only technique thus far that can overcome

[§]We mean "additively homomorphic" in the "classical" sense of a group homomorphic scheme, which allows an unbounded number of additions. This precludes schemes that allow only a limited number of additions.

the technical obstacles in the way of realizing "pure" ABFHE. By "pure" we mean that all circuits can be evaluated. This qualifier stems from the existence of an important relaxation of FHE proposed by Gentry [93] called leveled FHE. Leveled FHE supports evaluation of circuits of depth at most L, where L is a parameter that is specified in advance of generating the public parameters of the scheme. This is sufficient for many applications because if one knows the maximum depth of circuits that require evaluation, the public parameters can be generated to accommodate this. At Crypto 2013, Gentry, Sahai and Waters (GSW) presented an identity-based and an attribute-based leveled FHE scheme. Their attribute-based construction supports general-purpose access policies, which are represented as Boolean circuits. The GSW schemes are also notable because their security is based on the Learning with Errors (LWE) problem, a problem introduced by Regev [161] that has received considerable attention in cryptography due to a known worst-case reduction to a hard lattice problem. However, a limitation of the GSW identity-based and attribute-based schemes is that they are single-identity and single-attribute respectively. Therefore, an enticing goal is to construct a multi-attribute leveled ABFHE from LWE. In this thesis, we move a step closer to this target by constructing a multi-identity leveled IBFHE (i.e. a special case of multi-attribute leveled ABFHE with a simple class of access policies). We also highlight difficulties extending our result to multi-attribute ABFHE with more complex access policies.

Another contribution of this thesis is as follows. Suppose we have a scheme that can evaluate "shallow" circuits. More precisely, suppose this scheme can evaluate circuits of polylogarithmic depth in some parameter N. Then we present a construction, which uses the aforementioned scheme, that can evaluate circuits with at most N inputs, but of unbounded depth. If N is large, then one would expect our construction to meet all practical expectations because the number of inputs needed would not not typically (if at all) exceed N. This means we can shift our goal to finding schemes that can evaluate "shallow" circuits, which is an easier task. Leveled ABFHE meets this goal. As we have seen, we give a construction of multi-identity leveled IBFHE, which can be used to

instantiate this result.

The main contributions of this thesis are summarized as follows. The first three contributions are categorized as "concrete" constructions; in other words, they are more conducive to practical realizations. The final contribution is our aforementioned feasibility result, which at the present time is highly impractical.

- Additively Homomorphic Identity Based Encryption scheme for modular groups of small order.
- Black-box construction of ABFHE with bounded arity (number of inputs) from a leveled ABFHE scheme.
- Multi-Identity Leveled IBFHE.
- Feasibility result: Single-Attribute and Multi-Attribute ABFHE for all polynomial-time access policies.

1.4 Roadmap

In Chapter 2, the state of the art in homomorphic encryption, attribute based encryption and their intersection is investigated.

In Chapter 3, we formally define attribute based homomorphic encryption and establish the security definitions used throughout the thesis.

In Chapter 4, we explore attribute based group homomorphic encryption, and present a construction of an additively homomorphic IBE scheme.

In Chapter 5, our attention turns to attribute based fully homomorphic encryption (ABFHE). In the chapter, a black-box construction is given of an ABFHE scheme that can evaluate circuits with a bounded number of inputs.

In Chapter 6, we present a multi-identity leveled identity-based fully homomorphic encryption (IBFHE) scheme. While being important in its own right for use as a standalone construction, it also can be used to instantiate the result in the previous chapter.

In Chapter 7, we present feasibility results of single-attribute and multi-attribute ABFHE for all polynomial-time access policies, which completes the contributions of this thesis.

Finally in Chapter 8, we present the conclusions of our work in addition to future work.

Chapter 2

State of the Art

As shown in Figure 2.1, this chapter presents the state of the art in homomorphic encryption, identity/attribute based encryption and finally, the intersection between the two areas. We start with homomorphic encryption, then discuss identity and attribute based encryption, and conclude with a discussion of the overlap between these areas with a focus on where the contributions of this thesis fit in.

Remark Another work that considers access control in outsourced computation is by Alderman et al. [16]. Their work allows public verification and supports policies over senders and verifiers. The main difference with our approach is that we use homomorphic encryption where no restriction is placed on the computation that can be performed, for example the computation might be chosen at any time by any party. However, we do not make provisions for verifiablity of the computation (i.e. ensuring that it was carried out correctly). As mentioned in the introduction (see Section 1.1.1), verifiability is outside the scope of this thesis, but there are approaches that can work in tandem with our schemes to achieve verifiability.

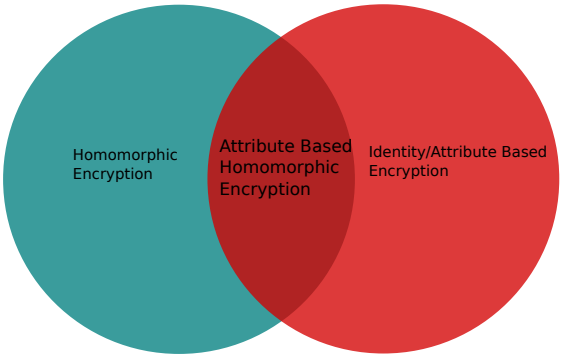

Fig. 2.1: Venn diagram of the intersection of thesis themes.

2.1 Homomorphic Encryption

Homomorphic encryption is the ability to operate on data while it remains encrypted. In a nutshell, this means the ability to perform operations on the ciphertexts that result in corresponding operations being performed on the underlying plaintexts. In other words, given an encryption $E(m_1)$ of value m_1 and an encryption $E(m_2)$ of value m_2, it is possible to obtain an encryption $E(f(m_1, m_2))$ of some function f of m_1 and m_2, without access to the secret key. Mathematically, this can be viewed as a homomorphism* between the ciphertext space and plaintext space.

Homomorphic encryption traces its roots to a paper by Rivest, Adleman and Dertouzos [162] from 1978, shortly after the emergence of Public Key Cryptography. The goal in their paper is to delegate computations to an untrusted server without the server learning the values of the inputs (*input privacy*) or the values of the outputs (*output privacy*). As a means to solving this problem, they proposed the notion of a *privacy homomorphism*, a deterministic encryption scheme that allows operations to be performed on encrypted data without access to the secret key. They proposed a number of candidate schemes, all of which were later broken [53]. An example of a multiplicative

*See the Glossary (G3).

privacy homomorphism is textbook RSA [163] [†]. An algebraic privacy homomorphism is one that supports both addition and multiplication in a ring[‡], such as $\mathbb{Z}_N$, where N is a product of distinct primes.

One of the issues with privacy homomorphisms is that they are deterministic, and hence not semantically secure[§]. Semantic security had not yet been advanced in the literature when [162] was published. It was formalized shortly thereafter in the seminal work of Goldwasser and Micali [107]. Indeed the modern notion of homomorphic encryption is understood to be probabilistic. Beginning with Goldwasser and Micali's pioneering work, provable security became a mainstay of the cryptography community, and achieving semantic security for a public-key encryption scheme became a minimum requirement. Consequently, we will confine ourselves to homomorphic encryption schemes that have been proven semantically secure under a well-defined computational assumption.

To illustrate the concept of homomorphic encryption (HE), we give an example of a symmetric HE scheme i.e. a scheme where the encryption and decryption keys are the same. We denote by $\mathcal{P}$ the set of plaintexts and $\mathcal{C}_K$ the set of valid ciphertexts under some key K. Let $D_K : \mathcal{C}_K \to \mathcal{P}$ denote the decryption function with key K. Let $\oplus$ be some operation on the plaintexts, and let $\boxplus$ be some operation on the ciphertexts. Then for any $c_1, c_2 \in \mathcal{C}_K$, we have that $D_K(c_1 \boxplus c_2) = D_K(c_1) \oplus D_K(c_2)$. Technically, the decryption function $D_K \colon \mathcal{C}_K \to \mathcal{P}$ is a homomorphism.

An important distinction is to be made between *fully homomorphic encryption* (FHE) and *partially homomorphic encryption* (PHE). The former informally means that any computable function can be performed on the encrypted data, whereas the latter facilitates certain functions to be evaluated, but not all.

Typically, at least in all known constructions, FHE is achieved by preserving the

[†]Textbook RSA is not semantically secure since it is deterministic. Its probabilistic variants Optimal Asymmetric Encryption Padding (OAEP) [26] and OAEP+ [169] lose the homomorphic property.

[‡]See the Glossary (G2).

[§]See the Glossary (G6).

ring structure of the plaintext space. As a result, both multiplication and addition are supported, which allows arbitrary arithmetic circuits to be evaluated. In contrast, PHE generally preserves the group[¶] structure of the plaintext space under one operation, although there are some notable exceptions which we will see shortly.

2.1.1 Partially Homomorphic Encryption

Before Gentry's breakthrough in 2009 [93], all homomorphic public-key cryptosystems were partially homomorphic. For example RSA can be viewed as homomorphic for modular multiplication [163], albeit it is not semantically secure because encryption is not probabilistic.

2.1.1.1 Group Homomorphisms

An important subclass of PHE is the class of public-key encryption schemes that admit a group homomorphism between their ciphertext space and plaintext space. This class corresponds to what is considered "classical" HE [18], where a single group operation is supported, most usually addition. Gjøsteen [101] examined the abstract structure of these cryptosystems in terms of groups, and characterized their security as relying on the hardness of a subgroup membership problem. Armknecht, Katzenbeisser and Peter [18] rigorously formalized the notion, and called it *group homomorphic encryption* (GHE). Before discussing well-known instances, we begin with the formal definition of GHE by Armknecht, Katzenbeisser and Peter [18].

Definition 2.1.1 (Definition 1 in [18])**.** *A public-key encryption scheme* $\mathcal{E} = (G, E, D)$ *is called* group homomorphic*, if for every* $(\mathsf{pk}, \mathsf{sk}) \leftarrow G(1^\lambda)$*, the plaintext space* $\mathcal{P}$ *and the ciphertext space* $\hat{\mathcal{C}}$ *(written in multiplicative notation) are non-trivial groups such that*

[¶]See the Glossary (G1).

- *the set of all encryptions* $\mathcal{C} := \{c \in \hat{\mathcal{C}} \mid c \leftarrow E_{\mathsf{pk}}(m), m \in \mathcal{P}\}$ *is a non-trivial subgroup of* $\hat{\mathcal{C}}$
- *the restricted decryption* $D^*_{\mathsf{sk}} := D_{\mathsf{sk}|\mathcal{C}}$ *is a group epimorphism (surjective homomorphism) i.e.*

$$D^*_{\mathsf{sk}} \text{ is surjective and } \forall c, c' \in \mathcal{C} \ : \ D_{\mathsf{sk}}(c \cdot c') = D_{\mathsf{sk}}(c) \cdot D_{\mathsf{sk}}(c')$$

- sk *contains an efficient* decision function $\delta : \hat{\mathcal{C}} \to \{0, 1\}$ *such that*

$$\delta(c) = 1 \iff c \in \mathcal{C}$$

- *the decryption on* $\hat{\mathcal{C}} \setminus \mathcal{C}$ *returns the symbol* $\perp$.

Goldwasser and Micali [107] constructed the first GHE scheme, which was also the first homomorphic scheme to be semantically secure. The Goldwasser-Micali (GM) cryptosystem supports addition modulo 2 i.e. the XOR operation. Breaking the semantic security of GM is equivalent to solving the quadratic residuosity problem, a well-established problem in number theory that is believed to be intractable.

One of the disadvantages of GM is that it can only encrypt a single bit at a time. A single bit of plaintext is encrypted as an element of $\mathbb{Z}_N$, where N is an RSA modulus; that is, a product of two large primes. The ciphertext expansion is therefore $\lg N$. Subsequent works generalized GM with a view to obtaining a more efficient scheme in terms of expansion, but also supporting an additive homomorphism in a larger plaintext space [83].

Benaloh [29] generalized GM to handle plaintexts in the range $\{0, \ldots, k-1\}$ where k is a prime. Thus, Benaloh is homomorphic for the additive group $(\mathbb{Z}_k, +)$. The expansion is then reduced to $\lg N / \lg k$. The cost of decryption (without precomputation) is estimated to be $O(\sqrt{k} \lg k)$ [83] (a precomputation step with the same cost can be carried out to speed up decryption). This cost places a limit on how large k can be; clearly, it must be polynomial in the security parameter[‖].

[‖] See the Glossary (G5).

Naccache and Stern [147] generalized Benaloh and obtained a cryptosystem with reduced decryption time and accommodation of large k, which can be superpolynomial. Under a particular choice of parameters, ciphertext expansion can be reduced to as low as 4 times the plaintext size.

Starting with GM, all of the schemes thus far have operated in $\mathbb{Z}_N$ where $N = pq$ with p and q being two large distinct primes. Okamoto and Uchiyama [149] instead explored the possibility of $N = p^2q$, and obtained a scheme with $k = p$ and a ciphertext expansion of 3 times the plaintext size.

Building on the Okamoto and Uchiyama construction, Paillier [154] proposed an improved scheme with a ciphertext expansion factor of 2, and low encryption and decryption cost. Paillier's scheme is one of the most efficient and widely-used additively homomorphic cryptosystems. As a result of its practicality, it is well-suited to applications such as electronic voting [74].

Damgård and Jurik [74] generalized Paillier to higher-order groups $\mathbb{Z}_{N^{s+1}}$ with $s > 0$. Paillier corresponds to the special case of $s = 1$. Damgård-Jurik allows plaintexts to lie in the set $\mathbb{Z}_{N^s}$, and as a result, expansion is lowered to $1 + 1/s$ at the expense of increased encryption and decryption cost. Damgård-Jurik has been adapted by [85] to the setting of elliptic curves over rings , but the resulting scheme has worse encryption and decryption performance. Despite several advances in the literature, Paillier's original system remains the most efficient additively homomorphic scheme.

The additively homomorphic schemes considered above, GM [107], Benaloh [29], Naccache-Stern [147], Okamoto-Uchiyama [149], Paillier [154] and Damgård-Jurik [74], are all instances of GHE; that is, they all satisfy Definition 2.1.1. Other examples of GHE from the literature include [75,101,102]. Of particular mention is Damgård's variant [75] of ElGamal [79] which influenced the definition of GHE by Armknecht et al. Specifically, the decision function δ was included in the definition to encompass Damgård's ElGamal; in all other cases, the decision function is trivial. Moreover, we remark that ElGamal meets the above definition for its multiplicative homomorphism. Although it is well-

known that ElGamal also supports modular addition [69], its reliance on a brute-force search to recover the result limits the size of the message space, and so it cannot be classified as meeting the definition of GHE with respect to addition.

GHE is the "classical" flavor of homomorphic encryption. It allows unbounded applications of the group operation. Furthermore, a ciphertext can be easily *rerandomized* by multiplying (assuming multiplicative notation) it with an encryption of 1. Doing so after any homomorphic evaluation ensures the resulting ciphertext has the same distribution as a freshly encrypted ciphertext. A scheme with this property is said to be *strongly homomorphic* using the terminology of [103].

2.1.1.2 "Bounded Homomorphisms"

Unlike GHE, there are schemes that do not allow an unbounded number of homomorphic operations to be performed. In such schemes, ciphertexts emerging from evaluation have a different distribution to fresh encryptions. Inherent in the ciphertexts is an implicit "noise" that grows as homomorphic evaluation is carried out [92]. Decryption is only guaranteed to be successful when the noise is below a certain threshold. Overflowing this threshold results in an undefined outcome. This type of homomorphism has been referred to as a "bounded homomorphism" [157] or a "pseudohomomorphism" [125, 137, 138]. Linear codes and lattices facilitate construction of this type of cryptosystem; additively homomorphic examples include [19, 105, 125, 137, 138, 157]. Two of these [19, 138] also allow multiplications, but at the cost of an exponential blowup in ciphertext size.

Gentry departs from using the term "bounded homomorphism" in [92] in favor of adopting a unified definition of HE that captures "bounded homomorphisms" and conventional group homomorphisms, among many others, as special cases. Before presenting his definition and focusing our attention on FHE, we will take a look at some of the schemes that support more than a single algebraic operation.

2.1.1.3 Evaluation of more than one homomorphic operation

As mentioned earlier, the schemes from [19, 138] support addition and multiplication, but the ciphertext size exponentially grows with the multiplicative depth.

Sanders, Young and Yung (SYY) [166] introduced a scheme that enables circuits in the class NC^1 to be evaluated. The class NC^1 is the class of decision problems decided by Boolean circuits with $n^{O(1)}$ gates and depth $O(\log n)$, where n is the number of inputs. The size of the evaluated ciphertexts is exponential in the depth d of the circuit. This explains why circuits in NC^1 are the only ones that can be feasibly evaluated. SYY makes use of a re-randomizable public-key cryptosystem, such as any GHE scheme, to achieve this goal. Suppose Alice encrypts her inputs and sends them to Bob. Bob can choose to "inattentively" evaluate a circuit $C \in \mathrm{NC}^1$ with Alice's inputs, which he remains oblivious to. Let d be the depth of C. Bob's work is dependant on the *size* (i.e. the total number of gates) of C, whereas the work Alice needs to do when performing decryption is dependant only on the depth d.

SYY is also *circuit-private* for the family of uniform Boolean circuits of depth d, which means that Alice learns nothing more about C beyond its depth and any information leaked by the output of the evaluation. Interesting functions in NC^1 include division, powering, addition, Boolean matrix multiplication and the majority function.

Ishai and Paskin [120] put forward a scheme that can evaluate *branching programs* of polynomial size. A branching program is a directed acyclic graph where the internal nodes are labeled with (Boolean) variables; such nodes have left and right children. Evaluation proceeds downwards from the root: at each internal node, one follows the left branch (resp. the right branch) if the value of the node's corresponding variable is 0 (resp. 1). The leaf nodes are labeled with the output of the program for that path, which corresponds to a unique assignment of the inputs. Branching programs are a powerful model of computation. Polynomial-sized branching programs with constant-width can decide exactly the decision problems in NC^1 [23]. Evaluated ciphertexts in

the Ishai-Paskin scheme are smaller than those in SYY [166].

Boneh, Goh and Nissim (BGN) [40] developed an interesting scheme using Bilinear pairings that could evaluate quadratic formulae over a finite field. More precisely. it supports a single multiplication and an arbitrary number of additions. The single multiplication is accomplished using a bilinear pairing, Like ElGamal and Benaloh, message recovery requires the computation of a discrete logarithm (i.e. the value is stored in the exponent), which limits the size of the message space. A well-suited special case is 2-DNF formulae, i.e. formulae in disjunctive normal form (DNF) whose conjunctions have at most 2 variables. BGN has small ciphertexts and is practical. It is useful for dedicated applications requiring dot products or 2-DNF formulae.

2.1.2 General Definition of Homomorphic Encryption

Gentry presented a general definition of HE that encompasses the various types of homomorphisms we have seen so far. In his definition, the homomorphic computations supported by a HE scheme are described by a class of supported circuits. There are other models of computation one could choose instead of circuits, including branching programs, formulas and finite automata, to name just a few. Circuits are as powerful as any other known *representation model*, and almost all the homomorphisms we have seen so far can be viewed naturally in terms of circuits. For example: a scheme that is group-homomorphic for the group $(\mathcal{P}, *)$ can evaluate any arithmetic circuit over $\mathcal{P}$ built with the "gate" $*$. Furthermore, BGN can be viewed as supporting arithmetic circuits built form the gates $\{+, *\}$ with multiplicative depth 1. The syntactic formalism of expressing a given homomorphic computation as a circuit does not preclude the encapsulation of schemes with inherent support for other representation models, e.g: branching programs in the case of Ishai-Paskin [120], because such schemes can internally convert a circuit from a supported family into another representation model prior to evaluation.

Now we consider Gentry's definition in more detail. A homomorphic encryption (HE) scheme $\mathcal{E}$ consists of four algorithms $\mathsf{Gen}, \mathsf{Encrypt}$, $\mathsf{Decrypt}$ and Eval. In addition

$\mathcal{E}$ has plaintext space $\mathcal{P}$, ciphertext space $\mathcal{C}$ and supports a class of circuits $\mathbb{C}$. Gen is a randomized algorithm that takes as input a security parameter λ and generates a public and private key pair $(\mathsf{pk}, \mathsf{sk})$. $\mathsf{Encrypt}$ is a randomized algorithm that takes as input a public key pk and a plaintext $\mu \in \mathcal{P}$, and outputs a ciphertext $\psi \in \mathcal{C}$. $\mathsf{Decrypt}$ is a deterministic algorithm that takes a secret key sk and a ciphertext $\psi \in \mathcal{C}$, and outputs a plaintext $\mu' \in \mathcal{P}$ if ψ is an encryption of μ' under the public key pk; it outputs $\perp$ otherwise. Finally, Eval is a randomized algorithm that takes as input a public key pk and a circuit $C \in \mathbb{C}$ along with ℓ ciphertexts $\psi_1, \ldots, \psi_\ell \in \mathcal{C}$, and outputs a ciphertext $\psi' \in \mathcal{C}$. Assuming that ψ_i encrypts $\mu_i \in \mathcal{P}$ for every $i \in [\ell]$, then ψ' encrypts $C(\mu_1, \ldots, \mu_\ell)$.

The computational complexity of all four algorithms must be polynomial in the security parameter λ. Let us consider a more formal definition of correctness:

Definition 2.1.2 (Correctness of Homomorphic Encryption, Definition 2.1.1 [92])**.** *A homomorphic encryption scheme $\mathcal{E}$ is said to be correct for circuits in $\mathbb{C}$ if, for every key-pair $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda)$, any circuit $C \in \mathbb{C}$, any plaintexts $\mu_1, \ldots, \mu_\ell$, and any ciphertexts $\psi_i \leftarrow \mathsf{Encrypt}(\mathsf{pk}, \mu_i)$ for $i \in [\ell]$, it is the case that:*

$$\textit{if } \psi' \leftarrow \mathsf{Eval}(\mathsf{pk}, C, \psi_1, \ldots, \psi_\ell), \quad \textit{then } \mathsf{Decrypt}(\mathsf{sk}, \psi') \to C(\mu_1, \ldots, \mu_\ell)$$

*except with negligible probability** over the random coins used in* Eval*.*

Correctness as per Definition 2.1.2 still allows trivial constructions of HE, as pointed out in [92]. In more detail, it is sufficient for Eval to simply output the circuit C together with the input ciphertexts $\psi_1, \ldots, \psi_\ell$. The burden is then placed on $\mathsf{Decrypt}$ to evaluate the circuit (it first decrypts the ψ_i). Such a construction trivially satisfies *correctness*.

To exclude such trivial schemes, Gentry introduces an additional condition, called *compactness*, that places an upper limit on the length of the ciphertexts outputted by Eval (we refer to such ciphertexts as *evaluated ciphertexts*). The length of such a

**See the Glossary (G4).

ciphertext is required to be at most some fixed polynomial in λ, and hence independent of the number of inputs ℓ, and in turn, the size of C. Gentry formally captures the requirement by restricting the size of the decryption circuit (i.e. a circuit representation of $\mathsf{Decrypt}$) to be at most some polynomial in λ. This implies that decryption must be independent of the number of inputs ℓ to Eval. It is easy to see that this excludes the trivial construction above. The formal definition employed throughout this thesis is based on limiting the ciphertext size, which we find to be more terse for our purposes.

Definition 2.1.3 (Compactness, Definition 3.4 [49])**.** [††] *A HE scheme $\mathcal{E}$ is said to be compact if there exists a polynomial $s(\cdot)$ such that the output of* Eval *on any input is at most $s(\lambda)$ bits long.*

We inherit the term "compactly evaluates" from Gentry [92] with the meaning that $\mathcal{E}$ "compactly evaluates" circuits in $\mathbb{C}$ if $\mathcal{E}$ is both *correct* (i.e. it satisfies Definition 2.1.2) and *compact* (i.e. it satisfies Definition 2.1.3) for $\mathbb{C}$. We also use this term with respect to classes of functions. In this way, we say that a scheme that is group-homomorphic for $(\mathcal{P}, *)$ compactly evaluates all functions in $\{x_1, \ldots, x_t \mapsto c * x_1 * \cdots * x_t : c \in \mathcal{P}, t \in \mathbb{Z}^+\}$, and BGN compactly evaluates 2-DNF formulae.

2.1.3 Fully Homomorphic Encryption

Let us start with the following simple definition that makes use of the notion "compactly evaluates" introduced earlier.

Definition 2.1.4 (Fully Homomorphic Encryption, Definition 2.1.4, [92])**.** *A scheme $\mathcal{E}$ is said to be fully homomorphic if it compactly evaluates all circuits.*

[††] Compactness, as defined, is a strong requirement, and it excludes some interesting schemes such as Ishai-Paskin [120] whose evaluated ciphertexts are linearly sized in the depth of the circuit. A weaker notion of compactness, referred to as "quasi-compactness" in [92], allows the size of the secret keys and ciphertexts to depend polynomially on the depth of the circuit. However, we work with the stronger notion of compactness in this work.

A fully homomorphic encryption (FHE) scheme can evaluate all polynomial-time computable functions. Strikingly, it achieves this without expanding the ciphertext size. For many applications, we need only the capability to evaluate circuits of some limited depth. Leveled FHE is a relaxation of FHE that can evaluate circuits of depth at most some positive integer d. While the size of the public key may depend polynomially on d, the computational complexity of the decryption algorithm must remain independent of d. A formal definition follows.

Definition 2.1.5 (Leveled Fully Homomorphic Encryption, Definition 2.1.5 [92])**.** *A family of HE schemes* $\{\mathcal{E}^{(d)} : d \in \mathbb{Z}^+\}$ *is said to be leveled fully homomorphic if, for all* $d \in \mathbb{Z}^+$*, they all use the same decryption circuit, and* $\mathcal{E}^{(d)}$ *compactly evaluates all circuits of depth at most* d*. Furthermore, it is required that the computational complexity of* $\mathcal{E}^{(d)}$*'s algorithms* Gen*,* $\mathsf{Encrypt}$ *and* Eval *be polynomial in* λ*,* d *and in the case of* Eval*, the size of the circuit* C*.*

2.1.3.1 Gentry's Bootstrapping Theorem

In [92, 93], Gentry proves a fundamental result about the construction of leveled FHE. Consider a HE scheme $\mathcal{E}$ that compactly evaluates circuits in some class $\mathbb{C}_{\mathcal{E}}$. Without loss of generality, we assume that $\mathcal{E}$ encrypts single bits i.e. $\mathcal{P} = \{0, 1\}$. Let $D_{\mathcal{E}}$ be $\mathcal{E}$'s decryption circuit. Now suppose that $\mathbb{C}_{\mathcal{E}}$ contains $D_{\mathcal{E}}$ and an "augmented" decryption circuit $D'_{\mathcal{E}}$. In more detail, $D'_{\mathcal{E}}$ consists of a universal gate such as a NAND gate whose two input wires connect to two independent copies of $D_{\mathcal{E}}$. Gentry calls a scheme that can compactly evaluate its own augmented decryption circuit *bootstrappable*. In a nutshell, the process that Gentry called "bootstrapping" involves performing decryption on a ciphertext *homomorphically*; the following scenario may help elucidate this concept.

Suppose that when Alice generates her key-pair, she first computes $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathcal{E}.\mathsf{Gen}(1^\lambda)$ and $(\mathsf{pk}', \mathsf{sk}') \leftarrow \mathcal{E}.\mathsf{Gen}(1^\lambda)$. She then encrypts the secret key bits of sk (using $\mathcal{E}$) with the public key pk'; that is, she obtains $\overline{\mathsf{sk}_i} \leftarrow \mathcal{E}.\mathsf{Encrypt}(\mathsf{pk}', \mathsf{sk}_i)$, where sk_i denotes the i-th bit of sk for $i \in [t]$, and t is the bit-length of sk. Alice publishes

$(\mathsf{pk}, \mathsf{pk}', \overline{\mathsf{sk}} := (\overline{\mathsf{sk}_1}, \ldots, \overline{\mathsf{sk}_t}))$ as her public key, and retains sk' as her private key. Let ψ be a ciphertext that encrypts some bit under pk. It is possible to transform ψ into a ciphertext ψ' that encrypts the *same* bit under pk'. To accomplish this, Bob encrypts each bit of $\psi := (\psi_1, \ldots, \psi_k) \in \{0,1\}^k$ under pk'; that is, he computes $\overline{\psi_i} \leftarrow \mathcal{E}.\mathsf{Encrypt}(\mathsf{pk}', \psi_i)$ for $i \in [k]$. Let $\overline{\psi} = (\overline{\psi_1}, \ldots, \overline{\psi_k})$. Accordingly, he can now use $\mathcal{E}.\mathsf{Eval}$ to *homomorphically* evaluate the decryption circuit $D_{\mathcal{E}}$ with encrypted inputs $\overline{\mathsf{sk}}$ and $\overline{\psi}$. More precisely, he computes $\psi' \leftarrow \mathcal{E}.\mathsf{Eval}(\mathsf{pk}', D_{\mathcal{E}}, \overline{\mathsf{sk}}, \overline{\psi})$, which encrypts the same bit as ψ, but under a different public key, namely pk'. Because the "augmented" decryption circuit $D_{\mathcal{E}'} \in \mathbb{C}_{\mathcal{E}}$ is supported by $\mathcal{E}$, an additional NAND gate can be performed on two ciphertexts like ψ'.

Gentry observed that a bootstrappable scheme $\mathcal{E}$ can be used to construct a leveled FHE scheme for any $d \in \mathbb{Z}^+$. Essentially, the above process is naturally extended to generate d key-pairs $(\mathsf{pk}^{(1)}, \mathsf{sk}^{(1)}), \ldots, (\mathsf{pk}^{(d)}, \mathsf{sk}^{(d)})$ for $\mathcal{E}$. Alice's public key contains $(\mathsf{pk}^{(1)}, \ldots, \mathsf{pk}^{(d)}, \overline{\mathsf{sk}^{(1)}}, \ldots, \overline{\mathsf{sk}^{(d-1)}})$ where $\overline{\mathsf{sk}^{(i)}}$ contains encryptions under $\mathsf{pk}^{(i+1)}$ of the secret key bits of $\overline{\mathsf{sk}^{(i)}}$ for $i \in [d-1]$. The public key pk_i is associated with a circuit depth of $i-1$ (i.e. level $i-1$). To evaluate a circuit (of NAND gates in this case), Bob computes a single NAND gate on the ciphertexts at each level before transforming the resulting ciphertexts to the public key of the next level via the procedure above.

2.1.4 Background on Lattices

Informally, a lattice can be viewed geometrically as a set of points in m-dimensional space with a periodic structure. Algebraically, an n-dimensional lattice L is a discrete additive subgroup of $\mathbb{R}^m$ for some integer $m \geq n$. A *basis* of a lattice is a set of n linearly independent vectors in $\mathbb{R}^m$; we can view a basis as the columns of a matrix $\mathbf{B} \in \mathbb{R}^{m \times n}$. The lattice $L(\mathbf{B}) \subset \mathbb{R}^m$ generated by a linearly independent basis $\mathbf{B} \in \mathbb{R}^{m \times n}$ is the set $\{\mathbf{B}\vec{\mathbf{w}} : \forall \vec{\mathbf{w}} \in \mathbb{Z}^n\}$.

We now define some specific hard problems on lattices which are of importance in cryptography.

Definition 2.1.6. *Shortest Vector Problem (SVP): Given a lattice L, find* a *shortest non-zero vector $\vec{\mathbf{v}} \in L$ (there may be more than one vector in L whose length is shortest); that is, $\|\vec{\mathbf{v}}\|$ is minimized in some specified norm.*

Definition 2.1.7. *Closest Vector Problem (CVP): Given a lattice L and vector $\vec{\mathbf{w}} \in \mathbb{R}^n$, find a vector $\vec{\mathbf{v}} \in L$ that is closest to $\vec{\mathbf{w}}$; that is, $\|\vec{\mathbf{w}} - \vec{\mathbf{v}}\|$ is minimized in some specified norm.*

Definition 2.1.8. *Shortest Independent Vector Problem (SIVP): Given an n-dimensional lattice L, find n linearly independent vectors $\vec{\mathbf{v_1}}, \ldots, \vec{\mathbf{v_n}}$ such that for every basis $\vec{\mathbf{b}}_1, \ldots, \vec{\mathbf{b}}_n \in \mathbb{R}^m$ for the lattice L, it holds that $\mathsf{max}\|\vec{\mathbf{v_i}}\| < \max \vec{\mathbf{b_i}}$.*

The above problems have approximation variants. In the case of SVP, given some approximation factor $\gamma \in \mathbb{R}$, the goal is to find a non-zero vector in the lattice whose length is at most γ times the length of the shortest non-zero vector. The approximation form of SIVP is defined similarly. In the approximation form of CVP, the goal becomes to find a vector in the lattice whose distance to the target vector is less than γ for some $\gamma \in \mathbb{R}$.

Remark SVP was shown by Ajtai to be NP-hard [12]. CVP is a harder problem than SVP, as SVP can be reduced to CVP [11].

2.1.5 Constructions of Fully Homomorphic Encryption

2.1.5.1 FHE from Ideals: Gentry's Paradigm

The first construction of FHE [93] was based on ideals, in particular, on ideal lattices. As mentioned in [173], ideals are a "natural mathematical object to construct fully homomorphic encryption since they natively support both addition and multiplication". Gentry's framework for an FHE cryptosystem based on ideals has the following form. A user's public and private key corresponds to an ideal I in a ring R. The private key consists of a *short* generator of I, while the public key is a random generator of I. A

ciphertext has a noise component drawn from an ideal J. Typically, we have $J = \langle 2 \rangle \subset R$ when the message space is $\{0, 1\}$. In summary, a ciphertext c that encrypts $m \in \{0, 1\}$ has the form $c = r_i + r_j + m \in R$ where $r_i \xleftarrow{\$} I$ and r_j is a "small" element drawn from J. It is easy to see that addition and multiplication of ciphertexts preserve this form due to the properties of an ideal. In [93], Gentry instantiates R with ideal lattices and obtains a bootstrappable scheme based on two assumptions: the first is a variant of the well-known closest vector problem on *ideal lattices* while the latter is a sparse subset-sum problem that is new and not so widely studied.

Subsequent works also follow the same "blueprint" as that above, and explicitly use ideals. They include a variant of Gentry's scheme due to Smart and Vercauteren [170], and an optimization thereof by Stehl and Steinfeld [171]. Moving away from lattices, van Dijk et al. [174] presented a construction that works over the integers (i.e. $R = \mathbb{Z}$) and assumes hardness of an *approximate GCD* problem. Finally, Brakserksi and Vaikuntanathan [51] based security on the Ring Learning with Errors (RLWE) problem introduced in [136], which has a worst case reduction[‡‡] to problems on ideal lattices. The central contribution in the latter two papers is a new somewhat homomorphic scheme; the authors then use the techniques of Gentry to make the scheme bootstrappable, and hence still rely on the hardness of the (relatively unstudied) sparse subset-sum problem.

2.1.5.2 FHE from Learning with Errors: Brakerski and Vaikuntanathan Paradigm

Another paradigm appeared in 2011 due to Brakerski and Vaikuntanathan [49]. They succeeded in developing a scheme whose security relied on the Learning with Errors problem. The Learning with Errors (LWE) problem was introduced by Regev [161] in 2005, and because of a known worst-case reduction from a hard lattice problem [141, 142, 156, 161], it has received considerable interest from the cryptography community.

Public-key encryption schemes based on LWE such as [97, 161] support a natural

[‡‡] See the Glossary (G8).

additive homomorphism with minimal changes. Brakerski and Vaikuntanathan observe that they too can support *compact* multiplication by using a technique they call *relinearization*. By using the additional technique of *dimension-modulus reduction*, they also achieve a bootstrappable scheme directly from LWE, and without requiring a sparse subset-sum assumption. As with all bootstrappable schemes, they must make a circular security assumption, but besides that, they move FHE on to solid ground. Security based on LWE alone is attractive due to the fact that it has a worst-case reduction from a hard lattice problem.

We call the above paradigm, centered on relinearization, the Brakserki-Vaikuntanathan (BV) paradigm. Building on this, a subsequent work achieved leveled FHE from standard LWE (without a circular security assumption) [48] by introducing the technique of *modulus switching*. Another novel twist to this paradigm due to Brakserki [47] achieved leveled FHE without modulus switching.

BV-based cryptosystems can also be easily adapted to work in a polynomial ring. Security is then based on the hardness of the Ring Learning with Errors (RLWE) problem instead of standard LWE. More concretely, security assumes the hardness of problems on *ideal lattices* instead of general lattices. Despite the special structure of ideal lattices, no algorithm has been found for well-known lattice problems that performs better on ideal lattices than on general lattices [136]. Much of the state of the art in implementing FHE has focused on RLWE-based schemes that employ the ideas above in the polynomial ring setting. The reason for this is the greater performance obtainable in such rings because multiplication can be implemented more efficiently; the complexity is $O(n \log n)$ as opposed to n^2. An example of a recent RLWE FHE scheme is [81].

2.1.5.3 FHE from NTRU

López-Alt, Tromer and Vaikuntanathan (LTV) [135] put forward an FHE scheme based on the public-key cryptosystem NTRU [117]. Remarkably, their construction supports evaluation with up to N different (independently-generated) keys, where N is a parame-

ter chosen in advance. They prove security assuming the hardness of RLWE and another problem, referred to as the Decisional Small Polynomial Ratio (DSPR) problem. The latter, however, is a non-standard assumption that has not been well studied in the literature.

2.1.5.4 Approximate Eigenvector Paradigm

At Crypto 2013, Gentry, Sahai and Waters [98] presented an alternative paradigm to BV in achieving leveled FHE, while still basing security on LWE. Their paradigm avoids the need for relinearization, modulus switching etc. In their paradigm, ciphertexts are matrices that are constructed so that their ("approximate") eigenvectors are the secret key, whose corresponding eigenvalue is the plaintext. The homomorphic properties follow from the inherent homomorphic properties of eigenvectors.

This completes our review of public-key FHE. In the next section, we take a look at the state of the art in identity-based and attribute-based encryption. Finally, we explore the overlap between what we have seen in this section and what appears in the next section; this then sets us up to examine the state of the art in attribute-based homomorphic encryption to date.

2.2 Identity/Attribute Based Encryption

2.2.1 Overview of Identity Based Encryption

Identity Based Encryption (IBE) was first proposed in 1985 by Shamir [167], and remained an open problem until 2001. In an IBE scheme, a Trusted Authority (TA) manages the system and issues secret keys to users corresponding to their identities. Examples of *identity* include an email address, phone number or social security number. The TA is sometimes known as the Private Key Generator (PKG) or Key Generation Center (KGC); however we will use the term TA in this thesis.

The TA generates a master public key and a master secret key, and publishes the

master public key. In this work, we use the term "public parameters" to refer to the master public key, following a recent trend in the literature. The master secret key is used to derive a secret key for a given identity. For example, suppose Bob is issued a secret key for his email address from the TA in the company C that he works for. Alice can send an encrypted message to Bob if she knows the public parameters of C (which may be available, for example, on its website) and Bob's email address. She does not have to fetch and verify a public key for Bob, which eliminates unwieldy PKI. A downside however is the inherent escrow that exists in IBE. In this case, C's TA can read Alice's message since it can derive a secret key for Bob's email address. For this reason, IBE on its own is more appropriate for company and organizational communication than personal communication. There are approaches to help minimize the escrow, including the use of multiple trusted authorities in deriving secret keys [130], certificate-based encryption [90] and certificateless cryptography [15].

Another downside is that revocation is more tricky than conventional PKI. If Bob's secret key is compromised, the attacker can decrypt all messages ever encrypted under that particular *identity string* (in this case, Bob's email address). One solution is to include a time-span as part of an *identity string* to indicate validity [38]. Let's say this is done on a daily basis. Accordingly, Alice encrypts a message with the "identity string" $\mathsf{bob}|\mathsf{date}$ where date denotes the current date. Alternatively Alice may encrypt a message to Bob for a date in the future [144] if the TA is trusted to not release the secret key to Bob before the specified future date. When a time window is embedded in the identity string, an attacker, in the event of a compromise, can decrypt messages associated with that time period. Issuing keys with a finer granularity further minimizes the scope of attack.

Formally, an Identity Based Encryption (IBE) scheme is a tuple of probabilistic polynomial time (PPT) algorithms $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt})$ defined with respect a message space $\mathcal{P}$, an identity space $\mathcal{I}$ and a ciphertext space $\mathcal{C}$ as follows:

- $\mathsf{Setup}(1^\lambda)$:

On input (in unary) a security parameter λ, generate public parameters PP and a master secret key MSK. Output $(\mathsf{PP}, \mathsf{MSK})$.

- $\mathsf{KeyGen}(\mathsf{MSK}, \mathsf{id})$:

  On input master secret key MSK and an identity id: derive and output a secret key $\mathsf{sk}_{\mathsf{id}}$ for identity id.

- $\mathsf{Encrypt}(\mathsf{PP}, \mathsf{id}, m)$:

  On input public parameters PP, an identity id, and a message $m \in \mathcal{P}$, output a ciphertext $c \in \mathcal{C}$ that encrypts m under identity id.

- $\mathsf{Decrypt}(\mathsf{sk}_{\mathsf{id}}, c)$:

  On input a secret key $\mathsf{sk}_{\mathsf{id}}$ for identity id and a ciphertext $c \in \mathcal{C}$, output m' if c is a valid encryption under id; output a failure symbol $\perp$ otherwise.

Like public-key encryption, semantic security in the context of IBE is equivalent to indistinguishability under a chosen plaintext attack (IND-CPA). In more detail, IND-CPA for IBE comes in two flavors - *selective* (denoted by IND-sID-CPA) and *full/adaptive* (denoted by IND-ID-CPA). In the former, the adversary has to choose an identity to attack prior to receiving the public parameters, whereas in the latter, the adversary can make arbitrary secret key queries before choosing a target identity. Informally, these security definitions capture the desired property of collusion resistance. In other words, a number of users who share their secret keys with each other should not be able to derive the secret key of some target identity. Formally, the security notions are defined by an adversary $\mathcal{A}$'s success in the following game(s).

- Set $\mathsf{id}^* \leftarrow \perp$.
- **(Selective-security only)**: $\mathcal{A}$ chooses a target identity $\mathsf{id}^* \leftarrow \mathcal{I}$ to attack.
- The challenger generates $(\mathsf{PP}, \mathsf{MSK}) \leftarrow \mathsf{Setup}(1^\lambda)$, and gives PP to $\mathcal{A}$.

- **Key Queries (1)**: $\mathcal{A}$ can make queries to an oracle $\mathcal{O}$ defined by

$$\mathcal{O}(\mathsf{id}) = \begin{cases} \mathsf{KeyGen}(\mathsf{MSK}, \mathsf{id}) & \text{if } \mathsf{id} \neq \mathsf{id}^* \\ \perp & \text{otherwise} \end{cases}.$$

- **(Full-security only)**: $\mathcal{A}$ chooses its target identity $\mathsf{id}^* \leftarrow \mathcal{I}$ now.
- **Challenge Phase**: $\mathcal{A}$ chooses two messages $m_0, m_1 \in \mathcal{P}$ and sends them to the challenger.
- The challenger uniformly samples a bit $b \xleftarrow{\$} \{0,1\}$, and returns $c^* \leftarrow \mathsf{Encrypt}(\mathsf{PP}, \mathsf{id}^*, m_b)$.
- **Key Queries (2)**: $\mathcal{A}$ makes additional queries to $\mathcal{O}$.
- **Guess**: $\mathcal{A}$ outputs a guess bit b'.

The adversary is said to win the above game if $b = b'$. Let $\mathcal{E}$ be an IBE scheme. We define the advantage $\mathbf{Adv}^{\mathsf{IND}\text{-}X\text{-}\mathsf{CPA}}_{\mathcal{E},\mathcal{A}}(\lambda)$ of $\mathcal{A}$ in the $\mathsf{IND}\text{-}X\text{-}\mathsf{CPA}$ game for $X \in \{\mathsf{sID}, \mathsf{ID}\}$ as

$$\mathbf{Adv}^{\mathsf{IND}\text{-}X\text{-}\mathsf{CPA}}_{\mathcal{E},\mathcal{A}}(\lambda) = \mathsf{Pr}[b = b'] - \frac{1}{2}$$

where the probability is taken over all the random coins used by the challenger and $\mathcal{A}$. We say that $\mathcal{E}$ is $\mathsf{IND}\text{-}X\text{-}\mathsf{CPA}$ secure if $\mathbf{Adv}^{\mathsf{IND}\text{-}X\text{-}\mathsf{CPA}}_{\mathcal{E},\mathcal{A}}(\lambda) = \mathsf{negl}(\lambda)$.

IBE was first securely realized in 2001 by Boneh and Franklin [38] based on bilinear pairings, and independently by Cocks [66] based on quadratic residuosity. Thus far, secure realizations have been achieved from bilinear pairings, quadratic residuosity and lattices, which we will look at momentarily.

IBE was extended to the notion of Hierarchical IBE (HIBE) by Horwitz and Lynn [118], and the first HIBE construction was presented shortly thereafter by Gentry and Silverberg [99]. A HIBE scheme gives identities a hierarchical structure. An identity string can be subdivided into sub-identities, which can in turn be further subdivided, up to some depth d. Delegation is also facilitated insofar as a secret key for some identity can be used to derive secret keys for its subordinate identities, and so on.

Before examining constructions of IBE, we will first describe Attribute Based Encryption (ABE), a generalization of IBE with support for more fine-grained access control.

2.2.2 Overview of Attribute Based Encryption

Sahai and Waters [164] introduced a generalization of IBE known as *Fuzzy IBE*. In a *Fuzzy IBE* scheme, an identity is viewed as a descriptive set of attributes. A secret-key holder for identity id' can decrypt a ciphertext encrypted with identity id if id' is "close" to id under some metric e.g: set overlap when the identities are viewed as sets of attributes. This gave rise to what the authors called *Attribute Based Encryption*, where a secret key authorized a party to decrypt ciphertexts associated with certain sets of attributes.

Goyal et al. [113] formulated two complimentary flavors of *Attribute Based Encryption* (ABE): Key-Policy ABE (KP-ABE) and Ciphertext-Policy ABE (CP-ABE). In KP-ABE, a user Alice encrypts her message with a descriptive tag or *attribute*[§§] while the TA issues secret keys for *access policies* that permit users to decrypt ciphertexts with certain attributes. In CP-ABE, on the other hand, an encryptor specifies an access policy when encrypting her message, and the TA issues secret keys to parties that correspond to attributes. So the situation is the reverse of KP-ABE. In fact, there is another lesser-known form called dual-policy ABE [21] that mixes both KP-ABE and CP-ABE.

Let us consider KP-ABE in slightly more detail. When encrypting a message m, Alice chooses a descriptive attribute a from some set $\mathbb{A}$. The trusted authority (TA) issues secret keys for *access policies* to users depending on their credentials. To be more precise, a policy $f\colon \mathbb{A} \to \{0,1\}$ can be viewed as a predicate whose domain is $\mathbb{A}$. Hence, if a user Bob is given a secret key for a policy f, he can decrypt messages with attributes that satisfy f. More precisely, let c_a be a ciphertext that encrypts the message m with some attribute $a \in \mathbb{A}$. Then Bob can recover the message m if and only if $f(a) = 1$.

[§§]Some authors refer to what we call an *attribute* as a "set of attributes". The latter notion is modelled by viewing an *attribute* as a set (of "subattributes").

More generally, to cover both KP-ABE and CP-ABE, consider the following formulation due to Gentry, Sahai and Waters [98]. In their formulation, both forms of ABE are generalized in a straightforward way. Let $\mathbb{X}$ be a set of strings known as *indices*, and let $\mathbb{Y}$ be some set of strings. Then consider a relation R defined on both $\mathbb{X}$ and $\mathbb{Y}$. A sender encrypts her message with an *index* $x \in \mathbb{X}$, and a receiver can only decrypt the message if he has a secret key sk_y for the string $y \in \mathbb{Y}$ with $R(x, y) = 1$. So in the KP-ABE flavor, we have $\mathbb{X} = \mathbb{A}$, $\mathbb{Y} = \mathbb{F}$ and $R(a, f) = f(a)$. In contrast, the CP-ABE flavor is represented as $\mathbb{X} = \mathbb{F}$, $\mathbb{Y} = \mathbb{A}$ and $R(f, a) = f(a)$. This generality is useful when we need to discuss commonalities between the two forms of ABE; one such case below is security definitions.

The notions of selective and adaptive security for IBE are defined analogously for ABE. In the selective security game, which we denote by IND-sel-CPA, the adversary chooses a target *index* x^* prior to receiving the public parameters. In the adaptive security game, which we denote by IND-AD-CPA, the adversary chooses a target index x^* after receiving the public parameters, and in addition, after a phase of secret key queries for strings y of its choice. However, in both games, the adversary is not allowed to query secret keys for strings y with $R(x^*, y) = 1$.

IBE can be viewed as a special case of ABE where $\mathbb{X} = \mathbb{Y}$ and R is the equality relation.

The major research directions for ABE have considered widening the expressiveness of the access policies, and strengthening the proofs of security. The latter entails three primary facets: (1). achieving adaptive over selective security; (2). working in the standard model instead of the random oracle model[¶¶] (ROM); and (3). relying only on well-established computational hardness assumptions.

Katz, Sahai and Waters [124] introduced a generalization of ABE, in particular KP-ABE, called Predicate Encryption (PE). A salient characteristic of PE is *attribute-hiding*, which means that a ciphertext reveals no information about its associated attribute.

[¶¶] See the Glossary (G7)

Alice encrypts her message with some attribute $a \in \mathbb{A}$, and sends the ciphertext c to Bob. Bob can decrypt c if he holds a key for a predicate $f\colon \mathbb{A} \to \{0,1\}$ with $f(a) = 1$. There are two forms of PE known as *match-concealing* and *match-revealing* [168]. In the former, Bob cannot learn which attribute $a' \in \mathsf{supp}(f)$ is associated with c (note that $\mathsf{supp}(f)$ denotes the *support* of f; that is, the set $\{a' \in \mathbb{A} : f(a') = 1\}$). However, the other form, *match-revealing*, is weaker and may leak to Bob the attribute a associated with c.

PE can be viewed in two ways. It can be viewed as a means to delegate computation to a third party i.e. allow the third party to perform a precise fixed function f on the encrypted data c, and thus limit what the third party learns about the data. This is particularly true if we remove the payload (i.e. the message) from the ciphertext, and focus our attention on the output of the predicate f. In the spirit of this viewpoint, a generalization known as Functional Encryption (FE) has been proposed [42, 152], which allows general functions (beyond Boolean-valued functions) to be evaluated. There have been several exciting developments in FE in recent years which are outside the scope of this thesis [10, 25, 87, 104, 108, 110, 111].

PE can also be viewed, like ABE, as a means to achieve more fine-grained access control. In this case, it retains its payload and the attribute serves as a guard that protects access to the payload.

2.2.3 Constructions from Bilinear Pairings

Let us start by formally defining a bilinear pairing (also known as a bilinear map). The following is based on the definition from [78] (Section 2).

Definition 2.2.1. *Let $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ be cyclic groups of order n. We write $\mathbb{G}_1$ and $\mathbb{G}_2$ additively, and $\mathbb{G}_T$ multiplicatively. A bilinear pairing $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is an efficiently computable map satisfying*

- Bilinearity: *$e(aP, bQ) = e(P, Q)^{ab}$ for all $P \in \mathbb{G}_1$, $Q \in \mathbb{G}_2$ and $a, b \in \mathbb{Z}_n$.*

- Non-degeneracy: *If P is a generator for $\mathbb{G}_1$ and Q is a generator for $\mathbb{G}_2$, then $e(P, Q) \neq 1$.*

The group $\mathbb{G}_T$ is known as the target group.

Examples of bilinear pairings (or their modifications) that are used in cryptography include the Tate Pairing [22, 86], Weil Pairing [37], and Ate Pairing [116]. Before Joux [122] employed the Weil Pairing to construct a one-round Diffie-Hellman key exchange with three parties, bilinear pairings had been mainly used in cryptanalysis against elliptic curves [84, 139]. At present, all known efficient realizations of bilinear pairings are based on elliptic curves. However, our discussion will remain at a high-level; we do not need to discuss concrete realizations of the bilinear pairings or the underlying groups.

Boneh and Franklin [38] presented the first IBE scheme in 2001 based on bilinear pairings. To give the reader an idea how IBE is realized with a bilinear pairing, we sketch the Boneh-Franklin scheme below.

2.2.3.1 Boneh-Franklin

The Boneh-Franklin scheme uses prime order groups; that is, $n = p$ for some prime p in Definition 2.2.1. Moreover, the pairing e used in Boneh-Franklin is symmetric; that is, $\mathbb{G}_1 = \mathbb{G}_2$.

Let $\mathbb{G}$ and $\mathbb{G}_T$ be cyclic groups of prime order p. Let $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be a bilinear pairing satisfying Definition 2.2.1. Boneh-Franklin relies on the hardness of the following computational problem in $(\mathbb{G}, \mathbb{G}_T, e)$.

Definition 2.2.2 ((Computational) Bilinear Diffie-Hellman (BDH) Problem, [38])**.** *Given (P, aP, bP, cP) for some $a, b, c \in \mathbb{Z}_p^*$ compute $W = e(P, P)^{abc} \in \mathbb{G}_T$. An algorithm $\mathcal{A}$ has advantage ϵ solving BDH in $(\mathbb{G}, \mathbb{G}_T, e)$ if*

$$\mathsf{Pr}[\mathcal{A}(P, aP, bP, cP) = e(P, P)^{abc}] \geq \epsilon$$

where the probability is over the random choice of $a, b, c \in \mathbb{Z}_p^*$, the random choice of $P \in \mathbb{G}$, and the random coins of $\mathcal{A}$.

Let $\mathbb{G}^* = \mathbb{G} \setminus \{\mathcal{O}_{\mathbb{G}}\}$ where $\mathcal{O}_{\mathbb{G}}$ is the identity element of $\mathbb{G}$. We now sketch the basic Boneh-Franklin scheme from [38], referred to therein as BasicIdent. We assume there is an algorithm $\mathcal{G}$ that takes a security parameter λ, and outputs an appropriately selected tuple $(\mathbb{G}, \mathbb{G}_T, e)$ such that the hardness of the BDH problem in $(\mathbb{G}, \mathbb{G}_T, e)$ matches the desired security level λ.

- **Setup**: Run $(\mathbb{G}, \mathbb{G}_T, e) \leftarrow \mathcal{G}(\lambda)$. Pick a generator P of $\mathbb{G}$. Choose two hash functions $H_1 : \{0,1\}^* \to \mathbb{G}^*$ and $H_2 : \mathbb{G}_2 \to \{0,1\}^m$ for some integer m, which are modeled as random oracles in the security proofs in [38]. Choose a random integer $s \xleftarrow{\$} \mathbb{Z}_p^*$, and compute $P_{\mathsf{pub}} \leftarrow sP$.

  The message space is $\mathcal{P} = \{0,1\}^m$. The ciphertext space is $\mathcal{C} = \mathbb{G} \times \{0,1\}^m$. The public parameters are $\mathsf{PP} := (\mathbb{G}, \mathbb{G}_T, e, m, P, P_{\mathsf{pub}}, H_1, H_2)$ (note that p is assumed to be implicit from the description of the groups). The master secret key is $\mathsf{MSK} := s \in \mathbb{Z}_p^*$.

- **KeyGen**: On input an identity string $\mathsf{id} \in \{0,1\}^*$: compute $Q_{\mathsf{id}} \leftarrow H_1(\mathsf{id}) \in \mathbb{G}^*$ and output $D_{\mathsf{id}} \leftarrow sQ_{\mathsf{id}}$, where s is the master secret key.

- **Encrypt**: To encrypt message $\mu \in \mathcal{P}$ under identity id: compute $Q_{\mathsf{id}} \leftarrow H_1(\mathsf{id}) \in \mathbb{G}^*$; choose random $r \in \mathbb{Z}_p^*$; set $\mathsf{CT} \leftarrow (R := rP, \psi := \mu \oplus H_2(e(Q_{\mathsf{id}}, P_{\mathsf{pub}})^r))$; and output ciphertext vector CT.

- **Decrypt**: To decrypt ciphertext $\mathsf{CT} := (R, \psi)$ with secret key $D_{\mathsf{id}} \in \mathbb{G}^*$ for identity id: compute $\mu \leftarrow \psi \oplus H_2(e(D_{\mathsf{id}}, R))$ and output $\mu \in \mathcal{P}$.

Observe that for any identity id, the element $Q_{\mathsf{id}} = H_1(\mathsf{id})$ can be uniquely represented as $Q_{\mathsf{id}} = uP \in \mathbb{G}^*$ for some integer $u \in \mathbb{Z}_p^*$, since P generates $\mathbb{G}$. It follows from the bilinearity property of the pairing e that the decryptor computes (before applying H_2) the following element in $\mathbb{G}_T$:

$$e(D_{\mathsf{id}}, R) = e(D_{\mathsf{id}}, rP) = e(sQ_{\mathsf{id}}, rP) = e(usP, rP) = e(P, P)^{usr} \in \mathbb{G}_T,$$

which matches the element in $\mathbb{G}_T$ that the encryptor computes (before applying H_2) i.e.

$$e(Q_{\mathsf{id}}, P_{\mathsf{pub}})^r = e(uP, sP)^r = e(P, P)^{usr} \in \mathbb{G}_T.$$

Correctness follows immediately from this fact.

We won't discuss details of the security proof here, but a key observation is that (uP, sP, rP) is a BDH problem instance. The solution to this instance, namely the element $e(P, P)^{usr} \in \mathbb{G}_T$, is used to hide the plaintext. The proof uses this fact to obtain a reduction from BDH to breaking the semantic security of the scheme, which relies on careful programming of the random oracles H_1 and H_2. Hence, Boneh-Franklin is adaptively secure in the random oracle model.

2.2.3.2 IBE Without Random Oracles

Bilinear pairings have served as fertile ground for achieving IBE. Security of these schemes has been commonly based on the decisional form of BDH, referred to as DBDH, and its variants. One of the paramount challenges after Boneh-Franklin was removing the random oracles. However, the earliest constructions [33, 55] in the standard model were only selectively secure.

Recall that in the selective security model, the attacker must commit to his target identity in advance of receiving the public parameters. Boneh and Boyen [34] showed how to generically convert a selectively-secure IBE into a fully secure (i.e. adaptively secure) IBE. This requires a restriction on the size of the identity space $\mathcal{I}$, but its size is still allowed to be exponentially large (in the security parameter) so this does not affect any practical considerations. Suppose $\mathcal{I}$ consists of all binary strings of length m; that is, $\mathcal{I} = \{0, 1\}^m$. An unfortunate consequence of the generic reduction from the selective model to the adaptive model is that it incurs a degradation in security by a factor of 2^m. What this means concretely is that to achieve adaptive security of 2^λ against an attacker, one must set the parameters of the scheme to achieve selective security of $2^{m+\lambda}$. So if $m = \lambda$ (say), then the security parameter of the scheme must be set to 2λ, which

leads to less efficient encryption/decryption performance and larger ciphertexts. Note that an adaptively-secure scheme obtained via this conversion has an exponential-time reduction from the complexity assumption (such as DBDH) on which selective security is based.

Boneh and Boyen [35] presented the first adaptively secure IBE system in the standard model with a polynomial-time reduction from a standard complexity assumption, namely DBDH. They pointed out however that their scheme serves largely as a possibility result due to the scheme's impracticality. Waters [175] simplified their scheme to make it considerably more efficient, although the public parameters are not compact and the security reduction is not "tight" (it loses a multiplicative factor of $O(q)$ where q is the number of queries). Gentry [91] proposed a more practical scheme with a "tighter" reduction, but relying on a stronger assumption (a variant of DBDH).

A significant breakthrough came in 2009 with the work of Waters [176], who proposed a new proof strategy, termed *dual system encryption*, that enabled him to construct an adaptively-secure IBE (and HIBE) with compact parameters under well-established assumptions (DBDH and the decisional linear assumption). Dual system encryption is a departure from the dominant proof paradigm, known as *partitioning*, that had been mainly used to prove the security of IBE schemes in the standard model. In the *partitioning* paradigm, the simulator (i.e. the reduction algorithm that uses the attacker to solve the *hard problem* on which security is based) splits the identity space into two classes: identities that it can derive a secret key for, and those that it cannot. The goal is to ensure that the adversary's target identity is in the latter class. This can be achieved with far greater ease in the selective model because the simulator knows the target identity when generating the public parameters. In the adaptive model, the main technique used in previous works [35, 91, 175] is to abort the simulator unless the adversary's queries all fall into the first class and the target identity falls into the second class - the simulator partitions appropriately to ensure that this event happens with inverse polynomial probability. Dual system encryption works differently. It splits keys

and ciphertexts into two types: *normal* and *semi-functional*. A semi-functional key cannot decrypt a semi-functional ciphertext, but all other combinations work (e.g: a semi-functional key can decrypt a normal ciphertext etc.). Besides this, the two types of objects are indistinguishable. The proof proceeds by first changing the challenge ciphertext to a semi-functional one, and then one-by-one, letting each queried secret key be semi-functional. Finally, stripping the simulator's ability to make normal keys results in a game where the simulator can use the attacker's advantage to solve the hard problem.

2.2.3.3 ABE

Table 2.1 summarizes the progress made on constructing ABE from bilinear maps, and categorizes the schemes based on type (KP-ABE / CP-ABE / PE), class of supported access policies and security proof. With regard to the latter, Table 2.1 mentions a variety of computational assumptions; we defer the interested reader to the corresponding paper for more information on these assumptions[***]. Due to the additional richness of ABE, proving security is more difficult than IBE. There has been much research in recent years towards expanding the class of access policies supported. Non-monotone formulae (i.e. Boolean formulae with negations) are the most expressive class for which an ABE scheme from bilinear maps has been proposed. Although the ABE scheme with the strongest security guarantee [133] supports only monotone formulae[†††], such formulae are still quite expressive. Inner-product predicates can handle CNF and DNF formulae, but of (poly)logarithmic degree only, which makes them less expressive than monotone formulae.

[***]MBDH = Decisional Modified Bilinear Diffie-Hellman (Definition 3 in [164]). GSD = General Subgroup Decision Assumption (Assumption 1 in [133]), 3-DH = 3-Party Diffie-Hellman Assumption, q-PBDHE = q-parallel Bilinear Diffie-Hellman Exponent Assumption (Defined in [133]). n-eDDH = n-Extended Decisional Diffie-Hellman Assumption (Definition 7 in [132])

[†††]There is an extension to represent non-monotone formulae by explicitly including "negative attributes" in the attribute sets.

