New Security Definitions, Constructions and Applications of Proxy Re-Encryption

por

David Alejandro Núñez Montañez

Memoria presentada para optar al título de
Doctor por la Universidad de Málaga

Directores:

Isaac Agudo Ruiz

Profesor Contratado Doctor

Javier López Muñoz

Catedrático de Universidad

Enero de 2016

D. Isaac Agudo Ruiz, Profesor Contratado Doctor del área de Ingeniería Telemática del Departamento de Lenguajes y Ciencias de la Computación de la Universidad de Málaga, y D. Fco. Javier López Muñoz, Catedrático de Universidad del área de Ingeniería Telemática del Departamento de Lenguajes y Ciencias de la Computación de la Universidad de Málaga,

CERTIFICAN QUE:

D. David Alejandro Núñez Montañez, Ingeniero en Informática, ha realizado en el Departamento de Lenguajes y Ciencias de la Computación de la Universidad de Málaga, bajo nuestra dirección, el trabajo de investigación correspondiente a su Tesis Doctoral, titulada:

New Security Definitions, Constructions and Applications of Proxy Re-Encryption

Revisado el presente trabajo, estimamos que puede ser presentado al tribunal que ha de juzgarlo, y autorizamos la presentación de esta Tesis Doctoral en la Universidad de Málaga.

Málaga, a 26 de octubre de 2015

Fdo: D. Isaac Agudo Ruiz
Profesor Contratado Doctor
Área de Ingeniería Telemática

Fdo: D. Fco. Javier López Muñoz
Catedrático de Universidad
Área de Ingeniería Telemática

Agradecimientos

El final de esta tesis doctoral coincide con el final de una etapa de mi vida, y al mismo tiempo, con el comienzo de una nueva. En este momento me doy cuenta de que aunque este trabajo lleve mi nombre, depende también del apoyo de muchas personas. Es por ello que tengo la necesidad de agradecer a aquellos que lo han hecho posible, y que, inevitablemente, forman ahora parte de mi vida.

En primer lugar, quiero agradecer a mis directores, Isaac y Javier, por su apoyo y guía constante para que este trabajo saliera adelante. De Isaac tengo que resaltar su capacidad asombrosa para inventar y pensar desde otras perspectivas, que me ha sacado del atolladero en muchas ocasiones. A Javier le debo una gratitud inmensa por darme la oportunidad, la libertad y la confianza de trabajar en esta tesis doctoral. Ambos son magníficos directores, no solo ya por su calidad científica, sino por su calidad humana. Les estaré siempre agradecido.

A mi familia, tanto la de siempre como la que se ha ido incorporando, y a mis amigos, les agradezco su apoyo constante. Aunque a veces estemos muy lejos, para mí siempre estáis muy cerca. De entre todos, quiero hacer una mención especial a mis padres, que me inculcaron el buscar lo que me apasione y me haga feliz. Es, sin lugar a dudas, el mejor consejo que he podido recibir.

A todos mis compañeros de NICS Labs, actuales y pasados, que han hecho que no conozca lo que es el síndrome post-vacacional. Un mensaje especial para Fran y Ana, mis compis de despacho: espero que esta tesis me sirva para subir en la jerarquía social y ganarme el mismo respeto que la impresora.

Finalmente, el agradecimiento más especial es para mis grandes amores: Lorena, desde el primer día me llenas de felicidad, y nada de esto sería posible sin tu compañía y complicidad. Daniel, aunque acabas de llegar, siento como si siempre hubieras estado conmigo. Me hacéis sentirme completo y os quiero con locura. No puedo olvidar tampoco a mi Aceituno, con su continua sonrisa de perro y sus besos a deshoras.

A todos vosotros, os doy las gracias de corazón.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The advent of generalized data outsourcing, epitomized by the cloud computing paradigm, has brought with it great concerns regarding security and privacy due to the loss of control that it implies. Traditional solutions, mainly based on the application of internal policies and standard access control strategies, simply reduce the problem to a trust issue and can be broken by service providers, either by accident or intentionally. Protecting outsourced information while at the same time reducing trust in service providers has become a demanding task. Cryptographic safeguards are a crucial mechanism towards this goal.

This thesis is devoted to the study of *proxy re-encryption* (PRE), a cryptographic primitive that constitutes a practical solution to this problem, from the perspective of both functionality and efficiency. Proxy re-encryption is a type of public-key encryption that also allows a proxy entity to transform ciphertexts from one public key to another, without learning anything about the underlying message. From a functional point of view, proxy re-encryption can be seen as a means of securely delegating access to encrypted information, representing therefore a natural candidate to construct cryptographically-enforced access control mechanisms. In addition, this primitive is in itself of great theoretical interest, since its security definitions have to simultaneously balance the security of ciphertexts and the possibility of transforming them through re-encryption, which represents a challenging dichotomy.

The contributions of this thesis follow a transversal approach, ranging from the very definitions of security models for proxy re-encryption to the specifics of applications. In the remainder of this chapter, we present the motivation, goals and contributions of this thesis, as well as the outline of the following chapters.

1.1 Motivation and Scope

It is a fact that the realization of the cloud computing paradigm has raised great expectations regarding performance, simplification of business processes, and, foremost, cost reduction [1]. At the same time, it is also true that these advancements have introduced new security and privacy risks [2]. Threat scenarios radically change when moving from resources that are fully controlled by the data owner to resources administrated by third party entities like public clouds.

Nowadays, the great majority of systems, including those deployed by cloud providers, base their security on preventing potential attackers from accessing internal servers and databases, where users' data is stored. To this end, there is a great variety of measures, with access control systems and network defense techniques being the most prominent. However, the premise of this approach is that the attackers should not be able to get inside a predetermined security perimeter, where the protected assets (in this case, users' data) reside. The harsh reality is that, although crucial, these types of measures are often not enough. In addition to external attackers, which include not only "hackers" but also nation-scale adversaries, accidental data disclosures and insider attacks are also a menacing possibility.

Countermeasures to these threats include the establishment of internal security policies and governance rules, and the reinforcement of access control strategies, but these simply reduce the situation to a trust problem. That is, in the end, there are no actual mechanisms that prevent cloud providers from breaking these measures, either by accident or intentionally. Moreover, in most cases, there is almost no risk of being discovered accessing users' information without their consent. An interesting conflict appears in this scenario – users want to go to the cloud for its benefits, but at the same time, they are unwilling to provide their data to entities that they do not necessarily trust. The adoption of cloud services has been slowed by this dichotomy from the beginning. The introduction of more advanced security mechanisms that enable users to benefit from cloud services and still ensure the confidentiality of their information could help reduce the trust assumptions in the cloud, and hence, to break the aforementioned dichotomy.

Therefore, it is necessary to depart from the traditional premise that shapes current cloud security and to assume that the measures defined above can be bypassed. A more realistic premise is to assume that the attackers have potential access to users' data [3]. Under this assumption, the only plausible solution is therefore the use of cryptography, so outsourced data is stored in encrypted form. Thus, when traditional security measures fail, attackers will only obtain encrypted data. In a way, the deployed encryption mechanisms become the ultimate safeguard of data confidentiality.

A critical principle of this solution is to design the system in such a way that even the provider itself does not have access to the corresponding decryption key; not doing this would again imply a strong trust assumption on the provider. However, a naive combination of this principle with traditional encryption primitives, both symmetric and asymmetric, can hinder the proper processing and sharing of outsourced information and negatively impact the functionality of the system. Therefore, this design decision requires the use of cryptographic primitives that transcend traditional ones, so data confidentiality can be guaranteed, but functionality still remain unaffected.

1.1.1 The Secure Data Sharing Scenario

We have previously discussed the need for weakening the trust assumptions in outsourced settings, such as the cloud. To this end, encrypting the data prior to outsourcing strikes as an essential requirement. At the same time, it is also necessary that the implemented encryption techniques allow the preservation of data sharing, which is one of the most basic functionalities that is expected in such scenarios. We refer generically to this setting as the *secure data sharing scenario*. We will later describe several examples of use cases that require secure sharing in this data outsourcing setting. Note that data sharing is one of the simplest functionalities that is expected in a setting like this. In fact, there are more advanced functionalities, such as searching, or even computing, over encrypted data; yet, the data sharing functionality is very challenging as the distribution of access rights becomes difficult once the information has been encrypted and outsourced.

Before continuing, it is necessary to understand which are the roles involved in the main interactions. In any data sharing scenario (regardless of whether there is outsourcing or not), there are three main separate roles: data producers, data owner, and data consumers. The most generic usage relation in this setting is that multiple data producers generate data which is owned by a data owner, who in turn can share it with multiple data consumers. An important aspect of this usage relation is that the data owner and data producers can be separate entities, not necessarily within the same security domain. This latter characteristic has great implications when it comes to designing secure, yet scalable, solutions, since this rules out conventional techniques, such as the use of public-key encryption alone. In addition, under our motivating scenario, we also introduce a fourth role that provides a data storage service, where information is encrypted. Figure 1.1

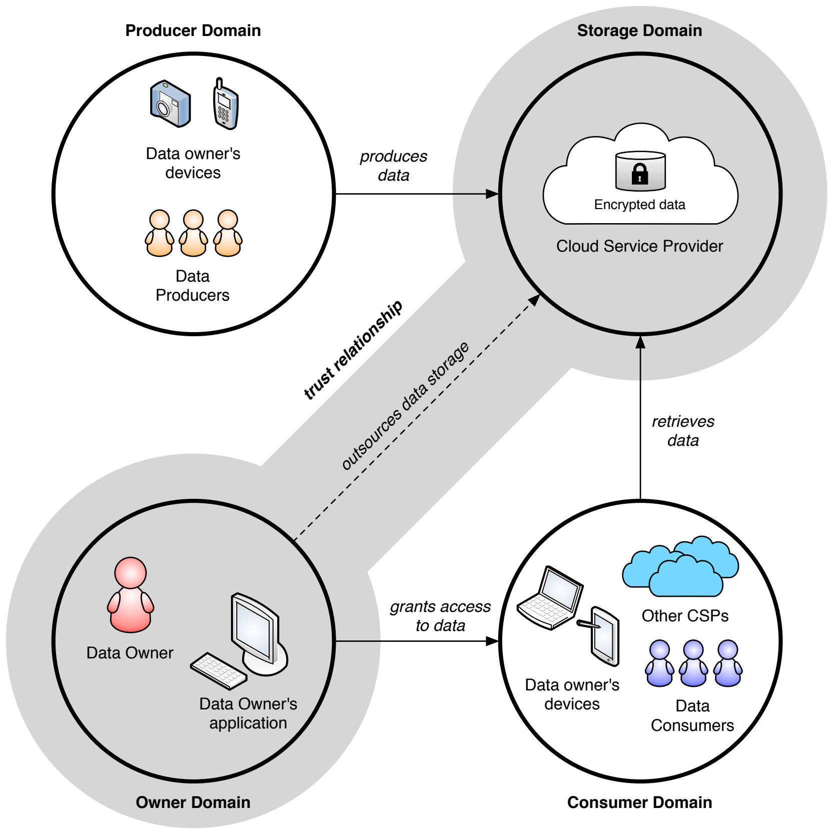


Figure 1.1: Roles and Domains in the Secure Data Sharing Scenario

depicts these roles and domains, as well as the relationships and interactions between them. The following is a more detailed description of the identified roles:

- Data Producer Domain: To this domain belong those entities that generate data. Since generation does not imply ownership, the distinction between data producer and data owner becomes necessary. However, data producers can participate in the protection of the data from the beginning, by encrypting it from the source. In addition, they can send the information directly to the storage provider, in this way decoupling this interaction from the data owner.
- Data Owner Domain: This domain is centered on the subject that owns the data that is to be securely shared. The main function of the data owner is to authorize consumers to access his data. Note that the data owner can also (and most times does) act as a data producer, but this does not rule out scenarios where separate entities participate in data production.

We assume that the data owner interacts with other actors through a user agent, usually a browser or a specific application, running on a trusted computer. This domain is assumed to be completely trusted by the data owner.

- Secure Storage Provider Domain: This domain is controlled by specialized entities that steward data owners' information and provide a sharing service, without being able to learn anything. Given that cloud computing is the most prominent instantiation of the considered scenario, we will also refer to this actor as the *cloud provider*. This domain is assumed to be semi-trusted, since we assume that the provider will provide the service correctly, but, at the same time, it may have incentives for trying to read the data. This trust assumption is explained below in more detail.
- Data Consumer Domain: This domain comprises the entities that are legitimate recipients of the information shared by the data owner, which include not only people, but also third-party services and data owner's devices. Data consumers access the shared information through the storage service provided by the cloud.
- External Domain: A fifth domain, omitted in Figure 1.1, comprises all the external actors that may have an interest in the protected information, such as hackers and nation-scale adversaries. However, none of these actors are either in charge of managing data or granted with any permission to read it. This type of adversary would have to deal with traditional security measures (e.g., physical security, firewalls, access control systems, etc.), and in a worst-case scenario, they should see nothing more than encrypted data. If the encryption scheme in use achieves an adequate security notion (e.g., indistinguishability under chosen-plaintext/ciphertext attacks), then we can assume that the data is secure.

Any information that is to be protected must be encrypted from the source (i.e., data producer domain) and decrypted by the legitimate recipients (i.e., data consumer domain). Therefore, from a visibility point of view, the goal is that the storage provider domain and external domain should only see encrypted data, in any case.

A naive solution for this scenario would be to use conventional encryption techniques (e.g., AES, RSA) and to share the decryption key with the parties designated by the data owner. Symmetric encryption cannot be used alone, since it implies that the same key is shared between producers, owner and consumers or, at least, that for each piece of encrypted data, producers and owner agree on a key, which is extremely inefficient. The typical solution would be to use some sort of hybrid encryption, where data is encrypted with a fresh random key using symmetric encryption, and the key is encrypted with public-key cryptography. The problem in our scenario is that the producers do not necessarily know in advance who are the intended consumers of the encrypted information. Therefore, the only possibility is that they encrypt the data under some common public key controlled by the data owner (e.g., the data owner's public key); this implies that the data owner has to decrypt the data and subsequently encrypt it with a key known by the intended consumers; this *decrypt-and-encrypt* solution requires the data owner to be available online to re-encrypt the data when needed, which is not always possible, apart from being extremely inefficient. The problem gets increasingly complex when one considers multiple pieces of data, and diverse producers and consumers.

It can be seen that this functionality, although simple, cannot be solved by traditional encryption techniques without resorting to complex key management procedures. In order to solve this problem, different types of cryptosystems have been proposed, *Proxy Re-Encryption* being the most prominent candidate. Proxy Re-Encryption (PRE) is a type of public-key encryption that allows a proxy entity to transform ciphertexts from one public key to another, without learning anything about the underlying data. Therefore, from a functional point of view, proxy re-encryption can be seen as a means of securely delegating access to encrypted information.

In this thesis we postulate that proxy re-encryption is a prime candidate for constructing cryptographically-enforced access control systems where the protected resource is stored externally, since it enables dynamic delegation to encrypted information. In a PRE-based solution, private data can reside in the cloud in encrypted form and be shared to authorized users by means of re-encryption, while still remaining confidential with regard to unauthorized parties and the cloud provider itself. In addition, proxy re-encryption allows the data owner to delegate the access after the data is encrypted, which is important since in a typical data sharing scenario it may not always be possible to identify beforehand the access conditions to the protected data. The use of encryption for protecting data at rest can decrease the risks associated to data disclosures in this kind of scenario, since outsourced information can only be effectively shared if access has been delegated by the data owner.

Secure data sharing is the most pressing functionality in the setting under consideration, since it can be seen as a first step towards realizing more advanced functionalities; for this reason, the scope of this thesis is set on it. Other functionalities, however, may require the aid of different cryptosystems. For example, there is a growing interest in the area of *Searchable Encryption* [4], a type of cryptosystem that permits searching over encrypted data; this makes sense when it is necessary to retrieve a specific piece of data from an encrypted dataset, although it does not tackle the issue of sharing the results. The area of Fully Homomorphic Encryption [5], which enables generic computations over encrypted information, has an even bigger appeal. This type of cryptosystem has received a great deal of attention, even from outside the research community [6] due to its dazzling potential, although as we have seen before, its current state is far from being practical. In addition, fully homomorphic encryption is a disproportionate solution for the problem of secure data sharing, since this problem does not require generic computation capabilities.

Threat model and trust assumptions

As argued, in this scenario the most powerful threats may come from the cloud provider domain, since we assume that it can bypass its internal security safeguards. Thus, we consider it as the main adversary. It is also important to remark that, in this scenario, data consumers represent a lesser danger, since they are not as empowered as storage providers: if they are not previously authorized by the data owner, then they can be considered as belonging to the external domain; conversely, if they have been granted access to the protected data, then they are legitimized to read the data, so once the information is released, it is impossible for any access control system to prevent it from being disclosed.

Based on the definitions given in [7], we identify three types of providers depending on the underlying trust assumptions:

- Trusted: The cloud provider is a fully trusted entity, which provides a service in a correct and truthful manner. This is the type of provider which is widely assumed nowadays, since users entrust their data to providers without demanding strong protection mechanisms and, hence, they suppose that the providers will always be trustworthy. However, as we have explained before, this is not a realistic model of provider for different reasons.
- Honest-but-curious: The provider follows the agreed protocol correctly, but it also stores or collects information about the users without their consent; for this reason, we may also say that the provider is *semi-trusted*. Depending on the nature of this information, there are two distinct, but not exclusive, subtypes of honest-but-curious cloud providers:
  - Data-curious: The provider has an incentive to read users' data that is in its custody. Such an incentive could be, for example, selling users' private information to third-parties for advertising or fraudulent

operations; other motivations could be industrial espionage or political repression.

  – Access-curious: The provider collects information related to the access patterns of the user, which enables it to track users' behavior and threaten their privacy.

  We additionally assume that honest-but-curious providers do not collude with authorized data consumers in order to read users' data; in fact, once a data consumer has access to the data of the user, it would be trivial to share it with the cloud provider.

- Malicious: The provider has the possibility to actively deceive the user, read the user's data and collect access patterns; it may also collude with other entities (e.g., data consumers) to do so. A provider of this type may not follow the agreed protocols, so users cannot trust that they will do so. This is the most difficult type of adversary; however, it is the most realistic, since it actually models all the capabilities of a real provider.

With this classification in mind, we will restrict the work in this thesis to *data-curious providers*, which behave correctly with respect to protocol fulfillment, but which have no hindrance to try to read users' data. We will assume therefore that the cloud provider may have some incentive to read users' data without their consent, but will not try to track users' behavior and access patterns.

Some application use cases

Once we have described the generic secure data sharing scenario, it is useful to think of plausible application use cases that can potentially benefit from the utilization of cryptographically-enforced access control systems. Some of these use cases are:

- Healthcare: These types of systems, which manage sensitive data such as patients' information, diagnostic results and medical personnel data, are increasingly being computerized and externalized. Consequently, it is of paramount importance to properly protect the information stored in these systems. This requirement stems not only from common sense, but also from a regulatory imperative: for example, in the United States, the Health Insurance Portability and Accountability Act (HIPAA) dictates that appropriate technical safeguards must be put in place in order to control access to sensitive information, including the use of encryption techniques [8]. At the same time, the materialization of widespread communication technologies has also brought the possibility of exchanging this data with legitimate parties (e.g., external hospitals, insurance companies, pharmacies, etc.) in order to improve the service offered to patients. Therefore, similar problems to the ones exposed above arise. This scenario has been seen traditionally as the quintessential use case for secure data sharing, since it clearly illustrates the assets to protect, the threats and the actors involved.

- Digital Rights Management (DRM): These systems are used to restrict the usage of copyrighted content, such as video, audio, books and software. The copyrighted content has to be shared with legitimate consumers, but at the same time, it is necessary to prevent access from any other entities. Thus, the relevance of the secure data sharing scenario to this use case is also immediate, especially when it is combined with a cloud-based delivery model, which is extremely common nowadays.
- Big Data Analytics in the Cloud: Big Data Analytics represents a new opportunity for organizations to transform the way they market services and products through the analysis of massive amounts of data. However, small and medium size companies are not often capable of acquiring and maintaining the necessary infrastructure for running Big Data Analytics on-premise, so the cloud paradigm represents a natural solution to this problem. The idea of providing Big Data Analytics as a Service, such as Google Cloud Hadoop [9] and MapR's Hadoop as a Service [10], is a very appealing solution for small organizations. This idea makes even more sense nowadays, since a lot of organizations are already operating using cloud services, and therefore, it is more sensible to perform analytics where the data is located (i.e., the cloud). The secure data sharing scenario fits in well with the outsourcing of Big Data processing to the cloud, since information can be stored in encrypted form in external servers in the cloud and processed only if access has been delegated.

It can be seen that all these use cases share a common trait: the possibility of data being disclosed or stolen is unacceptable. At the same time, they show that the sharing functionality has to be preserved, since it is necessary for creating added value services (e.g., better medical assistance, content distribution, analytics, etc.).

1.1.2 Proxy Re-Encryption

The previous sections have described the secure data sharing scenario. As pointed out, our research postulate is that proxy re-encryption is an appropriate candidate tool for supporting access delegation to encrypted information in this scenario. In this section, we give a more detailed overview of the concept of proxy re-encryption and how it fits in as part of a solution to this scenario.

The basic idea of proxy re-encryption is embodied by the ability of a proxy to transform ciphertexts under the public key of Alice into ciphertexts decryptable by Bob; to do so, the proxy must be in possession of a *re-encryption key* that enables this process. In addition, the proxy cannot learn any information about the encrypted messages, under any of the keys.

From an abstract viewpoint, a proxy re-encryption scheme can be seen as a mechanism for the *delegation of decryption rights*. For this reason, in PRE literature, the parties involved are named in terms of a relationship of delegation. A typical proxy re-encryption environment involves at least three parties:

- Delegator: This actor is the one that *delegates* his decryption rights using proxy re-encryption. We usually refer to the delegator as “Alice”.
- Delegatee: The delegatee is granted a delegated right to decrypt ciphertexts that, although were not intended for him in the first place, where re-encrypted for him with permission from the original recipient (i.e., the delegator). This actor usually takes the name “Bob”.
- Proxy: This actor is responsible for the re-encryption process that transforms ciphertexts under the delegator’s public key into ciphertexts that the delegatee can decrypt using his private key. The proxy uses the re-encryption key during this process, and does not learn any additional information.

Figure 1.2 depicts the main actors in a proxy re-encryption environment and their interactions. Since PRE is a special type of PKE, users also have a pair of public and private keys, as shown in the figure. Hence, anyone that knows a public key is capable of producing ciphertexts intended for the corresponding recipient; conversely, these ciphertexts can only be decrypted using the corresponding decryption key. The distinctive aspect is that ciphertext can be re-encrypted in order to be decryptable by a different private key than the one originally intended.

In addition to the actors shown in Figure 1.2, some schemes may involve more parties such as time servers (in the case of temporal proxy re-encryption schemes [11, 12]) and key generation centers (in the case of identity-based schemes [13, 14]). These are omitted here for the sake of generality.

Since proxy re-encryption realizes the functionality of decryption rights, we have argued that it is a prime candidate for constructing cryptographically-enforced

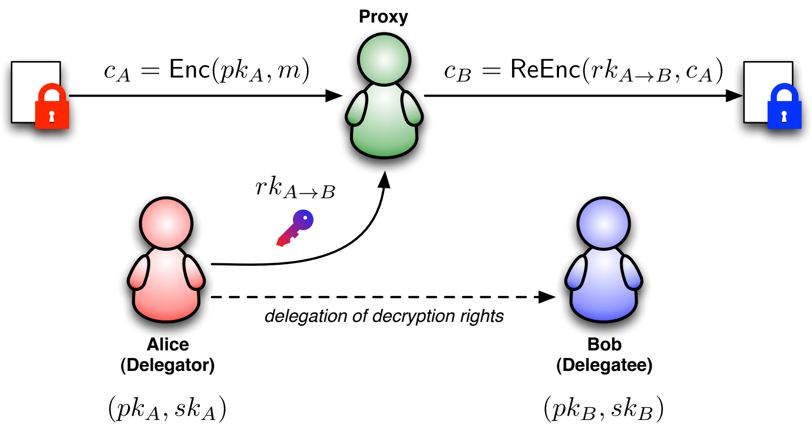

Proxy
$c_A = \mathsf{Enc}(pk_A, m)$
$c_B = \mathsf{ReEnc}(rk_{A\to B}, c_A)$
$rk_{A\to B}$
delegation of decryption rights
Alice (Delegator)
Bob (Delegatee)
(pk_A, sk_A)
(pk_B, sk_B)

Figure 1.2: Main actors and interactions in a PRE environment

access control systems where the protected resource is stored externally. Note that there is a direct correspondence between the actors involved in proxy re-encryption and those associated to the secure data sharing scenario, as shown in Table 1.1.

Table 1.1: Correspondence between PRE and the secure data sharing scenario

Proxy Re-Encryption	Secure Data Sharing
Delegator	Data Owner
Delegatee	Data Consumer
Proxy	Secure Storage Provider
Anyone	Data Producer
Ciphertexts	Outsourced information
Re-Encryption	Enforcement of access delegation

Taking this into consideration, it seems at first glance that the suitability of proxy re-encryption to our scenario is immediate. As hinted before, in a PRE-based solution to the secure data sharing scenario, private data is initially encrypted by data producers (which can be any entity that knows the proper public key) and outsourced to a semi-trusted proxy (i.e., storage provider in the cloud). By creating the corresponding re-encryption keys and giving them to the proxy, the data owner is effectively authorizing data consumers to access his data. The proxy enforces these access delegations through the re-encryption procedure using the corresponding re-encryption keys, while the information that is protected still remains confidential with respect to unauthorized parties and the proxy itself.

1.1.3 Challenges of Proxy Re-Encryption

The previous sections have introduced the original motivation of this thesis (i.e., the secure data sharing scenario) and an initial research postulate (i.e., the suitability of PRE as a solution to this scenario). There are, however, several challenges that need to be tackled before advancing the use of proxy re-encryption into the field of applications. These challenges are distributed among different abstraction levels:

- Security definitions: Challenges at this level are related with the very definition of what is considered secure in proxy re-encryption, from a cryptographic perspective. This involves different tasks, such as the analysis of security notions, modelization of adversaries, and investigation of properties of schemes, all of them essential for ensuring security in proxy re-encryption.
- Constructions: In this level, more concrete issues become relevant, such as performance (both from viewpoints of the necessary computations and the size of keys and ciphertexts), the exploration of new underlying mathematical structures for constructing schemes, and methods for producing more secure schemes generically.
- Applications: The secure data sharing scenario presented is a generic setting that can be instantiated in different use cases. The challenge here is to exploit the applicability of PRE to this scenario in order to devise concrete and useful applications.

These challenges are the dominant questions throughout this thesis, and hence, proxy re-encryption constitutes the central topic. Although the secure data sharing scenario acts as a motivation, the identified gaps have proved to be a stimulating subject worth researching in depth. The following is a more detailed account of the challenges we have identified in proxy re-encryption, classified according to the aforementioned abstraction levels.

Security definitions

In order to reason about the security of a cryptosystem, it becomes essential to have a proper definition and understanding of the notions that model its security. Since PRE schemes are also public-key encryption schemes, it is natural to "reuse" the security notions of PKE. However, in the PRE literature, these security notions are often defined in an ad-hoc manner for each scheme, with subtle variations and restrictions, caused by the lack of established definitions. This is particularly the case of the definition of attack models, which in the context of PRE are potentially richer than in PKE because of the added possibility of re-encryption.

In addition to this issue, the definition of security in the PRE context presents a peculiar dichotomy, since PRE constructions have to balance the security offered by the encryption scheme and the functionality of transforming ciphertexts through re-encryption. Both aspects are somewhat contradictory, since traditional security notions for encryption require that it should not be possible to manipulate ciphertexts in a meaningful way, a property called *non-malleability*; however, re-encryption of ciphertexts is actually a type of meaningful manipulation. For this reason, the re-encryption capability can be seen as an "attack vector" to the security of PRE schemes.

Constructions

There are several challenges associated with the construction of more secure and efficient PRE schemes. The most immediate is related to the construction of CCA-secure schemes. In the case of traditional PKE, several generic methods exist for achieving CCA-secure schemes from weakly secure cryptosystems. To the contrary, in PRE this process is completely ad-hoc for each scheme. Therefore, it would be desirable to count on analogous constructions that allow PRE schemes to "bootstrap" security by means of generic transformations.

On a more concrete note, another challenging aspect is performance. Recall that, from the motivating scenario, we postulated the use of cryptographically-enforced access control systems based on PRE. Depending on the particular application use case, these systems can be handling thousands of access requests per minute. Therefore, it is crucial that the cryptographic operations performed by both the cloud provider and the data consumers are sufficiently efficient, since these are, presumably, the most common operations. This not only has an effect on time (i.e., faster service), but also on economic cost (as more computations are tied to more costs incurred).

Finally, another topic worth exploring is that, to date, most PRE schemes are based on traditional number-theoretic foundations, such as cyclic groups where the Discrete Logarithm and Diffie-Hellman problems are hard, or those suitable for cryptographic pairings. However, there is a growing interest nowadays in other cryptographic foundations, such as lattices. Lattice-based cryptography is a promising field due to its potential with respect to post-quantum security; in some cases, performance can be greatly improved too, since lattice operations are highly parallelizable.

Applications

Once the topics of a more theoretical nature are tackled, it is necessary to work on the integration of PRE schemes within real systems. This implies dealing with protocols, architectures and implementations. These aspects are especially relevant, since often, the link between theoretical contributions and real-world systems is missing. The challenge here is to find the proper technologies that allow a seamless inclusion of a non-traditional cryptosystem such as proxy re-encryption.

Additionally, associated with the performance challenge described above, it is necessary to study the economic viability of any proposal involving cryptographic primitives, since the incurred costs may negate the utility of any devised application.

Finally, an interesting topic is to devise other potential use cases of PRE, beyond the secure data sharing scenario.

1.2 Goals and Contributions

The previous sections have given the initial motivation of this thesis, which led us to consider proxy re-encryption as a candidate cryptographic primitive to the suggested scenario. Associated with this primitive, we have presented a compilation of challenges, categorized by level of abstraction. These challenges are the driving factors that have shaped the research agenda of this thesis. Our ultimate objective with this work is therefore to contribute towards the resolution of these challenges, with the purpose of playing a part in the realization of PRE as a solution to the secure data sharing scenario. In parallel to these challenges, we next describe a set of goals that summarize our attitude towards this thesis. These goals are to be interpreted more as high-level principles, rather than specific research targets.