	Type	**Access Policies**	**Security**	**Assumption**	**Model**
Sahai and Waters [164]	KP-ABE	k-out-of-n threshold	Selective	MBDH	ROM
Goyal, Pandey, Sahai and Waters [113]	KP-ABE	Monotone Formulae	Selective	DBDH	Standard
Bethencourt, Sahai and Waters [30]	CP-ABE	Monotone Formulae	Adaptive		Generic Group Model + ROM
Cheung and Newport [57]	CP-ABE	AND gates	Selective	DBDH	Standard
Katz, Sahai and Waters [124]	PE (KP-ABE with attribute-hiding)	Inner-Product Predicates	Selective		Generic Group Model
Goyal, Jain, Pandey and Sahai [112]	CP-ABE	Monotone Formulae	Selective	DBDH	Standard
Ostrovksy, Sahai and Waters [153]	KP-ABE	Non-monotone Formulae	Selective	DBDH	Standard
Waters [177]	CP-ABE	Monotone Formulae	Selective	PBDHE	Standard
Lewko, Okamoto, Sahai, Takashima and Waters [132]	PE (KP-ABE with attribute-hiding)	Inner-Product Predicates	Adaptive	n-eDDH	Standard
Lewko and Waters [133]	CP-ABE	Monotone Formulae	Adaptive	3-DH, q-PBDHE, GSD	Standard

Table 2.1: ABE schemes from Bilinear Pairings

2.2.4 Constructions from Quadratic Residuosity

One of the earliest IBE schemes was presented by Cocks [66], which is adaptively secure in the random oracle model under the hardness of the quadratic residuosity (QR) problem. QR is a well-studied and standard problem from number theory, and basing security on such an assumption is therefore attractive. The major problem with Cocks' system is that it suffers from high ciphertext expansion, requiring two elements in $\mathbb{Z}_N$, where N is an RSA modulus, to encrypt a single bit of plaintext.

Boneh, Gentry and Hamburg (BGH) [39] constructed the first space-efficient variant of the Cocks scheme. The size of ciphertexts using their scheme is quite practical; an ℓ-bit message requires a ciphertext whose size is $\log_2 N + \ell + 1$ bits, which contrasts with $2\ell \cdot \log_2 N$ bits in Cocks. However, encryption time in their scheme is quartic in the security parameter, and thus has poor performance.

Cocks' IBE is not anonymous, but anonymous variants have been proposed in the literature [20, 39, 70]. So far, however, there has been no success in constructing schemes from QR with more expressive access policies than IBE.

2.2.5 Constructions from Lattices

Gentry, Peikert and Vaikuntanathan (GPV) [97] introduced the first semantically-secure IBE based on a hard problem on lattices. More specifically, they proved selective security in the random oracle model of their scheme assuming the hardness of the Learning with Errors (LWE) problem. LWE-based schemes with full security in the standard model followed [7, 8, 56]. The first LWE-based scheme to go beyond IBE in terms of the access policies supported was the inner-product predicate encryption scheme of Agrawal, Freeman and Vaikuntanathan, who achieved selective security under LWE. This was followed by an ABE for threshold functions, [9] which facilitated IBE; the security proof was also in the selective model.

In more recent times, more ground has been made against pairings-based construc-

tions. Gorbunov, Vaikuntanathan and Wee [109] described a selectively-secure scheme that could support circuits of polynomial depth as its class of access policies. This is the first ABE whose access policies can be arbitrary computations. The state-of-the-art up that point was support for Boolean formulae. Since then, another selectively-secure ABE for circuits was constructed from multilinear maps [88].

A trend the reader may notice is that the ABE scheme achieved from lattices so far tend to be selectively secure only. Of course, one can apply Boneh and Boyen's [34] generic conversion from a selectively-secure to a fully-secure scheme, but this is at the expense of much increased parameter sizes (recall that the conversion loses a factor of 2^n where n is the size of the attribute space). There does not seem to be a natural analogue in the lattice world to Water's dual-system approach in the pairings world; recall that that approach that led fruitfully to full security for pairings-based schemes.

Now we have discussed the state-of-the-art in homomorphic encryption and identity-attribute based encryption. In the next section, we explore the overlap between these two topics along with existing research directions, with a view to discerning where our research questions in this thesis fits in.

2.3 Identity-Based/Attribute-Based Homomorphic Encryption

There has been little research into homomorphic encryption in the attribute-based setting. Since IBE is the simplest meaningful special case of ABE[‡‡‡], it is a natural starting point to investigate identity-based homomorphic encryption (IBHE). We will focus first on the single-identity case i.e. where evaluation is supported only on ciphertexts with the same identity. The two most prominent subclasses of homomorphic encryption, GHE and FHE, will be examined in turn in an identity-based context.

[‡‡‡]ignoring public-key encryption.

2.3.0.1 Group Homomorphisms

Identity-Based Group Homomorphic Encryption (IBGHE) has not been formally studied in the literature. Many pairings-based IBE constructions either naturally or with minor modifications admit multiplicative group homomorphisms. As far as we are aware, there are no additive IBGHE schemes. There are no IBE schemes with even an additive homomorphism modulo 2 (i.e. XOR) such as that provided by the Goldwasser-Micali (GM) [106] scheme. XOR-homomorphic schemes such as GM have been used in many applications including sealed-bid auctions [158], biometric authentication [54], the Sanders, Young and Yung (SYY) [166] homomorphic scheme and Fischlin's 2-round protocol [82] for the millionaire's problem [178]. One of the contributions of this thesis is an XOR-homomorphic IBE scheme based on the quadratic residuosity problem. It remains open to construct an unbounded additively homomorphic IBE scheme for a "large" range such as Paillier [154]. Possibly a fruitful step in this direction would be Galbraith's variant of Paillier's cryptosystem based on elliptic curves over rings [85].

Many pairings-based IBE and ABE schemes can be adapted so that they support a multiplicative homomorphism. We list a selection of such schemes in Table 2.2 that represents the state of the art. The contributions of this thesis with respect to ABGHE are also listed in Table 2.2. Put simply, we present the first instance of an additively homomorphic ABGHE. As shown in the Table 2.2, our construction is identity-based and its supported message space is small. Nevertheless, it is the first instance of an additively homomorphic ABGHE, and as we have already seen, such schemes have many applications - more specific applications are discussed in Chapter 4. Note that in Table 2.2, the term "large" means superpolynomally large.

2.3.0.2 Evaluation of more than one homomorphic operation

Recall the BGN [40] scheme from Section 2.1.1.3 that could compactly evaluate quadratic formulae (such as 2-DNF formulae). A BGN-type scheme based on lattices was described

	Message Space	Operation	Access Policies	Security
Variant of Boneh-Franklin [38]	"Large" prime order group	Multiplication	IBE	DBDH in ROM
Katz, Sahai and Waters [124]	"Large" Prime order group	Multiplication	Inner Product Predicates	Selective (Generic Group Model)
Ostrovksy, Sahai and Waters [153]	"Large" Prime order group	Multiplication	Non-monotone Formulae	Selective under DBDH
xhIBE (Chapter 4)	$\{0,1\}$	XOR	IBE	Quadratic Residuosity in ROM
Generalization of xhIBE (Chapter 4)	$\{0,\ldots,e\}$ for poly-sized e	Addition mod e	IBE	e-th Residuosity in ROM

Table 2.2: Attribute Based Group Homomorphic Schemes

by Gentry, Halevi and Vaikuntanathan [94], and the authors point out that it can be adapted to the identity-based setting. Until 2013, this was the IBE scheme with the greatest "homomorphic capacity".

2.3.0.3 Fully Homomorphic Encryption

At his talk at CHES/Crypto 2010, Naccache [146] mentioned "identity-based fully homomorphic encryption" as one of a list of open problems. Towards this goal, it has been pointed out in [50] that some LWE-based FHE constructions can be modified to obtain a "weak" form of an identity-based FHE scheme using the trapdoor functions from [97]; that is, additional information is needed (beyond what can be non-interactively derived from a user's identity) in order to evaluate certain circuits and to perform bootstrapping. Therefore, the valued non-interactivity property of IBE is lost whereby no communication between encryptors and the TA is needed.

For many years, an IBE scheme that could compactly evaluate circuits of polynomial

depth (as in leveled FHE) or even logarithmic depth ("somewhat homomorphic encryption") remained open. At Crypto 2013, Gentry, Sahai and Waters presented the first identity-based (leveled) fully homomorphic encryption scheme [98].

Achieving leveled FHE in the identity-based setting turned out to be quite a tricky problem, for a variety of reasons. Prior to the work of Gentry, Sahai and Waters, there were two paradigms for constructing leveled FHE:

1. Gentry's original paradigm based on ideals, which was introduced in [93] (works which built on this include [170, 174]); and

2. Brakersi and Vaikuntanathan's paradigm based on the learning with errors (LWE) problem [49, 52] entailing techniques such as relinearization, modulus switching and dimension reduction.

It appeared like there was limited potential for obtaining identity-based FHE from the first paradigm because no secure IBE schemes had been constructed with this structure; that is, roughly speaking, no IBE scheme associated an identity with an ideal, and a secret key with a "short" generator for that ideal.

The second paradigm appeared more fruitful. Starting with the work of Gentry, Peikert and Vaikuntanathan (GPV) [97], constructions of IBE from LWE had emerged [7, 8, 56]. But it was not straightforward to adapt Brakersi and Vaikuntanathan's (BV) ideas to the identity-based setting. The main reason for this is that BV-type FHE relies on having "encryptions" of some secret key information, termed an *evaluation key*. If a user directly supplies this information to an evaluator out-of-band, then evaluation can be accomplished as in BV. IBE schemes where the evaluation key can be generated by the key holder, but cannot be derived non-interactively, have been termed "weak" [50, 60]. Due to the difficulty of non-interactively deriving an "encryption" of secret key information for a given identity (based on public information alone) meant that the BV paradigm also seemed inhospitable to IBE.

Gentry, Sahai and Waters (GSW) [98] developed a new paradigm from LWE where

the secret key is an *approximate* eigenvector of a ciphertext. Their construction is both elegant and asymptotically faster than existing FHE schemes. Furthermore, it does not rely on an evaluation key, which means that it can be adapted to support IBE. In fact, a "compiler" was proposed in [98] to transform an LWE-based IBE satisfying certain properties into an identity-based (leveled) fully homomorphic encryption (IBFHE) scheme, and it was noted that several existing LWE-based IBE schemes satisfy the required properties. The resulting IBFHE constructions are leveled i.e. they can evaluate circuits of bounded multiplicative depth (polynomial in the security parameter, and fixed prior to generation of the public parameters). However unlike their public-key counterparts, these constructions are not bootstrappable, since bootstrapping relies on "encryptions" of secret key information, akin to an evaluation key. As such, to the best of our knowledge, there are no known "pure" IBFHE schemes in the literature, since Gentry's bootstrapping theorem from [93] is the only known way of converting a leveled FHE scheme to a "pure" FHE scheme.

Remark Gentry, Sahai and Waters (GSW) [98] also present the first leveled ABFHE from the LWE problem. In their leveled ABGHE, the access policies are circuits of a depth that is bounded a priori. Like their leveled IBFHE constructions, their leveled ABFHE works in the single-attribute setting only.

The contributions in this thesis with respect to attribute-based FHE compared to the state of the art are summarized in Table 2.3. Our feasibility result in Chapter 7 gives a construction that is (1). a "pure" FHE scheme (i.e. it can evaluate all circuits); (2). it supports all polynomially-sized access policies; and (3). it is multi-attribute (i.e. supports evaluation with ciphertexts associated with different attributes). This scheme maximizes each of the facets of an ABHE scheme as described in Section 1.3 in Chapter 1. However this comes at the cost of relying on a primitive called indistinguishability obfuscation (defined in Chapter 7), which is computationally expensive in our construction.

Towards the goal of more practical ABFHE, we take a look at the other contributions listed in Table 2.3. In Chapter 6, we present an identity-based leveled FHE scheme that unlike GSW, supports evaluation on ciphertexts with different identities; hence it is *multi-identity*. Future work is to extend this result so that it supports richer access policies. In Chapter 5, we present a "compiler" that compiles a scheme that can evaluate polylogarithmic circuits (a leveled scheme more than suffices for example) into one that can evaluate circuits of bounded arity, but unbounded depth. So instantiating this scheme with our multi-identity scheme from Chapter 6 yields a fully-homomorphic scheme with a priori bounded arity. This is an important result because of the previously discussed difficulty of evaluating circuits of unbounded depth in the attribute-based setting. We can also instantiate this construction with GSW, and this gives us a single-attribute ABFHE with bounded arity. Our contributions leave open the following question: owing to the fact that our feasibility result in Chapter 7 shows the possibility of multi-attribute ABFHE for all poly-sized access policies, can we bridge the gap with the more "concrete" schemes put forward in this thesis?

	Class of Circuits	**Access Policies**	**Composition**	**Security**
Lattice BGN [94]	2-DNF Formulae	IBE	Single	LWE
GSW [98]	Leveled	Bounded poly-depth circuits	Single	Selective under LWE
MIBHE (Chapter 6)	Leveled	IBE	Multi	Selective under LWE in ROM
Construction from Chapter 5 instantiated with MIBHE	All Bounded-Arity Circuits	IBE	Multi	Selecitve under LWE in ROM, assuming secure Multi-Key FHE
Construction from Chapter 5 instantiated with GSW	All Bounded-Arity Circuits	Bounded poly-depth circuits	Single	Selecitve under LWE in ROM, assuming secure Multi-Key FHE
Multi-Attribute "pure" ABFHE from Chapter 7	All Poly-sized	All Poly-sized	Multi	Selective - assuming indistinguishability obfuscation

Table 2.3: Attribute Based Homomorphic Encryption Schemes

Chapter 3

Characterization of Attribute-Based Homomorphic Encryption

3.1 Overview

As described in the previous chapter, there are two primary, complimentary forms of attribute based encryption (ABE): Key-Policy ABE (KP-ABE) and Ciphertext-Policy ABE (CP-ABE) in which attributes and policies are reversed. The definitions and results of this thesis are not affected by replacing one form with the other (swapping attributes and access policies). For ease of exposition, we focus on the KP-ABE formulation *, since this better fits with the usage scenarios we have in mind.

Let us briefly recall the definition of key-policy attribute based encryption (KP-ABE) from the previous chapter. A trusted authority (TA) generates public parameters and a master secret key. It uses its master secret key to generate secret keys for *access policies*. Alice encrypts her data, using the public parameters, under an "attribute" of her choice

*A generalization of both forms is described in Section 2.2.2.

in some designated set of "attributes". An "attribute" serves as a descriptor for the data she is encrypting. Suppose the TA issues a secret key for some *access policy* to Bob. This access policy essentially describes which attributes he is authorized to access. Bob can decrypt Alice's ciphertext if its associated "attribute" satisfies his *access policy*.

Now let us formally model the core problem addressed by this thesis. Let $\mathbb{F}$ be a set of valid access policies which accept or reject members of a set of attributes $\mathbb{A}$. An access policy $f \in \mathbb{F}$ is represented as a predicate $\mathbb{A} \to \{0, 1\}$. So we say an attribute $a \in \mathbb{A}$ satisfies a policy $f \in \mathbb{F}$ if and only if $f(a) = 1$. Access policies are associated with certain authorized parties in the system, which we call *receivers*. A trusted authority (TA) is responsible for verifying a user's credentials and forming an access policy based thereon. In addition, the TA binds an access policy to a user by issuing a secret key for that policy to the user. Authentication is required to achieve this, but this is outside the scope of our system model. It is assumed that a perfectly secure channel exists between the TA and a user, and there is a means for the user to authenticate herself to the TA. We also assume that the TA is not always online. Every party is assumed to have a copy of the public parameters that the TA publishes. This is a once-off occurrence that happens on initialization / deployment of the system. Once a number of users (potential receivers) have been issued their secret keys, we may assume that the TA is offline for a period of time. To be more precise, the remaining phases of the protocol - encryption, evaluation and decryption - can be carried out without interaction with the TA.

Our goal is to facilitate joint computation on encrypted inputs contributed by multiple independent parties, who may be unaware of each other. These parties are termed *senders*. Let n be the number of senders. Each sender S_i has the liberty to encrypt her input data μ_i under an independently-chosen attribute $a_i \in \mathbb{A}$. We emphasize that we model a sender as a stateful entity who encrypts data under one particular attribute. A party that encrypts data under multiple attributes is modelled as different senders.

Another entity, termed the *evaluator*, who has access to the ciphertexts produced by the senders, can perform a computation C of her choice on any subset of the encrypted

data without being able to decrypt it. Therefore, the evaluator can obtain a *result* $\mu' = C(\mu_1, \ldots, \mu_n)$ of the computation in encrypted form. A ciphertext that encrypts the result of an evaluation is referred to as an *evaluated ciphertext*.

Intuitively, one would expect that the result of a joint computation be decryptable by an entity with an access policy that satisfies all the attributes associated with its inputs. More precisely, one would expect decryption to succeed if a receiver holds a secret key for a policy $f \in \mathbb{F}$ such that $f(a_1) = \cdots = f(a_n) = 1$.

We make the assumption that the length of all attributes in $\mathbb{A}$ is bounded by some fixed polynomial in the security parameter of the scheme. This assumption simplifies the presentation but does not constrain the scope of our definitions - it is straightforward to generalize the definitions to handle variable-length attributes.

We consider two basic settings that are interesting special cases of the above system. These two settings are informally described as follows:

1. (**multi-encryptor**): Evaluation supports ciphertexts with different attributes, but the number of senders n is upper limited by a parameter N (specified when generating the public parameters). The size of an *evaluated ciphertext* is allowed to depend polynomially on n.

2. (**multi-attribute**): Evaluation supports ciphertexts with different attributes, as long as the number of *distinct attributes* $\mathscr{d}$ is upper limited by a parameter $\mathscr{D}$ that is set a priori. The size of an *evaluated ciphertext* is allowed to depend polynomially on $\mathscr{d}$. However, no limit is placed on n. Note the *single-attribute* setting is a special case with $\mathscr{D} = 1$.

To summarize the two settings by their limitations: *multi-encryptor* limits the number of senders (i.e. encryptors) (and in turn, the number of distinct attributes); and *multi-attribute* limits the number of *distinct attributes*.

The most powerful setting is the *multi-attribute* setting. It allows an unbounded number of independent senders to contribute data, provided the number of distinct

attributes chosen is below some limit. Such a limit is also inherent in the *multi-encryptor* setting by virtue of the fact that the number of senders is limited.

Concretely, suppose we have n senders whose data sets are associated with one of $\mathscr{d} \leq n$ attributes from a set $\{a_1, \ldots, a_{\mathscr{d}}\} \subset \mathbb{A}$. Consider an "instantiation" MA of the *multi-attribute* setting with parameter $\mathcal{D} \geq \mathscr{d}$, along with an "instantiation" ME of the *multi-encryptor* setting with parameter $N \geq n$. So by this choice of parameters, we have that both systems accommodate evaluation on the sender's data sets. Let c_{MA} and c_{ME} denote the evaluated ciphertext computed by the evaluator in the case of MA and ME respectively. It follows that $|c_{\mathsf{MA}}|$ depends on $\mathscr{d}$ and $|c_{\mathsf{ME}}|$ depends on N. The most conservative setting of parameters to accommodate the above scenario is $\mathcal{D} = \mathscr{d}$ and $N = n$. But $n \geq \mathscr{d}$, so the ciphertext size in MA is at least as (asymptotically) efficient as the ciphertext size in ME.

The maximally expected $\mathcal{D}$ is always less than or equal to the maximally expected N, which highlights the greater power afforded by a multi-attribute system. However the main argument for the greater power afforded by a multi-attribute system is the fact that such a system can be used to generically construct a multi-encryptor system. The converse does not hold. This reduction gives us reason to explore the possibility of a multi-encryptor system because an impossibility result for multi-encryptor implies impossibility of multi-attribute.

Definition 3.1.1 (Degree of composition)**.** *Let $c_1, \ldots, c_\ell$ be input ciphertexts to an evaluation. Each ciphertext c_i is associated with an attribute $a_i \in \mathbb{A}$. The* ***degree of composition*** *of the evaluation is the number of* ***distinct*** *attributes among the a_i; that is, the cardinality of the set $|\{a_1, \ldots, a_\ell\}|$.*

We use the symbol $\mathscr{d}$ to denote the degree of composition. When the context is unambiguous, the term is abbreviated to *degree*. We use the symbol $\mathcal{D}$ to denote the *maximum* degree of composition supported by a particular system.

In this thesis, we chiefly focus on the multi-attribute setting as opposed to the multi-

encryptor setting. However, we give a construction of a multi-encryptor scheme in Appendix E. This construction is more practical than the multi-attribute schemes put forward in this thesis when there is a small number of independent senders; implementation aspects are discussed in Appendix E. The drawback of the multi-encryptor setting is that there is a bound on the number of senders, so this limits the applications it is suited for. We refer the reader to Appendix E for more information on the multi-encryptor setting.

3.1.1 Models of Access Control for Decryption

A model of access control for decryption specifies how decryption of an evaluated ciphertext is to be performed. Consider an evaluated ciphertext c' associated with d attributes $a_1, \ldots, a_d \in \mathbb{A}$. There are two primary models of decryption, each with different strengths and weaknesses. Both models will be considered in turn.

3.1.1.1 Atomic Access

The intended semantics of this model is that a user should only be able to decrypt an evaluated ciphertext c' if she has a secret key for a policy f that satisfies *all* d attributes $a_1, \ldots, a_d$. In other words, policies are enforced in an "all or nothing" manner. So in order to decrypt a ciphertext c', the decryptor needs a secret key for a policy f with $f(a_1) = \cdots = f(a_d) = 1$. Furthermore, it captures the natural requirement that a decryptor be authorized *completely* to access data associated with a particular attribute.

3.1.1.2 Non-Atomic Access - Collaborative Decryption

The interpretation in this model is that a group of users can pool together their secret keys to decrypt a ciphertext c'. In other words, there may not be a single $f \in \mathbb{F}$ that satisfies all d attributes (or no user holds a secret key for such an f), but the users may share secret keys for a set of policies that "covers all" d attributes. In other words, suppose the group of users have (between them) secret keys for policies $f_1, \ldots, f_k \in \mathbb{F}$.

In this model, they can decrypt c' if and only if for every $i \in [\mathcal{d}]$, there exists a $j \in [\mathcal{k}]$ such that $f_j(a_i) = 1$.

How is decryption performed? There are a few possible approaches:

1. Every user in the group shares their secret keys with each other, and all users can decrypt. However, this violates the *principle of least privilege* and gives users in the group access to data they might not have been explicitly authorized to access.

2. Perform decryption collaboratively using a multi-party computation (MPC) protocol. This approach has been suggested in other works including [135]. The advantage of this approach is that it does not leak any party's secret key to the other parties.

3. It is possible that a user has been issued secret keys for several policies. For example: ABE for disjunctive policies can be achieved with an IBE scheme where the TA issues secret keys for different identities (treated as "attributes") to the same user.

4. Collaborative decryption subsumes the *functionality* of the atomic model i.e. a user with a single policy f satisfying all $\mathcal{d}$ attributes can still decrypt on her own.

Our syntax for attribute based homomomorphic encryption (ABHE) presented in the next section generalizes both models. We do this by parameterizing an ABHE scheme with an integer $\mathcal{K} \in [\mathcal{D}]$, which specifies the maximum number of keys that can be passed to the decryption algorithm. The setting $\mathcal{K} = 1$ specifies the atomic model whereas the setting $\mathcal{K} = \mathcal{D}$ specifies the collaborative model. Note that this is only a syntactic rule, it does not pertain to enforcing the security property of either model. Our "default" model, assumed implicitly without further qualification, is the collaborative model. This is for several reasons, which we will enumerate now:

- In the identity-based setting, collaborative decryption is necessary. In this context, a single f is satisfied by only one attribute (i.e. identity). Suppose an evaluation is

performed on ciphertexts with *different* identities to yield an evaluated ciphertext c'. Clearly, there is no single secret key that is sufficient to decrypt c', since each secret key corresponds to exactly one identity. Because IBE is a special case of ABE, and very important in its own right, we want to ensure we allow multi-identity evaluation.

- As noted above, the collaborative model subsumes the *functionality* of the atomic model. The greater flexibility of permitting multiple users to collaboratively decrypt (such as via MPC) invites more applications.

- The security property in the atomic model is not useful if the group of parties gain access to the input ciphertexts and know the circuit that was evaluated, since in this way, they can decrypt the input ciphertexts and compute the result themselves.

3.2 Attribute Based Homomorphic Encryption

Recall the definition of ABE in Chapter 2. An ABE scheme with message space $\mathcal{P}$, attribute space $\mathbb{A}$ and class of supported access policies $\mathbb{F}$ is a tuple of probabilistic polynomial time (PPT) algorithms $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt})$.

Definition 3.2.1. *A (Key-Policy) Attribute-Based Homomorphic Encryption (ABHE) scheme $\mathcal{E}^{(\mathcal{D},\mathcal{K})}$ for an integer $\mathcal{D} > 0$ and an integer $\mathcal{K} \in [\mathcal{D}]$ is defined with respect to a message space $\mathcal{P}$, an attribute space $\mathbb{A}$, a class of access policies $\mathbb{F} \subseteq \mathbb{A} \to \{0,1\}$, and a class of circuits $\mathbb{C} \subseteq \mathcal{P}^* \to \mathcal{P}$. An ABHE scheme is a tuple of PPT algorithms* $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt}, \mathsf{Eval})$ *where* Setup, KeyGen, Encrypt *are defined equivalently to KP-ABE. We denote by $\mathcal{C}$ the ciphertext space. The decryption algorithm* Decrypt *and evaluation algorithm* Eval *are defined as follows:*

- $\mathsf{Decrypt}(\langle \mathsf{sk}_{f_1}, \ldots, \mathsf{sk}_{f_k} \rangle, c)$*: On input a sequence of $k \leq \mathcal{K}$ secret keys for policies $f_1, \ldots, f_k \in \mathbb{F}$ and a ciphertext c, output a plaintext $\mu' \in \mathcal{P}$ iff every attribute associated with c is satisfied by at least one of the f_i; output $\perp$ otherwise.*

- $\mathsf{Eval}(\mathsf{PP}, C, c_1, \ldots, c_\ell)$*: On input public parameters* PP*, a circuit* $C \in \mathbb{C}$ *and ciphertexts* $c_1, \ldots, c_\ell \in \mathcal{C}$*, output an* evaluated ciphertext $c' \in \mathcal{C}$*.*

More precisely, Eval *is required to satisfy the following properties:*

- *Over all choices of* $(\mathsf{PP}, \mathsf{MSK}) \leftarrow \mathsf{Setup}(1^\lambda)$*,* $C : \mathcal{P}^\ell \to \mathcal{P} \in \mathbb{C}$*, every* $\mathscr{d} \leq \mathcal{D}$*,* $a_1, \ldots, a_\ell \in \mathbb{A}$ *s.t* $|\{a_1, \ldots, a_\ell\}| = \mathscr{d}$*,* $\mu_1, \ldots, \mu_\ell \in \mathcal{P}$*,* $c_i \leftarrow \mathsf{Encrypt}(\mathsf{PP}, a_i, \mu_i)$ *for* $i \in [\ell]$*, and* $c' \leftarrow \mathsf{Eval}(\mathsf{PP}, C, c_1, \ldots, c_\ell)$*:*

  - ***Correctness***

    $$\mathsf{Decrypt}(\langle \mathsf{sk}_{f_1}, \ldots, \mathsf{sk}_{f_\mathscr{k}} \rangle, c') = C(\mu_1, \ldots, \mu_\ell) \text{ iff } \forall i \in [\mathscr{d}]\ \exists j \in [\mathscr{k}] \quad f_j(a_i) = 1 \tag{3.2.1}$$

    for any $\mathscr{k} \in [\mathcal{K}]$*, any* $f_1, \ldots, f_\mathscr{k} \in \mathbb{F}$*, and any* $\mathsf{sk}_{f_j} \leftarrow \mathsf{KeyGen}(\mathsf{MSK}, f_j)$ *for* $j \in [\mathscr{k}]$*.*

  - ***Compactness*** *There exists a fixed polynomial* $s(\cdot, \cdot)$ *for the scheme such that*

    $$|c'| \leq s(\lambda, \mathscr{d}). \tag{3.2.2}$$

The complexity of all algorithms may depend on $\mathcal{D}$. Furthermore, the size of freshly encrypted ciphertexts, the size of the public parameters and the size of secret keys may depend on $\mathcal{D}$. On the other hand, the size of the evaluated ciphertext c' must remain independent of $\mathcal{D}$ (along with the size of the circuit C), but it may depend on the *actual* number of distinct attributes, $\mathscr{d}$, used in the evaluation. Note that single-attribute ABHE is the special case where $\mathcal{D} = 1$ i.e. evaluation is correct only for ciphertexts associated with the same attribute. As mentioned earlier, $\mathcal{K} = 1$ represents the atomic model of decryption whereas $\mathcal{K} = \mathcal{D}$ represents the collaborative model. When the parameter $\mathcal{K}$ is omitted, it can be assumed that $\mathcal{K} = \mathcal{D}$; that is, the notation $\mathcal{E}^{(\mathcal{D})}$ is shorthand for $\mathcal{E}^{(\mathcal{D},\mathcal{D})}$.

Definition 3.2.2. *Multi-Attribute ABHE (MA-ABHE) is a primitive with the same syntax as ABHE except that its* Setup *algorithm takes an additional input* $\mathcal{D} > 0$*, which is*

the maximum degree of composition to support. An instance of MA-ABHE can be viewed as a family of ABHE schemes $\{\mathcal{E}^{(\mathcal{D})} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt}, \mathsf{Eval})\}_{\mathcal{D}>0}$.

Remark In the constructions considered in this work, $\mathbb{A}$ consists of attributes of fixed length. However the above definition is easily generalized to capture variable-length attributes, by letting $|c'|$ grow with the total length of the $\mathcal{d}$ distinct attributes.

A concrete ABHE scheme is characterized by three facets: 1). its supported computations (i.e. the class of circuits $\mathbb{C}$); 2). its supported access policies (the class of access policies $\mathbb{F}$); and 3). its supported composition defined by its maximum degree of composition, $\mathcal{D}$.

3.3 Security Definitions

3.3.1 Semantic Security

The semantic security definition for ABHE is the same as that for ABE, which is described in Section 2.2, except that the adversary has access to the Eval algorithm as well. To recap, there are two definitions of semantic security for ABE: selective and adaptive security. In the selective security game, the adversary chooses the attribute to attack before receiving the public parameters whereas in the adaptive game, the adversary chooses its target attribute after receiving the public parameters. We denote the selective definition by $\mathsf{IND\text{-}sel\text{-}CPA}$ and the adaptive definition by $\mathsf{IND\text{-}AD\text{-}CPA}$. When we discuss the special case of IBE, for consistency with the literature, we denote the selective and adaptive variants in the IBE context by $\mathsf{IND\text{-}sID\text{-}CPA}$ and $\mathsf{IND\text{-}ID\text{-}CPA}$ respectively.

3.3.2 Simulation Model of Evaluation

Let $\mathcal{D}$ and $\mathcal{K} \leq \mathcal{D}$ be fixed parameters denoting the maximum degree of composition and the maximum number of keys passed to the decryption algorithm respectively. Consider

ciphertexts $c_1, \ldots, c_\ell$ encrypted under attributes $a_1, \ldots, a_\ell$ respectively. We expect that a ciphertext c' resulting from an evaluation on $c_1, \ldots, c_\ell$ be decryptable by a set of policies $\{f_i\}_{i \in [\mathcal{k}]}$ with $\mathcal{k} \in [\mathcal{K}]$ if the following two conditions are satisfied: (1). the degree of composition $\mathcal{d}$ is less than $\mathcal{D}$ (i.e. $\mathcal{d} := |\{a_1, \ldots, a_\ell\}| \leq \mathcal{D}$) - for convenience we re-label the $\mathcal{d}$ *distinct* attributes as $a_1, \ldots, a_{\mathcal{d}}$; and (2). for every $i \in [\mathcal{d}]$, there exists a $j \in [\mathcal{k}]$ with $f_j(a_i) = 1$.

Ideally a user who does not have keys for such a set of policies $\{f_i\}_{i \in [\mathcal{k}]}$ should not learn anything about c' except that it is associated with the attributes $a_1, \ldots, a_{\mathcal{d}}$. This implies that such a user should not be able to efficiently decide whether c' was produced from $c_1, \ldots, c_\ell$ or an alternative sequence of ciphertexts $d_1, \ldots, d_{\ell'}$ with the same collection of distinct attributes $a_1, \ldots, a_{\mathcal{d}}$.

Definition 3.3.1 (EVAL-SIM Security)**.** *Let $F \subseteq \mathbb{F}$ be a set of policies, and let $A \subseteq \mathbb{A}$ be a set of attributes. For convenience, we define the predicate*

$$\mathsf{compat}(F, A) = \begin{cases} 1 & \text{if } \exists a \in A \; \forall f \in F \; f(a) = 0 \\ 0 & \text{otherwise .} \end{cases}$$

Let $\mathcal{E}$ be an ABHE scheme with parameters $\mathcal{D}$ and $\mathcal{K}$. We define the following experiments for a pair of PPT adversarial algorithms $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ and a PPT algorithm $\mathcal{S}$.

- $\mathbf{Exp}^{\mathsf{REAL}}_{\mathcal{E},\mathcal{A}}(\lambda)$ ***(Real World):***

  1. $(\mathsf{PP}, \mathsf{MSK}) \leftarrow \mathcal{E}.\mathsf{Setup}(1^\lambda)$.
  2. $(C, (a_1, \mu_1), \ldots, (a_\ell, \mu_\ell), \mathsf{st}) \leftarrow \mathcal{A}_1^{\mathcal{E}.\mathsf{KeyGen}(\mathsf{MSK}, \cdot)}(\mathsf{PP})$.
  3. *Let F be the set of policies queried by $\mathcal{A}_1$.*
  4. *Let $A := \{\mathfrak{a}_1, \ldots, \mathfrak{a}_{\mathcal{d}}\}$ be the distinct attributes in the collection $a_1, \ldots, a_\ell$.*
  5. *Assert $\mathcal{d} \leq \mathcal{D}$ and $\mathsf{compat}(F, A) = 1$; otherwise output a random bit and abort.*

6. $c_j \leftarrow \mathcal{E}.\mathsf{Encrypt}(\mathsf{PP}, a_j, \mu_j)$ *for* $j \in [\ell]$.

7. $c' \leftarrow \mathcal{E}.\mathsf{Eval}(\mathsf{PP}, C, c_1, \ldots, c_\ell)$.

8. $b \leftarrow \mathcal{A}_2^{\mathcal{O}(\mathsf{MSK},\cdot)}(\mathsf{st}, c', c_1, \ldots, c_\ell)$

9. *Output* b.

- $\mathbf{Exp}^{\mathsf{IDEAL}}_{\mathcal{E},\mathcal{A},\mathcal{S}}(\lambda)$ ***(Ideal World):***

1. $(\mathsf{PP}, \mathsf{MSK}) \leftarrow \mathcal{E}.\mathsf{Setup}(1^\lambda)$.

2. $(C, (a_1, \mu_1), \ldots, (a_\ell, \mu_\ell), \mathsf{st}) \leftarrow \mathcal{A}_1^{\mathcal{E}.\mathsf{KeyGen}(\mathsf{MSK},\cdot)}(\mathsf{PP})$.

3. *Let* F *be the set of policies queried by* $\mathcal{A}_1$.

4. *Let* $A := \{\mathfrak{a}_1, \ldots, \mathfrak{a}_{\mathfrak{d}}\}$ *be the distinct attributes in the collection* $a_1, \ldots, a_\ell$.

5. *Assert* $\mathfrak{d} \leq \mathcal{D}$ *and* $\mathsf{compat}(F, A) = 1$*; otherwise output a random bit and abort.*

6. $c_j \leftarrow \mathcal{E}.\mathsf{Encrypt}(\mathsf{PP}, a_j, \mu_j)$ *for* $j \in [\ell]$.

7. $c' \leftarrow \mathcal{S}(\mathsf{PP}, C, A)$.

8. $b \leftarrow \mathcal{A}_2^{\mathcal{O}(\mathsf{MSK},\cdot)}(\mathsf{st}, c', c_1, \ldots, c_\ell)$

9. *Output* b.

where $\mathcal{O}(\mathsf{MSK}, \cdot)$ *is defined as:*

- $\mathcal{O}(\mathsf{MSK}, f)$:

1. *If* $\mathsf{compat}(F \cup \{f\}, A) = 1$*: set* $F \leftarrow F \cup \{f\}$ *and output* $\mathcal{E}.\mathsf{KeyGen}(\mathsf{MSK}, f)$.

2. *Else output* $\perp$.

Then $\mathcal{E}$ *is said to be* EVAL-SIM*-secure if there exists a PPT simulator* $\mathcal{S}$ *such that for every pair of PPT algorithms* $\mathcal{A} := (\mathcal{A}_1, \mathcal{A}_2)$*, it holds that*

$$|\mathsf{Pr}[\mathbf{Exp}^{\mathsf{REAL}}_{\mathcal{E},\mathcal{A}} \to 1] - \mathsf{Pr}[\mathbf{Exp}^{\mathsf{IDEAL}}_{\mathcal{E},\mathcal{A},\mathcal{S}} \to 1]| < \mathsf{negl}(\lambda).$$

Note that the above definition relates to adaptive security. For selective security, the adversary must choose the attributes before receiving the public parameters. As a result, in the modified definition, $\mathcal{A}$ consists of three PPT algorithms $(\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3)$. Furthermore, $\mathcal{A}_1$ outputs a set of $\mathscr{d} \leq \mathcal{D}$ attributes $A := \{\mathfrak{a}_1, \ldots, \mathfrak{a}_{\mathscr{d}}\}$; $\mathcal{A}_2$ receives PP and outputs a circuit C along with a sequence of ℓ pairs (μ_i, a_i) for $i \in [\ell]$ where $\mu_i \in \mathcal{P}$ and $a_i \in A$. Finally, $\mathcal{A}_3$ is defined equivalently to $\mathcal{A}_2$ in the above definition. We denote the selective variant by $\mathsf{sel\text{-}EVAL\text{-}SIM}$.

An even stronger requirement is attribute privacy; in this case, the user without a satisfactory set of policies $\{f_i\}_{i \in [\mathscr{k}]}$ only learns the degree of composition, $\mathscr{d}$, from c' and not the actual attributes. The only change to the above definition to capture this is by passing $\mathscr{d}$ to the simulator $\mathcal{S}$ instead of A. We denote this variant by $\mathsf{ANON\text{-}EVAL\text{-}SIM}$.

Chapter 4

Attribute-Based Group-Homomorphic Encryption

The primary subclasses of homomorphic encryption are group homomorphic encryption (GHE) and fully homomorphic encryption (FHE). In a nutshell, a public key encryption scheme is said to be *group homomorphic* if its decryption algorithm is a group homomorphism [18]. The notion of GHE was characterized by Armknecht et al. [18], and in Chapter 2 we reviewed their formal definition. Although GHE only permits evaluation of a single algebraic operation, it is a very powerful primitive for the following reasons:

1. It is used as a building block in protocols for Private Information Retrieval [129], Electronic Voting [29, 67–69, 74], Oblivious Polynomial Evaluation [148], Private Outsourced Computation [166] and the Millionaire's Problem [82].

2. FHE is currently impractical for many applications, and even if it were to become more practical, it may add unnecessary overhead, especially in applications that only require a single algebraic operation. An example is data aggregation, illustrated by our wireless sensor network scenario in Chapter 1.

To reap the benefits of cryptographic access control in the applications where (public-key) GHE is employed, it is desirable to consider attribute-based GHE. As an example, consider Private Information Retrieval (PIR) [58]. PIR addresses the following problem. Suppose there is a database D with n items $x_1, \ldots, x_n$. Suppose a user wishes to query D to obtain item x_i in such a way that $i \in [n]$ remains private from D. A trivial solution is for D to send back the whole database, but this requires linear communication (in n). Hence, PIR is the problem of privately querying an item from a database with *sublinear* communication. PIR has been realized from GHE [129]. Now consider the case where the sender and receiver are different parties. Furthermore, the intended receiver may not be a known independent party with a public key, but rather one or more parties in an attribute-based infrastructure whose policies fulfill an attribute chosen by the sender that describes the data. In terms of our running WSN scenario, suppose a base station B queries a sensor node S for a particular reading corresponding to a specified time interval. In the event of S being compromised, B wishes that the query remain hidden from S; for example: interest in specific time intervals may be sensitive. Other nodes (such as other base stations) that overhear S's response might also be interested in the reading. Caching the reading when they receive it cuts down on communication. To allow only authorized nodes to access the reading, B encrypts the query with an appropriate attribute (e.g: $(\mathsf{type} := \mathsf{MOISTURE}, \mathsf{region} := R_1)$). These requirements can be satisfied by using the PIR protocol from [129] (which uses GHE) with an attribute-based GHE scheme instead of a public-key GHE scheme.

4.1 Formal Definition

In this section, we present a formal definition of attribute-based GHE (ABGHE), extending the definition of GHE by Armknecht et al [18].

Definition 4.1.1 (Attribute Based Group Homomorphic Encryption (ABGHE), Adapted from Definition 1 in [18])**.** *Let $\mathcal{E} = (G, K, E, D)$ be an ABE scheme with message space*

$\mathcal{P}$, attribute space $\mathbb{A}$, ciphertext space $\widehat{\mathcal{C}}$ and class of predicates $\mathbb{F}$. The scheme $\mathcal{E}$ is group homomorphic if for every $(\mathsf{PP}, \mathsf{MSK}) \leftarrow G(1^\kappa)$, every $f \in \mathbb{F} : \mathsf{supp}(f) \neq \emptyset$, and every $\mathsf{sk}_f \leftarrow K(\mathsf{MSK}, f)$, the message space $(\mathcal{P}, \cdot)$ is a non-trivial group, and there is a binary operation $* : \widehat{\mathcal{C}}^2 \rightarrow \widehat{\mathcal{C}}$ such that the following properties are satisfied for the restricted ciphertext space $\widehat{\mathcal{C}_f} = \{c \in \widehat{\mathcal{C}} : D_{\mathsf{sk}_f}(c) \neq \perp\}$:

GH.1: *The set of all encryptions $\mathcal{C}_f = \{c \mid c \leftarrow E(\mathsf{PP}, a, m), a \in \mathsf{supp}(f), m \in \mathcal{P}\} \subseteq \widehat{\mathcal{C}_f}$ is a non-trivial group with respect to the operation $*$.*

GH.2: *The restricted decryption $D^*_{\mathsf{sk}_f} := D_{\mathsf{sk}_f|\mathcal{C}_f}$ is surjective and $\forall c, c' \in \mathcal{C}_f \quad D_{\mathsf{sk}_f}(c * c') = D_{\mathsf{sk}_f}(c) \cdot D_{\mathsf{sk}_f}(c')$.*

Let us consider Definition 4.1.1 in more detail. Firstly observe that it can be viewed as a special case of ABHE as defined in Definition 3.2.1 from the previous chapter. Secondly, it follows the atomic model of decryption i.e. $\mathcal{K} = 1$. Let $f \in \mathbb{F}$ be any policy that is satisfied by at least one attribute i.e. $\mathsf{supp}(f) \neq \emptyset$. Furthermore, D_{sk_f} is the decryption function indexed by some secret key sk_f for f. We restrict ourselves to the set of ciphertexts $\widehat{\mathcal{C}_f} \in \widehat{\mathcal{C}}$ that decrypt to a plaintext under D_{sk_f}. In other words, this is the set of ciphertexts that do not yield the failure symbol $\perp$ on decryption with D_{sk_f}. Now the set of honest encryptions with any attribute satisfying f (let this be $\mathcal{C}_f$) should be a subset of $\widehat{\mathcal{C}_f}$. This is captured by GH.1 in Definition 4.1.1. However, GH.1 makes an even stronger demand. It requires that $\mathcal{C}_f$ be a non-trivial group with respect to the operation $*$. The homomorphism is described by GH.2. In our case, it means that for any honestly generated ciphertexts $c, c' \in \mathcal{C}_f$, we have $D_{\mathsf{sk}_f}(c * c') = D_{\mathsf{sk}_f}(c) \cdot D_{\mathsf{sk}_f}(c')$.

Is $\widehat{\mathcal{C}_f} = \mathcal{C}_f$? This is not always the case, and our identity-based XOR-homomorphic construction later exemplifies this. Let sk_f be any secret key for a policy f. Suppose there is a decision function $\delta_f : \widehat{\mathcal{C}} \rightarrow \{0, 1\}$ embedded in sk_f that can determine whether an element of $\widehat{\mathcal{C}}$ is an honest encryption that is decryptable by f i.e. $\delta_f(c) = 1 \iff c \in \mathcal{C}_f$. In this case, the decryption function D_{sk_f} simply outputs $\perp$ on input c if and only if

$\delta_f(c) = 0$; it outputs the recovered plaintext otherwise. As a result, we then indeed have that $\widehat{\mathcal{C}_f} = \mathcal{C}_f$. Armknecht et al. introduced the decision function in their definition of GHE for the public-key setting in order to assist their characterization of IND-CCA1 security. However, an efficient decision function does not always exist in the ABE setting. The reason for this is that the decryptor is only given *partial* secret key information sufficient for her policy f, but other information may remain computationally hidden from her without the master secret key. Therefore, a decryptor may not be able to efficiently tell whether a ciphertext c is in $\mathcal{C}_f$. To extend Armknecht et al.'s results, we can introduce an additional condition that requires $\widehat{\mathcal{C}_f}$ and $\mathcal{C}_f$ to be computationally indistinguishable to any PPT adversary with oracle access to $K(\mathsf{MSK}, \cdot)$. Later in the chapter, we show that this condition holds for our XOR-homomorphic scheme. In fact, we also show that $(\widehat{\mathcal{C}_f}, *)$ is itself a group in our scheme.

4.2 Properties

In this section we will establish some properties about ABGHE schemes. To help us in this task, we first define a particular ABGHE scheme which we make reference to throughout the section. Let $\mathcal{E} = (G, K, E, D)$ be a ABGHE scheme satisfying Definition 4.1.1 with message space $(\mathcal{P}, \cdot)$, attribute space $\mathbb{A}$, access policies $\mathbb{F}$, ciphertext space $\widehat{\mathcal{C}}$ and binary operation $* : \widehat{\mathcal{C}} \times \widehat{\mathcal{C}} \to \widehat{\mathcal{C}}$. Fix any $(\mathsf{PP}, \mathsf{MSK}) \leftarrow G(1^\lambda)$. Note that the identity element of $(\mathcal{P}, \cdot)$ is written as $1 \in \mathcal{P}$ since we use multiplicative notation. We assume that $\mathbb{F}$ is free of any *degenerate* policies; that is, policies f with $f(a) = 0\ \forall a \in \mathbb{A}$.

4.2.1 Partition of Access Policies

A fundamental property of an ABGHE scheme is that its class of access policies $\mathbb{F}$ can be partitioned into equivalence classes via a natural relation $\sim$. The relation is defined for any $f, g \in \mathbb{F}$ as

$$f \sim g \quad \text{iff } \mathsf{supp}(f) \cap \mathsf{supp}(g) \neq \emptyset.$$

Now $\sim$ is clearly reflexive and symmetric, but it is not necessarily transitive in the case of an arbitrary ABHE scheme. However if the scheme is group homomorphic, i.e. it satisfies Definition 4.1.1, then $\sim$ is also transitive, and hence an equivalence relation. We now show this formally.

Lemma 4.2.1 (transitivity of $\sim$)**.** *Let* $f_1, f_2, g \in \mathbb{F}$ *such that* $\mathsf{supp}(f_1) \cap \mathsf{supp}(g) \neq \emptyset$ *and* $\mathsf{supp}(f_2) \cap \mathsf{supp}(g) \neq \emptyset$*. Then* $\mathsf{supp}(f_1) \cap \mathsf{supp}(f_2) \neq \emptyset$*.*

The proof of Lemma 4.2.1 is given in Appendix B.

Each equivalence class in $\mathbb{F}/\sim$ consists of policies linked together because their support sets share a common attribute. The equivalence classes in $\mathbb{F}/\sim$ correspond to disjoint sets of attributes. For example, in the case of IBE, we have $|\mathbb{F}/\sim| = |\mathbb{A}|$. In contrast, for a more complex class of access policies, we may have $|\mathbb{F}/\sim| = 1$. This is particularly true when there is an access policy that is satisfied by all attributes. The following corollary follows immediately from Lemma 4.2.1.

Corollary 4.2.1. *If the tautology predicate* $\top$ *(i.e.* $\top(a) = 1\ \forall a \in \mathbb{A}$*) is in* $\mathbb{F}$*, then there exists an attribute* $\mathfrak{a} \in \mathbb{A}$ *such that* $f(\mathfrak{a}) = 1\ \forall f \in \mathbb{F}$*.*

The corollary tells us that if there is a policy that is satisfied by every attribute, then there is at least one attribute $\mathfrak{a}$ that satisfies every policy.

Multiplying a ciphertext c by a ciphertext created with attribute $\mathfrak{a}$ preserves the access restrictions of the ciphertext c. In other words, suppose d is an encryption under attribute $\mathfrak{a}$ and one obtains $e = c * d$, then any policy f that can decrypt c can also decrypt e. This follows immediately from GH.2. Thus encryptions under attribute $\mathfrak{a}$ can be seen as "neutral".

4.2.2 Subgroup Membership Problem

Armknecht et al. characterize the semantic security of (public-key) GHE as a subgroup membership problem, which can be generalized easily to the attribute-based setting. To

describe this, we first establish some notation. For any attribute $a \in \mathbb{A}$ and any plaintext $\mu \in \mathcal{P}$, we define the set $\mathcal{C}_\mu^{(a)}$ as the image of $E_{\mathsf{PP}}(a, \mu)$ i.e. the set of legally generated encryptions of μ under attribute a. In addition, we define $\mathcal{C}^{(a)} = \bigcup_{\mu \in \mathcal{P}} \mathcal{C}_\mu^{(a)}$. Recall that we are using multiplicative notation for groups and that we denote the identity element in $(\mathcal{P}, \cdot)$ by $1 \in \mathcal{P}$.

Suppose the adversary's target attribute is $a^* \in \mathbb{A}$. In the subgroup membership problem (SMP), he is given an element $c^* \in \mathcal{C}^{(a^*)}$ which is sampled in one of two ways: (1). the element c^* is uniformly sampled from $\mathcal{C}^{(a^*)}$; or (2). the element c^* is uniformly sampled from $\mathcal{C}_1^{(a^*)}$. The goal is to distinguish both of these distributions given oracle access to K_{MSK} conditioned on the fact that the adversary cannot query an $f \in \mathbb{F}$ with $f(a^*) = 1$. More precisely, we assume the hardness of a family of subgroup membership problems $\{\mathsf{SMP}_{a^*}\}_{a^* \in \mathbb{A}}$. It can be shown that solving a problem in this family is equivalent to attacking the semantic security of the scheme. For more details, we refer the reader to [18] wherein Armknecht et al. characterize the security of public-key GHE as a subgroup membership problem; the characterization holds analogously for ABGHE.

4.2.3 Generic Transformation for Multiple Attributes

As mentioned earlier, an ABGHE scheme natively follows the atomic model of decryption i.e. $\mathcal{K} = 1$. It is possible to construct a related scheme $\mathcal{E}' = (G', K', E', D')$ that is group homomorphic for $(\mathcal{P}, \cdot)$, but with $\mathcal{D} = \mathcal{K} = |\mathbb{A}|$. Technically $\mathcal{E}'$ is not an ABGHE since it doesn't satisfy Definition 4.1.1. Instead $\mathcal{E}'$ is a group homomorphic scheme that follows the collaborative model of decryption. Its salient feature is that ciphertexts grow linearly with the degree of composition. The transformation is presented in Section B.2 of Appendix B. This generic transformation for multiple attributes is useful for a variety of reasons:

- In the identity-based setting, it allows us to do multi-identity evaluation.

- In cases where there is no single policy that decrypts all $\mathscr{A}$ attributes (or no party has a secret key for such a policy), this gives us a way to do collaborative decryption.

- The resulting scheme $\mathcal{E}'$ serves as a good example of multi-attribute ABHE, and the only example we are aware of that has $\mathcal{D} = |\mathbb{A}|$.

4.2.4 Additively Homomorphic "Sub-Schemes"

It is a well-known that a scheme with a multiplicative homomorphism can be transformed into one with an additive homomorphism, where the addition takes place in the exponent, and a discrete logarithm problem must be solved to recover the result. This gives rise to the following theorem, which holds true in the public-key setting as well (a fortiori because public-key HE is a special case of ABHE):

Theorem 4.2.1. *Let $g \in \mathcal{P}$ be a generator of $(\mathcal{P}, \cdot)$. For any positive integer $M = \mathsf{poly}(\lambda)$ with $M \mid |\mathcal{P}|$, there is an additively homomorphic ABGHE scheme with plaintext group $(\mathbb{Z}_M, +)$.*

Proof. We define a new scheme $\mathcal{E}'$ whose setup and key generation algorithms are the same as $\mathcal{E}$. The element $h := g^{|MS|/M}$ is a generator for a subgroup of $\mathcal{P}$ of order M. One can define the encryption algorithm E' as follows: on input a message $\mu \in \{0, \ldots, M-1\}$ and attribute a, compute $c \leftarrow E_{\mathsf{PP}}(a, h^\mu)$ and output c. The image of $E'_{\mathsf{PP}}(a, \cdot)$ with domain $\mathbb{Z}_M$ is a subgroup of $E_{\mathsf{PP}}(a, \cdot)$ with domain $\mathcal{P}$ with respect to operation $*$. This satisfies GH.1. The decryption algorithm is defined as $D'_{\mathsf{sk}_f}(c) = \log_h(D_{\mathsf{sk}_f}(c))$. Let c be an encryption of $x \in \mathbb{Z}_M$ and c' be an encryption of $y \in \mathbb{Z}_M$. These elements can respectively be viewed as encryptions in the scheme $\mathcal{E}$ of $h^x \in \mathcal{P}$ and $h^y \in \mathcal{P}$ respectively. Because D satisfies GH.2, we have

$$D'_{\mathsf{sk}_f}(c*c') = \log_h D_{\mathsf{sk}_f}(c * c') = \log_h \left(D'_{\mathsf{sk}_f}(c) \cdot D'_{\mathsf{sk}_f}(c')\right) = \log_h \left(h^x \cdot h^y\right) = \log_h \left(h^{x+y}\right) = x+y.$$

Therefore, the scheme also satisfies GH.2. □

A related fact, and one that shows up more frequently, is when M does not divide the group order $|\mathcal{P}|$ and is instead some polynomially sized bound. In this case, we get a bounded (aka "quasi") additively homomorphic scheme, but it is not group homomorphic in the sense of Definition 4.1.1 since one cannot perform an unbounded number of homomorphic operations.

Günther et al. [114] modified the Boneh-Franklin IBE [38] so that it is additively homomorphic in a bounded sense (i.e. it is additively homomorphic for $\mathbb{Z}_M$ for some M that does not divide the order of the group $(\mathcal{P}, \cdot)$). In fact, we could interpret the construction of Günther et al. as first transforming Boneh-Franklin into an ABGHE with a multiplicative homomorphism and then applying the above transformation to yield a *bounded* additive homomorphism. The same transformation can be applied to other pairings-based IBE schemes including [91, 176]. In the next section, we look at existing ABGHE schemes that are multiplicatively homomorphic. We recommend that the reader keep in mind that a *bounded* additive homomorphism can be obtained from these schemes via the above transformation.

4.3 Existing ABGHE Schemes (Multiplicatively Homomorphic)

As aforementioned, variants of parings-based IBE schemes including [38, 91, 176] are ABGHE schemes with a multiplicative homomorphism. Furthermore, these can be transformed via the process described in the proof of Theorem 4.2.1 into a scheme with a *bounded* additive homomorphism. Günther et al. [114] described such a modification to the Boneh-Franklin IBE [38] to produce a *bounded* additively homomorphic scheme.

As we have seen, many pairings-based ABE schemes are multiplicatively homomorphic. To illustrate the properties of a concrete ABGHE, we now examine such a construction due to Katz, Sahai and Waters (KSW) [124] (Appendix C); we call this scheme KSW. The security of KSW relies on non-standard assumptions on bilinear groups, as-

sumptions that are justified by the authors in the generic group model.

Let m be a product of three "large" primes and let n be a positive integer that is polynomial in the security parameter. In KSW, an attribute is an n-dimensional vector over $\mathbb{Z}_m$ and a predicate (i.e. access policy) also corresponds to an n-dimensional vector over $\mathbb{Z}_m$. For $\vec{v} \in \mathbb{Z}_m^n$, a predicate $f_{\vec{v}} : \mathbb{Z}_m^n \to \{0, 1\}$ is defined by

$$f_{\vec{v}}(\vec{w}) = \begin{cases} 1 & \text{iff } \langle \vec{v}, \vec{w} \rangle = 0 \\ 0 & \text{otherwise} \end{cases}$$

These predicates are called inner-product predicates.

Roughly speaking, in a ciphertext, all components of its attribute vector $\vec{\mathbf{w}} \in \mathbb{Z}_m^n$ (which represent the sub-attributes) are blinded by the same uniformly random "blinding" element $b \in \mathbb{Z}_m$. The decryption algorithm multiplies each component by the corresponding component in the predicate vector, and the blinding element b is eliminated when the inner product evaluates to zero with all but negligible probability, which allows decryption to proceed.

Let $\vec{\mathbf{c_1}}$ and $\vec{\mathbf{c_2}}$ be ciphertexts with attribute vectors $\vec{\mathbf{a_1}} \in \mathbb{Z}_m^n$ and $\vec{\mathbf{a_2}} \in \mathbb{Z}_m^n$ respectively. It can be easily shown that the pairwise product $\vec{\mathbf{c'}} = \vec{\mathbf{c_1}} * \vec{\mathbf{c_2}}$ of $\vec{\mathbf{c_1}}$ and $\vec{\mathbf{c_2}}$ produces a ciphertext that is associated with both $\vec{\mathbf{a_1}}$ and $\vec{\mathbf{a_2}}$ in a somewhat "isolated" way. The effect this has is conjunctive. So a predicate vector $\vec{\mathbf{v}}$ has to satisfy $\langle \vec{\mathbf{v}}, \vec{\mathbf{a_1}} \rangle = 0$ and $\langle \vec{\mathbf{v}}, \vec{\mathbf{a_2}} \rangle = 0$ for decryption of $\vec{\mathbf{c'}}$ to succeed (except with negligible probability). Furthermore, the effect on the underlying plaintexts is multiplicative (in a group of order m). Therefore, KSW is an ABGHE scheme with a multiplicative homomorphism. Another property that KSW satisfies is attribute privacy - the attribute vector is hidden by the ciphertext.

KSW also helps us illustrate the aforementioned properties of ABGHE. Consider Corollary 4.2.1, which tells us that if a "tautology" predicate $\top$ (i.e. a predicate that holds true for every attribute) is in the class of supported policies, then there is an attribute $\mathfrak{a} \in \mathbb{A}$ that satisfies all policies. In the case of KSW, such a predicate $\top$ is

given by the zero vector. Accordingly, the attribute $\mathfrak{a}$ is also given by the zero vector.

On a technical note the ciphertexts in KSW are elements of the product group $\hat{\mathcal{C}} := \mathbb{G}_T \times \mathbb{G}^{2n+1}$ where $\mathbb{G}$ and $\mathbb{G}_T$ are groups of order m. The operation $*$ on $\hat{\mathcal{C}}$ corresponds to the operation of this product group. The plaintext group is $(\mathcal{P} := \mathbb{G}_T, \cdot)$. The identity element of the ciphertext space $\hat{\mathcal{C}}$ is $1_{\hat{\mathcal{C}}} := (1_{\mathbb{G}_T}, 1_{\mathbb{G}}, \ldots, 1_{\mathbb{G}}) \in \hat{\mathcal{C}}$ where $1_{\mathbb{G}_T}$ is the identity element of $\mathbb{G}_T$ and $1_{\mathbb{G}}$ is the identity element of $\mathbb{G}$. Note that the identity element $1_{\hat{\mathcal{C}}}$ of $\hat{\mathcal{C}}$ is an encryption of $1 \in \mathcal{P}$ under $\mathfrak{a}$, which is the zero attribute vector in KSW.

4.4 Additively Homomorphic Identity Based Encryption

As discussed in the previous section, there exist multiplicatively homomorphic ABGHE schemes, and furthermore, as we have seen, these can be converted into *bounded* additively homomorphic schemes. However, we are not aware of any (unbounded) additively homomorphic ABGHE. In this section, we present the first such scheme. Our construction is XOR-homomorphic (supports addition modulo 2) but it can be generalized to support addition modulo M for small M, as we show in Section 4.4.8. Our construction is identity-based and its security is based on the quadratic residuosity problem. Therefore, it is similar in many respects to the Goldwasser-Micali (GM) cryptosystem [106], which is also well-known to be XOR-homomorphic. Indeed, the GM scheme has found many practical applications due to its homomorphic property. In Section 4.4.9, we show how many of these applications benefit from an XOR-homomorphic scheme in the identity-based setting.

Our construction derives from the IBE scheme due to Cocks [66] which has a security reduction from the quadratic residuosity problem. To the best of our knowledge, a homomorphic variant has not been explored to date.

4.4.1 Quadratic Residues and Jacobi Symbols

Let m be an integer. A quadratic residue in the residue ring $\mathbb{Z}_m$ is an integer x such that $x \equiv y^2 \mod m$ for some $y \in \mathbb{Z}_m$. The set of quadratic residues in $\mathbb{Z}_m$ is denoted $\mathbb{QR}(m)$. If m is prime, it is easy to determine whether any $x \in \mathbb{Z}_m$ is a quadratic residue. If m is an odd prime number, we can define the Legendre symbol as a function of any integer $x \in \mathbb{Z}$ with respect to m as

$$\left(\frac{x}{m}\right) = \begin{cases} 1 & \text{if } x \in \mathbb{QR}(m) \\ -1 & \text{if } x \not\equiv 0 \mod m \text{ and } x \notin \mathbb{QR}(m) \\ 0 & \text{if } x \equiv 0 \mod m \end{cases}.$$

The above function can be generalized to positive odd moduli $M = m_1^{\alpha_1} \dots m_k^{\alpha_k}$ where $m_1, \dots, m_k$ are prime, and $\alpha_1, \dots, \alpha_k$ are positive integers. The generalization is called a Jacobi symbol and is defined as

$$\left(\frac{x}{M}\right) = \left(\frac{x}{m_1}\right)^{\alpha_1} \cdots \left(\frac{x}{m_k}\right)^{\alpha_k}$$

where $\left(\frac{x}{m_i}\right)$ denotes the Legendre symbol of x with respect to m_i for $1 \leq i \leq k$. The subset of $\mathbb{Z}_M$ with Jacobi symbol $+1$ is denoted by $\mathbb{J}(M)$; that is, $\mathbb{J}(M) = \{x \in \mathbb{Z} : \left(\frac{x}{M}\right) = 1\}$. Naturally, $\mathbb{QR}(M) \subseteq \mathbb{J}(M)$.

4.4.2 Quadratic Residuosity Problem

Let N be a product of two odd primes p and q. The quadratic residuosity problem is to determine, given input (N, x) where $x \in \mathbb{J}(N)$, whether or not $x \in \mathbb{QR}(N)$, and it is believed to be intractable.

4.4.3 Blum Integers

The schemes in this chapter make use of Blum integers. A Blum integer is a product of two primes that are both congruent to 3 modulo 4. As a result, we define $\mathsf{BlumGen}(1^\lambda)$ as

a PPT algorithm which takes as input a security parameter λ and outputs two equally-sized primes p and q, whose lengths depend on λ, such that

$$p \equiv q \equiv 3 \pmod 4.$$

4.4.4 Cocks Scheme

We define the encoding $\nu : \{0,1\} \to \{-1,1\}$ with $\nu(0) = 1$ and $\nu(1) = -1$. Formally, ν is a group isomorphism between $(\mathbb{Z}_2, +)$ and $(\{-1,1\}, *)$. A message bit is mapped to an element of $\{-1,1\}$ via ν (i.e. 0 (1 resp.) is encoded as 1 (-1 resp.)).

Let $H : \{0,1\}^* \to \mathbb{J}(N)$ be a full-domain hash that sends an identity string $\mathsf{id} \in \{0,1\}^*$ to an integer in $\mathbb{Z}_N$ whose Jacobi symbol is $+1$. A secret key in Cocks' system is a Rabin signature for id. What this means is that the secret is the square root of an integer $a \in \mathbb{Z}_N$, where a is obtained via H. For a random a, it is a hard problem to find a square root of a without the factorization of N. To guarantee existential unforgeability of such signatures, we need to model H as a random oracle in the security proof.

- $\mathsf{Cocks}.\mathbf{Setup}(1^\lambda)$:

  1. Repeat: $(p, q) \leftarrow \mathsf{BlumGen}(1^\lambda)$.

     Note that by definition of $\mathsf{BlumGen}$, we have $p \equiv q \equiv 3 \pmod 4$.

  2. $N \leftarrow pq$

  3. Output $(\mathsf{PP} := N, \mathsf{MSK} := (N, p, q))$

- $\mathsf{Cocks}.\mathbf{KeyGen}(\mathsf{MSK}, \mathsf{id})$:

  1. Parse MSK as (N, p, q).

  2. $a \leftarrow H(\mathsf{id})$.

  3. $r \leftarrow a^{\frac{N+5-p-q}{8}} \pmod N$.

     Therefore, either $r^2 \equiv a \pmod N$ or $r^2 \equiv -a \pmod N$.

4. Output $\mathsf{sk}_{\mathsf{id}} := (N, \mathsf{id}, r)$

Remark It is important that this algorithm always output the same square root, since otherwise N can be factored. To achieve this, one may store the root or calculate it deterministically as done so above.

- Cocks.**Encrypt**$(\mathsf{PP}, \mathsf{id}, m)$:

  1. Parse PP as N.
  2. $a \leftarrow H(\mathsf{id})$
  3. Generate $t_1, t_2 \xleftarrow{\$} \mathbb{Z}_N^*$ such that $\left(\frac{t_1}{N}\right) = \left(\frac{t_2}{N}\right) = \nu(m)$ (Recall that $\nu(m)$ maps $m \in \{0,1\}$ into $\{-1,1\}$).
  4. Output $\vec{\psi} := (t_1 + at_1^{-1}, t_2 - at_2^{-1})$

- Cocks.**Decrypt**$(\mathsf{sk}_{\mathsf{id}}, \vec{\psi})$:

  1. Parse $\vec{\psi}$ as $(\vec{\psi_1}, \vec{\psi_2})$
  2. Parse $\mathsf{sk}_{\mathsf{id}}$ as (N, id, r)
  3. $a \leftarrow H(\mathsf{id})$
  4. If $r^2 \equiv a \pmod N$, set $d \leftarrow \vec{\psi_1}$. Else if $r^2 \equiv -a \pmod N$, set $d \leftarrow \vec{\psi_2}$. Else output $\perp$ and abort.
  5. Output $\nu^{-1}(\left(\frac{d + 2r}{N}\right))$

The above scheme can be shown to be adaptively secure in the random oracle model assuming the hardness of the quadratic residuosity problem.

4.4.4.1 Anonymity and Galbraith's Test

Cocks' scheme is not anonymous. Boneh et al. [36] report a test due to Galbraith* that enables an attacker to distinguish the identity of a ciphertext. This is achieved

*Reported as emerging from personal communication in [39].

with overwhelming probability given multiple ciphertexts. It is shown by Ateniese and Gasti [20] that there is no "better" test for attacking anonymity. Briefly, let $a = H(\mathsf{id})$ be the public key derived from the identity id. Let c be a ciphertext in the Cocks' scheme. Galbraith's test is defined as

$$\mathsf{GT}(a, c, N) = \left(\frac{c^2 - 4a}{N}\right)$$

Now if c is a ciphertext encrypted with a, then $\mathsf{GT}(a, c, N) = +1$ with all but negligible probability. For $b \in \mathbb{Z}_N^*$ such that $b \neq a$, the value $\mathsf{GT}(b, c, N)$ is statistically close to the uniform distribution on $\{-1, 1\}$. Therefore, given multiple ciphertexts, it can be determined with overwhelming probability whether they correspond to a particular identity. Hence, this defeats anonymity. We will see later that a generalization of Galbraith's test is integral to our construction.

4.4.5 XOR-homomorphic Construction

Recall that a ciphertext in the Cocks scheme consists of two elements in $\mathbb{Z}_N$. Thus, we have

$$(c, d) \leftarrow \mathsf{Cocks.Encrypt}(\mathsf{PP}, \mathsf{id}, b) \in \mathbb{Z}_N^2$$

for some identity id and bit $b \in \{0, 1\}$. Also recall that only one element is actually used for decryption depending on whether $a := H(\mathsf{id}) \in \mathbb{QR}(N)$ or $-a \in \mathbb{QR}(N)$. If the former holds, it follows that a decryptor has a secret key r satisfying $r^2 \equiv a \pmod{N}$. Otherwise, a secret key r satisfies $r^2 \equiv -a \pmod{N}$. To simplify the description of the homomorphic property, we will assume that $a \in \mathbb{QR}(N)$ and therefore omit the second "component" d from the ciphertext. In fact, the properties hold analogously for the second "component" by simply replacing a with $-a$.

In our homomorphic scheme, each "component" of the ciphertext is represented by a pair of elements in $\mathbb{Z}_N^2$ instead of a single element as in the original Cocks scheme. As mentioned, we will omit the second such pair for the moment. Consider the following encryption algorithm E_a defined by

$\mathsf{E_a}(b : \{0,1\})$:

$t \xleftarrow{\$} \mathbb{Z}_N^*[\nu(b)]$

return $(t + at^{-1}, 2) \in \mathbb{Z}_N^2$.

Furthermore, define the decryption function $D_a(\vec{c}) = \nu^{-1}(c_0 + rc_1)$. The homomorphic operation $\boxplus : \mathbb{Z}_N^2 \times \mathbb{Z}_N^2 \to \mathbb{Z}_N^2$ is defined as follows:

$$\vec{c} \boxplus \vec{d} = (c_0 d_0 + a c_1 d_1, c_0 d_1 + c_1 d_0) \tag{4.4.1}$$

It is easy to see that $D_a(\vec{c} \boxplus \vec{d}) = D_a(\vec{c}) \oplus D_a(\vec{d})$:

$$\begin{aligned} D_a(\vec{c} \boxplus \vec{d}) &= D_a((c_0 d_0 + a c_1 d_1, c_0 d_1 + c_1 d_0)) \\ &= \nu^{-1}((c_0 d_0 + a c_1 d_1) + r(c_0 d_1 + c_1 d_0)) \\ &= \nu^{-1}(c_0 d_0 + r c_0 d_1 + r c_1 d_0 + r^2 c_1 d_1) \\ &= \nu^{-1}((c_0 + r c_1)(d_0 + r d_1)) \\ &= \nu^{-1}(c_0 + r c_1) \oplus \nu^{-1}(d_0 + r d_1) \\ &= D_a(\vec{c}) \oplus D_a(\vec{d}) \end{aligned} \tag{4.4.2}$$

Let $R_a = \mathbb{Z}_N[x]/(x^2 - a)$ be a quotient of the polynomial ring $R = \mathbb{Z}_N[x]$. It is more natural and convenient to view ciphertexts as elements of R_a and the homomorphic operation as multiplication in R_a. Furthermore, decryption equates to evaluation at the point r. Thus the homomorphic evaluation of two ciphertext polynomials $c(x)$ and $d(x)$ is simply $e(x) = c(x) * d(x)$ where $*$ denotes multiplication in R_a. Decryption becomes $\nu^{-1}(e(r))$. Moreover, Galbraith's test is generalized straightforwardly to the ring R_a:

$$\mathsf{GT}(a, c(x)) = \left(\frac{c_0^2 - c_1^2 a}{N}\right).$$

We now formally describe our variant of the Cocks scheme that supports an XOR homomorphism. The Setup and KeyGen algorithms are identical to those of the Cocks system, which is presented in Section 4.4.4. Consider the following algorithm $\mathcal{E}$ that takes an integer $a \in \mathbb{J}(N)$ and a plaintext bit $m \in \{0,1\}$ and outputs an element $c(x) \in R$. This algorithm is used to compute one "component" of a ciphertext.

- **Algorithm** $\mathcal{E}(a, m)$**:**

  1. Choose an integer $t \xleftarrow{\$} \mathbb{Z}_N^*$ uniformly such that
  $$\left(\frac{t}{N}\right) = \nu(m).$$
  2. Choose an integer $h \xleftarrow{\$} \mathbb{Z}_N$ uniformly.
  3. Compute $c(x) \leftarrow 2hx + (t + ah^2t^{-1}) \in R$
  4. Repeat steps 1-4 until $(t + ah^2t^{-1}) \in \mathbb{Z}_N^*$.
  5. Output $c(x)$.

  With overwhelming probability, $(t + ah^2t^{-1})$ will be invertible in $\mathbb{Z}_N$.

- In addition, we define a decryption algorithm $\mathcal{D}$ which takes an integer $r \in \mathbb{Z}_N$ and a polynomial in R as input, and outputs a bit $m \in \{0, 1\}$. This is defined as follows:

  Algorithm $\mathcal{D}(r, c(x))$**:**

  1. Compute $j = \left(\frac{c(r)}{N}\right) \in \{-1, 0, +1\}$.
  2. If $j = 0$, output $\perp$.
  3. Else output $\nu^{-1}(j) \in \{0, 1\}$.

We are now ready to fully specify our XOR-homomorphic scheme, which we call xhIBE. As mentioned above, we have xhIBE.Setup = Cocks.Setup and xhIBE.KeyGen = Cocks.KeyGen. The remaining algorithms xhIBE.Encrypt, xhIBE.Decrypt and xhIBE.Add are defined as follows.

- xhIBE.Encrypt(PP, id, m) :

  1. Parse PP as N.
  2. $a \leftarrow H(\mathsf{id})$.

3. Compute $c(x) \leftarrow \mathcal{E}(a, m)$.
4. Compute $d(x) \leftarrow \mathcal{E}(-a, m)$.
5. Output $\vec{\psi} := (c(x), d(x), a)$.

The third component a is necessary to perform homomorphic operations. See xhIBE.Add below.

- xhIBE.Decrypt($\mathsf{sk}_{\mathsf{id}}, \vec{\psi}$):
  1. Parse $\mathsf{sk}_{\mathsf{id}}$ as (N, id, r).
  2. Parse $\vec{\psi}$ as $(c(x), d(x), a)$.
  3. If $r^2 \equiv a \mod N$ and $\mathsf{GT}(a, c(x)) = 1$, output $\mathcal{D}(r, c(x))$.
  4. Else if $r^2 \equiv -a \mod N$ and $\mathsf{GT}(-a, d(x)) = 1$, output $\mathcal{D}(r, d(x))$.
  5. Else output $\perp$.
- xhIBE.Add($\mathsf{PP}, \vec{\psi_1}, \vec{\psi_2}$):
  1. Parse $\vec{\psi_1}$ as $(c_1(x), d_1(x), a)$
  2. Parse $\vec{\psi_2}$ as $(c_2(x), d_2(x), a)$
  3. Output $(c_1(x) * c_2(x) \pmod{x^2 - a}, d_1(x) *_{R_{-a}} d_2(x) \pmod{x^2 + a})$.

We briefly recall why the scheme is XOR-homomorphic. We will restrict our attention to the first component of a ciphertext for simplicity, since the situation is analogous for the second component with respect to $-a$ instead of a. Therefore, we assume that the secret key for identity id is $r \in \mathbb{Z}_N^*$ such that $r^2 = a \pmod N$ where $a = H(\mathsf{id})$. A plaintext bit encoded as an element of $\{-1, 1\}$ is recovered from a ciphertext polynomial $c(x)$ by computing $\left(\frac{c(r)}{N}\right)$. It is easy to see that $\left(\frac{c'(r)}{N}\right) = \left(\frac{c_1(r)}{N}\right) \cdot \left(\frac{c_2(r)}{N}\right) \in \{-1, 1\}$ where $c'(x) = c_1(x)c_2(x) \pmod{x^2 - a}$ (which is what is computed in xhIBE.Add). Note that $(\{-1, 1\}, *)$ and $(\{0, 1\}, \oplus)$ are isomorphic.

For the remainder of this section, we show that xhIBE fulfills the definition of a group homomorphic scheme, and that it is IND-ID-CPA secure under the quadratic residuosity assumption in the random oracle model

Let $a \in \mathbb{J}(N)$. Let $R_a = R/(x^2 - a)$. To simplify the presentation of the proofs, additional notation is needed. This notation is inherited from [20], and generalized to the ring R_a. Recall the generalization of Galbraith's test to the ring R as follows.

Definition 4.4.1 (Galbraith's Test over R)**.** *Define Galbraith's Test for the ring R as the function $\mathsf{GT} : \mathbb{Z}_N \times R \to \{-1, 0, +1\}$ given by*

$$\mathsf{GT}(a, c(x)) = \left(\frac{c_0^2 - c_1^2 a}{N}\right).$$

Note that N is omitted from GT because it is implicit in the description of R.

Define the subset $G_a \subset R_a$ as follows:

$$G_a = \{c(x) \in R_a : \mathsf{GT}(a, c(x)) = 1\}.$$

Therefore, this is the subset of R_a that passes Galbraith's test. Define the subset $\bar{G_a} \subset R_a$ as follows:

$$\bar{G_a} = \{c(x) \in R_a : \mathsf{GT}(a, c(x)) = -1\}.$$

Correspondingly, this is the subset of R_a that fails Galbraith's test. Now define the subset $S_a \subset G_a$[†]:

$$S_a = \{2hx + (t + ah^2t^{-1}) \in G_a \mid h \in \mathbb{Z}_N, t, (t + ah^2t^{-1}) \in \mathbb{Z}_N^*\}.$$

The subset S_a is precisely the image of the algorithm $\mathcal{E}(a, \cdot)$ defined earlier. We have the following lemma:

Lemma 4.4.1.

1. *$(G_a, *)$ is a multiplicative group in R_a.*

[†]This definition is stricter than its analog in [20] in that all elements are in G_a.

2. $(S_a, *)$ *is a subgroup of* G_a

Proof. We must show that G_a is closed under $*$. Let $c(x), d(x) \in G_a$, and let $e(x) = c(x) * d(x)$.

$$\begin{aligned}
\mathsf{GT}(a, e(x)) &= \left(\frac{e_0^2 - ae_1^2}{N}\right) \\
&= \left(\frac{(c_0d_0 + ac_1d_1)^2 - a(c_0d_1 + c_1d_0)^2}{N}\right) \\
&= \left(\frac{(c_0^2 - ac_1^2)(d_0^2 - ad_1^2)}{N}\right) \\
&= \left(\frac{(c_0^2 - ac_1^2)}{N}\right)\left(\frac{(d_0^2 - ad_1^2)}{N}\right) \\
&= \mathsf{GT}(a, c(x)) \cdot \mathsf{GT}(a, d(x)) \\
&= 1
\end{aligned}$$

Therefore, $e(x) \in G_a$.

It remains to show that every element of G_a is a unit. Let $z = c_0^2 - ac_1^2 \in \mathbb{Z}_N$. An inverse $d_1x + d_0$ of $c(x)$ can be computed by setting $d_0 = \frac{c_0}{z}$ and $d_1 = \frac{-c_1}{z}$ if it holds that z is invertible in $\mathbb{Z}_N$. Indeed such a $d_1x + d_0$ is in G_a. Now if z is not invertible in $\mathbb{Z}_N$ then $p|z$ or $q|z$, which implies that $\left(\frac{z}{p}\right) = 0$ or $\left(\frac{z}{q}\right) = 0$. But $\mathsf{GT}(a, c(x)) = \left(\frac{z}{N}\right) = \left(\frac{z}{p}\right)\left(\frac{z}{q}\right) = 1$ since $c(x) \in G_a$. Therefore, z is a unit in $\mathbb{Z}_N$, and $c(x)$ is a unit in G_a.

Finally, to prove (2), note that the members of S_a are exactly the elements $c(x)$ such that $c_0^2 - c_1^2a$ is a square, and it is easy to see that this is preserved under $*$ in R_a. □

Theorem 4.4.1. xhIBE *is a group homomorphic scheme with respect to the group operation of* $(\mathbb{Z}_2, +)$*. In other words,* xhIBE *satisfies Definition 4.1.1.*

Proof. Let $a = H(\mathsf{id})$ for any valid identity string id. We first need to show that xhIBE satisfies GH.1; that is, that the set of valid encryptions under id forms a group. A ciphertext outputted by $\mathsf{xhIBE.Encrypt}(\mathsf{PP}, \mathsf{id}, \cdot)$ is of the form $(c(x), d(x), a) \in S_a \times S_{-a} \times \mathbb{J}(N)$. Recall that by definition S_a and S_{-a} are precisely the image of $\mathcal{E}(a, \cdot)$ and

$\mathcal{E}(-a, \cdot)$ respectively, which are used by xhIBE.Encrypt to generate $c(x)$ and $d(x)$. By Lemma 4.4.1, G_a is a group and S_a is a non-trivial subgroup of G_a (the same holds analogously for G_{-a} and S_{-a}). Therefore, the set of elements $\mathcal{C}_{\mathsf{id}} := \{(c(x), d(x), a) : c(x) \in S_a, d(x) \in S_{-a}\}$ forms a group under the operation given by xhIBE.Add (let us call this $\boxplus$). Note also that $\boxplus$ is well-defined for pairs of elements in $\hat{\mathcal{C}} := R \times R \times \mathbb{J}(N)$.

The surjective homomorphism between $\mathcal{C}_{\mathsf{id}}$ and $\mathcal{P} := (\mathbb{Z}_2, +)$ has already been shown in the correctness derivation in equation 4.4.2. Therefore, the scheme satisfies GH.2. This completes the proof.

□

Theorem 4.4.2. xhIBE *is* IND-ID-CPA *secure in the random oracle model under the quadratic residuosity assumption.*

Proof. Let $\mathcal{A}$ be an adversary that breaks the IND-ID-CPA security of xhIBE. We use $\mathcal{A}$ to construct an algorithm $\mathcal{S}$ to break the IND-ID-CPA security of the Cocks scheme with the same advantage. $\mathcal{S}$ proceeds as follows:

1. Uniformly sample an element $h \xleftarrow{\$} \mathbb{Z}_N^*$. Receive the public parameters PP from the challenger $\mathcal{C}$ and pass them to $\mathcal{A}$.

2. $\mathcal{S}$ answers a query to H for identity id with $H'(\mathsf{id}) \cdot h^{-2}$ where H' is $\mathcal{S}$'s random oracle. The responses are uniformly distributed in $\mathbb{Z}_N[+1]$.

3. $\mathcal{S}$ answers a key generation query for id with the response $K(\mathsf{id}) \cdot h^{-1}$ where K is its key generation oracle.

4. When $\mathcal{A}$ chooses target identity id^*, $\mathcal{S}$ relays id^* to $\mathcal{C}$. Assume w.l.o.g that H has been queried for id, and that $\mathcal{A}$ has not made a secret key query for id^*. Further key generation requests are handled subject to the condition that $\mathsf{id} \neq \mathsf{id}^*$ for a requested identity id.

5. Let $a = H(\mathsf{id}^*)$. On receiving a challenge ciphertext (c, d) from $\mathcal{C}$, compute $c(x) \leftarrow 2hx + c \in R$ and $d(x) \leftarrow (2hx + d) * r(x) \in R$ where $r(x) \xleftarrow{\$} S_{-a}^{(0)}$ and $S_{-a}^{(0)}$ is the

second component of the set of legal encryptions of 0. From corollary **??**, $d(x)$ is uniformly distributed in $S_{-a}^{(b)}$ where the ciphertext (c, d) in the Cocks scheme encrypts the bit b. It follows that $(c(x), d(x))$ is a perfectly simulated encryption of b under identity id^* in xhIBE. Give $(c(x), d(x))$ to $\mathcal{A}$.

6. Output $\mathcal{A}$'s guess b'.

Since the view of $\mathcal{A}$ in an interaction with $\mathcal{S}$ is indistinguishable from its view in the real game, we conclude that the advantage of $\mathcal{S}$ is equal to the advantage of $\mathcal{A}$. □

4.4.6 Computational Indistinguishability of S_a and G_a

We now show that $S_a \underset{C}{\approx} G_a$ i.e. S_a and G_a are computationally indistinguishable without the factorization of N for $a \in \mathbb{J}(N)$.

Corollary 4.4.1 (Extension of Lemma 2.2 in [20])**.** *The distributions* $D_0 := \{(N, a, t + ah^2t^{-1}, 2h) : N \leftarrow \mathsf{Setup}(1^\lambda), a \xleftarrow{\$} \mathbb{J}(N), t, h \xleftarrow{\$} \mathbb{Z}_N^*)\}$ *and* $D_1 := \{(N, a, z_0, z_1) : N \leftarrow \mathsf{Setup}(1^\lambda), a \xleftarrow{\$} \mathbb{J}(N), z_0 + z_1 x \xleftarrow{\$} G_a \setminus S_a\}$ *are indistinguishable assuming the hardness of the quadratic residuosity problem.*

Proof. The corollary follows immediately from Lemma 2.2 in [20] Let $\mathcal{A}$ be an efficient adversary that distinguishes both distributions. Lemma 2.2 in [20] shows that the distributions $d_0 := (\{(N, a, t + at^{-1}) : N \leftarrow \mathsf{Setup}(1^\lambda), a \xleftarrow{\$} \mathbb{J}(N), t\}$ and $d_1 := \{(N, a, c) : N \leftarrow \mathsf{Setup}(1^\lambda), a \xleftarrow{\$} \mathbb{J}(N), c \xleftarrow{\$} \mathsf{GT}(c, a, N)\}$ are indistinguishable. Given a sample (N, a, c) of either the distribution d_0 or d_1, we can construct an algorithm $\mathcal{S}$ that uses $\mathcal{A}$ to distinguish between the distributions. The algorithm $\mathcal{S}$ generates $h \xleftarrow{\$} \mathbb{Z}_N^*$ and computes $b := h^{-2}a$. It passes the element $(N, b, c, 2h)$ to $\mathcal{A}$. The algorithm $\mathcal{S}$ aborts with the output of $\mathcal{A}$. If the sample is from d_0, then the input passed to $\mathcal{A}$ is sampled according to D_0; otherwise the input passed to $\mathcal{A}$ is sampled according to D_1. If $\mathcal{A}$ can distinguish D_0 and D_1 with non-negligible probability, then $\mathcal{S}$ can distinguish d_0 and d_1 with non-negligible probability, which contradicts Lemma 2.2 in [20]. □

Recall that the set of all encryptions for some policy f is defined as $\mathcal{C}_f = \{c \mid c \leftarrow E(\mathsf{PP}, a, m), a \in \mathsf{supp}(f), m \in \mathcal{P}\} \subseteq \widehat{\mathcal{C}_f}$. In the context of IBE, f is an access policy that holds true for a single attribute i.e. identity. We posed the question the earlier whether $\hat{\mathcal{C}}_f$ (the set of ciphertexts that do not yield $\perp$ on decryption with f) is equal to $\mathcal{C}_f$? Expressing this in terms of IBE, we ask whether $\hat{\mathcal{C}}_{\mathsf{id}} = \mathcal{C}_{\mathsf{id}}$ where $\hat{\mathcal{C}}_{\mathsf{id}}$ is the set of ciphertexts that do not yield $\perp$ on decryption with a secret key for id and $\mathcal{C}_f$ is the set of legally generated ciphertexts under id. Now in the case of our scheme, we have that $\hat{\mathcal{C}}_{\mathsf{id}} = G_a$ and $\mathcal{C}_{\mathsf{id}} = S_a$ where $a = H(\mathsf{id})$. Clearly, we have that $\hat{\mathcal{C}}_{\mathsf{id}} \neq \mathcal{C}_{\mathsf{id}}$. Moreover, there is no efficient decision function that can distinguish between an element of $\hat{\mathcal{C}}_{\mathsf{id}} := G_a$ and $\mathcal{C}_{\mathsf{id}} := S_a$ without the master secret key (i.e. the factorization of N). This is interesting because Armknecht et al. [18] include a decision function in their definition of public-key GHE that decides whether or not a given ciphertext is one that has been legally generated. They found that such a decision function could be defined for every public-key GHE scheme. However, our identity-based XOR-homomorphic scheme provides evidence that such a decision function cannot always exist in the attribute-based world.

4.4.7 Anonymous Variant

The notion of anonymity stems from that of key privacy put forward by Bellare et al. [24]. An IBE scheme is said to be anonymous if an adversary cannot distinguish which identity was used to create a ciphertext, even if the adversary gets to choose a pair of identities to distinguish between. Anonymous IBE is a useful primitive because it can be used to facilitate searching on encrypted data, to allow anonymous broadcasts to be made in a network, and to act as a countermeasure against traffic analysis. A multitude of anonymous IBEs have been constructed based on both pairings and lattices including [7, 36, 38, 45].

Anonymous variants of Cocks' IBE scheme whose security relies on the quadratic residuosity assumption have already been proposed in the literature [20, 39, 70]. The most efficient in terms of ciphertext size is due to Boneh, Gentry and Hamburg [39].

However, encryption time in their scheme is quartic in the security parameter, and thus has poor performance. The PEKS scheme in [70] performs better but still requires many Jacobi symbol computations when used as an anonymous IBE. The most time-efficient anonymous IBE to date was presented at CT-RSA 2009 by Ateniese and Gasti [20]. Their construction has similarly-sized ciphertexts to Cocks' original scheme while there is a drop of approximately 30% in performance compared to Cocks according to our experimental results (for a 1024-bit modulus used to encrypt a 128-bit symmetric key; note that IBE is typically used as part of a KEM-DEM). While this is still practical, it is desirable to obtain an anonymous IBE from quadratic residuosity whose performance is on par with the original Cocks scheme, especially for time-critical applications.

We now exploit the homomorphic property of our XOR-homomorphic construction to construct a new anonymous IBE from quadratic residuosity whose performance closely matches that of the original Cocks scheme. This scheme outperforms the Ateniese and Gasti scheme from [20]. Unfortunately, the size of ciphertexts in our scheme is double that of Cocks, and almost double that of [20]. However, we obtain anonymity using a different approach which we believe to be conceptually simpler. We prove this system ANON-IND-ID-CPA secure in the random oracle model and provide both an analytical and experimental comparison between our approach and that of [20].

4.4.7.1 Overview of our Anonymous IBE

It was observed by Galbraith that for any integer c generated in Cocks system, it is an invariant that

$$\left(\frac{c^2 - 4a}{N}\right) = 1.$$

We expect this to hold with probability negligibly close to $1/2$ for random a. Hence, an adversary has a non-negligible advantage attacking anonymity. In the XOR-homomorphic variant from Section 4.4.5, the integer c is replaced by a polynomial $c(x) = c_1 x + c_0$ in the quotient ring $R_a = \mathbb{Z}_N[x]/(x^2 - a)$. We can generalize the above test for polynomials

in R_a. Define

$$\mathsf{GT}(a, c(x)) = \left(\frac{c_0^2 - c_1^2 a}{N}\right).$$

Now we define two subsets $G_a = \{c(x) \in R_a : \mathsf{GT}(a, c(x)) = 1\}$ and $\bar{G_a} = \{c(x) \in R_a : \mathsf{GT}(a, c(x)) = -1\}$ of R_a. In addition, the set of legally generated ciphertext polynomials (i.e. those in the image of the encryption algorithm) is denoted by the set S_a. It was shown earlier that $S_a \underset{C}{\approx} G_a$ (computationally indistinguishable) even given access to the secret key r. It is also shown that G_a is a multiplicative group in R_a and S_a is a subgroup of G_a.

The main idea behind our anonymous IBE is to allow anonymized ciphertexts to be elements of $\bar{G_a}$ half of the time and G_a the other half. Therefore, the adversary cannot use Galbraith's test to distinguish identities. The main problem however is that we don't know what a "ciphertext" in $\bar{G_a}$ decrypts to without knowing the secret key. We can show that a random element in $\bar{G_a}$ can be sampled by multiplying any fixed element $g(x) \in \bar{G_a}$ by a uniformly random element of G_a. Our idea is to derive this fixed element $g(x)$ from the user's identity using a hash function (modelled as a random oracle in the security proofs), and then multiply it by an encryption of the desired message, which lies in S_a. Since S_a and G_a are computationally indistinguishable, the resultant element $c'(x)$ is also computationally indistinguishable from a random element in $\bar{G_a}$. It can also be shown that the homomorphic property holds even between polynomials in $\bar{G_a}$ and G_a. Therefore, $c'(x)$ is an encryption of the desired message XORed with whatever $g(x)$ decrypts to. Since the decryptor can determine what $g(x)$ decrypts to, she can recover the message. A formal description of the scheme along with a proof of security are provided in Appendix C.

The anonymous IBE retains the XOR homomorphic property, but there is one important caveat to be aware of. In order to perform homomorphic operations, the evaluator needs to know the identity. More precisely, he needs to know $a = H(\mathsf{id})$, which breaks anonymity. But our scheme is *universally anonymous* [115] insofar as anyone can anonymize the ciphertext when required. Therefore, the evaluator can anonymize the

ciphertext when he is finished his homomorphic evaluation.

4.4.8 Extension to larger message spaces

We now consider a generalization of our XOR homomorphic scheme to support addition modulo $e > 2$. Independently, Boneh, LaVigne and Sabin [41] described the same generalization with some small differences. The main idea is to use a generalization of quadratic residues known as power residues. So to provide homomorphic addition modulo e, one relies on the hardness of the e-th residuosity problem, a problem that is defined analogously to the quadratic residuosity problem and believed to be intractable.

We consider the case of prime e. The details of our construction remain the same except that we work in the ring $R = ZZ_N[x]/(x^e - a)$ instead of $ZZ_N[X]/(x^2 - a)$ as before. In addition, $a = H(\mathsf{id})$ is taken to be an element whose e-th power residue symbol modulo N is 1. The size of a ciphertext is e^2 elements of $\mathbb{Z}_N$. Boneh, LaVigne and Sabin describe a variant whose ciphertexts contain only e elements but this requires the seemingly uninstantiable notion of a hash function capable of hashing to e-th residues without the prime factorization of N. We refer the reader to [41] for a fuller discussion on the security and correctness properties of this generalization. A natural open problem is to reduce the ciphertext size to linear in e as opposed to quadratic.

4.4.9 Applications Overview

It turns out that XOR-homomorphic cryptosystems have been considered to play an important part in several applications. The most well-known and widely-used *unbounded* XOR-homomorphic public-key cryptosystem is Goldwasser-Micali (GM) [106], which is based on the quadratic residuosity problem. Besides being used in protocols such as private information retrieval (PIR), GM has been employed in some specific applications such as:

- Peng, Boyd and Dawson (PBD) [158] propose a sealed-bid auction system that

makes extensive use of the GM cryptosystem.

- Bringer et al. [54] apply GM to biometric authentication. It is used in two primary ways; (1) to achieve PIR and (2) to assist in computing the hamming distance[‡] between a recorded biometric template and a reference one.

In some of these applications, an XOR-homomorphic identity-based scheme may be of import.

Consider our wireless sensor network scenario from the introduction (Section 1.2.0.2). An XOR-homomorphic cryptosystem can be turned into an AND-homomorphic cryptosystem, as described by Sanders, Young and Yung (SYY) [166]; we defer the details to that work. Hence, our identity-based XOR-homomorphic scheme can be converted into a scheme with an AND homomorphism. Such a homomorphism is useful for aggregation in wireless sensor networks. Consider a wireless sensor network that senses movement and reports it to a base station. Suppose that movement only takes place infrequently. Each sensor node records whether or not movement was detected, and encrypts such as a YES/NO Boolean value (represented by 0 and 1 respectively). An aggregator node that receives several encrypted readings can use our cryptosystem to homomorphically AND the values together (without seeing the underlying values). The base station that receives the result of this aggregation can decrypt the ciphertext to determine whether there was any movement - if the decrypted value is 0, then movement was sensed somewhere; otherwise no movement was detected.

4.4.10 Performance

With regard to performance, our construction requires 8 multiplications in $\mathbb{Z}_N$ for a single homomorphic operation in comparison to a single multiplication in GM. Furthermore, the construction has higher ciphertext expansion than GM by a factor of 4. Encryption

[‡]The hamming distance between two equal-length vectors is the number of positions at which the vectors differ.

involves 2 modular inverses and 6 multiplications (only 4 if the strongly homomorphic property is forfeited). In comparison, GM only requires 1.5 multiplications on average.

4.5 Summary

In this chapter, a formal definition of attribute based group homomorphic encryption (ABGHE) was provided. Then we discussed several properties of ABGHE schemes. Next we looked at existing multiplicatively homomorphic ABGHE schemes that are based on pairings. We observed that these schemes support a bounded additive homomorphism (addition in the exponent) but are not group homomorphic. To illustrate the properties of ABGHE, we analyzed a multiplicatively homomorphic ABGHE based on pairings due to Katz, Sahai and Waters (KSW) [124], and highlighted its properties.

We presented a construction of an identity-based XOR-homomorphic scheme whose security is based on the quadratic residuosity problem. An anonymous variant was also discussed. Furthermore, an extension to support addition modulo m for small m was described. The space complexity of ciphertexts grows quadratically with m. Finally a simple use-case example for wireless sensor network scenarios is expounded.

Chapter 5

Evaluating Circuits with Bounded Arity

In the previous chapter, attribute-based group homomorphic encryption was investigated. We now turn our attention to the evaluation of more complex functions with the ultimate goal being to evaluate all Boolean circuits. Our goal in this chapter is to construct an ABHE scheme that compactly evaluates all Boolean circuits with N inputs, for any positive integer N, specified as a parameter. To achieve this we need leveled ABFHE and a primitive called multi-key FHE, which we will define momentarily. This result is significant for a variety of reasons:

1. The technique of bootstrapping is currently the only known way to evaluate circuits of unbounded depth. Obtaining ABHE for circuits of unbounded depth has been impeded by the fact that employing bootstrapping in the attribute-based setting (non-interactively) is particularly challenging since bootstrapping requires encryptions of the secret key bits to be available as part of the public key. One of the main results of this thesis is showing that bootstrapping is indeed possible (surprisingly) in the attribute-based setting, but its realization relies on complex computationally-expensive machinery, and thus serves largely as a possibility re-

sult. Our approach in this chapter circumvents the hurdle of bootstrapping in the attribute-based setting, at the expense of bounding the arity of the circuits supported by the resulting scheme.

2. Our construction can be instantiated for any bound N (the parameters grow accordingly). However the message space can be set to $\{0,1\}^w$ for an arbitrary positive integer w. Indeed, as we will see later, no such bound w is required at all in practice but is adopted here to fulfill the syntactic requirement of a finite message space. So every sender can encrypt an input of w bits, and we can set w as large as we want. Therefore N binary strings of length w can be used as "inputs" in an evaluation.

3. The central idea underpinning the approach in this chapter is resurrected later in the thesis in our possibility result for MA-ABFHE for general access policies. Furthermore, the approach undertaken here highlights a potential separation between the worlds of ABE and FHE in achieving MA-ABFHE.

Our work builds upon the notion of multikey FHE put forward by López-Alt, Tromer and Vaikuntanathan [135]. The authors of that work gave a concrete construction of multikey FHE based on NTRU [117]. We present a new multi-key FHE scheme based on Learning with Errors (LWE) in the next chapter.

We show that if there exists a secure multikey FHE scheme whose decryption circuit is of depth $\delta = \delta(\lambda, N)$ and there also exists a secure ABHE scheme $\mathcal{E}$ that is homomorphic for the class of Boolean circuits of depth δ , then there exists an ABHE scheme $\mathcal{E}'$ that is homomorphic for all Boolean circuits in $(\{0,1\}^w)^N \rightarrow \{0,1\}^w$ (i.e. with N independent inputs of width w bits and of arbitrary depth). In particular, $\mathcal{E}'$ inherits the parameters $\mathcal{D}$ and $\mathcal{K}$ from $\mathcal{E}$. This means that if $\mathcal{E}$ is instantiated from an MA-ABHE scheme MA with a desired $\mathcal{D}$, we easily obtain another MA-ABHE scheme MA' that yields a corresponding $\mathcal{E}'$ with the same $\mathcal{D}$.

Our result is (informally) captured in the following theorem statement. We state the theorem formally and provide a proof in Section 5.2.

Theorem 5.0.1 (Informal)**.** *Let N be a positive integer. Let w be a positive integer. Let λ be a security parameter. Suppose there exists a semantically secure multikey FHE scheme whose decryption circuit has depth $\delta = \delta(\lambda, N)$. Suppose there exists a sementically-secure (resp.* EVAL-SIM*-secure) leveled ABFHE scheme $\mathcal{E}_{\mathsf{lABFHE}}$ that can compactly evaluate circuits of depth δ. Then there exists a semantically secure (resp.* EVAL-SIM *secure) ABHE scheme (whose parameters $\mathcal{D}$ and $\mathcal{K}$ are the same as $\mathcal{E}_{\mathsf{lABFHE}}$) that can compactly evaluate all Boolean circuits with N inputs over the domain $\{0,1\}^w$.*

Our construction relies on multi-key FHE and leveled ABFHE. If we have a leveled ABFHE with a class of access policies $\mathbb{F}$, then we get a ("pure") ABFHE for the class of policies $\mathbb{F}$ with a bound N on the number of inputs. The main idea behind our approach is that an encryptor generates a key-pair $(\mathsf{pk}, \mathsf{sk})$ for the multi-key FHE scheme and it encrypts the secret key sk with the leveled ABFHE scheme to obtain ciphertext ψ. Then the encryptor encrypts every bit of plaintext (say w bits) with the multi-key FHE scheme using pk to obtain ciphertext $c_1, \ldots, c_w$. It sends the ciphertext $\mathsf{CT} := (\psi, c_1, \ldots, c_w)$. The evaluator evaluates the circuit on the multi-key FHE ciphertexts and obtains an encrypted result c'. Then it evaluates with the leveled ABFHE scheme the decryption circuit of the multi-key FHE scheme on c' together with the encrypted secret keys (the ψ ciphertexts). We obtain a ciphertext in the leveled ABFHE scheme that encrypts the result of the computation (i.e. what c' encrypts). The size of the resulting ciphertext is independent of N and the size of the circuit. By using our multi-key FHE scheme from the next chapter, we only need the leveled ABFHE scheme to have $L = O(\log N)$ levels where N is the bound on the number of inputs.

5.0.1 Building Blocks

5.0.1.1 Multikey FHE

Multi-Key FHE allows multiple independently-generated keys to be used together in a homomorphic evaluation. The syntax of multi-key FHE imposes a limit N on the number of such keys that can be supported. Furthermore, the size of the evaluated ciphertext does not depend on the size of the circuit (or number of inputs), but instead on the number of independent keys N that is supported. In order to decrypt, the parties who have the corresponding secret keys must collaborate such as in an MPC protocol.

Definition 5.0.1 (Based on Definition 2.1 in [135])**.** *A multi-key $\mathbb{C}$-homomorphic scheme family for a class of circuits $\mathbb{C}$ and message space $\mathcal{P}$ is a family of PPT algorithms $\{\mathcal{E}^{(N)} := (\mathsf{Gen}, \mathsf{Encrypt}, \mathsf{Decrypt}, \mathsf{Eval})\}_{N>0}$ where $\mathcal{E}^{(N)}$ is defined as follows:*

- $\mathsf{MKFHE.Gen}$ *takes as input the security parameter 1^λ and outputs a tuple $(\mathsf{pk}, \mathsf{sk}, \mathsf{vk})$ where pk is a public key, sk is a secret key and vk is an evaluation key.*
- $\mathsf{MKFHE.Encrypt}$ *takes as input a public key pk and a message $m \in \mathcal{P}$, and outputs an encryption of m under pk.*
- $\mathsf{MKFHE.Decrypt}$ *takes as input $1 \leq k \leq N$ secret keys $\mathsf{sk}_1, \ldots, \mathsf{sk}_k$ and a ciphertext c, and outputs a message $m' \in \mathcal{P}$.*
- $\mathsf{MKFHE.Eval}$ *takes as input a circuit $C \in \mathbb{C}$, and ℓ pairs $(c_1, \mathsf{vk}_1), \ldots, (c_\ell, \mathsf{vk}_\ell)$ and outputs a ciphertext c'.*

Informally, evaluation is only required to be *correct* if at most N keys are used in $\mathsf{MKFHE.Eval}$; that is, $|\{\mathsf{vk}_1, \ldots, \mathsf{vk}_\ell\}| \leq N$. Furthermore, the size of an evaluated ciphertext c' must only depend polynomially on the security parameter λ and the number of keys N, and not on the size of the circuit.

The IND-CPA security game for multi-key homomorphic encryption is the same as that for standard public-key encryption; note that the adversary is given the evaluation

key vk.

There are two multi-key FHE schemes to the best of our knowledge: the scheme of López-Alt, Tromer and Vaikuntanathan [135] based on NTRU and our multi-key FHE scheme in the next chapter based on Learning wtih Errors (LWE). Although our construction can work with any multi-key FHE, we obtain better efficiency if we use the multi-key FHE scheme in the next chapter. More precisely, the depth of the decryption circuit of the multi-key FHE in the next chapter is $O(\log N)$ (as opposed to $O(\log^2 N)$ in the case of the multi-key FHE from [135]) which results in fewer levels needed for the leveled ABFHE.

For the remainder of this chapter, we will denote an instance of a multi-key FHE by $\mathcal{E}_{\mathsf{MKFHE}}$.

5.0.1.2 Leveled ABFHE

Our approach uses a leveled ABFHE scheme in an essential way. A leveled ABFHE scheme allows one to evaluate a circuit of bounded depth. The bound on the depth L is chosen in advance of generating the public parameters. Gentry, Sahai and Waters [98] presented the first leveled ABFHE where the class of access policies consists of bounded-depth circuits. They based security on LWE. A leveled Identity-Based FHE (IBFHE) scheme from LWE is also presented in [98]. Furthermore a leveled IBFHE that is multi-identity (supports evaluation on ciphertexts with different identities) from LWE is presented in the next chapter.

Any of the above schemes can be used to instantiate our construction and its properties are inherited by our construction. Therefore if we use an identity-based scheme, our resulting construction is identity-based etc.

For the rest of the paper, we will denote a leveled ABFHE scheme by $\mathcal{E}_{\mathsf{lABFHE}}$ with message space $\mathcal{P}_{\mathcal{E}_{\mathsf{lABFHE}}}$, attribute space $\mathbb{A}_{\mathcal{E}_{\mathsf{lABFHE}}}$ and class of predicates $\mathbb{F}_{\mathcal{E}_{\mathsf{lABFHE}}}$.

5.0.2 Overview of Our Approach

The main idea behind our approach is to exploit multi-key FHE and leveled ABFHE to construct a new ABFHE scheme that can evaluate circuits with up to N inputs, where N is chosen before generating the public parameters. Let $\mathcal{E}_{\mathsf{MKFHE}}$ be a multi-key FHE scheme whose decryption circuit has depth $\delta(\lambda, N)$ where N is the number of independent keys tolerated and λ is the security parameter. Let $\mathcal{E}_{\mathsf{lABFHE}}$ be a leveled ABFHE scheme as described in Section 5.0.1.2 that can compactly evaluate circuits of depth $\delta(\lambda, N)$.

Let w be a positive integer. The supported message space of our scheme is $\mathcal{P} \triangleq \{0,1\}^w$. The supported attribute space is $\mathbb{A} \triangleq \mathbb{A}_{\mathcal{E}_{\mathsf{lABFHE}}}$ and the supported class of access policies is $\mathbb{F} \triangleq \mathbb{F}_{\mathcal{E}_{\mathsf{lABFHE}}}$. In other words, the attribute space and class of access policies is the same as the underlying leveled ABFHE scheme. Finally, the class of supported circuits is $\mathbb{C} \triangleq \mathcal{P}^N \to \mathcal{P}$.

Roughly speaking, to encrypt a message $\mu \in \mathcal{P}$ under attribute $a \in \mathbb{A}$ in our scheme, (1) a key triple $(\mathsf{pk}, \mathsf{sk}, \mathsf{vk})$ is generated for $\mathcal{E}_{\mathsf{MKFHE}}$; (2) μ is encrypted with $\mathcal{E}_{\mathsf{MKFHE}}$ under pk; (3) sk is encrypted with $\mathcal{E}_{\mathsf{lABFHE}}$ under attribute a; (4) the two previous ciphertexts along with vk constitute the ciphertext that is produced. Therefore, $\mathcal{E}_{\mathsf{MKFHE}}$ is used for hiding the message and for homomorphic computation, whereas $\mathcal{E}_{\mathsf{lABFHE}}$ enforces access control by appropriately hiding the secret keys for $\mathcal{E}_{\mathsf{MKFHE}}$.

The evaluator performs homomorphic evaluation on the multi-key FHE ciphertexts and obtains a result c'. It then homomorphically decrypts c' with the leveled ABFHE scheme using the encryptions of the secret keys for $\mathcal{E}_{\mathsf{MKFHE}}$. As a result we obtain a ciphertext whose length is independent of N and the circuit size, which satisfies our compactness condition.

In more concrete terms, we assume without loss of generality that the message space of $\mathcal{E}_{\mathsf{MKFHE}}$ is $\{0,1\}$, and we encrypt a w-bit message $\mu = (\mu_1, \ldots, \mu_w) \in \{0,1\}^w$ one bit at a time using $\mathcal{E}_{\mathsf{MKFHE}}$. Furthermore, let N be the maximum number of keys supported by

$\mathcal{E}_{\mathsf{MKFHE}}$. Our construction can therefore support the class of circuits $\mathbb{C} = \{(\{0,1\}^w)^N \rightarrow \{0,1\}^w\}$. We remind the reader that w can be arbitrarily large, and in practice, the length of plaintexts may be shorter than w. In practice, each sender's input may be of arbitrary size. However, there is a limit, N, on the number of independent senders i.e. the number of inputs to the circuit where the inputs are taken from the domain $\{0,1\}^w$.

5.0.3 Construction

We now present our construction, which we call bABFHE.

5.0.3.1 Setup

On input a security parameter λ and a bound N on the number of inputs to support, the following steps are performed:

1. Choose integer w.
2. Generate $(\mathsf{PP}_{\mathcal{E}_{\mathsf{lABFHE}}}, \mathsf{MSK}_{\mathcal{E}_{\mathsf{lABFHE}}}) \leftarrow \mathcal{E}_{\mathsf{lABFHE}}.\mathsf{Setup}(1^\lambda, 1^L)$ where $L = O(\log \lambda \cdot N)$ is the depth of the decryption circuit of $\mathcal{E}_{\mathsf{lABFHE}}$ for parameters λ and N.
3. Output $(\mathsf{PP} := (\mathsf{PP}_{\mathcal{E}_{\mathsf{lABFHE}}}, \lambda, N, w), \mathsf{MSK} := (\mathsf{PP}, \mathsf{MSK}_{\mathcal{E}_{\mathsf{lABFHE}}}))$.

5.0.3.2 Secret Key Generation

Given the master secret key $\mathsf{MSK} := (\mathsf{PP}, \mathsf{MSK}_{\mathcal{E}_{\mathsf{lABFHE}}})$ and a policy $f \in \mathbb{F}$, a secret key sk_f for f is generated as $\mathsf{sk}_f \leftarrow \mathcal{E}_{\mathsf{lABFHE}}.\mathsf{KeyGen}(\mathsf{MSK}_{\mathcal{E}_{\mathsf{lABFHE}}}, f)$. The secret key $\mathsf{SK}_f := (\mathsf{PP}, \mathsf{sk}_f)$ is issued to the user.

5.0.3.3 Encryption

On input public parameters $\mathsf{PP} := (\mathsf{PP}_{\mathcal{E}_{\mathsf{lABFHE}}}, \lambda, N, w)$, a binary string $\mu = (\mu_1, \ldots, \mu_w) \in \{0,1\}^w$ and an attribute $a \in \mathbb{A}$: the sender first generates a key triple for $\mathcal{E}_{\mathsf{MKFHE}}$; that is, she computes $(\mathsf{pk}, \mathsf{sk}, \mathsf{vk}) \leftarrow \mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Gen}(1^\lambda, 1^N)$. Then she runs $\psi \leftarrow \mathcal{E}_{\mathsf{lABFHE}}.\mathsf{Encrypt}(\mathsf{PP}_{\mathcal{E}_{\mathsf{lABFHE}}}, a, \mathsf{sk})$.

Subsequently she uses pk to encrypt each bit $\mu_i \in \{0,1\}$ in turn using $\mathcal{E}_{\mathsf{MKFHE}}$ for $i \in [w]$; that is, she computes $c_i \leftarrow \mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Encrypt}(\mathsf{pk}, \mu_i)$. Finally she outputs the ciphertext $\mathsf{CT} := (\mathsf{type} := 0, \mathsf{enc} := (\psi, \mathsf{vk}, (c_1, \ldots, c_w)))$.

Remark A ciphertext CT in our scheme has two components: the first is labeled with type and the second is labeled with enc. The former has two valid values: 0 and 1; 0 indicates that the ciphertext is "fresh" while 1 indicates that the ciphertext is the result of an evaluation. The value of the type component specifies how the enc component is to be parsed.

5.0.3.4 Evaluation

On input public parameters $\mathsf{PP} := (\mathsf{PP}_{\mathcal{E}_{\mathsf{lABFHE}}}, \lambda, N, w)$, a circuit $C \in \mathbb{C}$, and ciphertexts $\mathsf{CT}_1, \ldots, \mathsf{CT}_\ell$ with $\ell \leq N$, the evaluator performs the following steps. Firstly, the ciphertexts are assumed to be "fresh" ciphertexts generated with the encryption algorithm. In other words, their type components are all 0. Otherwise the evaluator outputs $\perp$. Consequently, the evaluator can parse CT_i as $(\mathsf{type} := 0, \mathsf{enc} := (\psi_i, \mathsf{vk}_i, (c_1^{(i)}, \ldots, c_w^{(i)})))$ for every $i \in [\ell]$. We denote by a_i the attribute associated with the $\mathcal{E}_{\mathsf{lABFHE}}$ ciphertext ψ_i. The maximum degree of composition of our construction is inherited from that of the underlying leveled ABFHE scheme $\mathcal{E}_{\mathsf{lABFHE}}$. We denote this as usual by $\mathcal{D}$. The evaluator derives the degree of composition as $\mathcal{d} \leftarrow |\{a_1, \ldots, a_\ell\}|$, and outputs $\perp$ and aborts unless $\mathcal{d} \leq \mathcal{D}$.

Next the evaluator computes

$$c' \leftarrow \mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Eval}(C, (c_1^{(1)}, \mathsf{vk}_1), \ldots, (c_w^{(1)}, \mathsf{vk}_1), \ldots, (c_1^{(\ell)}, \mathsf{vk}_\ell), \ldots, (c_w^{(\ell)}, \mathsf{vk}_\ell))$$

and encrypts this ciphertext with the leveled ABFHE scheme under any arbitrary a_i, say a_1; that is, the evaluator computes $\psi_{c'} \leftarrow \mathcal{E}_{\mathsf{lABFHE}}.\mathsf{Encrypt}(\mathsf{PP}_{\mathcal{E}_{\mathsf{lABFHE}}}, a_1, c')$. The final step is to evaluate using $\mathcal{E}_{\mathsf{lABFHE}}$ the decryption circuit $D_{\langle N,\lambda\rangle}{}^*$ of $\mathcal{E}_{\mathsf{MKFHE}}$:

$$\psi \leftarrow \mathcal{E}_{\mathsf{lABFHE}}.\mathsf{Eval}\big(\mathsf{PP}_{\mathcal{E}_{\mathsf{lABFHE}}}, D_{\langle N,\lambda\rangle}, \psi_{c'}, \psi_1, \ldots, \psi_\ell\big).$$

*for the specific case of parameters N and λ

The evaluator outputs the *evaluated ciphertext* $\mathsf{CT}' := (\mathsf{type} := 1, \mathsf{enc} := \psi)$.

Remark Observe that a "fresh" ciphertext has a different form to an evaluated ciphertext. Further evaluation with evaluated ciphertexts is not guaranteed by our construction. Hence it is a 1-hop homomorphic scheme using the terminology of Gentry, Halevi and Vaikuntanathan [95].

5.0.3.5 Decryption

To decrypt a ciphertext $\mathsf{CT} := (\mathsf{type}, \mathsf{enc})$ with a sequence of secret keys $\langle \mathsf{SK}_{f_1} := (\mathsf{PP}, \mathsf{sk}_{f_1}), \ldots, \mathsf{SK}_{f_\ell} := (\mathsf{PP}, \mathsf{sk}_{f_\ell}) \rangle$ for respective policies $f_1, \ldots, f_\ell \in \mathbb{F}$, a decryptor performs the following steps.

If CT is a "fresh" ciphertext (i.e. $\mathsf{type} = 0$), then enc is parsed as $(\psi, \mathsf{vk}, (c_1, \ldots, c_w))$ and the decryptor computes $\mathsf{sk} \leftarrow \mathcal{E}_{\mathsf{lABFHE}}.\mathsf{Decrypt}(\langle \mathsf{sk}_1, \ldots, \mathsf{sk}_\ell \rangle, \psi)$. If $\mathsf{sk} = \perp$, then the decryptor outputs $\perp$ and aborts. Otherwise, she computes

$$\mu_j \leftarrow \mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Decrypt}(\mathsf{sk}, c_j) \text{ for every } j \in [w]$$

and outputs the plaintext $\mu := (\mu_1, \ldots, \mu_w) \in \{0,1\}^w$.

If CT is an evaluated ciphertext (i.e. $\mathsf{type} = 1$), then the decryptor parses enc as ψ and computes $x \leftarrow \mathcal{E}_{\mathsf{lABFHE}}.\mathsf{Decrypt}(\langle \mathsf{sk}_1, \ldots, \mathsf{sk}_\ell \rangle, \psi)$. If $x = \perp$ the decryptor outputs $\perp$ and aborts; otherwise the plaintext $\mu := x \in \{0,1\}^w$ is outputted.

5.0.4 Formal Description

A formal description of the construction bABFHE is given in Figure 5.1. As mentioned previously, the parameters $\mathcal{D}$ (maximum degree of composition) and $\mathcal{K}$ (maximum number of decryption keys passed to $\mathsf{Decrypt}$) are inherited directly from the underlying leveled ABFHE scheme $\mathcal{E}_{\mathsf{lABFHE}}$. Although circuits in the supported class send a sequence of elements in the message space $\mathcal{P} := \{0,1\}^w$ to another element in the message space $\mathcal{P}$, we simplify the description here and assume that each circuit C outputs a single bit. A circuit $\hat{C}$ in our supported class can then be modelled as w such circuits.

Fig. 5.1: Formal Description of scheme bABFHE.

Setup$(1^\lambda, 1^N)$:

1. Choose integer w.
2. Let $g(\cdot,\cdot)$ be a polynomial associated with $\mathcal{E}_{\mathsf{MKFHE}}$ that gives the number of inputs to the decryption circuit for N keys and security parameter λ. Let $L = g(\lambda, N)$.
3. Generate $(\mathsf{PP}_{\mathcal{E}_{\mathsf{lABFHE}}}, \mathsf{MSK}_{\mathcal{E}_{\mathsf{lABFHE}}}) \leftarrow \mathcal{E}_{\mathsf{lABFHE}}.\mathsf{Setup}(1^\lambda, 1^L)$.
4. Output $(\mathsf{PP} := (\mathsf{PP}_{\mathcal{E}_{\mathsf{lABFHE}}}, \lambda, N, w), \mathsf{MSK} := \mathsf{MSK}_{\mathcal{E}_{\mathsf{lABFHE}}})$.

Encrypt(PP, a, μ) :

1. Parse PP as $(\mathsf{PP}_{\mathcal{E}_{\mathsf{lABFHE}}}, \lambda, N, w)$.
2. Parse μ as $(\mu_1, \ldots, \mu_w) \in \{0,1\}^w$.
3. $(\mathsf{pk}, \mathsf{sk}, \mathsf{vk}) \leftarrow \mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Gen}(1^\lambda, 1^N)$
4. $\psi \leftarrow \mathcal{E}_{\mathsf{lABFHE}}.\mathsf{Encrypt}(\mathsf{PP}_{\mathcal{E}_{\mathsf{lABFHE}}}, a, \mathsf{sk})$.
5. $c_i \leftarrow \mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Encrypt}(\mathsf{pk}, \mu_i)$ for $i \in [w]$.
6. Output $\mathsf{CT} := (\mathsf{type} := 0, \mathsf{enc} := (\psi, \mathsf{vk}, (c_1, \ldots, c_w)))$.

KeyGen(MSK, f) :

1. Parse MSK as $(\mathsf{PP}, \mathsf{MSK}_{\mathcal{E}_{\mathsf{lABFHE}}})$.
2. $\mathsf{sk}_f \leftarrow \mathcal{E}_{\mathsf{lABFHE}}.\mathsf{KeyGen}(\mathsf{MSK}_{\mathcal{E}_{\mathsf{lABFHE}}}, f)$.
3. Output $\mathsf{SK}_f := (\mathsf{PP}, \mathsf{sk}_f)$.

Decrypt$(\langle \mathsf{SK}_{f_1}, \ldots, \mathsf{SK}_{f_k} \rangle, \mathsf{CT})$:

1. If $k > \mathcal{K}$: output $\perp$ and abort.
2. Parse SK_{f_i} as $(\mathsf{PP}, \mathsf{sk}_{f_i})$ for $i \in [k]$.
3. Parse PP as $(\mathsf{PP}_{\mathcal{E}_{\mathsf{lABFHE}}}, \lambda, N, w)$.
4. Parse CT as $(\mathsf{type}, \mathsf{enc})$.
5. If $\mathsf{type} = 0$:
   (a) Parse enc as $(\psi, \mathsf{vk}, (c_1, \ldots, c_w))$
   (b) Compute $\mathsf{sk} \leftarrow \mathcal{E}_{\mathsf{lABFHE}}.\mathsf{Decrypt}(\langle \mathsf{sk}_1, \ldots, \mathsf{sk}_k \rangle, \psi)$.
   (c) If $\mathsf{sk} = \perp$: output $\perp$ and abort.
   (d) $\mu_i \leftarrow \mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Decrypt}(\mathsf{sk}, c_i)$ for $i \in [w]$.
   (e) Output $\mu := (\mu_1, \ldots, \mu_w) \in \{0,1\}^w$.
6. Else If $\mathsf{type} = 1$:
   (a) Parse enc as ψ.
   (b) Compute $x \leftarrow \mathcal{E}_{\mathsf{lABFHE}}.\mathsf{Decrypt}(\langle \mathsf{sk}_1, \ldots, \mathsf{sk}_k \rangle, \psi)$.
   (c) If $x = \perp$: output $\perp$ and abort.
   (d) Output $\mu := x \in \{0,1\}^w$.
7. Else output $\perp$.

Eval$(\mathsf{PP}, C, \mathsf{CT}_1, \ldots, \mathsf{CT}_\ell)$:

1. If $\ell > N$: output $\perp$ and abort.
2. Parse PP as $(\mathsf{PP}_{\mathcal{E}_{\mathsf{lABFHE}}}, \lambda, N, w)$.
3. For $i \in [\ell]$:
   (a) Parse CT_i as $(\mathsf{type} := 0, \mathsf{enc} := (\psi_i, \mathsf{vk}_i, (c_1^{(i)}, \ldots, c_w^{(i)})))$.
   (b) Set a_i as the attribute associated with ψ_i.
4. Set $d \leftarrow |\{a_1, \ldots, a_\ell\}|$ (degree of composition).
5. If $d > \mathcal{D}$: output $\perp$ and abort.
6. $c' \leftarrow \mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Eval}(C, (c_1^{(1)}, \mathsf{vk}_1), \ldots, (c_w^{(1)}, \mathsf{vk}_1), \ldots, (c_1^{(\ell)}, \mathsf{vk}_\ell), \ldots, (c_w^{(\ell)}, \mathsf{vk}_\ell))$.
7. $\psi_{c'} \leftarrow \mathcal{E}_{\mathsf{lABFHE}}.\mathsf{Encrypt}(\mathsf{PP}_{\mathcal{E}_{\mathsf{lABFHE}}}, a_1, c')$.
8. Let $D_{\langle N,\lambda \rangle}$ be the decryption circuit of $\mathcal{E}_{\mathsf{MKFHE}}$ for parameters N and λ.
9. $\psi \leftarrow \mathcal{E}_{\mathsf{lABFHE}}.\mathsf{Eval}\big(\mathsf{PP}_{\mathcal{E}_{\mathsf{lABFHE}}}, D_{\langle N,\lambda \rangle}, \psi_{c'}, \psi_1, \ldots, \psi_\ell\big)$.
10. Output $\mathsf{CT}' := (\mathsf{type} := 1, \mathsf{enc} := \psi)$.

5.0.5 Correctness

In the evaluation algorithm, the desired N-ary circuit C whose N inputs are over the domain $\{0,1\}^w$ is evaluated using the multi-key FHE scheme. Observe that C can be of arbitrary depth since the size of the resultant multi-key FHE ciphertext only depends on λ and N. We then encrypt this resulting ciphertext with $\mathcal{E}_{\mathsf{lABFHE}}$ in order to homomorphically evaluate the decryption circuit of $\mathcal{E}_{\mathsf{MKFHE}}$ using $\mathcal{E}_{\mathsf{lABFHE}}$. Consequently, we obtain a ciphertext whose size is independent of N as required by the compactness condition for ABHE.