As justified, one of the first goals of this thesis is to contribute towards a *better understanding of the definitions of security upon which design proxy re-encryption schemes*. As stated by Bellare, Hofheinz and Kiltz in [15], "*Cryptography is founded on definitions (...) In order to have firm foundations – in particular a unique interpretation and common understanding of results – it is important to have definitional unity, meaning that different definitions intending or claiming to represent the same notion should really do so*". The work in the first part of this thesis is inspired by this principle. In the previous section we described the challenges associated with the definitions of security of proxy re-encryption. In this thesis we aim to investigate the particularities of the attack models and security notions in PRE, which are often treated in an ad-hoc way, and propose more stable and generic definitions.

Once there is a better understanding of PRE security notions, an immediate goal is to *study techniques to improve the security of PRE schemes*. Ideally, there should be generic methods and heuristics for designing PRE schemes that achieve strong and meaningful security notions.

Another goal of this thesis is to *foster the design of efficient systems*, which show a judicious use of resources. Often, academic solutions do not focus on the performance facet, which can impact dramatically the design of the systems. Resources such as computational power are seemingly unlimited in the scenarios we are considering. However, the truth is that, although their cost can be potentially highly reduced, it can rapidly add up so as to make proposed solutions unfeasible from the economic standpoint. In this thesis, we consider it to be of paramount importance to not neglect this perspective and to study the impact in costs of any proposed solutions.

Finally, a principle that guides the development of this thesis is to *participate in the multiple strata that conform this fascinating topic*, from the very definitions of security to the specifics of applications, through the construction of new primitives. Although our point of departure is the use of proxy re-encryption as a cryptographic mechanism for protect data confidentiality, we intentionally do not want to circumscribe this work to a specific facet, and rather prefer to follow a more transversal approach.

With these goals in mind, the following is a list of the main contributions of this thesis, which can be linked with some of the challenges mentioned in the previous section:

- We review the basic concepts of proxy re-encryption, including definitions, security models and properties.
- We analyze of the current state of the art on proxy re-encryption schemes, including a comparative study of the performance of several schemes, both from the theoretical and experimental points of view. In addition, we review the state of research on applications of proxy re-encryption, in particular the case of the secure data sharing problem.
- We examine the notions of security for proxy re-encryption and identify a parametric family of attack models that not only considers the availability of the decryption oracle, but also that of re-encryption. This parametric

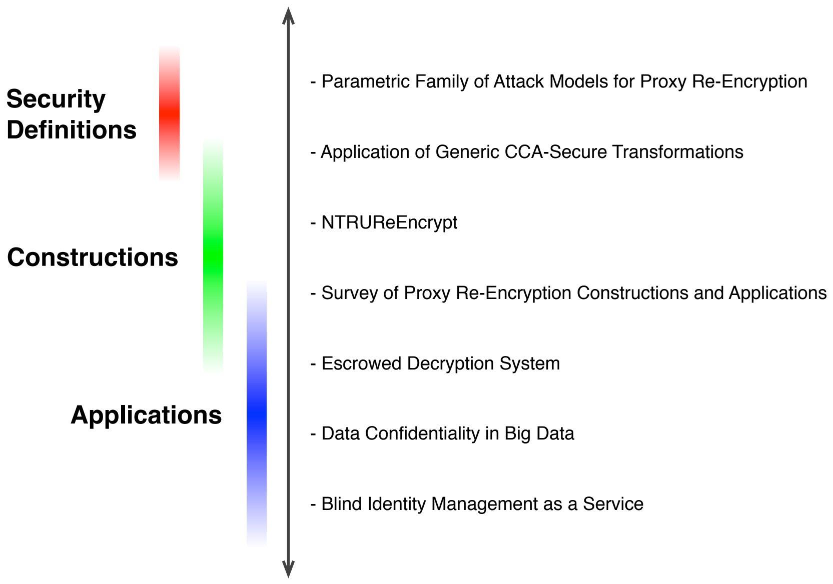

Security Definitions
Constructions
Applications
- Parametric Family of Attack Models for Proxy Re-Encryption
- Application of Generic CCA-Secure Transformations
- NTRUReEncrypt
- Survey of Proxy Re-Encryption Constructions and Applications
- Escrowed Decryption System
- Data Confidentiality in Big Data
- Blind Identity Management as a Service

Figure 1.3: Contributions of this thesis

family of attack models for PRE allows us to define a collection of security notions, whose relations we also analyze.

- We study the applicability of generic CCA-secure transformations to proxy re-encryption. In particular, we focus on the Fujisaki-Okamoto transformation, and formulate sufficient conditions that allow to use it directly in PRE. These conditions include a new proxy re-encryption property called *perfect key-switching*; we also make use of intermediate security notions derived from our parametric family. Additionally, we discuss the adaptation of similar transformations, namely REACT and GEM.

- We present NTRUReEncrypt, a new lattice-based proxy re-encryption scheme. This scheme is based on the NTRU cryptosystem and is extremely efficient, in comparison with current schemes. In addition, we describe a provably-secure version that is safe against chosen-plaintext attacks.

- We propose several applications of PRE, such as a model for privacy-preserving Identity Management as a Service, a system for delegating access to encrypted information in Big Data clusters, and an escrowed decryption system.

Figure 1.3 depicts these contributions along an axis that represents the level of abstraction. As mentioned before, our contributions range from the very definitions of security to the field of applications, by way of concrete constructions.

1.3 Thesis outline

In this chapter, we have begun by introducing the practical motivation of this thesis, which is the protection of sensitive data in outsourced environments, such as the cloud. We have seen that this scenario poses several challenges from the point of view of privacy and security, and we have briefly explained the usual countermeasures. After this, our conclusion is that it is necessary to devise more advanced safeguards, based on the use of cryptographic mechanisms, in order to protect the confidentiality of sensitive data against a broader class of threats. This have led us to defend the use of proxy re-encryption as a prime example of such cryptographic mechanisms, which constitutes a central research postulate of this thesis. In this chapter, we have also given a brief overview of what is proxy re-encryption and described the main challenges associated to this cryptosystem.

Before proceeding, Chapter 2 overviews basic concepts and definitions that are used in this thesis. As a substantial part of this thesis is heavily based on provable security, we here provide a general review of this area of cryptography.

Once the essential foundations have been established, it is necessary to study the state of current research, in order to broaden our understanding of proxy re-encryption, and consequently, identify research gaps. Chapter 3 presents an analysis of the state of the art of this type of cryptosystem, not only from the perspective of specific constructions, but also applications. Firstly, we survey the main proxy re-encryption schemes so far, and provide a detailed analysis of their characteristics. In line with the goal of fostering the design of efficient systems, we also study the performance of selected schemes, both theoretically and empirically. Secondly, we review applications of proxy re-encryption, with a special focus on data sharing in the cloud. In this part we analyze in more detail our research postulate – that proxy re-encryption constitutes a feasible solution for this scenario, both from the functional and efficiency perspectives.

In Chapter 4 we study the conventional security definitions for proxy re-encryption schemes, which are based on those inherited from public key encryption (PKE). One of the principal building blocks of these security definitions is the attack model, which defines the capabilities of an adversary in a security game. PRE is inherently more complex than PKE, but attack models for PRE have not been developed further. To this respect, we define a parametric family of attack models for PRE, based on the availability of both the decryption and re-encryption oracles during the security game, that enables the definition of a set of intermediate security notions. We analyze some relations among these notions of security, and in particular, the separations that arise when the re-encryption oracle leaks re-encryption keys. In addition, we discuss which of these security notions represent meaningful adversarial models for PRE. Finally, we show how a recent PRE scheme is based on a security model that does not capture chosen-ciphertext attacks through re-encryption, and for which we describe an attack under a more realistic security notion. This attack emphasizes the fact that PRE schemes that leak re-encryption keys cannot achieve strong security notions.

Chapter 5 presents new proxy re-encryption constructions, consisting of two separate contributions. First, we explore the use of lattice-based cryptography for constructing more efficient PRE schemes. In this chapter, we present NTRUReEncrypt, a new bidirectional and multihop proxy re-encryption scheme based on NTRU [16], a widely known lattice-based cryptosystem. We give two versions of our scheme: the first one is based on the conventional NTRU encryption scheme and, although it lacks a security proof, remains as efficient as its predecessor; the second one is a provably-secure variant that is safe against chosen-plaintext attacks. For the second part, we focus on the construction of more secure PRE schemes by means of generic transformations. To this end, we study the adaptation of conventional generic transformations, such as Fujisaki-Okamoto [17] and REACT [18], originally designed to achieve CCA-security. We show that a direct and naive application of these transformations leads to flawed schemes, and give several failed examples from the literature. In addition, we propose an extension of the Fujisaki-Okamoto transformation for PRE, which achieves a weaker form of CCA-security in the random oracle model, and identify the conditions for applying it.

Chapter 6 is devoted to new applications of proxy re-encryption, with a clear focus on the secure data sharing scenario. As described in this introduction, this scenario can be seen as a generalization of different application use cases. In this chapter, we describe the integration of PRE within some of these use cases. Our first proposal is BlindIdM, a model for privacy-preserving Identity Management as a Service, with the intention of enabling organizations to outsource their identity management to the cloud in a secure way, without the cloud provider being able to read the identity information. We show how PRE can be integrated to SAML 2.0, a standard identity management protocol [19], as an example of instantiation of the BlindIdM model. A second proposal is presented next, this time focusing on the Big Data Analytics use case, as introduced in Section 1.1.1. We describe an extension to the Apache Hadoop system [20] where stored data is always encrypted and encryption keys do not need to be shared between different data sources; the use of proxy re-encryption allows stored data to be re-encrypted into ciphered data that the cluster nodes can decrypt with their own keys when a job is submitted. The last application we present differs from the others, as it is not directly related to the cloud data sharing scenario. In Section 6.3 we describe an escrowed encryption system based on proxy re-encryption. The goal of this system is to serve as a typical public-key encryption scheme, but at the same time, the decryption procedure is escrowed by means of proxy re-encryption; a key aspect of this proposal is that the escrowed decryption can only be achieved with the collaboration of a set of trusted custodians, which are specialized entities chosen by the users.

Finally, Chapter 7 summarizes the contributions of this thesis and describes lines of future work and open research problems.

1.4 Publications and Funding

The work in this thesis has led to several publications in journals and international conferences. Next, we provide a list of these contributions, organized by the type of publication:

Articles in ISI-JCR Journals

- D. Nuñez, and I. Agudo. BlindIdM: A Privacy-Preserving Approach for Identity Management as a Service. In *International Journal of Information Security*, vol. 13, issue 2, Springer, pp. 199-215, 2014. ISI JCR Impact Factor 2014: 0.963.

International conference papers

- D. Nuñez, I. Agudo, and J. Lopez. A Parametric Family of Attack Models for Proxy Re-Encryption. In *Proceedings of the 28th IEEE Computer Security Foundations Symposium (CSF)*, IEEE Computer Society, pp. 290-301, 2015.
- D. Nuñez, I. Agudo, and J. Lopez. NTRUReEncrypt: An Efficient Proxy Re-Encryption Scheme Based on NTRU. In *Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security (AsiaCCS)*, pp. 179-189, 2015.
- D. Nuñez, I. Agudo, and J. Lopez. Delegated Access for Hadoop Clusters in the Cloud. In *Proceedings of the 6th IEEE International Conference on Cloud Computing Technology and Science (CloudCom 2014)*, IEEE Computer Society, pp. 374-379, 2014.

- D. Nuñez, I. Agudo, and J. Lopez. Integrating OpenID with Proxy Re-Encryption to enhance privacy in cloud-based identity services. In *Proceedings of the 4th IEEE International Conference on Cloud Computing Technology and Science (CloudCom 2012)*, IEEE Computer Society, pp. 241-248, 2012.

Other research papers under submission at this stage:

- D. Nuñez, I. Agudo, and J. Lopez. Analysis of Proxy Re-Encryption Schemes and their Application to Cloud Data Sharing. Submitted to *Future Generation Computer Systems*. ISI JCR Impact Factor 2014: 2.786.

- D. Nuñez, I. Agudo, and J. Lopez. On the Application of Generic CCA-Secure Transformations to Proxy Re-Encryption. Submitted to *Security and Communication Networks*. ISI JCR Impact Factor 2014: 0.72.

Apart from these publications, there are other works that have been developed in parallel and that are indirectly related to this thesis. These additional publications are listed next, sorted by date:

- I. Agudo, D. Nuñez, G. Giammatteo, P. Rizomiliotis, and C. Lambrinoudakis. Cryptography Goes to the Cloud. In Proceedings of the 1st International Workshop on Security and Trust for Applications in Virtualised Environments (STAVE 2011), C. Lee, J-M. Seigneur, J. J. Park, and R. R. Wagner Eds., Communications in Computer and Information Science 187, Springer, pp. 190–197, 2011.

- C. Alcaraz, I. Agudo, D. Nuñez, and J. Lopez. Managing Incidents in Smart Grids à la Cloud. In Proceedings of the 3rd IEEE International Conference on Cloud Computing Technology and Science (CloudCom 2011), IEEE Computer Society, pp. 527-531, 2011.

The work in this thesis has been funded by a FPI fellowship from the Junta de Andalucía as part of the project PISCIS (P10-TIC-06334). In addition, parts of this work have received support from the projects FISICCO (P11-TIC-07223), PERSIST (TIN2013-41739-R), ARES (CSD2007-00004), and SPRINT (TIN2009-09237).

Chapter 2

Preliminaries

This section provides an overview of basic topics upon which this thesis is grounded. Apart from basic notion used throughout this thesis, we review here the most common cryptographic primitives. In addition, we also address the topic of provable security, an area of cryptography that deals with the analysis of the security of cryptosystems using mathematical proofs.

2.1 Mathematical Notation

We use the $\oplus$ symbol to denote the exclusive-or operator, and $||$ to denote the concatenation operator. The $\perp$ symbol is used for representing an error message. We denote by $\|\cdot\|$ and $\|\cdot\|_\infty$ the vector Euclidean norm and infinite norm, respectively.

Throughout this thesis, we use the asymptotic notations $\omega(\cdot), O(\cdot)$ and $\Omega(\cdot)$. The expression $\omega(g(n))$ denotes the class of functions that dominate $g(n)$ asymptotically (i.e., for sufficiently large n). That is, $f(n) = \omega(g(n))$ means that $f(n)$ grows strictly faster than $g(n)$. For example, a linear function (e.g., $f(n) = n$) dominates a logarithmic function (e.g., $g(n) = \log n$), since, asymptotically, it grows much faster; thus, $n = \omega(\log n)$. A slightly different notion is $O(g(n))$, which denotes the class of functions that set an upper bound on $g(n)$, asymptotically. That is, $f(n) = O(g(n))$ means that $f(n)$ grows no faster than $g(n)$. Conversely, $\Omega(g(n))$ denotes the class of functions that set a lower bound on $g(n)$, asymptotically. That is, $f(n) = \Omega(g(n))$ means that $f(n)$ grows no slower than $g(n)$.

Based on this notation, we say that a function $f(n)$ is negligible if $f(n) = n^{-\omega(1)}$.

Finally, in some cases we use the expression $poly(n)$ to denote the class of functions which are expressible as polynomials in n.

2.2 Cryptographic primitives

In this section we review the basic cryptographic primitives that we use throughout this thesis, such as symmetric encryption, public-key encryption and hash functions.

2.2.1 Symmetric Encryption

Symmetric encryption, or more accurately, symmetric-key encryption, is a class of encryption algorithms where the parties involved share a common secret (i.e., the key), which is used is used when they want to communicate secretly with each other [21]. Therefore, the sender uses the key to encrypt the message, and the receiver uses the same key to decrypt it.

The following is a generic definition of the syntax of any symmetric encryption scheme.

Definition 2.1 (SKE scheme). *A symmetric-key encryption scheme is a tuple of algorithms (KeyGen, Enc, Dec):*

- $\mathsf{KeyGen}(\lambda) \to K$. *On input security parameter λ, the key generation algorithm* KeyGen *outputs a key $K \in \mathcal{K}$.*
- $\mathsf{Enc}(K, m) \to c$. *On input a key K and a message $m \in \mathcal{M}$, the encryption algorithm* Enc *outputs a ciphertext $c \in \mathcal{C}$.*
- $\mathsf{Dec}(K, c) \to m$. *On input a key K and a ciphertext $c \in \mathcal{C}$, the decryption algorithm* Dec *outputs a message $m \in \mathcal{M}$.*

The sets of all possible keys, messages and ciphertexts are denoted by $\mathcal{K}, \mathcal{M}$ and $\mathcal{C}$, respectively.

Once we have defined the syntax, we can establish the correctness condition for symmetric encryption. In order to say that a symmetric encryption scheme is *correct* it must satisfy a basic condition: for every key $K \in \mathcal{K}$ and every $m \in \mathcal{M}$, then it holds that $\mathsf{Dec}(K, \mathsf{Enc}(K, m)) = m$.

The main two types of symmetric encryption algorithms are:

- *Stream ciphers*, where each "digit" of the message is processed independently with a digit of the key. The One-Time Pad (OTP) is a prime example of stream cipher, where the message is XORed with a random key of the same length; the security of the OTP crucially depends on that the key is never reused (i.e., the key can only be used "one time").
- *Block ciphers*, which are designed to operate over units of a predetermined size (i.e., blocks). For instance, the Advanced Encryption Standard (AES), originally proposed by Daemen and Rijmen [22] and standarized by the US' National Institute of Standards and Technology (NIST), and widely recognized as the most used block cipher nowadays, has a block size of 128 bits.

2.2.2 Public Key Encryption

As opposed to symmetric encryption, in public-key encryption (PKE) it is not assumed that communicating parties share any common secret. Instead, PKE requires two separate keys, one *public*, used to encrypt messages, and one *private*, used to decrypt the corresponding ciphertexts; these two keys are linked to each other by some mathematical properties that depend on each PKE scheme. For this reason, public-key encryption is sometimes called *asymmetric encryption.*

The syntax of any PKE scheme is as follows:

Definition 2.2 (PKE scheme)**.** *A public-key encryption scheme is a tuple of algorithms (*KeyGen*,* Enc*,* Dec*):*

- $\mathsf{KeyGen}(\lambda) \to (pk, sk)$. *On input security parameter* λ*, the key generation algorithm* KeyGen *outputs a pair of public and private keys* (pk, sk)*.*
- $\mathsf{Enc}(pk, m) \to c$. *On input the public key* pk *and a message* $m \in \mathcal{M}$*, the encryption algorithm* Enc *outputs a ciphertext* $c \in \mathcal{C}$*.*
- $\mathsf{Dec}(sk, c) \to m$. *On input the secret key* sk *and a ciphertext* $c \in \mathcal{C}$*, the decryption algorithm* Dec *outputs a message* $m \in \mathcal{M}$ *or the symbol* $\perp$ *indicating* c *is invalid.*

The plaintext and ciphertext spaces are denoted by $\mathcal{M}$ *and* $\mathcal{C}$*, respectively.*

As for symmetric encryption, it is necessary to define under what conditions a PKE is correct. Although similar to symmetric case, this definition takes the asymmetry of the keys into consideration. The correctness condition for PKE schemes is as follows: for every pair of public and private keys (pk, sk) that results of $\mathsf{KeyGen}(\lambda)$ and every $m \in \mathcal{M}$, then it holds that $\mathsf{Dec}(sk, \mathsf{Enc}(pk, m)) = m$.

2.2.3 Hash Functions

A cryptographic hash function (or simply, hash function) is a deterministic algorithm H that maps inputs of arbitrary size to fixed-length outputs in an efficient way. In order to be considered secure, hash functions must meet some minimum requirements, such as pre-image resistance (it should not be possible to find an input that hashes to a given output), second pre-image resistance (it should not be possible to find a second input that hashes to the same value as a given input), and collision resistance (it should not be possible to find two inputs that hash to the same value).

2.3 Provable Security

In this section we briefly review some of the basic concepts on provable security, and in particular, those specific to public-key encryption. Later on, these concepts will be reused when analyzing the security of proxy re-encryption schemes. As described before, provable security deals with the analysis of the security of cryptosystems using mathematical proofs. Often, these proofs are offered in the form of *reductions*, which are formal argumentations where an algorithm that transforms one problem (e.g., factoring large numbers)into another problem (e.g., breaking some security goal of a given cryptosystem) is defined. It could appear at first sight that this "reductionist approach" is somewhat weak due to its conditional nature [23], but, in fact, it had made possible the rigorous analysis of the security of a myriad of cryptosystems, in an elegant, yet powerful, way.

In order to create these proofs, it becomes necessary first to rigorously model the security goals of the cryptosystem and the adversarial capabilities that are allowed. This is explained in more detail below.

2.3.1 Security Notions of PKE

In the context of PKE, security notions are usually defined as the combination of a security goal and an attack model [24]. Throughout this thesis, we focus on the *indistinguishability of encryptions* (IND) goal, which formalizes the inability of an adversary to distinguish which message a given ciphertext encrypts. A weaker goal is *one-wayness* (OW), which represents the inability of an adversary to extract the underlying plaintext from a given ciphertext. The strongest goal is *non-malleability* (NM), where an adversary should not be able to produce ciphertexts such that, for a given ciphertext, the plaintexts are meaningfully related. With regard to attack models, three options are usually considered: (i) *chosen-plaintext attack* (CPA), (ii) *non-adaptive chosen-ciphertext attack* (CCA1), and (iii) *adaptive chosen-ciphertext attack* (CCA2). In a CPA model, the only capability of the adversary is to encrypt plaintexts of her choice (althought this capability is inherent in a public-key cryptosystem). Under CCA1, the adversary is also given a decryption capability (i.e., a decryption oracle) but only for its use before receiving the challenge ciphertext. Finally, in the CCA2 model, the adversary may use the decryption oracle in any moment, with the only restriction of not asking for the decryption of the challenge ciphertext. Therefore, a security notion can be seen as a tuple *goal-atk*, where *goal* $\in$ {OW, IND, NM}, and *atk* $\in$ {CPA, CCA1, CCA2}.

It can be seen that these attack models are differentiated by the changes on the decryption capabilities of the adversary, which can be modeled through the availability of a *decryption oracle*. Informally, a decryption oracle is a function $\mathcal{O}_{dec}(\cdot)$ that the adversary can query on any ciphertext c (except the challenge ciphertext c^*) and that outputs the decryption of c with the target secret key. No additional oracles are necessary for describing the above notions of security for PKE. As mentioned above, in this thesis we focus on the indistinguishability goal, so we are concerned with three possible security notions for PKE. The following definitions, adapted from [24], comprise these security notions in a formal manner.

Definition 2.3 (Indistinguishability game [24]). *Let* Π=(KeyGen, Enc, Dec) *be a public-key encryption scheme,* $A = (A_1, A_2)$ *a polynomial-time adversary, and* Ω_1 *and* Ω_2 *the set of available oracles for* A_1 *and* A_2*, respectively. For atk* $\in$ {CPA, CCA1, CCA2}, $n \in \mathbb{N}$, *and* $\delta \in \{0, 1\}$, *the indistinguishability of encryptions game is defined by the experiment*

```
Experiment $\mathbf{Exp}^{\textsf{IND-}atk}_{\Pi,A,\delta}(n)$
    $(pk^*, sk^*) \xleftarrow{R} \mathsf{KeyGen}(n)$;        $(m_0, m_1, s) \leftarrow A_1(pk^*)$;
    $c^* \leftarrow \mathsf{Enc}(pk^*, m_\delta)$;          $d \leftarrow A_2(m_0, m_1, s, c^*)$;
    return $d$
```

where

$$\begin{array}{lll} \text{If } atk = \mathsf{CPA} & \text{then } \Omega_1 = \varnothing & \text{and } \Omega_2 = \varnothing \\ \text{If } atk = \mathsf{CCA1} & \text{then } \Omega_1 = \{\mathcal{O}_{dec}\} & \text{and } \Omega_2 = \varnothing \\ \text{If } atk = \mathsf{CCA2} & \text{then } \Omega_1 = \{\mathcal{O}_{dec}\} & \text{and } \Omega_2 = \{\mathcal{O}_{dec}\} \end{array}$$

Definition 2.4 (Advantage [24])**.** *Let Π be a public-key encryption scheme and A a polynomial-time adversary. For atk $\in \{\mathsf{CPA}, \mathsf{CCA1}, \mathsf{CCA2}\}$ and $n \in \mathbb{N}$, the advantage of A is given by*

$$\mathbf{Adv}^{\mathsf{IND}\text{-}atk}_{\Pi,A}(n) = |\Pr[\mathbf{Exp}^{\mathsf{IND}\text{-}atk}_{\Pi,A,1}(n) = 1] - \Pr[\mathbf{Exp}^{\mathsf{IND}\text{-}atk}_{\Pi,A,0}(n) = 1]|$$

We say that the encryption scheme Π is IND-atk secure if the advantage $\mathbf{Adv}^{\mathsf{IND}\text{-}atk}_{\Pi,A}(n)$ is negligible.

2.3.2 Hardness Assumptions

In previous sections, we mentioned that provable-secure cryptographic schemes base their security upon the hardness of problems that are conjectured to be hard. That is, the proofs of security of these schemes are in fact reductions to the difficulty of some of these hard problems, which are usually well-defined and studied, to the extent that no efficient algorithms are known to solve them. The conjectures about the difficulty of such problems are known in cryptography as *hardness assumptions*. In this section we briefly describe some of the problems that lead to the hardness assumptions referred throughout this thesis, categorized by their nature.

Discrete Logarithm and Diffie-Hellman Problems

This family of problems is circumscribed to certain kinds of algebraic groups that have cryptographic applications. Before proceeding with the definition of these hard problems, it is necessary to give some general definitions, although we will assume that the reader is familiar with the topic.

A group, denoted by $\mathbb{G}$, is an algebraic structure that combines a set of elements with a binary operation defined over them, that satisfies the closure and associativity properties, and that guarantees the existence of the identity and inverse elements. In the following, we will assume that $\mathbb{G}$ is a group of order q, whose operation is denoted by multiplication. By the definition of an algebraic group, the multiplication of any pair of elements of the group yields a result within the group; in other words, the group is closed under the multiplication operation. Knowing this, if one operates an element $x \in \mathbb{G}$ by itself several times, this can be seen as an exponentiation. Since the group is closed under the multiplication, then $x^n \in \mathbb{G}$ for any $n \in \mathbb{Z}$. A generator g is an element of $\mathbb{G}$ such that $\mathbb{G} = \{g^1, g^2, ..., g^q\}$. It is said then that g generates $\mathbb{G}$ and denoted as $\mathbb{G} =< g >$.

The inverse function to the exponentiation is called the discrete logarithm. Informally, let $y = x^n$, then $n = \log_x y$ is the discrete logarithm of y in base x. The discrete logarithm is a cornerstone concept in cryptography, since for certain kinds of groups, it can be shown that no efficient algorithm exists for computing it. Among these groups, we find, for example, the large prime order subgroups of $\mathbb{Z}_p$ for p prime. Based on this, the following is a definition of the problem of computing the discrete logarithm for generic groups.

Definition 2.5 (Discrete Logarithm Problem (DL)). *Given a tuple $(g, g^a) \in \mathbb{G}^2$, where $a \in \mathbb{Z}_q$, the DL problem in $\mathbb{G}$ is to compute a.*

Although the DL assumption appears at the heart of most cryptosystems based on a group structure, it is often considered an overly weak assumption [25]. Based on the DL problem, Diffie and Hellman defined a key exchange protocol in their seminal paper "New Directions on Cryptography" [26]. The security of their protocol implicitely defined what later was dubbed as the Computational Diffie-Hellman problem (or simply, the Diffie-Hellman problem).

Definition 2.6 (Computational Diffie-Hellman Problem (CDH)). *Given a tuple $(g, g^a, g^b) \in \mathbb{G}^3$, where $a, b \in \mathbb{Z}_q$, the CDH problem in $\mathbb{G}$ is to compute g^{ab}.*

There is no efficient algorithm for solving this problem, for a sufficiently large group. In fact, the most efficient method for solving CDH is to solve the DL problem: on input a tuple (g, g^a, g^b), one computes a as the discrete logarithm of g^a in base g, and outputs $(g^b)^a$ as the result. The CDH assumption has been used in many cryptosystems, but as in the case of the DL assumption, sometimes it is also too weak. For this reason, the decisional version of the CDH problem is often considered.

Definition 2.7 (Decisional Diffie-Hellman Problem (DDH)). *Given a tuple $(g, g^a, g^b, g^c) \in \mathbb{G}^4$, where $a, b, c \in \mathbb{Z}_q$, the DDH problem in $\mathbb{G}$ is to decide whether $c = ab$.*

The Decisional Diffie-Hellman problem is specially useful in cryptography since it directly implies the semantic security of multitude of cryptosystems, such as the ElGamal encryption scheme [25].

It is easy to see why DDH is not harder than CDH: if one could solve CDH, then the result of such algorithm could be used for trivially deciding if $g^c = g^{ab}$. However, the converse – that CDH is not harder than DDH – is not true for generic groups [27]. That is, there is a gap between these two problems. Based on this fact, Okamoto and Pointcheval presented in [28] the following problem, which precisely covers this gap.

Definition 2.8 (Gap Diffie-Hellman Problem (gap-DH))**.** *Given a tuple* $(g, g^a, g^b) \in \mathbb{G}^3$*, where* $a, b \in \mathbb{Z}_q$*, the gap-DH problem in* $\mathbb{G}$ *is to compute* g^{ab} *with the help of an oracle that solves the DDH problem in* $\mathbb{G}$*.*

The gap-DH problem has been used in the security reductions of several cryptosystems. In particular, we are interested on its role in the REACT [18] and GEM [29] generic transformations, which we study in Section 5.2.

Pairing-based Problems

Informally, a bilinear pairing is a map between a pair of group elements to another element from a certain target group. What make pairings interesting for cryptography is the *bilinear property*: let $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ be a bilinear pairing[1]; the bilinear property of pairings states that:

$$e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$$

It can be seen that the introduction of a pairing between $\mathbb{G}$ and $\mathbb{G}_T$ makes the DDH problem in $\mathbb{G}$ easy: on input a DDH tuple $(g, g^a, g^b, g^c) \in \mathbb{G}^4$, one can make use of the bilinear property of the pairing to decide whether $g^c = g^{ab}$ by checking if the equation $e(g, g^c) = e(g^a, g^b)$ holds. It can be seen that this equation is equivalent to $e(g, g)^c = e(g, g)^{ab}$, which is true if and only if $c = ab$. For this reason, pairings were proposed first as a means for cryptanalysis. Later on, "constructive" uses of pairings were also proposed, which has made possible the creation of a great variety of cryptosystems. To this end, in pairing-friendly groups one usually considers other types of hard problems; the most prominent are similar in form to the Diffie-Hellman problems, but adapted to the use of pairings.

[1]Although this definition distinguish between $\mathbb{G}_1$ and $\mathbb{G}_2$, from now on we will assume that the pairing is *symmetric*, so $\mathbb{G}_1$ and $\mathbb{G}_2$ are the same group and will be denoted as $\mathbb{G}$.

Definition 2.9 (Computational Bilinear Diffie-Hellman Problem (CBDH))**.** *Given a tuple* $(g, g^a, g^b, g^c) \in \mathbb{G}^4$*, where* $a, b, c \in \mathbb{Z}_q$*, the CBDH problem is to compute* $e(g, g)^{abc}$*.*

Definition 2.10 (Decisional Bilinear Diffie-Hellman Problem (DBDH))**.** *Given a tuple* $(g, g^a, g^b, g^c, e(g, g)^d) \in \mathbb{G}^4 \times \mathbb{G}_T$*, where* $a, b, c, d \in \mathbb{Z}_q$*, the DBDH problem is to compute* $e(g, g)^{abc}$*.*

In this thesis we also make use of other, more involved, pairing-based hard problems. The l-Decisional Bilinear Diffie-Hellman Inversion problem was first proposed by Dodis and Yampolskiy in [30]; later, Boneh et al. proposed in [31] a generalization called the l-weak Decisional Bilinear DH Inversion problem.

Definition 2.11 (l-Decisional Bilinear DH Inversion problem [30])**.** *Given a tuple* $(g, g^a, g^{a^2}, ..., g^{a^l}, e(g, g)^d) \in \mathbb{G}^{l+1} \times \mathbb{G}_T$*, the l-DBDHI problem in* $(\mathbb{G}, \mathbb{G}_T)$ *is to decide whether* $d = a^{l+1}$*.*

Definition 2.12 (l-weak Decisional Bilinear DH Inversion problem [31])**.** *Given a tuple* $(g, g^a, g^{a^2}, ..., g^{a^l}, g^b, e(g, g)^d) \in \mathbb{G}^{l+2} \times \mathbb{G}_T$*, the l-wDBDHI problem in* $(\mathbb{G}, \mathbb{G}_T)$ *is to decide whether* $d = a^{l+1}b$*.*

Lattice-based Problems

A lattice $\mathcal{L}(B)$ is the set of all integer combinations of the basis $B = \{b_1, ..., b_n\}$ of n linearly independent vectors. That is, lattice $\mathcal{L}(B)$ is defined as:

$$\mathcal{L}(B) = \{B \cdot z \ : \ z \in \mathbb{Z}^n\}$$

In cryptography, we are interested on integer lattices, i.e., those where $B \in \mathbb{Z}^{n\times n}$, and, specially, on q-ary lattices, which are the modular version of integer lattices.

Elements in a lattice can be represented as vectors in the space where the basis is defined. In the case of integer lattices, the space is $\mathbb{Z}^{n\times n}$, so elements of the lattice can be seen as integers vectors. For this reason, lattice-based schemes are usually based on elementary operations over vectors and matrices, such as vector/matrix addition, inner product, etc.

There are several hard problem associated to lattices. In this thesis, we will focus in particular in the Learning With Errors (LWE) and the Short Integer Solution (SIS) families of problems.

Learning With Errors (LWE) This problem was introduced by Regev in 2005 [32], and has been used for creating several cryptosystems for public key encryption [33, 34], identity-based encryption [35], and even fully homomorphic encryption [36].

Definition 2.13 (Learning With Errors problem (LWE) [34]). *For given a dimension n, a modulus q, a noise distribution ψ over $\mathbb{Z}_q$, and an unknown vector $\vec{s} \in \mathbb{Z}_q^n$ sampled uniformly at random, the LWE problem is to find $\vec{s}$ given a polynomial number m of samples of the form $(\vec{a}_i, \vec{b}_i = \vec{a}_i \cdot \vec{s} + x_i) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$, where $\vec{a}_i \in \mathbb{Z}_q^n$ is uniformly random and x_i is sampled from the noise distribution ψ, for $i \in \{1, ..., m\}$.*

Typically, the noise distribution ψ is chosen to be a Gaussian centered around 0. Note that this is the computational (or "search") version of the LWE problem. One can define a decisional version, where the problem is to distinguish between the above ditribution of samples and uniformly random samples from $\mathbb{Z}_q^n \times \mathbb{Z}_q$. Both versions have been proven to be equivalent [32].

From a practical viewpoint, however, the LWE problem is not usually considered since its instantiation requires key sizes and computation times that are at least quadratic in the security parameter [37]. Lyubaskevsky et al. proposed the use of lattices with additional algebraic structure, and introduced in [37] the Ring Learning With Errors (Ring-LWE) problem, which can be seen as a variant of the LWE problem but in a ring setting.

Let $\Phi(x) = x^n + 1$, with n a power of 2; that is, $\Phi(x)$ is the $2n$-th cyclotomic polynomial and is irreducible over the rationals. Let $\mathcal{R}$ be the ring of integer polynomials $\mathbb{Z}[x]$ modulo $\Phi(x)$; that is, $\mathcal{R} = \mathbb{Z}[x]/\Phi(x)$. Elements of $\mathcal{R}$, which are residues modulo $\Phi(x)$, can be represented by integer polynomials of degree less than n, we often treat them as vectors in $\mathbb{Z}^n$. Let q be a prime integer such that $q = 1 \mod 2n$, and $\mathcal{R}_q = \mathcal{R}/q = \mathbb{Z}_q[x]/\Phi(x)$. Elements of $\mathcal{R}_q$ can be represented by polynomials of degree less than n with coefficients in $\mathbb{Z}_q$, so we often treat them as vectors in $\mathbb{Z}_q^n$. The Ring-LWE problem is defined as follows.

Definition 2.14 (Ring-LWE problem). *Let $s \in \mathcal{R}_q$ a uniformly random ring element and ψ a distribution over $\mathcal{R}_q$, then we define $A_{s,\psi}$ as the distribution that samples pairs of the form (a, b), where a is chosen uniformly from $\mathcal{R}_q$ and $b = a \cdot s + e$, for some e sampled from ψ. The distribution $A_{s,\psi}$ is also called the Ring-LWE distribution. The Ring-LWE problem is to distinguish distribution $A_{s,\psi}$ from a uniform distribution over $\mathcal{R}_q \times \mathcal{R}_q$.*

Note that in this case, we are directly defining the decisional version of this problem. The Ring-LWE is interesting because it produces much more efficient cryptosystems, as opposed to the traditional LWE problem.

Short Integer Solution The Short Integer Solution (SIS) problem was proposed by Ajtai in [38], where it was proven to be at least as hard as other lattice problems, and can be seen as a variant of the subset-sum problem over a special additive group [37]. The SIS problem has been used for defining one-way collision-resistant hash functions [38, 39], and digital signature schemes [35, 40].

Definition 2.15 (Short Integer Solution Problem (SIS)). *Given a uniform matrix $A \in \mathbb{Z}_q^{n\times m}$, and a real β, the SIS problem is to find a "short" nonzero vector $z \in \mathbb{Z}^m$ such that $A \cdot z = 0 \mod q$ and $||z||_2 \leq \beta$. Note that when $m \geq n \log q$, short solutions are guaranteed to exist.*

The following is a generalization of the SIS problem that is usually considered in lattice-based cryptosystems.

Definition 2.16 (Inhomogenous Short Integer Solution Problem (ISIS)). *Given a uniform matrix $A \in \mathbb{Z}_q^{n\times m}$, a vector $b \in \mathbb{Z}_q^n$, and a real β, the ISIS problem is to find a "short" nonzero vector $z \in \mathbb{Z}^m$ such that $A \cdot z = b \mod q$ and $||z||_2 \leq \beta$. Note that when $m \geq n \log q$, short solutions are guaranteed to exist.*

2.3.3 The Random Oracle Model

Hash functions are an essential tool in many cryptographic schemes and protocols. For this reason, it is also necessary to take them into consideration when proving the security of schemes and protocols. However, it is difficult to construct these proofs from known properties of hash functions (e.g., collusion resistance).

Because of this problem, it is often necessary to resort to a *random oracle*, which is an ideal abstraction of a hash function. More specifically, a random oracle is a function that, for each possible input, outputs a uniformly distributed random value. Random oracles are deterministic, in the sense that they must output the same value for the same input. In addition, the simulator of the random oracle is free to read previous inputs and outputs, and even to determine its behavior, as long as the output remains uniformly distributed. The Random Oracle model was first proposed by Bellare and Rogaway in [41], and has been used extensively since then.

Chapter 3

A Systematic Analysis of Proxy Re-Encryption

In this chapter, we provide a detailed overview of the concept of proxy re-encryption, and survey the current state of research, both for constructions and applications, following a systematic approach. We review the main proxy re-encryption schemes so far, and provide a detailed analysis of their characteristics. Additionally, we also study the efficiency of selected schemes, both theoretically and empirically. Finally, we discuss some applications of proxy re-encryption, with a focus on data sharing in the cloud, which is the original motivating scenario in this thesis.

3.1 Introduction

In 1997, Mambo and Okamoto [42] introduced the notion of "*proxy cryptosystem*", after envisioning the possible applications of a cryptosystem capable of transforming ciphertexts intended for Alice into ciphertexts decryptable by Bob. A naive solution to this problem could be that the original recipient decrypts the ciphertext and subsequently encrypts the original message with the public key of the new recipient. However, the *decrypt-and-encrypt* solution implies that the original recipient must be available for re-encrypting ciphertexts when needed, which is not always feasible. Moreover, this solution is very inefficient from the communication viewpoint. Mambo and Okamoto's proposal was just a more efficient approach than the naive *decrypt-and-encrypt*.

Another naive solution to this problem could be to share the original secret key with a proxy entity, so it can first decrypt the ciphertext and then re-encrypt it using the public key of the new recipient. However, this solution is clearly not suitable for all cases, as it requires a high level of trust in the proxy, enough to entrust him the secret key of the original recipient. It is clear that this solution would introduce numerous risks.

Blaze, Bleumer and Strauss proposed in 1998 the first solution to this problem [43], with the introduction of what they called "*atomic proxy cryptography*", now known as "*proxy re-encryption*" (PRE). In this solution, a semi-trusted proxy entity is granted a *re-encryption key*, a special kind of key used during the transformation of ciphertexts for Alice into ciphertexts for Bob; this transformation process is performed without the proxy being able to read the underlying message. Since then, there have been multiple proposals of proxy re-encryption schemes, based on different hardness assumptions and primitives. In this chapter we analyze the most important contributions on proxy re-encryption constructions.

From an abstract viewpoint, a proxy re-encryption scheme realizes the functionality of *delegation of decryption rights*, since it allows a user (i.e., the *delegator*) to delegate the decryption rights to another user (i.e., the *delegatee*). This functionality is extremely useful in many applications; in this chapter, we also review several proposed applications of proxy re-encryption. In particular, we are especially interested on the application of proxy re-encryption to the problem of access delegation to encrypted data, which has a natural relevance on the secure data sharing scenario, as explained in the first chapter.

3.1.1 Research Methodology

In order to perform a thorough survey on proxy re-encryption schemes and applications, we followed an ad-hoc methodology to identify and filter publications based on bibliometric criteria. A comprehensive bibliography on proxy re-encryption schemes and applications, carefully maintained by Shao [44], served as a first raw source of publications. On top of that, we manually added several relevant publications that were not included in this bibliography, which stem from our own study of the literature or through queries for relevant keywords to search engines. The result of this phase is two lists of publications, one focused on schemes (together with attacks and security analyses) and the other on applications. The list of schemes originally had 83 publications and the list of applications 69, totaling 152.

Once the list of publications was prepared, it was necessary to define a filter for the list of proxy re-encryption schemes, given the workload associated to their analysis. Although, in general, most of the papers were preliminarily studied, some of them were filtered out. We used the number of cites for each paper, as measured by Google Scholar, as an heuristic metric of the relevance of the paper. For instance, non-recent publications (e.g., before 2009) which have no cites yet, were marked as not relevant. However, manual verification of the discarded publications was required in order to rule out false negatives. The result of this phase is a collection of 58 publications, where 13 were from schemes and 45 from applications.

3.2 Preliminaries

In this section we provide the basic definitions and concepts that will serve as the basis of our analysis. This includes syntax definition, security models and relevant properties.

3.2.1 Syntax of PRE schemes

Basically, a PRE scheme has two types of functions: those that generate key material (KeyGen and $\mathsf{ReKeyGen}$), and those that deal with ciphertexts and messages (Enc, ReEnc, and Dec). Some of this functions are alike PKE functions: KeyGen produces pairs of public and secret keys, Enc generates a ciphertext that encrypts a message according to a certain public key, while Dec deciphers the ciphertext using the corresponding secret key. On top of these functions, a PRE scheme also defines functions to support the re-encryption functionality: $\mathsf{ReKeyGen}$ produces a re-encryption key between Alice and Bob, and ReEnc uses this key to transform ciphertexts originally intended for Alice into ciphertexts decryptable by Bob using his secret key.

Most of proxy re-encryption schemes comply with the diagram shown in Figure 3.1, which depicts the flow of messages, ciphertexts and keys in a PRE environment.

The following is a general definition of the syntax of a proxy re-encryption scheme, based on the ones from Canetti and Hohenberger [45] and Ateniese et al. [11]:

Definition 3.1 (PRE scheme)**.** *A proxy re-encryption scheme is a tuple of algorithms (*KeyGen*,* $\mathsf{ReKeyGen}$*,* Enc*,* ReEnc*,* Dec*):*

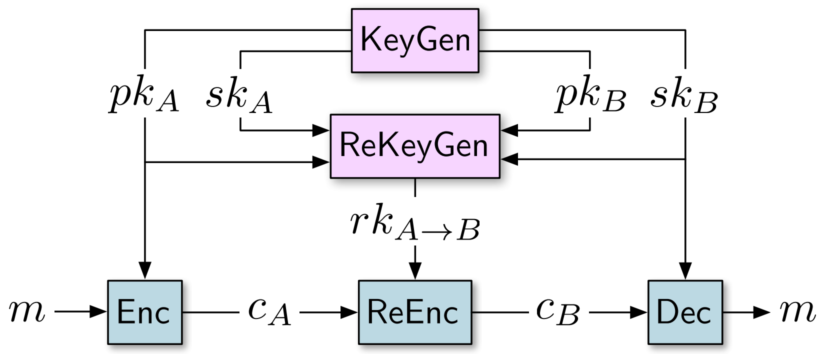


Figure 3.1: General diagram of a proxy re-encryption scheme

- $\mathsf{KeyGen}(n) \to (pk_A, sk_A)$. *On input security parameter n, the key generation algorithm* KeyGen *outputs (pk_A, sk_A), the public and secret keys for user A.*
- $\mathsf{ReKeyGen}(pk_A, sk_A, pk_B, sk_B) \to rk_{A\to B}$. *On input the pair of public and secret keys (pk_A, sk_A) for user A and the pair of public and secret keys (pk_B, sk_B) for user B, the re-encryption key generation algorithm* ReKeyGen *outputs a re-encryption key $rk_{A\to B}$.*
- $\mathsf{Enc}(pk_A, m) \to c_A$. *On input the public key pk_A and a message $m \in \mathcal{M}$, the encryption algorithm* Enc *outputs a ciphertext $c_A \in \mathcal{C}$.*
- $\mathsf{ReEnc}(rk_{A\to B}, c_A) \to c_B$. *On input a re-encryption key $rk_{A\to B}$ and a ciphertext $c_A \in \mathcal{C}$, the re-encryption algorithm* ReEnc *outputs a second ciphertext $c_B \in \mathcal{C}$ or the error symbol $\perp$ indicating c_A is invalid.*
- $\mathsf{Dec}(sk_A, c_A) \to m$. *On input the secret key sk_A and a ciphertext $c_A \in \mathcal{C}$, the decryption algorithm* Dec *outputs a message $m \in \mathcal{M}$ or the error symbol $\perp$ indicating c_A is invalid.*

The plaintext and ciphertext spaces are denoted by $\mathcal{M}$ and $\mathcal{C}$, respectively.

Note that, although this definition is wide enough, there are more general definitions of the syntax of PRE schemes, such as the one from Ateniese et al. [11], where instead single encryption and decryption algorithms, there are sets of algorithms $\overrightarrow{\mathsf{Enc}}$ and $\overrightarrow{\mathsf{Dec}}$, defined over different ciphertext spaces.

Figure 3.2 shows the relations among plaintext and ciphertext spaces for different kinds of PRE schemes, where (3.2a) represents PRE schemes with a single ciphertext space, while (3.2b) shows the case of two ciphertext spaces. Examples of PRE schemes with a single ciphertext space are [43, 45, 46, 47, 48, 49], while [11, 14, 50, 12] are schemes with two ciphertext spaces. Some schemes, such as

$$\begin{tikzcd}
\mathcal{C} \arrow[loop above, "\mathsf{ReEnc}"] \arrow[d, bend left, "\mathsf{Dec}"] \\
\mathcal{M} \arrow[u, bend left, "\mathsf{Enc}"]
\end{tikzcd}$$

(a) Single ciphertext space

$$\begin{tikzcd}
\mathcal{C}_1 \arrow[dr, bend left, "\mathsf{Dec}_1"] & & \mathcal{C}_2 \arrow[ll, bend right, "\mathsf{ReEnc}"'] \arrow[dl, bend right, "\mathsf{Dec}_2"'] \\
& \mathcal{M} \arrow[ul, bend left, "\mathsf{Enc}_1"] \arrow[ur, bend right, "\mathsf{Enc}_2"'] &
\end{tikzcd}$$

(b) Two ciphertext spaces

Figure 3.2: Transformations between plaintext and ciphertext spaces

[13], exhibit an expansive nature on re-encryption, and technically, are defined over an infinite number of ciphertext spaces, since each re-encryption induces a different space. For the sake of simplicity, we opt for the generic (and simpler) syntax with a single ciphertext space.

3.2.2 Security Models of Proxy Re-Encryption

Being proxy re-encryption an extension of public-key encryption, it is natural that the security models for PRE extend those of PKE. However, the ability to re-encrypt ciphertexts presents an interesting challenge when facing the definitions of security for PRE. On the one hand, PRE constructions have to guarantee the security objectives of the scheme, such as confidentiality and validity of ciphertexts. On the other hand, they have to allow the transformation of ciphertexts by means of re-encryption. Intuitively, both goals seem to conflict with each other.

Security Notions for PRE

Similarly to PKE, the most usual security notions in PRE are *indistinguishability against chosen-plaintext attacks* (IND-CPA) and *indistinguishability against chosen-ciphertext attacks* (IND-CCA). Both notions capture inability of an adversary to distinguish ciphertexts for known messages, and are differentiated by when the oracles are available for the adversary. These security notions are formally defined as a two-phase security game: during the first phase, the adversary can use the available oracles, constrained by some conditions; next, before the second phase starts, the adversary freely chooses two messages and receives the challenge ciphertext, which is an encryption of one of them at random; next, he can use the available oracles, again, constrained by some conditions; and finally, he has to guess which of the messages was encrypted.

The restrictions applicable during the game are what actually differentiates the security notions. When the oracles are completely restricted throughout the game, the security notion is IND-CPA, while when they are permitted, the notion is IND-CCA. It is possible to define more fine-grained notions for IND-CCA, as in PKE: if oracles are forbidden during the second phase, then one obtains *indistinguishability against non-adaptive chosen-ciphertext attacks* (IND-CCA1); if not, we are alluding to the *indistinguishability against adaptive chosen-ciphertext attacks* (IND-CCA2) notion. The IND-CCA1 notion is seldom targeted in proxy re-encryption.

For defining IND-CCA security in PRE it is necessary to forbid certain queries from the adversary. Just as in PKE the adversary cannot query the decryption oracle with the challenge ciphertext, in PRE the adversary should not be able to trivially win the game through queries to the decryption, re-encryption and re-encryption key generation oracles; however, such restrictions should allow the adversary to still query the decryption and re-encryption oracles with any ciphertext which is not derived from the challenge ciphertext. The restrictions to the oracles in PRE are usually defined using the concept of *derivatives of the challenge ciphertext*, first proposed by Canetti and Hohenberger in [45], which captures the idea of ciphertexts that are connected to the challenge by means of oracle queries. See Section 4.2 in the next chapter for more details about the concept of derivatives, and for a detailed description of possible attack models, parametrized as a function of the availability of the decryption and re-encryption oracles.

An interesting and orthogonal concept to the aforementioned security notions is *Replayable CCA* (RCCA) originally defined for PKE [51]. This is a weaker form of IND-CCA where the adversary is able to make innocuous modifications to ciphertexts, as long as the original message is not altered. This notion naturally fits in the context of PRE, since the goal of PRE is to transform ciphertexts from one user to another, without changing the message.

Assumptions

The security models for PRE are also shaped by some additional assumptions. Perhaps the more important of these assumptions in the PRE context is the *corruption model*. In a *static corruption model*, the adversary must decide in advance whether to *corrupt* a user or not before asking for the generation of the user's keypair; with the term corruption we are referring to the adversary knowing the secret keys of the user. In contrast, in an *adaptive corruption model*, the adversary is free to corrupt users in any moment.

Another interesting assumption is related to how the adversary obtains keys. In the *knowledge of secret key model*, the challenger generates the key material of all users, while in the *chosen key model* the adversary can adaptively choose public keys for malicious users [12]. See [52] for more insights about these models.

As in PKE, the vast majority of proxy re-encryption schemes are examples of provable security. In some cases, the proofs of the security are given in the *random oracle model*, where hash functions are assumed to behave as random oracles, an idealization where the hash function is deterministic but its output is uniformly distributed at random in its image domain. When this assumption is not present, we say that we are in the *standard model*.

The security of provable-secure schemes is defined in terms of reductions to hard problems. In other words, the schemes are proven secure, assuming that certain problem is hard. There is a multitude of hardness assumptions, some of them more prominent than others. The most usual are the Computational and Decisional Diffie-Hellman (CDH and DDH) problems in the case of generic groups, the Bilinear Decisional Diffie-Hellman (DBDH) problem in the case of groups with bilinear pairings, and the Learning With Errors (LWE) problem in the case of lattice-based schemes.

3.2.3 Properties

In this section we review the main properties associated to proxy re-encryption schemes. Some of these properties are related to the ability of an adversary (including the proxy and the delegatees) for deriving key material. We will use the notation introduced by Wang et al. in [53] for describing some of these properties in a more formal manner. We will denote by $X + Y \rightarrow Z$ if element Z can be derived from elements X and Y; public information, such as the public parameters and the public keys, is intrinsically assumed. The converse case will be denoted by $X + Y \nrightarrow Z$.

Directionality

This property is associated to direction of the delegation, and is embodied by the re-encryption key. The delegation can be either *unidirectional* or *bidirectional*.

Unidirectional delegation means that decryption rights are delegated only from delegator to delegatee, and not in the other direction. Using the notation described above, we say that a PRE scheme is *unidirectional* when the following applies:

$$rk_{A\to B} \nrightarrow rk_{B\to A}$$

Otherwise, the scheme is bidirectional, so it is possible to compute $rk_{B\to A}$ from $rk_{A\to B}$. The fact that a PRE scheme is bidirectional is not necessarily negative, but depending on the situation, it may not be a desirable characteristic. Since PRE can be seen as a mechanism for delegation of decryption rights, bidirectional schemes would be appropriate for scenarios where the trust relationship between delegators and delegatees is symmetric. However, this situation is not common for most applications.

For bidirectional schemes, there is a recurring pattern that appears in some of them, related to how the re-encryption keys are constructed. Informally, these re-encryption keys contain a combination of the delegatee's secret key and the inverse of the delegator's secret key, so when they are used during re-encryption, the delegator's secret key in the ciphertext can be substituted by the delegatee's. More formally, this pattern appears in schemes where public keys are a function of the secret keys, and the secret key space and re-encryption key space are the same. Let us assume that the underlying structure of the secret key space has a group structure, where $*$ is the operator. Then, in this pattern, re-encryption keys are of the form $rk_{A\to B} = sk_A^{-1} * sk_B$, where sk_A^{-1} is the inverse of sk_A for the $*$ operator. Therefore, it is clear that when this pattern emerges, the converse re-encryption key can be easily computed as $rk_{B\to A} = rk_{A\to B}^{-1}$, so the scheme is bidirectional. A second result is that it is possible that a collusion between the proxy and one of the participants extracts the secret key of the other (e.g., $sk_A = rk_{A\to B}^{-1} * sk_B$). We refer to this pattern as the "*BBS pattern*", as the BBS98 scheme [43] was the first example. Other notable examples are the schemes from Canetti and Hohenberger [45], Weng et al. [54] and Xagawa and Tanaka [46]. Usually, the underlying group is multiplicative, although, for example, the scheme from Xagawa and Tanaka uses an additive group. For this reason, we preferred to describe this pattern with the generic operator $*$.

Number of Uses

We say a PRE scheme is *single-use* if ciphertexts are re-encryptable just once. This characteristic is usually associated to schemes with multiple ciphertext spaces, since this way the re-encryption can be constructed as a one-way transformations between ciphertext spaces, e.g., by means of pairings.

On the contrary, if ciphertexts are re-encryptable multiple times, the PRE scheme is said to be *multi-use.* This characteristic is usually associated to schemes with a single ciphertext space. We further classify multi-use schemes in three types according to the way they achieve this property:

- *True multi-use schemes*: The main characteristic of this type of schemes is that the re-encryption function does not alter the form of the ciphertexts. That is, re-encrypted ciphertexts are identical in shape to original ciphertexts, except maybe for the random elements. For this reason, re-encryption preserves the size of ciphertexts and the running time of the decryption does not depend on the number of re-encryptions of the ciphertext. Examples of this type of multi-use schemes are [43] and [45]. As an illustration, let us study the re-encryption procedure in [43]: the encryption of a message m for the user A is of the form $c_A = (mg^k, g^{ak})$, being k a random secret, and the re-encryption key for user A to B is computed as $rk_{A\to B} = \frac{b}{a}$, where a and b are the secret keys of users A and B respectively. The re-encryption function simply exponentiates the second element of the ciphertext to the re-encryption key, so the re-encrypted ciphertext is $c_B = (c_{A,1}, c_{A,2}^{rk_{A\to B}}) = (mg^k, (g^a)^{\frac{b}{a}}) = (mg^k, g^{bk})$.

- *Expansive multi-use schemes*: Some multi-use schemes are based on an iterative method of key encapsulation. The idea is that a random secret, generated for each re-encryption key, acts as trapdoor for a trapdoor function applied to the random secret of the previous re-encryption; the current random secret is encrypted with the encryption function of the PRE scheme, so it can be further re-encrypted. Thus, the ciphertext contains a sort of chain of random secrets for each re-encryption, scrambled using trapdoor functions. The result of this procedure is that the ciphertext size grows linearly with the number of re-encryptions, and, as a consequence, the cost of decryption depends on the number of previous decryptions. A general method for constructing expansive multi-use schemes, depicted in Figure 3.3, is as follows: Let g_T be a trapdoor function with trapdoor T and $x_1, ..., x_N$ random secrets, then the general form of a ciphertext re-encrypted N times will be:

$$c_N = (g_{x_1}(m), g_{x_2}(x_1), ..., g_{x_N}(x_{N-1}), \mathsf{Enc}(pk_N, x_N))$$

  Re-encryption keys are of the form $rk_{i\to j} = (f(sk_i, x_j), \mathsf{Enc}(pk_j, x_j))$, where f_T is also a trapdoor function. The challenge in designing a scheme of this kind is to define the re-encryption function so the combination of $\mathsf{Enc}(pk_i, x_i)$ and $f_{sk_i}(x_j))$ results in $g_{x_j}(x_i)$, for all x_i, x_j, and without the proxy learning anything.

$$\begin{tikzcd}
c_0 = \mathsf{Enc}(pk_0, m) \arrow[dd, dashed] & \\
& rk_{0\to 1} = (f_{sk_0}(x_1), \mathsf{Enc}(pk_1, x_1)) \arrow[dl, dashed] \arrow[dl] \\
c_1 = (g_{x_1}(m), \mathsf{Enc}(pk_1, x_1)) \arrow[dd, dashed] & \\
& rk_{1\to 2} = (f_{sk_1}(x_2), \mathsf{Enc}(pk_2, x_2)) \arrow[dl, dashed] \arrow[dl] \\
c_2 = (g_{x_1}(m), g_{x_2}(x_1), \mathsf{Enc}(pk_2, x_2)) &
\end{tikzcd}$$

Figure 3.3: Re-encryption process in expansive multi-use PRE schemes

- *Limited multi-use schemes*: Whereas the previous kinds of multiuse schemes support an indefinite number of re-encryptions, some recent proxy re-encryption schemes, in particular those based on lattices [46, 47, 48, 49], present a limited version of the multi-use property, since the re-encryption function introduces noise to the ciphertext. For this reason, and depending on the parameters used, the accumulated noise makes the decryption procedure to fail after a certain number of re-encryptions. This may even happen after just one re-encryption. Thus, schemes of this type are in an intermediate area between single-use and multi-use. For instance, the scheme we propose in Section 5.1 is of this type, as the number of possible re-encryptions varies with the parameters used; in particular, the average number of re-encryptions that is supported varies from 5 to 50. Another interesting example is the scheme from Kirshanova [48], which is allegedly single-use, although that would ultimately depend on the choice of parameters.

Collusion-safeness

This property conveys the safeness of the delegator's secret key against collusion attacks made by the delegatee and the proxy; that is, delegator's secret key cannot be derived from the re-encryption key and the delegatee's secret key:

$$rk_{A\to B} + sk_B \nrightarrow sk_A$$

However, this property may not be sufficient for some purposes. In most cases, such a collusion does reveal a *weak secret* associated to the delegator's secret key sk_A. In some schemes, such as [11] and [12], this weak secret can be used to create new re-encryption keys or to decrypt re-encryptable ciphertexts; to a certain extent, the weak secret key enables to achieve the functionality of the re-encryption key generation function and the decryption function.