5.1 Security

5.1.1 Semantic Security

Without loss of generality we assume that the message space $\mathcal{P}_{\mathcal{E}_{\mathsf{lABFHE}}}$ of $\mathcal{E}_{\mathsf{lABFHE}}$ is big enough to represent secret keys in $\mathcal{E}_{\mathsf{MKFHE}}$ and binary strings in $\mathcal{P}$.

Lemma 5.1.1. *If $\mathcal{E}_{\mathsf{lABFHE}}$ is an* IND-X-CPA*-secure leveled ABFHE scheme and $\mathcal{E}_{\mathsf{MKFHE}}$ is an* IND-CPA*-secure multi-key FHE scheme, then* bABFHE *is* IND-X-CPA *where $X \in \{\mathsf{sel}, \mathsf{AD}\}$.*

Proof. We prove the lemma by means of a hybrid argument.

Hybrid 0 IND-X-CPA game for bABFHE.

Hybrid 1 Same as Hybrid 0 except with one difference. Let $a^\star \in \mathbb{A}$ be the target attribute chosen by the adversary $\mathcal{A}$. The challenger uses a modified Encrypt algorithm to compute the leveled ABFHE ciphertext corresponding to a^* by replacing Step 4 with $\psi \leftarrow \mathcal{E}_{\mathsf{lABFHE}}.\mathsf{Encrypt}(\mathsf{PP}_{\mathcal{E}_{\mathsf{lABFHE}}}, a^*, 0^{|\mathsf{sk}|})$ where $0^{|\mathsf{sk}|}$ is a string of zeros whose length is the same as the multi-key FHE secret key generated in Step 3 of Encrypt. The algorithm is otherwise unchanged.

We claim that any poly-time $\mathcal{A}$ that can distinguish between Hybrid 0 and Hybrid 1 with a non-negligible advantage can break the IND-X-CPA security of $\mathcal{E}_{\mathsf{lABFHE}}$. An

adversary $\mathcal{B}$ that uses $\mathcal{A}$ proceeds as follows. When $\mathcal{A}$ chooses a target attribute $a^\star$, $\mathcal{B}$ generates a key-triple for $\mathcal{E}_{\mathsf{MKFHE}}$ i.e. it computes

$$(\mathsf{pk}, \mathsf{sk}, \mathsf{vk}) \leftarrow \mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Gen}(1^\lambda, 1^N).$$

Then it gives a^* to its challenger along with two messages $x_0 := \mathsf{sk}$ and $x_1 := 0^{|\mathsf{sk}|}$. Note that we assume for simplicity that both messages are in $\mathcal{P}_{\mathcal{E}_{\mathsf{lABFHE}}}$; if multiple messages (say k) are required then the usual hybrid argument can be applied which loses a factor of k. Subsequently, $\mathcal{B}$ embeds the challenge leveled ABFHE ciphertext as the ψ component of its own challenge ciphertext CT^*. It computes the remaining components of CT^* as in the $\mathsf{Encrypt}$ algorithm. If ψ encrypts x_0, then $\mathcal{B}$ perfectly simulates Hybrid 0. Otherwise, $\mathcal{B}$ perfectly simulates Hybrid 1. Note that secret key queries made by $\mathcal{A}$ can be perfectly simulated by $\mathcal{B}$. Thus, if $\mathcal{A}$ has a non-negligible advantage distinguishing between the hybrids, then $\mathcal{B}$ has a non-negligible advantage attacking the IND-X-CPA security of $\mathcal{E}_{\mathsf{lABFHE}}$.

For $i \in [w]$:

Hybrid 1 + i Same as Hybrid $1 + (i-1)$ with the exception that the challenger does not encrypt message bit $\mu_i^{(0)}$ or $\mu_i^{(1)}$ (using $\mathcal{E}_{\mathsf{MKFHE}}$) chosen by $\mathcal{A}$. Instead it encrypts some fixed message bit $\beta \in \{0,1\}$.

We now show that if $\mathcal{A}$ can efficiently distinguish between Hybrid $1+i$ and Hybrid $1+(i-1)$, then there is a PPT algorithm $\mathcal{G}$ that can use $\mathcal{A}$ to attack the IND-CPA security of $\mathcal{E}_{\mathsf{MKFHE}}$. Let pk and vk be the public key and evaluation key that $\mathcal{G}$ receives from its challenger. When $\mathcal{A}$ chooses $\mu^{(0)} \in \{0,1\}^w$ and $\mu^{(1)} \in \{0,1\}$, $\mathcal{G}$ simply gives $\mu_i^{(b)}$ and β to its IND-CPA challenger where b is the bit it uniformly samples in its simulation of the IND-X-CPA challenger. Let $c^\star$ be the challenge ciphertext it receives from the IND-CPA challenger. It sets $c_i \leftarrow c^*$ in the challenge ciphertext CT^*. If $c^\star$ encrypts $\mu_i^{(b)}$, then the view of $\mathcal{A}$ is identical to Hybrid $1+(i-1)$. Otherwise, the view of $\mathcal{A}$ is identical to Hybrid $1+i$. Therefore, a non-negligible advantage obtained by $\mathcal{A}$ implies a non-negligible advantage for $\mathcal{G}$ in the IND-CPA game, and thus contradicts the IND-CPA

security of $\mathcal{E}_{\mathsf{MKFHE}}$.

Finally observe that the adversary has a zero advantage in Hybrid $1+w$ because the challenge ciphertext contains no information about the challenger's bit.

□

5.1.2 EVAL-SIM Security

Recall the simulation-based security definition from Section 3.3.2, which we called EVAL-SIM security. In the following lemma, we show that bABFHE inherits EVAL-SIM security from $\mathcal{E}_{\mathsf{lABFHE}}$.

Lemma 5.1.2. *Let $\mathcal{E}_{\mathsf{MKFHE}}$ be an* IND-CPA *secure multi-key FHE scheme. Let $\mathcal{E}_{\mathsf{lABFHE}}$ be an X-*EVAL-SIM *secure ABHE scheme with $X \in \{\mathsf{sel}, \mathsf{AD}\}$. Then* bABFHE *is X-*EVAL-SIM *secure.*

Proof. By the hypothesized X-EVAL-SIM security of $\mathcal{E}_{\mathsf{lABFHE}}$, there exists a PPT simulator $\mathcal{S}_{\mathcal{E}_{\mathsf{lABFHE}}}$ such that for all PPT adversaries $\mathcal{A}_{\mathcal{E}_{\mathsf{lABFHE}}} := (\mathcal{A}_{\mathcal{E}_{\mathsf{lABFHE}},1}, \mathcal{A}_{\mathcal{E}_{\mathsf{lABFHE}},2})$ we have

$$|\mathsf{Pr}[\mathbf{Exp}^{\mathsf{REAL}}_{\mathcal{E}_{\mathsf{lABFHE}},\mathcal{A}_{\mathcal{E}_{\mathsf{lABFHE}}}} \to 1] - \mathsf{Pr}[\mathbf{Exp}^{\mathsf{IDEAL}}_{\mathcal{E}_{\mathsf{lABFHE}},\mathcal{A}_{\mathcal{E}_{\mathsf{lABFHE}}},\mathcal{S}_{\mathcal{E}_{\mathsf{lABFHE}}}} \to 1]| < \mathsf{negl}(\lambda). \quad (5.1.1)$$

Remark Note that in this proof we use the definition for adaptive EVAL-SIM security, which is slightly different to that for sel-EVAL-SIM security, but the argument holds analogously for the latter.

A simulator $\mathcal{S}$ can be constructed using $\mathcal{S}_{\mathcal{E}_{\mathsf{lABFHE}}}$ in order to achieve X-EVAL-SIM security for bABFHE. The simulator $\mathcal{S}$ runs as follows:

- $\boldsymbol{\mathcal{S}}(\mathsf{PP}, C, \{a_1, \ldots, a_{\mathscr{d}}\})$ with $\mathscr{d} \leq \mathscr{D}$, $a_1, \ldots, a_{\mathscr{d}} \in \mathbb{A}$ and $C \in \mathbb{C}$:
  1. Parse PP as $(\mathsf{PP}_{\mathcal{E}_{\mathsf{lABFHE}}}, \lambda, N, w)$.
  2. Let $D_{\langle N,\lambda\rangle}$ be the decryption circuit of $\mathcal{E}_{\mathsf{MKFHE}}$ for parameters N and λ.
  3. Output $\mathcal{S}_{\mathcal{E}_{\mathsf{lABFHE}}}(\mathsf{PP}_{\mathcal{E}_{\mathsf{lABFHE}}}, D_{\langle N,\lambda\rangle}, \{a_1, \ldots, a_{\mathscr{d}}\})$.

We claim that if there exists a PPT adversary $\mathcal{A} := (\mathcal{A}_1, \mathcal{A}_2)$ with a non-negligible advantage distinguishing the real distribution and ideal distribution for bABFHE (with respect to $\mathcal{S}$), then there exists a PPT adversary $\mathcal{A}_{\mathcal{E}_{\mathsf{lABFHE}}} := (\mathcal{A}_{\mathcal{E}_{\mathsf{lABFHE}},1}, \mathcal{A}_{\mathcal{E}_{\mathsf{lABFHE}},2})$ with a non-negligible advantage distinguishing the real distribution and ideal distribution for $\mathcal{E}_{\mathsf{lABFHE}}$ (with respect to $\mathcal{S}_{\mathcal{E}_{\mathsf{lABFHE}}}$). If this claim were to hold it would contradict the hypothesized X-EVAL-SIM security of $\mathcal{E}_{\mathsf{lABFHE}}$, which seals the lemma. To prove the claim, we show how to construct $(\mathcal{A}_{\mathcal{E}_{\mathsf{lABFHE}},1}, \mathcal{A}_{\mathcal{E}_{\mathsf{lABFHE}},2})$ from $(\mathcal{A}_1, \mathcal{A}_2)$. The algorithm $\mathcal{A}_{\mathcal{E}_{\mathsf{lABFHE}},1}$ is given as input the public parameters $\mathsf{PP}_{\mathcal{E}_{\mathsf{lABFHE}}}$ for $\mathcal{E}_{\mathsf{lABFHE}}$. We denote its key generation oracle by $\mathcal{O}_1$. It runs as follows.

1. Set $\mathsf{PP} := (\mathsf{PP}_{\mathcal{E}_{\mathsf{lABFHE}}}, \lambda, N, w)$ (the parameters N and w are fixed elsewhere).
2. Run $(C, (a_1, \mu_1), \ldots, (a_\ell, \mu_\ell), \mathsf{st}) \leftarrow \mathcal{A}_1^{\mathcal{O}_1}(\mathsf{PP})$.
3. For $i \in [\ell]$:
   (a) Parse μ_i as $(\mu_1^{(i)}, \ldots, \mu_w^{(i)}) \in \{0,1\}^w$.
   (b) $(\mathsf{pk}_i, \mathsf{sk}_i, \mathsf{vk}_i) \leftarrow \mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Gen}(1^\lambda, 1^N)$
   (c) $c_j^{(i)} \leftarrow \mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Encrypt}(\mathsf{pk}, \mu_j^{(i)})$ for $j \in [w]$.
4. Set $\mathit{d} \leftarrow |\{a_1, \ldots, a_\ell\}|$ (degree of composition).
5. $c' \leftarrow \mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Eval}(C, (c_1^{(1)}, \mathsf{vk}_1), \ldots, (c_w^{(1)}, \mathsf{vk}_1), \ldots, (c_1^{(\ell)}, \mathsf{vk}_\ell), \ldots, (c_w^{(\ell)}, \mathsf{vk}_\ell))$.
6. Let $D_{\langle N,\lambda\rangle}$ be the decryption circuit of $\mathcal{E}_{\mathsf{MKFHE}}$ for parameters N and λ.
7. Set $\mathsf{state} \leftarrow (\mathsf{st}, \mathsf{PP}, (\mathsf{vk}_1, (c_1^{(1)}, \ldots, c_w^{(1)})), \ldots, (\mathsf{vk}_\ell, (c_1^{(\ell)}, \ldots, c_w^{(\ell)})))$.
8. Output $(D_{\langle N,\lambda\rangle}, (a_1, c'), (a_1, \mathsf{sk}_1), \ldots, (a_\ell, \mathsf{sk}_\ell), \mathsf{state})$.

The algorithm $\mathcal{A}_{\mathcal{E}_{\mathsf{lABFHE}},2}$ is given as input the state state (generated in $\mathcal{A}_{\mathcal{E}_{\mathsf{lABFHE}},1}$), the evaluated ciphertext ψ' along with the $\ell + 1$ "input ciphertexts" (which we denote by $\psi_{c'}, \psi_1, \ldots, \psi_\ell$) and attributes $\{a_1, \ldots, a_{\mathit{d}}\}$. We denote its key generation oracle by $\mathcal{O}_2$. It runs as follows.

1. Parse state as $(\mathsf{st}, \mathsf{PP}, (\mathsf{vk}_1, (c_1^{(1)}, \ldots, c_w^{(1)})), \ldots, (\mathsf{vk}_\ell, (c_1^{(\ell)}, \ldots, c_w^{(\ell)})))$.
2. Parse PP as $(\mathsf{PP}_{\mathcal{E}_{\mathsf{lABFHE}}}, \lambda, N, w)$.
3. Generate bABFHE input ciphertext $\mathsf{CT}_i \leftarrow (\mathsf{type} := 0, \mathsf{enc} := (\psi_i, \mathsf{vk}_i, (c_1^{(i)}, \ldots, c_w^{(i)})))$ for $i \in [\ell]$.
4. Generate bABFHE evaluated ciphertext $\mathsf{CT}' \leftarrow (\mathsf{type} := 1, \mathsf{enc} := \psi')$.
5. Run $b \leftarrow \mathcal{A}_2^{\mathcal{O}_2}(\mathsf{st}, \mathsf{CT}', \mathsf{CT}_1, \ldots, \mathsf{CT}_\ell)$.
6. Output b.

If ψ' is generated with $\mathcal{E}_{\mathsf{lABFHE}}.\mathsf{Eval}$ (i.e. the real distribution) then CT' is distributed identically to the output of $\mathsf{bABFHE.Eval}$. On the other hand, if ψ' is generated with $\mathcal{S}_{\mathcal{E}_{\mathsf{lABFHE}}}$ (i.e. the ideal distribution), then CT' is distributed identically to $\mathcal{S}$. Therefore, a non-negligible advantage against bABFHE implies a non-negligible advantage against $\mathcal{E}_{\mathsf{lABFHE}}$.

□

5.2 Main Result

Theorem 5.2.1. *Let N be a positive integer. Let w be a positive integer. Let λ be a security parameter. Suppose there exists an* IND-CPA *secure multi-key FHE scheme $\mathcal{E}_{\mathsf{MKFHE}}$ whose decryption circuit has depth $\delta(N, \lambda)$. Suppose there exists a leveled ABFHE scheme $\mathcal{E}_{\mathsf{lABFHE}}$ that can compactly evaluate circuits of depth δ. Then there exists an ABHE scheme $\mathcal{E}$ (whose parameters $\mathcal{D}$ and $\mathcal{K}$ are the same as $\mathcal{E}_{\mathsf{lABFHE}}$) that can compactly evaluate all Boolean circuits in $\{(\{0,1\}^w)^N \to \{0,1\}^w\}$ i.e. the class of Boolean circuits of unbounded depth with N inputs over the domain $\{0,1\}^w$, such that*

1. *$\mathcal{E}$ is* IND-X-CPA *secure if $\mathcal{E}_{\mathsf{lABFHE}}$ is* IND-X-CPA *secure.*
2. *$\mathcal{E}$ is X-*EVAL-SIM *secure if $\mathcal{E}_{\mathsf{lABFHE}}$ is X-*EVAL-SIM *secure.*

for $X \in \{\mathsf{sel}, \mathsf{AD}\}$.

Proof. Instantiating our scheme bABFHE from Section 5.0.3 with the multi-key FHE scheme $\mathcal{E}_{\mathsf{MKFHE}}$ and the ABHE scheme $\mathcal{E}_{\mathsf{lABFHE}}$, the theorem follows by appealing to Lemma 5.1.1 (IND-X-CPA security) and Lemma 5.1.2 (X-EVAL-SIM security). □

Corollary 5.2.1. *Let N be a positive integer. Assuming the hardness of LWE, there exists a* IND-sel-CPA *secure ABFHE that can compactly evaluate circuits with N inputs.*

Proof. We can instantiate the multi-key FHE scheme in our construction with the multi-key FHE from the next chapter, whose security is based on LWE. Furthermore we can instantiate the leveled ABFHE in our construction with the leveled ABFHE of Gentry, Sahai and Waters [98], which is shown to be selectively secure under LWE. □

5.2.1 Discussion

We could instantiate $\mathcal{E}_{\mathsf{MKFHE}}$ with the multi-key FHE scheme of López-Alt, Tromer and Vaikuntanathan [135]. However its decryption circuit has depth $O(\log^2(N \cdot \lambda))$ as opposed to $O(\log(N \cdot \lambda))$ for our multi-key FHE scheme from the next chapter, which means that the leveled ABFHE scheme must be set up to accomodate more levels, which in turn causes the parameters to blow up. Suppose we set N to be a large value so as not to practically limit the number of inputs to a circuit. As a result, N dominates λ. Therefore we need the leveled ABFHE to evaluate roughly $O(\log N)$ levels. Concretely, suppose we were to pick a very large value of N, say $N = 2^{32}$, then we need a leveled ABFHE that can evaluate on the order of 32 levels.

5.3 Application Scenario

Recall our medical records scenario from the introduction (Section 1.2.0.1). Three senders encrypt sensitive medical data under appropriate attributes and send it to an

evaluator for computation. Sender 1 encrypts her data m_1 under attribute "CARDIOLOGY". Sender 2 encrypts his data m_2 with attribute "MATERNITY". Sender 3 encrypts her data m_3 with attribute "CARDIOLOGY". We now show how this example ties in to the contributions in this chapter. The number of independent senders in this case is 3. In reality the number of independent senders may be much larger. Suppose the maximum number of expected independent senders is (say) $N = 2^{32}$. In other words, no more than 2^32 medical researchers or doctors ever contribute medical data to the same computation.

Let us instantiate our construction in this chapter with the multikey FHE from the next chapter. We can instantiate $\mathcal{E}_{\mathsf{lABFHE}}$ with the leveled multi-identity scheme from the next chapter. We need to set up this scheme to handle $O(\log N)$ levels which is on the order of 32 for our example. Note that this leveled scheme is identity-based and hence only allows simple access policies (such as disjunctive policies). Our construction can evaluate circuits of arbitrary depth, supporting up to 2^{32} independent senders. This accommodates our scenario above for the three independent senders, assuming the ABHE $\mathcal{E}_{\mathsf{lABFHE}}$ accommodates the access policies in the scenario. In particular, the receiver has an access policy f with $f(\text{"CARDIOLOGY"}) = 1$ and $f(\text{"MATERNITY"}) = 1$. In other words, the receiver can decrypt a ciphertext with attribute "CARDIOLOGY" or attribute "MATERNITY" (or both in the case of the result of an evaluation). Note that the computation carried out by the evaluator on the data contributed by the senders may be arbitrary.

5.4 Summary

In this chapter, we proposed a black-box construction of ABFHE with support for circuits with a bounded number of inputs N. Our construction relies on multi-key FHE and leveled ABFHE. This overcomes roadblocks to achieving fully homomorphic encryption in the attribute-based setting. Our construction can evaluate circuits of arbitrary depth,

but has a limit on the arity of circuits supported i.e. the number of inputs. If the bound on the number of inputs is satisfactory, then our scheme can evaluate all circuits that arise in practice.

Chapter 6

Multi-Identity Leveled Homomorphic Encryption

Our main result in the previous chapter tells us that an ABFHE capable of evaluating circuits with a bounded number of inputs can be constructed from multi-key FHE and leveled ABFHE. Recall that a leveled FHE scheme allows an evaluator to evaluate a circuit of an a priori bounded depth L. The parameter L must be specified in advance when generating the public parameters of the scheme, whose size may depend on L. Furthermore, a leveled homomorphic scheme supports any value of L, but the size of the resulting public parameters, ciphertexts and secret keys may depend polynomially on L. The time complexity of all algorithms may depend polynomially on L. In contrast, a "pure" fully homomorphic encryption scheme allows circuits of unlimited depth to be evaluated. However, for many applications in practice, a leveled scheme is adequate. So besides serving as a building block for our construction in the previous chapter, a leveled ABFHE is very useful in its own right. In fact, at the expense of a limited circuit depth (which as aforementioned, can be chosen to satisfy application requirements, and may indeed suffice for all evaluations), a leveled ABFHE overcomes the bound on arity that our construction in the previous chapter suffers from. In summary, there are two

primary reasons to explore a concrete construction of leveled ABFHE: (1). to instantiate our construction in the previous chapter; and (2). as a standalone primitive with the homomorphic capacity to evaluate circuits of bounded depth and unbounded arity.

At Crypto 2013, Gentry, Sahai and Waters presented the first *leveled* identity-based fully homomorphic encryption (IBFHE) scheme [98] and the first *leveled* attribute-based fully homomorphic encryption (ABFHE) scheme that are secure under the hardness of the Learning with Errors (LWE) problem, a problem introduced by Regev [161] that has received considerable attention in cryptography due to a known worst-case reduction to a hard lattice problem.

Gentry, Sahai and Waters described a compiler [98], which we call the GSW compiler, to transform an LWE-based IBE satisfying certain properties into a leveled IBFHE. They showed that all known LWE-based IBE schemes are compatible with their compiler. However, the GSW compiler only works in the *single-identity* setting. In other words, the resulting IBFHE can only evaluate on ciphertexts created with the same identity.

Gentry, Sahai and Waters also described a compiler for leveled ABFHE that works in the *single-attribute* setting. Unlike their IBFHE compiler, their ABFHE compiler is only compatible with certain LWE-based ABE schemes. One such scheme is a slight variant of the circuit-based ABE of Gorbunov et al. [109], which Gentry, Sahai and Waters show can be compiled into a *single-attribute* leveled ABFHE. Extending this compiler to work in the *multi-attribute* setting appears to be highly non-trivial.

In this chapter we present a compiler for multi-identity leveled IBFHE, and we give an instance of an IBE scheme that can be successfully compiled into a multi-identity leveled IBFHE that is provably secure under LWE, albeit in the random oracle model. This is the first multi-identity leveled IBFHE to the best of our knowledge. We also show that our techniques fall short of working with the attribute-based setting; constructing a multi-attribute leveled ABFHE from LWE is an important question for future work, as is removing the random oracle from our own multi-identity construction. We are unable to prove that our multi-identity leveled IBFHE in this chapter is EVAL-SIM secure; this

is also an open problem for future work. Note that our multi-identity leveled IBFHE is 1-hop homomorphic insofar as after evaluation is complete, no further homomorphic evaluation can be carried out; removing this limitation is another goal for future work.

Furthermore, in this chapter, we present a multi-key FHE scheme from the Learning with Errors (LWE) problem. Our multi-key FHE has several advantages over the multi-key FHE scheme of López-Alt, Tromer and Vaikuntanathan [135]. Firstly, it does not rely on non-standard assumptions (the scheme from [135] relies on the Decisional Small Polynomial Ratio (DSPR) assumption). Secondly, its decryption circuit is of depth $O(\log N)$ (where N is the number of keys tolerated) which means it is more suitable to instantiate our construction in the previous chapter. Finally it admits a one-round distributed decryption protocol as shown by Mukherjee and Wichs [145]. In addition, Mukherjee and Wichs [145] used our multi-key FHE to achieve 2-round multi-party computation from LWE.

Remark Technically speaking, the GSW compiler does not produce a *leveled* IBFHE in the standard sense, because the size of its evaluated ciphertexts are not independent of L. This is not mentioned explicitly in [98] and the authors still refer to their identity-based construction as "leveled". Since our compiler for multi-identity leveled IBFHE is built on the GSW single-identity compiler, our compiler suffers the same problem. For the moment, we overlook this fact and return to it in Section 6.6.

6.1 Multi-Identity Leveled IBFHE

Definition 6.1.1. *A Multi-Identity Leveled IBFHE scheme is defined with respect to a message space $\mathcal{P}$, an identity space $\mathcal{I}$, a class of circuits $\mathbb{C} \subseteq \mathcal{P}^* \to \mathcal{P}$ and ciphertext space $\mathcal{C}$. A Multi-Identity Leveled IBFHE scheme is a tuple of PPT algorithms* $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt}, \mathsf{Eval})$ *defined as follows:*

- $\mathsf{Setup}(1^\lambda, L, \mathcal{D})$*:*

  On input (in unary) a security parameter λ, a number of levels L (circuit depth

to support) and the number of distinct identities $\mathcal{D}$ that can be tolerated in an evaluation, generate public parameters PP *and a master secret key* MSK*. Output* $(\mathsf{PP}, \mathsf{MSK})$.

- $\mathsf{KeyGen}(\mathsf{MSK}, \mathsf{id})$*:*
  On input master secret key MSK *and an identity* id*: derive and output a secret key* $\mathsf{sk}_{\mathsf{id}}$ *for identity* id.

- $\mathsf{Encrypt}(\mathsf{PP}, \mathsf{id}, \mu)$*:*
  On input public parameters PP*, an identity* id*, and a message* $\mu \in \mathcal{P}$*, output a ciphertext* $c \in \mathcal{C}$ *that encrypts* μ *under identity* id.

- $\mathsf{Decrypt}(\mathsf{sk}_{\mathsf{id}_1}, \ldots, \mathsf{sk}_{\mathsf{id}_k}, c)$*:*
  On input $k \leq \mathcal{D}$ *secret keys* $\mathsf{sk}_{\mathsf{id}_1}, \ldots, \mathsf{sk}_{\mathsf{id}_k}$ *for (resp.) identities* $\mathsf{id}_1, \ldots, \mathsf{id}_k$ *and a ciphertext* $c \in \mathcal{C}$*, output* $\mu' \in \mathcal{P}$ *if* c *is a valid encryption under identities* $\mathsf{id}_1, \ldots, \mathsf{id}_k$*; output a failure symbol* $\perp$ *otherwise.*

- $\mathsf{Eval}(\mathsf{PP}, C, c_1, \ldots, c_\ell)$*: On input public parameters* PP*, a circuit* $C \in \mathbb{C}$ *and ciphertexts* $c_1, \ldots, c_\ell \in \mathcal{C}$*, output an* evaluated ciphertext $c' \in \mathcal{C}$.

More precisely, the scheme is required to satisfy the following properties:

- *Let* L *and* $\mathcal{D}$ *be positive integers. Over all choices of* $(\mathsf{PP}, \mathsf{MSK}) \leftarrow \mathsf{Setup}(1^\lambda, L, \mathcal{D})$, $C : \mathcal{P}^\ell \to \mathcal{P} \in \{C \in \mathbb{C} : \mathsf{depth}(C) \leq L\}$*, every* $d \leq \mathcal{D}$*,* $\mathsf{id}_1, \ldots, \mathsf{id}_\ell \in \mathcal{I}$ *s.t* $|\{\mathsf{id}_1, \ldots, \mathsf{id}_\ell\}| = d$*,* $\mu_1, \ldots, \mu_\ell \in \mathcal{P}$*,* $c_i \leftarrow \mathsf{Encrypt}(\mathsf{PP}, \mathsf{id}_i, \mu_i)$ *for* $i \in [\ell]$*, and* $c' \leftarrow \mathsf{Eval}(\mathsf{PP}, C, c_1, \ldots, c_\ell)$*:*

  - ***Correctness***

    $$\mathsf{Decrypt}(\mathsf{sk}_1, \ldots, \mathsf{sk}_d, c') = C(\mu_1, \ldots, \mu_\ell) \tag{6.1.1}$$

    for any $\mathsf{sk}_i \leftarrow \mathsf{KeyGen}(\mathsf{MSK}, \mathsf{id}_i)$ *for* $i \in [d]$

  - ***Compactness***

    $$|c'| \leq \mathsf{poly}(\lambda, d) \tag{6.1.2}$$

The time complexity of all algorithms may depend polynomially on L and $\mathcal{D}$, in addition to λ. The size of the public parameters, "fresh" ciphertexts and secret keys may depend polynomially on L and $\mathcal{D}$, in addition to λ. The size of an evaluated ciphertext depends only on λ and the degree of composition $d \leq \mathcal{D}$ of the evaluation.

One relaxation of Definition 6.1.1 is to weaken the compactness condition to allow the size of an evaluated ciphertext c' to also depend on L. This is in line with the notion of "leveled" IBFHE achieved by Gentry, Sahai and Waters, as mentioned earlier. Another relaxation of Definition 6.1.1 is to further weaken the compactness condition by allowing the size of c' to depend on the maximum degree of composition, $\mathcal{D}$, instead of the *actual* degree of composition, d. This corresponds more closely to the compactness condition of multikey FHE (see Section 5.0.1.1).

For the remainder of this chapter, the term *multi-identity leveled IBFHE* should be understood to mean the primitive described by Definition 6.1.1 with its compactness condition relaxed to

$$|c'| \leq \mathsf{poly}(\lambda, L, \mathcal{D}). \tag{6.1.3}$$

Our central result in this chapter is informally summarized in the following theorem statement. The theorem is formally stated and proven later in the chapter.

Theorem 6.1.1 (Informal)**.** *There exists a multi-identity leveled IBFHE scheme that is selectively secure under the Learning With Errors problem in the random oracle model.*

6.1.0.1 Multi-Key FHE

Our compiler for multi-identity IBFHE also works in the public-key setting. As a result, we can obtain a multi-key FHE [135] from LWE in the standard model. In fact, multi-identity IBFHE can be seen as an identity-based analog to multi-key FHE. The syntax of multi-key FHE from [135] entails a parameter N, which specifies the maximum number of independent keys tolerated in an evaluation. The size of the parameters and ciphertexts are allowed to depend polynomially on N. Note that N is fixed and specified in advance

of generating the scheme's parameters. To the best of our knowledge, our multi-key FHE scheme is the first such scheme that is based on a well-established problem such as LWE; the construction from [135] relies on a non-standard computational assumption referred to therein as the Decisional Small Polynomial Ratio (DSPR) assumption. Our scheme positively answers the question raised in [135] as to whether other multi-key FHE schemes exist supporting polynomially-sized N. Another advantage of our multi-key FHE is that its decryption circuit has depth $O(\log \lambda \cdot N)$ as opposed to $O(\log^2 \lambda \cdot N)$ in the scheme from [135]; this means that to invoke Theorem 5.0.1 from Chapter 5, one only needs an ABHE for circuits in NC^1, as opposed to NC^2.

6.1.1 Our Approach: Intuition

We now give an informal sketch of our approach to achieving multi-identity IBFHE. This section is intended to provide an intuition and many of the details are deferred to later in the chapter.

Remark Like [98], we omit the qualifier "leveled" for the rest of this chapter since we focus only on leveled (IB)FHE in this chapter.

We remind the reader that a matrix $\mathbf{M}$ is denoted by an uppercase symbol written in boldface, and a vector $\vec{\mathbf{v}}$ is denoted by a lowercase symbol written in boldface. The i-th element of $\vec{\mathbf{v}}$ is denoted by v_i. The inner product of two vectors $\vec{\mathbf{a}}, \vec{\mathbf{b}} \in \mathbb{Z}_q^n$ for some dimension n is written as $\langle a, b \rangle$.

6.1.1.1 GSW single-identity IBFHE

We start by briefly discussing the homomorphic properties of the GSW IBFHE schemes from [98]. This discussion applies to *any* IBFHE constructed with their compiler. A ciphertext in their scheme is an $N \times N$ matrix $\mathbf{C}$ over $\mathbb{Z}_q$ whose entries are "small" with respect to q. Note that N is a parameter that will be discussed later. A secret key for an identity id is an N-dimensional vector $\vec{\mathbf{v}_{\mathsf{id}}} \in \mathbb{Z}_q^N$ with at least one "large" coefficient;

let this coefficient (say the i-th one) be $v_{\mathsf{id},i} \in \mathbb{Z}_q$. The scheme can encrypt "small" messages μ; an example to keep in mind is a message in $\{0,1\}$. We say the matrix $\mathbf{C}$ *encrypts* μ under identity id if $\mathbf{C} \cdot \vec{\mathbf{v}_{\mathsf{id}}} = \mu \cdot \vec{\mathbf{v}_{\mathsf{id}}} + \vec{\mathbf{e}} \in \mathbb{Z}_q^N$ where $\vec{\mathbf{e}}$ is a "small" noise vector (i.e. roughly speaking, each of its coefficients is much less than q). As such, $\vec{\mathbf{v}_{\mathsf{id}}}$ is an *approximate* eigenvector for the matrix $\mathbf{C}$ with eigenvalue μ.

Homomorphic Operations

Suppose $\mathbf{C_1}$ and $\mathbf{C_2}$ encrypt μ_1 and μ_2 respectively; that is, $\mathbf{C_j} \cdot \vec{\mathbf{v}_{\mathsf{id}}} = \mu_j \cdot \vec{\mathbf{v}_{\mathsf{id}}} + \vec{\mathbf{e}_\mathbf{j}}$ for $j \in \{1,2\}$. An additive homomorphism is supported. Let $\mathbf{C}^+ = \mathbf{C_1} + \mathbf{C_2}$. Then we have $\mathbf{C}^+ \cdot \vec{\mathbf{v}_{\mathsf{id}}} = (\mu_1 + \mu_2) \cdot \vec{\mathbf{v}_{\mathsf{id}}} + (\vec{\mathbf{e_1}} + \vec{\mathbf{e_2}})$. The error only grows slightly here, and as long as it remains "small", we can recover the sum $(\mu_1 + \mu_2)$. A multiplicative homomorphism is also supported. Let $\mathbf{C}^\times = \mathbf{C_1} \cdot \mathbf{C_2}$. Then we have

$$\begin{aligned}
\mathbf{C}^\times \cdot \vec{\mathbf{v}_{\mathsf{id}}} &= \mathbf{C_1} \cdot (\mu_2 \cdot \vec{\mathbf{v}_{\mathsf{id}}} + \vec{\mathbf{e_2}}) \\
&= \mu_2 \cdot (\mu_1 \cdot \vec{\mathbf{v}_{\mathsf{id}}} + \vec{\mathbf{e_1}}) + \mathbf{C_1} \cdot \vec{\mathbf{e_2}} \\
&= \mu_1 \cdot \mu_2 \cdot \vec{\mathbf{v}_{\mathsf{id}}} + \mu_2 \cdot \vec{\mathbf{e_1}} + \mathbf{C_1} \cdot \vec{\mathbf{e_2}} \\
&= \mu_1 \cdot \mu_2 \cdot \vec{\mathbf{v}_{\mathsf{id}}} + \text{"small"}.
\end{aligned}$$

6.1.1.2 Different Identities

Now we give a flavor of how our multi-identity scheme operates. Suppose $\mathbf{C_1}$ encrypts μ_1 under identity id_1 and $\mathbf{C_2}$ encrypts μ_2 under identity id_2. Let $\vec{\mathbf{v_1}}$ and $\vec{\mathbf{v_2}}$ be the secret key vectors for id_1 and id_2 respectively. It holds that $\mathbf{C_1} \cdot \vec{\mathbf{v_1}} = \mu_1 \cdot \vec{\mathbf{v_1}} + \vec{\mathbf{e_1}}$ and $\mathbf{C_2} \cdot \vec{\mathbf{v_2}} = \mu_2 \cdot \vec{\mathbf{v_2}} + \vec{\mathbf{e_2}}$ where $\vec{\mathbf{e_1}}, \vec{\mathbf{e_2}} \in \mathbb{Z}_q^N$ are short vectors.

We would like to be able to perform homomorphic computation on both $\mathbf{C_1}$ and $\mathbf{C_2}$ together; that is, use them both as inputs to the same circuit. Here we denote the circuit by $C \in \mathbb{C}$. Suppose we could produce a resulting $2N \times 2N$ ciphertext matrix $\hat{\mathbf{C}}' \in \mathbb{Z}_q^{2N \times 2N}$ that encrypts $\mu' = C(\mu_1, \mu_2)$. More precisely, suppose that

$$\hat{\mathbf{C}}' \cdot \begin{bmatrix} \vec{\mathbf{v_1}} \\ \vec{\mathbf{v_2}} \end{bmatrix} = \mu' \cdot \begin{bmatrix} \vec{\mathbf{v_1}} \\ \vec{\mathbf{v_2}} \end{bmatrix} + \vec{\mathbf{e}'}$$

where $\vec{\mathbf{e}'}$ is "short". Note that the size of $\hat{\mathbf{C}}'$ just depends (polynomially) on the number of distinct identities (2 in this example).

Let $\vec{\mathbf{v}} \in \mathbb{Z}_q^{2N}$ be the vertical concatenation of the two vectors $\vec{\mathbf{v_1}}$ and $\vec{\mathbf{v_2}}$. We could exploit the homomorphic properties described above to obtain $\hat{\mathbf{C}}'$ if we could somehow transform $\mathbf{C_1}$ and $\mathbf{C_2}$ into $2N \times 2N$ matrices $\hat{\mathbf{C}}_\mathbf{1}$ and $\hat{\mathbf{C}}_\mathbf{2}$ respectively such that $\hat{\mathbf{C}}_\mathbf{j} \cdot \vec{\mathbf{v}} = \mu_j \cdot \vec{\mathbf{v}} +$ "small" for $j \in \{1, 2\}$. Technically this transformation turns out to be difficult; we show how to abstractly accomplish it in Section 6.3 and concretely in Section 6.4.

6.2 The Gentry, Sahai and Waters (GSW) Leveled IBFHE

6.2.1 Learning with Errors

The Learning with Errors (LWE) problem was introduced by Regev [161]. The goal of the computational form of the LWE problem is to determine an n-dimensional secret vector $\vec{\mathbf{s}} \in \mathbb{Z}_q^n$ given a polynomial number of samples $(\vec{\mathbf{a_i}}, b_i) \in \mathbb{Z}_q^{n+1}$ where $\vec{\mathbf{a_i}}$ is uniform over $\mathbb{Z}_q^n$ and $b_i \leftarrow \langle \vec{\mathbf{a_i}}, \vec{\mathbf{s}} \rangle + e_i \in \mathbb{Z}_q$ is the inner product of $\vec{\mathbf{a_i}}$ and $\vec{\mathbf{s_i}}$ perturbed by a small *error* $e_i \in \mathbb{Z}$ that is sampled from a distribution χ over $\mathbb{Z}$. We call the distribution χ an error distribution (or noise distribution). The decision variant of the problem is to distinguish such samples $(\vec{\mathbf{a_i}}, b_i) \in \mathbb{Z}_q^{n+1}$ from uniform vectors over $\mathbb{Z}_q^{n+1}$. The decisional variant is more commonly used in cryptography, and is most relevant to our own work. As a result, without further qualification, when we refer to LWE throughout this thesis we are referring to the decisional variant.

Definition 6.2.1 ((Decisional) Learning with Errors (LWE) Problem [161])**.** *Let λ be a security parameter. For parameters $n = n(\lambda)$, $q = q(\lambda) \geq 2$, and a distribution $\chi = \chi(\lambda)$ over $\mathbb{Z}$, the $LWE_{n,q,\chi}$ problem is to distinguish the following distributions:*

- ***Distribution 0**: The i-th sample $(\vec{\mathbf{a_i}}, b_i) \in \mathbb{Z}_q^{n+1}$ is computed by uniformly sampling $\vec{\mathbf{a_i}} \xleftarrow{\$} \mathbb{Z}_q^n$ and $b_i \xleftarrow{\$} \mathbb{Z}_q$.*
- ***Distribution 1**: Generate uniform vector $\vec{\mathbf{s}} \xleftarrow{\$} \mathbb{Z}_q^n$. The i-th sample $(\vec{\mathbf{a_i}}, b_i) \in \mathbb{Z}_q^{n+1}$*

is computed by uniformly sampling $\vec{\mathbf{a_i}} \xleftarrow{\$} \mathbb{Z}_q^n$*, sampling an error value* $e_i \xleftarrow{\$} \chi$ *and computing* $b_i \leftarrow \langle \vec{\mathbf{a_i}}, \vec{\mathbf{s}} \rangle + e_i$*.*

Definition 6.2.2 (B-bounded distributions (Definition 2 [98]))**.** *A distribution ensemble* $\{D_n\}_{n \in \mathbb{N}}$*, supported over the integers, is called* B*-bounded if*

$$\mathsf{Pr}_{e \xleftarrow{\$} D_n}[|e| > B] = \mathsf{negl}(n).$$

Definition 6.2.3 (GapSVP_γ)**.** *Let* n *be a lattice dimension, and let* d *be a real number. Then* $GapSVP_\gamma$ *is the problem of deciding whether an* n*-dimensional lattice has a nonzero vector shorter than* d *(an algorithm should accept in this case) or no nonzero vector shorter than* $\gamma(n) \cdot d$ *(an algorithm should reject in this case); an algorithm is allowed to error otherwise.*

Theorem 6.2.1 (Theorem 1 [98])**.** *Let* $q = q(n) \in \mathbb{N}$ *be either a prime power or a product of small (*$\mathsf{poly}(n)$*) distinct primes, and let* $B \geq \omega(\log n) \cdot \sqrt{n}$*. Then there exists an efficient sampleable* B*-bounded distribution* χ *such that if there is an efficient algorithm that solves the average-case* $LWE_{n,q,\chi}$ *problem, then:*

- *There is an efficient quantum algorithm that solves* $GapSVP_{\tilde{O}(nq/B)}$ *on any* n*-dimensional lattice.*
- *If* $q > \tilde{O}(2^{n/2})$*, then there is an efficient classical algorithm for* $GapSVP_{\tilde{O}(nq/B)}$ *on any* n*-dimensional lattice.*

6.2.2 GSW Approximate Eigenvector Cryptosystem

Recall our brief overview of the GSW IBFHE construction earlier from Section 6.1.1.1. The following exposition describes this construction in more detail. Note that the public-key GSW scheme is similar to the identity-based variant. As such, to simplify the notation, the following discussion deals with the public-key setting, but the ideas apply to both.

Definition 6.2.4 (Section 1.3.2 from [98]). ***B*-boundedness:** *Let $B < q$ be an integer. Let $\mathbf{C}$ be a ciphertext matrix that encrypts μ. Let $\vec{\mathbf{v}}$ be a secret key vector such that $\mathbf{C} \cdot \vec{\mathbf{v}} = \mu \cdot \vec{\mathbf{v}} + \vec{\mathbf{e}}$. Then $\mathbf{C}$ is said to be B-bounded (with respect to $\vec{\mathbf{v}}$) if the magnitude of μ is at most B, the magnitude of all the entries of $\mathbf{C}$ is at most B, and $|||\vec{\mathbf{e}}|||_\infty \leq B$.*

Let $\mathbf{C_1}$ and $\mathbf{C_2}$ be two B-bounded ciphertext matrices. Then $\mathbf{C}^+ = \mathbf{C_1} + \mathbf{C_2}$ is $2B$-bounded. Furthermore, $\mathbf{C}^\times = \mathbf{C_1} \cdot \mathbf{C_2}$ is $(N+1)^{B^2}$-bounded. As the authors of [98] point out, the error grows worse than B^{2^L}, where L is the multiplicative depth of a circuit being evaluated. The modulus q can be chosen to exceed this bound, but we must be careful to ensure that the ratio q/B is at most subexponential in N to guarantee security (see Theorem 6.2.1). Hence, only circuits of logarithmic multiplicative depth can be evaluated. This gives us a somewhat-homomorphic scheme.

To evaluate deeper circuits, namely those with polynomial multiplicative depth, we must keep the entries of the ciphertext matrices "small". To achieve this, Gentry, Sahai and Waters propose a technique called *flattening*. Consider the following definition.

Definition 6.2.5 (Section 1.3.3 from [98]). ***B*-strong-boundedness:** *Let $B < q$ be an integer. Let $\mathbf{C}$ be a ciphertext matrix that encrypts μ. Let $\vec{\mathbf{v}}$ be a secret key vector such that $\mathbf{C} \cdot \vec{\mathbf{v}} = \mu \cdot \vec{\mathbf{v}} + \vec{\mathbf{e}}$. Then $\mathbf{C}$ is said to be B-strongly-bounded (with respect to $\vec{\mathbf{v}}$) if the magnitude of μ is at most 1, the magnitude of all the entries of $\mathbf{C}$ is at most 1, and $|||\vec{\mathbf{e}}|||_\infty \leq B$.*

An example of a B-strongly-bounded ciphertext is a matrix $\mathbf{C}$ with binary entries that encrypts a plaintext bit $\mu \in \{0, 1\}$, provided the coefficients of its corresponding $\vec{\mathbf{e}}$ vector have magnitude at most B. Let $\mathbf{C_1}$ and $\mathbf{C_2}$ be ciphertext matrices that encrypt $\mu_1 \in \{0, 1\}$ and $\mu_2 \in \{0, 1\}$ respectively. A NAND gate can be evaluated on two ciphertexts $\mathbf{C_1}$ and $\mathbf{C_2}$ as follows:

$$\mathbf{C_3} = \mathbf{I_N} - \mathbf{C_1} \cdot \mathbf{C_2},$$

where $\mathbf{I_N}$ is the $N \times N$ identity matrix. The matrix $\mathbf{C_3}$ encrypts $\mu_1 \barwedge \mu_2 \in \{0, 1\}$. Now if $\mathbf{C_1}$ and $\mathbf{C_2}$ are B-strongly-bounded, then the coefficients of $\mathbf{C_3}$'s error vector have

magnitude at most $(N+1)B$, which is in contrast to $(N+1)B^2$ above where $\mathbf{C_1}$ and $\mathbf{C_2}$ were just B-bounded. Suppose there were some way to preserve strong-boundedness in $\mathbf{C_3}$ (i.e. to ensure the magnitude of its entries remained at most 1). Then it would be the case that $\mathbf{C_3}$ is $(N+1)B$-strongly-bounded. As a result, the error level would grow to at most $(N+1)^L B$ when evaluating a circuit of NAND gates of depth L. Therefore it would be possible to evaluate circuits of polynomial depth by letting q/B be subexponential. However, how can we preserve strong-boundedness? It is necessary to introduce some basic operations to help describe how strong boundedness is preserved. These operations serve as useful tools for our own constructions later.

6.2.2.1 Basic Operations

Let $\ell_q = \lfloor \lg q \rfloor + 1$. Let $\vec{\mathbf{v}} \in \mathbb{Z}_q^{m'}$ be a vector of some dimension m' over $\mathbb{Z}_q$. Let $N = m' \cdot \ell_q$.

- $\mathbf{BitDecomp}(\vec{\mathbf{v}})$: We define an algorithm BitDecomp that takes as input a vector $\vec{\mathbf{v}} \in \mathbb{Z}_q^{m'}$ and outputs an N-dimensional vector $(v_{1,0}, \ldots, v_{1,\ell_q-1}, \ldots, v_{k,0}, \ldots, v_{k,\ell_q-1})$ where $v_{i,j}$ is the j-th bit in v_i's binary representation (ordered from least significant to most significant).

- $\mathbf{BitDecomp^{-1}}(\vec{\mathbf{v'}})$: We define an "inverse" algorithm BitDecomp$'$ that takes an N-dimensional vector $\vec{\mathbf{v'}} = (v'_{1,0}, \ldots, v'_{1,\ell_q-1}, \ldots, v'_{k,0}, \ldots, v'_{k,\ell_q-1})$, and outputs a m'-dimensional vector $(\sum_{j=0}^{\ell_q-1} 2^j \cdot v'_{1,j}, \ldots, \sum_{j=0}^{\ell_q-1} 2^j \cdot v'_{k,j})$. Note that the input vector $\vec{\mathbf{v'}}$ need not be binary, the algorithm is well-defined for any input vector in $\mathbb{Z}_q^N$.

- $\mathbf{Flatten}(\vec{\mathbf{v'}})$: The algorithm Flatten takes as input an N-dimensional vector $\vec{\mathbf{v'}} \in \mathbb{Z}_q^N$ and outputs an N-dimensional binary vector $\mathsf{BitDecomp}(\mathsf{BitDecomp}'(\vec{\mathbf{v'}}) \in \{0,1\}^N$.

- $\mathbf{Powersof2}(\vec{\mathbf{v}})$: The algorithm Powersof2 takes a m'-dimensional vector $\vec{\mathbf{v}} \in \mathbb{Z}_q^{m'}$ and outputs an N-dimensional vector $(v_1, 2v_1, \ldots, 2^{\ell_q-1}v_1, \ldots, v_k, 2v_k, \ldots, 2^{\ell_q-1}v_k)$.

We also define $\mathsf{BitDecomp}$, $\mathsf{BitDecomp}'$ and $\mathsf{Flatten}$ for matrix inputs; in this case, the respective algorithm is applied to each row independently.

We restate the following straightforward facts from [98] (Section 1.3.3): Let $\vec{\mathbf{a}}, \vec{\mathbf{b}} \in \mathbb{Z}_q^{m'}$ be m'-dimensional vectors, and let $\vec{\mathbf{a'}} \in \mathbb{Z}_q^N$ be an N-dimensional vector:

- $\langle \mathsf{BitDecomp}(\vec{\mathbf{a}}), \mathsf{Powersof2}(\vec{\mathbf{b}}) \rangle = \langle \vec{\mathbf{a}}, \vec{\mathbf{b}} \rangle$.
- $\langle \vec{\mathbf{a'}}, \mathsf{Powersof2}(\vec{\mathbf{b}}) \rangle = \langle \mathsf{BitDecomp}^{-1}(\vec{\mathbf{a'}}), \vec{\mathbf{b}} \rangle = \langle \mathsf{Flatten}(\vec{\mathbf{a'}}), \mathsf{Powersof2}(\vec{\mathbf{b}}) \rangle$.

6.2.2.2 Flattening

With the help of $\mathsf{BitDecomp}$, $\mathsf{BitDecomp}^{-1}$, $\mathsf{Powersof2}$ and $\mathsf{Flatten}$, we can tackle the problem of preserving strong boundedness after a NAND operation. In order to make the coefficients of $\mathbf{C_3}$ above have magnitude at most 1, Gentry, Sahai and Waters propose to apply $\mathsf{Flatten}$ to the matrix $\mathbf{C_3}$. Thus, we compute $\mathbf{C}^{\mathsf{NAND}} \leftarrow \mathsf{Flatten}(\mathbf{C_3})$ to produce the output ciphertext of the NAND gate. Now for this to work, the vector $\vec{\mathbf{v}}$ must have a special form. More precisely, $\vec{\mathbf{v}}$ is computed as $\mathsf{Powersof2}(\vec{\mathbf{s}}) \in \mathbb{Z}_q^N$ for some secret key vector $\vec{\mathbf{s}} \in \mathbb{Z}_q^{m'}$ for some m'. Furthermore, the parameter N is defined as $N = m' \cdot \ell_q$, where $\ell_q = \lfloor \lg q \rfloor + 1$. With this form of secret key vector $\vec{\mathbf{v}}$, it holds that $\mathsf{Flatten}(\mathbf{C}) \cdot \vec{\mathbf{v}} = \mathbf{C} \cdot \vec{\mathbf{v}}$ for any $N \times N$ matrix $\mathbf{C}$. So $\mathbf{C}^{\mathsf{NAND}}$ will have entries in $\{0, 1\}$ and thus be strongly-bounded.

6.2.3 GSW Compiler for IBE in the Single-Identity Setting

The Gentry, Sahai and Waters (GSW) compiler from Crypto 2013 [98] (Section 4) allows transformation of an IBE scheme based on the Learning with Errors (LWE) problem into a related IBFHE scheme, provided the IBE scheme satisfies the following properties:

1. **Property 1 (Ciphertext and secret key vectors):** The secret key for identity id and a ciphertext created under id are vectors $\vec{\mathbf{s}_{\mathsf{id}}}, \vec{\mathbf{c}_{\mathsf{id}}} \in \mathbb{Z}_q^{m'}$ for some m'. The first coefficient of $\vec{\mathbf{s}_{\mathsf{id}}}$ is 1.

2. **Property 2 (Small Dot Product):** If $\vec{\mathbf{c}_{\mathsf{id}}}$ encrypts 0, then $\langle \vec{\mathbf{c}_{\mathsf{id}}}, \vec{\mathbf{s}_{\mathsf{id}}} \rangle$ is "small".

3. **Property 3 (Security):** Encryptions of 0 are indistinguishable from uniform vectors over $\mathbb{Z}_q$ under the hardness of LWE.

As noted in [98] all known LWE-based IBE schemes satisfy the above properties e.g: [7, 8, 56, 97].

Let $\mathcal{E}$ be an IBE satisfying the Properties 1-3 above. Then $\mathcal{E}$ can be transformed into a single-identity IBFHE scheme $\mathcal{E}'$.

The public parameters PP generated by $\mathcal{E}.\mathsf{Setup}$ includes a modulus q and an integer m' representing the length of both secret key and ciphertext vectors in $\mathcal{E}$. Let $\ell_q = \lfloor \lg q \rfloor + 1$ and $N = m' \times \ell_q$.

To encrypt a message $\mu \in \{0, 1\}$ under identity $\mathsf{id} \in \mathcal{I}$, the encryptor generates N encryptions of 0 using $\mathcal{E}$. More precisely, she computes $\vec{\mathbf{e_i}} \leftarrow \mathcal{E}.\mathsf{Encrypt}(\mathsf{PP}, \mathsf{id}, 0) \in \mathbb{Z}_q^{m'}$ for every $i \in [N]$. The set of N vectors $\vec{\mathbf{e_1}}, \ldots, \vec{\mathbf{e_N}}$ form the rows of an $N \times m'$ matrix $E \in \mathbb{Z}_q^{N \times m'}$. Finally the encryptor computes the $N \times N$ ciphertext matrix $\mathbf{C} \in \{0, 1\}^{N \times N}$ as follows

$$\mathbf{C} \leftarrow \mathsf{Flatten}(\mu \cdot \mathbf{I_N} + \mathsf{BitDecomp}(\mathbf{E}))$$

where $\mathbf{I_N}$ denotes the $N \times N$ identity matrix.

A secret key in $\mathcal{E}'$ for identity id is an N-dimensional vector $\vec{\mathbf{v}_{\mathsf{id}}}$ derived from a secret key $\vec{\mathbf{s}_{\mathsf{id}}}$ for identity id in $\mathcal{E}$. This is computed as $\vec{\mathbf{v}_{\mathsf{id}}} \leftarrow \mathsf{Powersof2}(\vec{\mathbf{s}_{\mathsf{id}}})$. Decryption of a ciphertext $\mathbf{C}$ with $\vec{\mathbf{v}_{\mathsf{id}}}$ is as follows. By construction of $\vec{\mathbf{v}_{\mathsf{id}}}$, it has at least one "large" coefficient; denote this by $v_{\mathsf{id},i}$, To perform decryption, we take the i-th row $\vec{\mathbf{c_i}}$ of matrix $\mathbf{C}$, compute the inner product $x \leftarrow \langle \vec{\mathbf{c_i}}, \vec{\mathbf{v}_{\mathsf{id}}} \rangle = \mu \cdot v_{\mathsf{id},i} + e_i$ and output the plaintext $\mu \leftarrow \lfloor x / v_{\mathsf{id},i} \rceil$. This is correct because

$$\mathbf{C} \cdot \vec{\mathbf{v}_{\mathsf{id}}} = \mu \cdot \vec{\mathbf{v}_{\mathsf{id}}} + \mathbf{E} \cdot \vec{\mathbf{s}_{\mathsf{id}}} = \mu \cdot \vec{\mathbf{v}_{\mathsf{id}}} + \text{"small"}$$

where $\mathbf{E} \cdot \vec{\mathbf{s}_{\mathsf{id}}}$ is "small" as a consequence of Property 2. It is also easy to see that semantic security for $\mathcal{E}'$ follows immediately from the fact that $\mathcal{E}$ satisfies Property 3.

6.3 A Compiler for Multi-Identity Leveled IBFHE

In this section, we present a new compiler that can transform an LWE-based IBE into a *multi-identity* IBFHE. As we will see, achieving multi-identity IBFHE is far more difficult than single-identity IBFHE.

6.3.1 Intuition

Suppose $\mathcal{E}$ is an LWE-based IBE that satisfies properties 1 - 3 above. We can apply the GSW compiler to yield an IBFHE scheme $\mathcal{E}'$ in the single-identity setting. Our goal is to construct a compiler for the multi-identity setting. Consider two ciphertexts $\mathbf{C_1}$ and $\mathbf{C_2}$ that encrypt μ_1 and μ_2 under identities id_1 and id_2 respectively. Let $\vec{\mathbf{s_1}}$ and $\vec{\mathbf{s_2}}$ be secret keys in the scheme $\mathcal{E}$ for identities id_1 and id_2 respectively. Accordingly, a decryptor can compute $\vec{\mathbf{v_1}} \leftarrow \mathsf{Powersof2}(\vec{\mathbf{s_1}})$ and $\vec{\mathbf{v_2}} \leftarrow \mathsf{Powersof2}(\vec{\mathbf{s_2}})$. It holds that $\mathbf{C_1} \cdot \vec{\mathbf{v_1}} = \mu_1 \cdot \vec{\mathbf{v_1}} + \vec{\mathbf{e_1}}$ and $\mathbf{C_2} \cdot \vec{\mathbf{v_2}} = \mu_2 \cdot \vec{\mathbf{v_2}} + \vec{\mathbf{e_2}}$ where $\vec{\mathbf{e_1}}, \vec{\mathbf{e_2}} \in \mathbb{Z}_q^N$ are short vectors.

We would like to be able to perform homomorphic computation on both $\mathbf{C_1}$ and $\mathbf{C_2}$ together; that is, use them both as inputs in the same circuit. Here we denote the circuit by $C \in \mathbb{C}$. We expect the size of the resulting ciphertext to grow if $\mathsf{id}_1 \neq \mathsf{id}_2$. This is intuitive because the resulting ciphertext must *encode* information about *both* identities. Assume that $\mathsf{id}_1 \neq \mathsf{id}_2$. The compactness condition of multi-identity IBFHE allows the size of the resulting ciphertext to depend polynomially on the number of *distinct* identities d (in this case $d = 2$). Suppose we could produce a resulting $2N \times 2N$ ciphertext matrix $\mathbf{C}' \in \mathbb{Z}_q^{2N \times 2N}$ that encrypts $\mu' = C(\mu_1, \mu_2)$. More precisely, suppose that

$$\mathbf{C}' \cdot \begin{bmatrix} \vec{\mathbf{v_1}} \\ \vec{\mathbf{v_2}} \end{bmatrix} = \mu' \cdot \begin{bmatrix} \vec{\mathbf{v_1}} \\ \vec{\mathbf{v_2}} \end{bmatrix} + \vec{\mathbf{e}'}$$

where $\vec{\mathbf{e}'}$ is "short". The size of the ciphertext matrix is quadratic in the number of distinct identities, and thus satisfies the compactness condition. How can such a matrix $\mathbf{C}'$ be computed?

The main idea behind our approach is to transform each input ciphertext matrix (i.e. $\mathbf{C_1}$ and $\mathbf{C_2}$ in this example) into a corresponding $\not{d}N \times \not{d}N$ "expanded matrix" where $\not{d}$ is the number of distinct identities (i.e. $\not{d} = 2$ in our example).

Consider any input ciphertext matrix $\mathbf{C} \in \mathbb{Z}_q^{N \times N}$ that encrypts a plaintext μ under identity id_1. We denote by $\hat{\mathbf{C}} \in \mathbb{Z}_q^{2N \times 2N}$ its corresponding "expanded matrix". We require this expanded matrix to satisfy

$$\hat{\mathbf{C}} \cdot \begin{bmatrix} \vec{\mathbf{v_1}} \\ \vec{\mathbf{v_2}} \end{bmatrix} = \mu \cdot \begin{bmatrix} \vec{\mathbf{v_1}} \\ \vec{\mathbf{v_2}} \end{bmatrix} + \text{"small"}.$$

Now $\hat{\mathbf{C}}$ can be viewed as consisting of 2×2 submatrices in $\mathbb{Z}_q^{N \times N}$. We denote the submatrix on row i and column j as $\hat{\mathbf{C}}_{i,j} \in \mathbb{Z}_q^{N \times N}$. To satisfy the "top" part of the above equation, it is sufficient to set $\hat{\mathbf{C}}_{1,1} \leftarrow \mathbf{C}$ and $\hat{\mathbf{C}}_{1,2} \leftarrow \mathbf{0}$. To satisfy the "bottom" part of the equation, we need to find matrices $\mathbf{X}, \mathbf{Y} \in \{0,1\}^{N \times N}$ such that

$$\mathbf{X} \cdot \vec{\mathbf{v_1}} + \mathbf{Y} \cdot \vec{\mathbf{v_2}} = \mu \cdot \vec{\mathbf{v_2}} + \text{"small"}.$$

We refer to a pair of solution matrices $(\mathbf{X}, \mathbf{Y})$ as a "mask" because of the fact that they hide the plaintext μ from a party that does not have a secret key for the recipient identity. In this section, we will abstract over the process of finding solution matrices $\mathbf{X}$ and $\mathbf{Y}$ with respect to arbitrary identities. Towards this goal, we introduce an abstraction called a *masking system*. In short, a masking system allows an encryptor to produce information $U \in \{0,1\}^*$ that allows an evaluator to derive matrices $\mathbf{X}$ and $\mathbf{Y}$ that solve the above equation with respect to any arbitrary identity. Informally, an adversary without a secret key for the *recipient identity* (id_1 in the above example) learns nothing about μ given U, but can still efficiently derive solution matrices $\mathbf{X}$ and $\mathbf{Y}$ with respect to any chosen identity. This notion is formalized in the next section, where we present our compiler. A concrete construction of a masking system is presented in Section 6.4.2.

6.3.2 Abstract Compiler

We start by describing an abstract framework for multi-identity IBFHE from Learning with Errors (LWE). Our compiler uses the aforementioned abstraction which we call a *masking system*. An additional prerequisite for an IBE scheme $\mathcal{E}$ (beyond Properties 1-3) to work with our compiler is that there exists a masking system $\mathsf{MS}_{\mathcal{E}}$ for $\mathcal{E}$. First we provide a formal definition of a masking system.