Transitivity

A PRE scheme is said to be transitive if the proxy alone is able to re-delegate decryption rights. That is, it can combine re-encryption keys to produce new ones. More formally:

$$rk_{A \to B} + rk_{B \to C} \to rk_{A \to C}$$

As in the bidirectional case, transitiveness is not negative per se; it depends heavily on the scenario where the scheme is applied. However, transitive delegation is troublesome in general, because it is difficult for the original delegator to foresee the potential delegations which could occur.

Schemes that present the BBS pattern are also transitive. This is easy to check: if $rk_{A \to B} = sk_A^{-1} \cdot sk_B$ and $rk_{B \to C} = sk_B^{-1} \cdot sk_C$, then $rk_{A \to C} = rk_{A \to B} \cdot rk_{B \to C}$.

Interactivity

Recall that the general syntax of PRE presented in Section 3.2.1 included the secret key of the delegatee sk_B in the re-encryption key generation function. If this key is not needed, then the scheme is *not interactive*, since re-encryption keys for delegatees can be produced without their participation; that is, the delegator is able to create a re-encryption key using his own secret key and the delegatee's public key. More formally:

$$sk_A \to rk_{A \to X}, \forall X$$

On the contrary, an interactive scheme implies the participation of the delegatee; in most cases interactivity is an undesired property, as it introduces a communication overhead, and more worryingly, may be vulnerable to collusion-attacks for extracting the secret keys involved. However, interactive schemes can be used to implement *delegation acceptance* by the delegatee [11]; that is, re-encryption keys cannot be generated without the consent and participation of the delegatee.

It is worth mentioning, that the BBS pattern introduced before produces interactive schemes, since the re-encryption key generation necessarily uses the secret keys of both users. Canetti and Hohenberger describe in [45] a simple secure three-party protocol for computing the re-encryption key that involves the proxy and both users. The protocol is as follows:

1. A selects a random blinding factor r from the secret keys' space, and sends $r * sk_A^{-1}$ to B and r to the proxy.
2. B computes $r * sk_A^{-1} * sk_B$ and sends it to the proxy.
3. The proxy computes $rk_{A\to B} = (r * sk_A^{-1} * sk_B) * r^{-1}$.

The final outcome of the protocol is that the proxy knows the re-encryption key but no secret keys, while the users do not reveal their secret keys to each other.

Other properties

- Temporary: Some schemes take the temporal dimension into consideration, so the delegation of decryption rights is only valid for a specific period of time. This property was first introduced by Ateniese et al. in [11].
- Conditional: It is also possible to define keywords that restrict the re-encryption functionality, in a conditional vein. Therefore, conditional PRE represents a fine-grained generalization of traditional PRE. In conditional PRE, re-encryption keys are associated to a certain keyword, so the proxy is only capable of re-encrypting ciphertexts that are tagged with that keyword. This notion was introduced by Weng et al. in [55].
- Non-transferability: This property, first considered by Ateniese et al. in [11], captures the idea of the inability of a collusion of proxy and delegatees to re-delegate decryption rights. More formally:

$$rk_{A\to B} + sk_B \nrightarrow rk_{A\to X}, \forall X$$

- Proxy invisibility: A PRE scheme is said to be proxy-invisible if a delegatee is unable to distinguish a ciphertext computed under her public key from a re-encrypted ciphertext, originally encrypted under another public key [11]. That is, the proxy is “invisible”, in the sense that the delegatee cannot discern whether the proxy has transformed the ciphertexts or not.
- Perfect Key-Switching: A stronger property than proxy invisibility is perfect key-switching, which we propose in Section 5.2. Informally, a PRE scheme satisfies this property when the re-encryption cleanly switches one public key for another, without altering the format of the ciphertext nor

the random coins used; that is, the key-switching that takes places during re-encryption is perfect. Examples of this type of scheme are the BBS98 [43] and CH07 [45] schemes. It is clear that perfect key-switching implies proxy invisibility: a re-encrypted ciphertext has exactly the same form as a ciphertext originally encrypted with the delegatee's public key. However, the converse (i.e., proxy invisibility implies perfect key-switching) is not necessarily true, since a proxy-invisible PRE scheme may alter the original randomness during the re-encryption (e.g., AFGH06a [11]).

3.3 Analysis of Proxy Re-Encryption Schemes

In this section we review and analyze the main proxy re-encryption schemes, which result from the bibliometric process presented in Section 3.1.1. A total of 13 publications were selected, and since some of them proposed several schemes, the total number of analyzed schemes is 19. We only considered those proposed schemes which were accompanied by a proof of security; in some cases, variants of the proposals are sketched, but without presenting a demonstration of security and correctness, or even a full description of the construction.

The goal of this analysis is to study the characteristics of each of these schemes, taking in consideration the concepts presented in the previous section, without explaining in detail the constructions from the cryptographic point of view. The review makes a comparative analysis possible, which is described subsequently. Finally, we also present a performance analysis of a selection of these schemes, both theoretically and empirically.

3.3.1 Review of Proxy Re-Encryption Schemes

An early notion, reminiscent of proxy re-encryption, was presented in 1997 by Mambo and Okamoto [42], although their proposal implied that the original recipient must be available for re-encrypting ciphertexts when needed, which is not always feasible. Blaze, Bleumer and Strauss proposed in 1998 the first proxy re-encryption scheme [43], which complies with the established notion of proxy re-encryption. Since then numerous schemes have been proposed. In this section we review a selection of these schemes.

In order to properly identify schemes, each scheme was labeled with the author's initials and year of publication, and if necessary, an additional alphabetic index for differentiating schemes within the same publication (e.g., AFGH06a).

BBS98 scheme

This scheme was the first to show an actual construction that complied with the notion of proxy re-encryption. Given the simplicity and elegance of the construction, we describe it here for illustration purposes. This scheme is based on ElGamal and works in a multiplicative cyclic group $\mathbb{G} \subset \mathbb{Z}_p$ of prime order q, where $p = 2q + 1$ is a prime, and g a generator of $\mathbb{G}$. Alice generates a random secret key $sk_A = a \in \mathbb{Z}_q$ and publish her public keys $pk_A = g^a \in \mathbb{G}$. Bob proceeds analogously for generating his key pair. In order to send an encrypted message m to Alice, one samples a random integer $r \in \mathbb{Z}_q$ and uses her public key pk_A to compute the following pair:

$$c_A = (pk_A^r, m \cdot g^r) = (g^{ar}, m \cdot g^r)$$

Since $\mathbb{G} \subset \mathbb{Z}_p$, all operations are performed modulo p. The re-encryption key is computed using the secret keys of the users, which implies that the scheme is interactive, as follows:

$$rk_{A \to B} = \frac{sk_B}{sk_A} = \frac{b}{a} \in \mathbb{Z}_q$$

Once the proxy has the re-encryption key, it is able to transform a ciphertext c_A, intended for Alice, in a ciphertext c_B, whose recipient is Bob; in order to do so, the proxy takes $c_A = (c_{A,1}, c_{A,2})$, and computes:

$$c_B = (c_{A,1}^{rk_{A \to B}}, c_{A,2}) = ((g^{ar})^{\frac{b}{a}}, m \cdot g^r) = (g^{br}, m \cdot g^r)$$

It can be seen that the re-encryption process simply switches one key for another, cleanly, making this scheme an example of the perfect key-switching property. Therefore, the decryption process is exactly the same for all the ciphertexts. In order to retrieve the original message, the recipient (e.g., Bob) uses his secret key to compute:

$$m = \frac{c_{B,2}}{(c_{B,1})^{\frac{1}{sk_B}}}$$

This scheme is multi-use and proven CPA-secure in the standard model. It is the prime representative of the pattern for bidirectional schemes presented in Section 3.2.3 (i.e., the BBS pattern). Therefore, it is also interactive, transitive and not resistant to collusions.

AFGH06 schemes

Ateniese, Fu, Green and Hohenberger proposed in [56] and [11] new proxy re-encryption schemes based on bilinear pairings. These schemes were the first to present the idea of multiple ciphertext spaces, as shown in Figure 3.2b. Original encryptions are referred as “second-level ciphertexts”, while re-encrypted ciphertexts are “first-level ciphertexts”. Hence, the re-encryption function is a transformation between the second-level ciphertext space and the first-level one.

Their first scheme, AFGH06a, is unidirectional, single-use, not interactive, not transitive, and proxy invisible; it is also collusion-safe, although colluding adversaries may compute the weak secret g^{a_1} of the delegator that allows to decrypt second-level ciphertexts and create re-encryption keys. The second scheme, AFGH06b, is a temporary variation of the previous one, where the validity of re-encryption keys is bounded to a specific time period. It is similar to AFGH06a, but it introduces a trusted third party that broadcasts a random value associated to each time period. Thus, this scheme permits the revocation of all previous delegations just by making a change in a global parameter (i.e., the current time period). However, in order to generate re-encryption keys, the delegator needs that the delegatee compute and publish a *delegation acceptance value*, which makes this scheme *interactive*. Moreover, the re-encryption key generation must occur in the same time period (or before) than the encryption process, which limits the flexibility of the scheme. Both schemes, AFGH06a and AFGH06b, are proven CPA-secure in the standard model. The authors also propose several applications of proxy re-encryption, and in particular they implement an access control server for a secure file system using PRE. Additionally, the authors provide the first implementation [57] and performance measurements of a proxy re-encryption scheme.

GA07 schemes

Green and Ateniese proposed in [13] the first identity-based proxy re-encryption schemes (IB-PRE). Being identity-based, these schemes use the identities of the delegator and delegatees as their public keys. The authors present two IB-PRE schemes, which we will refer as GA07a and GA07b. Their first scheme, GA07a, is unidirectional and offers multi-use capabilities, at the expense of being only CPA-secure in the random oracle model. The scheme is also non-interactive, and non-transitive. However, this scheme is not collusion-safe, as the proxy and the delegatee can collude and pool their keys to obtain the secret key of the delegator. This scheme is based in the Boneh-Franklin IBE scheme [58], and, in principal, can reuse an existing deployment, as it uses the same type of parameters and keys. The multi-use property is achieved using an expansive construction, so the ciphertexts grow on each re-encryption. This scheme was the first to show this type of construction.

Their second scheme, GA07b is single-use, and is allegedly CCA-secure in the random oracle model, although Koo et al. present in [59] an attack that uses the re-encryption oracle during the second phase of the security game, so we will consider it as CCA1-secure. This scheme is based on the Gentry-Silverberg HIBE scheme [60]. It is also important to note that, in the same vein as most IBE-based schemes, these schemes require a Key Generation Center that issues a private key for each identity and that must maintain a master secret key. Both schemes are also pairing-based.

CH07 scheme

Canetti and Hohenberger present in [45] a CCA-secure bidirectional scheme in the standard model. They initially construct a CCA-secure scheme in the random oracle model, and modify it for the standard model. We will only consider the latter version, since the modifications are minimal. The CH07 scheme presents the BBS pattern, so it is also interactive, transitive, and not resistant to collusions. CH07 introduces CCA-security by integrating a one-time signature into the ciphertexts, following the CHK paradigm [51]. Informally, the solution proposed by the authors is to sign a portion of the ciphertext, which remains unaffected by the re-encryption; otherwise, the signature is invalidated. The remaining part of the ciphertext is what actually changes during re-encryption. In order to validate this part, the signed portion includes some extra information that permits to check that the re-encrypted part has only changed the underlying public key. Another main contribution of the authors is several definitions of CCA-security for bidirectional proxy re-encryption schemes, both game- and simulation-based.

CT07 schemes

Chu and Tzeng presented in [14] two IB-PRE schemes, built upon Waters IBE construction [61]. In fact, their security proofs are reductions to the security of the Waters IBE scheme, which in turn is secure under the DBDH assumption. Similarly to the GA07a scheme, the proposed schemes follow the expansive multi-use construction. The first scheme, CT07a, is CPA-secure in the standard model, unidirectional and not interactive, but not collusion-resistant, as shown by [12]. The second scheme, CT07b, is reported CCA-secure, but Shao and Cao show in [62] that anyone is able to re-randomize ciphertexts, which makes the scheme secure in a weaker notion, namely RCCA; this scheme is also not resistant to collusions.

Mat07 scheme

Matsuo presented in [63] two IB-PRE schemes. The first one is a rather peculiar proposal, called *"hybrid proxy re-encryption"*, where ciphertexts encrypted using a public key encryption scheme can be re-encrypted to ciphertexts under an identity-based encryption scheme; due to this unusual nature, we will not analyze it here. The second one, Mat07, is an identity-based proxy re-encryption scheme. A significant characteristic of this scheme is that it introduces a new entity called "Re-Encryption Key Generator" that is in charge of producing re-encryption keys and that receives a copy of the master secret key; in principle, the Private Key Generator could also take this role. Thus, in this setting the original delegator is deprived of the re-encryption key generation capabilities, which are now taken by the Re-Encryption Key Generator. The scheme is unidirectional, single-use, collusion-resistant and is proven CPA-secure on the standard model. It is also interactive, as it requires the secret key of the delegatee, and transitive, since it is trivial to compute the re-encryption key $rk_{ID \rightarrow ID''}$ from $rk_{ID \rightarrow ID'}$ and $rk_{ID' \rightarrow ID''}$.

ABH09 scheme

Another interesting proposal is presented in [64], where the authors define the notion of *key privacy* in the context of proxy re-encryption, which prevents the proxy to derive the identities of both sender and receiver from a re-encryption key. Their scheme, ABH09, is constructed with bilinear pairings and is proven CPA-secure in the standard model. The scheme is unidirectional, single-hop, resistant to collusions, not interactive and not transitive.

The key privacy property is proven by means of a specific security game defined by the authors, which serves as a reference to other key-private PRE schemes. Additionally, they also present two necessary conditions for a PRE scheme to be key private. The first condition is that the re-encryption function should not be deterministic. The second condition is that a restricted PKE-version of the scheme should also be key private, as defined in [65] for PKE.

WDLC10 scheme

Weng et al. proposed in [54] two bidirectional schemes without pairings, both CCA-secure in the random oracle model under the hardness of the Computational Diffie-Hellman (CDH) problem. Unlike most of previous proposals, these schemes are not based in bilinear pairing operations, which makes them, in principle, more efficient. These schemes achieve CCA-security by integrating Schnorr signatures [66] with a PRE-version of hashed ElGamal encryption scheme. The first scheme, WDLC10a, follows the BBS pattern for bidirectional schemes. Therefore, it is interactive, transitive and not resistant to collusions. In contrast to other schemes that follow this pattern, WDLC10a is single-use. The second scheme, WDLC10b, is very similar to the previous one, but produces re-encryption keys in a different way in order to be non-transitive.

LV11 schemes

Libert and Vergnaud proposed, first in [67] and later on in [12], several unidirectional schemes with chosen-ciphertext security in the standard model. Specifically, they achieve the RCCA notion, which is a relaxed version of CCA security. The first scheme, LV11a, is similar to AFGH06a, but takes ideas from CH07, such as the use of one-time signatures. It is unidirectional, single-use, collusion-resistant, not interactive and not transitive. Since it is defined in the RCCA model, it is possible to re-randomize ciphertexts. Seo et al. [68] detected an error in one of their security proofs, in which the adversary was able to distinguish the simulation from a real attack, and propose a way to amend it. The second scheme, LV11b, is a temporal version of the previous one; as opposed to the temporal scheme from Ateniese, AFGH06b, this scheme is not interactive. Additionally, it is the first PRE scheme that considers the chosen-key model. In addition to these schemes, Libert and Vergnaud also propose several variations of their schemes, without providing extensive descriptions and proofs. In particular, they show how the temporal scheme can be extended to allow temporal windows, instead of just single periods of time; they also introduce a conditional version of their scheme.

XT10 scheme

Xagawa and Tanaka presented in [46] the first proxy re-encryption scheme based on lattices. In particular, this scheme is based on the Learning With Errors (LWE) problem [32]. This scheme is reminiscent to the BBS98 scheme, but adapted to a lattice-based setting. It presents the same BBS pattern, although represented additively, so $rkAB = sk_B - sk_A$. Therefore, this scheme is bidirectional, interactive, transitive and not resistant to collusions. The scheme is also multi-use, more specifically, of the limited multi-use type.

ABPW13 scheme

Aono et al. proposed in [47] a lattice-based encryption scheme, ABPW13, which is proven CPA-secure in the standard model. This scheme is based upon a lattice cryptosystem from Lindner and Peikert [33], whose hardness relies on the LWE problem. ABPW13 is unidirectional, interactive, not transitive and limited multi-use. The scheme is not resistant to collusions from the proxy and the delegatee, as shown recently by [69]. The authors also state that ABPW13 is key-private; however, Nishimaki and Xagawa recently noted in [70] that, although the scheme ensures the anonymity of the delegator, the delegatee is exposed trivially, since its public key is contained in the re-encryption key, achieving then a partial form of key-privacy. A strong contribution from the authors is the estimation of security parameters for achieving desired levels of security. This estimation was done by approximating the cost of an exhaustive search attack to LWE.

On top of this scheme, the authors construct a "CCA-secure" version in the random oracle model, using the generic conversion from Fujisaki and Okamoto [71]. However, this version is flawed because it does not perform the validation step during decryption of re-encrypted ciphertexts. If one forces this check, decryption will always fail in the case of re-encryption, breaking the correctness of the scheme.

Kir14 scheme

Kirshanova presented in [48] a lattice-based PRE scheme, which was reported to be CCA1-secure. The scheme is based on the CCA1-secure PKE scheme from Micciancio and Peikert [40], and its security is associated to the hardness of lattice problems, namely, the Learning With Errors and Short Integer Solution (SIS) [38] problems. Basically, Kirshanova extended the original PKE scheme to support re-encryptions, using the technique presented in [40] for trapdoor delegation. The scheme Kir14 is unidirectional, not interactive and resistant to collusions. The authors state that the scheme is single-use, although technically we consider it of the limited multiuse type, as the choice of parameters is what determines the number of possible re-encryptions. The authors prove on what conditions the re-encryption process preserves the correctness, but this proof only considers one hop; still, it would be possible that some sets of parameters permit multiple re-encryptions. We show in Section 4.4 an attack to this scheme in the CCA1 setting, and prove it actually satisfies a weaker notion of CCA1 where no re-encryption oracle is provided to the adversary. The attack makes use of a property first introduced by Canetti and Hohenberger in [45], called *privacy* *of re-encryption keys*, which captures the notion of an adversary being unable to extract re-encryption keys from the knowledge of original and re-encrypted ciphertexts. We refined this property and show its relation to queries to the re-encryption oracle. We show then how the Kir14 scheme leaks re-encryption keys when the re-encryption oracle is present, which is implied by the CCA1 security notion.

NAL15 schemes

For the sake of completeness, we also include in this analysis two proposals contained in this thesis (see Section 5.1). These proposals are based on the NTRU cryptosystem [16]. The first scheme, NAL15a, is originally named "NTRUReEncrypt" since it is a PRE-version of the original NTRUEncrypt scheme. NAL15a is bidirectional, multihop, and interactive, but not collusion-resistant, as it follows the usual BBS pattern. As the underlying NTRU scheme, the underlying structure of this scheme is a polynomial ring, so the operations performed are simply additions and multiplications of polynomials, which can be computed very efficiently, and can even be parallelized. Therefore, the key strength of this scheme is its performance. Experimental results show that this scheme outperforms others by an order of magnitude, in a similar way than NTRU does with respect to other PKE schemes. However, the original NTRU scheme does not have a proof of security, in the provable sense, and its security (and therefore of NAL15a) is only conjectured, which has been called the "NTRU assumption" [72]. The parameters for NTRU are then computed with regard to its resistance to attacks. To overcome this problem, we also propose a provably-secure variant that is proven CPA-secure under the hardness of the Ring-LWE problem, a variation of the LWE problem under a polynomial ring; this scheme is an extension of a provable-secure version of NTRU proposed by Stehlé and Steinfeld [73]. This second PRE scheme, NAL15b, is identical to the previous one with respect to the properties.

3.3.2 Summary and Comparison

We now summarize and compare the analyzed schemes in a single table. Table 3.1 shows this comparison, according to the following criteria:

- Directionality: A single arrow ($\rightarrow$) is used to represent a unidirectional scheme, whereas a double arrow ($\leftrightarrow$) denotes a bidirectional one.

- Use: Single-use schemes are represented by the letter 'S', true multi-use schemes by the initials 'TM', expansive multi-use schemes by 'EM', and limited multi-use schemes by 'LM'.
- Number of ciphertext spaces (#spaces): This column specifies the number of ciphertext spaces on which the scheme is defined.
- Security: The achieved notion of security (e.g., CPA, CCA, RCCA) is presented, as well as whether is based or not in the random oracle model (RO or SM).
- The "Based on" column describes the underlying structure on which the scheme is constructed. The possible values are "Group" for generic groups, "Pairing" for groups with bilinear pairings, and "Lattice" for schemes that are lattice-based. Additionally, the hardness assumption is also presented.
- Collusion resistance: Schemes that are resistant to collusions are marked with ✓, and with × otherwise.
- Non-transitive: Schemes that are non-transitive are marked with ✓, and with × otherwise.
- Non-interactive: Schemes that are non-interactive are marked with ✓, and with × otherwise.
- The last column shows additional relevant characteristics of the analyzed schemes

3.3.3 Efficiency

This section is devoted to analyzing the performance of several PRE schemes. This analysis will be made from two points of view: theoretical and empirical. The latter kind of study is is seldom tackled in the literature.

As described in the general syntax of PRE, there are several functions in a PRE scheme. However, from a functional point of view, we are mainly interested in studying the computational costs for three of them: encryption of messages intended to be re-encrypted, re-encryption of ciphertexts, and decryption of re-encrypted ciphertexts. This decision is justified by the actual use of proxy re-encryption in applications, where ciphertexts are meant to be re-encrypted. Therefore, functions such as non-delegatable encryption (denoted usually by "first-level encryption") are out of the scope of this analysis, since could be achieve by means of traditional encryption schemes.

Table 3.1: Comparison of PRE schemes

Scheme	Dir.	Use	#spaces	Security	Based on	Coll. res.	Non-trans.	Non-int.	Other characteristics
BBS98 [43]	↔	TM	1	CPA (SM)	Group (DDH)	×	×	×	Perfect key-switching
AFGH06a [11]	→	S	2	CPA (SM)	Pairing (eDBDH)	✓	✓	✓	Proxy invisible
AFGH06b [11]	→	S	2	CPA (SM)	Pairing (DBDH)	✓	×	×	Temporary
CH07 [45]	↔	TM	1	CCA (SM)	Pairing (DBDH)	×	×	×	Perfect key-switching
GA07a [13]	→	EM	N	CPA (RO)	Pairing (DBDH)	×	✓	✓	Identity-based
GA07b [13]	→	S	2	CCA1 (RO)	Pairing (DBDH)	×	✓	✓	Identity-based
CT07a [14]	→	EM	N	CPA (SM)	Pairing (DBDH)	×	✓	✓	Identity-based
CT07b [14]	→	EM	N	RCCA (SM)	Pairing (DBDH)	×	✓	✓	Identity-based
Mat07 [63]	→	S	1	CPA (SM)	Pairing (DBDH)	✓	×	×	Identity-based
ABH09 [64]	→	S	2	CPA (SM)	Pairing (eDBDH)	✓	✓	✓	Key-private
WDLC10a [54]	↔	S	2	CCA (RO)	Group (CDH)	×	×	×	
WDLC10b [54]	↔	S	2	CCA (RO)	Group (CDH)	×	✓	×	
LV11a [12]	→	S	2	RCCA (SM)	Pairing (3w-DBDHI)	✓	✓	✓	
LV11b [12]	→	S	2	RCCA (SM)	Pairing (1w-DBDHI)	✓	✓	✓	Temporary, Chosen-Key model
XT10 [46]	↔	LM	1	CPA (SM)	Lattice (LWE)	×	×	×	
ABPW13 [47]	→	LM	1	CPA (SM)	Lattice (LWE)	×	✓	×	Partial Key-Private
Kir14 [48]	→	LM	1	CCA1* (SM)	Lattice (LWE)	✓	×	✓	
NAL15a [49]	↔	LM	1	-	Lattice (NTRU)	×	×	×	
NAL15b [49]	↔	LM	1	CPA (SM)	Lattice (Ring-LWE)	×	×	×	

Theoretical analysis

In this section we analyze and compare schemes from the theoretical standpoint. We omit from our analysis the schemes that are lattice-based, since their underlying structure varies greatly, the theoretical analysis of costs is very intricate and the results are not directly comparable. Therefore, the underlying structure of the analyzed schemes is groups, either generic ones or pairing-based.

Usually, computational costs are analyzed in terms of the main operations performed, which in this case are the exponentiation and the pairing. These costs are denoted by t_e and t_p, respectively. It is important to note that, in the case of pairing groups, the computational costs of operations are different depending on the group where are performed; in fact, operations in $\mathbb{G}$ are more expensive than in $\mathbb{G}_T$. For this reason, when necessary we will make the distinction between operations in $\mathbb{G}$ (denoted by t_e) and in $\mathbb{G}_T$ (denoted by t_{e_T}). Some of the schemes make use of a one-time signatures (OTS) for reaching CCA-security. In this case, the signature of a message using OTS, including key pair generation, is denoted by t_S, while the verification of a signature is denoted by t_V. Finally, the decryption of expansive multi-use schemes varies with the number N of re-encryptions.

The results of this analysis are presented in Table 3.2. It is interesting to see the great difference in performance between schemes. For instance, the BBS98 schemes only needs a few exponentiations, as opposed to more complex schemes, such as LV11a and CH07, which require up to five pairing operations for the decryption. This is due to the additional costs incurred by the achievement of CCA-security. Notable schemes are WDLC10a and WDLC10b, since they provide CCA-security but only require a few exponentiations. Note also how in the case of expansive multi-use schemes, the cost of decryption depends on the number of re-encryptions.

With regard to space costs, these are mainly driven by the size of group elements. Note also that, in the case of pairing groups, $\mathbb{G}_T$ admits shorter representations than $\mathbb{G}$. As in the case of computational costs, some schemes make use of OTS; the size of a signature is denoted by $|\sigma|$, while the size of a verification key is $|svk|$. The cost of ciphertexts in some schemes also depends on the size of the original message, $|m|$. Finally, in some cases, elements of $\mathbb{Z}_q$ are also used. Table 3.3 shows the results of this analysis.

It can be seen how expansive schemes increase the size of ciphertext linearly on each re-encryption. Similarly to the analysis of computational costs, CPA-schemes usually have lower size of ciphertexts, since CCA-schemes need to include additional elements for the validation of the ciphertexts (e.g., signatures).

Table 3.2: Computational costs of selected PRE schemes

Scheme	Encryption	Re-encryption	Decryption
BBS98	$2t_e$	t_e	t_e
AFGH06a	$t_e + t_{e_T}$	t_p	t_{e_T}
AFGH06b	$t_p + t_e + t_{e_T}$	t_p	t_{e_T}
CH07	$t_p + 4t_e + t_{e_T} + t_S$	$4t_p + 2t_e + t_V$	$5t_p + 2t_e + t_{e_T} + t_V$
GA07a	$t_p + 2t_e$	t_p	Nt_p
GA07b	$t_p + 3t_e$	$4t_p + 2t_e$	$2t_p + t_e$
CT07a	$2t_e + t_{e_T}$	$2t_p$	$(N+2)t_p$
CT07b	$3t_e + t_{e_T} + t_S$	$2t_e + t_V$	$(7N+3)t_p + 2t_e + (N+1)t_V$
Mat07	$t_p + 3t_e + t_{e_T}$	$t_p + t_{e_T}$	$2t_p$
ABH09	$2t_e + t_{e_T}$	$4t_p + 2t_{e_T}$	t_{e_T}
WDLC10a	$3t_e$	$3t_e$	$2t_e$
WDLC10b	$3t_e$	$3t_e$	$2t_e$
LV11a	$3t_e + t_{e_T} + t_S$	$2t_p + 4t_e + t_V$	$5t_p + t_e + t_{e_T} + t_V$
LV11b	$5t_e + t_{e_T} + t_S$	$4t_p + 8t_e + t_V$	$9t_p + t_e + t_{e_T} + t_V$

Empirical analysis

We complement the theoretical analysis with an experimental evaluation, based on previous work [49]. A selection of these schemes was implemented, with representation of all the kinds of PRE schemes; this time, lattice-based schemes were also included. The selected schemes are BBS98, AFGH06a, WDLC10a, LV11a, ABPW13, and NAL15a. Our execution environment was an Intel Core 2 Duo processor @ 2.66 GHz with 4 GB of RAM.

Group-based schemes were implemented in Java using elliptic curve cryptography over a prime field. The NIST P-256 curve was used, which provides 128 bits of security [74].

Pairing-based schemes were implemented using the jPBC library [75], a pairing-based cryptography library for Java. As for the cryptographic details, we used a supersingular curve of the form $y^2 = x^3 + x$, with a 256-bit group order and 3072 bits for the field size, which achieves 128 bits of security (against the discrete logarithm problem in $\mathbb{G}$ and $\mathbb{G}_T$) [76][77]. In this case, the pairing is symmetric (i.e., Type 1 pairing), which implies that operations in $\mathbb{G}$ are costlier than in $\mathbb{G}_T$; for example, using these parameters, exponentiations in $\mathbb{G}$ are 10 times costlier

Table 3.3: Space costs of selected PRE schemes

Scheme	CT	Re-encrypted CT
BBS98	$2\|\mathbb{G}\|$	$2\|\mathbb{G}\|$
AFGH06a	$\|\mathbb{G}\| + \|\mathbb{G}_T\|$	$2\|\mathbb{G}_T\|$
AFGH06b	$\|\mathbb{G}\| + \|\mathbb{G}_T\|$	$2\|\mathbb{G}_T\|$
CH07	$\|svk\| + 3\|\mathbb{G}\| + \|\mathbb{G}_T\| + \|\sigma\|$	$\|svk\| + 3\|\mathbb{G}\| + \|\mathbb{G}_T\| + \|\sigma\|$
GA07a	$\|\mathbb{G}\| + \|\mathbb{G}_T\|$	$(N+1)(\|\mathbb{G}\| + \|\mathbb{G}_T\|)$
GA07b	$2\|\mathbb{G}\| + \|\mathbb{G}_T\| + \|m\|$	$\|\mathbb{G}\| + \|\mathbb{G}_T\| + 2\|m\| + \|id\|$
CT07a	$2\|\mathbb{G}\| + \|\mathbb{G}_T\|$	$(N+2)\|\mathbb{G}\| + (N+1)\|\mathbb{G}_T\|$
CT07b	$3\|\mathbb{G}\| + \|\mathbb{G}_T\| + \|svk\| + \|\sigma\|$	$(6N+3)\|\mathbb{G}\| + (N+1)(\|\mathbb{G}_T\| + \|svk\| + \|\sigma\|)$
Mat07	$2\|\mathbb{G}\| + \|\mathbb{G}_T\|$	$2\|\mathbb{G}\| + \|\mathbb{G}_T\|$
ABH09	$2\|\mathbb{G}\| + \|\mathbb{G}_T\|$	$2\|\mathbb{G}_T\|$
WDLC10a	$2\|\mathbb{G}\| + \|m\| + \|\mathbb{Z}_q\|$	$2\|\mathbb{G}\| + \|m\|$
WDLC10b	$2\|\mathbb{G}\| + \|m\| + \|\mathbb{Z}_q\|$	$\|\mathbb{G}\| + \|m\|$
LV11a	$\|svk\| + 2\|\mathbb{G}\| + \|\mathbb{G}_T\| + \|\sigma\|$	$\|svk\| + 4\|\mathbb{G}\| + \|\mathbb{G}_T\| + \|\sigma\|$
LV11b	$\|\mathbb{Z}_q\| + \|svk\| + 3\|\mathbb{G}\| + \|\mathbb{G}_T\| + \|\sigma\|$	$\|\mathbb{Z}_q\| + \|svk\| + 7\|\mathbb{G}\| + \|\mathbb{G}_T\| + \|\sigma\|$

Table 3.4: Experimental performance of several PRE schemes (in ms.)

Scheme	Encryption	Re-encryption	Decryption
BBS98	11.07	11.48	11.21
AFGH06a	22.76	83.52	13.76
WDLC10a	22.52	22.29	11.89
LV11a	155.27	386.93	443.87
ABPW13	1.17	20.50	0.47
NAL15a	0.43	1.15	1.22

than in $\mathbb{G}_T$. For efficiency reasons, we have made extensive use of exponentiation and pairing preprocessing of frequently-used elements.

With regard to lattice-based schemes, we implemented NAL15a as an extension of an available open-source Java implementation of NTRU [78]. We used the *ees1171ep1* parameter set from [79], which is designed for 256 bits of security. The ABPW13 was implemented using the SageMath software, using the set of parameters proposed by the authors that correspond to 143 bits of security and 128-bit plaintexts.

Table 3.4 shows the cost in ms. of the main operations of the selected PRE schemes. These figures were measured as the mean CPU time of 10.000 executions for each type of operation. Roughly, the theoretical costs presented before are translated to an empirical setting. It is worth mentioning the high performance that lattice-based schemes exhibit. Both schemes, ABPW13 and NAL15a, present similar figures regarding encryption and decryption, although ABPW13 is one order of magnitude slower when it comes to re-encryption. Note, however, that the results of these experiments are highly dependent on implementation issues, such as the choice of language, parameters (type of curve, size of fields, type of pairing, etc.), the underlying libraries and the use of preprocessing. For this reason, any comparative analysis based on these figures has to take these aspects into consideration.

3.4 Applications of Proxy Re-Encryption

The second part of this chapter is devoted to the analysis of the applications of proxy re-encryption. As described in the introduction, we performed a review of almost 70 papers regarding applications, of which 45 were finally analyzed in detail.

We also followed a bibliometric approach for drawing conclusions on this part. In particular, we classified each of the reviewed application according to certain criteria: objective, scenario and functionality. The first criterion is related to the security objectives that are intended to tackle with the application of PRE; possible objectives are confidentiality, privacy, authentication and accountability. The second criterion was clearly dominated by the cloud scenario, although other scenarios are also considered, such as wireless networks. The third criterion is associated to the intended functionality that is constructed with PRE, being access control and key management the most prominent.

Once the classification is finished we counted the number of occurrences for each category, and obtained the results shown in Table 3.5. The principal finding after this study is that, not surprisingly, most of devised PRE applications are centered on the combination of the *confidentiality* objective, the *cloud computing* scenario, and the *access control* functionality. In other words, the typical use case for proxy re-encryption is the construction of cryptographically-enforced access control system in the clouds, so data can be shared to authorized users while remaining confidential with regard to unauthorized parties and the cloud provider itself. This result is compatible with our initial research postulate, by which we consider PRE as a valuable part of the solution to the secure data sharing scenario. The next section discusses the role of PRE in this scenario in more detail.

Table 3.5: Bibliometric Analysis of PRE Applications

Criteria	Classification	Coverage
Objective	Confidentiality	80%
	Privacy	10%
	Authentication	8%
	Accountability	2%
Scenarios	Cloud	53%
	Wireless Network	8%
	Others	33%
Functionality	Access Control	67%
	Key Management	24%
	Communication	4%

3.4.1 PRE and the Secure Data Sharing Scenario

As described in the introduction of this thesis, and illustrated in Figure 1.1, the aforementioned scenario can be decomposed in several security domains, which match naturally to the actors involved in PRE. When PRE is integrated within the domains of the secure data sharing scenario, the data owner acts as a delegator who transfers access rights to encrypted information to data consumers, who act as delegatees; a proxy entity (i.e., the cloud) participates in this delegation process, transforming encrypted information by means of re-encryption. Figure 3.4 depicts this more detailed view, including the associated key material that is present on each domain.

From an abstract standpoint, data sharing in the cloud can be seen as a problem of access control, where the protected resource is encrypted information. Proxy re-encryption is then used to construct a cryptographically-enforced access control. Ateniese et al. proposed in [11] a generic way to do this, as shown in Figure 3.5. Let us assume three main actors: (i) the data owner, with public and private keys pk_A and sk_A; (ii) a consumer, with public and private keys pk_B and sk_B; and (iii) the cloud, which acts as a proxy and allows access to encrypted data through re-encryption. The data to be protected is encrypted by the data owner with a fresh symmetric encryption key, the *data key*, which is in turn encrypted his public key pk_A using the PRE scheme, thus creating an *encrypted lockbox*. This lockbox, which contains the encrypted data and data key, can now be stored in an untrusted repository, such as the cloud. Note also that the production of encrypted data can be performed by any entity that knows the public key of the data owner, supporting this way multiple data sources. For granting access to his data, the data owner generates re-encryption keys for authorized users and hands

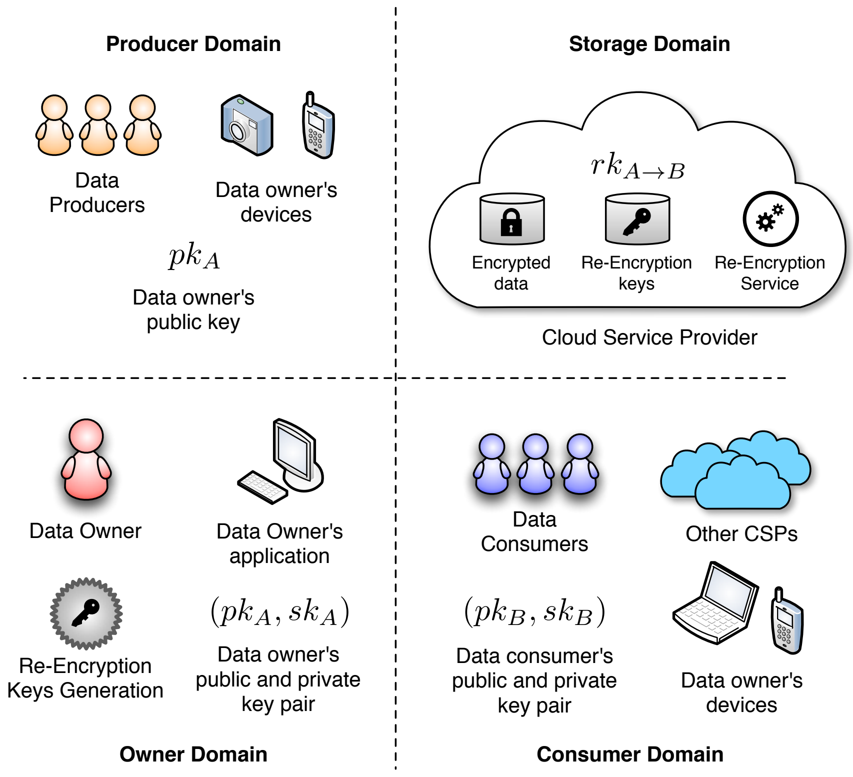


Figure 3.4: Integration of PRE and Secure Data Sharing Domains

them to the proxy entity. As mentioned before, the re-encryption key $rk_{A \to B}$ can be seen as an access delegation token that the owner of the data creates in order to enable user U to access to his data. When the delegatee user requests access to the encrypted data, the proxy re-encrypts the lockbox. Finally, the delegatee user decrypts the lockbox with his private key sk_B.

This template for cryptographically-enforced access control stems naturally from the main functionality of PRE, which is delegation of access rights to encrypted data. Note that it is not necessary for the data owner to be online. The only requirement is that the cloud is available (which is an intrinsic characteristic of the cloud paradigm). The data owner can be off-line, while consumers interact directly with the cloud provider in order to access to authorized information. Note that this assumes a honest-but-curious trust model, where the cloud provider may have an incentive for accessing users' data without their permission, but at the same time, it is assumed to behave honestly with respect its functionality.

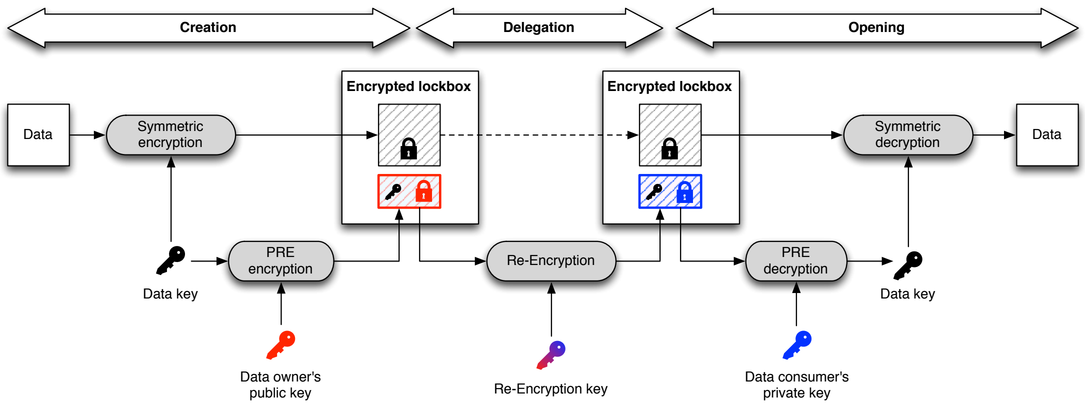


Figure 3.5: Lifecycle of a PRE-based lockbox

Review of the literature

In this section we review the state of research on solutions for cloud data sharing that use proxy re-encryption. Most of them share a similar essence, although vary greatly with regards to specific constructions and designs.

In [80], Yu et al. propose a system for data sharing in the cloud, using a combination of Key-Policy Attribute-Based Encryption (KP-ABE) and PRE. Data is encrypted using KP-ABE and stored in the cloud, so only users in possession of an appropriate collection of attribute secret keys can decrypt the data. Besides managing encrypted data, the cloud also manages attribute secret keys of the users, except for one special secret key which is required for all decryptions, so the cloud cannot decipher anything. The reason why the cloud manages these secret keys is to handle revocation of users. When a data owner revokes certain users, then new keys must be provided for the remaining users, and encrypted data must be re-encrypted. Both issues are handled by the cloud using PRE, so the data owner simply generates re-encryption keys for transforming not only ciphertexts, but also attribute public and secret keys. This added functionality is possible by carefully integrating the BBS98 scheme with a KP-ABE scheme. For efficiency reasons, the re-encryptions are performed in a "lazy" way, that is, only when an access request from a user is made. [81] propose a modification to Yu et al. design in order to avoid collusions between the provider and revoked users, but their proposal consists basically in replacing the cloud provider with a trusted third party, which implies relying on stronger trust assumptions. [82, 83, 84] describe similar approaches but for integration with Ciphertext-Policy Attribute-Based Encryption (CP-ABE), where the access structure is associated to the encrypted data rather than to the user attribute key.

Lin et al. propose in [85] a combination of threshold encryption with PRE. This proposal fits the general template of PRE application for data sharing, with the exception being that the proxy entity is distributed among several servers in order to support a decentralized architecture. On an access request, a randomly chosen subset of these servers re-encrypt the data, and since each of these servers store a share of the data owner's secret key, they also perform partial decryptions of the encrypted data. The delegatee user combines the partially decryptions in order to obtain the requested data.

Jia et al. describe in [86] a classical instantiation of the PRE-based access control template. The proposed system uses the CPA-secure identity-based PRE scheme from [13], GA07a, where re-encryption keys represent authorization tokens between users. Revocation is handled by the data owner by asking the provider to replace re-encryption keys with new ones.

Liu et al. propose in [87] a time-constrained access control scheme, combining once again PRE and ABE. In this case, ABE is used for describing time-based access control policies, whereas PRE is used for updating the time attributes. Their proposed system follows the typical template for PRE-based access control in the cloud.

The proposal from [88] is different from the previous ones in that it is a general model that can be instantiated with different PRE schemes. The proposals presented before require a specific PRE scheme (most of the times, a variation of the classic BBS98 scheme), carefully tailored for its integration with other primitives and protocols, such as ABE schemes. This general model modifies the encrypted lockbox scheme presented before, including ABE in the following manner. The data key k is split in two different keys k_1 and k_2 by means of a XOR operation, so $k = k_1 \oplus k_2$. Next, PRE is used for encrypting k_1 and ABE for k_2. If a consumer requests access to the encrypted data, then he must possess the necessary ABE secret keys and the cloud must have the corresponding re-encryption key for performing the re-encryption. In that case, then the consumer is able to retrieve k_1 and k_2, and hence, to decrypt the data with the data key k. In this model, fine-grained access control is provided by ABE, whereas PRE makes revocation possible. This model is an example of use of re-encryption keys as authorization tokens, where the presence of re-encryption keys in the cloud provider means that the consumer is authorized to access the data.

Xiong et al. present CloudSeal in [89], a cloud-based data sharing system that integrates PRE with a secret sharing scheme. This proposal is slightly similar to the one from Yu et al., but instead of ABE, it uses a secret sharing scheme. The main drawback is that it assumes the existence of secure channels between the data owner and data consumers, which nullifies the necessity of the proposal.

Han et al. propose in [90] an identity-based PRE scheme suitable for intra- and inter-domain data sharing. The main feature of this scheme it that it bounds the re-encryption keys not only to the consumers's identity but also to a specific ciphertext (i.e., a shared file). This design choice implies that the data owner must create a different re-encryption key for each pair of data consumer and shared file, but also limits the chances that the cloud provider re-encrypts arbitrary data. Lin et al. also propose a similar idea [91], but with a hierarchical PRE instead of identity-based PRE.

Other similar examples are found for more specific applications. For example, the PRE-based access control template is used in [92, 93] for creating an Identity-as-a-Service model where the cloud identity provider cannot access the identity information. In [94], PRE is used for delegating access to encrypted search indexes, in the context of privacy-aware searches. Proxy re-encryption is integrated to the MapReduce paradigm for privacy-preserving Big Data processing in [95]. Other examples are found for service aggregation [96], de-duplication in secure cloud storage [97], and privacy-preserving location-based services [98].

Findings

Among the principal findings after this review, is the occurrence of common patterns that appear among these proposals. For instance, the most prominent is that authorization is realized by considering re-encryption keys as authorization tokens. Thus, $rk_{A \to B}$ can be seen as an authorization from data owner A to user B. In this case, the enforcement of access rights is naturally realized by means of re-encryption.

When it comes to revocation of access rights, PRE can be used in two ways. The first way is applicable when re-encryption keys are used as authorization tokens between users; in this case, the owner simply instructs the cloud provider to delete the re-encryption keys [11, 88]. Therefore, it is necessary that the cloud provider is trusted enough to ensure that the keys are actually deleted. The second way, used by Yu et al. [80], is to re-encrypt the data so it cannot be decrypted by the revoked user. In this case, it is also necessary to trust that the cloud provider is performing the re-encryption and that it is not maintaining a copy of the unre-encrypted data.

Another interesting finding concerns other cryptographic primitives that usually accompany PRE. The main is Attribute-Based Encryption (ABE), which is used in combination with PRE in at least 9 of the 45 papers (that is, a fifth part).

Another cryptographic technique commonly used in conjunction with PRE is Threshold Encryption.

Principal Motivations for this Use Case

As mentioned in Section 3.1, proxy re-encryption can be seen as a cryptographic primitive for accomplishing access delegation to encrypted information. Therefore, it has a natural application in the construction of cryptographically-enforced access control systems, in particular, for scenarios where the protected resource is stored externally (e.g. the cloud).

In a PRE-based solution for the cloud, private data can reside in the cloud in encrypted form and be shared to authorized users, while still remain confidential with regard to unauthorized parties and the cloud provider itself. The use of encryption for protecting data at rest can decrease the risks associated to data disclosure in this kind of scenario, since outsourced information can only be effectively shared if access has been delegated by the data owner. In this Section we have reviewed several proposals that revolve around this basic model.

We believe that this represents a great advance with respect to the state of current cloud services, where users' data is fully controlled by the cloud provider, or, at its best, data is encrypted with keys controlled by the provider. Data owners are obliged to trust that the provider will make proper use of his data and will guarantee its protection. Therefore, a PRE-based solution could enable users to outsource their data to cloud providers without necessarily establishing a strong trust relation with them, relying instead in the protection granted by the underlying cryptographic mechanism.

At this point, it is worth studying what kind of incentives may motivate cloud providers to implement this kind of solutions, that is, handle data in encrypted form. Note that, in the first place, cloud providers would lose control over the user's data, which is currently a valuable asset. Moreover, providers may incur in more expenses as a result of implementing these additional security mechanisms. However, there are two main incentives that could encourage cloud providers to adopt this solution.

The first incentive is compliance and minimization of liability. An intrinsic characteristic of the cloud is that it enables ubiquitous access to data, which can, potentially, infringe privacy and data protection regulations. Moreover, the cloud can also be used to host and process information that is of problematic nature, such as, illegal or defamatory material [99]. Therefore, cloud providers are affected by specific laws and regulations regarding privacy, data protection and copyright, among others. Although cloud providers currently try to reduce their liability through specific clauses in SLAs, legal responsibility for the data in the cloud also lies on the side of the provider. In contrast, in a PRE-based solution where the outsourced data is encrypted prior to reaching the cloud, liability of the cloud provider is drastically minimized, as it is unable to read user's data, for which it does not hold the decryption keys. As a consequence of this, users should be the ones designated as liable and subject to the enforcement of key disclosure laws.

Data confidentiality as an added value represents a second type of incentive. This way cloud provider could offer secure data processing and storage as an added value, establishing a competitive advantage over the rest. This characteristic could be an important driver in users' decision process, as there is an increasing interest in this kind of services. In our opinion, there is room in the current cloud service landscape for a business model based on the respect for users' privacy and data confidentiality.

Economic analysis

One of the postulates of this thesis is that the application of proxy re-encryption as a cryptographic mechanism for supporting secure data sharing in the cloud is practical. However, a thorough analysis of the practicality of this kind of solution should also determine whether it is economically viable or not.

There is a vast amount of proposals for improving the security, reliability and functionality of cloud services, but, at the same time, there are almost no critical analyses about the economic impact that arises from the implementation of such proposals. In particular, most of proponents of the use of cryptography only provide theoretical analysis of security and computation complexity, but do not study whether their proposals are economically feasible or not, even when their solutions often imply an intensive use of computation or communication resources.

In [7], Chen and Sion analyze the economic impact of the outsourcing of computation and storage to the cloud. The key contribution of this work is the quantification of the costs associated to this outsourcing, which are driven by several factors, such as hardware, energy and personnel. The authors break down the expenses derived from computation, storage and network services, using the *picocent* (which corresponds to 10^{-12} USD cents) as unit of cost. Table 3.6 shows these estimations, which correspond to 2010. Additionally, we provide an adjustment of these prices to the year 2015. Although some factors that influence

Table 3.6: Estimated cost (in picocents) for cloud resources

Resource	Cost (2010)	Cost (2015)
Computation (per CPU cycle)	0.93 – 2.36	0.65 – 1.65
Network (per bit)	800 – 6 000	560 – 4200
Storage (per bit per year)	100	70

Table 3.7: Comparison of re-encryption costs in the cloud

Scheme	Time (ms)	Cost (cents)	#re-encryptions/cent
BBS98	11.48	4.58E-05	21 844
AFGH06a	83.52	3.33E-04	3 003
WDLC10a	22.29	8.89E-05	1 1250
LV11a	386.9	1.54E-03	648
ABPW13	20.5	8.17E-05	12 233
NAL15a	1.15	4.59E-06	218 063

these costs remain stable (e.g., infrastructures, energy and personnel), the core resource, which is hardware, increases with time its efficiency per unit of cost. Computing resources are the best example, since the cost per CPU cycle has decreased exponentially through history, following Moore's law. A simple analysis based on the prices of Amazon EC2 shows that these have decreased roughly a 30% from 2010 to 2015 (a yearly 7%); there are, however, some indications that real costs for the cloud provider have decreased even more and that their margins of benefits have progressively increased [100].

Based on these figures, we estimate the cost per re-encryption operation for several of the PRE schemes analyzed in this survey. A similar economic assessment about the use of proxy re-encryption in a cloud setting is presented in [92]. Recall that the re-encryption is the basic operation that the cloud provider performs in a PRE-based access control solution for the cloud. Table 3.7 shows these estimations, assuming a cost of 1.5 picocents per CPU cycle. These figures vary greatly depending on the selected scheme and its properties, but demonstrate that the cloud provider could perform in the order of thousands of re-encryptions per cent.

For illustration purposes, let us assume a scenario similar to that suggested in [92], where a cloud provider implements a PRE-based access control solution that performs a million re-encryptions per day, one for each access request it receives. From the figures of the last table, it can be seen that the total cost of these operations over the course of a year range from $5,631 for the LV11a scheme, to just $17 for the NAL15a scheme. In our opinion, these expenses are reasonable for a cloud provider, considering the costs that it could incur in the case of a disclosure or security breach. These costs are difficult to estimate, but at the very least such incidents would have a negative impact with regard to loss of customers and reputation.

3.4.2 Other Applications of PRE

Although protecting outsourced information in the cloud is a natural application, there are other interesting uses for proxy re-encryption, where *group key management* is the most remarkable. In this case, PRE can be used for different purposes such as distributing keys, revoking access, performing key escrows, etc. This has multiple applications, such as DRM protection [101, 102, 103, 104], and security in multicast communications [105, 106, 107, 108].

Among alternative uses of PRE we find privacy-preserving solutions for RFID [109, 110, 111], authentication in VANETs [112, 113], location privacy [114], privacy in online social networks [115], anonymity in P2P communication [116], and access control in other scenarios [117, 118].

3.5 Summary

In this chapter, we survey and analyze the current state of research on proxy re-encryption, for both constructions and applications. The most prominent proxy re-encryption schemes are studied in the light of relevant properties and security models. We additionally provide a comparative analysis of the performance of selected schemes, both from the theoretical and experimental points of view.

With regards to the applications, we perform an extensive analysis of the available literature following a bibliometric approach. As a result, we confirm that secure data sharing in the cloud is currently the use case with the most potential for proxy re-encryption. In a PRE-based solution to cloud data sharing, private data can reside in the cloud in encrypted form and be shared to authorized users by means of re-encryption, while still remain confidential with regard to unauthorized parties and the cloud provider itself, reducing this way the risks traditionally associated to this scenario. We postulate that the cloud data sharing problem could certainly benefit from the use of proxy re-encryption, given the functionalities and performance levels that it provides. Moreover, it is more realistic than other solutions, such as those based on homomorphic encryption.

In the next chapters, we will tackle some of the challenges associated to proxy re-encryption described in the introduction. In particular, the next chapter is devoted to analyzing the very definitions of security in PRE.

Chapter 4

Parametrizing Security Notions for Proxy Re-Encryption

The previous chapter gave deep insight into the basic concepts of proxy re-encryption, including some of the most representative schemes and applications. Now we put the focus on security definitions, the fundamental abstractions that shape the idea of security in proxy re-encryption. As proxy re-encryption is a special type of public-key encryption, it is natural to “reuse” the security notions of the latter (see Section 2.3.1). Thus, PRE schemes in the literature aim to achieve IND-CPA [11], IND-CCA1 [48] and IND-CCA2 [45] security. However, the desired security notions are often defined in an ad-hoc manner for each scheme, with subtle variations and restrictions, caused by the lack of established definitions of the security notions of PRE, and ultimately, of the attack models, which in the case of PRE are potentially richer than in PKE because of the possibility of re-encryption. While in the PKE setting, the different attack models are determined by the availability of the decryption oracle (i.e., not available in CPA, only before the challenge in CCA1, or throughout the game in CCA2), in the PRE setting it is reasonable to base the attack models on the availability of both the decryption and the re-encryption oracles.

In this chapter we explore these subtleties by means of a parametric family of attack models that considers the availability of both oracles separately. In turn, the resulting set of attack models enables fine-grained security notions for PRE. Based on these notions, we study some of the relations that arise between them, and especially, the separations, which further support the importance of the re-encryption oracle. The identified separations stem from the study of a new property of PRE, called “privacy of re-encryption keys”, and that formalizes the concept that re-encryption keys should not be leaked through the re-encryption function; we show that when a scheme does not satisfy this property, it cannot achieve a proper security notion for PRE.

Finally, in light of this formalization, we show how a recent "CCA1-secure" proxy re-encryption scheme from PKC 2014 [48] does not actually achieve a meaningful chosen-ciphertext security notion for PRE, since it breaks when one considers an oracle for re-encryption. The methodology described for this attack, based on the leakage of re-encryption keys, supports the idea that PRE schemes should not leak re-encryption keys.

4.1 Introduction

In the context of PKE, there is a wide consensus on the consideration of indistinguishability against adaptive ciphertext attacks (IND-CCA2) as the right notion of security, although weaker notions such as IND-CCA1 may be sufficient in other contexts, such as homomorphic encryption. See [119] for a detailed exposition on the importance of security against chosen ciphertext attacks. The arguments for CCA2-security also hold in the case of proxy re-encryption. Canetti and Hohenberger motivate the necessity of CCA2-security with an example of a decryption oracle in a scenario of encrypted email forwarding, where email user Alice can set the mail server to forward all her encrypted emails to a designated recipient Bob, without giving her secret keys to either the mail server or Bob [45]. In this example, an attacker might gain access to a decryption oracle by producing ciphertexts, emailing them to Alice and then hoping that she responds with "*Did you send the following attachment to me?*", together with the decrypted ciphertext. If instead of this, it is Bob who responds to the attacker attaching the re-encrypted ciphertext, then the attacker would gain access to a re-encryption oracle. The motivation for considering the re-encryption oracle also comes from the study of applications of proxy re-encryption from the literature, where the re-encryption function is deployed as a service, so participants of the system can re-encrypt ciphertexts from one user to another. This service effectively simulates a re-encryption oracle. In fact, expecting the availability of this oracle is a weaker assumption than presuming the availability of a decryption oracle, since the latter requires knowledge of secret keys, and therefore, it probably would be better protected and less likely to be available.

While in the PKE context a *chosen-ciphertext attack* (CCA) means that the adversary is free to choose ciphertexts for querying a decryption oracle, in the PRE context a CCA adversary should also be able to freely re-encrypt ciphertexts. That is, the re-encryption functionality should be captured by any meaningful

CCA model for PRE. In fact, the re-encryption oracle alone is able, in some cases, to simulate a decryption oracle, as shown in Section 4.3.3. This further supports our claims about its relevance and raises questions about what are proper security notions for PRE.

4.2 Definitions of Security for Proxy Re-Encryption

The definitions of security in the context of PRE extend those of PKE due to the possibility of re-encrypting ciphertexts. That is, in addition to the KeyGen, Enc and Dec functions, a PRE scheme defines two more, ReKeyGen and ReEnc, which are associated with this additional capability (as described in Definition 3.1 in Section 3.2.1). This incorporation is what led us to conceive a more comprehensive characterization of the attack models and security notions for PRE.

Before proceeding, recall that we distinguished in Section 3.2.1 between schemes with a single ciphertext space from those with multiple ciphertext spaces. The former are usually associated with multi-hop PRE schemes, although there are some recent single-hop schemes based on lattices that also present this characteristic [47, 48]. The latter are, on the contrary, usually associated with unidirectional schemes, since this way there may be transformations between ciphertext spaces that are valid only in one direction. In this chapter, we restrict ourselves to PRE schemes defined over a single ciphertext space, although our results can be extended to the other case.

The additional re-encryption capability that is introduced in proxy re-encryption clearly makes the definitions of security more complex. As in the decryption case, the re-encryption capability can be modeled by the access to a re-encryption oracle $\mathcal{O}_{reenc}$. Thus, it is natural to think of a collection of attack models with different access levels to the decryption and re-encryption oracles. In this section we describe a parametric set of chosen-ciphertext attack models, which depends solely on the availability of these oracles for each phase of the security game. This family of attack models in turn implies a collection of security notions. As mentioned before, we will restrict ourselves to the indistinguishability goal.

It is reasonable to argue that meaningful chosen-ciphertext attack models for proxy re-encryption should provide re-encryption capabilities, in the form of a re-encryption oracle. However, access to this oracle is not sufficient guarantee that proper re-encryption capabilities are granted to the adversary. In order to not make the security game trivial for the adversary, it is necessary that the definition of the re-encryption oracle includes some restrictions. At the same time, it is also important that these restrictions are not too strong, since this would achieve weaker notions of security, as pointed out by Canetti and Hohenberger in [45]. In particular, the re-encryption oracle should be able to re-encrypt any ciphertext that is not derived from the challenge ciphertext, between *any* pair of users, including the target user. Otherwise, the security model would be too restrictive, leading to weaker security notions. Therefore, it is paramount that the proposed attack models are accompanied by a reasonable definition of the oracles available, as well as their restrictions.