Definition 6.3.1. *Let $\mathcal{E}$ be an IBE scheme satisfying Properties 1-3. A* masking system *for $\mathcal{E}$ is a pair of PPT algorithms* $(\mathsf{GenUnivMask}, \mathsf{DeriveMask})$ *defined as follows:*

- $\mathsf{GenUnivMask}(\mathsf{PP}, \mathsf{id}, \mu)$ *takes as input public parameters* PP *for $\mathcal{E}$, an identity* $\mathsf{id} \in \mathcal{I}$ *and a message $\mu \in \{0,1\}$, and outputs $U \in \{0,1\}^*$ (referred to as a* universal mask*).*

- $\mathsf{DeriveMask}(\mathsf{PP}, U, \mathsf{id}')$ *takes as input public parameters* PP *for $\mathcal{E}$, a* universal mask *$U \in \{0,1\}^*$ and an identity $\mathsf{id}' \in \mathcal{I}$, and outputs a pair of matrices $(\mathbf{X}, \mathbf{Y}) \in (\mathbb{Z}_q^{N \times N})^2$.*

A masking system $(\mathsf{GenUnivMask}, \mathsf{DeriveMask})$ *must satisfy the following properties:*

- ***Correctness:*** *Let $w(\cdot)$ be a polynomial associated with the masking system. Let $w = w(\lambda)$. We refer to w as the error expansion factor. For correctness, it is required that for any $(\mathsf{PP}, \mathsf{MSK}) \leftarrow \mathcal{E}.\mathsf{Setup}(1^\lambda)$, any identities $\mathsf{id}, \mathsf{id}' \in \mathcal{I}$, any secret keys $\vec{\mathbf{v}_{\mathsf{id}}} \leftarrow \mathsf{Powersof2}(\mathcal{E}.\mathsf{KeyGen}(\mathsf{MSK}, \mathsf{id})) \in \mathbb{Z}_q^N$ and $\vec{\mathbf{v}_{\mathsf{id}'}} \leftarrow \mathsf{Powersof2}(\mathcal{E}.\mathsf{KeyGen}(\mathsf{MSK}, \mathsf{id}')) \in \mathbb{Z}_q^N$, and any $\mu \in \{0,1\}$, and over all*

  - $U \leftarrow \mathsf{GenUnivMask}(\mathsf{PP}, \mathsf{id}, \mu)$,
  - $(\mathbf{X}, \mathbf{Y}) \leftarrow \mathsf{DeriveMask}(\mathsf{PP}, U, \mathsf{id}')$

  it holds that

$$\mathbf{X}\vec{\mathbf{v}_{\mathsf{id}}} + \mathbf{Y}\vec{\mathbf{v}_{\mathsf{id}'}} = \mu \cdot \vec{\mathbf{v}_{\mathsf{id}'}} + \vec{\mathbf{e}} \tag{6.3.1}$$

  where $\|\|\vec{\mathbf{e}}\|\|_\infty \leq w \cdot B$.

- ***Security:*** *The masking system is said to be secure if all PPT adversaries have a negligible advantage in the following modified* IND-X-CPA *game for* $\mathcal{E}$ *where* $X \in \{\mathsf{sID}, \mathsf{ID}\}$. *The only change in the security game is that the adversary is given* $U^* \leftarrow \mathsf{GenUnivMask}(\mathsf{PP}, \mathsf{id}^*, \mu_b)$ *in place of the challenge ciphertext in the original game, where* $b \xleftarrow{\$} \{0,1\}$ *is the challenger's random bit,* id^* *is the adversary's target identity, and* μ_0 *and* μ_1 *are the challenge messages chosen by the adversary.*

Our compiler can compile an IBE scheme $\mathcal{E}$ into a IBFHE scheme $\mathcal{E}'$ if the following conditions are met (for completeness, we restate Properties 1-3 above):

CP.1: (Ciphertext and secret key vectors): The secret key for identity id and a ciphertext created under id are vectors $\vec{\mathbf{s}_{\mathsf{id}}}, \vec{\mathbf{c}_{\mathsf{id}}} \in \mathbb{Z}_q^{m'}$ for some m'. The first coefficient of $\vec{\mathbf{s}_{\mathsf{id}}}$ is 1.

CP.2: (Small Dot Product): . If $\vec{\mathbf{c}_{\mathsf{id}}}$ encrypts 0 under identity id, then $\vec{\mathbf{e}} = \langle \vec{\mathbf{c}_{\mathsf{id}}}, \vec{\mathbf{s}}_{\mathsf{id}} \rangle$ is "small" where $\vec{\mathbf{s}}_{\mathsf{id}}$ is generated as in CP.1. Formally, $\vec{\mathbf{e}}$ is B-bounded; that is, $\|\|\vec{\mathbf{e}}\|\|_\infty \leq B$.

CP.3: (Security): Encryptions of 0 are indistinguishable from uniform vectors over $\mathbb{Z}_q$ under the hardness of LWE.

CP.4: (Masking System): There exists a masking system $(\mathsf{GenUnivMask}, \mathsf{DeriveMask})$ for $\mathcal{E}$ meeting the correctness and security conditions of Definition 6.3.1.

Let $\mathsf{MS}_{\mathcal{E}} = (\mathsf{MS}_{\mathcal{E}}\mathsf{GenUnivMask}, \mathsf{MS}_{\mathcal{E}}\mathsf{DeriveMask})$ be a *masking system* for $\mathcal{E}$ that satisfies CP.4. A formal description is now given of a generic scheme, which we call mIBFHE, that uses $\mathcal{E}$ and $\mathsf{MS}_{\mathcal{E}}$. We have $\mathsf{mIBFHE.Setup} = \mathcal{E}.\mathsf{Setup}$ and $\mathsf{mIBFHE.KeyGen} = \mathcal{E}.\mathsf{KeyGen}$. The remaining algorithms are described as follows.

6.3.2.1 Encryption

To encrypt a message μ under identity $\mathsf{id} \in \mathcal{I}$, an encryptor performs the following steps. The encryptor computes the universal mask

$$U \leftarrow \mathsf{MS}_{\mathcal{E}}.\mathsf{GenUnivMask}(\mathsf{PP}, \mathsf{id}, \mu)$$

and outputs the ciphertext $\mathsf{CT} := (\langle \mathsf{id} \rangle, \mathsf{type} := 0, \mathsf{enc} := U)$. Setting the type component of CT to 0 indicates a "fresh" ciphertext.

6.3.2.2 Evaluation

The evaluator is given as input a circuit $C \in \mathbb{C}$ and a collection of ℓ ciphertexts $\mathsf{CT}_1 := (\langle \mathsf{id}_1 \rangle, \mathsf{type} := 0, \mathsf{enc} := U_1), \ldots, \mathsf{CT}_\ell := (\langle \mathsf{id}_\ell \rangle, \mathsf{type} := 0, \mathsf{enc} := U_\ell)$.

Consider the set of *distinct* identities $I = \{\mathsf{id}_1, \ldots, \mathsf{id}_\ell\}$. Suppose that $|I| = \mathscr{d} \leq \ell$ is the number of distinct identities. If $\mathscr{d} > \mathcal{D}$ (i.e. the maximum supported degree of composition is exceeded), the evaluator aborts the evaluation. For simplicity we relabel the distinct identities as $\mathsf{id}_1, \ldots, \mathsf{id}_{\mathscr{d}}$. Thus, each distinct identity in the collection is associated with a unique index in $[\mathscr{d}]$. Before evaluation can be performed, each ciphertext must be "transformed" into a $\mathscr{d}N \times \mathscr{d}N$ matrix, which we call an *expanded matrix*. This is achieved as follows.

Let $(\langle \mathsf{id}_r \rangle, \mathsf{type} := 0, \mathsf{enc} := U)$ be a ciphertext whose associated identity has been assigned the index $r \in [\mathscr{d}]$. A matrix $\hat{\mathbf{C}} \in \mathbb{Z}_q^{\mathscr{d}N \times \mathscr{d}N}$ is formed as follows. Start by setting $\hat{\mathbf{C}}$ to the zero matrix. Now $\hat{\mathbf{C}}$ can be viewed as consisting of $\mathscr{d} \times \mathscr{d}$ submatrices in $\mathbb{Z}_q^{N \times N}$. We denote the submatrix on row i and column j as $\hat{\mathbf{C}}_{\mathbf{i},\mathbf{j}} \in \mathbb{Z}_q^{N \times N}$.

For $i \in [\mathscr{d}]$:

1. Run $(\mathbf{X_i}, \mathbf{Y_i}) \leftarrow \mathsf{MS}_{\mathcal{E}}.\mathsf{DeriveMask}(\mathsf{PP}, U, \mathsf{id}_i)$.
2. Set $\hat{\mathbf{C}}_{\mathbf{i},\mathbf{i}} \leftarrow \mathbf{Y_i}$.
3. Set $\hat{\mathbf{C}}_{\mathbf{i},\mathbf{r}} \leftarrow \mathsf{Flatten}(\hat{\mathbf{C}}_{\mathbf{i},\mathbf{r}} + \mathbf{X_i})$. (The reason for addition here is to handle the special case of $i = r$).

This completes the process for computing the expanded matrix $\hat{\mathbf{C}}$. Consider an example where $r = 1$ and $\mathcal{d} > 2$. The expanded matrix looks like the following:

$$\hat{\mathbf{C}} = \begin{pmatrix} (\mathsf{Flatten}(\mathbf{X_1} + \mathbf{Y_1}) & & \\ \mathbf{X_2} & \mathbf{Y_2} & \\ \vdots & & \ddots \\ \mathbf{X}_{\mathcal{d}} & & \mathbf{Y}_{\mathcal{d}} \end{pmatrix}$$

Perform the steps above to produce the expanded matrix $\hat{\mathbf{C}}^{(\mathbf{i})}$ for every input ciphertext CT_i. Then the circuit $C \in \mathbb{C}$ is evaluated gate-by-gate (NAND gates) on the expanded matrices to yield a $\mathcal{d}N \times \mathcal{d}N$ matrix $\hat{\mathbf{C}}'$. Suppose each $\hat{\mathbf{C}}^{(\mathbf{i})}$ encrypts $\mu_i \in \{0, 1\}$. Then $\hat{\mathbf{C}}'$ encrypts $C(\mu_1, \ldots, \mu_\ell)$. Finally, the evaluation algorithm outputs the tuple $\mathsf{CT}' := (\langle \mathsf{id}_1, \ldots, \mathsf{id}_{\mathcal{d}} \rangle, \mathsf{type} := 1, \mathsf{enc} := \hat{\mathbf{C}}')$. Setting the type component to 1 indicates an evaluated ciphertext. Note that the scheme is 1-hop homomorphic.

6.3.2.3 Decryption

On input a ciphertext $\mathsf{CT} := (\langle \mathsf{id}_1, \ldots, \mathsf{id}_{\mathcal{d}} \rangle, \mathsf{type}, \mathsf{enc})$ and a sequence of secret keys $\vec{\mathbf{v}}_{\vec{\mathsf{id}}_1}, \ldots, \vec{\mathbf{v}}_{\vec{\mathsf{id}}_{\mathcal{d}}} \in \mathbb{Z}_q^N$ where $\vec{\mathbf{v}}_{\vec{\mathsf{id}}_i}$ is a secret key for id_i for $i \in [\mathcal{d}]$, the decryptor performs the following steps. Form the column vector $\vec{\mathbf{v}}$ as the vertical concatenation of the column vectors $\vec{\mathbf{v}}_{\vec{\mathsf{id}}_1}, \ldots, \vec{\mathbf{v}}_{\vec{\mathsf{id}}_{\mathcal{d}}}$. If $\mathsf{type} = 0$, parse enc as the universal mask U, compute $(\mathbf{X}, \mathbf{Y}) \leftarrow \mathsf{MS}_{\mathcal{E}}.\mathsf{DeriveMask}(\mathsf{PP}, U, \mathsf{id}_1)$ and set $\mathbf{C} \leftarrow \mathbf{X} + \mathbf{Y}$. Else if $\mathsf{type} = 1$, parse enc as $\hat{\mathbf{C}}$ and set $\mathbf{C} \leftarrow \hat{\mathbf{C}}$.

Recall that the first ℓ_q components of $\vec{\mathbf{v}}$ are $1, \ldots, 2^{\ell_q - 1}$. Let i be an index such that $v_i = 2^i \in (q/4, q/2]$. Compute $d_i \leftarrow \langle \vec{\mathbf{c}}_{\mathbf{i}}, \vec{\mathbf{v}} \rangle$ where $\vec{\mathbf{c}}_{\mathbf{i}}$ is the i-th row of $\mathbf{C}$ and output $\mu' \leftarrow \lfloor d_i / v_i \rceil \in \{0, 1\}$. This works to recover the message because as a result of Equation 6.3.1 (in Definition 6.3.1), we have

$$\mathbf{C}\vec{\mathbf{v}} = \mu \cdot \vec{\mathbf{v}} + \vec{\mathbf{e}}$$

with $\||\vec{\mathbf{e}}\||_\infty \leq w \cdot B$, where w is the error expansion factor associated with the masking system $\mathsf{MS}_{\mathcal{E}}$.

6.3.2.4 Correctness

Lemma 6.3.1. *Let B be a bound such that all freshly encrypted ciphertexts are B-strongly-bounded. Let $\mathcal{D}$ and L be positive integers. If $q > 8 \cdot w \cdot B(\mathcal{D}N + 1)^{L*}$, then the scheme* mIBFHE *is correct and can evaluate NAND-based Boolean circuits of depth L with any degree of composition $\mathscr{d} \leq \mathcal{D}$.*

Proof. Let the $\mathscr{d} \leq \mathcal{D}$ distinct identities involved in an evaluation be $\mathsf{id}_1, \ldots, \mathsf{id}_{\mathscr{d}}$. Consider an expanded matrix derived from a "fresh" ciphertext $\mathsf{CT} = (\langle \mathsf{id}_i \rangle, \mathsf{type} := 0, \mathsf{enc} := U)$ associated with identity id_i for some $i \in [\mathscr{d}]$. Let $\vec{\mathbf{v}}_j$ be a secret key that decrypts ciphertexts with identity id_j for $j \in [\mathscr{d}]$. Let $\vec{\hat{v}}$ be the column vector consisting of the concatenation of $\vec{\mathbf{v}}_1, \ldots, \vec{\mathbf{v}}_{\mathscr{d}}$. Let $\hat{\mathbf{C}}$ be the expanded matrix for CT computed with respect to identities $\mathsf{id}_1, \ldots, \mathsf{id}_{\mathscr{d}}$ and $(\mathbf{X_j}, \mathbf{Y_j}) \leftarrow \mathsf{MS}_{\mathcal{E}}.\mathsf{DeriveMask}(\mathsf{PP}, U, \mathsf{id}_j)$ for $j \in [\mathscr{d}]$. Now by construction, $\hat{\mathbf{C}}$ consists of $\mathscr{d} \times \mathscr{d}$ submatrices in $\mathbb{Z}_q^{N \times N}$. There are 2 non-zero submatrices on $N - 1$ rows when $\hat{\mathbf{C}}$ is viewed as $\mathscr{d} \times \mathscr{d}$ matrix over $\mathbb{Z}_q^{N \times N}$, and one non-zero submatrix on the i-th row. The correctness condition for the masking system $\mathsf{MS}_{\mathcal{E}}$ gives us

$$\begin{pmatrix} \mathsf{Flatten}(\mathbf{Y_1} & & \mathbf{X_1}) & & \\ & \ddots & \vdots & & \\ & & \mathbf{X_i} + \mathbf{Y_i} & & \\ & & \vdots & \ddots & \\ & & \mathbf{X}_{\mathscr{d}} & & \mathbf{Y}_{\mathscr{d}} \end{pmatrix} \cdot \begin{bmatrix} \vec{\mathbf{v_1}} \\ \vdots \\ \vec{\mathbf{v_i}} \\ \vdots \\ \vec{\mathbf{v}}_{\mathscr{d}} \end{bmatrix} = \begin{bmatrix} \mathbf{X_1}\vec{\mathbf{v_1}} + \mathbf{Y_1}\vec{\mathbf{v_1}} \\ \vdots \\ \mathbf{X_i}\vec{\mathbf{v_i}} + \mathbf{Y_i}\vec{\mathbf{v_i}} \\ \vdots \\ \mathbf{X}_{\mathscr{d}}\vec{\mathbf{v}}_{\mathscr{d}} + \mathbf{Y}_{\mathscr{d}}\vec{\mathbf{v}}_{\mathscr{d}} \end{bmatrix} = \mu \cdot \begin{bmatrix} \vec{\mathbf{v_1}} \\ \vdots \\ \vec{\mathbf{v_i}} \\ \vdots \\ \vec{\mathbf{v}}_{\mathscr{d}} \end{bmatrix} + \text{'small'} \; .$$

Since each of these submatrices is B-strongly-bounded, it follows that $\hat{\mathbf{C}} \cdot \vec{\hat{v}} = \mu \cdot \vec{\hat{v}} + \vec{\hat{e}}$ where the coefficients of the error vector $\vec{\hat{\mathbf{e}}}$ are bounded by $w \cdot B$.Therefore, $\vec{\hat{C}}$ is $w \cdot B$-strongly-bounded. Multiplying two $\mathscr{d}N \times \mathscr{d}N$ expanded matrices in a NAND operation produces a matrix that is $w \cdot B(\mathscr{d}N + 1)$-strongly-bounded. After L successive levels,

*Note that N (which depends on n) is itself dependent on $\lg q$. For security, it is required that $q/B = 2^{n^\epsilon}$ for some $\epsilon \in (0, 1)$. A discussion on parameters is provided in Section 6.5.

the bound on the error is $w \cdot B(\mathit{d}N+1)^L$. For correctness of decryption we need $w \cdot B(\mathit{d}N+1)^L < q/8$. Since we have $\mathit{d} \leq \mathcal{D}$, it follows that

$$w \cdot B(\mathit{d}N+1)^L \leq w \cdot B(\mathcal{D}N+1)^L \leq \frac{8 \cdot w \cdot B(\mathcal{D}N+1)^L}{8} < \frac{q}{8}.$$

□

Theorem 6.3.1. *Let $\mathcal{E}$ be an IBE scheme satisfying CP.1 - CP.4. Then $\mathcal{E}$ can be transformed into a multi-identity IBFHE scheme $\mathcal{E}'$.*

Proof. The proof of the theorem is constructive. By CP.4, there exists a masking system $\mathsf{MS}_{\mathcal{E}}$ for $\mathcal{E}$. The multi-identity IBFHE scheme $\mathcal{E}'$ that we obtain is mIBFHE instantiated with $\mathcal{E}$ and $\mathsf{MS}_{\mathcal{E}}$. By Lemma 6.3.1, the scheme is correct. CP.4 implies that $\mathcal{E}'$ is IND-X-CPA secure for some $X \in \{\mathsf{sID}, \mathsf{ID}\}$. □

6.4 Concrete Construction of Multi-Identity Leveled IBFHE

To exploit our compiler from the last section to obtain a multi-identity IBFHE, we need to find an LWE-based IBE scheme $\mathcal{E}$ that satisfies CP.1 - CP.4. The major obstacle is finding a scheme for which a secure masking system can be constructed. A natural starting point is the IBE of Cash, Hofheinz, Kiltz and Peikert (CHKP) [56], which is $\mathsf{IND\text{-}ID\text{-}CPA}$ secure in the standard model. This IBE was adapted by Gentry, Sahai and Waters ([98] Appendix A.1) to work with their compiler. There are difficulties however in developing a secure masking system for this IBE. Instead, we consider the IBE of Gentry, Peikert and Vaikuntanathan (GPV) [97]. Unfortunately this scheme is only secure under LWE in the random oracle model. On the plus side, we show that it enjoys the distinction of admitting a secure masking system, and as a consequence of Theorem 6.3.1 can be compiled into a multi-identity IBFHE scheme.

6.4.1 The Gentry, Peikert and Vaikuntanthan (GPV) IBE

In the GPV scheme, the TA needs to use a lookup table [†] to store secret keys that are issued to users in order to ensure that only a single unique secret key is ever issued for a given identity. This is required for the security proof in the random oracle model.

A hash function $H : \{0,1\}^* \to \mathbb{Z}_q^n$ (modeled as a random oracle in the security proof) is used to map an identity string $\mathsf{id} \in \{0,1\}^*$ to a vector $\vec{\mathbf{z}_{\mathsf{id}}} \in \mathbb{Z}_q^n$. Due to space constraints a formal description of the GPV scheme is deferred to Appendix **??**. It is easy to see that GPV fulfills CP.1 and CP.2. Furthermore, GPV can be shown to be $\mathsf{IND\text{-}sID\text{-}CPA}$ secure in the random oracle model [97] under LWE, and CP.3 follows from the security proof. It remains to construct a masking system for GPV.

6.4.2 A masking system for GPV

6.4.2.1 Relaxation: support for a single identity

As a warm up, we consider a relaxation of a masking system. In this relaxation, it is sufficient to find $\mathbf{X}$ and $\mathbf{Y}$ for only *one* identity id', specified by the encryptor. More precisely, let id be the recipient's identity and let $\mathsf{id}' \neq \mathsf{id}$ be another identity known to the encryptor. Furthermore, let $\vec{\mathbf{v}}$ be a secret key for id and let $\vec{\mathbf{v}'}$ be a secret key for id'. Then the goal is to allow the evaluator to find matrices $\mathbf{X}$ and $\mathbf{Y}$ satisfying

$$\mathbf{X} \cdot \vec{\mathbf{v}} + \mathbf{Y} \cdot \vec{\mathbf{v}'} = \mu \cdot \vec{\mathbf{v}'} + \text{“small”},$$

where μ is the plaintext. For every $i \in N$, we need to find row vectors $\vec{\mathbf{x_i}}$ and $\vec{\mathbf{y_i}}$ with $\langle \vec{\mathbf{x_i}}, \vec{\mathbf{v}} \rangle + \langle \vec{\mathbf{y_i}}, \vec{\mathbf{v}'} \rangle = \mu \cdot \vec{\mathbf{v}'} + \text{“small”}$.

A trivial way to do this is for the encryptor to set $\vec{\mathbf{x_i}} \leftarrow \vec{\mathbf{0}}$ and $\vec{\mathbf{y_i}} \leftarrow \mathsf{Flatten}((\underbrace{0}_{1,\ldots,i-1}, \mu, \underbrace{0}_{i+1,\ldots,N}) + \mathsf{BitDecomp}(\mathcal{E}.\mathsf{Encrypt}(\mathsf{PP}, \mathsf{id}', 0)) \in \{0,1\}^N$ where the latter is a GSW row encryption of μ under identity id'. Observe that such an $\vec{\mathbf{x_i}}$ and $\vec{\mathbf{y_i}}$ serve as a solution to the above

[†] Alternatively with the additional assumption of a PRF, a lookup table could be avoided by deterministically deriving secret keys (i.e. obtaining random coins from the PRF).

equation. However, it is easy to see that such a trivial solution violates semantic security, since a decryptor with a secret key $\vec{\mathbf{v}'}$ for id' (and no secret key for id) can still recover the plaintext μ.

One strategy for remedying the above approach is to prevent a key holder for identity id' from recovering μ from $\vec{\mathbf{y}_\mathbf{i}}$ by appropriately hiding some components of $\vec{\mathbf{y}_\mathbf{i}}$. Let us take a look at the structure of $\vec{\mathbf{y}_\mathbf{i}}$ when $\mathcal{E}$ is GPV. It is of the form

$$\mathsf{Flatten}((\underbrace{0}_{1,\ldots,i-1}, \mu, \underbrace{0}_{i+1,\ldots,N}) + \mathsf{BitDecomp}((\langle \vec{\mathbf{z}_{\mathsf{id}'}}, \vec{\mathbf{r}} \rangle + e, \vec{\mathbf{r}} \cdot \mathbf{A} + \vec{\mathbf{f}}) \in \mathbb{Z}_q^{m'})$$

where $e \xleftarrow{\$} \chi$, $\vec{\mathbf{f}} \xleftarrow{\$} \chi^m$, $\vec{\mathbf{r}} \xleftarrow{\$} \mathbb{Z}_q^n$ and $\vec{\mathbf{z}_{\mathsf{id}'}} = H(\mathsf{id}') \in \mathbb{Z}_q^n$. Suppose we instead generate $\vec{\mathbf{y}_\mathbf{i}}$ as

$$\vec{\mathbf{y}_\mathbf{i}} \leftarrow \mathsf{Flatten}((\underbrace{0}_{1,\ldots,i-1}, \mu, \underbrace{0}_{i+1,\ldots,N}) + \mathsf{BitDecomp}((0, \vec{\mathbf{r}} \cdot \mathbf{A} + \vec{\mathbf{f}}).$$

Now what we have done here is effectively set the first ℓ_q components of $\vec{\mathbf{y}_\mathbf{i}}$ to 0 with the exception of the special case $i \in [\ell_q]$ which we will handle separately later. As a result of this modification, we will have $\langle \vec{\mathbf{y}_\mathbf{i}}, \vec{\mathbf{v}'} \rangle \approx -\langle \vec{\mathbf{z}_{\mathsf{id}'}}, \vec{\mathbf{r}} \rangle + \mu \cdot 2^{i \mod \ell_q}$ (the symbol $\approx$ denotes equality up to "small" differences). Therefore, to cancel out the term $\langle \vec{\mathbf{z}_{\mathsf{id}'}}$, $\rangle, we need to ensure that we set such that \langle \vec{\mathbf{x}_\mathbf{i}}, \vec{\mathbf{v}} \rangle \approx \langle \vec{\mathbf{z}_{\mathsf{id}'}}, \vec{\mathbf{r}} \rangle$.

The approach we take to achieve this is to *blind* the element $\langle \vec{\mathbf{z}_{\mathsf{id}'}}, \vec{\mathbf{r}} \rangle$ with a a GPV encryption of zero under identity id such that it can only be *unblinded* with a secret key for identity id (note that the value cannot be recovered outright; instead a noisy approximation is obtained). For simplicity we define the algorithm Blind which takes an identity id and a value $v \in \mathbb{Z}_q$ and outputs a vector $\mathsf{Flatten}((c_1 + v, c_2, \ldots, c_{m'}))$ where $\vec{\mathbf{c}} \leftarrow \mathcal{E}.\mathsf{Encrypt}(\mathsf{PP}, \mathsf{id}, 0)$. So to provide an $\vec{\mathbf{x}_\mathbf{i}}$ counterpart to the vector $\vec{\mathbf{y}_\mathbf{i}}$ we generated above, we set $\vec{\mathbf{x}_\mathbf{i}} \leftarrow \mathsf{Blind}(\mathsf{id}, (\langle \vec{\mathbf{z}_{\mathsf{id}'}}, \vec{\mathbf{r}} \rangle)$ where $\vec{\mathbf{r}}$ is the vector used in the generation of $\vec{\mathbf{y}_\mathbf{i}}$ above. It follows that $\langle \vec{\mathbf{x}_\mathbf{i}}, \vec{\mathbf{v}} \rangle + \langle \vec{\mathbf{y}_\mathbf{i}}, \vec{\mathbf{v}'} \rangle = \mu \cdot \vec{\mathbf{v}'} +$ "small".

There are subtleties that we have overlooked. For security reasons, we need to change how we generate $\vec{\mathbf{x}_\mathbf{i}}$ and $\vec{\mathbf{y}_\mathbf{i}}$ for $i \in [\ell_q]$. This is because for the first ℓ_q components of $\vec{\mathbf{y}_\mathbf{i}}$ as generated above, the plaintext μ is not hidden; it is effectively sent in the clear.

However we can resolve this issue by setting $\vec{\mathbf{x}_\mathbf{i}} \leftarrow \mathsf{Blind}(\mathsf{id}, \mu \cdot 2^{i-1}$ $\vec{\mathbf{y}_\mathbf{i}} \leftarrow \vec{\mathbf{0}}$ and simply setting $\vec{\mathbf{y}_\mathbf{i}} \leftarrow \vec{\mathbf{0}}$.

However there is still a major weakness in this approach. Suppose a decryptor has access to two decryption vectors $\vec{\mathbf{u}'}, \vec{\mathbf{v}'} \in \mathbb{Z}_q^N$ that decrypt ciphertexts with identity id'. For example, the TA might have generated distinct secret key vectors when issuing keys to different parties, and the parties may have shared that information.

It is easy to see that

$$\mathbf{Y} \cdot \vec{\mathbf{u}'} - \mathbf{Y} \cdot \vec{\mathbf{v}'} = \mu \cdot (\vec{\mathbf{u}'} - \vec{\mathbf{v}'}) + \text{“small”},$$

which allows the decryptor to easily determine $\mu \in \{0, 1\}$. Hence a necessary condition for the approach to work is that there be a unique secret key vector for every identity. In fact, this is the primary reason our techniques do not work for ABE. Technically, this restriction means that the system can only support simple classes of access policies, namely classes of predicates with disjoint support sets, which includes the special case of IBE. Fortunately, in the GPV scheme, only a single secret key is ever issued for a given identity.

6.4.2.2 Support for all identities

The algorithm above allows an encryptor to create a secure “mask” for a specific identity that he knows. But how can we create a succinct “universal mask” from which “masks” for arbitrary identities can be derived? To achieve this, we need to take a look at the structure of vector $\vec{\mathbf{x}_\mathbf{i}}$ in our masking system, which is constructed as $\vec{\mathbf{x}_\mathbf{i}} \leftarrow \mathsf{Blind}(\mathsf{id}, \langle \vec{\mathbf{z}_{\mathsf{id}'}}, \vec{\mathbf{r}} \rangle)$ where id' is known to the encryptor. But what if id' is an arbitrary identity (i.e. not simply one that is known beforehand by the encryptor but one that is chosen by the evaluator at evaluation time)? In this case, we need to obtain an $\vec{\mathbf{x}_\mathbf{i}}$ that blinds $\langle \vec{\mathbf{z}_{\mathsf{id}'}}, \vec{\mathbf{r}} \rangle$. Our goal is to include information in the universal mask that we derive so that for any identity id' one can derive an $\vec{\mathbf{x}_\mathbf{i}}$ that blinds $\langle \vec{\mathbf{z}_{\mathsf{id}'}}, \vec{\mathbf{r}} \rangle$ where $\vec{\mathbf{z}_{\mathsf{id}'}} = H(\mathsf{id}')$.

Recall the following property of $\mathsf{BitDecomp}$ from Section 6.2.2.1:

$$\langle \vec{\mathbf{z}_{\mathsf{id}'}}, \vec{\mathbf{r}} \rangle = \langle \mathsf{BitDecomp}(\vec{\mathbf{z}_{\mathsf{id}'}}), \mathsf{Powersof2}(\vec{\mathbf{r}}) \rangle.$$

Our approach is to *blind* each coefficient of $\mathsf{Powersof2}(\vec{\mathbf{r}})$, whose length is $\ell_q \cdot n$. We produce a matrix $\mathbf{B}^{(i)} \in \mathbb{Z}_q^{(\ell_q \cdot n) \times m'}$ by letting $\vec{\mathbf{b}_\mathbf{j}^{(\mathbf{i})}} \leftarrow \mathsf{BitDecomp}^{-1}(\mathsf{Blind}(\mathsf{id}, p_j))$ where p_j be the j-th coefficient of $\mathsf{Powerof2}(\vec{\mathbf{r}})$. Then to generate $\vec{\mathbf{x}_\mathbf{i}}$, one computes $\vec{\mathbf{x}_\mathbf{i}} \leftarrow \mathsf{Flatten}(\mathsf{BitDecomp}(\vec{\mathbf{z}_{\mathsf{id}'}}) \cdot \mathbf{B}^{(i)})$. Note that $\vec{\mathbf{y}_\mathbf{i}}$ is generated as before.

More precisely what we have is shown is how to generate $\mathbf{B}^{(i)}$ and $\vec{\mathbf{y}}_i$ for $i \in [\ell_q]$. Recall that in our previous masking system we generated $\vec{\mathbf{x}_\mathbf{i}}$ and $\vec{\mathbf{y}_\mathbf{i}}$ differently for $i \in [\ell_q]$. This will also apply here. Instead of computing $\mathbf{B}^{(i)}$ for $i \in [\ell_q]$, we instead merely compute $\vec{\mathbf{x}_\mathbf{i}} \leftarrow \mathsf{Blind}(\mathsf{id}, \mu \cdot 2^{i-1})$ and $\vec{\mathbf{y}_\mathbf{i}} \leftarrow \vec{\mathbf{0}}$. This completes the description of our masking system.

We now formally present our masking system for GPV. (which we call $\mathsf{MS}_{\mathsf{GPV}}$). Let $\eta = \ell_q \cdot n$.

$\mathsf{MS}_{\mathsf{GPV}}$.GenUnivMask$(\mathsf{PP}, \mathsf{id}, \mu)$:

1. For $i \in [\ell_q]$:
   (a) Set $\vec{\mathbf{x}_\mathbf{i}} \leftarrow \mathsf{Blind}(\mathsf{id}, \mu \cdot 2^{i-1})$
   (b) Set $\vec{\mathbf{y}_\mathbf{i}} \leftarrow \vec{\mathbf{0}}$
2. For $\ell_q < i \leq N$:
   (a) Generate $\vec{r} \xleftarrow{\$} \mathbb{Z}_q^n$ and sample a short error vector $\vec{\mathbf{e}} \xleftarrow{\$} \chi^{m'}$.
   (b) For $j \in [\eta]$:
      i. Set $\vec{\mathbf{b}_\mathbf{j}^{(\mathbf{i})}} \leftarrow \mathsf{BitDecomp}^{-1}(\mathsf{Blind}(\mathsf{id}, p_j)) \in \mathbb{Z}_q^{m'}$ where p_j be the j-th coefficient of $\mathsf{Powerof2}(\vec{\mathbf{r}})$
   (c) Form matrix $\mathbf{B}^{(\mathbf{i})}$ from rows $\vec{\mathbf{b}_\mathbf{1}^{(\mathbf{i})}}, \ldots, \vec{\mathbf{b}_\eta^{(\mathbf{i})}}$.
   (d) Set $\vec{\mathbf{y}_\mathbf{i}} \leftarrow \mathsf{Flatten}((\underbrace{0}_{1,\ldots,i-1}, \mu, \underbrace{0}_{i+1,\ldots,N}) + \mathsf{BitDecomp}((0, \vec{\mathbf{r}} \cdot \mathbf{A} + \vec{\mathbf{f}})))$

3. Form matrix $\mathbf{Y}$ from rows $\vec{\mathbf{y_1}}, \ldots, \vec{\mathbf{y_N}}$.

4. Output $U := (\vec{\mathbf{x_1}}, \ldots, \vec{\mathbf{x}_{\ell_q}}, \mathbf{Y}, \mathbf{B}^{(\ell_q+1)}, \ldots, \mathbf{B}^{(N)})$.

$\mathsf{MS_{GPV}.DeriveMask}(\mathsf{PP}, U, \mathsf{id}') :$

1. Parse U as $(\vec{\mathbf{x_1}}, \ldots, \vec{\mathbf{x}_{\ell_q}}, \mathbf{Y}, \mathbf{B}^{(\ell_q+1)}, \ldots, \mathbf{B}^{(N)})$.

2. Compute $\vec{\mathbf{z}_{\mathsf{id}'}} \leftarrow H(\mathsf{id}')$.

3. For $\ell_q < i \leq N$:

   (a) Set $\vec{\mathbf{x_i}} \leftarrow \mathsf{Flatten}(\mathsf{BitDecomp}(\vec{\mathbf{z}_{\mathsf{id}'}}) \cdot \mathbf{B}^{(i)})$

4. Form $\mathbf{X} \in \{0,1\}^{N \times N}$ from $\vec{\mathbf{x_1}}, \ldots, \vec{\mathbf{x_N}}$.

5. Output $(\mathbf{X}, \mathbf{Y})$.

It is easy to see from the definition of $\mathsf{MS_{GPV}.DeriveMask}$ that the error expansion factor is $w = \eta + 1$. This is because each row in an expanded matrix is formed from a row of $\mathbf{X}$ and a row of $\mathbf{Y}$. But the former decomposes into a sum of η ciphertexts (and hence error terms).

Theorem 6.4.1. *[Informal] The masking system* $\mathsf{MS_{GPV}}$ *is selectively secure in the random oracle model (i.e.* $\mathsf{MS_{GPV}}$ *meets the security condition of Definition 6.3.1).*

A formal statement of Theorem 6.4.1 along with the proof is given in Section 6.7.

6.4.3 Applying the Compiler

It is now possible to put all the pieces together. In more detail, we can now apply our compiler to the IBE scheme GPV with the masking system $\mathsf{MS_{GPV}}$ to yield an $\mathsf{IND\text{-}sID\text{-}CPA}$ secure multi-identity IBFHE in the random oracle model.

Theorem 6.1.1. *There exists a multi-identity leveled IBFHE scheme that is* IND-sID-CPA *secure in the random oracle model under the hardness of LWE.*

Proof. Let $\mathcal{D}$ be a maximum degree of composition to support, and let L be a desired number of levels. Let λ be the security parameter. We show there exists a leveled IBFHE scheme with maximum degree of composition $\mathcal{D}$, maximum circuit depth L and security parameter λ.

Choose dimension parameter $n = n(\lambda, L)$ and bound $B = B(n)$. Lemma 6.3.1 requires

$$q > 8 \cdot w \cdot B(\mathcal{D}N + 1)^L \tag{6.4.1}$$

to ensure correctness. Note that w is the expansion factor of the masking system. Now the error expansion factor of $\mathsf{MS}_{\mathsf{GPV}}$ is $w = \eta + 1$. But this can be simplified to N[‡]. Theorem 6.4.1 requires $m \geq 2n \lg q$, and we have $N = (m+1)\lg q$. We need to set q first before setting these parameters (m and N) because of their dependence on q. To do so, q must be expressed without dependence on N. It can be straightforwardly derived from the inequality 6.4.1 that a suitable q is given by

$$q = B \cdot 2^{O(L \lg n\mathcal{D})}$$

with additional care taken to ensure q/B is subexponential in n.

Our parameter settings ensure that the GPV scheme meets CP.1, CP.2 and CP.3, three of the prerequisites for our compiler in Section 6.3. Furthermore, the masking system $\mathsf{MS}_{\mathsf{GPV}}$ is secure (via Theorem 6.4.1). As a result, CP.4 is additionally satisfied. Therefore, Theorem 6.3.1 ensures there exists a secure leveled IBFHE scheme, which by virtue of our parameter settings above (which meet Lemma 6.3.1), can correctly evaluate L-depth circuits over ciphertexts with at most $\mathcal{D}$ distinct identities . □

[‡] $w = \eta + 1 = \ell_q \cdot n + 1 \leq \ell_q \cdot m < N$.

6.4.4 Multi-Key FHE

If we replace the GPV IBE with the Dual-Regev public-key encryption scheme from [97], then we can obtain a multi-key FHE. The only change in the masking system is that identity vectors (i.e. $\vec{\mathbf{z}_{\mathsf{id}}} = H(\mathsf{id}) \in \mathbb{Z}_q^n$) are replaced with public-key vectors in $\mathbb{Z}_q^n$. As a result, the random oracle H is no longer needed, and security holds in the standard model. However the ciphertexts are prohibitively large; see Section 6.5.5 for an illustration of the extent of their impracticality. To reduce the ciphertext size, we adapt the scheme to work over polynomial rings instead of vectors.

Our compiler is compatible with several public-key RLWE schemes including the scheme of Lyubashevsky, Peikert and Regev (LPR) [136], which Gentry, Sahai and Waters adapt to the approximate eigenvector framework in [98]. The only issue we need focus on here that is not discussed in [98] is our masking system. Fortunately the approach underlying our masking system for GPV is directly applicable to LPR. Instead of blinding inner products over $\mathbb{Z}_q$, one blinds products in the ring $R_q = \mathbb{Z}_q[x]/f(x)$. In LPR, the modulus polynomial is $f(x) = x^d + 1$ for some $d = d(\lambda)$. The public parameters include a uniformly random element $a(x) \in R_q$. The public key of a user is an element $b(x) \in R_q$ of the form $b(x) = a(x)s(x) + e(x)$, where the secret key $s(x)$ is a uniformly random polynomial in R_q and $e(x)$ is an error polynomial drawn from an error distribution χ_R (analogous to χ but defined over $R = \mathbb{Z}[x]/f(x)$). A ciphertext $\vec{\mathbf{c}}$ in LPR that encrypts zero under the public key $b(x)$ is a pair of elements $(c_1(x), c_2(x)) \in R_q$ where $c_1(x) = b(x)r(x) + e_1(x)$ and $c_2(x) = a(x)r(x) + e_2(x)$ with $r(x), e_1(x), e_2(x)$ independently sampled from χ_R . This scheme can be compiled via the GSW compiler to yield a fully-homomorphic system whose ciphertexts are $2\ell_q \times 2\ell_q$ matrices over R_q, where $\ell_q = \lfloor \lg q \rfloor + 1$.

Let $\mathsf{pk} = b(x) \in R_q$ be the public key of the recipient in the following discussion. Recall the masking system from Section 6.4.2. Adapting it to the scheme above, a universal mask consists of two matrices $\mathbf{Y} \in R_q^{2\ell_q \times 2}$ and $\mathbf{B} \in R_q^{(\eta \cdot 2\ell_q) \times 2}$ with $\eta = \ell_q$.

Consider the i-th row $\vec{\mathbf{y_i}} \in R_q^2$ of $\mathbf{Y}$ for $i > \ell_q$. The second column of $\vec{\mathbf{y_i}}$ is of the form $a(x)r(x) + e(x) + \mu'$ for some $r(x), e(x) \xleftarrow{\$} \chi_R$ where $\mu' = \mu \cdot 2^{i \mod \ell_q}$ is a shifted version of the message $\mu \in \{0, 1\}$. Let $\mathsf{pk}' = b'(x) \in R_q$ be an arbitrary public key. Our goal is to produce an LPR ciphertext that *blinds* the product $b'(x)r(x) \in R_q$. This can be obtained from a set of ℓ_q ciphertexts $\{(e_1^{(j)}(x), e_2^{(j)}(x))\}_{0 \leq j < \ell_q}$ in which $(e_1^{(j)}(x), e_2^{(j)}(x))$ blinds the element $2^j r(x) \in R_q$ for $0 \leq j < \ell_q$ More precisely to compute a ciphertext $(t_1(x), t_2(x))$ that blinds the product $b'(x)r(x)$, one computes $t_1(x) \leftarrow \sum_{k=0}^{d-1} \sum_{j=0}^{\ell_q - 1} b'_{k,j} e_1^{(j)}(x) x^k$ and $t_2(x) \leftarrow \sum_{k=0}^{d-1} \sum_{j=0}^{\ell_q - 1} b'_{k,j} e_2^{(j)}(x) x^k$ where $b'_{(k,j)} \in \{0, 1\}$ is the j-th bit of b'_k for $0 \leq k < d$ and $0 \leq j < \ell_q$. The elements $\{(e_1^{(j)}(x), e_2^{(j)}(x))\}_{0 \leq j < \ell_q}$ form the rows of a $\ell_q \times 2$ submatrix of the blinding matrix $\mathbf{B}$; this submatrix corresponds to the i-th row. However, since there are $2\ell_q$ rows in $\mathbf{Y}$, this means that $\mathbf{B}$ is a $(2\ell_q \cdot \ell_q) \times 2$ matrix over R_q. Furthermore, $\mathbf{Y}$ is a $2\ell_q \times 2$ matrix over R_q. Since a fresh ciphertext in our scheme consists of the pair $(\mathbf{B}, \mathbf{Y})$, we have that it consists of $((2\ell_q \cdot \ell_q) \cdot 2) + 4\ell_q = 4\ell_q(1 + \ell_q)$ elements of R_q. Choosing $n = 16384$ and $\ell_q = 462$ (this 33% smaller than the value that satisfies our correctness bound due to experimental results that suggests the noise grows slower than expected [131]) for 80 bits of security [131] and to allow evaluation of $L = 40$ levels with $N = 100$ distinct keys yields a ciphertext size of approximately 754 GB per bit of plaintext. Suppose one were to use the scheme to encrypt an 80-bit symmetric key, we would obtain a 59 TB ciphertext, which is severely impractical.

In the next section, parameters are discussed for this scheme.

6.5 Parameters for our Scheme

Before discussing how parameters are chosen for our scheme, more background is needed on preimage sampling.

6.5.1 Background on Preimage Sampling

Let $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ be a matrix. We define the lattice $\Lambda^{\perp}(\mathbf{A}) = \{\vec{\mathbf{x}} \in \mathbb{Z}^m : \mathbf{A} \cdot \vec{\mathbf{x}} = \vec{\mathbf{0}} \bmod q\}$ as the space of vectors orthogonal to the rows of $\mathbf{A}$ modulo q. There exist efficient algorithms to generate a statistically uniform matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ together with a short basis $\mathbf{S} \in \mathbb{Z}^{m \times m}$ for $\Lambda^{\perp}(\mathbf{A})$ [13, 17]. Such an algorithm will be simply called $\mathsf{TrapGen}$ here; that is, we will write $(\mathbf{A}, \mathbf{S}) \leftarrow \mathsf{TrapGen}(n, m, q)$. We denote by $\tilde{\mathbf{S}}$ the Gram-Schmidt orthonormalization of a basis $\mathbf{S}$. Let $\mathfrak{L} = \|\tilde{\mathbf{S}}\|$ be the norm of $\mathbf{S}$. There are instances of $\mathsf{TrapGen}$ that achieve $\mathfrak{L} = m^{1+\epsilon}$ for any $\epsilon > 0$ [97], although this has been improved upon in other works [142]. Hence, our setting of $\mathfrak{L}$ later will be a conservative choice.

Let d and t be positive integers with $d \leq t$. Let $\mathbf{B} \in \mathbb{R}^{d \times t}$ be a basis for a d-dimensional lattice $\Lambda(\mathbf{B}) \subset \mathbb{R}^t$. Then the discrete Gaussian distribution on $\Lambda(\mathbf{B})$ with center $\vec{\mathbf{c}} \in \mathbb{R}^t$ and standard deviation $\sigma \in \mathbb{R}$ is denoted by $D_{\Lambda(\mathbf{B}),s,\vec{\mathbf{c}}}$. When $\vec{\mathbf{c}}$ is understood to be zero, the center parameter is omitted.

Gentry, Peikert and Vaikuntanthan [97] describe an algorithm to sample from a discrete Gaussian distribution on an arbitrary lattice. They describe an efficient probabilistic algorithm $\mathsf{SampleD}(\mathbf{B}, \sigma, \vec{\mathbf{c}})$ that samples from a distribution that is statistically close to $D_{\Lambda(\mathbf{B}),\sigma,\vec{\mathbf{c}}}$, provided $\sigma \geq \|\tilde{\mathbf{B}}\| \cdot \omega(\sqrt{\log d})$.

Consider the function $f_A : \mathbb{Z}_q^m \to \mathbb{Z}_q^n$ defined by $f(\vec{\mathbf{x}}) = \mathbf{A} \cdot \vec{\mathbf{x}} \in \mathbb{Z}_q^n$. Given any vector $\vec{\mathbf{u}} \in \mathbb{Z}_q^n$, a *preimage* of $\vec{\mathbf{u}}$ under f_A is any $\vec{\mathbf{x}} \in \mathbb{Z}_q^m$ with $f_A(\vec{\mathbf{x}}) = \vec{\mathbf{u}}$.

It turns out $\mathsf{SampleD}$ can be used to efficiently to find *short preimages* $\vec{\mathbf{x}} \in \mathbb{Z}_q^m$ such that $\mathbf{A} \cdot \vec{\mathbf{x}} = \vec{\mathbf{u}} \in \mathbb{Z}_q^n$ for an arbitrary vector $\vec{\mathbf{u}} \in \mathbb{Z}_q^n$. Consider the following algorithm $\mathsf{SamplePre}$ from [97]. Note that s is a parameter for which possible settings are given in the next section.

- $\mathbf{SamplePre}(\mathbf{S}, \mathbf{A}, \vec{\mathbf{u}})$: Find an arbitrary solution $\vec{\mathbf{t}} \in \mathbb{Z}_q^m$ (via linear algebra) such that $\mathbf{A} \cdot \vec{\mathbf{t}} = \vec{\mathbf{u}} \bmod q$. Sample a vector $\vec{\mathbf{e}} \xleftarrow{\$} D_{\Lambda^{\perp}(\mathbf{A}),s,-\vec{\mathbf{t}}}$ by running $\vec{\mathbf{e}} \leftarrow \mathsf{SampleD}(\mathbf{S}, s, -\vec{\mathbf{t}})$, and output the vector $\vec{\mathbf{x}} \leftarrow \vec{\mathbf{e}} + \vec{\mathbf{t}}$.

We remind the reader that there are improved variants of $\mathsf{SamplePre}$ in the literature [142].

6.5.2 Preimage Distribution

We need $s \geq \mathfrak{L} \cdot \omega(\sqrt{\log m})$ to satisfy Theorem 5.9 of [97]. Let $B_{\mathsf{preimage}} \geq \sqrt{n} \cdot s$. Then the probability of the magnitude of any coefficient of a preimage vector exceeding B_{preimage} is exponentially small in n via a standard tail inequality for a normal distribution [§]. One possible setting is $s = \mathfrak{L} \cdot \log m$, and $B_{\mathsf{preimage}} = \sqrt{n} \cdot s$.

6.5.3 Noise Distribution

To satisfy Theorem 6.2.1, we need the noise distribution χ to be B_χ-bounded for some B_χ (to satisfy Theorem 6.2.1, we require q/B_χ to be at most subexponential). Setting $\chi \leftarrow D_{\mathbb{Z},r}$ with $r = \log m$ and $B_\chi \geq \sqrt{n} \cdot r$ ensures that χ is B_χ-bounded, since by the aforementioned tail inequality, we have that $\mathsf{Pr}[x \xleftarrow{\$} D_{\mathbb{Z},r}, |x| > B_\chi]$ is exponential in n.

6.5.4 Parameter B (B-strong-boundedness)

"Fresh" ciphertexts in our scheme are B-strongly-bounded. The parameter B is derived from the product of B_{preimage} and B_χ, since when the ciphertext matrix is multiplied by a secret key vector, the resulting error vector is formed from the inner product of the noise vector in the ciphertext (drawn from χ) and the secret key (a sampled preimage). Concretely, with the suggested parameter setting, we have $B = \mathfrak{L} \cdot n \cdot \log^2 m$. It is necessary that q/B_1 is at most subexponential in N. However, our analysis simplifies this by taking q/B to be subexponential; however, since B_{preimage} is polynomial in N, it also holds that q/B_χ is subexponential.

[§] A normal variable with standard deviation σ is within $t \cdot \sigma$ standard deviations of its mean, except with probability at most $\frac{1}{t} \cdot \frac{1}{e^{t^2/2}}$ [97].

6.5.5 Sample Parameters and Ciphertext Size

Gentry, Sahai and Waters simplify their analysis by taking n to be a fixed parameter. This is a simplification because q/B must be subexponential in n, and q depends on L; therefore in actuality n depends on L.

Let L be the desired number of levels and let $\mathcal{D}$ be the desired maximum degree of composition. According to Lemma 6.3.1, correctness requires that

$$q > 8 \cdot w \cdot B(\mathcal{D}N + 1)^L. \tag{6.5.1}$$

In Section 6.5.1, it was mentioned that $\mathfrak{L} \approx m$. Putting this together with the derivation of B above in Section 6.5.4 gives $B = mn \cdot \log^2 m$, where $m \geq 2n \lg q$ from Theorem 6.4.1. Choosing B in this way means that it is not too large and allows us to derive $\lg q$ from the inequality 6.5.1 above as follows: $\lg q = O(L(\lg \mathcal{D} + \lg n))$.

Consider the following concrete parameters. Suppose we require a circuit depth of $L = 40$ and a degree of composition up to $\mathcal{D} = 100$. We can satisfy the correctness constraint given by 6.5.1 by setting $\lg q = \lceil c \cdot L(\lg \mathcal{D} + \lg L) = 4 \cdot 40(\lg 100 + \lg 40) \rceil = 1915$ (the constant $c = 4$ was chosen to meet the condition) and choosing the dimension to be $n = 2000$. However the size of freshly encrypted ciphertexts in our leveled IBFHE scheme with these parameters is greater than one exabyte (i.e. $> 2^{30}$ gigabytes) per bit of plaintext, which is extremely impractical. This illustrates the impracticality of our scheme, but it also highlights the impracticality of the GSW leveled IBFHE and ABFHE schemes, which have only marginally smaller ciphertexts (we simply have $\mathcal{D} = 1$ instead).

6.5.6 Basing Security on NTRU and Optimizations

No space savings are apparent if our multi-identity scheme is adapted to the RLWE setting. A recent paper by Ducas, Lyubashevsky and Prest [77] show that the GPV sampling algorithm can be instantiated with a particular distribution of lattices, known as *NTRU lattices* after the NTRU cryptosystem [117]; these lattices give a nearly optimal length for the lattice trapdoor. This has the effect of reducing a primary parameter,

namely m, in GPV , and by extension, in our multi-identity leveled IBFHE. We defer the details to their paper, but it is sufficient to note here that GPV ciphertexts when adapted to RLWE are reduced to two elements in a polynomial ring $R_q = ZZ_q[x]\ f(x)$. So this means that the ciphertext size, if we assume the hardness of solving lattice problems over NTRU lattices, is roughly the same as our multikey FHE,

Recall our analysis of parameter settings for our multikey FHE from Section 6.4.4. For an appropriate choice of parameters to evaluate circuits with $L = 40$ levels and up to $\mathcal{D} = 100$ distinct keys, and with 80-bits of security, each ciphertext consumed 754 GB; encrypting an 80-bit symmetric key to use hybrid homomorphic encryption then requires 59 TB per ciphertext. There is an optimization we can apply that reduces the ciphertext size to under a terrabyte at the expense of an increased size in the evaluated ciphertexts. The main idea is to reuse the matrix $\mathbf{X}$ over multiple plaintext bits by associating a user with κ identities.

6.6 Size of Evaluated Ciphertexts

As mentioned in the previous section, n is not a fixed parameter that depends solely on the security level λ. Instead n grows with both L and $\mathcal{D}$ because q/B must be subexponential in n to guarantee security. There is an optimization that applies to both our construction and the GSW constructions in terms of the size of evaluated ciphertexts. Decryption only requires a single row of a ciphertext matrix (see Section 6.3.2.3), so an evaluated ciphertext can have size $\mathscr{d} \cdot N$ where $\mathscr{d}$ is the degree of composition of the evaluation. Let this vector be denoted by $\vec{\hat{c}} \in \{0,1\}^{\mathscr{d} \cdot N}$. Applying $\mathsf{BitDecomp}^{-1}$, the vector $\vec{\mathbf{c}} \leftarrow \mathsf{BitDecomp}^{-1}(\vec{\hat{c}}) \in \mathbb{Z}_q^{m'}$ is obtained. As explained in [98], if we include additional information in the public parameters, the technique of modulus reduction [49] can be employed to each coefficient in $\vec{\mathbf{c}}$ so that the size of each coefficient can be made independent of $\mathcal{D}$ and L; their size must still depend on $\mathscr{d}$ to ensure correctness, but this is allowed for by the compactness condition. However, while every coefficient can be

reduced, the dimension cannot be reduced. This is because the technique of dimension reduction [49] appears to be only compatible with the public key setting since it relies on publishing encryptions of the secret key. We defer the details to [49]. So the length of the ciphertext vector is the length of $\vec{\mathbf{c}}$, namely m', which in turn depends on both L and $\mathcal{D}$. Therefore, technically speaking, our multi-identity IBFHE in addition to both the IBFHE and ABFHE constructions of Gentry, Sahai and Waters are not *leveled* in the strict sense.

6.7 Formal Statement and Proof of Theorem 6.4.1

Corollary 6.7.1 (Corollary 5.4 [97])**.** *Let n be a positive integer, and let q be a prime. Let $m \geq 2n \lg q$. Then for all but a $2q^{-n}$ fraction of all $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and for any $s \geq \omega(\sqrt{\log m})$, the distribution of the syndrome $\vec{\mathbf{u}} = \mathbf{A}\vec{\mathbf{e}} \mod q$ is statistically close to uniform over $\mathbb{Z}_q^n$, where $\vec{\mathbf{e}} \sim D_{\mathbb{Z}^m, s}$.*

Theorem 6.4.1. *Let n, m, q be chosen to meet Corollary 6.7.1. Let χ be a B_χ-bounded distribution where B_χ satisfies Theorem 6.2.1. Let* $\mathsf{TrapGen}$ *be an algorithm that generates a statistically uniform matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ together with a basis $\mathbf{S} \in \mathbb{Z}^{m \times m}$ such that $\|\tilde{\mathbf{S}}\| \leq \mathfrak{L}$ except with negligible probability. Let $s \geq \mathfrak{L} \cdot \omega(\sqrt{\log m})$. Let the scheme* GPV *be instantiated with* $\mathsf{TrapGen}$ *and the* $\mathsf{SamplePre}$ *algorithm (with parameter s) described in Section 6.5.1.*

Then the masking system $\mathsf{MS}_{\mathsf{GPV}}$ is selectively secure in the random oracle model (i.e. $\mathsf{MS}_{\mathsf{GPV}}$ meets the security condition of Definition 6.3.1) under the hardness of $LWE_{n,q,\chi}$.

Proof. We prove the theorem by means of a hybrid argument.

Game 0: This is the standard selective security game described in Definition 6.3.1.

Game 1: The following changes are made in this game. Let $\mathsf{id}^* \in \mathcal{I}$ be the adversary's target identity.

1. The matrix $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n \times m}$ is generated as uniformly random.

2. The vector $\vec{\mathbf{z}_{\mathsf{id}^*}} \xleftarrow{\$} \mathbb{Z}_q^n$ is generated as uniformly random.

3. The random oracle H is simulated as follows: if the adversary $\mathcal{A}$ queries H on identity $\mathsf{id} \in \mathcal{I}$, run:

   (a) If $\mathsf{id} = \mathsf{id}^*$, then return $\vec{\mathbf{z}_{\mathsf{id}^*}}$.

   (b) Else if $(\mathsf{id}, \vec{\mathbf{s}_{\mathsf{id}}}, \vec{\mathbf{z}_{\mathsf{id}}}) \in \mathsf{store}$, return $\vec{\mathbf{z}_{\mathsf{id}}}$.

   (c) Else sample $\vec{\mathbf{t}_{\mathsf{id}}} \xleftarrow{\$} D_{\mathbb{Z}^{m'-1},s}$, compute $\vec{\mathbf{z}_{\mathsf{id}}} \leftarrow \mathbf{A} \cdot \vec{\mathbf{t}_{\mathsf{id}}} \mod q$, set $\vec{\mathbf{s}_{\mathsf{id}}} \leftarrow (1, -\vec{\mathbf{t}_{\mathsf{id}}}) \in \mathbb{Z}_q^{m'}$, add $(\mathsf{id}, \vec{\mathbf{s}_{\mathsf{id}}}, \vec{\mathbf{z}_{\mathsf{id}}})$ to store and return $\vec{\mathbf{z}_{\mathsf{id}}}$.

   (d) Secret key queries are answered as follows. Suppose $\mathcal{A}$ queries a secret key for identity $\mathsf{id} \neq \mathsf{id}^*$. We assume w.l.o.g. that $\mathcal{A}$ has first queried H on id. In response to the query, $\vec{\mathbf{s}_{\mathsf{id}}}$ is returned where $(\mathsf{id}, \vec{\mathbf{s}_{\mathsf{id}}}, \vec{\mathbf{z}_{\mathsf{id}}}) \in \mathsf{store}$.

We claim that $\mathcal{A}$'s view in Game 0 is statistically close to $\mathcal{A}'s$ view in Game 1. The first two changes above follow immediately from the definition of GPV (in particular, the trapdoor basis generation algorithm employed guarantees that a near uniform $\mathbf{A}$ can be generated). In regard to the simulation of H, Corollary 6.7.1 implies that the vector $H(\mathsf{id})$ when $\mathsf{id} \neq \mathsf{id}^*$ is statistically close to uniform. Finally, with regard to the distribution of secret keys, Lemma 5.2 from [97] states that a preimage $\vec{\mathbf{t}_{\mathsf{id}}}$ sampled with $\mathsf{SamplePre}$ (with parameter s) in $\mathsf{GPV.KeyGen}$ is identically distributed to $\vec{\mathbf{t}_{\mathsf{id}}} \sim D_{\mathbb{Z}^{m'-1},s}$ conditioned on $\mathbf{A}_{\mathsf{id}} \cdot \vec{\mathbf{t}_{\mathsf{id}}} = \vec{\mathbf{z}_{\mathsf{id}}} \mod q$. It follows that the secret keys $\vec{\mathbf{s}_{\mathsf{id}}}$ in Game 1 have the same distribution as Game 0.

For $i \in [\ell_q]$:

Game i + 1: This game is the same as the previous game except that Step 1a of $\mathsf{MS_{GPV}.GenUnivMask}$ for iteration i (*only*) is replaced with

$$\vec{\mathbf{x}_\mathbf{i}} \leftarrow \mathsf{BitDecomp}(\vec{\mathbf{t}}).$$

where $\vec{\mathbf{t}} \xleftarrow{\$} \mathbb{Z}_q^{m'}$.

Given an LWE instance $\vec{\mathbf{x}^*} \in \mathbb{Z}_q^{m'}$, one can easily generate $\vec{\mathbf{x}_\mathbf{i}}$ according to Game i or Game $i+1$. Suppose a distinguisher $\mathcal{D}$ has a non-negligible advantage distinguishing

between Game i and Game $i+1$. We can use $\mathcal{D}$ to construct an algorithm $\mathcal{B}$ that can solve an LWE instance. Given an appropriate number of samples from either the distribution $D_0 := \{\{(\vec{\mathbf{u}}_j, \langle \vec{\mathbf{u}}_j, \vec{\mathbf{s}} \rangle + e_j) : \vec{\mathbf{u}_\mathbf{j}} \xleftarrow{\$} \mathbb{Z}_q^n, e)_j \xleftarrow{\$} \chi\} : \vec{\mathbf{s}} \xleftarrow{\$} \mathbb{Z}_q^n\}$ or the distribution $D_1 := \{\{(\vec{\mathbf{u}}_j, \vec{\mathbf{v}}_j) : \vec{\mathbf{u}}_j, \vec{\mathbf{v}_\mathbf{j}} \xleftarrow{\$} \mathbb{Z}_q^n\}\}$, the $\vec{\mathbf{u}}_j$ are used to construct $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and $\vec{\mathbf{z}_{\mathsf{id}^*}} \in \mathbb{Z}_q^n$. The algorithm $\mathcal{B}$ simulates the random oracle H as explained above, and answers secret key queries in the manner described above. Note that the distribution of $\mathbf{A}$ and $\vec{\mathbf{z}_{\mathsf{id}^*}}$ remain unchanged.

The algorithm $\mathcal{B}$ runs the same variant of $\mathsf{MS_{GPV}.GenUnivMask}$ as the previous game. The only difference is that on the i-th iteration, it replaces Step 1a with

$$\vec{\mathbf{x}_\mathbf{i}} \leftarrow \mathsf{BitDecomp}(\vec{\mathbf{x}^*} + (\mu \cdot 2^i, 0, \ldots, 0))$$

where $\vec{\mathbf{x}^*} \in \mathbb{Z}_q^{m'}$ is an LWE challenge vector that is either $\vec{\mathbf{s}} \cdot \vec{\mathbf{z}_{\mathsf{id}^*}} \parallel \mathbf{A} + \vec{\mathbf{e}} \in \mathbb{Z}_q^{m'}$ or a uniformly random $\vec{\mathbf{t}^*} \in \mathbb{Z}_q^{m'}$. In the former case, the view is statistically close to Game i whereas the view in the latter case is statistically close to Game $i+1$. It follows that $\mathcal{B}$ can output $\mathcal{D}$'s guess to solve an LWE instance. The games are thus indistinguishable by the hypothesized hardness of LWE.