4.2.1 Oracles

The following definitions of oracles and their restrictions are adapted from those of Canetti and Hohenberger [45]. A core concept that shapes the restrictions is the notion of *derivatives of the challenge.* As stated before, the security game could be trivial for the adversary if there were no restrictions, since in certain cases a re-encryption oracle can be used to simulate a decryption oracle. For example, consider the case of the challenge ciphertext c^*, which can be potentially re-encrypted from the target public key pk^* to any other one. Canetti and Hohenberger introduce the notion of derivatives as a means for shaping the oracle restrictions. Informally, the derivatives of the challenge are those pairs (pk, c) that are linked to (pk^*, c^*) through queries to the re-encryption and re-encryption key generation oracles, and which would allow trivial attacks from the adversary.

Definition 4.1 (Derivatives of the challenge)**.** *The set of derivatives of* (pk^*, c^*) *is defined inductively, as follows:*

- (pk^*, c^*) *is a derivative of itself.*
- *If* (pk_j, c_j) *is a derivative of* (pk_i, c_i) *and* (pk_i, c_i) *is a derivative of* (pk^*, c^*)*, then* (pk_j, c_j) *is a derivative of* (pk^*, c^*)*.*
- *If the adversary has issued a re-encryption query* (pk_i, pk_j, c_i) *and obtained a ciphertext* c_j *as response, then* (pk_j, c_j) *is a derivative of* (pk_i, c_i)*.*
- *If the adversary has issued a re-encryption key generation query* (pk_i, pk_j)*, and* $\mathsf{Dec}(pk_j, c_j) \in \{m_0, m_1\}$*, then* (pk_j, c_j) *is a derivative of all pairs* (pk_i, c)*.*

Once this concept has been established, we will describe the oracles involved in the security game. Apart from the decryption and re-encryption oracles, it is necessary to provide additional oracles for dealing with the inherently multi-user nature of proxy re-encryption. These oracles provide the adversary of keys for multiple users, which can be either honest or corrupt depending on whether the adversary is unaware of the corresponding secret key or not.

Let $n \in \mathbb{N}$ be the security parameter for a PRE scheme, and $\mathcal{I}_H$ and $\mathcal{I}_C$ be the sets of indices of honest and corrupt users, respectively. By definition, the target user is deemed honest, so its index $i^* \in \mathcal{I}_H$. The public and private keys of target user are pk_{i^*} and sk_{i^*}, respectively; we will, however, notate them as pk^* and sk^*, for simplicity. We define the following oracles, which will be made available to the adversary during the security game and which can be invoked multiple times in any order:

- Honest key generation $\mathcal{O}_{honest}$: The oracle obtains a new keypair $(pk_i, sk_i) \leftarrow \mathsf{KeyGen}(n)$, adds index i to $\mathcal{I}_H$, and returns the public key pk_i.
- Corrupt key generation $\mathcal{O}_{corrupt}$: The oracle obtains a new keypair $(pk_i, sk_i) \leftarrow \mathsf{KeyGen}(n)$, adds index i to $\mathcal{I}_C$, and returns the pair (pk_i, sk_i).
- Re-encryption key generation $\mathcal{O}_{rkgen}$: On input a pair of public keys (pk_i, pk_j), the oracle returns the re-encryption key $rk_{i\to j} \leftarrow \mathsf{ReKeyGen}(pk_i, sk_i, pk_j, sk_j)$. The adversary is only allowed to make queries where $i \neq j$, and either $i, j \in \mathcal{I}_H$ or $i, j \in \mathcal{I}_C$.
- Re-encryption $\mathcal{O}_{reenc}$: On input (pk_i, pk_j, c), where $i \neq j$ and $i, j \in \mathcal{I}_H \cup \mathcal{I}_C$, the oracle returns the re-encrypted ciphertext $c' \leftarrow \mathsf{ReEnc}(rk_{i\to j}, c)$. The adversary is not allowed to make queries where $j \in \mathcal{I}_C$ and (pk_i, c) is a derivative of (pk^*, c^*).
- Decryption $\mathcal{O}_{dec}$: On input (pk_i, c), where $i \in \mathcal{I}_H \cup \mathcal{I}_C$, the oracle returns $m \leftarrow \mathsf{Dec}(sk_i, c)$. The adversary is not allowed to make queries where (pk_i, c) is a derivative of (pk^*, c^*).

The ability to re-encrypt and decrypt ciphertexts is modeled by $\mathcal{O}_{reenc}$ and $\mathcal{O}_{dec}$. The restrictions on the derivatives of the challenge ciphertext disallow trivial attacks from the adversary, but are flexible enough to support a wide range of queries. For example, it should be possible for the adversary to issue re-encryption queries of the form $(pk^*, pk_j, \hat{c})$, where pk_j is corrupt, but $(pk^*, \hat{c})$ is not a derivative of (pk^*, c^*).

The key generation oracles $\mathcal{O}_{honest}$, $\mathcal{O}_{corrupt}$, and $\mathcal{O}_{rkgen}$ are always available for the adversary,subject to the restrictions above. It can be seen that we assume a *static corruption* model, where the adversary must decide to corrupt a user or not before asking for the generation of the user's keypair [64], hence the distinction between honest and corrupt key generation. In addition, we also put common restrictions regarding how the adversary obtains keys. In particular, we are assuming the *knowledge of secret key* model, where the challenger generates the key material of all users. A stronger model is the *chosen key* model, where the adversary can adaptively choose public keys for malicious users [12].

The restrictions on the re-encryption key generation oracle have a strong influence on the security model; in particular, we disallow the generation of re-encryption keys between honest and corrupt users, as in some cases, this may lead to trivial attacks from the adversary. We stress that the intention of this work is to support definitional unity for PRE by providing a universal framework that captures the essence of CCA-security, thus avoiding the definition of security notions that are particular for each PRE subfamily. Although it would be possible to define stronger notions by removing the restrictions on this oracle, this would render these definitions useless for schemes with useful PRE properties, such as multi-hop or transitiveness, which are of interest for many applications. For instance, Libert and Vergnaud [12] permit all possible re-encryption key generation queries, except those from the target user to a corrupt one. However, this only works when the scheme is single-hop, non-transitive and resistant to collusions.

There are several examples that justify the restrictions for honest-to-corrupt re-encryption keys. For instance, in multi-hop PRE, an adversary could trivially win the game by first re-encrypting the challenge ciphertext to a honest user key, and then using the honest-to-corrupt key to re-encrypt it to a corrupt user key, thus trivially winning the game. The adversary could also achieve a similar result if the scheme is transitive. Another example is found in not collusion-resistant schemes, where the adversary can use a honest-to-corrupt key and a corrupt secret key to extract the honest secret key. With regard to the restrictions to corrupt-to-honest keys, there are two main reasons for that. On the one hand, adversaries in non-interactive PRE schemes do not require the private key of the delegatee for computing re-encryption keys, so the re-encryption key generation oracle would be superfluous. On the other hand, interactive schemes are usually bidirectional, which means that it is possible to compute a honest-to-corrupt key from a corrupt-to-honest key, so the arguments above apply. Therefore, although it is possible to remove these restrictions, it would make the resulting security notions too particular and not relevant for many PRE schemes, which conflicts with our original intention to attain definitions that are universal.

4.2.2 Attack Models

Inspired by the mnemonic defined in [24] for attack models CCA1 and CCA2, where the number denotes the last adversarial stage during which she has access to a decryption oracle, we analogously define a parametric set of attack models for proxy re-encryption based on the last adversarial stage during which the adversary has access to the decryption and re-encryption oracles. Thus, our parametric set of attack models is characterized by a pair of indices $i, j \in \{0, 1, 2\}$, so $\mathsf{CCA}_{i,j}$ denotes an attack model where the adversary has a decryption oracle until Phase i, and a re-encryption oracle until Phase j. As extreme cases, we have that a "pure" CPA model is then denoted by $\mathsf{CCA}_{0,0}$, whereas a "pure" CCA model is represented by $\mathsf{CCA}_{2,2}$. There is, however, a range of intermediate attack models, differentiated by the last stage during which the adversary has access to the decryption and re-encryption oracles. In the following, we formalize this concept.

As stated, what actually varies among the proposed attack models is the availability of the decryption and re-encryption oracles. That is, throughout the game the adversary has access to the key generation oracles $\mathcal{O}_{honest}$, $\mathcal{O}_{corrupt}$, and $\mathcal{O}_{rkgen}$. We denote as Ω^{kg} to the set that comprises these oracles. Now, let Ω_1^{cca} and Ω_2^{cca} denote the sets of additional oracles that are available in Phases 1 and 2, respectively; the possible values that these sets can take are therefore $\varnothing$, $\{\mathcal{O}_{reenc}\}$, $\{\mathcal{O}_{dec}\}$, and $\{\mathcal{O}_{dec}, \mathcal{O}_{reenc}\}$. Let $\Omega_1 = \Omega_1^{cca} \cup \Omega^{kg}$ and $\Omega_2 = \Omega_2^{cca} \cup \Omega^{kg}$ be the sets of oracles available in Phases 1 and 2, respectively. Since the set Ω^{kg} of key generation oracles is always present, then the sets Ω_1^{cca} and Ω_2^{cca} fully characterize the possible attack models. We can describe the available oracles in a more formal manner as follows. Let $\mathsf{CCA}_{i,j}$ be an attack model for PRE and $k \in \{1, 2\}$, then $\mathcal{O}_{dec} \in \Omega_k^{cca}$, for $i \geq k$, and $\mathcal{O}_{reenc} \in \Omega_k^{cca}$, for $j \geq k$.

Table 4.1 describes the possible attack models for proxy re-encryption as a function of the availability of the oracles, characterized by the sets Ω_1^{cca} and Ω_2^{cca}. The different combinations of available oracles produce a parametric set of attack models, with 9 possible choices. Note that we disallow the possibility of having an oracle in Phase 2 but not in Phase 1, as the wording used in the definition of the parametric attack models uses the term "until Phase" for defining oracles' availability and not "in Phase". As a particular case of these attack models we have that when $\Omega_1^{cca} = \Omega_2^{cca} = \varnothing$, the attained model $\mathsf{CCA}_{0,0}$ is indeed CPA, since no decryption and re-encryption oracles are provided.

Not all possible attack models of this set are meaningful in the context of proxy re-encryption. In fact, it seems reasonable to consider the access to a re-encryption oracle to be easier than to a decryption oracle, since usually, re-encryption is performed as a service by an online semi-trusted entity, whereas decryption capabilities are retained by the users. Hence, attack models in which there are more decryption than re-encryption capabilities, and in particular, those where a decryption oracle is provided but not a re-encryption one ($\mathsf{CCA}_{1,0}$ and $\mathsf{CCA}_{2,0}$),

Table 4.1: Parametric Family of Attack Models for PRE

Ω_1^{cca}	Ω_2^{cca}	Attack model
$\varnothing$	$\varnothing$	$\mathsf{CCA}_{0,0} = \mathsf{CPA}$
$\{\mathcal{O}_{reenc}\}$	$\varnothing$	$\mathsf{CCA}_{0,1}$
$\{\mathcal{O}_{reenc}\}$	$\{\mathcal{O}_{reenc}\}$	$\mathsf{CCA}_{0,2}$
$\{\mathcal{O}_{dec}\}$	$\varnothing$	$\mathsf{CCA}_{1,0}$
$\{\mathcal{O}_{dec}, \mathcal{O}_{reenc}\}$	$\varnothing$	$\mathsf{CCA}_{1,1}$
$\{\mathcal{O}_{dec}, \mathcal{O}_{reenc}\}$	$\{\mathcal{O}_{reenc}\}$	$\mathsf{CCA}_{1,2}$
$\{\mathcal{O}_{dec}\}$	$\{\mathcal{O}_{dec}\}$	$\mathsf{CCA}_{2,0}$
$\{\mathcal{O}_{dec}, \mathcal{O}_{reenc}\}$	$\{\mathcal{O}_{dec}\}$	$\mathsf{CCA}_{2,1}$
$\{\mathcal{O}_{dec}, \mathcal{O}_{reenc}\}$	$\{\mathcal{O}_{dec}, \mathcal{O}_{reenc}\}$	$\mathsf{CCA}_{2,2}$

do not seem to be appropriate for shaping security in a PRE scenario. This is discussed in more detail in Section 4.3.4.

We later show general results that separate some of the security notions that arise from these attack models. In particular, we demonstrate how the leakage of re-encryption keys through $\mathcal{O}_{reenc}$ induces a separation between notions of security. In addition, in Section 4.4 we show a concrete example of a PRE scheme that is proven secure under a $\mathsf{CCA}_{1,0}$ model, but that fails when one considers a $\mathsf{CCA}_{1,1}$ model.

4.2.3 Security Notions for PRE

In the same fashion as for PKE, security notions for PRE are constructed by combining a security goal (in this case, indistinguishability of encryptions) and the proposed family of attack models. The following definitions are based upon the definition of security for PKE given in Section 2.3.1.

Definition 4.2. *Let Π=(KeyGen, ReKeyGen, Enc, Dec, ReEnc,) be a PRE scheme, $A = (A_1, A_2)$ a polynomial-time adversary, and Ω_1 and Ω_2 be the set of available oracles for A_1 and A_2, respectively. For $i, j \in \{0, 1, 2\}$, $\delta \in \{0, 1\}$, and $n \in \mathbb{N}$,* *the indistinguishability game is defined by the experiment*

$$
\begin{array}{ll}
\multicolumn{2}{l}{\text{Experiment } \mathbf{Exp}_{\Pi,A,\delta}^{\textsf{\textit{IND-CCA}}_{i,j}}(n)} \\
\quad (pk^*, sk^*) \xleftarrow{R} \mathsf{KeyGen}(n); & (m_0, m_1, s) \leftarrow A_1(pk^*); \\
\quad c^* \leftarrow \mathsf{Enc}(pk^*, m_\delta); & d \leftarrow A_2(m_0, m_1, s, c^*) \\
\quad \text{return } d &
\end{array}
$$

It is required that in the case that $i = 2$ *or* $j = 2$*, the oracle queries from adversary* A_2 *have to satisfy the restrictions about derivatives of the challenge ciphertext* c^**. The sets of available oracles* Ω_1^{cca} *and* Ω_1^{cca} *are defined in accordance to the attack model* $\textsf{\textit{CCA}}_{i,j}$*, as shown in Table 4.1.*

Definition 4.3. *Let* Π *be a proxy re-encryption scheme and* A *a polynomial-time adversary. For* $i, j \in \{0, 1, 2\}$ *and* $n \in \mathbb{N}$*, the advantage of* A *is given by*

$$\mathbf{Adv}_{\Pi,A}^{\textsf{\textit{IND-CCA}}_{i,j}}(n) = |\Pr[\mathbf{Exp}_{\Pi,A,1}^{\textsf{\textit{IND-CCA}}_{i,j}}(n) = 1] - \Pr[\mathbf{Exp}_{\Pi,A,0}^{\textsf{\textit{IND-CCA}}_{i,j}}(n) = 1]|$$

We say that the PRE scheme Π *is* $\textsf{\textit{IND-CCA}}_{i,j}$ *secure if the advantage* $\mathbf{Adv}_{\Pi,A}^{\textsf{\textit{IND-CCA}}_{i,j}}$ *is negligible.*

Note that, in the security game described by the experiment, we are assuming the *selective model*, since the challenger fixes the target public key pk^* at the beginning. This choice is made for the sake of clarity in the proof, and is also followed by other relevant references, such as [11, 12].

Another aspect worth mentioning arises from how we disallow oracle queries that include the challenge ciphertext (or a derivative). As shown in [15] by Bellare et al., there are different ways for formalizing this in the IND-CCA security game for PKE. They identify four styles (SP, SE, BP, and BE), which result from the combination of two orthogonal factors: the first factor is to disallow these queries only in the second phase ("S") or in both phases ("B"); the second factor depends on whether the adversary is penalized a posteriori in case she makes such queries ("P"), or, simply, this possibility is excluded from the experiment ("E"). In the light of this formalization, the limitations posed on the oracle queries from A_2 in the definition of our experiment imply that our security notions fall under the IND-CCA-SE category, as we only exclude adversaries that query derivatives of the challenge ciphertext in phase 2; no restrictions exist on phase 1, as the set of derivatives would be empty in this phase.

Finally, once the security notions have been defined, it is interesting to illustrate them with examples of related schemes. It is reasonable to expect that most of these representatives belong to the central security notions, namely $\mathsf{IND\text{-}CCA}_{0,0}$, $\mathsf{IND\text{-}CCA}_{1,1}$, and $\mathsf{IND\text{-}CCA}_{2,2}$ (see Section 4.3.4 for a discussion on why we consider these notions as central). For example, the schemes from Blaze et al. [43] and Ateniese et al. [11] are widely-known examples of $\mathsf{IND\text{-}CCA}_{0,0}$ security. Canetti and Hohenberger present in [45] a classic instance of scheme secure under $\mathsf{IND\text{-}CCA}_{2,2}$. Interestingly, and to the best of our knowledge, there are no examples of $\mathsf{IND\text{-}CCA}_{1,1}$-secure schemes in the literature; Kirshanova presented in [48] a scheme which was allegedly secure under this notion, but we show in Section 4.4 that this scheme is actually $\mathsf{IND\text{-}CCA}_{1,0}$ secure. As for the rest of the notions, we argue in Section 4.3.4 that they should be considered as degenerate notions due to their asymmetry in the adversarial capabilities. These notions are usually related to attacked schemes: for instance, Koo et al. present in [59] an attack to the scheme from Green and Ateniese [13] that uses the re-encryption oracle during the second phase; thus, assuming no other attack is found, the scheme from Green and Ateniese could be considered as a representative of $\mathsf{IND\text{-}CCA}_{2,1}$ security, as the attack is not valid under this security notion.

4.3 Relations among PRE security notions

As explained in Section 2.3.1, attack models can be combined with security goals to form security notions. We restrict this thesis to the indistinguishability goal. Hence, we derive a parametric set of security notions $\mathsf{IND\text{-}CCA}_{i,j}$, for $i, j \in \{0, 1, 2\}$.

In this section we study the relations between these notions, and in particular, the separations that arise when the adversary is able to learn re-encryption keys from a ciphertext and its re-encrypted value. In addition, we study how these relations are influenced when we consider the multihop property. Finally, we discuss which of these security notions should be considered meaningful, which can be useful when addressing the definitions of security for a PRE scheme.

4.3.1 Implications among PRE security notions

The relations that naturally arise between these security notions can be represented diagrammatically, as depicted in Figure 4.1. This figure shows the hierarchical relations that are consequence from the trivial implications between the security notions. The following result demonstrates the trivial fact that a PRE scheme that is secure under some security notion remains secure when one low-

$$\begin{tikzcd}
\mathsf{IND\text{-}CCA}_{2,2} \arrow[r] \arrow[d] & \mathsf{IND\text{-}CCA}_{2,1} \arrow[r] \arrow[d] & \mathsf{IND\text{-}CCA}_{2,0} \arrow[d] \\
\mathsf{IND\text{-}CCA}_{1,2} \arrow[r] \arrow[d] & \mathsf{IND\text{-}CCA}_{1,1} \arrow[r] \arrow[d] & \mathsf{IND\text{-}CCA}_{1,0} \arrow[d] \\
\mathsf{IND\text{-}CCA}_{0,2} \arrow[r] & \mathsf{IND\text{-}CCA}_{0,1} \arrow[r] & \mathsf{IND\text{-}CCA}_{0,0} = \mathsf{IND\text{-}CPA}
\end{tikzcd}$$

Figure 4.1: Implications among security notions for PRE

ers the re-encryption capabilities of the adversary. This result is represented in Figure 4.1 by the horizontal arrows.

Theorem 4.4 ($\mathsf{IND\text{-}CCA}_{i,j} \Rightarrow \mathsf{IND\text{-}CCA}_{i,j-1}$)**.** *For $i \in \{0,1,2\}$ and $j \in \{1,2\}$, if a PRE scheme is secure in the sense of $\mathsf{IND\text{-}CCA}_{i,j}$, then it is also secure in the sense of $\mathsf{IND\text{-}CCA}_{i,j-1}$.*

Proof of Theorem 4.4. We prove the contrapositive: if a PRE scheme Π is not secure in the sense of $\mathsf{IND\text{-}CCA}_{i,j-1}$, then it is not secure in the $\mathsf{IND\text{-}CCA}_{i,j}$ sense either. The former means that there is an algorithm B that breaks Π in the $\mathsf{IND\text{-}CCA}_{i,j-1}$ sense with non-negligible advantage ε. From B, we can define an algorithm A for breaking Π in the $\mathsf{IND\text{-}CCA}_{i,j}$ sense and which behaves exactly as B, since the oracles provided to B are also provided to A. Adversary A breaks Π in the $\mathsf{IND\text{-}CCA}_{i,j}$ sense with the same non-negligible advantage ε. □

Similarly, the next result follows from considering the decryption oracle, instead of the re-encryption one. This result is represented in Figure 4.1 by the vertical arrows.

Theorem 4.5 ($\mathsf{IND\text{-}CCA}_{i,j} \Rightarrow \mathsf{IND\text{-}CCA}_{i,j-1}$)**.** *For $i \in \{1,2\}$ and $j \in \{0,1,2\}$, if a PRE scheme is secure in the sense of $\mathsf{IND\text{-}CCA}_{i,j}$, then it is also secure in the sense of $\mathsf{IND\text{-}CCA}_{i-1,j}$.*

The proof is analogous to the one for Theorem 4.4.

$$\begin{tikzcd}
 & & \mathsf{IND\text{-}CCA}_{2,2} \arrow[dl, dashed] \arrow[dr, dashed] & & \\
 & \mathsf{IND\text{-}CCA}_{1,2} \arrow[dl, dashed] \arrow[dr, dashed] & & \mathsf{IND\text{-}CCA}_{2,1} \arrow[dl, dashed] \arrow[dr, dashed] & \\
\mathsf{IND\text{-}CCA}_{0,2} \arrow[dr, dashed] & & \mathsf{IND\text{-}CCA}_{1,1} \arrow[dl, dashed] \arrow[dr, dashed] & & \mathsf{IND\text{-}CCA}_{2,0} \arrow[dl, dashed] \arrow[ul, "/" marking, "\textbf{Thm. 4.7}"'] \arrow[llld, "/" marking, "\textbf{Thm. 4.8}"'] \\
 & \mathsf{IND\text{-}CCA}_{0,1} \arrow[dr, dashed] & & \mathsf{IND\text{-}CCA}_{1,0} \arrow[dl, dashed] & \\
 & & \mathsf{IND\text{-}CCA}_{0,0} & &
\end{tikzcd}$$

Figure 4.2: Separations among notions of security.

4.3.2 Separations among PRE Security Notions

Aside from the relations that arise from the initial – and rather trivial – implications among security notions, we are also interested in the separations that exist between them. Figure 4.2 shows the separations that we address in this thesis; the initial implications shown in Figure 4.1 are depicted with light dashed arrows for the sake of clarity. Implications of the form $\mathsf{IND\text{-}CCA}_{i,j} \Rightarrow \mathsf{IND\text{-}CCA}_{i',j'}$ mean that each PRE scheme that is secure in the $\mathsf{IND\text{-}CCA}_{i,j}$ sense, it is also secure in the sense of $\mathsf{IND\text{-}CCA}_{i',j'}$. On the contrary, a separation $\mathsf{IND\text{-}CCA}_{i,j} \not\Rightarrow \mathsf{IND\text{-}CCA}_{i',j'}$ means that there is at least a PRE scheme that is secure in the sense of $\mathsf{IND\text{-}CCA}_{i,j}$ but not in the sense of $\mathsf{IND\text{-}CCA}_{i',j'}$.

The separations that we analyze arise from exploiting the vulnerabilities of PRE schemes that do not satisfy the *private re-encryption keys* property. This property was first described in Remark 2.6 of [45], which stated that an adversary should not be able to learn re-encryption keys from a ciphertext and its re-encrypted value. The scheme analyzed in Section 4.4 is an example of this type of scheme. The following is a more formal and general definition of this property.

Definition 4.6 (Private Re-Encryption Keys)**.** *Let Π be a proxy re-encryption scheme and A a polynomial-time adversary. Let $\Omega = \{\mathcal{O}_{honest}, \mathcal{O}_{corrupt}, \mathcal{O}_{reenc}\}$ be the set of available oracles for A, and pk the public key of a honest user. The scheme Π satisfies the* private re-encryption keys *property if the probability that A computes a valid $rk_{pk \to pk'}$, for some public key pk' of her choice, is negligible.*

Note that this definition is oblivious to the nature of the delegatee's public key pk'. If this public key is from a honest user, then this is enough for forbidding the possibility that the scheme achieves $\mathsf{IND\text{-}CCA}_{2,1}$ or $\mathsf{IND\text{-}CCA}_{2,2}$ security, as shown next in Theorem 4.7. A more worrying situation is that pk' belongs to a corrupt user (so A knows sk'), since this completely rules out the chance of the scheme to even be $\mathsf{IND\text{-}CCA}_{0,1}$ secure. This produces a greater separation between notions, as demonstrated in Theorem 4.8.

The original definition of the private re-encryption keys property by Canetti and Hohenberger contains the outline of a plausible attack to schemes that do not fulfill this property. The following result, conveyed by Theorem 4.7, formalizes this attack in order to establish a separation between security notions for PRE. The basic idea behind this attack is that if the attacker can learn $rk_{pk^* \to pk'}$ through queries to $\mathcal{O}_{reenc}$, where pk' belongs to a honest user, then she can win the security game by locally re-encrypting c^* to c', and then querying $\mathcal{O}_{dec}$ with input (pk', c'). Note that this latter query is legal since the re-encryption key has been leaked by queries to $\mathcal{O}_{reenc}$, so it complies with the restrictions for derivatives of the challenge ciphertext. However, it can be seen that this attack strategy only works when the adversary has access to a decryption oracle during the second phase and to a re-encryption oracle in any phase; that is, it can be only of type $\mathsf{CCA}_{2,1}$ or $\mathsf{CCA}_{2,2}$. As a consequence, it is not possible to conduct this attack in any other attack model (i.e., where a re-encryption oracle is not present or where there is no decryption oracle in phase 2).

In principle, this suggests that this attack strategy produces two separations: one from $\mathsf{IND\text{-}CCA}_{1,2}$ to $\mathsf{IND\text{-}CCA}_{2,1}$ (which corresponds to the case when the attack does not work because the decryption oracle is not available in the second phase), and another from $\mathsf{IND\text{-}CCA}_{2,0}$ to $\mathsf{IND\text{-}CCA}_{2,1}$ (the attack does not work because the re-encryption oracle is never available). In the publication associated to this chapter [120], we formalized each separation as theorems and provided a proof. However, we have notice since then some flaws in these proofs. While the fix for the proof of the second separation is immediate, the first one does not seem to be trivial, since it is not possible to generate the re-encryption keys to leak, and therefore, to define the re-encryption oracle in the $\mathsf{CCA}_{1,2}$ attack model. Luckily, the first separation is not relevant with respect to the findings presented in this chapter (and the associated publication). For this reason, we provide here an amendment for the second separation, expressed by the next theorem.

Theorem 4.7 ($\mathsf{IND\text{-}CCA}_{2,0} \not\Rightarrow \mathsf{IND\text{-}CCA}_{2,1}$)**.** *If there exists a PRE scheme* Π *that is secure in the sense of* $\mathsf{IND\text{-}CCA}_{2,0}$*, then there exists a PRE scheme* Π' *that is secure in the sense of* $\mathsf{IND\text{-}CCA}_{2,0}$ *but which is not secure in the sense of* $\mathsf{IND\text{-}CCA}_{2,1}$*.*

```
Algorithm Enc'($pk, x$)
   $y \leftarrow$ Enc($pk, x$)
   return $(0, y)$

Algorithm Dec'($sk, (y_1, y_2)$)
   If $y_1 = 0$ then return Dec($sk, y_2$)
   else return $\perp$

Algorithm ReEnc'($rk, (y_1, y_2)$)
   If $y_1 = 0$ then return
   $(0,$ ReEnc($rk, y_2$)$)$
   else return $(1, rk)$
```

Figure 4.3: Modified algorithms in Π'

Proof of Theorem 4.7. First, assume that there exists a PRE scheme $\Pi =$ (KeyGen, ReKeyGen, Enc, Dec, ReEnc) that is secure in the sense of IND-CCA$_{1,2}$; otherwise, the theorem is vacuously true. We now define scheme Π' =(KeyGen', ReKeyGen', Enc', Dec', ReEnc'), where KeyGen' = KeyGen, ReKeyGen' = ReKeyGen, and Enc', Dec', and ReEnc') are defined as shown in Figure 4.3. Informally, Π' simply leaks the re-encryption key during a re-encryption when the first component is different from 0. That is, we are constructing a proxy re-encryption scheme which does not fulfill the property of private re-encryption keys. A re-encryption oracle for this scheme should behave in the same way (i.e., it cannot output a random value instead of the corresponding re-encryption key), since otherwise it could be immediately detected by the adversary by verifying the correctness of the re-encryption.

Claim 4.7.1. Π' *is not secure in the sense of IND-CCA$_{2,1}$.*

Proof of Claim 4.7.1. We define an adversary $A = (A_1, A_2)$ that breaks Π' in the sense of IND-CCA$_{2,1}$, with probability 1 and in polynomial time, as follows. As we are working in the selective model, A_1 receives the target public key pk^* at the beginning of the game. In addition, A_1 gets a honest public key pk_h through oracle $\mathcal{O}_{honest}$. According to the definition of the CCA$_{2,1}$ model, $\mathcal{O}_{reenc}$ is available during the first phase. A_1 queries this oracle with $(pk^*, pk_h, (1, z))$, for a randomly generated ciphertext z. Note that this query is legitimate, since challenge ciphertext has not been generated yet. A_1 receives $rk_{pk^* \to pk_h}$ as part of the output of the re-encryption oracle, includes it in the state s and outputs (m_0, m_1, s). At the guess phase, the challenge ciphertext is $(0, c^*)$, so A_2 receives $(m_0, m_1, s, (0, c^*))$ and extracts $rk_{pk^* \to pk_h}$ from s. Now it computes ReEnc'$(rk_{pk^* \to pk_h}, (0, c^*)) = (0, c_h)$. Finally, since the decryption oracle $\mathcal{O}_{dec}$ is available during the second phase, according to the definition of the CCA$_{2,1}$ model, it makes the query $\mathcal{O}_{dec}(pk_h, (0, c_h))$ to recover the message m_δ and determines δ with probability 1. This query is legitimate because $(pk_h, (0, c_h))$ cannot be considered a derivative of the challenge, as the adversary did not obtain the re-encryption key from $\mathcal{O}_{rkgen}$.

Claim 4.7.2. Π' *is secure in the sense of* $\textsf{IND-CCA}_{2,0}$

Proof of Claim 4.7.2. The proof is by contradiction. Suppose that Π' is not secure in the sense of $\textsf{IND-CCA}_{2,0}$. Then, there exists an algorithm B that breaks Π' in the $\textsf{IND-CCA}_{2,0}$ sense with non-negligible advantage ε. From B, we can define an adversary $A = (A_1, A_2)$ that breaks Π in the $\textsf{IND-CCA}_{2,0}$ sense as follows. Algorithm A_1 simply outputs the same results than B_1 for input pk^*, while algorithm A_2 takes input (m_0, m_1, s, c^*) and outputs the result d from calling B_2 with parameters $(m_0, m_1, s, (0, c^*))$. The computations of B are done by A simulating B's decryption oracle $\mathcal{O}^B_{dec}$ using its own oracle $\mathcal{O}^A_{dec}$, as follows: on input $(pk, (c_1, c_2))$ to $\mathcal{O}^B_{dec}$, A verifies that $c_1 = 0$ and outputs $\mathcal{O}^A_{dec}(pk, c_2)$; otherwise, it outputs $\perp$. The simulation is perfect, and A breaks Π in the $\textsf{IND-CCA}_{2,0}$ sense with the same advantage ε.

□

We present now a different attack strategy, which produces a greater separation between PRE security notions. Suppose that the adversary is able to extract $rk_{pk^* \to pk_x}$ through calls to $\mathcal{O}_{reenc}$, where pk_x belongs to a corrupt user. Then, she wins the security game by locally re-encrypting the challenge ciphertext c^* to c_x, which she can decipher using sk_x. The only oracle query to $\mathcal{O}_{reenc}$ is legal since it does not involve a derivative of the challenge ciphertext. It can be seen that this attack strategy can only work when there is a re-encryption oracle available (i.e., any attack model above $\textsf{CCA}_{0,1}$), thus producing a separation between $\textsf{IND-CCA}_{2,0}$ and $\textsf{IND-CCA}_{0,1}$.

Theorem 4.8 ($\textsf{IND-CCA}_{2,0} \not\Rightarrow \textsf{IND-CCA}_{0,1}$). *If there exists a PRE scheme* Π *that is secure in the sense of* $\textsf{IND-CCA}_{2,0}$*, then there exists a PRE scheme* Π' *that is secure in the sense of* $\textsf{IND-CCA}_{2,0}$ *but which is not secure in the sense of* $\textsf{IND-CCA}_{0,1}$.

Proof of Theorem 4.8. The proof in this case is very similar to that of Theorem 4.7, except that we now prove that Π' is not secure in the sense of $\textsf{IND-CCA}_{0,1}$.

Claim 4.8.1. Π' *is not secure in the sense of* $\textsf{IND-CCA}_{0,1}$.

Proof of Claim 4.8.1. This proof is similar to the one of Claim 4.7.1. We define an adversary $A = (A_1, A_2)$ that breaks Π' in the sense of $\textsf{IND-CCA}_{0,1}$, with probability 1 and in polynomial time, as follows. Since we are working in the selective model, A_1 receives the target public key pk^* at the beginning of the game. In addition, A_1 gets a pair of corrupt public and private keys (pk_x, sk_x) through oracle $\mathcal{O}_{corrupt}$. Recall that, according to the definition of the $\mathsf{CCA}_{0,1}$ attack model, the only additional oracle available in the phase 1 is $\mathcal{O}_{reenc}$. A_1 queries this oracle with $(pk^*, pk_x, (1, z))$, for a randomly generated ciphertext z. A_1 receives $rk_{pk^* \to pk_x}$ as part of the output of the re-encryption oracle, includes it in the state s and outputs (m_0, m_1, s). In phase 2, the challenge ciphertext is $(0, c^*)$, so A_2 receives $(m_0, m_1, s, (0, c^*))$ and extracts $rk_{pk^* \to pk_x}$ from s. Now it computes $\mathsf{ReEnc'}(rk_{pk^* \to pk_x}, (0, c^*)) = (0, c_x)$. Finally, it evaluates $\mathsf{Dec'}(sk_x, (0, c_x))$ to recover the message m_δ and determines δ with probability 1.

$\square$

The last theorem entails the following corollary, which formalizes the idea that there are PRE schemes that are secure in the sense of $\mathsf{IND\text{-}CCA}_{i,0}$, but fail once a re-encryption oracle is introduced.

Corollary 4.9. *If there exists a PRE scheme Π that is secure in the sense of $\mathsf{IND\text{-}CCA}_{i,0}$, for $i \in \{0, 1, 2\}$, then there exists a PRE scheme Π' that is secure in the sense of $\mathsf{IND\text{-}CCA}_{i,0}$ but which is not secure in the sense of $\mathsf{IND\text{-}CCA}_{i,j}$, for $j \in \{1, 2\}$.*

The scheme described in Section 4.4 illustrates this corollary, since it is $\mathsf{IND\text{-}CCA}_{1,0}$ secure (i.e., it does not consider a re-encryption oracle), but not $\mathsf{IND\text{-}CCA}_{1,1}$ secure, as we show how the attacker can use the re-encryption oracle to win the security game. This attack further supports the separation between PKE-based notions (i.e., those that do not consider a re-encryption oracle) and the rest.

4.3.3 Relations for multihop PRE

In this section we study the effect on the security notions that appears from the following observation: assuming a proper re-encryption oracle (that is, one that enables re-encryption from a honest party to a corrupt one), then the adversary can construct a decryption oracle using the re-encryption one. She solely has to re-encrypt ciphertexts to a corrupt user and decrypt afterwards, since she knows the secret key. More formally, let x be a corrupt user controlled by the adversary, then for all public keys pk_i and all ciphertexts c, where (pk_i, c) is not a derivative of (pk^*, c^*), the decryption oracle can be simulated by the adversary as follows:

$$\mathcal{O}_{dec}(pk_i, c) = \mathsf{Dec}(sk_x, \mathcal{O}_{reenc}(pk_i, pk_x, c)) \tag{4.1}$$

This equivalence only holds in the multihop case, since when c is already a re-encryption from other ciphertext, then it is not possible to re-encrypt it again; that is, for single-hop PRE schemes this equivalence only holds when the ciphertext has not been re-encrypted (i.e., a second-level ciphertext [11]). On the contrary, multihop PRE schemes permit to re-encrypt ciphertexts indefinitely. Note also that the query $\mathcal{O}_{reenc}(pk_i, pk_x, c)$ is legal as long as (pk_i, c) is not a derivative of (pk^*, c^*). This result allows us to establish additional implications between security notions, for the case of multihop PRE.

In the $\mathsf{IND\text{-}CCA}_{0,1}$ scenario, the re-encryption oracle is only provided in Phase 1 and no decryption oracle is available. For a multihop scheme, it is easy to check that Equation 4.1 describes a valid simulation of the decryption oracle during Phase 1. Since the challenge ciphertext has not yet been generated, neither the re-encryption nor decryption oracles have any kind of restriction on the input ciphertext. Thus, if a re-encryption oracle for Phase 1 is provided, the decryption oracle becomes redundant, since it can be easily simulated by the adversary. Therefore, $\mathsf{IND\text{-}CCA}_{0,1} \Rightarrow \mathsf{IND\text{-}CCA}_{1,1}$.

Now we assume that the re-encryption oracle is also available in Phase 2. In this case, there is a restriction on the input ciphertexts, as described in Section 4.2, so it cannot be a derivative of the challenge ciphertext. Note that the very same restriction is applicable to the decryption oracle during the same phase. As in the previous case, Equation 4.1 describes a valid simulation of the decryption oracle, this time for both phases. Similarly, if a re-encryption oracle is provided until Phase 2, then the decryption oracle becomes redundant, as it can be simulated by the adversary using the re-encryption oracle. Therefore, $\mathsf{IND\text{-}CCA}_{0,2} \Rightarrow \mathsf{IND\text{-}CCA}_{2,2}$.

These two results, together with the fact that $\mathsf{IND\text{-}CCA}_{1,1} \Rightarrow \mathsf{IND\text{-}CCA}_{0,1}$ and $\mathsf{IND\text{-}CCA}_{2,2} \Rightarrow \mathsf{IND\text{-}CCA}_{0,2}$, imply that $\mathsf{IND\text{-}CCA}_{1,1} \equiv \mathsf{IND\text{-}CCA}_{0,1}$ and $\mathsf{IND\text{-}CCA}_{2,2} \equiv \mathsf{IND\text{-}CCA}_{0,2}$. Therefore, the previous set of relations between security notions can be simplified for the multihop case, as shown in Figure 4.4. For clarity, the notions of $\mathsf{IND\text{-}CCA}_{0,2}$ and $\mathsf{IND\text{-}CCA}_{1,2}$ have been assimilated into $\mathsf{IND\text{-}CCA}_{2,2}$, and $\mathsf{IND\text{-}CCA}_{0,1}$ into $\mathsf{IND\text{-}CCA}_{1,1}$.

These results strengthen the idea that the re-encryption oracle is somewhat more powerful than the decryption oracle, and that in some cases, renders it completely redundant.

$$\begin{tikzcd}
\text{IND-CCA}_{2,2} \arrow[dd] \arrow[dr] & & \\
& \text{IND-CCA}_{2,1} \arrow[dl] \arrow[dr] & \\
\text{IND-CCA}_{1,1} \arrow[dd] \arrow[dr] & & \text{IND-CCA}_{2,0} \arrow[ll, "/" marking] \arrow[ul, "/" marking] \arrow[dl] \\
& \text{IND-CCA}_{1,0} \arrow[dl] & \\
\text{IND-CCA}_{0,0} & &
\end{tikzcd}$$

Figure 4.4: Relations among security notions for multihop PRE

4.3.4 Discussion

From the parametric family of attack models for PRE we propose, we have developed in this thesis a set of security notions and showed the relations among them. However, that does not necessarily mean that all the notions are equally meaningful for the context of PRE. Recall that one of the main characteristics of PRE is that re-encryption capabilities are granted to some semi-trusted entity that acts as *proxy*, transforming ciphertexts from one public key to another. In this case, it seems reasonable to think of the proxy as a re-encryption oracle, and since it is a semi-trusted entity, its availability for an adversary may be easier than in the case of the decryption oracle. In fact, there are several examples of devised applications of PRE where re-encryption is performed as a service by an online semi-trusted entity, whereas decryption capabilities are retained by the users [11, 101].

Hence, attack models in which the decryption capabilities are greater than for re-encryption, and particularly, those where a decryption oracle is provided but not a re-encryption one ($\mathsf{CCA}_{2,0}$ and $\mathsf{CCA}_{1,0}$), do not seem to be appropriate for shaping security in scenarios where proxy re-encryption is applied. The scheme we analyze in Section 4.4 is an example of construction that is only secure under this kind of attack model. In our opinion, it seems difficult to strongly justify any asymmetry between the adversarial capabilities concerning decryption and re-encryption. For that reason, we consider that the representative attack models for PRE are $\mathsf{CCA}_{2,2}$, $\mathsf{CCA}_{1,1}$, and $\mathsf{CCA}_{0,0}$ (which can be denoted as CPA). The rest of attack models (and therefore, security notions) can be considered as the formalization of transitional models, but that fail to properly capture adversarial capabilities of meaningful scenarios. Nevertheless, these transitional models can be of use when reasoning about security notions, and as intermediate milestones during the design of PRE schemes, as proposed in the next chapter.

In the next section, we describe an example of a recent PRE scheme, which is allegedly "CCA1-secure". However, our analysis shows that, according to our definitions of attack models for PRE, this scheme is in fact $\mathsf{IND\text{-}CCA}_{1,0}$ secure, since it does not consider a re-encryption oracle. Furthermore, it cannot achieve a better security notion since it leaks the re-encryption keys when the re-encryption oracle is introduced. This impossibility is a consequence of Theorem 4.8.

4.4 Security Analysis of a Recent "CCA1-secure" PRE scheme

The scheme analyzed in this section was proposed by Kirshanova at PKC 2014 [48]. Although, the scheme is said to be "CCA1-secure", its security proof does not provide meaningful re-encryption capabilities to the adversary. An analysis of the security proof reveals that this scheme is actually secure under $\mathsf{IND\text{-}CCA}_{1,0}$, because the decryption oracle is available until the challenge, but it lacks of a re-encryption oracle. As discussed in Section 4.3.4, we do not consider this as a meaningful notion for PRE because of the asymmetry between the adversarial capabilities concerning decryption and re-encryption. In fact, $\mathsf{IND\text{-}CCA}_{1,0}$ is, together with $\mathsf{IND\text{-}CCA}_{0,1}$, the closest notion to $\mathsf{IND\text{-}CCA}_{0,0} \equiv \mathsf{IND\text{-}CPA}$. In this section we describe an attack to this scheme under a more significant notion, namely $\mathsf{IND\text{-}CCA}_{1,1}$. That is, this scheme cannot achieve $\mathsf{IND\text{-}CCA}_{1,1}$ in its current form. The main idea behind this attack is that failure to fulfill the private re-encryption keys property leads to insecure schemes when one considers the re-encryption oracle. The proposed attack is applicable even when a re-encryption oracle is provided only before the challenge phase. In addition, such oracle must not have strong restrictions, since this would imply an overly weak security model, as discussed in Section 4.2. In particular, it should be possible to ask for the re-encryption of ciphertexts that are unrelated to c^*, from the target user to a corrupt one.

4.4.1 Description of the scheme

First we will describe the scheme from Kirshanova. This is one of the first PRE schemes based on the hardness of lattice problems, specifically, the Learning

With Errors (LWE) [32] and Short Integer Solution (SIS) [38] problems. Some details, such as a comprehensive characterization of the random distributions used and the validity checks in the decryption algorithm, are omitted for the sake of clarity. We refer the reader to [48] for a complete description of the scheme, and to [40] for more details on the original PKE scheme upon Kirshanova's scheme is based.

Let n be the security parameter. The modulus $q = poly(n)$ is a large prime power and $k = \log q$. Set $\bar{m} = O(nk)$ and $m = \bar{m} + 2nk$. Let D_R be a random distribution over $\mathbb{Z}^{\bar{m}\times nk}$ that samples the secret keys R and D_e a random distribution over $\mathbb{Z}^m$ that samples error vectors e; details about these distributions are omitted. Invert_O and Sample_O are two algorithms for inverting the LWE-function and sampling preimages from the SIS-function, respectively. Let $G \in \mathbb{Z}_q^{n\times nk}$ be a specially-constructed public matrix that makes these algorithms efficient. The scheme is as follows:

- $\mathsf{KeyGen}(n)$: choose $A_0 \leftarrow \mathbb{Z}_q^{n\times \bar{m}}$, $R_1, R_2 \leftarrow D_R$ and an invertible matrix $H \leftarrow \mathbb{Z}_q^{nk\times nk}$. Next, define $A_1 = -A_0R_1 \in \mathbb{Z}_q^{n\times nk}$ and $A_2 = -A_0R_2 \in \mathbb{Z}_q^{n\times nk}$, and compose the matrix $A = [A_0|A_1|A_2] \in \mathbb{Z}_q^{n\times m}$. The public key is the pair $pk = (A, H)$, while the secret key is matrix $sk = [R_1|R_2] \in \mathbb{Z}^{\bar{m}\times 2nk}$.

- $\mathsf{Enc}(pk = ([A_0|A_1|A_2], H), m \in \{0,1\}^{nk})$: choose a non-zero invertible matrix H_u, and a vector $s \leftarrow \mathbb{Z}_q^n$. Set $A_u = [A_0|A_1 + HG|A_2 + H_uG]$. Sample error vector $e \leftarrow D_e$. Compute $b^t = 2(s^tA_u \mod q) + e^t + (0, 0, enc(m))^t \mod 2q$, where the first zero vector has dimension $\bar{m}$, the second has dimension nk and enc is an encoding function. Output the ciphertext $c = (H, b) \in \mathbb{Z}_q^{n\times n} \times \mathbb{Z}_{2q}^m$.

- $\mathsf{Dec}(pk = ([A_0|A_1|A_2], H), sk = [R_1|R_2], c = (H_u, b))$: Using matrix H_u compute $A_u = [A_0|A_1 + HG|A_2 + H_uG]$. With the secret key call algorithm $\mathsf{Invert}_O([R_1|R_2], A_u, b \mod q, H_u)$. As output we receive two vectors $z \in \mathbb{Z}_q^n$ and $e \in \mathbb{Z}_q^m$ that satisfy $b^t = z^tA + e^t \mod q$. Let $v = b - e \mod 2q$. Compute
$$v^t \begin{bmatrix} R_1 & R_2 \\ I & 0 \\ 0 & I \end{bmatrix} \mod 2q$$
and apply enc^{-1} to the last nk coordinates.

- $\mathsf{ReKeyGen}(pk = ([A_0|A_1|A_2], H), sk = [R_1|R_2], pk' = ([A_0'|A_1'|A_2'], H'))$: Let $Y = [A_0'|A_1' + H'G|A_2' - A_2]$ and y_i be the i-th column of Y. Execute $\mathsf{Sample}_O(y_i, [A_0|A_1], R_1, H)$ for each column vector y_i and concatenate the column vector outputs to form matrix X. It can be seen that this matrix satisfies that $[A_0|A_1]X = Y$. Parse matrix X as $[X_0|X_1|X_2]$, where the block $X_0 \in \mathbb{Z}^{(\bar{m}+nk)\times\bar{m}}$ is the output corresponding to the first part of Y, $X_1 \in \mathbb{Z}^{(\bar{m}+nk)\times nk}$ to the second one, and $X_2 \in \mathbb{Z}^{(\bar{m}+nk)\times nk}$ to the last one [1]. Finally, output the re-encryption key $rk_{pk\to pk'}$:

$$rk_{pk\to pk'} = \begin{bmatrix} X_0 & X_1 & X_2 \\ 0 & 0 & I \end{bmatrix}$$

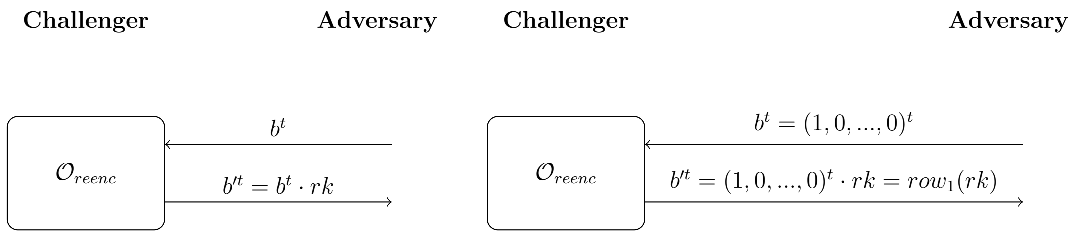
- $\mathsf{ReEnc}(rk_{pk\to pk'}, c = (H_u, b))$: to change the underlying public key in the ciphertext component b, compute $b'^t = b^t \cdot rk_{pk\to pk'}$. Finally, output $c' = (H_u, b')$.

4.4.2 Attack under the $\mathsf{IND\text{-}CCA}_{1,1}$ setting

In this section we describe an attack to Kirshanova's scheme under the $\mathsf{IND\text{-}CCA}_{1,1}$ setting. Once we give the attacker the capability of using a re-encryption oracle, and given the fact that the re-encryption function in the Kirshanova PRE scheme does not check for the validity of its inputs, then the attacker can query the re-encryption oracle with a series of deceptive ciphertexts, which ultimately leak the re-encryption key from the target user to a corrupt one. Note that these queries are completely legal, as the input ciphertexts are not related to the challenge ciphertext (in fact, when the re-encryption key is leaked the challenge ciphertext has not been produced yet). Once the attacker asks for the challenge ciphertext, she can re-encrypt it for a corrupt user under her control, and distinguish the original challenge message. This attack constitutes an instantiation of one of the generic attacks used for separating security notions described in Section 4.3, namely the one used in the proof of Theorem 4.8.

Assuming a $\mathsf{IND\text{-}CCA}_{1,1}$ setting, then we can construct a $\mathsf{CCA}_{1,1}$ attack (i.e., only consulting the decryption and re-encryption oracles before the challenge) as follows. First, as we are working in the selective model, the challenger gives the target public key pk^* to the attacker at the beginning of the game. In addition, the attacker asks for a pair of corrupt public and private keys (pk_x, sk_x) to oracle $\mathcal{O}_{corrupt}$. Next, recall that the re-encryption function simply multiplies the original ciphertext by the re-encryption key, as shown in Equation 4.2 and depicted by Figure 4.5a, without performing any kind of prior or subsequent checks:

[1]In the original scheme, matrix X is further decomposed into smaller blocks (e.g., X_0 is decomposed in X_{00} and X_{01}), but this is not necessary for our analysis.



(a) The re-encryption oracle (b) Using a unit vector as input

Figure 4.5: Leaking re-encryption keys

$$\begin{aligned}[b'^t_0|b'^t_1|b'^t_2] &= [b^t_0|b^t_1|b^t_2] \cdot \begin{bmatrix} X_0 & X_1 & X_2 \\ 0 & 0 & I \end{bmatrix} \\ &= [b^t_0|b^t_1] \cdot [X_0|X_1|X_2] + [0|0|b^t_2] \end{aligned} \quad (4.2)$$

The attacker can take advantage from the behavior of the re-encryption oracle and ask for the re-encryption of specially-crafted "trap" ciphertexts $b^t_i = [b^t_{i,0}|b^t_{i,1}|b^t_{i,2}]$, so that $[b^t_{i,0}|b^t_{i,1}] = \hat{u}_i$, where $\hat{u}_i \in \mathbb{Z}_{2q}^{1\times\bar{m}+nk}$ is the i-th row unit vector, for $1 \leq i \leq \bar{m} + nk$, and $b^t_{i,2}$ is chosen randomly in $\mathbb{Z}_{2q}^{1\times nk}$. Next, the attacker proceeds with $\bar{m} + nk$ queries to $O_{reenc}(pk^*, pk_x, (H_i, b_i))$, for random H_i, to obtain re-encrypted ciphertexts (H_i, b'_i). It can be seen that, after removing the term $[0|0|b^t_{i,2}]$, the vector b'_i from the re-encrypted ciphertexts will have the form $b'_i = \hat{u} \cdot [X_0|X_1|X_2]$, which corresponds to the i-th row of the re-encryption key. A simplified illustration of this stage of the attack is shown in Figure 4.5b. Finally, the attacker simply stacks the results together to obtain the matrix X of the re-encryption key $rk_{pk^*\to pk_x}$. Note that, originally, re-encryption keys are defined over $\mathbb{Z}$, but with these queries she actually obtains its representation in $\mathbb{Z}_{2q}$; nonetheless, the obtained re-encryption key is equivalent for making computations, and thus, equally valid.

The knowledge of this re-encryption key gives the attacker enough power to decrypt the challenge ciphertext during the second phase of the game, and hence distinguish the original message. Note that the queries to the re-encryption oracle can be carried out before the challenge ciphertext is generated, so it is complies with the $\mathsf{IND\text{-}CCA}_{1,1}$ security notion (i.e., it cannot be considered an adaptive attack).

This attack can be refined as follows. Assume that the re-encryption oracle could "detect" the aforementioned queries as trap ciphertexts and return a random output (althought that could mean producing invalid re-encryptions for valid ciphertexts that happen to have the same form). In this case, the attacker may be unable to detect the situation, since she does not know the underlying message of the trap ciphertexts (i.e., she does not know whether or not the trap ciphertexts are valid encryptions under target public key pk^*).

We now describe another way of extracting the sought-after re-encryption key, by concealing these trap vectors in the following way. The attacker produces m ciphertexts $(H_i, p_i^t) = \mathsf{Enc}(pk^*, m_i)$, for random $m_i \in \{0,1\}^{nk}$, and constructs a matrix $P \in \mathbb{Z}_{2q}^{m\times m}$, so that each p_i^t is the i-th row of P. Note that, by the hardness assumption of the encryption scheme, the distribution of vectors p_i^t is within negligible statistical distance from the uniform distribution, as described by the LWE hardness assumption. Hence, with overwhelming probability, P is an invertible matrix, and $P^{-1}P = I$. In the case that P were not invertible, the attacker could simply "resample" more ciphertexts and produce a new P.

Next, the attacker makes m queries $\mathcal{O}_{reenc}(pk^*, pk_x, (H_i, p_i^t))$, with $1 \leq i \leq m$. By the definition of the re-encryption function, the outcome of each query contains the vector $p_i'^t = p_i^t \cdot rk_{pk^*\to pk_x}$. Let $P' \in \mathbb{Z}_{2q}^{m\times m}$, so that each $p_i'^t$ is the i-th row of P'. Hence, it can be seen that $P' = P \cdot rk_{pk^*\to pk_x}$. Finally, she simply computes $rk_{pk^*\to pk_x} = P^{-1} \cdot P' = P^{-1} \cdot P \cdot rk_{pk^*\to pk_x}$. Using this re-encryption key, the attacker decrypts the challenge ciphertexts as described before.

4.4.3 Analysis of the attack

There are two main reasons that explain why this attack is possible. The first is related to the construction itself, whereas the second concerns the underlying security model.

Firstly, the scheme leaks the re-encryption key through queries to the re-encryption oracle. That is, it fails to satisfy the private re-encryption keys property. This characteristic is what makes possible for the attacker to learn the re-encryption key from the target user to one under her control, thus winning the security game. This is a consequence of how the re-encryption function is constructed, which is simply the multiplication of a ciphertext by the re-encryption key. Thus, when the input ciphertexts are carefully chosen, it is possible to recover the desired re-encryption key. A possible solution to thwart this attack consists on adding a small noise after the multiplication. Another issue with the re-encryption function is that it is performed without any kind of validity check on the input. This poses an interesting problem since this validity check should be performed without the proxy seeing the original plaintexts. The usual solution to this problem is to make the ciphertexts *publicly verifiable* [54]. A consequence of this property is that it prevents the proxy from acting as an oracle [121], which is exactly the weak point of the analyzed scheme.

Secondly, the adversarial model used in this scheme (i.e., $\mathsf{CCA}_{1,0}$) does not provide access to a re-encryption oracle during the security game, so the scheme is oblivious to the fact that it is insecure when one considers access to this oracle, as in $\mathsf{CCA}_{1,1}$. As discussed in Section 4.3.4, the only attack models deemed meaningful for PRE are $\mathsf{CCA}_{2,2}$, $\mathsf{CCA}_{1,1}$, and $\mathsf{CCA}_{0,0}$, as there should be very strong reasons for justifying any asymmetry between the adversarial capabilities concerning decryption and re-encryption. In addition, the re-encryption capabilities should be wide enough to allow the adversary to ask for the re-encryption of ciphertexts that are unrelated to the challenge ciphertext, for any pair of users.

4.5 Summary

In this chapter we have studied the notions of security for proxy re-encryption by identifying a parametric family of attack models that not only considers the availability of the decryption oracle (as in PKE), but also the re-encryption oracle. Although this seems to be a rather natural step, to the best of our knowledge, it has not been explicitly considered before. This parametric family of attack models for PRE allows us to define a collection of security notions, whose relations we analyze. This type of study is necessary to achieve a better understanding of the definitions of security upon which design proxy re-encryption schemes. For instance, one immediate application of this work is found in Section 5.2, where we use it to reason about the construction of generic CCA-secure transformations in the context of proxy re-encryption.

Part of this chapter has been concerned with studying separations between PRE notions of security. In particular, we have established separations between security notions by exploiting the private re-encryption keys property, or more accurately, the failure to fulfill this property. The consequence of these separations is that schemes that leak re-encryption keys through queries to the re-encryption oracle cannot achieve strong security notions. These results strengthen the idea that meaningful chosen-ciphertext attacks for PRE should consider both oracles (decryption and re-encryption).

In addition, these separation results have been illustrated with an example of a recent PRE scheme from PKC 2014 [48] that is said to be "CCA1-secure", but that fails to achieve a meaningful notion of security for PRE, since we construct a chosen-ciphertext attack based on queries to the re-encryption oracle.

Chapter 5

New Proxy Re-Encryption Constructions

One of the main interests that drive the work on this thesis is to investigate more concrete aspects of proxy re-encryption schemes, such as those related to performance and security constructions. This chapter is divided according these two aspects.

In the first half we explore the use of lattice-based cryptography for constructing more efficient PRE schemes. In particular, we propose new proxy re-encryption schemes based on NTRU [16], a widely known lattice-based cryptosystem. Lattice-based cryptography is a promising field, not only due to its potential regarding post-quantum security, but also because its performance can be outstanding in some cases. We provide two different schemes: the first one is based on the conventional NTRU cryptosystem and, although it lacks a security proof, remains as efficient as its predecessor; the second one is based on a variant of NTRU proposed by Stehlé and Steinfeld [122], which is proven CPA-secure under the Ring-LWE assumotion. To the best of our knowledge, our proposals are the first proxy re-encryption schemes to be based on the NTRU primitive. In addition, we provide experimental results to demonstrate the efficiency of our proposal, as well as a comparison with previous proxy re-encryption schemes, which confirms that our first scheme outperforms the rest by an order of magnitude.

The second half of this chapter has a different focus, as it is dedicated to the construction of CCA-secure schemes. In the case of traditional public-key encryption, there are several generic methods for achieving CCA-secure schemes from weakly secure cryptosystems, such as the Fujisaki-Okamoto [71, 17] and REACT [18] transformations. In PRE, however, this process is completely ad-hoc for each scheme. Therefore, it would be desirable to count on analogous constructions that allow PRE schemes to "bootstrap" security by means of generic transformations. In this section, we study the adaptation of some of these transformations to proxy re-encryption and find both positive and negative results: although direct application is not possible (as it leads to flawed constructions), we devise viable extensions that enable to enhance the security of existing schemes.

5.1 Proxy Re-Encryption Schemes Based on NTRU

5.1.1 Introduction

Since the introduction of proxy re-encryption by Blaze et al. in 1998 [43], many PRE schemes have been proposed. The vast majority of these schemes are based on bilinear pairings, which have several drawbacks. For instance, pairings are known to be computationally intensive operations. With respect to security, pairing-based proxy re-encryption schemes ultimately rely on the hardness of the discrete logarithm problem. This poses a problem in the case that efficient cryptanalytic attacks against the discrete logarithm problem are developed or quantum computation becomes practical. This problem is also applicable to other schemes not based on pairings but that also depend on the discrete logarithm or on integer factorization assumptions.