As a shorthand for Game $(\ell_q + 1) + (i - \ell_q - 1) \cdot (\eta + 1) + j$, we use the notation Game (i, j) for $\ell_q < i \leq N$ and $j \in [\eta + 1]$.

For $\ell_q < i \leq N$:

For $j \in [\eta]$:

- **Game** $(\mathbf{i}, \mathbf{j})$: This game is the same as the previous game except that we change the way that the j-th row of $\mathbf{B}^{(\mathbf{i})}$ is generated in $\mathsf{MS_{GPV}.GenUnivMask}$. More precisely, Step 2(b)i of algorithm $\mathsf{MS_{GPV}.GenUnivMask}$ is replaced with

  $$\vec{\mathbf{b}_\mathbf{j}^{(\mathbf{i})}} \leftarrow \mathsf{BitDecomp}(\vec{\mathbf{t}})$$

  with $\vec{\mathbf{t}} \xleftarrow{\$} \mathbb{Z}_q^{m'}$. for the *specific case* of the i-th iteration of the outer loop and the j-th iteration of the inner loop.

An analogous argument to the argument made above concerning the indistinguishability of Game $\mathfrak{i}$ and $\mathfrak{i}+1$ for $\mathfrak{i} \in [\ell_q]$ can be made here to show that a non-negligible advantage distinguishing between the games implies a non-negligible advantage against LWE.

Remark At this stage, note that $\mathbf{B}^{(\mathbf{i})}$ from $\mathsf{MS}_{\mathsf{GPV}}.\mathsf{GenUnivMask}$ is uniform over $\mathbb{Z}_q^{\eta \times m'}$; in particular it does not rely on any $\vec{\mathbf{r}}$ associated with a $\vec{\mathbf{y}}_\mathbf{i}$ nor does it rely on μ.

Game $(\mathbf{i}, \eta + \mathbf{1})$: The modification in this game is as follows. Step 2d of $\mathsf{MS}_{\mathsf{GPV}}.\mathsf{GenUnivMask}$ for the i-th iteration is replaced with

$$\vec{\mathbf{y}}_\mathbf{i} \leftarrow \mathsf{Flatten}((\mathsf{BitDecomp}((0, \vec{\mathbf{t}})).$$

with $\vec{\mathbf{t}} \xleftarrow{\$} \mathbb{Z}_q^{m'}$

Once again an analogous LWE-based argument to that above shows that one can embed an LWE challenge when generating $\vec{\mathbf{y}}_\mathbf{i}$ such that indistinguishability between the games implies a non-negligible advantage against LWE.

We conclude the proof by observing that in Game $(N, \eta + 1)$, the plaintext bit μ has been eliminated entirely from the generation of the universal mask U. It follows that an adversary has a zero advantage guessing the challenger's bit b, since no information about b is incorporated in the universal mask U given to the adversary. □

6.8 Application Scenario

We once again return to the medical records scenario from the introduction (Section 1.2.0.1) to illustrate the contributions of this chapter. Suppose the scheme used in this scenario is the leveled IBFHE from this chapter. Each of the three senders can encrypt their data under the appropriate identity string, either "CARDIOLOGY" or "MATERNITY". Assuming the computation to be carried out by the evaluator is known in advance to be

representable by a circuit of depth at most L, then the scheme can be set up to accommodate L levels i.e. the public parameters can be generated accordingly. This is the major difference from our discussion of the same scenario in the previous chapter, where no such limit L was imposed. Instead, in the previous chapter, there was a limit imposed on the number of independent senders. Here we have no limit on the number of independent senders but we merely have a limit on the degree of composition, $\mathcal{D}$. Suppose we have $\mathcal{D} = 100$ such that data sets with up to 100 different attributes can be used in an evaluation. There can be an arbitrary number of senders provided the degree of composition d is at most $\mathcal{D}$.

Recall that the receiver in the scenario has an access policy f with $f(\text{“CARDILOGY”}) = 1$ and $f(\text{“MATERNITY”}) = 1$. We can model disjunctive policies such as this in an identity-based context. For example: to issue the receiver with a secret key for f, the TA can issue a secret key $\mathsf{sk}_{\text{CARDIOLOGY}}$ for identity “CARDIOLOGY” and a secret key $\mathsf{sk}_{\text{MATERNITY}}$ for identity “MATERNITY”; both secret keys constitute the secret key for f. To perform decryption, the receiver first tries to decrypt with $\mathsf{sk}_{\text{CARDIOLOGY}}$ and then if that fails, tries to decrypt with $\mathsf{sk}_{\text{MATERNITY}}$. Provided the depth of the circuit to be evaluated is at most L, our leveled IBFHE fully meets the needs of the scenario[¶].

Our leveled IBFHE can also be used to instantiate the construction from the previous chapter. As a result, we can eliminate the limit L on the depth of the circuit to be evaluated. This is advantageous if we don't know the depth of the circuit prior to generating the public parameters. The trade-off is that we now have a limit on the number of independent senders, say $N = 100$, keeping with our scenario outline from the previous chapter. As long as the number of independent senders is less than that limit, our construction from the previous chapter instantiated with our leveled IBFHE from this chapter fully meets the needs of the scenario.

[¶]Of course a more complex access policy f cannot be handled unless we have a leveled ABFHE with support for more complex access policies. Disjunctive policies is all our leveled IBFHE can natively handle.

6.9 Summary

In this chapter, we presented a multi-identity leveled IBFHE scheme based on the LWE problem. Our construction is built on the single-identity leveled IBFHE of Gentry, Sahai and Waters [98]. We described a compiler to transform an LWE-based IBE into a multi-identity leveled IBFHE provided the IBE satisfies certain properties. One of the requirements is that the IBE admits an abstraction we call a *masking system*. We developed a concrete masking system for the IBE of Gentry, Peikert and Vaikuntanathan (GPV) [97], and showed it to be selectively secure in the random oracle model under LWE. As a result of our compiler, we can compile GPV into a multi-identity leveled IBFHE that is selectively secure in the random oracle model under LWE.

By employing similar ideas to our multi-identity construction, we observe that if one replaces IBE with a public-key scheme, one obtains a multi-key FHE scheme from LWE. This is the first multikey FHE from a well-established problem such as LWE[‖] Furthermore, the decryption circuit for the multikey FHE is in NC^1 i.e. it has depth $O(logN \cdot \lambda)$ where N is the number of independent keys. This is in contrast to the other multikey FHE from the literature [135] whose decryption circuit is in NC^2. This has very positive implications for instantiating our construction from Chapter 5, since by using our multikey FHE, one needs an ABHE scheme supporting circuits in NC^1 as opposed to NC^2.

[‖] When we say multikey FHE, we mean one that supports a non-constant number of keys. López-Alt, Tromer and Vaikuntanathan [135] presented a scheme from LWE for a constant number of keys. Their construction for a non-constant number of keys relies on a non-standard assumption, namely the Decisional Small Polynomial Ratio problem.

Chapter 7

Bootstrapping and Fully Homomorphic Constructions

So far in this thesis, we have been building towards "pure" FHE in the attribute-based setting, but none of our constructions thus far facilitates arbitrary computation on encrypted data. The construction presented in Chapter 5 can evaluate circuits of bounded arity. The construction in Chapter 6 can evaluate circuits of bounded depth (i.e. leveled FHE). The question as to whether attribute-based (or indeed identity-based) "pure" FHE is possible remains open.

In the public-key setting, a leveled FHE scheme can be transformed into a "pure" FHE scheme (i.e. a scheme supporting evaluation of circuits of unlimited depth) via Gentry's *bootstrapping* theorem [93]. However we could not apply the process of bootstrapping to our multi-identity leveled IBFHE in the previous chapter, and thus could not achieve "pure" IBFHE.

In brief, the process of *bootstrapping* entails using the scheme to homomorphically evaluate its own decryption circuit. More precisely, ciphertexts in existing FHE schemes contain a level of "noise". As long as this "noise" remains below a certain threshold, decryption can be performed correctly. The goal of bootstrapping is to return the noise

to a reduced level, so homomorphic operations can continue to be performed. This is achieved by publishing encryptions of the secret key bits, and homomorphically evaluating the scheme's decryption circuit on a "noisy" ciphertext to produce a ciphertext with less noise.

At his talk at CHES/Crypto 2010, Naccache [146] mentioned "identity-based fully homomorphic encryption" as an open problem. As we saw in the previous chapter, Gentry, Sahai and Waters presented the first *leveled* identity-based fully homomorphic encryption (IBFHE) scheme [98]. Furthermore, we extended this result in the previous chapter to the multi-identity setting i.e. our scheme supports evaluation on ciphertexts with different identities.

Achieving fully homomorphic encryption (FHE) in the identity-based setting turned out to be quite a tricky problem, for a variety of reasons, as discussed in Chapter 2.

To the best of our knowledge, there are no known "pure" IBFHE schemes in the literature, since Gentry's bootstrapping theorem from [93] is the only known way of converting a leveled FHE scheme to a "pure" FHE scheme.

In this chapter, we construct the first "pure" IBFHE scheme, which definitively resolves the question raised by Naccache [146] as to the feasability of "identity-based fully homomorphic encryption". Furthermore, we construct a "pure" multi-attribute ABFHE for all polynomial time access policies.

7.0.1 Contributions

7.0.1.1 Construction of "Pure" IBFHE

We construct the first "pure" IBFHE scheme using the technique of "punctured programming" [165], a powerful tool combining an indistinguishability obfuscator [87] with a puncturable pesudorandom function (PRF) [43, 46, 126],

7.0.1.2 "Pure" Multi-Attribute ABFHE for general access policies

We present the first ABFHE that supports evaluation on ciphertexts with different attributes. This scheme is also the first "pure" ABFHE.

7.0.1.3 A Compiler from leveled IBFHE to "Pure" IBFHE

We exploit indistinguishability obfuscation in constructing a compiler from a leveled IBFHE satisfying certain properties to a bootstrappable, and hence "pure", IBFHE. Our main idea is to include in the public parameters an obfuscation of a program (with the master secret key embedded) so that the evaluator can non-interactively derive an "evaluation key" for any identity. Although our compiler falls short of working with arbitrary leveled IBFHE schemes, we establish sufficient conditions for a leveled IBFHE to satisfy in order for it to be bootstrappable. This leads us to an interesting characterization of compatible schemes, which also encompasses our positive result above.

7.1 Building Blocks

7.1.1 Indistinguishability Obfuscation

Garg et al. [87] recently introduced a candidate construction of an indistinguishability obfuscator based on multi-linear maps. Many of our constructions in this chapter depend on the notion of indistinguishability obfuscation. Here we give a brief overview of its syntax and security definition.

Definition 7.1.1 (Indistinguishability Obfuscation (Based on Definition 7 from [104]))**.** *A uniform PPT machine $i\mathcal{O}$ is called an indistinguishability obfuscator for every circuit class $\{\mathcal{C}_\lambda\}$ if the following two conditions are met:*

- ***Correctness:*** *For every $\lambda \in \mathbb{N}$, for every $C \in \mathcal{C}_\lambda$, for every x in the domain of C, we have that*

$$\Pr C'(x) = C(x) : C' \leftarrow i\mathcal{O}(C) = 1.$$

- ***Indistinguishability:*** *For every $\lambda \in \mathbb{N}$, for all pairs of circuits $C_0, C_1 \in \mathcal{C}_\lambda$, if $C_0(x) = C_1(x)$ for all inputs x, then for all PPT adversaries $\mathcal{A}$, we have:*

$$|\Pr \mathcal{A}(i\mathcal{O}(C_0)) = 1| - |\Pr \mathcal{A}(i\mathcal{O}(C_1)) = 1| \leq \mathsf{negl}(\lambda).$$

7.1.2 Puncturable Pseudorandom Functions

A puncturable pseudorandom function (PRF) is a constrained PRF $(\mathsf{Key}, \mathsf{Eval})$ with an additional PPT algorithm $\mathsf{Puncture}$. Let $n(\cdot)$ and $m(\cdot)$ be polynomials. Our definition here is based on [104] (Definition 3.2). A PRF key K is generated with the PPT algorithm Key which takes as input a security parameter λ. The Eval algorithm is deterministic, and on input a key K and an input string $x \in \{0,1\}^{n(\lambda)}$, outputs a string $y \in \{0,1\}^{m(\lambda)}$.

A puncturable PRF allows one to obtain a "punctured" key $K' \leftarrow \mathsf{Puncture}(K, S)$ with respect to a subset of input strings $S \subset \{0,1\}^{n(\lambda)}$ with $|S| = \mathsf{poly}(\lambda)$. It is required that $\mathsf{Eval}(K, x) = \mathsf{Eval}(K', x) \quad \forall x \in \{0,1\}^{n(\lambda)} \setminus S$, and for any poly-bounded adversary $(\mathcal{A}_1, \mathcal{A}_2)$ with $S \leftarrow \mathcal{A}_1(1^\lambda) \subset \{0,1\}^{n(\lambda)}$ and $|S| = \mathsf{poly}(\lambda)$, any key $K \leftarrow \mathsf{Key}(1^\lambda)$, any $K' \leftarrow \mathsf{Puncture}(K, S)$, and any $x \in S$, it holds that

$$\Pr \mathcal{A}_2(K', x, \mathsf{Eval}(K, x)) = 1 - \Pr \mathcal{A}_2(K', x, u) = 1 \leq \mathsf{negl}(\lambda)$$

where $u \xleftarrow{\$} \{0,1\}^{m(\lambda)}$.

7.2 Construction of "Pure" IBFHE

We now construct a "pure" IBFHE from indistinguishability obfuscation. The main idea is to use the technique of punctured programming, which involves using indistinguishability obfuscation together with a puncturable PRF. In our case, we use the puncturable PRF for the derivation of a user's public key from her identity. Moreover, a unique key pair for a public-key encryption (PKE) scheme can be associated with every identity. If the PKE scheme is also "pure" fully-homomorphic, then we obtain a "pure" IBFHE

scheme. Let $\mathcal{E}_{\mathsf{FHE}} := (\mathsf{Gen}, \mathsf{Encrypt}, \mathsf{Decrypt}, \mathsf{Eval})$ be a public-key FHE. We denote by $\mathcal{PK}_{\mathsf{FHE}}$ and $\mathcal{SK}_{\mathsf{FHE}}$ its public-key and private-key space respectively. Consider the following function $F_{\mathsf{MapPK}} \colon \mathcal{I} \to \mathcal{PK}_{\mathsf{FHE}}$ that maps an identity $\mathsf{id} \in \mathcal{I}$ to a public key for $\mathcal{E}_{\mathsf{FHE}}$:

Program $F_{\mathsf{MapPK}}(\mathsf{id})$:

1. Compute $r_{\mathsf{id}} \leftarrow \mathsf{PRF.Eval}(K, \mathsf{id})$.
2. Compute $(\mathsf{pk}_{\mathsf{id}}, \mathsf{sk}_{\mathsf{id}}) \leftarrow \mathcal{E}_{\mathsf{FHE}}.\mathsf{Gen}(1^\lambda; r_{\mathsf{id}})$.
3. **Output** $\mathsf{pk}_{\mathsf{id}}$

A formal description of a scheme $\hat{\mathcal{E}}_*$ that uses an obfuscation of F_{MapPK} is as follows.

- $\hat{\mathcal{E}}_*.\mathsf{Setup}(1^\lambda)$: Compute $K \leftarrow \mathsf{PRF.Key}(1^\lambda)$, compute obfuscation $H \leftarrow i\mathcal{O}(F_{\mathsf{MapPK}})$ of F_{MapPK} with K embedded. Output (H, K) (note that H constitutes the public parameters and K constitutes the master secret key).
- $\hat{\mathcal{E}}_*.\mathsf{KeyGen}(K, \mathsf{id})$: Compute $r_{\mathsf{id}} \leftarrow \mathsf{PRF.Eval}(K, \mathsf{id})$, compute $(\mathsf{pk}_{\mathsf{id}}, \mathsf{sk}_{\mathsf{id}}) \leftarrow \mathcal{E}_{\mathsf{FHE}}.\mathsf{Gen}(1^\lambda; r_{\mathsf{id}})$, and output $\mathsf{sk}_{\mathsf{id}}$.
- $\hat{\mathcal{E}}_*.\mathsf{Encrypt}(H, \mathsf{id}, m)$: Compute $\mathsf{pk}_{\mathsf{id}} \leftarrow H(\mathsf{id})$ and output $\mathcal{E}_{\mathsf{FHE}}.\mathsf{Encrypt}(\mathsf{pk}_{\mathsf{id}}, m)$.
- $\hat{\mathcal{E}}_*.\mathsf{Decrypt}(\mathsf{sk}_{\mathsf{id}}, c)$: Output $\mathcal{E}_{\mathsf{FHE}}.\mathsf{Decrypt}(\mathsf{sk}_{\mathsf{id}}, c)$.
- $\hat{\mathcal{E}}_*.\mathsf{Eval}(H, C, c_1, \ldots, c_\ell)$: Compute $\mathsf{pk}_{\mathsf{id}} \leftarrow H(\mathsf{id})$ and output $\mathcal{E}_{\mathsf{FHE}}.\mathsf{Eval}(\mathsf{pk}_{\mathsf{id}}, C, c_1, \ldots, c_\ell)$.

Lemma 7.2.1. *Assuming indistinguishability obfuscation, a secure puncturable PRF and an* IND-CPA*-secure public-key FHE scheme* $\mathcal{E}_{\mathsf{FHE}}$*, the scheme* $\hat{\mathcal{E}}_*$ *is* IND-sID-CPA *secure.*

Proof. We prove the lemma via a hybrid argument.

Game 0: This is the real system.

Game 1: This is the same as Game 0 except for the following changes. Suppose the adversary chooses id^* as the identity to attack. We compute $K' \leftarrow \mathsf{PRF.Puncture}(K, \mathsf{id}^*)$ and answer secret key requests using K' instead of K.

The adversary cannot detect any difference between the games since for all $\mathsf{id} \neq \mathsf{id}^*$, it holds that $\mathsf{PRF.Eval}(K, \mathsf{id}) = \mathsf{PRF.Eval}(k', \mathsf{id})$.

Game 2 This is the same as Game 1 except that we make the following changes to F_{MapPK}:

- Add before step 1: if $\mathsf{id} = \mathsf{id}^*$, then output $\mathsf{pk}_{\mathsf{id}^*}$ (defined below). Else run steps 1 - 3.
- Replace K with K'.

where $(\mathsf{pk}_{\mathsf{id}^*}, \mathsf{sk}_{\mathsf{id}}) \leftarrow \mathcal{E}_{\mathsf{FHE}}.\mathsf{Gen}(1^\lambda; r_{\mathsf{id}^*})$ and $r_{\mathsf{id}^*} \leftarrow \mathsf{PRF.Eval}(K, \mathsf{id}^*)$.

Observe that the modified function is identical to F_{MapPK}, and due to the security of indistinguishability obfuscation, their respective obfuscations are thus computationally indistinguishable.

Game 3: This is the same as Game 2 except that we change how $\mathsf{pk}_{\mathsf{id}^*}$ is computed. We do this indirectly by changing how r_{id^*} is computed instead. More precisely, we choose a uniformly random string $r_{\mathsf{id}^*} \xleftarrow{\$} \{0,1\}^m$ where m is the length of the pseudorandom outputs of $\mathsf{PRF.Eval}$ i.e. $m = |\mathsf{PRF.Eval}(K, \mathsf{id}^*)|$.

By the security of the puncturable PRF, we have that

$$\{(K', \mathsf{id}^*, \mathsf{PRF.Eval}(K, \mathsf{id}^*)\} \underset{C}{\approx} \{(K', \mathsf{id}^*, r) : r \xleftarrow{\$} \{0,1\}^m)\}.$$

It follows that Game 2 and Game 3 are computationally indistinguishable.

Game 4: This is the same as Game 3 except that we replace the challenge ciphertext with an encryption of a random message. The adversary has a zero advantage in this game.

If a PPT adversary $\mathcal{A}$ can distinguish between Game 3 and Game 4, then there exists a PPT adversary $\mathcal{B}$ that can use $\mathcal{A}$ to attack the $\mathsf{IND\text{-}sID\text{-}CPA}$ security of $\mathcal{E}_{\mathsf{FHE}}$. When $\mathcal{B}$

receives the challenger's public key pk, it sets $\mathsf{pk}_{\mathsf{id}^*} \leftarrow \mathsf{pk}$ where id^* is the target identity chosen by $\mathcal{A}$. Note that $\mathsf{pk}_{\mathsf{id}^*}$ has the same distribution as that from Game 3. Suppose m_0 and m_1 are the messages chosen by $\mathcal{A}$. $\mathcal{B}$ samples a random bit b, and also samples a random message $m' \xleftarrow{\$} \mathcal{M}$, and sends (m_b, m') to the IND-CPA challenger, who responds with a challenge ciphertext c^*. Then $\mathcal{B}$ relays c^* to $\mathcal{A}$ as the challenge ciphertext. Let b' denote the random bit chosen by the challenger. If $b' = 0$, then the game is distributed identically to Game 3; otherwise if $b' = 1$ it is distributed identically to Game 4. It follows that any $\mathcal{A}$ with a non-negligible advantage distinguishing between the games contradicts the hypothesized IND-CPA security of $\mathcal{E}_{\mathsf{FHE}}$. □

Theorem 7.2.1. *Assuming indistinguishability obfuscation, one-way functions and fully homomorphic encryption, there exists an* IND-sID-CPA*-secure "pure" IBFHE scheme i.e. an identity-based scheme that can homomorphically evaluate all circuits.*

Proof. The construction $\hat{\mathcal{E}}_*$ is fully homomorphic if the underlying PKE scheme $\mathcal{E}_{\mathsf{FHE}}$ is fully homomorphic. Lemma 7.2.1 shows that $\hat{\mathcal{E}}_*$ is IND-sID-CPA secure assuming indistinguishability obfuscation, one-way functions and the IND-CPA security of $\mathcal{E}_{\mathsf{FHE}}$. The result follows. □

Note that because our IBFHE relies on (public-key) "pure" FHE and because all constructions of "pure" FHE that we know of require a circular security assumption, it naturally follows that our IBFHE also requires a circular security assumption. Furthermore, our IBFHE is only shown to be selectively secure. While there is a generic transformation from a selectively-secure IBE to a fully-secure IBE [34], this transformation incurs a degradation in security by a factor of 2^s where $s = |\mathcal{I}|$ is the size of the identity space. Obtaining a fully secure "pure" IBFHE "directly" remains an open problem. These remarks also apply to our attribute-based constructions, which are presented next.

7.3 "Pure" Multi-Attribute ABFHE for General Access Policies

7.3.1 Single-Attribute Construction

The scheme $\hat{\mathcal{E}}_*$ can be extended to an Attribute Based Encryption (ABE) scheme. Recall that in a (key-policy) ABE scheme, an encryptor associates an attribute $a \in \mathbb{A}$ with her message, whereas a decryptor can only successfully decrypt a ciphertext with attribute $a \in \mathbb{A}$ if he holds a secret key for a policy (i.e. a predicate) $f\colon \mathbb{A} \to \{0,1\}$ with $f(a) = 1$. We denote by $\mathbb{F}$ the class of supported policies. Therefore, in an ABE scheme, the trusted authority issues secret keys for policies instead of identities as in IBE. The fundamental difference is that there is no longer a one-to-one correspondence between attributes and policies (which is the case in IBE).

Beyond notationally replacing the set of identities $\mathcal{I}$ with a set of attributes $\mathbb{A}$ in $\hat{\mathcal{E}}_*$, nothing changes for setup, encryption and evaluation. The primary change takes place with respect to key generation. In KeyGen, given a punctured PRF key K' and a policy $f \in \mathbb{F}$, we return as the secret key for f an obfuscation $d_f \leftarrow i\mathcal{O}(F_{\mathsf{MapSK}_f})$, where F_{MapSK_f} is defined as follows with respect to f:

Program $F_{\mathsf{MapSK}_f}(\mathsf{a})$:

1. If $f(a) = 0$, **Output** $\perp$.
2. Compute $r_a \leftarrow \mathsf{PRF.Eval}(K, a)$.
3. Compute $(\mathsf{pk}_a, \mathsf{sk}_a) \leftarrow \mathcal{E}_{\mathsf{FHE}}.\mathsf{Gen}(1^\lambda; r_a)$.
4. **Output** sk_a.

Decryption is straightforward: given a secret key for f, namely the obfuscation d_f, a decryptor simply computes $\mathsf{sk}_a \leftarrow d_f(a)$ (she can store sk_a for future use to avoid re-evaluating d_f) where a is the attribute associated with ciphertext c, and then computes

the plaintext $m \leftarrow \mathcal{E}_{\mathsf{FHE}}.\mathsf{Decrypt}(\mathsf{sk}_a, c)$. Hence, we obtain an ABFHE for general-purpose policies f.

7.3.2 Multi-Attribute Construction

One of the limitations of our ABFHE construction is that homomorphic evaluation is restricted to the single-attribute setting. In other words, homomorphic evaluation is only supported for ciphertexts with the same attribute. In fact, this is the case for the only known leveled ABFHE in the literature [98].

Let $\mathcal{D}$ be an upper bound on the number of distinct attributes supported when homomorphically evaluating a circuit. In multi-attribute ABFHE, the main syntactic change is that the size of an evaluated ciphertext is allowed to depend on $\mathscr{d} \leq \mathcal{D}$, which is the number of distinct attributes used in an evaluation. Also, $\mathcal{D}$ is a parameter that is specified in advance of generating the public parameters.

To be more precise, consider ciphertexts $c_1, \ldots, c_\ell$ passed to the Eval algorithm. Each of the ℓ ciphertexts may have a different attribute. Thus there is at most $\mathscr{d} \leq \ell$ distinct attributes in this set. As long as $\mathscr{d} \leq \mathcal{D}$, the scheme can handle the evaluation of a circuit. Let $c' \leftarrow \mathsf{Eval}(\mathsf{PP}, C, c_1, \ldots, c_\ell)$ be an evaluated ciphertext, where PP is the public parameters and C is a circuit. It is required that $|c'| = \mathsf{poly}(\lambda, \mathscr{d})$.

In the previous chapter, we developed a multi-identity leveled IBFHE. We noted that we could not extend our techniques therein to achieve leveled IBFHE for more complex policies than IBE; multi-attribute ABFHE remained elusive. Now recall our multi-attribute ABHE construction from Chapter 5. This construction employed multikey FHE in an integral way to achieve FHE for bounded-arity circuits, assuming the existence of an ABHE for polylog circuits. It turns out the essence of this construction is of significant import here also. The major limitation of our construction from Chapter 5 is that the arity of the circuit is bounded. More concretely, this means that the number of independent senders who contribute data is a priori bounded, which is not ideal in many scenarios. Instead, we would like to evaluate circuits of unbounded arity and

instead merely place a limit on the degree of composition. Furthermore, to fulfill the requirements of ABHE, we would like the size of evaluated ciphertexts to depend only on the degree of composition. We now show how to use multikey FHE and the techniques discussed so far in the chapter to accomplish multi-attribute "pure" ABFHE for general-purpose access policies. This is the ultimate contribution of this thesis; it shows that we can maximize all facets of ABHE: (1). supported circuits; (2). supported access policies; and (3). composition.

Multi-Attribute ABFHE can be viewed as an attribute-based analog to multi-key FHE from [135]. In multi-key FHE, the size of evaluated ciphertexts depends on an a priori fixed parameter M, which represents the number of independent keys tolerated by the scheme. Hence data encrypted under at most M distinct public keys $\mathsf{pk}_1, \ldots, \mathsf{pk}_M$ can be used together in an evaluation.

We exploit multi-key FHE to construct a multi-attribute ABFHE. Our scheme is very similar to our (single-attribute) ABFHE scheme described above in Section 7.3. The main change is that $\mathcal{E}_{\mathsf{FHE}}$ is replaced with a multi-key FHE scheme $\mathcal{E}_{\mathsf{MKFHE}}$ (such as the NTRU-based scheme from [135]). The latter is instantiated with parameter M supplied when generating the public parameters. Suppose a collection of input ciphertexts $c_1, \ldots, c_\ell$ are associated with a set of $k \leq M$ distinct attributes $a_1, \ldots, a_k \in \mathbb{A}$. Hence, an evaluated ciphertext c' is associated with a *set* $A = \{a_1, \ldots, a_k\}$.

Decryption depends on the intended semantics. One may wish that the decryption process is collaborative i.e. there may not be a single f that satisfies all k attributes, but users may share secret keys for a set of policies $\{f\}$ that "covers all" k attributes. Alternatively, and this is the approach taken in [62], it may be desired that a user can only decrypt c' if she has a secret key for a policy f that satisfies *all* k attributes. We take the former approach here because as discussed in Chapter 3 (Section 3.1.1.2), this model gives more flexibility, thus allowing more applications.

In our scheme, secret keys are the same as those in the single-attribute scheme from the previous section; that is, a secret key for f is on obfuscation $d_f \leftarrow i\mathcal{O}(F_{\mathsf{MapSK}_f})$

of the program F_{MapSK_f}. Let c' be a ciphertext associated with k distinct attributes $a_1, \ldots, a_k$. To decrypt c' with a secret key d_f for policy f, a decryptor does the following: if $f(a_i) = 1$ for every $i \in [k]$, compute $\mathsf{sk}_{a_i} \leftarrow d_f(a_i)$, and output $m \leftarrow \mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Decrypt}(\{\mathsf{sk}_{a_1}, \ldots, \mathsf{sk}_{a_k}\}, c')$; otherwise output $\perp$. Suppose a user has secret keys for t different policies $f_1, \ldots, f_t$. As long as every attribute a_i satisfies at least one of these policies, the user can obtain the corresponding sk_{a_i} and decrypt the $\mathcal{E}_{\mathsf{MKFHE}}$ ciphertext c in the same manner as above.

7.3.2.1 sel-EVAL-SIM Security

To prove sel-EVAL-SIM security of our multi-attribute ABFHE above, we need to make an additional assumption. We require the multikey FHE scheme $\mathcal{E}_{\mathsf{MKFHE}}$ to satisfy a stronger notion than IND-CPA security that we call *multikey privacy*. Informally, this means that an attacker cannot distinguish which of two known sets of public keys was used to encrypt a given ciphertext provided both sets have the same cardinality and both sets contain at least one public key whose corresponding secret key is unknown to the attacker. The formal security game is captured in the following experiment.

Let $\mathcal{O}$ be an oracle that returns a key tuple $(\mathsf{pk}, \mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{Gen}(1^\lambda)$ for the multikey FHE scheme $\mathcal{E}_{\mathsf{MKFHE}}$ when queried for an index $i \in \mathbb{N}$. It returns the same response when queried on the same index. Similarly, let $\mathcal{O}'$ be an oracle that returns a key tuple $(\mathsf{pk}, \mathsf{vk})$ where $(\mathsf{pk}, \mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{Gen}(1^\lambda)$. Both oracles generate fresh keys for $\mathcal{E}_{\mathsf{MKFHE}}$ with $\mathcal{O}$ providing both public and secret information associated with the key, and $\mathcal{O}'$ providing only public information. The adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ is a pair of PPT algorithms.

Experiment $\mathsf{MKPriv}(\mathcal{A}_1, \mathcal{A}_2)$:

1. $(\mathsf{state}, C, m_{0,1}, \ldots, m_{0,\ell}, m_{1,1}, \ldots, m_{1,\ell}, v_{0,1}, \ldots, v_{0,\ell}, v_{1,1}, \ldots, v_{1,\ell}) \leftarrow \mathcal{A}_1^{\mathcal{O},\mathcal{O}'}(1^\lambda)$.

2. Suppose $\mathcal{A}_1$ makes a total of $Q = q + q'$ queries. Assume w.l.o.g. that $\mathcal{A}_1$ queries $\mathcal{O}$ on $1, \ldots, q$ to yield $(\mathsf{pk}_i, \mathsf{sk}_i, \mathsf{vk}_i)$ for $1 \leq i \leq q$, and it queries $\mathcal{O}'$ on $q+1, \ldots, Q$ to yield $(\mathsf{pk}_i, \mathsf{sk}_i)$ for $q+1 \leq i \leq Q$.

3. Abort with a random bit unless the following conditions are met for $i \in \{0,1\}$:

   (a) $v_{i,1}, \ldots, v_{i,\ell} \in [Q]$.

   (b) $v_{i,j} > q$ for some j (this implies that $q' \geq 1$ and at least one key to be used in evaluation came from $\mathcal{O}'$).

4. Generate a uniformly random bit $b \xleftarrow{\$} \{0,1\}$.

5. Compute $c_{i,j} \leftarrow \mathsf{Enc}(\mathsf{pk}_{v_{i,j}}, m_{i,j})$ for $i \in \{0,1\}$ and $j \in [\ell]$.

6. Compute

   $c^* \leftarrow \mathsf{Eval}(C, (c_{b,1}, \mathsf{vk}_{v_{b,1}}), \ldots, (c_{b,\ell}, \mathsf{vk}_{v_{b,\ell}}))$.

7. $b' \leftarrow \mathcal{A}_2(\mathsf{state}, c^*, c_{0,1}, \ldots, c_{0,\ell}, c_{1,1}, \ldots, c_{1,\ell})$.

8. Output 1 if $b' = b$ and output 0 otherwise.

A multikey FHE scheme is said to be *multikey-private* if for any pair of PPT algorithms $(\mathcal{A}_1, \mathcal{A}_2)$, it holds that

$$\mathsf{Pr}[\mathsf{MKPriv}(\mathcal{A}_1, \mathcal{A}_2) \Rightarrow 1] - \frac{1}{2} < \mathsf{negl}(\lambda).$$

Observe that this formulation of multikey FHE privacy requires Eval to be nondeterministic. Otherwise, it is trivial for an adversary to guess the challenger's random coin by merely calling Eval with both sequences of ciphertexts.

Lemma 7.3.1. *There exists a multikey FHE scheme from [135] that is multikey-private under the Decisional Small Polynomial Ratio (DSPR) and Ring Learning With Errors (R-LWE) assumptions.*

Proof. Ciphertexts in this scheme are indistinguishable from uniform elements in a ring provided a party does not have secret keys for all keys used. □

To help the reader follow the proof below, we first recall the definition of sel-EVAL-SIM security from Chapter 3 (Section 3.3.2). The goal is that there exists a simulator $\mathcal{S}$ such that no adversarial triple of PPT algorithms $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3)$ can distinguish between the real distribution (which uses the real system) and ideal distribution (which uses $\mathcal{S}$). To recap: the algorithm $\mathcal{B}_1$ outputs a set of attributes $A = \{\mathfrak{a}_1, \ldots, \mathfrak{a}_{\mathcal{A}}\} \subseteq \mathbb{A}$; the algorithm $\mathcal{B}_2$ takes as input the public parameters PP and outputs a circuit C, a sequence of pairs $(a_1, \mu_1), \ldots, (a_\ell, \mu_\ell)$ with $a_i \in A$ and $\mu_i \in \mathcal{M}$ for $i \in [\ell]$, and state st; the algorithm $\mathcal{B}_3$ takes as input state st, a challenge ciphertext c^* and a sequence of ciphertexts $c_1, \ldots, c_\ell$ - it outputs a guess bit $b \in \{0, 1\}$.

Theorem 7.3.1. *Our multi-attribute ABFHE scheme, instantiated with a multikey FHE that is multikey private, is* sel-EVAL-SIM *secure.*

Proof. We show sel-EVAL-SIM security with respect to the following simulator $\mathcal{S}$. The simulator $\mathcal{S}$, on input public parameters PP, circuit C and set of attributes $A = \{\mathfrak{a}_1, \ldots, \mathfrak{a}_{\mathcal{A}}\}$ performs the steps: generate $\mathcal{A}$ key triples for the multikey FHE: $(\mathsf{pk}_i, \mathsf{vk}_i, \mathsf{sk}_i) \leftarrow \mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Gen}(1^\lambda)$ for $i \in [\mathcal{A}]$; generate random bits $b_i \xleftarrow{\$} \{0, 1\}$ for $i \in [\ell]$; choose $v_1, \ldots, v_\ell \in [\mathcal{A}]$, encrypt $c_i \leftarrow \mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Encrypt}(\mathsf{pk}_{v_i}, b_i)$ for $i \in [\ell]$ and output $c' \leftarrow \mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Eval}(C, (c_1, \mathsf{vk}_{v_1}), \ldots, (c_\ell, \mathsf{vk}_{v_\ell}))$.

Suppose there is an adversary $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3)$ that attacks the sel-EVAL-SIM security of our multi-attribute ABFHE. Then there is an adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ that attacks the multikey privacy of $\mathcal{E}_{\mathsf{MKFHE}}$. The algorithm $\mathcal{A}_1$ runs as follows:

- Run $\mathcal{B}_1$ to get attributes $A = \{\mathfrak{a}_1, \ldots, \mathfrak{a}_{\mathcal{A}}\}$.
- Choose random $k \xleftarrow{\$} [\mathcal{A}]$.
- Query $\mathcal{O}$ for all $i \in [\mathcal{A}] \setminus \{k\}$ to get $(\mathsf{pk}_i, \mathsf{vk}_i, \mathsf{sk}_i)$. Query $\mathcal{O}'$ on k to get $(\mathsf{pk}_k, \mathsf{vk}_k)$.
- Run $\mathcal{B}_2$ to get $(C, (a_1, \mu_1), \ldots, (a_\ell, \mu_\ell), \mathsf{st})$. $\mathcal{B}_2$'s secret key queries are handled as follows:

- If f is queried with $f(\mathfrak{a}_k) = 1$, abort with a random bit.
- An obfuscation of a modified version of f_{MapSK_f} is returned. In the modified version, for each $i \in [\mathcal{d}] \setminus \{k\}$. the secret key sk_i is hard-coded for input $\mathfrak{a}_i$. Due to the indistinguishability property, $\mathcal{B}_2$'s view is indistinguishable from the original view.

- Let $v_{0,i}$ be the index such that $a_i = \mathfrak{a}_{v_{0,i}}$ for $i \in [\ell]$.
- Choose $v_{1,1}, \ldots, v_{1,\ell} \in [\mathcal{d}]$.
- Choose random $b_1, \ldots, b_\ell \xleftarrow{\$} \{0,1\}$.
- Output $(C, \mu_1, \ldots, \mu_\ell, b_1, \ldots, b_\ell, v_{0,1} \ldots, v_{0,\ell}, v_{1,1}, \ldots, v_{1,\ell}, \mathsf{state} := \mathsf{st})$.

The probability that $\mathcal{A}_1$ does not abort is at least $1/\mathcal{d}$. To see this, observe that there must be at least one attribute that satisfies no queried policy. The probability that this attribute is $\mathfrak{a}_k$ is $1/\mathcal{d}$.

The algorithm $\mathcal{A}_2$ receives as input $\mathsf{state} := \mathsf{st}$, a challenge ciphertext c^* and two sequences of ciphertexts $c_{0,1}, \ldots, c_{0,\ell}$ and $c_{1,1}, \ldots, c_{1,\ell}$. The algorithm $\mathcal{A}_2$ runs as follows:

- It runs $\gamma \leftarrow \mathcal{B}_3(\mathsf{st}, c^*, c_{0,1}, \ldots, c_{0,\ell})$.
- It outputs $\mathcal{B}_3$'s guess $\gamma \in \{0,1\}$.

Recall that the challenge c^* is generated from $(c_{b,1}, \mathsf{vk}_{v_{b,1}}) \ldots, (c_{b,\ell}, \mathsf{vk}_{v_{b,\ell}})$ for either $b = 0$ or $b = 1$. If $b = 0$, then c^* is generated as in the real system. If $b = 1$, then c^* generated in an identical manner to the simulator $\mathcal{S}$. Therefore, if $\mathcal{B}_3$ has a non-negligible advantage against sel-EVAL-SIM security, then this translates into a non-negligible advantage against multikey privacy. □

7.3.2.2 Discussion

The implications of the "pure" multi-attribute ABFHE presented in this chapter are significant since it maximizes all facets of ABHE, namely supported class of circuits,

supported class of access policies and composition (multi-attribute). It maximizes the supported class of access policies because it supports all polynomial time access policies. The proof of semantic security follows directly from the proof of security of our "pure" IBFHE - the only difference is that an FHE scheme is replaced with a multikey FHE, and hence we rely on the IND-CPA security of the latter instead of the former. However, we can only prove the scheme selectively secure. To achieve full security, we must employ the reduction due to Boneh and Boyen [34] as described in Chapter 2, Section 2.2.3.2, which converts a selectively secure scheme into one that is fully secure. However this conversion looses an exponential factor in the tightness of the reduction, which basically means that the public parameters must be set with security parameter $O(2 \cdot \lambda)$ to achieve security of 2^{λ} against an attacker. Hence the public parameters are larger, making the scheme less efficient. However, since our scheme serves mainly as a theoretical possibility result, the applicability of the reduction of Boneh and Boyen means that we can argue that there exists a fully secure "pure" multi-attribute ABFHE.

Similarly, after applying the Boneh and Boyen conversion, we yield a scheme that is EVAL-SIM secure instead of sel-EVAL-SIM secure. The importance of the EVAL-SIM security definition is that an adversary cannot link a collection of input ciphertexts to a ciphertext resulting from an evaluation. The adversary may learn the circuit that was used and the d attributes that were used in the evaluation, but no more than that*.

*In fact, if the underlying multikey FHE used to instantiate our construction satisfies circuit privacy and multikey privacy (which is the case for the multikey FHE from [135]), then we can even achieve a stronger security property than our definition of EVAL-SIM security, namely one where all the adversary learns about an evaluated ciphertext is its degree of composition.

7.4 Making Existing Leveled IBFHE Schemes Bootstrappable

Indistinguishability obfuscation is computationally expensive and our constructions in this chapter require the encryptor and evaluator to run an obfuscated program for encryption and evaluation respectively. It is desirable to curtail this expense to the infrequent times when bootstrapping is in fact needed. In Appendix D, we present a compiler for this case, and identify sufficient conditions for the compiler to be applicable.

7.5 Application Scenario

We revisit the medical records scenario from the introduction (Section 1.2.0.1) to illustrate our multi-attribute ABFHE from this chapter. In the scenario, three senders contribute encrypted data to an evaluator who performs computation on the encrypted data. Two of the senders encrypt their data under the attribute "CARDIOLOGY" while the other sender encrypts his data under the attribute "MATERNITY". A receiver with access policy f with $f(\text{"CARDIOLOGY"}) = 1$ and $f(\text{"MATERNITY"}) = 1$ can decrypt the result of the computation C. A multi-attribute ABFHE fully accommodates this scenario without limitations. More precisely, no limit is placed on the computation C, which may be arbitrary. Furthermore, there is no limit on the number of independent senders (or the arity of the circuit), as we have seen in the construction from Chapter 5. Finally, the access policy f may be arbitrarily complex, as long as it runs in polynomial time because our multi-attribute ABFHE supports all polynomial-time access policies. Hence, our multi-attribute ABFHE is the ideal candidate for achieving the goals of the scenario because it places no limitations on either the computation to be performed or the complexity of the receiver's access policy. The only necessary bound is on the degree of composition. In this scenario, the degree of composition is 2 (the distinct attributes are "CARDIOLOGY" and "MATERNITY"). The maximum degree of composition sup-

ported may be set to (say) 100. This is the only limiting factor on the evaluation to be performed with our multi-attribute ABFHE, but this is the case for all constructions in this thesis since it is an inherent property of multi-attribute ABHE.

7.6 Summary

In this chapter, a construction of "pure" IBFHE was presented based on puncturable pseudorandom functions (PRF) and indistinguishability obfuscation. We extended this result to obtain a construction of "pure" ABFHE for all polynomial time access policies. Following on from this, and leveraging multikey FHE in a similar manner to our construction from Chapter 5, we obtained a "pure" multi-attribute ABFHE for all polynomial-time access policies. We showed this scheme to be both IND-sel-CPA and sel-EVAL-SIM secure.

We then presented a compiler to transform a leveled IBFHE satisfying certain conditions into a bootstrappable, and hence "pure", IBFHE.

Chapter 8

Conclusions and Future Work

In this thesis we considered homomorphic encryption in the attribute-based setting. We defined the formal syntax of attribute-based homomorphic encryption (ABHE) in Chapter 3. Our definition of this primitive is general enough to capture homomorphic evaluation over ciphertexts with different attributes. Using the terminology put forward in this thesis, the number of *distinct* attributes in a given evaluation is referred to as its *degree of composition*. A compactness condition was defined for ABHE that requires the size of a ciphertext resulting from an evaluation (called an *evaluated ciphertext*) to depend polynomially on the security parameter and the degree of composition. Our definition endows an ABHE scheme with two parameters $\mathcal{D}$ and $\mathcal{K}$. The first gives the maximum degree of composition supported by the scheme, while the latter gives the maximum number of decryption keys that can be passed to the decryption algorithm. The latter in turn determines the *model of decryption*. In Chapter 3, two such models were defined: the first model captures collaborative decryption (this corresponds to the case of $1 < \mathcal{K} \leq \mathcal{D}$) in which a collection of policies that between them "cover all" the attributes associated with a ciphertext, are sufficient to decrypt the ciphertext; the second model captures an "atomic" notion (this corresponds to the case $\mathcal{K} = 1$) where a decryptor needs a single policy satisfying every attribute associated with a ciphertext

in order to decrypt that ciphertext.

Following on from this, we defined, in addition to the standard indistinguishability security game, a *simulation*-based definition of security - abbreviated as EVAL-SIM. This captures the desirable property that a number of senders should not be able to tell whether their input ciphertexts were used in a particular evaluation to produce a given evaluated ciphertext, provided they cannot decrypt the evaluated ciphertext. So an evaluated ciphertext may therefore leak (to any party unauthorized to decrypt it) its associated attributes along with the circuit that was evaluated, but nothing more. This definition can be further strengthened by not leaking any more than the degree of composition (let's call it d), thus hiding the circuit (i.e. computational circuit privacy) and the d distinct attributes (i.e. attribute-privacy).

Equipped with these notions, this thesis then addressed the two primary forms of homomorphic encryption, namely group homomorphic encryption (GHE) and fully homomorphic encryption (FHE), in an attribute-based context. In Chapter 4, attribute-based group homomorphic encryption (ABGHE) was formally defined. Schemes from the literature that meet this definition were discussed; these schemes admit a multiplicative homomorphism. It was observed that there are no additively homomorphic schemes in the literature. The chapter presents the first such scheme: an identity-based XOR-homomorphic scheme, which is shown to be semantically secure in the random oracle model under the quadratic residuosity assumption. XOR-homomorphic schemes have been employed in many applications including biometric authentication, sealed-bid auctions and a 2-round protocol for the millionaire's problem - an identity-based scheme allows these applications to be adapted to the identity-based setting, enabling the benefits that setting provides. Our scheme can also be generalized from supporting addition modulo 2 (i.e. XOR) to addition modulo m for small m; the ciphertext size grows quadratically with m. Consequently, m must be polynomially sized. In future work, it would be valuable to explore more space-efficient additively homomorphic schemes, and move beyond the identity-based functionality to support richer access policies. The

"ideal" additively homomorphic scheme would support superpolynomially-sized m (like Paillier [154] in the public-key setting) and a rich class of access policies (such as the class of Boolean formulas), and be provably fully secure in the standard model under a well-established assumption. Chapter 4 advances towards this goal by giving the first instance of an additively homomorphic ABGHE scheme.

In Chapter 5, we turned our attention towards the second primary form of homomorphic encryption, FHE, and obtained a valuable result. We showed that a variant of attribute-based FHE (ABFHE) where the arity of the supported circuits is bounded by some integer N (which can be specified in advance of generating the public parameters) can be constructed from multi-key FHE and leveled ABFHE. To be more precise, we presented a "compiler" that can transform a leveled ABFHE capable of evaluating Boolean circuits of depth $O(\log(N\lambda))$ into an ABFHE (with the same $\mathcal{D}$) that can evaluate all Boolean circuits with N inputs over the domain $\{0,1\}^w$ for some arbitrarily large w (the parameter w is needed to satisfy the syntax whereas in practice, the N *inputs* can be taken from $\{0,1\}^+$). The result lets us trade "breadth" for "depth" because we can evaluate circuits of unbounded depth. This is important because it circumvents impediments in the attribute-based setting to realizing the technique of *bootstrapping*, which is the only known way of evaluating circuits of unbounded depth. Our result relies on multikey FHE [135], a primitive that facilitates homomorphic evaluation on data encrypted under multiple independently-generated keys.

A constraint on our result from Chapter 5 is that only circuits using inputs from N senders can be evaluated on. On the other hand, since the parameters of the scheme grow with $O(\log N)$, we can choose N to be exponentially large and hence, in practice this will likely suffice for most practical evaluations.

In Chapter 6, our focus turns to constructing a *multi-attribute* leveled ABFHE scheme. By *leveled*, it is meant that the scheme can only evaluate circuits whose depth is at most some a priori fixed parameter L (it can be instantiated for any L). This suffices for many applications because it may be known beforehand that only computations of

a particular complexity ever need to be evaluated. Furthermore, *multi-attribute* means the scheme can be instantiated for any maximum degree of composition, $\mathcal{D}$. Recall that this means that ciphertexts with up to $\mathcal{D}$ different attributes can be used together in an evaluation. So in summary, a multi-attribute leveled scheme can be instantiated with parameters L and $\mathcal{D}$.

We succeed in constructing such a scheme for the identity-based class of access policies. In other words, we present a multi-identity leveled identity-based FHE (IBFHE). This construction is shown to be selectively-secure in the random oracle model under the hardness of the Learning with Errors (LWE) problem, a standard cryptographic assumption, with a worst-case reduction to hard problems on lattices. This is the first concrete construction of multi-identity leveled FHE. While there is a generic transformation from a selectively-secure scheme to a fully-secure scheme [34], this transformation incurs a degradation in security by a factor of 2^s where $s = |\mathcal{I}|$ is the size of the identity space. Obtaining full security "directly" remains an open problem. Another goal of future work is to eliminate the random oracle, and prove security in the standard model. Our techniques in Chapter 6 are shown to be incompatible for access policies beyond the identity-based functionality, and hence new techniques are needed to construct multi-attribute leveled ABFHE.

We can derive some corollaries from the conclusions of Chapter 6. One of these is that we can invoke our compiler from Chapter 5 with the multi-identity leveled IBFHE from Chapter 6 (this scheme can be naturally set up to evaluate circuits of polylogarithmic depth) to realize a multi-identity scheme that can evaluate all Boolean circuits with N inputs over the domain $\{0,1\}^w$ (recall that w can be arbitrarily large).

Another contribution of Chapter 6 that comes about as an artifact of our main construction is a multikey FHE scheme with security in the standard model from LWE. This is an important contribution for two reasons. Firstly, it is the first multikey FHE scheme from a well-established assumption such as LWE; the only other multikey FHE in the literature [135] relies on a non-standard assumption, namely the Decisional Small

Polynomial Ratio (DSPR) assumption. Secondly our multikey FHE has a decryption circuit in NC^1, as opposed to NC^2 in case of the other multikey FHE scheme from the literature [135]. This contribution lets us derive another corollary. We can employ the compiler from Chapter 5 and instantiate the multikey FHE requirement with our multikey FHE scheme. This means we can strengthen the result by only assuming LWE in the proof of security and only requiring an ABHE capable of evaluating circuits of depth $O(\log(N\lambda))$ where N is the desired upper bound in arity. The downside is that our multikey FHE is considerably less efficient than the one from [135].

In Chapter 7 we derive one of the strongest results of the thesis. We present a theoretical feasibility result for "pure" ABFHE (i.e. supporting evaluation of any polynomial-size circuit), both single-attribute and multi-attribute. Furthermore, our result is obtained constructively - we present a construction of both single-attribute and multi-attribute ABFHE from the notion of indistinguishability obfuscation. An immediate consequence of this is a positive answer to Naccache's open problem of "identity-based fully homomorphic encryption" proposed at his talk at CHES/Crypto 2010 [146]. This is quite a surprising result. This stems from the fact that bootstrapping is the only known way of achieving "pure" FHE but bootstrapping is extremely challenging in the attribute-based setting. This is because bootstrapping with a particular public key relies on having an encryption under that public key of the corresponding secret key. In the public-key setting, this can be achieved by publishing an encryption of the secret key as part of the public key (a circular security assumption is necessary to prove security). This approach is particularly challenging in the attribute-based setting because for every attribute there must be a way to derive an encryption of a secret key under that attribute from the public parameters alone!

While our constructions in Chapter 7 serve as interesting feasibility results, they rely on the computationally intensive machinery of indistinguishability obfuscation, rendering them impractical at the present time.

8.0.1 Future Work

In this section, we amalgamate directions mentioned for future work. Chapter 4 raises the open problem of constructing additively homomorphic IBE for a superpolymomially-sized message space. Furthermore, constructing an attribute-based additively homomorphic scheme would be another goal of future work.

In Chapter 6, a multi-identity leveled IBFHE from the LWE problem is presented. An outstanding goal is to construct a multi-attribute leveled ABFHE from LWE. This is an enticing problem for future work. Note that our multi-identity leveled IBFHE relies on the random oracle model; another goal of future work is to achieve security in the standard model.

In Chapter 7, a feasibility result is presented for multi-attribute ("pure") ABFHE, which relies on the computationally expensive machinery of indistinguishability obfuscation. Finding an equivalent scheme that does not rely on indistinguishability obfuscation is an open problem as is a scheme that only requires indistinguishability obfuscation for infrequently occurring operations, such as the operation of bootstrapping. We presented a compiler to transform a leveled IBFHE satisfying certain properties into a "pure" IBFHE. However finding a leveled IBFHE that satisfies the required properties is a topic for future work. The same holds with respect to ABFHE.

In this thesis, our work has assumed the semi-honest model in which the evaluator is expected to be semi-honest. As such, we have not considered verifiability - this is an open issue for work future work.

Bibliography

[1] CVE-2014-0160. https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-0160. Accessed: 18-08-2014.

[2] Electronic Frontier Foundation: NSA Primary Sources. https://www.eff.org/nsa-spying/nsadocs. Accessed: 19-11-2014.

[3] Half a million widely trusted websites vulnerable to Heartbleed bug. http://news.netcraft.com/archives/2014/04/08/half-a-million-widely-trusted-websites-vulnerable-to-heartbleed-bug.html. Accessed: 18-08-2014.

[4] Hybrid Argument (CRYPTUTOR). http://crypto.cs.uiuc.edu/wiki/index.php/Hybrid_argument. Accessed: 27-11-2014.

[5] Wild at Heart: Were Intelligence Agencies Using Heartbleed in November 2013. https://www.eff.org/deeplinks/2014/04/wild-heart-were-intelligence-agencies-using-heartbleed-november-2013. Accessed: 18-08-2014.

[6] Twitter, 2005. http://www.twitter.com. Last Accessed on 23/09/14.

[7] Agrawal, S., Boneh, D., and Boyen, X. Efficient lattice (H)IBE in the standard model. In *Proc. of Eurocrypt'10* (2010), vol. 6110 of *LNCS*, pp. 553–572.

[8] Agrawal, S., Boneh, D., and Boyen, X. Lattice basis delegation in fixed dimension and shorter-ciphertext hierarchical ibe. In *CRYPTO* (2010), pp. 98–115.

[9] Agrawal, S., Boyen, X., Vaikuntanathan, V., Voulgaris, P., and Wee, H. Functional encryption for threshold functions (or fuzzy ibe) from lattices. In *Public Key Cryptography* (2012), M. Fischlin, J. Buchmann, and M. Manulis, Eds., vol. 7293 of *Lecture Notes in Computer Science*, Springer, pp. 280–297.

[10] Agrawal, S., Gorbunov, S., Vaikuntanathan, V., and Wee, H. Functional encryption: New perspectives and lower bounds. Cryptology ePrint Archive, Report 2012/468, 2012. `http://eprint.iacr.org/`.

[11] Agrell, E., Eriksson, T., Vardy, A., Vardy, E., and Zeger, K. Closest point search in lattices. *IEEE Trans. Inform. Theory 48* (2002), 2201–2214.

[12] Ajtai, M. Generating hard instances of lattice problems (extended abstract). In *STOC '96: Proceedings of the twenty-eighth annual ACM symposium on Theory of computing* (New York, NY, USA, 1996), ACM, pp. 99–108.

[13] Ajtai, M. Generating hard instances of the short basis problem. In *ICAL '99: Proceedings of the 26th International Colloquium on Automata, Languages and Programming* (London, UK, 1999), Springer-Verlag, pp. 1–9.

[14] Akinyele, J. A., Pagano, M. W., Green, M. D., Lehmann, C. U., Peterson, Z. N., and Rubin, A. D. Securing electronic medical records using attribute-based encryption on mobile devices. In *Proceedings of the 1st ACM Workshop on Security and Privacy in Smartphones and Mobile Devices* (New York, NY, USA, 2011), SPSM '11, ACM, pp. 75–86.

[15] Al-Riyami, S. S., and Paterson, K. G. Certificateless public key cryptography.

In *ASIACRYPT* (2003), C.-S. Laih, Ed., vol. 2894 of *Lecture Notes in Computer Science*, Springer, pp. 452–473.

[16] Alderman, J., Cid, C., Crampton, J., and Janson, C. Access control in publicly verifiable outsourced computation. Cryptology ePrint Archive, Report 2014/762, 2014. http://eprint.iacr.org/.

[17] Alwen, J., and Peikert, C. Generating shorter bases for hard random lattices. Cryptology ePrint Archive, Report 2008/521, 2008. http://eprint.iacr.org/2008/521.

[18] Armknecht, F., Katzenbeisser, S., and Peter, A. Group homomorphic encryption: characterizations, impossibility results, and applications. *Designs, Codes and Cryptography* (2012), 1–24.

[19] Armknecht, F., and Sadeghi, A.-R. A new approach for algebraically homomorphic encryption. *IACR Cryptology ePrint Archive 2008* (2008), 422. http://eprint.iacr.org/2008/422.

[20] Ateniese, G., and Gasti, P. Universally anonymous IBE based on the quadratic residuosity assumption. In *Proceedings of the The Cryptographers' Track at the RSA Conference 2009 on Topics in Cryptology* (Berlin, Heidelberg, 2009), CT-RSA '09, Springer-Verlag, pp. 32–47.

[21] Attrapadung, N., and Imai, H. Dual-policy attribute based encryption. In *ACNS* (2009), M. Abdalla, D. Pointcheval, P.-A. Fouque, and D. Vergnaud, Eds., vol. 5536 of *Lecture Notes in Computer Science*, pp. 168–185.

[22] Barreto, P. S. L. M., Kim, H. Y., Lynn, B., and Scott, M. Efficient algorithms for pairing-based cryptosystems. In *CRYPTO* (2002), M. Yung, Ed., vol. 2442 of *Lecture Notes in Computer Science*, Springer, pp. 354–368.

[23] Barrington, D. A. Bounded-width polynomial-size branching programs recognize exactly those languages in nc1. In *Proceedings of the Eighteenth Annual ACM Symposium on Theory of Computing* (New York, NY, USA, 1986), STOC '86, ACM, pp. 1–5.

[24] Bellare, M., Boldyreva, A., Desai, A., and Pointcheval, D. Key-privacy in public-key encryption. Springer-Verlag, pp. 566–582.

[25] Bellare, M., and O'Neill, A. Semantically-secure functional encryption: Possibility results, impossibility results and the quest for a general definition. Cryptology ePrint Archive, Report 2012/515, 2012. `http://eprint.iacr.org/`.

[26] Bellare, M., and Rogaway, P. Optimal asymmetric encryption. Research report RC 19610 (86198), IBM T. J. Watson Research Center, Yorktown Heights, NY, USA, June 1994. Appears in Advances in Cryptology — Eurocrypt 94 Proceedings, 1994.

[27] Ben-David, A., Nisan, N., and Pinkas, B. Fairplaymp: a system for secure multi-party computation. In *Proceedings of the 15th ACM Conference on Computer and Communications Security* (New York, NY, USA, 2008), ACM, pp. 257–266.

[28] Ben-Sasson, E., Chiesa, A., Genkin, D., Tromer, E., and Virza, M. Snarks for c: Verifying program executions succinctly and in zero knowledge. In *CRYPTO (2)* (2013), R. Canetti and J. A. Garay, Eds., vol. 8043 of *Lecture Notes in Computer Science*, Springer, pp. 90–108.

[29] Benaloh, J. D. C. *Verifiable Secret-ballot Elections*. PhD thesis, Yale University, New Haven, CT, USA, 1987. AAI8809191.

[30] Bethencourt, J., Sahai, A., and Waters, B. Ciphertext-policy attribute-based encryption. In *Proceedings of the 2007 IEEE Symposium on Security and*

Privacy (Washington, DC, USA, 2007), SP '07, IEEE Computer Society, pp. 321–334.

[31] Bilge, L., and Dumitras, T. Before we knew it: an empirical study of zero-day attacks in the real world. In *Proceedings of the 2012 ACM conference on Computer and communications security* (2012), ACM, pp. 833–844.

[32] Blum, M., Feldman, P., and Micali, S. Non-interactive zero-knowledge and its applications (extended abstract). In *STOC* (1988), J. Simon, Ed., ACM, pp. 103–112.

[33] Boneh, D., and Boyen, X. Efficient selective-id secure identity-based encryption without random oracles. In *EUROCRYPT* (2004), C. Cachin and J. Camenisch, Eds., vol. 3027 of *Lecture Notes in Computer Science*, Springer, pp. 223–238.

[34] Boneh, D., and Boyen, X. Efficient selective-id secure identity based encryption without random oracles. *IACR Cryptology ePrint Archive 2004* (2004), 172. `http://eprint.iacr.org/2004/172`.

[35] Boneh, D., and Boyen, X. Secure identity based encryption without random oracles. In *CRYPTO* (2004), M. K. Franklin, Ed., vol. 3152 of *Lecture Notes in Computer Science*, Springer, pp. 443–459.

[36] Boneh, D., Crescenzo, G. D., Ostrovsky, R., and Persiano, G. Public key encryption with keyword search. In *EUROCRYPT* (2004), C. Cachin and J. Camenisch, Eds., vol. 3027 of *Lecture Notes in Computer Science*, Springer, pp. 506–522.

[37] Boneh, D., and Franklin, M. Identity-based encryption from the weil pairing. *SIAM J. Comput. 32*, 3 (2003), 586–615.

[38] Boneh, D., and Franklin, M. K. Identity-based encryption from the weil pairing. In *CRYPTO '01: Proceedings of the 21st Annual International Cryptol-*

ogy Conference on Advances in Cryptology (London, UK, 2001), Springer-Verlag, pp. 213–229.

[39] Boneh, D., Gentry, C., and Hamburg, M. Space-efficient identity based encryption without pairings. In *FOCS* (2007), IEEE Computer Society, pp. 647–657.

[40] Boneh, D., Goh, E.-J., and Nissim, K. Evaluating 2-DNF formulas on ciphertexts. In *TCC* (2005), J. Kilian, Ed., vol. 3378 of *Lecture Notes in Computer Science*, Springer, pp. 325–341.

[41] Boneh, D., LaVigne, R., and Sabin, M. Identity-based encryption with eth residuosity and its incompressibility. Project report, Stanford 2013. `http://www.truststc.org/education/reu/13/Papers/SabinM_Paper.pdf`.

[42] Boneh, D., Sahai, A., and Waters, B. Functional encryption: Definitions and challenges. In *Theory of Cryptography*, Y. Ishai, Ed., vol. 6597 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2011, pp. 253–273.

[43] Boneh, D., and Waters, B. Constrained pseudorandom functions and their applications. In *ASIACRYPT (2)* (2013), K. Sako and P. Sarkar, Eds., vol. 8270 of *Lecture Notes in Computer Science*, Springer, pp. 280–300.

[44] Bos, J. W., Lauter, K., Loftus, J., and Naehrig, M. Improved security for a ring-based fully homomorphic encryption scheme. In *IMA Int. Conf.* (2013), M. Stam, Ed., vol. 8308 of *Lecture Notes in Computer Science*, Springer, pp. 45–64.

[45] Boyen, X., and Waters, B. Anonymous hierarchical identity-based encryption (without random oracles). In *CRYPTO* (2006), C. Dwork, Ed., vol. 4117 of *Lecture Notes in Computer Science*, Springer, pp. 290–307.

[46] Boyle, E., Goldwasser, S., and Ivan, I. Functional signatures and pseudorandom functions. In Krawczyk [128], pp. 501–519.

[47] Brakerski, Z. Fully homomorphic encryption without modulus switching from classical gapsvp. In *CRYPTO* (2012), R. Safavi-Naini and R. Canetti, Eds., vol. 7417 of *Lecture Notes in Computer Science*, Springer, pp. 868–886.

[48] Brakerski, Z., Gentry, C., and Vaikuntanathan, V. (leveled) fully homomorphic encryption without bootstrapping. In *ITCS* (2012), S. Goldwasser, Ed., ACM, pp. 309–325.

[49] Brakerski, Z., and Vaikuntanathan, V. Efficient fully homomorphic encryption from (standard) lwe. In *FOCS* (2011), R. Ostrovsky, Ed., IEEE, pp. 97–106.

[50] Brakerski, Z., and Vaikuntanathan, V. Efficient Fully Homomorphic Encryption from (Standard) LWE. Cryptology ePrint Archive, Report 2011/344 Version: 20110627:080002, 2011. `http://eprint.iacr.org/`.

[51] Brakerski, Z., and Vaikuntanathan, V. Fully homomorphic encryption from ring-lwe and security for key dependent messages. In *Advances in Cryptology - CRYPTO 2011 - 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings* (2011), pp. 505–524.

[52] Brakerski, Z., and Vaikuntanathan, V. Fully Homomorphic Encryption from Ring-LWE and Security for Key Dependent Messages, Advances in Cryptology – CRYPTO 2011. vol. 6841 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, Berlin, Heidelberg, 2011, ch. 29, pp. 505–524.

[53] Brickell, E. F., and Yacobi, Y. On privacy homomorphisms. In *EUROCRYPT* (1988), vol. 304 of *Lecture Notes in Computer Science*, Springer, pp. 117–125.

[54] Bringer, J., Chabanne, H., Izabachène, M., Pointcheval, D., Tang, Q., and Zimmer, S. An application of the Goldwasser-Micali cryptosystem to biometric authentication. In *Proceedings of the 12th Australasian conference on*

Information security and privacy (Berlin, Heidelberg, 2007), ACISP'07, Springer-Verlag, pp. 96–106.

[55] Canetti, R., Halevi, S., and Katz, J. A forward-secure public-key encryption scheme. In *EUROCRYPT* (2003), E. Biham, Ed., vol. 2656 of *Lecture Notes in Computer Science*, Springer, pp. 255–271.

[56] Cash, D., Hofheinz, D., Kiltz, E., and Peikert, C. Bonsai trees, or how to delegate a lattice basis. In Gilbert [100], pp. 523–552.

[57] Cheung, L., and Newport, C. C. Provably secure ciphertext policy abe. In *ACM Conference on Computer and Communications Security* (2007), P. Ning, S. D. C. di Vimercati, and P. F. Syverson, Eds., ACM, pp. 456–465.

[58] Chor, B., Kushilevitz, E., Goldreich, O., and Sudan, M. Private information retrieval. *Journal of the Association for Computing Machinery 45*, 6 (Nov. 1998), 965–981.

[59] Chow, R., Golle, P., Jakobsson, M., Shi, E., Staddon, J., Masuoka, R., and Molina, J. Controlling data in the cloud: outsourcing computation without outsourcing control. In *CCSW* (2009), R. Sion and D. Song, Eds., ACM, pp. 85–90.

[60] Clear, M., Hughes, A., and Tewari, H. Homomorphic encryption with access policies: Characterization and new constructions. In *Progress in Cryptology AFRICACRYPT 2013*, A. Youssef, A. Nitaj, and A. Hassanien, Eds., vol. 7918 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2013, pp. 61–87.

[61] Clear, M., and Mc Goldrick, C. Attribute-based fully homomorphic encryption with a bounded number of inputs. *AfricaCrypt 2016* (2016).

[62] Clear, M., and McGoldrick, C. Policy-Based Non-interactive Outsourcing

of Computation using multikey FHE and CP-ABE. *Proceedings of the 10th Internation Conference on Security and Cryptography, SECRYPT 2013* (2013).

[63] Clear, M., and McGoldrick, C. Bootstrappable identity-based fully homomorphic encryption. In *Cryptology and Network Security - 13th International Conference, CANS 2014, Heraklion, Crete, Greece, October 22-24, 2014. Proceedings* (2014), pp. 1–19.

[64] Clear, M., and McGoldrick, C. Multi-identity and multi-key leveled fhe from learning with errors. In *CRYPTO (2)* (2015), R. Gennaro and M. Robshaw, Eds., vol. 9216 of *Lecture Notes in Computer Science*, Springer, pp. 630–656.

[65] Clear, M., Tewari, H., and McGoldrick, C. Anonymous ibe from quadratic residuosity with improved performance. In Pointcheval and Vergnaud [160], pp. 377–397.

[66] Cocks, C. An identity based encryption scheme based on quadratic residues. In *Proceedings of the 8th IMA International Conference on Cryptography and Coding* (London, UK, 2001), Springer-Verlag, pp. 360–363.

[67] Cohen, J. D., and Fischer, M. J. A robust and verifiable cryptographically secure election scheme. In *Proceedings of the 26th Annual Symposium on Foundations of Computer Science* (Washington, DC, USA, 1985), IEEE Computer Society, pp. 372–382.

[68] Cramer, R., Franklin, M. K., Schoenmakers, B., and Yung, M. Multi-authority secret-ballot elections with linear work. In *EUROCRYPT* (1996), U. M. Maurer, Ed., vol. 1070 of *Lecture Notes in Computer Science*, Springer, pp. 72–83.

[69] Cramer, R., Gennaro, R., and Schoenmakers, B. A secure and optimally efficient multi-authority election scheme. In *Advances in cryptology — EUROCRYPT '97: International Conference on the Theory and Application of Crypto-*

graphic Techniques, Konstanz, Germany, May 11–15, 1997: proceedings (Berlin, Germany / Heidelberg, Germany / London, UK / etc., 1997), W. Fumy, Ed., vol. 1233 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 103–118. Sponsored by the International Association for Cryptologic Research (IACR).

[70] Crescenzo, G. D., and Saraswat, V. Public key encryption with searchable keywords based on jacobi symbols. In *INDOCRYPT* (2007), K. Srinathan, C. P. Rangan, and M. Yung, Eds., vol. 4859 of *Lecture Notes in Computer Science*, Springer, pp. 282–296.

[71] Cristofaro, E. D., and Soriente, C. Extended capabilities for a privacy-enhanced participatory sensing infrastructure (PEPSI). *IEEE Transactions on Information Forensics and Security 8*, 12 (2013), 2021–2033.

[72] Cristofaro, E. D., and Soriente, C. Participatory privacy: Enabling privacy in participatory sensing. *IEEE Network 27*, 1 (2013), 32–36.

[73] Cristofaro, E. D., Soriente, C., Tsudik, G., and Williams, A. Hummingbird: Privacy at the time of twitter. In *IEEE Symposium on Security and Privacy* (2012), IEEE Computer Society, pp. 285–299.

[74] Damgård, I., and Jurik, M. A generalisation, a simplification and some applications of paillier's probabilistic public-key system. In *Proceedings of the 4th International Workshop on Practice and Theory in Public Key Cryptography: Public Key Cryptography* (London, UK, UK, 2001), PKC '01, Springer-Verlag, pp. 119–136.

[75] Damgrd, I. Towards practical public key systems secure against chosen ciphertext attacks. In *CRYPTO* (1991), J. Feigenbaum, Ed., vol. 576 of *Lecture Notes in Computer Science*, Springer, pp. 445–456.

[76] De Cristofaro, E., and Soriente, C. Short paper: Pepsi—privacy-enhanced

participatory sensing infrastructure. In *Proceedings of the Fourth ACM Conference on Wireless Network Security* (New York, NY, USA, 2011), WiSec '11, ACM, pp. 23–28.

[77] Ducas, L., Lyubashevsky, V., and Prest, T. Efficient identity-based encryption over ntru lattices. Cryptology ePrint Archive, Report 2014/794, 2014. `http://eprint.iacr.org/`.

[78] Dutta, R., Barua, R., and Sarkar, P. Pairing-based cryptographic protocols : A survey. Cryptology ePrint Archive, Report 2004/064, 2004. `http://eprint.iacr.org/`.

[79] El-Gamal, T. A public-key cryptosystem and a signature scheme based on discrete logarithms. *IEEE TIT IT-31*, 4 (1985), 469–472.

[80] et al., W. H. Fast library for number theory (version 2.4). `http://www.flintlib.org`, 2013.

[81] Fan, J., and Vercauteren, F. Somewhat practical fully homomorphic encryption. *IACR Cryptology ePrint Archive 2012* (2012), 144.

[82] Fischlin, M. A cost-effective pay-per-multiplication comparison method for millionaires. In *CT-RSA* (2001), D. Naccache, Ed., vol. 2020 of *Lecture Notes in Computer Science*, Springer, pp. 457–472.

[83] Fontaine, C., and Galand, F. A survey of homomorphic encryption for non-specialists. *EURASIP J. Inf. Secur. 2007* (January 2007), 15:1–15:15.

[84] Frey, G., Müller, M., and Rück, H.-G. The Tate pairing and the discrete logarithm applied to elliptic curve cryptosystems. *IEEE Transactions on Information Theory 45*, 5 (1999), 1717–1719.

[85] Galbraith, S. D. Elliptic Curve Paillier Schemes. *J. Cryptology 15*, 2 (2002), 129–138.

[86] Galbraith, S. D., Harrison, K., and Soldera, D. In *ANTS*, C. Fieker and D. R. Kohel, Eds., Lecture Notes in Computer Science, Springer, pp. 324–337.

[87] Garg, S., Gentry, C., Halevi, S., Raykova, M., Sahai, A., and Waters, B. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *FOCS* (2013), IEEE Computer Society, pp. 40–49.

[88] Garg, S., Gentry, C., Halevi, S., Sahai, A., and Waters, B. Attribute-based encryption for circuits from multilinear maps. In *CRYPTO (2)* (2013), R. Canetti and J. A. Garay, Eds., vol. 8043 of *Lecture Notes in Computer Science*, Springer, pp. 479–499.

[89] Gennaro, R., Gentry, C., and Parno, B. Non-interactive verifiable computing: outsourcing computation to untrusted workers. In *Proceedings of the 30th annual conference on Advances in Cryptology* (Berlin, Heidelberg, 2010), CRYPTO'10, Springer-Verlag, pp. 465–482.

[90] Gentry, C. Certificate-based encryption and the certificate revocation problem. In *EUROCRYPT* (2003), E. Biham, Ed., vol. 2656 of *Lecture Notes in Computer Science*, Springer, pp. 272–293.

[91] Gentry, C. Practical identity-based encryption without random oracles. In *EUROCRYPT* (2006), S. Vaudenay, Ed., vol. 4004 of *Lecture Notes in Computer Science*, Springer, pp. 445–464.

[92] Gentry, C. *A fully homomorphic encryption scheme*. PhD thesis, Stanford University, 2009. `crypto.stanford.edu/craig`.

[93] Gentry, C. Fully homomorphic encryption using ideal lattices. *Proceedings of the 41st annual ACM Symposium on Theory of Computing STOC 09*, September (2009), 169.

[94] Gentry, C., Halevi, S., and Vaikuntanathan, V. A Simple BGN-Type Cryptosystem from LWE. In Gilbert [100], pp. 506–522.

[95] Gentry, C., Halevi, S., and Vaikuntanathan, V. i-hop homomorphic encryption and rerandomizable yao circuits. In *CRYPTO* (2010), T. Rabin, Ed., vol. 6223 of *Lecture Notes in Computer Science*, Springer, pp. 155–172.

[96] Gentry, C., Lewko, A. B., Sahai, A., and Waters, B. Indistinguishability obfuscation from the multilinear subgroup elimination assumption. *IACR Cryptology ePrint Archive 2014* (2014), 309.

[97] Gentry, C., Peikert, C., and Vaikuntanathan, V. Trapdoors for hard lattices and new cryptographic constructions. In *STOC '08: Proceedings of the 40th annual ACM symposium on Theory of computing* (New York, NY, USA, 2008), ACM, pp. 197–206.

[98] Gentry, C., Sahai, A., and Waters, B. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In *CRYPTO (2013)* (2013), R. Canetti and J. A. Garay, Eds., vol. 8042 of *Lecture Notes in Computer Science*, Springer, pp. 75–92.

[99] Gentry, C., and Silverberg, A. Hierarchical id-based cryptography. In *ASIACRYPT* (2002), Y. Zheng, Ed., vol. 2501 of *Lecture Notes in Computer Science*, Springer, pp. 548–566.

[100] Gilbert, H., Ed. *Advances in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, French Riviera, May 30 - June 3, 2010. Proceedings* (2010), vol. 6110 of *Lecture Notes in Computer Science*, Springer.

[101] Gjøsteen, K. Homomorphic cryptosystems based on subgroup membership prob-

lems. In *Proceedings of the 1st international conference on Progress in Cryptology in Malaysia* (Berlin, Heidelberg, 2005), Mycrypt'05, Springer-Verlag, pp. 314–327.

[102] Gjsteen, K. Symmetric subgroup membership problems. In *Public Key Cryptography* (2005), S. Vaudenay, Ed., vol. 3386 of *Lecture Notes in Computer Science*, Springer, pp. 104–119.

[103] Goldwasser, S. Lecture: Introduction to homomorophic encryption, February 2011. `http://www.cs.bu.edu/~reyzin/teaching/s11cs937/notes-shafi-1.pdf`. Last Checked on March 31st 2013.

[104] Goldwasser, S., Goyal, V., Jain, A., and Sahai, A. Multi-input functional encryption. Cryptology ePrint Archive, Report 2013/727, 2013. `http://eprint.iacr.org/`.

[105] Goldwasser, S., and Kharchenko, D. Proof of plaintext knowledge for the ajtai-dwork cryptosystem. In *TCC* (2005), J. Kilian, Ed., vol. 3378 of *Lecture Notes in Computer Science*, Springer, pp. 529–555.

[106] Goldwasser, S., and Micali, S. Probabilistic encryption & how to play mental poker keeping secret all partial information. In *Proceedings of the fourteenth annual ACM symposium on Theory of computing* (New York, NY, USA, 1982), STOC '82, ACM, pp. 365–377.

[107] Goldwasser, S., and Micali, S. Probabilistic encryption. *Journal of Computer and System Sciences 28*, 2 (1984), 270–299. See also preliminary version in 14th STOC, 1982.

[108] Gorbunov, S., Vaikuntanathan, V., and Wee, H. Functional encryption with bounded collusions via multi-party computation. In *Advances in Cryptology . CRYPTO 2012*, R. Safavi-Naini and R. Canetti, Eds., vol. 7417 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2012, pp. 162–179.

[109] Gorbunov, S., Vaikuntanathan, V., and Wee, H. Attribute-based encryption for circuits. In *Proceedings of the 45th annual ACM symposium on Symposium on theory of computing* (New York, NY, USA, 2013), STOC '13, ACM, pp. 545–554.

[110] Gordon, S. D., Katz, J., Liu, F.-H., Shi, E., and Zhou, H.-S. Multi-input functional encryption. Cryptology ePrint Archive, Report 2013/774, 2013. `http://eprint.iacr.org/`.

[111] Gordon, S. D., Katz, J., Liu, F.-H., Shi, E., and Zhou, H.-S. Multi-input functional encryption. Cryptology ePrint Archive, Report 2013/774, 2013. `http://eprint.iacr.org/`.

[112] Goyal, V., Jain, A., Pandey, O., and Sahai, A. Bounded ciphertext policy attribute based encryption. In *ICALP (2)* (2008), L. Aceto, I. Damgrd, L. A. Goldberg, M. M. Halldrsson, A. Inglfsdttir, and I. Walukiewicz, Eds., vol. 5126 of *Lecture Notes in Computer Science*, Springer, pp. 579–591.

[113] Goyal, V., Pandey, O., Sahai, A., and Waters, B. Attribute-based encryption for fine-grained access control of encrypted data. In *Proceedings of the 13th ACM conference on Computer and communications security* (New York, NY, USA, 2006), CCS '06, ACM, pp. 89–98.

[114] Günther, F., Manulis, M., and Peter, A. Privacy-enhanced participatory sensing with collusion resistance and data aggregation. In *Cryptology and Network Security - 13th International Conference, CANS 2014, Heraklion, Crete, Greece, October 22-24, 2014. Proceedings* (2014), pp. 321–336.

[115] Hayashi, R., and Tanaka, K. Universally anonymizable public-key encryption. In *ASIACRYPT* (2005), B. K. Roy, Ed., vol. 3788 of *Lecture Notes in Computer Science*, Springer, pp. 293–312.

[116] Hess, F., Smart, N. P., and Vercauteren, F. The eta pairing revisited. *IEEE Transactions on Information Theory 52*, 10 (2006), 4595–4602.

[117] Hoffstein, J., Pipher, J., and Silverman, J. H. NTRU: a ring-based public key cryptosystem. *Lecture Notes in Computer Science 1423* (1998), 267–288.

[118] Horwitz, J., and Lynn, B. Toward hierarchical identity-based encryption. In *EUROCRYPT '02: Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques* (London, UK, 2002), Springer-Verlag, pp. 466–481.

[119] Ibraimi, L., Asim, M., and Petkovic, M. Secure management of personal health records by applying attribute-based encryption, July 2009.

[120] Ishai, Y., and Paskin, A. Evaluating branching programs on encrypted data. In *TCC* (2007), S. P. Vadhan, Ed., vol. 4392 of *Lecture Notes in Computer Science*, Springer, pp. 575–594.

[121] Jhanwar, M., and Barua, R. A variant of boneh-gentry-hamburgs pairing-free identity based encryption scheme. In *Information Security and Cryptology*, M. Yung, P. Liu, and D. Lin, Eds., vol. 5487 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2009, pp. 314–331.

[122] Joux, A. A one round protocol for tripartite diffie-hellman. In *ANTS* (2000), W. Bosma, Ed., vol. 1838 of *Lecture Notes in Computer Science*, Springer, pp. 385–394.

[123] Kandias, M., Virvilis, N., and Gritzalis, D. The insider threat in cloud computing. In *CRITIS* (2011), S. Bologna, B. M. Hmmerli, D. Gritzalis, and S. D. Wolthusen, Eds., vol. 6983 of *Lecture Notes in Computer Science*, Springer, pp. 93–103.

[124] Katz, J., Sahai, A., and Waters, B. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In *Proceedings of the theory and applications of cryptographic techniques 27th annual international conference on Advances in cryptology* (Berlin, Heidelberg, 2008), EUROCRYPT'08, Springer-Verlag, pp. 146–162.

[125] Kawachi, A., Tanaka, K., and Xagawa, K. Multi-bit cryptosystems based on lattice problems. In *Public Key Cryptography* (2007), T. Okamoto and X. Wang, Eds., vol. 4450 of *Lecture Notes in Computer Science*, Springer, pp. 315–329.

[126] Kiayias, A., Papadopoulos, S., Triandopoulos, N., and Zacharias, T. Delegatable pseudorandom functions and applications. In *ACM Conference on Computer and Communications Security* (2013), A.-R. Sadeghi, V. D. Gligor, and M. Yung, Eds., ACM, pp. 669–684.

[127] Kortchinsky, K. Cloudburst - a vmware guest to host escape story. BlackHat USA 2009, Las Vegas, USA.

[128] Krawczyk, H., Ed. *Public-Key Cryptography - PKC 2014 - 17th International Conference on Practice and Theory in Public-Key Cryptography, Buenos Aires, Argentina, March 26-28, 2014. Proceedings* (2014), vol. 8383 of *Lecture Notes in Computer Science*, Springer.

[129] Kushilevitz, E., and Ostrovsky, R. Replication is not needed: Single database, computationally-private information retrieval. In *Proceedings of the 38th Annual Symposium on Foundations of Computer Science* (Washington, DC, USA, 1997), FOCS '97, IEEE Computer Society, pp. 364–.

[130] Lee, B., Boyd, C., Dawson, E., Kim, K., Yang, J., and Yoo, S. Secure key issuing in id-based cryptography. In *ACSW Frontiers* (2004), J. M. Hogan, P. Montague, M. K. Purvis, and C. Steketee, Eds., vol. 32 of *CRPIT*, Australian Computer Society, pp. 69–74.

[131] Lepoint, T., and Naehrig, M. A comparison of the homomorphic encryption schemes FV and YASHE. In Pointcheval and Vergnaud [160], pp. 318–335.

[132] Lewko, A. B., Okamoto, T., Sahai, A., Takashima, K., and Waters, B. Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. In *EUROCRYPT* (2010), H. Gilbert, Ed., vol. 6110 of *Lecture Notes in Computer Science*, Springer, pp. 62–91.

[133] Lewko, A. B., and Waters, B. New proof methods for attribute-based encryption: Achieving full security through selective techniques. In *CRYPTO* (2012), R. Safavi-Naini and R. Canetti, Eds., vol. 7417 of *Lecture Notes in Computer Science*, Springer, pp. 180–198.

[134] Liu, A., and Ning, P. Tinyecc: A configurable library for elliptic curve cryptography in wireless sensor networks. In *IPSN '08: Proceedings of the 7th international conference on Information processing in sensor networks* (Washington, DC, USA, 2008), IEEE Computer Society, pp. 245–256.

[135] López-Alt, A., Tromer, E., and Vaikuntanathan, V. On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption. In *Proceedings of the 44th symposium on Theory of Computing* (New York, NY, USA, 2012), STOC '12, ACM, pp. 1219–1234.

[136] Lyubashevsky, V., Peikert, C., and Regev, O. On ideal lattices and learning with errors over rings. In Gilbert [100], pp. 1–23.

[137] Melchor, C. A., Castagnos, G., and Gaborit, P. Lattice-based homomorphic encryption of vector spaces. In *ISIT* (2008), F. R. Kschischang and E.-H. Yang, Eds., IEEE, pp. 1858–1862.

[138] Melchor, C. A., Gaborit, P., and Herranz, J. Additive homomorphic

encryption with t-operand multiplications, 2008. carlos.aguilar@unilim.fr 14140 received 5 Sep 2008, last revised 18 Sep 2008.

[139] Menezes, A., Okamoto, T., and Vanstone, S. A. Reducing elliptic curve logarithms to logarithms in a finite field. *IEEE Transactions on Information Theory 39*, 5 (1993), 1639–1646.

[140] Micciancio, D. A first glimpse of cryptography's holy grail. *Commun. ACM 53*, 3 (Mar. 2010), 96–96.

[141] Micciancio, D., and Mol, P. Pseudorandom knapsacks and the sample complexity of LWE search-to-decision reductions. In *Advances in Cryptology - CRYPTO 2011 - 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings* (2011), pp. 465–484.

[142] Micciancio, D., and Peikert, C. Trapdoors for lattices: Simpler, tighter, faster, smaller. In *EUROCRYPT* (2012), pp. 700–718.

[143] Mohan, A., Bauer, D., Blough, D. M., Ahamad, M., Bamba, B., Krishnan, R., Liu, L., Mashima, D., and Palanisamy, B. A patient-centric, attribute-based, source-verifiable framework for health record sharing.

[144] Mont, M. C., Harrison, K., and Sadler, M. The hp time vault service: exploiting ibe for timed release of confidential information. In *WWW* (2003), G. Hencsey, B. White, Y.-F. R. Chen, L. Kovcs, and S. Lawrence, Eds., ACM, pp. 160–169.

[145] Mukherjee, P., and Wichs, D. Two round mutliparty computation via multi-key fhe. Cryptology ePrint Archive, Report 2015/345, 2015. `http://eprint.iacr.org/`.

[146] Naccache, D. Is theoretical cryptography any good in practice?, August 2010. Talk given at CHES 2010 and Crypto 2010.

[147] Naccache, D., and Stern, J. A new public key cryptosystem based on higher residues. In *ACM Conference on Computer and Communications Security* (1998), L. Gong and M. K. Reiter, Eds., ACM, pp. 59–66.

[148] Naor, M., and Pinkas, B. Oblivious polynomial evaluation. *SIAM J. Comput. 35*, 5 (2006), 1254–1281.

[149] Okamoto, T., and Uchiyama, S. A new public-key cryptosystem as secure as factoring. *Lecture Notes in Computer Science 1403* (1998), 308–318.

[150] Oliveira, L., Scott, M., Lopez, J., and Dahab, R. Tinypbc: Pairings for authenticated identity-based non-interactive key distribution in sensor networks. In *Networked Sensing Systems, 2008. INSS 2008. 5th International Conference on* (June 2008), pp. 173–180.

[151] Oliveira, L. B., Aranha, D. F., Morais, E., Daguano, F., L?pez, J., and Dahab, R. Tinytate: Computing the tate pairing in resource-constrained sensor nodes. *Network Computing and Applications, IEEE International Symposium on 0* (2007), 318–323.

[152] O'Neill, A. Definitional issues in functional encryption. *IACR Cryptology ePrint Archive 2010* (2010), 556.

[153] Ostrovsky, R., Sahai, A., and Waters, B. Attribute-based encryption with non-monotonic access structures. In *ACM Conference on Computer and Communications Security* (2007), P. Ning, S. D. C. di Vimercati, and P. F. Syverson, Eds., ACM, pp. 195–203.

[154] Paillier, P. Public-key cryptosystems based on composite degree residuosity classes. In *EUROCRYPT* (1999), J. Stern, Ed., vol. 1592 of *Lecture Notes in Computer Science*, Springer, pp. 223–238.

[155] Pass, R., Seth, K., and Telang, S. Indistinguishability obfuscation from semantically-secure multilinear encodings. In *CRYPTO (1)* (2014), J. A. Garay and R. Gennaro, Eds., vol. 8616 of *Lecture Notes in Computer Science*, Springer, pp. 500–517.

[156] Peikert, C. Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In *STOC* (2009), M. Mitzenmacher, Ed., ACM, pp. 333–342.

[157] Peikert, C., and Waters, B. Lossy trapdoor functions and their applications. *SIAM J. Comput. 40*, 6 (2011), 1803–1844.

[158] Peng, K., Boyd, C., and Dawson, E. A multiplicative homomorphic sealed-bid auction based on goldwasser-micali encryption. In *ISC* (2005), J. Zhou, J. Lopez, R. H. Deng, and F. Bao, Eds., vol. 3650 of *Lecture Notes in Computer Science*, Springer, pp. 374–388.

[159] Pirretti, M., Traynor, P., McDaniel, P., and Waters, B. Secure attribute-based systems. *Journal of Computer Security 18*, 5 (2010), 799–837.

[160] Pointcheval, D., and Vergnaud, D., Eds. *Progress in Cryptology - AFRICACRYPT 2014 - 7th International Conference on Cryptology in Africa, Marrakesh, Morocco, May 28-30, 2014. Proceedings* (2014), vol. 8469 of *Lecture Notes in Computer Science*, Springer.

[161] Regev, O. On lattices, learning with errors, random linear codes, and cryptography. In *STOC '05: Proceedings of the thirty-seventh annual ACM symposium on Theory of computing* (New York, NY, USA, 2005), ACM, pp. 84–93.

[162] Rivest, R., Adleman, L., and Dertouzos, M. On data banks and privacy homomorphisms. Academic Press, pp. 169–177.

[163] Rivest, R. L., Shamir, A., and Adleman, L. A method for obtaining digital signatures and public key cryptosystems. *Communications of the Association for Computing Machinery 21*, 2 (Feb. 1978), 120–126.

[164] Sahai, A., and Waters, B. Fuzzy identity-based encryption. In *Proceedings of the 24th Annual International Conference on Theory and Applications of Cryptographic Techniques* (Berlin, Heidelberg, 2005), EUROCRYPT'05, Springer-Verlag, pp. 457–473.

[165] Sahai, A., and Waters, B. How to use indistinguishability obfuscation: Deniable encryption, and more. *IACR Cryptology ePrint Archive 2013* (2013), 454.

[166] Sander, T., Young, A. L., and Yung, M. Non-interactive cryptocomputing for nc^1. In *FOCS* (1999), IEEE Computer Society, pp. 554–567.

[167] Shamir, A. Identity-based cryptosystems and signature schemes. *Lecture Notes in Computer Science 196* (1985), 47–53.

[168] Shi, E. *Evaluating predicates over encrypted data*. PhD thesis, Pittsburgh, PA, USA, 2008. AAI3341267.

[169] Shoup, V. Oaep reconsidered. In *CRYPTO* (2001), J. Kilian, Ed., vol. 2139 of *Lecture Notes in Computer Science*, Springer, pp. 239–259.

[170] Smart, N., and Vercauteren, F. Fully homomorphic encryption with relatively small key and ciphertext sizes. In *Public Key Cryptography – PKC 2010*, P. Nguyen and D. Pointcheval, Eds., vol. 6056 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2010, pp. 420–443.

[171] Stehl, D., and Steinfeld, R. Faster fully homomorphic encryption. In *ASIACRYPT* (2010), M. Abe, Ed., vol. 6477 of *Lecture Notes in Computer Science*, Springer, pp. 377–394.

[172] Szczechowiak, P., Kargl, A., Scott, M., and Collier, M. On the application of pairing based cryptography to wireless sensor networks. In *WiSec '09: Proceedings of the second ACM conference on Wireless network security* (New York, NY, USA, 2009), ACM, pp. 1–12.

[173] Vaikuntanathan, V. Computing blindfolded: New developments in fully homomorphic encryption. In *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA, October 22-25, 2011* (2011), R. Ostrovsky, Ed., IEEE Computer Society, pp. 5–16.

[174] van Dijk, M., Gentry, C., Halevi, S., and Vaikuntanathan, V. Fully homomorphic encryption over the integers. In *Advances in Cryptology – EUROCRYPT 2010*, H. Gilbert, Ed., vol. 6110 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2010, pp. 24–43.

[175] Waters, B. Efficient identity-based encryption without random oracles. In *EUROCRYPT* (2005), R. Cramer, Ed., vol. 3494 of *Lecture Notes in Computer Science*, Springer, pp. 114–127.

[176] Waters, B. Dual system encryption: Realizing fully secure ibe and hibe under simple assumptions. In *CRYPTO* (2009), S. Halevi, Ed., vol. 5677 of *Lecture Notes in Computer Science*, Springer, pp. 619–636.

[177] Waters, B. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In *Public Key Cryptography* (2011), D. Catalano, N. Fazio, R. Gennaro, and A. Nicolosi, Eds., vol. 6571 of *Lecture Notes in Computer Science*, Springer, pp. 53–70.

[178] Yao, A. Protocols for secure computation. In *23rd annual Symposium on Foundations of Computer Science, November 3–5, 1982, Chicago, IL* (1109 Spring Street, Suite 300, Silver Spring, MD 20910, USA, 1982), IEEE, Ed., IEEE Computer So-

ciety Press, pp. 160–164. IEEE catalog no. 82CH1806-9. IEEE Computer Society order no. 440.

Appendix A

Glossary

G1: Group: A group is a set S together with a binary operation $* : S \times S \to S$ satisfying the following properties:

- The operation $*$ is associative.
- There is a unique element $1 \in S$ (note we use multiplicative notation), called the *identity element* such that for all $x \in S$, we have $x * 1 = 1 * x = x$.
- For every $x \in S$, there is an element $x^{-1} \in S$ with $x * x^{-1} = x^{-1} * x = 1$.

We denote a group with the pair $(S, *)$. A group is said to be Abelian if it is commutative i.e. $x * y = y * x$ for every $x, y \in S$.

G2: Ring: A ring is a set R together with two binary operations $+ : R \times R \to R$ and $\cdot : R \times R \to R$ that satisfies the following properties. Note that $+$ is called *addition* and $\cdot$ is called *multiplication*.

- $(R, +)$ is an Abelian group.
- There is a multiplicative identity element i.e. an element $1 \in R$ with $x \cdot 1 = 1 \cdot x = x$ for every $x \in R$.
- The operation $\cdot$ is associative.

- Multiplication distributes over addition; that is, for every $x, y, z \in R$ we have
  - $x \cdot (y + z) = (x \cdot y) + (x \cdot z)$
  - $(y + z) \cdot x = (y \cdot x) + (z \cdot x)$.

Let N be a positive integer. An example of a ring is the set of integers modulo N, which is denoted by $\mathbb{Z}_N$.

G3: Homomorphism: A homomorphism is a structure-preserving map between two mathematical objects. A homomorphism between two algebraic structures (e.g: groups) $(A, *)$ and $(B, \circ)$ is a map $f : A \to B$ that satisfies the following property:

$$f(x * y) = f(x) \circ f(y) \text{ for all } x, y \in A.$$

A ring homomorphism between two rings R and S is a map f that satisfies the above with respect to *both* operations of the ring.

G4: Negligible Probability: A quantity is said to be negligible with respect to some parameter κ, written $\mathsf{negl}(\kappa)$, if it is asymptotically bounded from above by the reciprocal of all polynomials in κ.

G5: Security Parameter: The parameter that represents the difficulty of breaking a scheme. This parameter also measures the input size - so other parameters can be expressed in terms of it.

G6: Semantic Security (IND-CPA**:** An encryption scheme is said to be semantically secure if an adversary that runs in polynomial time who is given a ciphertext c cannot learn anything about what c encrypts, except with negligible probability. This notion has been shown to be equivalent to the following game played between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$. Note that $\mathcal{A}$ is a probabilistic polynomial time (PPT) algorithm.

- $\mathcal{C}$ gives $\mathcal{A}$ the parameters of the scheme. If the scheme is public-key, the adversary is given the public key.

- The adversary chooses two equal-length messages $\mu_0, \mu_1 \in \mathcal{M}$ from the scheme's message space $\mathcal{M}$.
- The challenger chooses a bit b at random. The challenger gives an encryption of μ_b to the adversary.
- The adversary outputs a guess $b' \in \{0, 1\}$.

The adversary wins the game if $b = b'$. The advantage of $\mathfrak{A}$ is defined as the probability of it winning the above game minus $1/2$. If no polynomial-time adversary has non-negligible advantage, the scheme is said to be IND-CPA secure (indistinguishability under a chosen plaintext attack). As aforementioned, IND-CPA is equivalent to that of *semantic security*, and as such, the two terms are used interchangeably.

G7: Random Oracle Model: A random oracle $\mathcal{O}$ is a theoretical black box that can be *queried* with an element from its input domain, and it responds with an element from its output domain. For every unique query x, it answers with a randomly chosen element y of its output domain, and it consistently answers with the same element y on all subsequent queries for x. In the random oracle model, a function (such as a hash function) that is used in a cryptographic scheme is replaced by a random oracle in the proof of security.

G8: Reductions: Let $\mathfrak{p}$ be a problem. Suppose there is an algorithm $\mathcal{A}$ that breaks the security of a scheme $\mathcal{E}$ e.g: $\mathcal{A}$ attacks the IND-CPA security of $\mathcal{E}$. Suppose $\mathcal{A}$ can be used to solve $\mathfrak{p}$ with extra work w. If the extra work w can be done in polynomial time, then an algorithm such as $\mathcal{A}$ that breaks $\mathcal{E}$ can be used to efficiently (as in polynomial time) solve $\mathfrak{p}$. Hence $\mathfrak{p}$ can be polynomially reduced to breaking $\mathcal{E}$, which implies breaking $\mathcal{E}$ is at least as hard as solving $\mathfrak{p}$.

G9: Hybrid Argument: The following description is based on [4]. A hybrid argument is a proof strategy to show that two distributions are computationally indistinguishable. A sequence of polynomially many (in the security parameter)

distributions $\mathcal{D}_1, \ldots, \mathcal{D}_t$ (referred to as hybrids) are defined. The distributions $\mathcal{D}_1$ and $\mathcal{D}_t$ are the ones to be shown computationally indistinguishable. This is done by proving each pair of adjacent distributions to be computationally indistinguishable. This is achieved usually by changing one aspect of the distribution such as replacing a cryptographic primitive with its idealization. Because computational indistinguishability is transitive across a polynomial number of distributions, the distributions $\mathcal{D}_1$ and $\mathcal{D}_t$ are therefore computationally indistinguishable.

Appendix B

Properties of Attribute Based Group Homomorphic Encryption

in this section we will establish some properties about ABGHE schemes. To help us in this task, we first define a particular ABGHE scheme which we make reference to throughout the section. Let $\mathcal{E} = (G, K, E, D)$ be a ABGHE scheme satisfying Definition 4.1.1 with message space $(\mathcal{M}, \cdot)$, attribute space $\mathbb{A}$, access policies $\mathbb{F}$, ciphertext space $\widehat{\mathcal{C}}$ and binary operation $* : \widehat{\mathcal{C}} \times \widehat{\mathcal{C}} \to \widehat{\mathcal{C}}$. Fix any $(\mathsf{PP}, \mathsf{MSK}) \leftarrow G(1^\lambda)$. Note that the identity element of $(\mathcal{M}, \cdot)$ is written as $1 \in \mathcal{M}$. We assume that $\mathbb{F}$ is free of any *degenerate* policies; that is, policies f with $f(a) = 0 \; \forall a \in \mathbb{A}$.

B.1 Partition of Access Policies

As discussed in Chapter 4, a fundamental property of an ABGHE scheme is that its class of access policies $\mathbb{F}$ can be partitioned into equivalence classes via a natural relation $\sim$. The relation is defined for any $f, g \in \mathbb{F}$ as

$$f \sim g \quad \text{iff } \mathsf{supp}(f) \cap \mathsf{supp}(g) \neq \emptyset.$$

Now $\sim$ is clearly reflexive and symmetric, but it is not necessarily transitive in the case of an arbitrary ABHE scheme. However if the scheme is group homomorphic, i.e. it satisfies Definition 4.1.1, then $\sim$ is also transitive, and hence an equivalence relation. We now give the proof of Lemma 4.2.1 first stated in Chapter 4.

Lemma 4.2.1 (transitivity of $\sim$)**.** *Let* $f_1, f_2, g \in \mathbb{F}$ *such that* $\mathsf{supp}(f_1) \cap \mathsf{supp}(g) \neq \emptyset$ *and* $\mathsf{supp}(f_2) \cap \mathsf{supp}(g) \neq \emptyset$*. Then* $\mathsf{supp}(f_1) \cap \mathsf{supp}(f_2) \neq \emptyset$*.*

Proof. By GH.1 in Definition 4.1.1 we have that $\mathcal{C}_{f_1} \subset \widehat{\mathcal{C}}$, $\mathcal{C}_{f_2} \subset \widehat{\mathcal{C}}$ and $\mathcal{C}_g \subseteq \widehat{\mathcal{C}}$ are non-trivial groups under the operation $*$. Let e be the identity element of $\mathcal{C}_g$. For any $x \in \mathcal{C}_{f_1} \cap \mathcal{C}_g$ we have $x * e = x$. Therefore $e \in \mathcal{C}_{f_1}$. Analogously, we have $e \in \mathcal{C}_{f_2}$. It follows from GH.2 in Definition 4.1.1 that $D_{\mathsf{sk}_{f_1}}(e) = D_{\mathsf{sk}_{f_2}}(e) = 1 \in \mathcal{M}$ for any $\mathsf{sk}_{f_1} \leftarrow K(\mathsf{MSK}, f_1)$ and $\mathsf{sk}_{f_2} \leftarrow K(\mathsf{MSK}, f_2)$. It follows that e is associated with an attribute that satisfies both f_1 and f_2. □

B.2 Generic Transformation for Multiple Attributes

As mentioned in Chapter 4, an ABGHE scheme natively follows the atomic model of decryption i.e. $\mathcal{K} = 1$. It is possible to construct a related scheme $\mathcal{E}' = (G', K', E', D')$ that is group homomorphic for $(\mathcal{M}, \cdot)$, but with $\mathcal{D} = \mathcal{K} = |\mathbb{A}|$. Technically $\mathcal{E}'$ is not an ABGHE since it doesn't satisfy Definition 4.1.1. Instead $\mathcal{E}'$ is a group homomorphic scheme that follows the collaborative model of decryption. Its salient feature is that ciphertexts grow linearly with the degree of composition.

We assume without loss of generality that there is a strict total order $\prec$ defined on $\mathbb{A}$. We also assume that $\mathcal{E}$ is not attribute-hiding; more precisely, the attribute associated with a ciphertext is readily obtained from the ciphertext (if this is not naturally the case, it can be done by appending the attribute to the ciphertext). Therefore, we define the function $\mathsf{attr} : \hat{\mathcal{C}} \to \mathbb{A}$ that gives the attribute associated with a ciphertext.

Now the ciphertext space $\hat{\mathcal{C}}'$ of $\mathcal{E}'$ is defined as $\hat{\mathcal{C}}' \triangleq \{(c_1, \ldots, c_t) \in \hat{\mathcal{C}}^* \mid c_1, \ldots, c_t \in \hat{\mathcal{C}}, \mathsf{attr}(c_1) \prec \cdots \prec \mathsf{attr}(c_t), |t| \leq |\mathbb{A}|\}$. The encryption algorithm E' is set to E, and it

outputs ciphertexts in $\mathcal{C} \subseteq \hat{\mathcal{C}} \subseteq \hat{\mathcal{C}}'$. We define a binary operation $\dagger : \hat{\mathcal{C}}' \times \hat{\mathcal{C}}' \to \hat{\mathcal{C}}'$ as follows. Given two ciphertexts $\mathsf{CT}_1 := (c_1^{(1)}, \ldots, c_{t_1}^{(1)}) \in \hat{\mathcal{C}}'$ and $\mathsf{CT}_2 := (c_1^{(2)}, \ldots, c_{t_2}^{(2)}) \in \hat{\mathcal{C}}'$, we compute $\mathsf{CT}_3 := \mathsf{CT}_1 \dagger \mathsf{CT}_2$ in the following way: (1). apply merge sort to the list of elements in CT_1 and CT_2 with respect to the total order $\prec$; (2). for every pair of adjacent elements $c^{(1)}, c^{(2)}$ with matching attributes, compute $c \leftarrow c^{(1)} * c^{(2)}$ and replace the occurrence of $c^{(1)}, c^{(2)}$ in the list with c. The resulting list of elements in CT_3 has length equal to $t_1 + t_2 - t'$ where t' is the number of matching attributes. So the size of an evaluated ciphertext grows linearly with the degree of composition.

However we only want a decryptor to learn a single value $\mu \in \mathcal{M}$. In other words, she should not be able learn about the *components* of this value, where each *component* is encrypted under a distinct attribute. To resolve this, a re-randomization step is performed after computing $\mathsf{CT}_3 = \mathsf{CT}_1 \dagger \mathsf{CT}_2$. Let $\mathsf{CT}_3 = (c_1, \ldots, c_{\mathcal{d}})$. The evaluator generates uniformly random $r_1, \ldots, r_{\mathcal{d}-1} \xleftarrow{\$} \mathcal{M}$, and sets $r_{\mathcal{d}} \leftarrow (r_1 \cdots r_{\mathcal{d}-1})^{-1}$. Therefore we have $r_1 \cdots r_{\mathcal{d}} = 1 \in \mathcal{M}$. Then the evaluator sets $c_i \leftarrow c_i * E_{\mathsf{PP}}(\mathsf{attr}(c_i), r_i)$ for every $i \in [\mathcal{d}]$ and outputs $\mathsf{CT}' := (c_1, \ldots, c_{\mathcal{d}})$.

A decryptor uses secret keys for her policies $f_1, \ldots, f_{\mathcal{k}}$ to decrypt CT' as follows. Firstly, she uses one of her policies to recover $\mu_i \in \mathcal{M}$ from each component c_i for $i \in [\mathcal{d}]$. Then she outputs $\mu_1 \cdots \mu_{\mathcal{d}} \in \mathcal{M}$.

Appendix C

Time-Performant Anonymous IBE from Quadratic Residuosity

C.0.1 Security Definition for Anonymous IBE (ANON-IND-ID-CPA)

An IBE scheme is said to be anonymous if any PPT adversary has only a negligible advantage in the following game. This is referred to as ANON-IND-ID-CPA security. At the beginning of the game, the adversary $\mathcal{A}$ is handed the public parameters. It then proceeds to make queries for secret keys corresponding to identities $\mathsf{id}_1, \ldots, \mathsf{id}_{q_1}$ for some integer q_1 that is polynomial in the security parameter. Then it sends to the challenger two identities id_0^* and id_1^* such that $\mathsf{id}_0^* \neq \mathsf{id}_1^* \neq \mathsf{id}_i$ for $1 \leq i \leq q_1$. It also sends two messages m_0 and m_1. The challenger samples a bit b uniformly, and sends the encryption of m_b under id_b^* to $\mathcal{A}$. In the final phase, $\mathcal{A}$ is allowed to query secret keys for further identities $\mathsf{id}_{q_1+1}, \ldots, \mathsf{id}_{q_1+q_2}$ where q_2 is polynomial in the security parameter, and $\mathsf{id}_0^* \neq \mathsf{id}_1^* \neq \mathsf{id}_{q_1+i}$ for $1 \leq i \leq q_2$. Finally, $\mathcal{A}$ outputs a guess b' and is said to win if $b' = b$.

C.0.2 Overview of our construction

In order to understood our construction, the reader must be familiar with the XOR-homomorphic IBE presented in Chapter 4.

Let a be an integer in $\mathbb{J}(N)$. Then let R_a be the quotient ring $R/(x^2 - a)$. Recall the generalization of Galbraith's test to the ring R as follows.

Definition C.0.1 (Galbraith's Test over R)**.** *Define Galbraith's Test for the ring R as the function $\mathsf{GT} : \mathbb{Z}_N \times R \to \{-1, 0, +1\}$ given by*

$$\mathsf{GT}(a, c(x), N) = \left(\frac{c_0^2 - c_1^2 a}{N}\right).$$

Define the subset $G_a \subset R_a$ as follows:

$$G_a = \{c(x) \in R_a : \mathsf{GT}(a, c(x), N) = 1\}.$$

Therefore, this is the subset of R_a that passes Galbraith's test. Define the subset $\bar{G_a} \subset R_a$ as follows:

$$\bar{G_a} = \{c(x) \in R_a : \mathsf{GT}(a, c(x), N) = -1\}.$$

Correspondingly, this is the subset of R_a that fails Galbraith's test. Now define the subset $S_a \subset G_a$:

$$S_a = \{2hx + (t + ah^2t^{-1}) \in G_a \mid h \in \mathbb{Z}_N, t, (t + ah^2t^{-1}) \in \mathbb{Z}_N^*\}.$$

The subset S_a is precisely the image of the algorithm $\mathcal{E}$ defined in Chapter 4, Section 4.4.5, which takes as input an integer $a \in \mathbb{J}(N)$ (i.e. $\left(\frac{a}{N}\right) = 1$) along with a message bit $m \in \{0, 1\}$ and produces an element of S_a that encrypts m. This is central to the XOR-homomorphic scheme xhIBE from Chapter 4. Like Cocks' original scheme, xhIBE requires a ciphertext to have two components. As such, $\mathcal{E}$ can be viewed as the encryption algorithm for a single component. Accordingly, to encrypt a message m in xhIBE, the sender runs $\mathcal{E}(a, m)$ and $\mathcal{E}(-a, m)$ to produce the first and second component of a ciphertext respectively.

Let $g(x) \in \bar{G_a}$. Below are some basic facts which we prove in Section C.0.4.

1. $g(x)G_a = \bar{G_a}$.

2. $\{h(x) \xleftarrow{\$} \bar{G_a}\} \approx \{g(x)h'(x) \mid h'(x) \xleftarrow{\$} G_a\}$.

3. $\{h(x) \xleftarrow{\$} \bar{G_a}\} \underset{C}{\approx} \{g(x)h'(x) \mid h'(x) \xleftarrow{\$} S_a\}$.

Property 3 states that the uniform distribution defined over $\bar{G_a}$ and the distribution of multiplying $g(x)$ by uniformly random elements from S_a are computationally indistinguishable (without access to p and q).

We need two hash functions. Like Cocks' scheme, a full-domain hash $H : \{0,1\}^* \rightarrow \mathbb{J}(N)$ is employed that maps identity strings to elements of $\mathbb{Z}_N$ whose Jacobi symbol is $+1$. Another hash function $H' : \{0,1\}^* \rightarrow R$ is needed that maps an identity string id to an element $g(x) \in R$ such that $\mathsf{GT}(H(\mathsf{id}), g(x), N) = \mathsf{GT}(-H(\mathsf{id}), g(x), N) = -1$ i.e. the $g(x)$ is taken to pass Galbraith's test for both $a = H(\mathsf{id})$ and $-a$. Roughly speaking, an example of constructing such as hash function using H is via a form of rejection sampling i.e. to sample $g'(x)_i \xleftarrow{\$} H(\mathsf{id} \parallel i)$ for consecutive integers $i > 0$ until $\mathsf{GT}(a, g'(x)_i, N) = \mathsf{GT}(-a, g'(x)_i, N) = -1$. In the security proofs, H is modelled as a random oracle on $\mathbb{J}(N)$ and H' is modelled as a random oracle whose response when queried on id is distributed according to the uniform distribution on $\bar{G}_{H(\mathsf{id})} \cap \bar{G}_{-H(\mathsf{id})}$. To anonymize a ciphertext *component* (recall that this discussion is simplified to deal with a single component of a ciphertext corresponding to $a = H(\mathsf{id})$, the steps are repeated for the case of $-a$) $c(x)$ associated with an identity id, the following steps are performed:

1. $a \leftarrow H(\mathsf{id})$

2. $c'(x) \leftarrow \mathcal{E}(a, 0)$.

3. Uniformly sample a bit $b \xleftarrow{\$} \{0,1\}$.

4. If $b = 0$, output $c'(x)c(x)$.

5. Else compute $g(x) \leftarrow H'(\mathsf{id})$, and output $g(x)c(x)c'(x)$.

Note that the construction is universally anonymous in that anyone can anonymize a ciphertext without having the secret key for the target identity and without access to the random coins used by the encryptor.

The decryption function $\mathcal{D}'$ for our construction is defined in terms of $\mathcal{D}$.

$$\mathcal{D}'(r, c(x)) = \begin{cases} \mathcal{D}(r, c(x)) \oplus \mathcal{D}(r, g(x)) & \text{if } c(x) \in \bar{G}_a \\ \mathcal{D}(r, c(x)) & \text{if } c(x) \in G_a \\ \perp & \text{otherwise} \end{cases}$$

C.0.3 Formal Description

Our scheme is referred to as UAIBE for the remainder of the paper; a formal description is as follows. **$\mathsf{Setup}(1^\lambda)$:** On input a security parameter 1^λ in unary, generate $(p, q) \leftarrow \mathsf{BlumGen}(1^\lambda)$. Compute $N = pq$. Output public parameters $\mathsf{PP} = (N, H, H')$ and master secret key $\mathsf{MSK} = (N, p, q)$, where H is a hash function $H : \{0,1\}^* \to \mathbb{J}(N)$, and H' is a hash function $H' : \{0,1\}^* \to R$ with the property that for any identity $\mathsf{id} \in \{0,1\}^*$, $a \leftarrow H(\mathsf{id})$ and $g(x) \leftarrow H'(\mathsf{id})$, it holds that

$$\mathsf{GT}(a, g(x), N) = \mathsf{GT}(-a, g(x), N) = -1.$$

$\mathsf{KeyGen}(\mathsf{MSK}, \mathsf{id})$: On input master secret key $\mathsf{MSK} = (N, p, q)$ and identity $\mathsf{id} \in \{0,1\}^*$, perform the following steps:

1. Compute $a \leftarrow H(\mathsf{id}) \in \mathbb{J}(N)$.
2. If $r \in \mathbb{QR}(N)$, compute the square root $r = a^{1/2}$;
3. Else compute $r = (-a)^{1/2}$.
4. Output (N, id, r) as the secret key for identity id.

See the description of Cocks' scheme in Section 4.4.4 for a convenient way to compute a square root in $\mathbb{Z}_N$ deterministically.

Encrypt($\mathsf{PP}, \mathsf{id}, m$): On input public parameters $\mathsf{PP} = (N, H, H')$, an identity $\mathsf{id} \in \{0,1\}^*$, and message $m \in \{0,1\}$ run:

1. Compute $a \leftarrow H(\mathsf{id}) \in \mathbb{J}(N)$.
2. Compute $g(x) \leftarrow H'(\mathsf{id}) \in R$.
3. Compute $c(x) \leftarrow \mathcal{E}(a, m)$.
4. Compute $d(x) \leftarrow \mathcal{E}(-a, m)$.
5. Uniformly sample two bits $v_1, v_2 \xleftarrow{\$} \{0,1\}$.
6. If $v_1 = 1$, then set $c(x) \leftarrow c(x) * g(x)$.
7. If $v_2 = 1$, then set $d(x) \leftarrow d(x) * g(x)$.
8. Output $\vec{c} := (c(x), d(x))$.

Decrypt($\mathsf{sk}_{\mathsf{id}}, \vec{c}$): On input a secret key $\mathsf{sk}_{\mathsf{id}} = (N, \mathsf{id}, r)$ and a ciphertext $\vec{c} = (c(x), d(x))$, do:

1. Compute $a \leftarrow H(\mathsf{id}) \in \mathbb{J}(N)$.
2. Compute $g(x) \leftarrow H'(\mathsf{id}) \in R$.
3. If $r^2 \equiv a \mod N$, set $e(x) \leftarrow c(x)$. Else if $r^2 \equiv -a \mod N$, set $e(x) \leftarrow d(x)$. Else output $\perp$ and abort.
4. If $\mathsf{GT}(r^2 \mod N, e(x)) = -1$, set $e(x) \leftarrow e(x) * g(x)$.
5. Output $\mathcal{D}(r, e(x))$.

C.0.4 Security

Lemma C.0.1. *Let* $f(x), g(x) \in R_a$. *Then* $\mathsf{GT}(a, f(x)g(x), N) = \mathsf{GT}(a, f(x), N) \cdot \mathsf{GT}(a, g(x), N)$.

Proof. Consider the product $v(x) = f(x)g(x) \in R_a$. We have that $v_0 = f_0g_0 + f_1g_1a$ and $v_1 = f_0g_1 + f_1g_0$. It is easy to verify that

$$\left(\frac{(f_0g_0 + f_1g_1a)^2 - (f_0g_1 + f_1g_0)^2a}{N}\right) = \left(\frac{(f_0^2 - af_1^2)(g_0^2 - ag_1^2)}{N}\right) = \mathsf{GT}(a, f(x), N)\cdot\mathsf{GT}(a, g(x), N).$$

. □

Lemma C.0.2. *Let $g(x) \in \bar{G_a}$. Then $g(x) \cdot G_a = \bar{G_a}$.*

Proof. By Lemma C.0.1, $g(x)h(x) \in \bar{G_a}$ for any $h(x) \in G_a$.

By Lemma 1 in [60], G_a is a multiplicative group in R_a. Hence, $|g(x)\cdot G_a| = |G_a|$. We claim that every $t(x) \in \bar{G_a}$ can be expressed as $g(x)t'(x)$ for some $t'(x) \in G_a$. Assume the contrary for the purpose of contradiction i.e. there exists a $t(x) \notin g(x)\cdot G_a$. It follows that $t(x) \cdot G_a \cap g(x) \cdot G_a = \emptyset$. But by Lemma C.0.1, $t(x)^2 \in G_a$ and $g(x)t(x) \in G_a$. From the commutativity of R_a, we have $g(x) \cdot t(x)^2 = t(x) \cdot (t(x)g(x))$, which implies that $t(x) \cdot G_a \cap g(x) \cdot G_a \neq \emptyset$, a contradiction. The lemma follows. □ □

We include the following result from [60] that is used in the proofs below.

Corollary C.0.1. *Let $g(x) \in \bar{G_a}$. Then*

1. $\{h(x) \xleftarrow{\$} \bar{G_a}\} \approx \{g(x)h'(x) \mid h'(x) \xleftarrow{\$} G_a\}$.
2. $\{h(x) \xleftarrow{\$} \bar{G_a}\} \underset{C}{\approx} \{g(x)h'(x) \mid h'(x) \xleftarrow{\$} S_a\}$.

Proof. (1). From Lemma C.0.2, each element in $\bar{G_a}$ can be represented as $g(x)h'(x)$ for a unique $h'(x) \in G_a$. Therefore, if $h'(x)$ is sampled uniformly from G_a, then $h'(x)g(x)$ is uniformly distributed in $\bar{G_a}$.

(2). By Corollary 4.4.1 in Chapter 4, $G_a \underset{C}{\approx} S_a$ without knowledge of the prime factors of N, and thus this property follows from (1). □ □

Theorem C.0.1. *UAIBE is ANON-IND-ID-CPA-secure in the random oracle model assuming the hardness of the quadratic residuosity problem.*

Proof. We prove the theorem by showing that a poly-bounded adversary has a negligible advantage distinguishing between the following series of games.

Game 0 This is the ANON-IND-ID-CPA game between the challenger and an adversary $\mathcal{A}$ with the scheme UAIBE as described in Section C.0.3.

Game 1 The only change in this game from Game 0 is as follows. Let b denote the bit chosen by the challenger to choose either between the tuples (id_0, m_0) or (id_1, m_1) supplied by the adversary. Let $a = H(\mathsf{id}_b)$. Instead of encrypting m_b, we instead encrypt a random bit $b' \in \{0,1\}$ i.e. we have $c(x) \leftarrow \mathcal{E}(a, b')$ and $d(x) \leftarrow \mathcal{E}(-a, b')$.

We argue that if there is an efficient distinguisher $\mathcal{A}$ that can distinguish between Game 0 and Game 1, then there is efficient adversary $\mathcal{B}$ that can use $\mathcal{A}$ to attack the IND-ID-CPA security of xhIBE. Secret key queries from $\mathcal{A}$ are relayed to $\mathcal{B}$'s oracle. When $\mathcal{A}$ chooses its challenge tuples (id_0, m_0) and (id_1, m_1), perform the following:

1. If $b' = m_b$, output a random bit and abort.
2. Else choose challenge identity $\mathsf{id}^* = \mathsf{id}_b$.
3. When $\mathcal{B}$'s IND-ID-CPA challenger responds with a challenge ciphertext $(c(x)^*, d(x)^*)$, choose two random bits $u_0, u_1 \xleftarrow{\$} \{0,1\}$: if $u_0 = 1$, set $c(x)^* \leftarrow c(x)^* g(x)$; if $u_1 = 1$, set $d(x)^* \leftarrow d(x)^* g(x)$ where $g(x) \leftarrow H'(\mathsf{id}^*)$ (this oracle can be provided by $\mathcal{B}$).
4. Give $(c(x)^*, d(x)^*)$ to $\mathcal{A}$, and output $\mathcal{A}$'s guess.

If $\mathcal{A}$ has advantage ϵ distinguishing games Game 0 and Game 1, then $\mathcal{B}$ has an advantage of $\frac{1}{2}\epsilon$.

Game 2 To recap, note that the challenge ciphertexts in Game 1 have the distribution $\{(c(x), d(x)) \xleftarrow{\$} S_a \times S_{-a} : a = H(\mathsf{id}_b), b \xleftarrow{\$} \{0,1\}\}$. This is because by definition for any $a \in \mathbb{J}(N)$, we have $S_a = \mathsf{image}(\mathcal{E}(a, \cdot))$ and $S_{-a} = \mathsf{image}(\mathcal{E}(-a, \cdot))$. The next step is to replace S_a with G_a. Instead of setting $c(x) \leftarrow \mathcal{E}(a, b')$ where $a = H(\mathsf{id}_b)$, we choose $c(x) \xleftarrow{\$} G_a$.

Corollary 4.4.1 in Chapter 4 shows that $S_a \underset{C}{\approx} G_a$ for any $a \in \mathbb{J}(N)$ without access to the factorization of N. We follow a similar argument to the above to "embed" the challenge element from either S_a or G_a. We handle secret key queries without the factors of N by programming the oracle responses from H. Suppose the adversary queries the secret key for an identity id'. Assume without loss of generality that it first queries the random oracle H on id'. On the first such query, we uniformly sample a secret key $r' \xleftarrow{\$} \mathbb{Z}_N^*$, set $a' \leftarrow r'^2 \mod N \in \mathbb{J}(N)$, store the tuple (id', r', a') and return a'. This has the correct distribution and secret keys can easily be extracted. A non-negligible advantage distinguishing Game 1 and Game 2 translates to a non-negligible advantage distinguishing the distributions S_a and G_a, which contradicts Corollary 2 in [60].

Game 3 The change from Game 2 to Game 3 is similar to that from Game 1 to Game 2, namely the second ciphertext component $d(x)$ is sampled from G_{-a} instead of S_{-a} where $a = H(\mathsf{id}_b)$. The argument for indistuinguishability is analogous to that of the last game.

Game 4 This game is identical to Game 3 except that instead of setting $a \leftarrow H(\mathsf{id}_b)$, we instead set $a \xleftarrow{\$} \mathbb{J}(N)$. Furthermore, step 2 of $\mathsf{Encrypt}$ is replaced with $g(x) \leftarrow \bar{G}_a \cup \bar{G}_{-a} \in R$.

Clearly, the adversary has a zero advantage in this game since a ciphertext reveals nothing about the challenger's bit b. We now show that a ciphertext in Game 4 is indistinguishable from a ciphertext in Game 3. Observe that *each component* of the latter is computationally indistinguishable from a uniformly random element of the set of units in R. The units in R are precisely those elements $u(x)$ satisfying

$$\mathsf{GT}(a', u(x), N) \in \{-1, 1\}$$

with respect to any $a' \in \mathbb{J}(N)$; that is, the set of units is $G_{a'} \cup \bar{G}_{a'}$.

In Game 3, half of the time the ciphertext component $c(x)$ (resp. $d(x)$) is uniformly distributed in $\bar{G}_a$ (resp. $\bar{G}_{-a}$) according to Corollary C.0.1, and the other half it is uniformly distributed in G_a (resp. G_{-a}), by definition of Game 3. Thus, each component

is a uniformly random element of the set of units in R. But similarly, we have that each component of a ciphertext in Game 4 is also uniformly distributed in the set of units in R. Therefore, both games are indistinguishable to a poly-bounded adversary.

We can conclude that an adversary's advantage is negligible distinguishing between Game 0 and Game 4, which implies that its advantage attacking the ANON-IND-ID-CPA security of UAIBE is also negligible. □ □

C.0.5 Comparison with Ateniese and Gasti's Construction

Our proposed construction has several advantages. Firstly, it is arguably conceptually simpler than existing anonymous variants of Cocks' scheme. Furthermore, like the construction put forward in [20], it is universally anonymous, which may be useful in settings where messages pass through multiple systems, some of which need to know the recipient's identity whereas others should not be privy to this information. Hence, a trusted proxy can be tasked with anonymizing ciphertexts without access to the secret key. The scheme is also group-homomorphic for the XOR operation; this is useful in some settings as discussed in [60], although anonymity must be sacrificed for homomorphic operations to be performed. Another advantage of our scheme is that it faster run-time performance than other anonymous IBEs based on quadratic residuosity. We elaborate more on its performance in this section by comparing it to its nearest rival (in terms of run-tie performance), namely the Ateniese and Gasti (AG) scheme from [20]. However, the most significant downside of the scheme is its poor space efficiency; ciphertext expansion is double that of Cocks, and almost double that of AG.

C.0.6 Analysis of Ateniese and Gasti's Construction (AG)

Encryption in the AG scheme requires a number of Galbraith test computations per bit of plaintext. Recall that evaluating a Galbraith test entails a costly Jacobi symbol computation. The main intuition behind AG is to "embed" a Cocks ciphertext within a sequence of integers T_i. Its position, k, in such a sequence is distributed according to a

geometric distribution with parameter $p = 1/2$. Furthermore, the terms $T_1, \ldots, T_{k-1}$ are chosen such that $\mathsf{GT}(a, T_i, N) = -1$ for $i \in [k-1]$. The intuition behind this approach is grounded in the fact that Galbraith's test can be shown (see Section 2.3 in [20]) to be the "best test" possible in attacking the anonymity of Cocks' scheme. Since the probability of a random element in $\mathbb{Z}_N^*$ passing Galbraith's test is $1/2$, the position of the first element in a random sequence to pass Galbraith's test is distributed according to a geometric distribution with parameter $p = 1/2$. A hash function is used to generate the sequence of integers based on short binary strings incorporated in an AG ciphertext. It sufficient here to note that ℓ is a global parameter in AG that determines the number of such binary strings (this is closely related to the number of Galbraith tests that must be performed on average during encryption).

Let Y be a random variable representing the number of Galbraith tests evaluated in AG per bit of plaintext. A lower bound for the expected value $E[Y]$ of Y can be derived as

$$E[Y] \geq 4(1 + (\log \kappa - 1) \cdot 2^{-\ell})$$

where κ is the security parameter. A rough lower bound on the variance $\mathsf{Var}(Y)$ is

$$\mathsf{Var}(Y) \geq 2^{2-2\ell}(-8 + 7 \cdot 2^{2\ell} + 2^{1+\ell} - 3 \cdot 2^{2+\ell}\ell).$$

Ateniese and Gasti found $\ell = 6$ to be a good compromise between ciphertext size and performance. Setting $\ell = 6$ results in a mean number of Galbraith tests per bit of plaintext of ≈ 4.22 with a standard deviation of ≈ 6.92. Our scheme on the other hand does not require any Galbraith test to be performed during encryption.

C.1 Experimental Results

To perform an empirical comparison between our scheme and AG, both schemes were implemented in C using the OpenSSL library. Our implementation was based on code provided by the authors of [20]. The following experiment was run for each of the four

schemes: Cocks, AG, UAIBE and JB. The latter is a shorthand for our modification to the construction of Jhanwar and Barua described in [121], which in turn is a variant of the non-anonymous IBE system from [39]. Note that JB is not anonymous and its inclusion here is to demonstrate the fact that it achieves comparable efficiency to Cocks. Hence, AG and UAIBE are the two anonymous schemes being compared.

1. For each t in the set $\{1024, 2048, 3072, 4096\}$:
   (a) A modulus N of t bits is generated along with primes p and q that constitute the master secret key.
   (b) The public key a and secret key r are derived for some predefined identity string id. A random 128-bit message m is generated.
   (c) The following is repeated 50 times:
      i. Encrypt m under identity id to produce ciphertext c.
      ii. Decrypt c with secret key r and verify the decrypted message matches m.
      iii. The time elapsed performing step 3.(a) and 3.(b) is calculated.
   (d) An average over the times calculated in step 3.(c) is obtained.

The code was compiled with optimization flag '-02' using GCC version 4.4.5-8 with OpenSSL version 0.9.8o. The benchmarks were executed on a machine with 4 GB of RAM and an Intel Core i5-3340M CPU clocked at 2.70 GHz. The benchmark machine was running GNU/Linux 3.2.41 (x86-64). Our implementation however was unoptimized and did not exploit parallelization. For the interested reader, the implementation of encryption in Cocks, AG and UAIBE involved precomputation of random integers with Jacobi symbol -1 and $+1$. This is not needed for JB.

The results of the experiment (average encryption times) are shown in Figure C.1. Note that UAIBE and Cocks exhibit similar performance whereas JB is only marginally less efficient than Cocks. On the other hand, AG performs notably worse than UAIBE

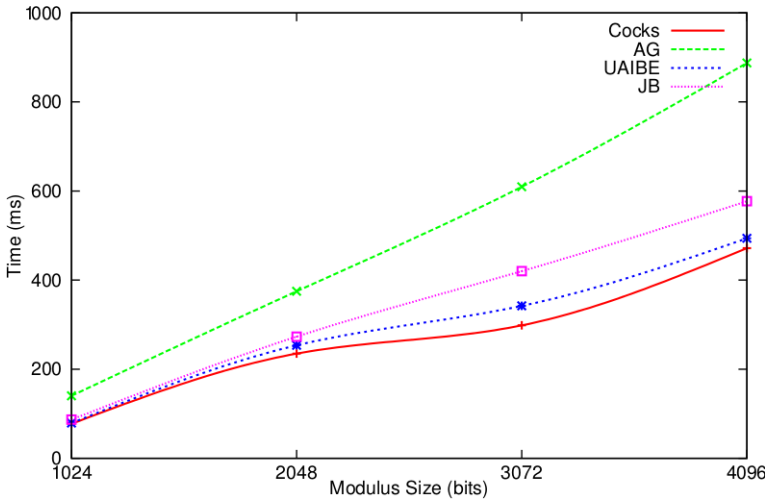

Fig. C.1: Average times to encrypt a 128-bit message for Cocks, AG and UAIBE.

on average. To illustrate the comparison, encryption and decryption times for all four schemes for the case of a 1024-bit modulus are presented in Table C.1.

Table C.1: Encryption and decryption times in milliseconds for a 128-bit message with a key size of 1024 bits, averaged over 50 runs.

Scheme	Encryption -Mean (Std Dev)	Decryption - Mean (Std Dev)
Cocks	77.39 (3.05)	13.32 (0.14)
AG	140.35 (19.22)	40.79 (1.68)
UAIBE	79.02 (3.14)	27.52 (0.41)
JB	86.78 (0.93)	21.97 (0.42)

Appendix D

Compilers for Bootstrappable IBFHE

D.1 A Compiler to Transform a Leveled IBFHE into a "Pure" IBFHE

So far we have obtained "pure" IBFHE, ABFHE and multi-attribute ABFHE schemes. Although these constructions are impractical at the current time, they serve as possibility results for these primitives. Next we turn our attention to obtaining a "compiler" to transform an arbitrary leveled IBFHE into a bootstrappable IBFHE, and as a consequence, a "pure" IBFHE. One of the primary reasons for this is efficiency. One of the reasons our previous constructions are impractical is that they rely on indistinguishability obfuscation for the frequently used process of deriving a public-key for a user's identity. With appropriate parameters, bootstrapping is a process that might be carried out infrequently - or needed only in especially rare occasions. Therefore, preserving the performance of existing leveled IBFHEs for encryption, decryption and evaluation of "not-too-deep" circuits is desirable. But having the capability to bootstrap, even if expensive, is useful in those cases where evaluation of a deep circuit is needed. This is

particularly true in the identity-based setting because keys cannot be generated on a once-off basis as they might be in many applications* of public-key FHE, nor can they be changed as frequently, since all users of the identity-based infrastructure are affected.

Intuitively, the central idea to make a leveled IBFHE scheme bootstrappable is as follows. Firstly, we include an obfuscation of a program in the public parameters. This program "hides" the master secret key (trapdoor) of the scheme. Such a program can use the trapdoor to generate a secret key for an identity, and then use that secret key to output a bootstrapping key that is derived from the secret key. Hence, an evaluator can run the obfuscated program to non-interactively accomplish bootstrapping.

However in order to prove selective security of such a scheme, we need to remove all secret key information for the adversary's target identity. The reason for this is that our obfuscator is not a virtual black-box obfuscator i.e. we cannot argue that the obfuscated program leaks no information about the trapdoor to the adversary. Therefore, certain properties are needed of a leveled IBFHE scheme $\mathcal{E}$ before it is admissible for our "compiler".

D.1.1 Bootstrappable IBFHE

Let us recall the definition of leveled IBFHE. This definition is for the single-identity setting, which we restrict ourselves for the moment to simplify notation.

Definition D.1.1. *A Leveled IBFHE scheme with message space $\mathcal{M}$, identity space $\mathcal{I}$, a class of circuits $\mathbb{C} \subseteq \mathcal{M}^* \to \mathcal{M}$ and ciphertext space $\mathcal{C}$ is a tuple of PPT algorithms* $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt}, \mathsf{Eval})$ *defined as follows:*

- $\mathsf{Setup}(1^\lambda, 1^L)$*:*

  On input (in unary) a security parameter λ, and a number of levels L (maximum

*For many applications of public-key FHE, leveled FHE is usually adequate because a new key pair can be generated on a once-off basis for a particular circuit, whose depth is known, and a leveled FHE can be parameterized accordingly.

circuit depth to support) generate public parameters PP *and a master secret key* MSK*. Output* $(\mathsf{PP}, \mathsf{MSK})$.

- KeyGen, $\mathsf{Encrypt}$ *and* $\mathsf{Decrypt}$ *are defined the same as IBE.*

- $\mathsf{Eval}(\mathsf{PP}, C, c_1, \ldots, c_\ell)$*: On input public parameters* PP*, a circuit* $C \in \mathbb{C}$ *and ciphertexts* $c_1, \ldots, c_\ell \in \mathcal{C}$*, output an* evaluated ciphertext $c' \in \mathcal{C}$.

More precisely, the scheme is required to satisfy the following properties:

- *Over all choices of* $(\mathsf{PP}, \mathsf{MSK}) \leftarrow \mathsf{Setup}(1^\lambda)$, $\mathsf{id} \in \mathcal{I}$, $C : \mathcal{M}^\ell \to \mathcal{M} \in \{C \in \mathbb{C} : \mathsf{depth}(C) \leq L\}$, $\mu_1, \ldots, \mu_\ell \in \mathcal{M}$, $c_i \leftarrow \mathsf{Encrypt}(\mathsf{PP}, \mathsf{id}, \mu_i)$ *for* $i \in [\ell]$*, and* $c' \leftarrow \mathsf{Eval}(\mathsf{PP}, C, c_1, \ldots, c_\ell)$*:*

  - ***Correctness***

    $$\mathsf{Decrypt}(\mathsf{sk}, c') = C(\mu_1, \ldots, \mu_\ell) \tag{D.1.1}$$

    for any $\mathsf{sk} \leftarrow \mathsf{KeyGen}(\mathsf{MSK}, \mathsf{id})$.

  - ***Compactness***

    $$|c'| = \mathsf{poly}(\lambda) \tag{D.1.2}$$

In a leveled fully homomorphic encryption scheme, the size of the public parameters along with the size of keys are allowed to depend on L.

There are different ways to define bootstrapping; the formulation here was chosen to best fit with the results in this chapter. We assume without loss of generality that the class of circuits $\mathbb{C}$ supported by the scheme is built from a set of binary operations e.g: $\{\oplus, \odot\}$ i.e. $\oplus : \mathcal{M} \times \mathcal{M} \to \mathcal{M}$ and $\odot : \mathcal{M} \times \mathcal{M} \to \mathcal{M}$.

Definition D.1.2. *A leveled IBFHE is said to be bootstrappable if there exists a pair of PPT algorithms*
$(\mathsf{GenBootstrapKey}, \mathsf{Bootstrap})$ *defined as follows:*

- $\mathsf{GenBootstrapKey}(\mathsf{PP}, \mathsf{id})$: *takes as input public parameters* PP *and an identity* id*, and outputs a* bootstrapping key $\mathsf{bk}_{\mathsf{id}}$.

- $\mathsf{Bootstrap}(\mathsf{PP}, \mathsf{bk}_{\mathsf{id}}, c)$ *takes as input public parameters* PP*, a* bootstrapping key $\mathsf{bk}_{\mathsf{id}}$ *for identity* id*, and a ciphertext* $c \in \mathcal{C}$*, and outputs a ciphertext* $c' \in \mathcal{C}$*.*

Over all $(\mathsf{PP}, \mathsf{MSK})$*: for every pair of ciphertexts* $c_1, c_2 \in \mathcal{C}$*, all identities* id *and all secret keys* $\mathsf{sk}_{\mathsf{id}}$ *and for all* $\circ \in \{\oplus, \odot\}$*:*

$$\begin{aligned} &\mathsf{Decrypt}(\mathsf{sk}_{\mathsf{id}}, \mathsf{Eval}(\circ, \mathsf{Bootstrap}(\mathsf{PP}, \mathsf{id}, c_1), \mathsf{Bootstrap}(\mathsf{PP}, \mathsf{id}, c_2)) \\ &= \mathsf{Decrypt}(\mathsf{sk}_{\mathsf{id}}, c_1) \circ \mathsf{Decrypt}(\mathsf{sk}_{\mathsf{id}}, c_2). \end{aligned}$$

Informally, what the above definition says is that at least one additional homomorphic operation (either $\oplus$ or $\odot$) can be applied to a pair of "refreshed" (i.e. bootstrapped) ciphertexts before bootstrapping is needed again. For a more thorough discussion on bootstrapping, we refer the reader to [93].

D.1.2 Weakly-bootstrappable IBFHE

Our starting point is leveled IBFHE schemes, such as those constructed via the GSW compiler from [98], that support bootstrapping when given "encryptions" of secret key bits. We refer to such "encryptions" of secret key bits as a *bootstrapping key*. As mentioned earlier, there is no known way (in current schemes) to non-interactively derive a bootstrapping key for a given identity from the public parameters alone. The only way bootstrapping can be achieved in such schemes is when a bootstrapping key is passed to the evaluator out-of-band, which breaks an attractive property of IBE, namely that all keys are derivable from the public parameters and a user's identity alone.

We now give a formal definition for a leveled IBFHE that supports bootstrapping when supplied with a bootstrapping key, and we say such a scheme is *weakly bootstrappable*. The main difference between *weakly bootstrappable* and *bootstrappable* (see Definition D.1.2) is that the former requires a secret key for an identity in order to generate a bootstrapping key, whereas the latter only needs an identity. Note that the leveled IBFHEs from [98] are weakly bootstrappable.

Definition D.1.3. *A leveled IBFHE scheme $\mathcal{E}$ is said to be weakly bootstrappable if there exists a pair of PPT algorithms* $(\mathsf{WGenBootstrapKey}, \mathsf{Bootstrap})$ *where* Bootstrap *is defined as in Definition D.1.2 and* WGenBootstrapKey *is defined as follows:*

- $\mathsf{WGenBootstrapKey}(\mathsf{PP}, \mathsf{sk}_{\mathsf{id}})$: *takes as input public parameters* PP *and a secret key* $\mathsf{sk}_{\mathsf{id}}$ *for identity* id*, and outputs a* bootstrapping key $\mathsf{bk}_{\mathsf{id}}$.

Like a bootstrappable leveled IBFHE, a weakly-bootstrappable leveled IBFHE requires a circular security assumption to be made to prove IND-sID-CPA security. This is because an adversary is given $\mathsf{bk}_{\mathsf{id}^*}$ for her target identity id^*, which consists of encryptions of secret key bits.

D.1.3 Single-Point Trapdoor Puncturability

The next requirement we place on a leveled IBFHE to work with our compiler is called single-point trapdoor puncturability. Intuitively, this means that there is a way to "puncture" the master secret key (aka trapdoor) T to yield a *proper subset* $T' \subset T$ that is missing information needed to derive a secret key for a given identity id^*. Furthermore, for all other identities $\mathsf{id} \neq \mathsf{id}^*$, the punctured trapdoor contains enough information to efficiently derive the *same* secret key for id as one would derive with the original trapdoor T, assuming we are given the same randomness. A formal definition will help to elucidate this notion.

Definition D.1.4. *An IBE scheme $\mathcal{E}$ is single-point trapdoor-puncturable if there exists PPT algorithms* TrapPuncture *and* SimKeyGen *with*

- $\mathsf{TrapPuncture}(T, \mathsf{id}^*)$*: On input trapdoor T and identity* id^**, output a "punctured trapdoor" $T' \subset T$ with respect to* id^*.

- $\mathsf{SimKeyGen}(T', \mathsf{id})$*: On input a "punctured trapdoor" T' with respect to some identity* id^**, and an identity* id*, output a secret key for* id *if* $\mathsf{id} \neq \mathsf{id}^*$*, and $\perp$ otherwise.*

and these algorithms satisfy the following conditions for any $(\mathsf{PP}, T) \leftarrow \mathcal{E}.\mathsf{Setup}(1^\lambda)$, $\mathsf{id}^* \in \mathcal{I}$ *and* $T' \leftarrow \mathsf{TrapPuncture}(T, \mathsf{id}^*) \subset T$*:*

$$\mathcal{E}.\mathsf{KeyGen}(T, \mathsf{id}) = \mathsf{SimKeyGen}(T', \mathsf{id}) \quad \forall \mathsf{id} \in \mathcal{I} \setminus \{\mathsf{id}^*\}. \tag{D.1.3}$$

D.1.4 Our Compiler

Let $\mathcal{E}$ be a *leveled* IBFHE scheme. The required properties that $\mathcal{E}$ must satisfy for compatibility with our compiler are:

Property 1: (**Weakly-Bootstrappable**) $\mathcal{E}$ is weakly-bootstrappable i.e. there exists a pair of PPT algorithms $(\mathsf{WGenBootstrapKey}, \mathsf{Bootstrap})$ satisfying Definition D.1.3.

Property 2: (**Single-Point Trapdoor-Puncturable**) $\mathcal{E}$ is single-point trapdoor-puncturable i.e. there exists a pair of PPT algorithms $(\mathsf{TrapPuncture}, \mathsf{SimKeyGen})$ satisfying Definition D.1.4.

Property 3: (**Indistinguishability given punctured trapdoor**) For all $\mathsf{id} \in \mathcal{I}$ and $m \in \mathcal{M}$: for every $\mathsf{sk}_{\mathsf{id}^*} \leftarrow \mathcal{E}.\mathsf{KeyGen}(T, \mathsf{id}^*)$, and $\mathsf{bk}_{\mathsf{id}^*} \leftarrow \mathsf{WGenBootstrapKey}(\mathsf{PP}, \mathsf{sk}_{\mathsf{id}^*})$, the distributions

$$\{(\mathsf{PP}, T', \mathsf{bk}_{\mathsf{id}^*}, \mathcal{E}.\mathcal{E}.\mathsf{Encrypt}(\mathsf{PP}, \mathsf{id}^*, m)\} \underset{C}{\approx} \{(\mathsf{PP}, T', \mathsf{bk}_{\mathsf{id}^*}, \mathcal{E}.\mathcal{E}.\mathsf{Encrypt}(\mathsf{PP}, \mathsf{id}^*, m')) : m' \xleftarrow{\$} \mathcal{M}\}$$

are computationally indistinguishable.

There are concrete schemes that *almost* meet all three properties. One such example is the leveled IBFHE from Appendix A of [98]. This scheme admits algorithms $(\mathsf{TrapPuncture}, \mathsf{SimKeyGen})$ that satisfy a relaxation of Equation D.1.3 in Definition D.1.4, namely the requirement of equality is relaxed to statistical indistinguishability; more precisely it holds that

$$\mathcal{E}.\mathsf{KeyGen}(T, \mathsf{id}) \underset{S}{\approx} \mathsf{SimKeyGen}(T', \mathsf{id}) \quad \forall \mathsf{id} \in \mathcal{I} \setminus \{\mathsf{id}^*\}$$

for any $\mathsf{id} \in \mathcal{I}$. However, we have been unable to find a leveled IBFHE scheme (from the GSW compiler) that meets the stronger condition of Equation D.1.3.

Note that it is only necessary that $\mathsf{SimKeyGen}$ run in polynomial time - the essential challenge is to derive some "canonical" secret key for an identity given *less* trapdoor information (but the same randomness).

D.1.4.1 Formal Description

We now proceed with a formal description of a bootstrappable scheme $\hat{\mathcal{E}}_1$ that is constructed using a scheme $\mathcal{E}$ satisfying the above properties. Let $(\mathsf{WGenBootstrapKey}, \mathsf{Bootstrap})$ be a pair of PPT algorithms meeting Property 1.

Consider the following program F_{GenBK} to generate a bootstrapping key:

Program $F_{\mathsf{GenBK}}(\mathsf{id})$:

1. Compute $r_1 \parallel r_2 \leftarrow \mathsf{PRF.Eval}(K, \mathsf{id})$.
2. Compute $\mathsf{sk}_{\mathsf{id}} \leftarrow \mathsf{KeyGen}(T, \mathsf{id}; r_1)$.
3. **Output** $\mathsf{WGenBootstrapKey}(\mathsf{PP}_{\mathcal{E}}, \mathsf{sk}_{\mathsf{id}}; r_2)$.

The scheme $\hat{\mathcal{E}}_1$ includes an obfuscation of this program (with key K and trapdoor T) for the purpose of bootstrapping:

- $\hat{\boldsymbol{\mathcal{E}}}_{\mathbf{1}}\mathbf{.Setup}(1^\lambda)$: Set $(\mathsf{PP}_{\mathcal{E}}, T) \leftarrow \mathcal{E}.\mathsf{Setup}(1^\lambda)$. Compute $K \leftarrow \mathsf{PRF.Key}(1^\lambda)$. Compute $\beta \leftarrow i\mathcal{O}(F_{\mathsf{GenBK}})$. Output $(\mathsf{PP} := (\mathsf{PP}_{\mathcal{E}}, \beta), \mathsf{MSK} := T)$.
- $\hat{\boldsymbol{\mathcal{E}}}_{\mathbf{1}}\mathbf{.KeyGen} = \mathcal{E}.\mathsf{KeyGen}$; $\hat{\boldsymbol{\mathcal{E}}}_{\mathbf{1}}\mathbf{.Encrypt} = \mathcal{E}.\mathsf{Encrypt}$; $\hat{\boldsymbol{\mathcal{E}}}_{\mathbf{1}}\mathbf{.Decrypt} = \mathcal{E}.\mathsf{Decrypt}$.
- $\hat{\boldsymbol{\mathcal{E}}}_{\mathbf{1}}\mathbf{.Bootstrap}(\mathsf{PP}, \mathsf{id}, c)$: Parse PP as $(\mathsf{PP}_{\mathcal{E}}, \beta)$. Set $\mathsf{bk}_{\mathsf{id}} \leftarrow \beta(\mathsf{id})$. Output $\mathsf{Bootstrap}(\mathsf{PP}_{\mathcal{E}}, \mathsf{bk}_{\mathsf{id}}, c)$.

The main idea is that $\hat{\mathcal{E}}_1$ includes an obfuscation $\beta \leftarrow i\mathcal{O}(F_{\mathsf{GenBK}})$ in its public parameters so an evaluator can derive a bootstrapping key $\mathsf{bk}_{\mathsf{id}}$ for a given identity id and then invoke $\mathsf{Bootstrap}$.

Theorem D.1.1. *Assuming indistinguishability obfuscation, one-way functions, $\hat{\mathcal{E}}_1$ is* IND-sID-CPA *secure if $\mathcal{E}$ satisfies Property 1 - Property 3.*

Proof. We prove the theorem via a hybrid argument.

Game 0: This is the real system.

Game 1: This is the same as Game 0 except for the following changes. Suppose the adversary chooses id^* as the identity to attack. Compute $r_1 \| r_2 \leftarrow \mathsf{PRF.Eval}(K, \mathsf{id}^*)$ and compute $\mathsf{bk}_{\mathsf{id}^*} \leftarrow \mathsf{WGenBootstrapKey}(\mathsf{PP}_{\mathcal{E}}, \mathsf{sk}_{\mathsf{id}^*}; r_2)$ where $\mathsf{sk}_{\mathsf{id}^*} \leftarrow \mathsf{KeyGen}(T, \mathsf{id}^*; r_1)$. Make the following changes to F_{GenBK}, which we call F'_{GenBK}, and set $\beta \leftarrow i\mathcal{O}(F'_{\mathsf{GenBK}})$

1. if $\mathsf{id} = \mathsf{id}^*$, then output $\mathsf{bk}_{\mathsf{id}^*}$.
2. Else: Run Step 1 - 3 of F_{GenBK}.

Observe that F_{GenBK} is identical to F'_{GenBK} since $\mathsf{bk}_{\mathsf{id}^*}$ is computed above in the same manner as F_{GenBK}. The games are indistinguishable due to the security of indistinguishability obfuscation.

Game 2 This is the same as Game 1 except with the following changes. Compute a punctured PRF key $K' \leftarrow \mathsf{PRF.Puncture}(K, \mathsf{id}^*)$ that is defined for all strings except the input string id^*, where id^* is the "target" identity chosen by the adversary. Replace all occurrences of K in F'_{GenBK} with K'. We call the modified function F''_{GenBK}.

Observe that $F'_{\mathsf{GenBK}} = F''_{\mathsf{GenBK}}$ because $\mathsf{PRF.Eval}(K, \mathsf{id}) = \mathsf{PRF.Eval}(k', \mathsf{id})$ for all $\mathsf{id} \neq \mathsf{id}^*$. Therefore, the games are indistinguishable due to the security of indistinguishability obfuscation.

Game 3: This is the same as Game 2 except that we change how $\mathsf{bk}_{\mathsf{id}^*}$ is computed. We do this indirectly by changing how $r_1 \| r_2 \leftarrow \mathsf{PRF.Eval}(K, \mathsf{id}^*)$ is computed instead. More precisely, we choose a uniformly random string $r'_1 \| r'_2 \xleftarrow{\$} \{0,1\}^m$ where m is the length of the pseudorandom outputs of $\mathsf{PRF.Eval}$ i.e. $m = |\mathsf{PRF.Eval}(K, \mathsf{id}^*)|$.

By the security of the puncturable PRF, we have that

$$\{(K', \mathsf{id}^*, \mathsf{PRF.Eval}(K, \mathsf{id}^*)\} \underset{C}{\approx} \{(K', \mathsf{id}^*, r) : r \xleftarrow{\$} \{0,1\}^m)\}.$$

It follows that Game 2 and Game 3 are computationally indistinguishable.

Game 4: This is the same as Game 3 except that we make the following changes. We compute a punctured trapdoor $T' \subset T$ using the $\mathsf{TrapPuncture}$ algorithm (which exists by Property 2) i.e. $T' \leftarrow \mathsf{TrapPuncture}(T, \mathsf{id}^*)$. We answer secret key queries with $\mathsf{SimKeyGen}(T', \cdot)$. The games cannot be distinguished by an adversary as a result of Equation D.1.3 in Definition D.1.4 (single-point trapdoor puncturability).

Game 5: The only change in this game is that we set $\beta \leftarrow i\mathcal{O}(F'''_{\mathsf{GenBK}})$ where F'''_{GenBK} is the same as F''_{GenBK} except $\mathsf{sk}_{\mathsf{id}}$ is computed as

$$\mathsf{sk}_{\mathsf{id}} \leftarrow \mathsf{SimKeyGen}(T', \mathsf{id}; r_1).$$

As a result of Equation D.1.3 in Definition D.1.4 (single-point trapdoor puncturability), we have that $F'''_{\mathsf{GenBK}} = F''_{\mathsf{GenBK}}$ and hence their obfuscations are indistinguishable to a PPT adversary by the security of indistinguishability obfuscation.

Game 6: Note that Game 5 removes all references to T. In this game, we produce the challenge ciphertext given to the adversary as an encryption of a uniformly random message $m' \xleftarrow{\$} \mathcal{M}$. The adversary has a zero advantage in this game.

An efficient distinguisher $\mathcal{D}$ that can distinguish between Game 5 and Game 6 can be used to violate Property 3. Let b be the challenger's random bit. Let m_0 and m_1 be the messages chosen by the adversary. Given a challenge instance of Property 3 of the form $(\mathsf{PP}, T', \mathsf{bk}_{\mathsf{id}^*}, c^*)$ where id^* is the adversary's target identity, and c^* is an encryption of either m_b or a uniformly random element in $\mathcal{M}$. Note that PP, T' and $\mathsf{bk}_{\mathsf{id}^*}$ are distributed identically to both Game 5 and Game 6. Hence, we can construct an algorithm to perfectly simulate $\mathcal{D}$'s view, and give c^* to $\mathcal{D}$ as the challenge ciphertext. If c^* encrypts m_b, Game 5 is perfectly simulated; otherwise if c^* encrypts a random message, Game 6 is perfectly simulated. It follows that a non-negligible advantage distinguishing between Game 5 and Game 6 implies a non-negligible advantage distinguishing the LHS and RHS distributions of Property 3. □

Note that the construction $\hat{\mathcal{E}}_*$ from Section 7.2 satisfies Property 1 - Property 3.

We now discuss the failure of single-point trapdoor-punturability in current LWE-based leveled IBFHE schemes.

D.1.4.2 Failure of single-point trapdoor-puncturability in LWE-based IBFHE schemes

All concrete trapdoor-puncturable weakly-bootstrappable IBFHE constructions are based on LWE[†]. More precisely, they are transformations via the GSW compiler [98] of LWE-based IBEs built from the preimage sampleable functions from [97] and the basis extension technique introduced in [56]. These *underlying* IBEs include the Binary Tree Encoding HIBE from [56] and the IBE from [109], and exclude the schemes from [7,8,97] since these schemes are not trapdoor-puncturable in the sense captured by Definition D.1.4.

We can classify all LWE-based schemes we are aware of that satisfy Definition D.1.4 in the following way, Consider an identity space $\mathcal{I} = \{0,1\}^\ell$ for some fixed integer ℓ. The public parameters in these schemes include matrices $\mathbf{A}_{\mathbf{1},\mathbf{0}}, \mathbf{A}_{\mathbf{1},\mathbf{1}}, \ldots, \mathbf{A}_{\ell,\mathbf{0}}, \mathbf{A}_{\ell,\mathbf{1}}$. An encryption of a message under an identity $\mathsf{id} = \mathsf{id}_1 \ldots \mathsf{id}_\ell \in \{0,1\}^\ell$ is performed as a dual-Regev [97, 161] encryption with the matrix $\mathbf{A}_{\mathsf{id}} = \mathbf{A}_{\mathbf{1},\mathsf{id}_\mathbf{1}} \parallel \cdots \parallel \mathbf{A}_{\mathbf{1},\mathsf{id}_\ell}$. Consider the function $f_{\mathbf{A}_{\mathsf{id}}}(\vec{x}) = \mathbf{A}_{\mathsf{id}}\vec{x} \mod q$, and let $\vec{u}$ be a public vector. It is a hard problem to find a "short" preimage of $\vec{u}$ under $f_{\mathbf{A}_{\mathsf{id}}}$. Such a preimage $\vec{e}$ is a secret key for identity $\mathsf{id} \in \{0,1\}^\ell$. In the real system, the matrices $\mathbf{A}_{\mathbf{i},\mathbf{b}}$ for $i \in [\ell], b \in \{0,1\}$ are generated together with trapdoors $\mathbf{T}_{\mathbf{i},\mathbf{b}}$ using a trapdoor generation algorithm such as that from [142]. So we have $T = \{\mathbf{T}_{\mathbf{i},\mathbf{b}}\}_{i\in[\ell],b\in\{0,1\}}$. Owing to basis extension techniques, a "short" preimage in $f^{-1}_{\mathbf{A}_{\mathsf{id}}}(\vec{u})$ can be sampled given only a single $\mathbf{T}_{\mathbf{j},\mathsf{id}_\mathbf{j}}$ for some $j \in [\ell]$. It follows that for a target identity id^*, the punctured trapdoor is $T' = \{\mathbf{T}_{\mathbf{i},\mathbf{1}-\mathsf{id}^*_\mathbf{i}}\}_{i\in[\ell]} \subset T$. However, although any trapdoor can be used to sample statistically close "short" preimages, we are not aware of any method for these trapdoors to find the same preimage in polynomial time, even when the same randomness is used. As such, there is no known efficient

[†]With the exception of the scheme from Section 7.2 based on punctured programming.

simulator $\mathsf{SimKeyGen}$ that can satisfy Equation D.1.3.

D.2 Alternative Approach: Using an obfuscated program for bootstrapping

Consider the following program $F_{\mathsf{Bootstrap}}$ that performs the bootstrapping operation:

Program $F_{\mathsf{Bootstrap}}(\mathsf{id}, c)$:

1. Compute $r_1 \parallel r_2 \leftarrow \mathsf{PRF.Eval}(K, \mathsf{id})$.
2. Compute $r_3 \leftarrow \mathsf{PRF.Eval}(K, \mathsf{id} \parallel c)$.
3. Compute $\mathsf{sk}_{\mathsf{id}} \leftarrow \mathsf{KeyGen}(T, \mathsf{id}; r_1)$.
4. Compute $\mathsf{bk}_{\mathsf{id}} \leftarrow \mathsf{WGenBootstrapKey}(\mathsf{PP}_{\mathcal{E}}, \mathsf{sk}_{\mathsf{id}}; r_2)$.
5. **Output** $\mathsf{Bootstrap}(\mathsf{PP}_{\mathcal{E}}, \mathsf{bk}_{\mathsf{id}}, c; r_3)$.

We define another scheme $\hat{\mathcal{E}}_2$ that is defined in the same way as $\hat{\mathcal{E}}_1$ with the following changes:

1. An obfuscation $\beta \leftarrow i\mathcal{O}(F_{\mathsf{Bootstrap}})$ is generated in the setup algorithm and included in the public parameters.
2. The bootstrapping algorithm $\hat{\mathcal{E}}_2.\mathsf{Bootstrap}$, on input identity id and ciphertext c, simply becomes equivalent to computing $\beta(\mathsf{id}, c)$.

Once again the proof strategy proceeds in the same manner as the previous approach. When we move from T to T', it becomes necessary to ensure that on input an identity $\mathsf{id} \neq \mathsf{id}^*$ to $F_{\mathsf{Bootstrap}}$, performing bootstrapping with a bootstrapping key based on a *different* underlying secret key (i.e. one generated with T' instead of T) can produce an identical ciphertext to the ciphertext outputted by the original $F_{\mathsf{Bootstrap}}$ above. More

precisely, what is needed here is an algorithm $\mathsf{SimBootstrap}$ such that for any $\mathsf{id} \in \mathcal{I} \setminus \{\mathsf{id}^*\}$, $\mathsf{sk}_{\mathsf{id}} \leftarrow \mathsf{KeyGen}(T, \mathsf{id}; r_1)$, $\mathsf{bk}_{\mathsf{id}} \leftarrow \mathsf{WGenBootstrapKey}(\mathsf{PP}_{\mathcal{E}}, \mathsf{sk}_{\mathsf{id}}; r_2)$, randomness r_1, r_2, r_3, and ciphertext $c \in \mathcal{C}$, it holds that

$$\mathsf{Bootstrap}(\mathsf{PP}_{\mathcal{E}}, \mathsf{bk}_{\mathsf{id}}, c; r_3) = \mathsf{SimBootstrap}(\mathsf{PP}_{\mathcal{E}}, T', \mathsf{id}, c, r_1, r_2, r_3). \tag{D.2.1}$$

The barrier to realizing such an algorithm $\mathsf{SimBootstrap}$ for presently-known IBFHEs hinges on the fact that a ciphertext c^* obtained from a homomorphic evaluation of a circuit C is dependent on the plaintext inputs to C, even leaving aside the output of C that is encrypted by c^*. Thus, homomorphically evaluating the decryption circuit, as in bootstrapping, with encryptions of two different secret keys (i.e. two different bootstrapping keys) results in two different resultant ciphertexts. However, it is non-trivial to "wipe" from c^* the unique trace left by a secret key $\mathsf{sk}_{\mathsf{id}}$ without access to $\mathsf{sk}_{\mathsf{id}}$.

Note that if $\mathcal{E}$ is *single-point* trapdoor-puncturable (i.e. it satisfies Equation D.1.3), then it is easy to construct an algorithm $\mathsf{SimBootstrap}$ that satisfies D.2.1. Such a $\mathsf{SimBootstrap}$ would use $\mathsf{SimKeyGen}$ with T' and r_1 to generate $\mathsf{sk}_{\mathsf{id}}$, then $\mathsf{sk}_{\mathsf{id}}$ and r_2 to generate $\mathsf{bk}_{\mathsf{id}}$. Hence, single-point trapdoor-puncturability implies security of both approaches. However due to the fact that $F_{\mathsf{Bootstrap}}$ subsumes F_{GenBK}, the additional complexity of $F_{\mathsf{Bootstrap}}$ is extraneous for a single-point trapdoor-puncturable $\mathcal{E}$, since the process of bootstrapping itself can be more efficiently performed directly by the evaluator. As a result, Approach 1 is preferable for a single-point trapdoor-puncturable scheme.

Appendix E

Multi-Encryptor Setting

In the multi-encryptor setting, there is a bound N placed on the number of independent senders. Each sender may contribute an input of unbounded size for evaluation. The size of an evaluated ciphertext is allowed to depend on $n \leq N$, i.e the number of independent senders whose inputs were used in the evaluation. The syntax of Multi-Encryptor Attribute Based Homomorphic Encryption (ME-ABHE) includes an algorithm GenKey that a sender uses to generate a public key pk. Then she uses pk in the encryption algorithm to encrypt all her input bits. To avoid confusion with the key extraction algorithm KeyGen, we re-name it to $\mathsf{Extract}$ in this section. A formal definition of ME-ABHE follows.

Definition E.0.1. *A (Key-Policy) Multi-Encryptor Attribute-Based Homomorphic Encryption (ME-ABHE) scheme $\mathcal{E}^{(N,\mathcal{K})}$ for an integer $N > 0$ and an integer $\mathcal{K} \in [N]$ is defined with respect to a message space $\mathcal{M}$, an attribute space $\mathbb{A}$, a class of access policies $\mathbb{F} \subseteq \mathbb{A} \to \{0,1\}$, and a class of circuits $\mathbb{C} \subseteq \mathcal{M}^* \to \mathcal{M}$. An ME-ABHE scheme is a tuple of PPT algorithms* $(\mathsf{Setup}, \mathsf{Extract}, \mathsf{GenKey}, \mathsf{Encrypt}, \mathsf{Decrypt}, \mathsf{Eval})$ *where* Setup *and* $\mathsf{Extract}$ *(aka* KeyGen*) are defined equivalently to KP-ABE. We denote by $\mathcal{C}$ the ciphertext space. The other algorithms* GenKey, $\mathsf{Encrypt}$, $\mathsf{Decrypt}$ *and* Eval *are defined as follows:*

- $\mathsf{GenKey}(\mathsf{PP}, a)$*: On input public parameters* PP *and an attribute $a \in \mathbb{A}$, generate*

and output a public key pk_a *for the attribute* a*.*

- $\mathsf{Encrypt}(\mathsf{PP}, \mathsf{pk}_a, \mu)$*: On input public parameters* PP*, a public key* pk_a *for attribute* a*, and a message* $\mu \in \mathcal{M}$*, output an encryption* c *of* μ *under attribute* a*.*

- $\mathsf{Decrypt}(\langle \mathsf{sk}_{f_1}, \ldots, \mathsf{sk}_{f_k} \rangle, c)$*: On input a sequence of* $k \leq \mathcal{K}$ *secret keys for policies* $f_1, \ldots, f_k \in \mathbb{F}$ *and a ciphertext* c*, output a plaintext* $\mu' \in \mathcal{M}$ *iff every attribute associated with* c *is satisfied by at least one of the* f_i*; output* $\perp$ *otherwise.*

- $\mathsf{Eval}(\mathsf{PP}, C, c_1, \ldots, c_\ell)$*: On input public parameters* PP*, a circuit* $C \in \mathbb{C}$ *and ciphertexts* $c_1, \ldots, c_\ell \in \mathcal{C}$*, output an* evaluated ciphertext $c' \in \mathcal{C}$*.*

More precisely, Eval *is required to satisfy the following properties:*

- *We define the function* attr *that given a public key* pk *returns the attribute associated with* pk*. Over all choices of* $(\mathsf{PP}, \mathsf{MSK}) \leftarrow \mathsf{Setup}(1^\lambda)$*,* $C : \mathcal{M}^\ell \to \mathcal{M} \in \mathbb{C}$*,* $a_1, \ldots, a_n$ *with* $n \leq N$*,* $\mathsf{pk}_{a_j} \leftarrow \mathsf{GenKey}(\mathsf{PP}, a_j)$ *for* $j \in [n]$*,* $\mu_1, \ldots, \mu_\ell \in \mathcal{M}$*,* $\mathsf{pk}_1, \ldots, \mathsf{pk}_\ell \in \{\mathsf{pk}_{a_1}, \ldots, \mathsf{pk}_{a_n}\}$ *with* $|\{\mathsf{pk}_1, \ldots, \mathsf{pk}_\ell\}| = n$*,* $c_i \leftarrow \mathsf{Encrypt}(\mathsf{PP}, \mathsf{pk}_i, \mu_i)$ *for* $i \in [\ell]$*, and* $c' \leftarrow \mathsf{Eval}(\mathsf{PP}, C, c_1, \ldots, c_\ell)$*:*

  - ***Correctness***

    $$\mathsf{Decrypt}(\langle \mathsf{sk}_{f_1}, \ldots, \mathsf{sk}_{f_k} \rangle, c') = C(\mu_1, \ldots, \mu_\ell) \text{ iff } \forall i \in [n]\ \exists j \in [k] \quad f_j(\mathfrak{a}_i) = 1 \tag{E.0.1}$$

    where $\mathfrak{a}_i = \mathsf{attr}(\mathsf{pk}_i)$*, for any* $k \in [\mathcal{K}]$*, any* $f_1, \ldots, f_k \in \mathbb{F}$*, and any* $\mathsf{sk}_{f_j} \leftarrow \mathsf{KeyGen}(\mathsf{MSK}, f_j)$ *for* $j \in [k]$*.*

  - ***Compactness*** *There exists a fixed polynomial* $s(\cdot, \cdot)$ *for the scheme such that*

    $$|c'| \leq s(\lambda, n). \tag{E.0.2}$$

  The complexity of all algorithms may depend on N.

We present a construction of ME-ABHE. Our construction relies on multi-key fully homomorphic encryption (FHE), attribute based encryption (ABE) and a pseudorandom function (PRF).

E.1 Our Construction

E.1.1 Prerequisites

Let $\mathcal{E}_{\mathsf{MKFHE}}{}^{(N)} = (\mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Gen}, \mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Encrypt}, \mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Decrypt}, \mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Eval})$ be a multikey FHE scheme that tolerates evaluation with N independent keys. For an overview of the syntax of multikey FHE, see Chapter 5, Section 5.0.1.1. We assume that $\mathcal{E}_{\mathsf{ABE}}$ has message space $\mathcal{M}_{\mathcal{E}_{\mathsf{MKFHE}}} = \{0,1\}$. Our scheme has message space $\mathcal{M} \triangleq \mathcal{M}_{\mathcal{E}_{\mathsf{ABE}}}$.

Let $\mathcal{E}_{\mathsf{ABE}} = (\mathcal{E}_{\mathsf{ABE}}.\mathsf{Setup}, \mathcal{E}_{\mathsf{ABE}}.\mathsf{KeyGen}, \mathcal{E}_{\mathsf{ABE}}.\mathsf{Encrypt}, \mathcal{E}_{\mathsf{ABE}}.\mathsf{Decrypt}$ be an ABE scheme with message space $\mathcal{M}_{\mathcal{E}_{\mathsf{ABE}}}$, attribute space $\mathbb{A}_{\mathcal{E}_{\mathsf{ABE}}}$ and class of access policies $\mathbb{F}_{\mathcal{E}_{\mathsf{ABE}}}$. Our scheme has attribute space $\mathbb{A} \triangleq \mathbb{A}_{\mathcal{E}_{\mathsf{ABE}}}$ and class of access policies $\mathbb{F} \triangleq \mathbb{F}_{\mathcal{E}_{\mathsf{ABE}}}$.

Let $\mathcal{E}_{\mathsf{PRF}} = (\mathcal{E}_{\mathsf{PRF}}.\mathsf{Key}, \mathcal{E}_{\mathsf{PRF}}.\mathsf{Eval})$ be a pseudorandom function (PRF) with polynomials $k_{\mathcal{E}_{\mathsf{PRF}}}(\cdot)$, $m_{\mathcal{E}_{\mathsf{PRF}}}(\cdot)$ and $n_{\mathcal{E}_{\mathsf{PRF}}}(\cdot)$ such that $\mathcal{E}_{\mathsf{PRF}}.\mathsf{Key}$, on input a security parameter λ, outputs a key in $\{0,1\}^{k_{\mathcal{E}_{\mathsf{PRF}}}(\lambda)}$, and $\mathcal{E}_{\mathsf{PRF}}.\mathsf{Eval}$, on input a key in $\{0,1\}^{k_{\mathcal{E}_{\mathsf{PRF}}}(\lambda)}$ and an input string in $\{0,1\}^{m_{\mathcal{E}_{\mathsf{PRF}}}(\lambda)}$, outputs a string in $\{0,1\}^{n_{\mathcal{E}_{\mathsf{PRF}}}(\lambda)}$. When the security parameter λ is understood, we abbreviate the above parameters as $k_{\mathcal{E}_{\mathsf{PRF}}}$, $m_{\mathcal{E}_{\mathsf{PRF}}}$ and $n_{\mathcal{E}_{\mathsf{PRF}}}$ respectively.

Without loss of generality, we assume that $\mathcal{M}_{\mathcal{E}_{\mathsf{ABE}}}$ is large enough to contain a PRF key $K \in \{0,1\}^{k_{\mathcal{E}_{\mathsf{PRF}}}}$.

We now present our construction, which we call $\mathsf{meABFHE}$.

E.1.2 Setup

On input a security parameter λ and a maximum number of independent encryptors N, the following steps are performed:

1. Generate $(\mathsf{PP}_{\mathcal{E}_{\mathsf{ABE}}}, \mathsf{MSK}_{\mathcal{E}_{\mathsf{ABE}}}) \leftarrow \mathcal{E}_{\mathsf{ABE}}.\mathsf{Setup}(1^\lambda)$.

2. Output $(\mathsf{PP} := (\mathsf{PP}_{\mathcal{E}_{\mathsf{ABE}}}, \lambda, N), \mathsf{MSK} := (\mathsf{PP}, \mathsf{MSK}_{\mathcal{E}_{\mathsf{ABE}}}))$.

E.1.3 Secret Key Extraction (Extract)

The extraction algorithm Extract is defined as follows. Given the master secret key $\mathsf{MSK} := (\mathsf{PP}, \mathsf{MSK}_{\mathcal{E}_{\mathsf{ABE}}})$ and a policy $f \in \mathbb{F}$, a secret key sk_f for f is generated as $\mathsf{sk}_f \leftarrow \mathcal{E}_{\mathsf{ABE}}.\mathsf{KeyGen}(\mathsf{MSK}_{\mathcal{E}_{\mathsf{ABE}}}, f)$. The secret key $\mathsf{SK}_f := (\mathsf{PP}, \mathsf{sk}_f)$ is issued to the user.

E.1.4 Key Generation (GenKey)

On input public parameters $\mathsf{PP} := (\mathsf{PP}_{\mathcal{E}_{\mathsf{ABE}}}, \lambda, N)$ and an attribute $a \in \mathbb{A}$, run the following steps. Generate a PRF key $K \leftarrow \mathcal{E}_{\mathsf{PRF}}.\mathsf{Key}(1^\lambda)$. Encrypt K with the ABE scheme $\mathcal{E}_{\mathsf{ABE}}$ using attribute $a \in \mathbb{A}$; that is, compute $\psi \leftarrow \mathcal{E}_{\mathsf{ABE}}.\mathsf{Encrypt}(\mathsf{PP}_{\mathcal{E}_{\mathsf{ABE}}}, a, K)$. Consider the following subroutine DeriveKey:

- $\mathsf{DeriveKey}(\lambda, K)$: Let t be the number of random bits used by $\mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Gen}$. Compute $v \leftarrow \lceil t/n_{\mathcal{E}_{\mathsf{PRF}}} \rceil$ and generate randomness $r \leftarrow \mathcal{E}_{\mathsf{PRF}}.\mathsf{Eval}(K, 1) \parallel \cdots \parallel \mathcal{E}_{\mathsf{PRF}}.\mathsf{Eval}(K, v) \in \{0,1\}^{v \cdot n_{\mathcal{E}_{\mathsf{PRF}}}}$. Note that $t \leq v \cdot n_{\mathcal{E}_{\mathsf{PRF}}}$ and thus r is sufficient randomness for $\mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Gen}$. Output $(\mathsf{pk}_{\mathcal{E}_{\mathsf{MKFHE}}}, \mathsf{sk}_{\mathcal{E}_{\mathsf{MKFHE}}}, \mathsf{vk}_{\mathcal{E}_{\mathsf{MKFHE}}}) \leftarrow \mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Gen}(1^\lambda; r)$.

Run $(\mathsf{pk}_{\mathcal{E}_{\mathsf{MKFHE}}}, \mathsf{sk}_{\mathcal{E}_{\mathsf{MKFHE}}}, \mathsf{vk}_{\mathcal{E}_{\mathsf{MKFHE}}}) \leftarrow \mathsf{DeriveKey}(\lambda, K)$. Output $\mathsf{pk} \leftarrow (\psi, \mathsf{pk}_{\mathcal{E}_{\mathsf{MKFHE}}}, \mathsf{vk}_{\mathcal{E}_{\mathsf{MKFHE}}})$.

E.1.4.1 Encryption

On input public parameters $\mathsf{PP} := (\mathsf{PP}_{\mathcal{E}_{\mathsf{ABE}}}, \lambda, N)$, a public key pk and a message $\mu \in \mathcal{M}$, run the following steps. Parse pk as $(\psi, \mathsf{pk}_{\mathcal{E}_{\mathsf{MKFHE}}}, \mathsf{vk}_{\mathcal{E}_{\mathsf{MKFHE}}})$. Encrypt μ with $\mathcal{E}_{\mathsf{MKFHE}}$; that is, compute $c \leftarrow \mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Encrypt}(\mathsf{pk}_{\mathcal{E}_{\mathsf{MKFHE}}}, \mu)$. Output $\mathsf{CT} := (\langle(\psi, \mathsf{vk}_{\mathcal{E}_{\mathsf{MKFHE}}})\rangle, c)$.

E.1.4.2 Evaluation

On input public parameters $\mathsf{PP} := (\mathsf{PP}_{\mathcal{E}_{\mathsf{ABE}}}, \lambda, N)$, a circuit $C \in \mathbb{C}$, and ciphertexts $\mathsf{CT}_1, \ldots, \mathsf{CT}_\ell$, the evaluator performs the following steps. Firstly, the ciphertexts are as-

sumed to be "fresh" ciphertexts generated with the encryption algorithm. In other words, the evaluator can parse CT_i as $(\langle(\psi_i, \mathsf{vk}_i)\rangle, c_i)$ for every $i \in [\ell]$. Then we compute the number of distinct senders $n \leftarrow |\{(\psi_1, \mathsf{vk}_1), \ldots, (\psi_\ell, \mathsf{vk}_\ell)\}| \leq N$ (i.e. the number of distinct public keys used to generate the collection of ciphertexts). This is because each public key is associated with a unique pair (ψ_i, vk_i). For ease of exposition, we label these n unique pairs as $(\hat{\psi}_1, \hat{\mathsf{vk}}_1), \ldots, (\hat{\psi}_n, \hat{\mathsf{vk}}_n)$. Compute $c' \leftarrow \mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Eval}(C, (c_1, \mathsf{vk}_1), \ldots, (c_\ell, \mathsf{vk}_\ell))$ and output $\mathsf{CT}' := (\langle(\hat{\psi}_1, \hat{\mathsf{vk}}_1), \ldots, (\hat{\psi}_n, \hat{\mathsf{vk}}_n)\rangle, c')$. If the scheme is only required to be 1-hop homomorphic. then the evaluation keys can be removed from CT'. This is assumed to be the case for a *final* evaluated ciphertext, which is what is considered when we measure the size of an evaluated ciphertext.

E.1.4.3 Decryption

To decrypt a ciphertext CT with a sequence of secret keys $\langle \mathsf{SK}_{f_1} := (\mathsf{PP}, \mathsf{sk}_{f_1}), \ldots, \mathsf{SK}_{f_k} := (\mathsf{PP}, \mathsf{sk}_{f_k})\rangle$ for respective policies $f_1, \ldots, f_k \in \mathbb{F}$, a decryptor performs the following steps.

Parse CT as $(\langle(\psi_1, \mathsf{vk}_1) \ldots, (\psi_n, \mathsf{vk}_n)\rangle, c')$. Output $\perp$ and abort if $n > N$. For every $i \in [n]$, let a_i be the attribute associated with ψ_i. For every $i \in [n]$, assert there exists an $f \in \{f_1, \ldots, f_k\}$ with $f(a_i) = 1$; choose an arbitrary such f and label it with $\mathfrak{f}_i$. Otherwise output $\perp$ ans abort. Compute $K_i \leftarrow \mathcal{E}_{\mathsf{ABE}}.\mathsf{Decrypt}(\mathsf{sk}_{\mathfrak{f}_i}, \psi_i)$. Compute $(\mathsf{pk}^{(i)}_{\mathcal{E}_{\mathsf{MKFHE}}}, \mathsf{sk}^{(i)}_{\mathcal{E}_{\mathsf{MKFHE}}}, \mathsf{vk}^{(i)}_{\mathcal{E}_{\mathsf{MKFHE}}}) \leftarrow \mathsf{DeriveKey}(\lambda, K_i)$.

Output $\mu' \leftarrow \mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Decrypt}(\mathsf{sk}^{(1)}_{\mathcal{E}_{\mathsf{MKFHE}}}, \ldots, \mathsf{sk}^{(n)}_{\mathcal{E}_{\mathsf{MKFHE}}}, c')$.

E.2 Parameters

In practice, each sender, whose input is γ bits, need only send the structure $(\psi, \mathsf{vk}, c_1, \ldots, c_\gamma)$, where ψ is an $\mathcal{E}_{\mathsf{ABE}}$ ciphertext, vk is an $\mathcal{E}_{\mathsf{MKFHE}}$ evaluation key and each c_i for $i \in [\gamma]$ is a $\mathcal{E}_{\mathsf{MKFHE}}$ ciphertext. Of course γ can be of any size. We could avoid the usage of a PRF in the key generation algorithm by computing ψ as an ABE encryption of a secret key for $\mathcal{E}_{\mathsf{MKFHE}}$, as opposed to an encryption of a PRF key. However, as we will see, the size

of a secret key for $\mathcal{E}_{\mathsf{MKFHE}}$ is large compared to that of a PRF key. Furthermore, the size of the latter is constant for a given security level λ, whereas the size of the former grows with the desired bound N on the number of senders.

Let $R = ZZ[x]/\phi(x)$ be a polynomial ring with $\phi(x) = x^d + 1$ and d a power of 2. When multiplying any two elements $a(x), b(x) \in R$, the (ℓ_∞) norm of their product grows by at most a factor δ, called the "expansion factor". For the special case of $\phi(x) = x^d + 1$ where d is a power of 2, we have $\delta = n$ [44]. Using the formula in Section 3.3 of [131] (Equation 2 in that paper), we can obtain the maximum value $\mathsf{max}_{\log_2 q}$ of $\log_2 q$ to ensure 80 bits of security; we calculate this as $\mathsf{max}_{\log_2 q} \approx 48000$.

To support a circuit depth of $L = 50$ and a maximum number of senders $N = 60$, we need to set the dimension parameter to $d = 2^{20}$. The theoretical noise limit for this parameter is above the maximum allowed q. Fortunately, Lepoint and Naehrig [131] report that it in practice, the noise grows more slowly than the theoretical limit. In their case, the found that one could reduce q by 33% and still achieve correctness with high probability. Taking this account, we can set q accordingly. One possible setting that satisfies all our criteria is $q = 46691$. With these parameters, a single ciphertext for the multikey FHE scheme is approximately 5.7 GB. While this is extremely large, we can avail of hybrid homomorphic encryption i.e. encrypt the plaintexts using a symmetric cipher and encrypt the symmetric key with the multikey FHE scheme. In hybrid homomorphic encryption, the evaluator first homomorphically evaluates the decryption circuit of the symmetric cipher prior to carrying out the desired homomorphic computation. For example, Lepoint and Naehrig evaluate minimalist block cipher SIMON - the variant with 32 rounds requires 32 levels. Our parameters handle this, leaving 50 - 32 = 18 levels for further evaluation.

E.3 Implementation

We extended the implementation of Lepoint and Naehrig [131] to support multiple keys; in effect, this is an implementation of the multikey FHE scheme of López-Alt, Tromer and Vaikuntanathan [135]. The implementation uses the library FLINT [80] for arithmetic.

To give the reader an understanding of the present state of affairs for multi-key homomorphic encryption, we chose to evaluate a useful circuit. As part of our implementation, we developed a basic "compiler" for a simple functional language we called "Simple Circuit Description Language" (SCDL). SCDL is a simple language to describe an arithmetic circuit over some ring. It has two basic operations, addition (+) and multiplication (*). The language can be interpreted in any ring. For example, when interpreted in the Boolean field, "+" corresponds to XOR and "*" corresponds to AND. To illustrate the syntax, below is the definition of the function "equals" that tests the equality of its two arguments:

```
constant one = 1
func equal(x, y) = (x + y) + one
```

The code above first defines the constant "one". The next line defines the function "equals" as the XOR of its two arguments XORed with "one". Similarly we can define the "or" function:

```
func or(x, y) = x + y + (x*y)
```

Now we define a function that determines whether an 8-bit value is greater than another 8-bit value:

```
func gt(X : 8, Y : 8) = or(X[7]*not(Y[7]),\
    or(X[7]*not(Y[7]),\
    or(equal(X[7], Y[7])*X[6]*not(Y[6]),\
    or(equal(X[7], Y[7])*equal(X[6], Y[6])*X[5]*not(Y[5]),\
    or(equal(X[7], Y[7])*equal(X[6], Y[6])*equal(X[5], Y[5])*\
```

```
X[4]*not(Y[4]),\
or(equal(X[7], Y[7])*equal(X[6], Y[6])*equal(X[5], Y[5])*\
equal(X[4], Y[4])*X[3]*not(Y[3]),\
or(equal(X[7], Y[7])*equal(X[6], Y[6])*equal(X[5], Y[5])*\
equal(X[4], Y[4])*equal(X[3], Y[3])*X[2]*not(Y[2]),\
or(equal(X[7], Y[7])*equal(X[6], Y[6])*equal(X[5], Y[5])*\
equal(X[4], Y[4])*equal(X[3], Y[3])*equal(X[2], Y[2])*\
X[1]*not(Y[1]),\
  equal(X[7], Y[7])*equal(X[6], Y[6])*equal(X[5], Y[5])*\
  equal(X[4], Y[4])*equal(X[3], Y[3])*equal(X[2], Y[2])*\
  equal(X[1], Y[1])*X[0]*not(Y[0]))))))))))
```

We compiled the above function into a circuit representation via our "compiler" and we homomorphically evaluated the circuit using our implementation of multi-key FHE. Note that the multiplicative depth of this circuit is 3. The parameters we chose were as follows: $d = 512$, $\log_2 q = 570$. Furthermore, the standard deviation of the noise distribution was set to 8. The private keys were randomly sampled from $\{-1, 0, +1\}^d$. Empirically we determined that a maximum of 4 independent keys could be tolerated when evaluating the above circuit.

The code was compiled with optimization flag '-03' along with OpenMP using g++ version 4.7.2. The experiments were executed on a laptop with 4 GB of RAM and an Intel Core i5-3340M CPU clocked at 2.70 GHz. In each experiment, a number was keys was chosen to be used in the range 1 to 4. In other words, in the k-th experiment for $k \in [4]$, k keys were used. Each input plaintext was assigned to one of the k keys. This was done in a round-robin fashion, where adjacent inputs were assigned to the next key in sequence. Each input plaintext was then encrypted with the key it was assigned to. This spreads the inputs among the keys. Each experiment involved evaluating the above circuit (i.e. the greater-than circuit) with the ciphertexts generated as described. We ran

Table E.1: Run times and noise levels ($\log_2$) for evaluation of the 8-bit greater-than circuit with different keys.

Number of keys	Run time - Mean (s)	Noise level ($\log_2$)
1	129.08	274
2	207.85	380.2
3	285.01	560.9
4	354.06	566.5

each experiment 10 times and obtained the mean run time for the evaluation along with the mean noise level in the resulting ciphertext. More precisely, we take the log of the noise level, which with our parameters takes on a value between 0 and $\log_2 q - 1 = 569$ bits. As we can see from Table E.1, 4 keys is the most we can tolerate since the noise level is almost at the threshold, which is $\log_2 q - 1 = 569$. The table also tells us that the average run time for 4 keys is ≈ 2.74 times that for one key, which shows the overhead of additional keys. It must be noted that assigning the inputs to different keys in a round-robin manner (as we have done) results in the worst performance because the gates at every level involve multiple keys and are thus more costly to evaluate. In practice, one might expect inputs from different keys to be combined with each other at a later stage in the circuit, which would lead to better performance.

The implementation we extended of Lepoint and Naehrig [131] uses the library FLINT [80] for arithmetic, which exploits parallelization using OpenMP. To parallelize further, one could distribute work to different worker nodes.