Apart from bilinear pairings, the use of other foundations for constructing proxy re-encryption schemes has received little attention. Recently, some lattice-based proxy re-encryption schemes have been proposed [46, 47]. Lattice-based cryptography is a promising field, chiefly because of its role in post-quantum cryptography, but in some cases, also for its performance. As a prime example of widely-accepted and efficient lattice-based cryptosystem we find NTRU, which was introduced in 1996 by Hoffstein, Pipher and Silverman [16]. Since then, it has remained as the most practical lattice-based public-key cryptosystem, at the point of being standardized by IEEE Std 1363.1-2008 [79] and ANSI X9.98-2010 [123]. One of the main reasons of NTRU's popularity is its overwhelming performance with respect to traditional public-key cryptosystems. For instance, optimized GPU versions of NTRU proved to be up to 1000 times faster than RSA and 100 times faster than ECC [124]. These results are explained by the simplicity of its basic underlying operation, which is the convolution, i.e., the polynomial multiplication. NTRU uses polynomials with relatively small coefficients, so multiplications can be efficiently performed, even in constrained devices [125]. Another implication of this is that the size of the keys is relatively compact, when compared to other lattice-based schemes. In addition, the convolution operation can even be parallelized, which is very convenient as multicore and GPU processors are becoming more relevant nowadays.

One of the main drawbacks of NTRU is that it lacks a formal security proof, so its security stems from the practicality of best known attacks. This has meant that, over the years, NTRU has followed a somewhat "break-and-repair" approach, where advances in attacks have stimulated changes in the scheme and its set of parameters, although it has remained essentially the same. In order to advance towards solving this gap, Stehlé and Steinfeld proposed in 2011 a provable-secure variant of NTRU [73, 122], whose security is based on the assumed hardness of lattice problems.

In this chapter, we propose new bidirectional proxy re-encryption schemes based on NTRU. Our first result, NTRUReEncrypt, is a slight modification of the conventional NTRU encryption scheme; this proposal extends the original scheme in order to support re-encryption, thus it remains just as efficient. However, it lacks of a formal security proof, as does NTRU. For this reason, we also present PS-NTRUReEncrypt, a provably-secure version based on the variant of the NTRU primitive proposed by Stehlé and Steinfeld in [73, 122]. To the best of our knowledge these are the first proxy re-encryption schemes based on NTRU, and one of the first to be based on lattices. In addition, we complement our proposal with an experimental comparison of the performance of several proxy re-encryption schemes, which confirms that our first scheme outperforms the rest by an order of magnitude.

5.1.2 Definitions

In this section, we provide some basic definitions relevant to our proposal. Before starting, we note that we will reuse the generic syntax definition given in Section 3.2.1. Firstly, we formulate the concept of correctness, based on the one given by [45]; in this case, as these kinds of schemes are usually multihop, we refer to multihop correctness (i.e., ciphertexts that can be re-encrypted multiple times). Secondly, we specify the security definition of a bidirectional CPA-secure proxy re-encryption scheme; note that this is a particular instantiation of the security notions provided in the previous section, namely, IND-CCA$_{0,0}$.

Definition 5.1 (Multihop Correctness)**.** *A bidirectional PRE scheme (Setup, KeyGen, ReKeyGen, Enc, ReEnc, Dec) is multihop correct with respect to plaintext space $\mathcal{M}$ if:*

- *For all (pk_A, sk_A) output by* KeyGen *and all messages $M \in \mathcal{M}$, it holds that*

$\mathsf{Dec}(sk_A, \mathsf{Enc}(pk_A, M)) = M.$

- *For any sequence of pairs* (pk_i, sk_i) *output by* KeyGen*, with* $0 \le i \le N$*, all re-encryption keys* $rk_{j \to j+1}$ *output by* $\mathsf{ReKeyGen}(sk_j, sk_{j+1})$*, with* $j < N$*, all messages* $M \in \mathcal{M}$*, and all ciphertexts* C_1 *output by* $\mathsf{Enc}(pk_1, M)$*, it holds that:*

$$\mathsf{Dec}(sk_N, \mathsf{ReEnc}(rk_{N-1\to N}, ...\mathsf{ReEnc}(rk_{1\to 2}, C_1))) = M$$

If for any $M \in \mathcal{M}$ *correctness holds only with probability 1 minus a negligible quantity, we say that the scheme is correct with respect to* $\mathcal{M}$*.*

Definition 5.2 (Bidirectional PRE CPA-security game)**.** *Let* λ *be the security parameter,* $\mathcal{A}$ *an adversary, and* $\mathcal{I}_H, \mathcal{I}_C$ *the sets of indices of honest and corrupt users, respectively. The game consists of an execution of* $\mathcal{A}$ *with the following oracles, which can be invoked multiple times in any order, subject to the constraints below:*

Phase 0: The challenger takes a security parameter 1^λ*, obtains global parameters* $params \leftarrow \mathsf{Setup}(1^\lambda)$ *and initializes sets* $\mathcal{I}_H, \mathcal{I}_C$ *to* $\varnothing$*. The challenger generates the public key* pk^* *of target user* i^**, adds* i^* *to* $\mathcal{I}_H$*, and sends* pk^* *to the adversary.*

Phase 1:

- *Honest key generation* $\mathcal{O}_{honest}$*: The oracle obtains a new keypair* $(pk_i, sk_i) \leftarrow \mathsf{KeyGen}(\lambda)$*, adds index* i *to* $\mathcal{I}_H$*, and returns the public key* pk_i*.*
- *Corrupted key generation* $\mathcal{O}_{corrupt}$*: The oracle obtains a new keypair* $(pk_i, sk_i) \leftarrow \mathsf{KeyGen}(n)$*, adds index* i *to* $\mathcal{I}_C$*, and returns the pair* (pk_i, sk_i)*.*

Phase 2:

- *Re-encryption key generation* $\mathcal{O}_{rkgen}$*: On input a pair of public keys* (pk_i, pk_j)*, the oracle returns the re-encryption key* $rk_{i\to j} \leftarrow \mathsf{ReKeyGen}(pk_i, sk_i, pk_j, sk_j)$*. The adversary is only allowed to make queries where* $i \neq j$*, and either* $i, j \in \mathcal{I}_H$ *or* $i, j \in \mathcal{I}_C$*.*
- *Challenge oracle* $\mathcal{O}_{challenge}$*: This oracle can be queried only once. On input* (M_0, M_1)*, the oracle chooses a bit* $\delta \leftarrow \{0, 1\}$ *and returns the challenge ciphertext* $C^* \leftarrow \mathsf{Enc}(pk^*, M_\delta)$*, where* pk^* *corresponds to the public key of target user* i^**.*

Phase 3:

- *Decision: Eventually,* $\mathcal{A}$ *outputs guess* $\delta' \in \{0, 1\}$*.* $\mathcal{A}$ *wins the game if and only if* $\delta' = \delta$*.*

As already noted in [64], we only allow queries to $\mathcal{O}_{rkgen}$ where users are either both corrupt or both honest. Otherwise, in a bidirectional setting these queries would corrupt honest users and could be used to simulate a decryption oracle, which is not considered in CPA security. That is, we are adopting a static corruption model. Note that we have not included a re-encryption oracle. On the one hand, in the case of CPA security and multihop schemes, the addition of a re-encryption oracle that allows to re-encrypt ciphertexts from honest to corrupt users could be used to simulate a decryption oracle, which is by definition out of the scope of CPA. On the other hand, restricting the re-encryption oracle only to the honest-to-honest and corrupt-to-corrupt cases would result in a superfluous oracle, since the same capabilities are achieved by means of $\mathcal{O}_{rkgen}$. For these reasons, a re-encryption oracle is not provided.

5.1.3 NTRUReEncrypt: A Bidirectional PRE Scheme Based On NTRU

In this section we firstly introduce the conventional NTRU encryption scheme and describe its operation. Then, we extend it in order to construct a bidirectional multihop proxy re-encryption scheme.

The NTRU Encryption Scheme

The NTRU encryption scheme, originally proposed by Hoffstein, Pipher and Silverman in [16], is one of the first public key encryption schemes based on lattices. The main reason behind the interest in NTRU is its efficiency, which is much better than other public-key cryptosystems and is even comparable to symmetric ciphers. In addition, its security is conjectured to be based on hard problems over lattices, although it lacks a formal proof.

We now briefly describe the operation of NTRU. The original NTRU cryptosystem is defined over the quotient ring $\mathcal{R}_{NTRU} = \mathbb{Z}[x]/(x^n - 1)$, where n is a prime parameter. Thus, elements of the ring $\mathcal{R}_{NTRU}$ are integer polynomials of degree less than n; the operators $+$ and $\cdot$ denote addition and multiplication in $\mathcal{R}_{NTRU}$, respectively. Other parameters of NTRU are the integer q, which is a small power of 2 of the same order of magnitude than n, and the small polynomial $p \in \mathcal{R}_{NTRU}$, which usually takes values $p = 3$ or $p = x + 2$. In general, operations over polynomials will be performed in $\mathcal{R}_{NTRU}$ modulo q or p, which will be denoted respectively as $\mathcal{R}_{NTRU}/q$ and $\mathcal{R}_{NTRU}/p$.

In NTRU, the private key sk consists of a polynomial $f \in \mathcal{R}_{NTRU}$ chosen at random, with a determined number of coefficients equal to 0, -1, and 1. The polynomial f must have an inverse in $\mathcal{R}_{NTRU}/q$ and $\mathcal{R}_{NTRU}/p$, respectively, f_q^{-1} and f_p^{-1}. For efficiency, f can be chosen to be congruent to 1 modulo p. The public key pk consists of the polynomial $h = p \cdot g \cdot f_q^{-1} \mod q$, where $g \in \mathcal{R}_{NTRU}$ is chosen at random.

The encryption process is very simple: a plaintext M from message space $\mathcal{R}_{NTRU}/p$ is encrypted to ciphertext $C = h \cdot s + M \mod q$, where s is a small random polynomial in $\mathcal{R}_{NTRU}$. For decrypting a ciphertext C, one must first compute $C' = f \cdot C$ and reduce it modulo q. If the original polynomial C' is sufficiently small then the result of the reduction is $pgs + fM \in \mathcal{R}_{NTRU}/q$. Next, C' is reduced modulo p, obtaining fM. Finally, this result is multiplied by f_p^{-1} to extract the original message M; note that the last step is redundant if f was selected congruent to 1 modulo p.

NTRUReEncrypt

Based on the NTRU encryption scheme, we define the proxy re-encryption scheme NTRUReEncrypt. Our proposal is essentially an extension of the scheme that includes the definition of the re-encryption and re-encryption key generation algorithms. One of the fundamental differences of our scheme with respect to the original NTRU is that the secret polynomial f has to be congruent to $1 \mod p$, not for efficiency reasons, but because it is necessary to correctly decrypt re-encrypted ciphertexts.

The scheme Now we present our scheme NTRUReEncrypt. In the following, the delegator is represented by user A, whereas the delegatee is user B. Our scheme is specified by the following algorithms:

- KeyGen(): The output of the key generation algorithm for user A is a pair of public and secret keys (pk_A, sk_A). If first chooses a pair of polynomials $(f_A, g_A) \in \mathcal{R}_{NTRU}^2$ at random, with a determined number of coefficients equal to 0, -1, and 1, as in the conventional NTRU scheme. The only added requirement is that f_A has to be congruent to 1 modulo p. As in the original case, f_A must have an inverse in $\mathcal{R}_{NTRU}/q$, denoted by f_A^{-1}. Note that now it is not necessary the inverse of f_A modulo p. The private key sk_A is the polynomial f_A, whereas the public key pk_A consists of the polynomial $h_A = p \cdot g_A \cdot f_A^{-1} \mod q$.

- ReKeyGen(sk_A, sk_B): On input the secret keys $sk_A = f_A$ and $sk_B = f_B$, the re-encryption key generation algorithm ReKeyGen computes the re-encryption key between users A and B as $rk_{A\to B} = sk_A \cdot sk_B^{-1} = f_A \cdot f_B^{-1}$. The re-encryption key can be computed by means of a simple three-party protocol, so neither A, B nor the proxy learns any secret key. The protocol, originally proposed in [45], is as follows: A selects a random $r \in \mathcal{R}_{NTRU}/q$ and sends $r \cdot f_A \mod q$ to B and r to the proxy; next, B sends $r \cdot f_A \cdot f_B^{-1} \mod q$ to the proxy, who computes $rk_{A\to B} = f_A \cdot f_B^{-1} \mod q$.
- Enc(pk_A, M): On input the public key pk_A and a message $M \in \mathcal{R}_{NTRU}/p$, the encryption algorithm Enc generates a small random polynomial $s \in \mathcal{R}_{NTRU}$, and outputs the ciphertext $C_A = h_A s + M$.
- ReEnc$(rk_{A\to B}, C_A)$: On input a re-encryption key $rk_{A\to B}$ and a ciphertext C_A, the re-encryption algorithm ReEnc samples a random polynomial $e \in \mathcal{R}_{NTRU}$ and outputs ciphertext $C_B = C_A \cdot rk_{A\to B} + pe$.
- Dec(sk_A, C_A): On input the secret key $sk_A = f_A$ and a ciphertext C_A, the decryption algorithm Dec computes $C'_A = (C_A \cdot f_A) \mod q$ and outputs the original message $M = (C'_A \mod p)$.

Note that if ciphertexts are not re-encrypted, NTRUReEncrypt behaves in exactly the same way as the original NTRU encryption scheme.

Correctness Now, we informally explain the reason why the re-encryption process works. Re-encrypted ciphertexts are of the form:

$$C_B = C_A \cdot rk_{A\to B} + pe = (pg_A f_A^{-1} s + M) \cdot f_A f_B^{-1} + pe = pg_A f_B^{-1} s + pe + f_A f_B^{-1} M$$

When decrypting a re-encrypted ciphertext, the delegatee multiplies the ciphertext with the secret key f_B:

$$C_B \cdot f_B = (pg_A f_B^{-1} s + pe + f_A f_B^{-1} M) \cdot f_B = pg_A s + pef_B + f_A M$$

Now, taking modulo p, we get rid of the additional terms. Recall that we require secret key polynomial f_A to fulfill $f_A = 1 \mod p$, so $(C_B \cdot f_B) \mod p = (pg_A s + pef_B + f_A M) \mod p = M$, which is the original message.

The inclusion of the random term e during the re-encryption phase is necessary in order to prevent a simple ciphertext-only attack from the delegatee. Imagine that a re-encrypted ciphertext is of the form $C_B = C_A \cdot rk_{A\to B} = C_A \cdot f_A \cdot f_B^{-1}$; that is, without the random term e. Then, the delegatee could extract the secret key of the delegator based on the observation that $C_B \cdot f_B = C_A \cdot f_A$ holds, since

Table 5.1: Space costs of NTRUReEncrypt

Element	Size
Keys	$O(n \cdot \log_2 q)$
Message	$O(n)$
Ciphertext	$O(n \cdot \log_2 q)$
Ciphertext expansion	$O(\log_2 q)$

$C_B = C_A \cdot rk_{A \to B}$. Next, assuming that C_A is invertible modulo q, the attacker can extract the secret key by computing $f_A = C_A^{-1} \cdot C_B \cdot f_B$.

The scheme is also multihop, that is, it supports multiple re-encryptions. However, this property is limited, since the addition of the error term during the re-encryption produces an increasing error that grows on each hop, until eventually, decryption fails. Our experiments show that this depends heavily on the choice of parameters. This issue is discussed in more detail in Section 5.1.5.

Analysis The resulting proxy re-encryption scheme preserves the performance level of the original NTRU; a detailed experimentation on this matter is presented in Sections 5.1.5 and 5.1.5. These experimental results show that NTRUReEncrypt outperforms other proxy re-encryption schemes by an order of magnitude.

A theoretical analysis of the computational costs associated with NTRUReEncrypt shows that the main algorithms, namely encryption, decryption and re-encryption, only need a single multiplication (actually, re-encryption takes an additional multiplication for computing the term pe, but this can be computed beforehand and, in addition, the polynomial p is small and known in advance). As already stated in the introduction, the core operation in NTRU is the multiplication of polynomials, which can be done in $O(n \log n)$ time using the Fast Fourier Transform (FFT) [124].

As for the space costs associated to NTRUReEncrypt, Table 5.1 presents a theoretical analysis of the size of the messages, keys, and ciphertexts, as well as a measure of the ciphertext expansion it produces. It can be seen that the expansion is logarithmic in the parameter q, and that the size of keys is within $O(n \log_2 q)$. This can be put in contrast with other lattice-based cryptosystems, such as the proxy re-encryption scheme from Aono et al. [47], where keys and messages are typically matrices whose dimensions grow linearly with the parameter n. This implies that the size of these matrices is at least quadratic with respect to n.

Table 5.2 illustrates this fact by showing a comparison between Aono's scheme

Table 5.2: Comparison of space costs (in KB)

Size	Aono et al. [47]	NTRUReEncrypt
Public keys	60.00	1.57
Secret key	60.00	1.57
Re-Encryption key	2520.00	1.57
Ciphertext	0.66	1.57

and ours. The space costs from Aono's scheme are determined by the parameters shown in [47] for the case of 143 bits of security and 128-bit plaintexts, whereas our figures are calculated using a parameter set that achieves 256 bits of security and supports up to 186-bits plaintexts. More details about the parameters used are given in Section 5.1.5. It can be seen that our space costs are lower in comparison, even considering the difference in bits of security. In the case of re-encryption keys, the difference in size is remarkable as it grows up to 3 orders of magnitude. On the other hand, our ciphertexts are slightly bigger. Another interesting fact is that we use the same kind of polynomials for conveying all the types of keys and ciphertexts, so the space costs are the same in all cases (1.57 KB).

It is easy to check that our scheme is bidirectional. Given $rk_{A\to B} = f_A f_B^{-1}$, the proxy can easily compute $rk_{B\to A} = (rk_{A\to B})^{-1} = f_B f_A^{-1}$. This property is a consequence of how re-encryption keys are constructed ($rk_{A\to B} = sk_A \cdot sk_B^{-1}$), in the same way as in other bidirectional schemes [43, 45, 54].

With respect to the security of the scheme, we cannot overcome the problem of NTRU lacking a formal security proof. However, we can study it from a practical way, as NTRU does. For instance, the time required for a lattice-based attack to the public key is conjectured to be exponential in n [16]. This is also relevant to the re-encryption scheme since extracting private keys from the re-encryption key is at least as hard as attacking NTRU public key, as they are constructed in the same way: for instance, the public key of user B is of the form $pg_B f_B^{-1}$, whereas re-encryption key $rk_{A\to B}$ is of the form $f_A f_B^{-1}$. In NTRU, the polynomials f are larger than the polynomials g, i.e., they have more non-zero terms, so the term f_A of a re-encryption key would be larger than the term pg_B of a public key. However, as it happens to the majority of bidirectional proxy re-encryption schemes [43, 45, 54], our scheme is not collusion-safe. That is, it is vulnerable to a collusion of the proxy and the users, since extracting the delegator's secret key f_A is as simple as computing $rk_{A\to B} \cdot f_B$. The same problem applies to the delegatee's secret key.

As a way to provide a formal security proof of our scheme, we explored how this problem has been addressed in the literature. An interesting approach is made by Stehlé and Steinfeld in [73, 122], where they present a variant of NTRU which is proven CPA-secure under lattice-based assumptions. Based on this solution, we build a second version of our NTRU-based proxy re-encryption scheme, called PS-NTRUReEncrypt, which is provably-secure.

5.1.4 A Provably Secure Version of NTRUReEncrypt

In this section we provide a second proxy re-encryption scheme, called PS-NTRUReEncrypt, that is provable secure under a hard problem for lattices, namely the Ring-LWE assumption. Before explaining our provably-secure scheme, we must first establish some background concepts; after that, we describe the NTRU variant proposed in [73, 122] by Stehlé and Steinfeld, which serves as a basis for our second scheme.

Notation and definitions

Most of these definitions and notation are taken from [73, 122], and were already mentioned in Section 2.3.2, but we gather them here for the sake of convenience. Let $\Phi(x) = x^n + 1$, with n a power of 2; that is, $\Phi(x)$ is the $2n$-th cyclotomic polynomial. Let q be a prime integer such that $q = 1 \mod 2n$. Let $\mathcal{R}$ be the ring $\mathbb{Z}[x]/\Phi(x)$, and $\mathcal{R}_q = \mathcal{R}/q = \mathbb{Z}_q[x]/\Phi(x)$. We will denote the set of invertible elements of $\mathcal{R}_q$ as $\mathcal{R}_q^\times$. Although elements of $\mathcal{R}$ are polynomials, we often treat them as vectors.

The Ring Learning With Errors (Ring-LWE) problem is a hard decisional problem based on lattices introduced by Lyubaskevsky et al. in [37], which we describe in Definition 2.14. In this chapter, we use an adapted version of this problem proposed in [73, 122] by Stehlé and Steinfeld.

Definition 5.3 (The Ring-LWE problem [73, 122]). *Let $s \in \mathcal{R}_q$ and ψ a distribution over $\mathcal{R}_q^\times$, then we define $A_{s,\psi}^\times$ as the distribution that samples pairs of the form (a, b), where a is chosen uniformly from $\mathcal{R}_q^\times$ and $b = a \cdot s + e$, for some e sampled from ψ. The distribution $A_{s,\psi}^\times$ is also called the Ring-LWE distribution. The Ring-LWE problem is to distinguish distribution $A_{s,\psi}^\times$ from a uniform distribution over $\mathcal{R}_q^\times \times \mathcal{R}_q$. The Ring-LWE assumption is that this problem is computationally infeasible.*

This variant of the Ring Learning With Errors problem is denoted in [73, 122] as the R-LWE$^\times_{\text{HNF}}$ problem; however, we will not use that notation for simplicity. The error distributions ψ are sampled from a family of distributions Ψ_α, with parameter α; for more details of the definition of these distributions, see [73, 122].

Provably Secure NTRU

Stehlé and Steinfeld proposed in [73, 122] a variant of the NTRU primitive which is proven CPA-secure under the hardness assumption of the Ring-LWE problem. This scheme is defined over the rings $\mathcal{R}$ and $\mathcal{R}_q$, determined by parameters n and q, as defined in previous section. The plaintext space $\mathcal{M}$ corresponds to the ring $\mathcal{R}/p$, where $p \in \mathcal{R}_q^\times$ is also a parameter of the scheme; as conventional NTRU, typical values for this parameter are $p = 3$ and $p = x + 2$, but in this scheme one may choose $p = 2$, since q is prime. The parameter α characterizes the family of distributions Ψ_α, and the parameter σ is the standard deviation of the Gaussian distribution used during the key generation algorithm. Thus, global parameters are a tuple $(n, q, p, \alpha, \sigma)$. The algorithms of this encryption scheme are the following:

- $\mathsf{KeyGen}()$: The output of this algorithm for user A is the pair of public and secret keys $(pk_A, sk_A) \in \mathcal{R}_q^\times \times \mathcal{R}_q^\times$. Let $D_{\mathbb{Z}^n,\sigma}$ be the Gaussian distribution over $\mathbb{Z}^n$ with standard deviation σ. The keys are computed as follows:
  1. Sample f' from $D_{\mathbb{Z}^n,\sigma}$; let $f_A = 1+p\cdot f'$; if $(f_A \mod q) \notin \mathcal{R}_q^\times$, resample.
  2. Sample g_A from $D_{\mathbb{Z}^n,\sigma}$; if $(g_A \mod q) \notin \mathcal{R}_q^\times$, resample.
  3. Compute $h_A = p \cdot g_A \cdot f_A^{-1}$.
  4. Return secret key $sk_A = f_A$ and $pk_A = h_A$.
- $\mathsf{Enc}(pk_A, M)$: On input the public key pk_A and a message $M \in \mathcal{M}$, sample noise polynomials s, e from a distribution from Ψ_α, and output ciphertext $C_A = h_A s + pe + M \in \mathcal{R}_q$.
- $\mathsf{Dec}(sk_A, C_A)$. On input the secret key $sk_A = f_A$ and a ciphertext C_A, the decryption algorithm Dec computes $C'_A = C_A \cdot f_A$ and outputs the message $M = (C'_A \mod p) \in \mathcal{M}$.

It can be seen that the operation of this scheme is very similar to the original NTRU, except for the generation of the keys and the inclusion of a noise term during encryptions. The authors prove that this NTRU variant is CPA-secure, over the hardness of the Ring-LWE problem. They also prove the correctness of their proposal, by establishing the conditions for avoiding decryption failures. Both proofs will serve as a basis for ours.

PS-NTRUReEncrypt

In this section we present the scheme PS-NTRUReEncrypt, which is based on the provably-secure variant of NTRU described in the previous section. This scheme uses the same global parameters as before, a tuple $(n, q, p, \alpha, \sigma)$. This scheme is defined by the following algorithms:

- KeyGen(): The output of the key generation algorithm for user A is the pair of public and secret keys $(pk_A, sk_A) \in \mathcal{R}_q^\times \times \mathcal{R}_q^\times$. The keys are computed as follows:
    1. Sample f' from $D_{\mathbb{Z}^n,\sigma}$; let $f_A = 1+p\cdot f'$; if $(f_A \mod q) \notin \mathcal{R}_q^\times$, resample.
    2. Sample g_A from $D_{\mathbb{Z}^n,\sigma}$; if $(g_A \mod q) \notin \mathcal{R}_q^\times$, resample.
    3. Compute $h_A = p \cdot g_A \cdot f_A^{-1}$.
    4. Return secret key $sk_A = f_A$ and $pk_A = h_A$.
- ReKeyGen(sk_A, sk_B): On input the secret keys $sk_A = f_A$ and $sk_B = f_B$, the re-encryption key generation algorithm ReKeyGen computes the re-encryption key between users A and B as $rk_{A\to B} = sk_A \cdot sk_B^{-1} = f_A \cdot f_B^{-1}$.
- Enc(pk_A, M): On input the public key pk_A and a message $M \in \mathcal{M}$, the encryption algorithm Enc samples noise polynomials s, e from a distribution from Ψ_α, and outputs ciphertext $C_A = h_A s + pe + M \in \mathcal{R}_q$.
- ReEnc$(rk_{A\to B}, C_A)$: On input a re-encryption key $rk_{A\to B}$ and a ciphertext C_A, the re-encryption algorithm ReEnc samples noise polynomial e' from a distribution from Ψ_α and outputs ciphertext $C_B = C_A \cdot rk_{A\to B} + pe' \in \mathcal{R}_q$.
- Dec(sk_A, C_A). On input the secret key $sk_A = f_A$ and a ciphertext C_A, the decryption algorithm Dec computes $C'_A = C_A \cdot f_A$ and outputs the message $M = (C'_A \mod p) \in \mathcal{M}$.

It can be seen that this scheme is an extension of the one from Stehlé and Steinfeld, by including the definition of the re-encryption and re-encryption key generation algorithms. Note also that the same three-party protocol for computing the re-encryption key described in Section 5.1.3 for NTRUReEncrypt can be applied here.

Analysis The properties of PS-NTRUReEncrypt are the same as our first scheme. These properties are:

Table 5.3: Time costs of PS-NTRUReEncrypt

Operation	On-line	Off-line
Encryption	t_m	$t_m + 2t_s$
Re-Encryption	t_m	$t_m + t_s$
Decryption	t_m	–

- Bidirectional: Given $rk_{A \to B} = f_A f_B^{-1}$, one can easily compute $rk_{B \to A} = (rk_{A \to B})^{-1} = f_B f_A^{-1}$.
- Multihop: It supports multiple re-encryptions, as shown in Section 5.1.4. Note that an error term is included during the re-encryption process, so the noise grows on each hop. Nevertheless, the correctness conditions presented before guarantee that the decryption succeeds with overwhelming probability for proper parameters (which include the devised maximum number of hops, N). See also the results presented on Section 5.1.5.
- Not collusion-safe: This scheme suffers from the same vulnerability as NTRUReEncrypt, so secret keys can be extracted from the re-encryption key if the proxy colludes with a user involved. This problem is common in bidirectional proxy re-encryption schemes.

Table 5.3 shows the computational costs associated to PS-NTRUReEncrypt. As for our first scheme NTRUReEncrypt, the main operation here is the multiplication of polynomials, which is denoted by t_m. In addition, another potentially costly operation is sampling from the noise distribution, denoted by t_s. We distinguish here between on-line and off-line operations, as the latter can be performed beforehand (e.g., sampling noise terms).

With regard to the theoretical space costs, this scheme shares the same results as NTRUReEncrypt, presented in Table 5.1. However, when considering an experimental instantiation, the parameters used would not be the same, which implies that the experimental costs vary. In addition, given that we lack of a formal analysis for quantifying the level of security that is provided by the scheme, it is not possible to make a direct comparison. An experimental analysis of the costs of PS-NTRUReEncrypt is presented in Section 5.1.5.

Security proof In this section, we proceed to prove that the scheme PS-NTRUReEncrypt is CPA-secure. This proof is also based on the one given in [122, Lemma 3.8].

Theorem 5.4. *Suppose that n is a power of 2, and q a prime number such that*

$q = 1 \mod 2n$. Let $\gamma \in (0, \frac{1}{3}), \varepsilon > 0, p \in \mathcal{R}_q^\times$ and $\sigma \geq n\sqrt{ln(8nq)} \cdot q^{\frac{1}{2}+\gamma}$. If there exists and IND-CPA attack against PS-NTRUReEncrypt *with probability $\frac{1}{2} + \varepsilon$, then there exists an algorithm that solves the Ring-LWE problem with probability $\frac{\varepsilon}{2} - q^{-\Omega(n)}$.*

Proof of Theorem 5.4. Let us assume, by contradiction, that we have an adversary $\mathcal{A}$ that breaks PS-NTRUReEncrypt with probability $\frac{1}{2} + \varepsilon$. We construct an algorithm $\mathcal{B}$ against the Ring-LWE problem as follows:

Let $\mathcal{O}_{sample}$ be an oracle that samples tuples (h', C') from either the uniform distribution over $\mathcal{R}_q^\times \times \mathcal{R}_q$ or $A_{s,\psi}^\times$. Algorithm $\mathcal{B}$ first gets a sample (h', C') from $\mathcal{O}_{sample}$. Next, $\mathcal{B}$ generates target public key $pk^* = h^* = p \cdot h'$ and gives pk^* to $\mathcal{A}$. Note that, regardless of the input distribution, h' will be sampled uniformly from $\mathcal{R}_q^\times$, and since $p \in \mathcal{R}_q^\times$, then $h^* = p \cdot h'$ is uniformly random in $\mathcal{R}_q^\times$; however, by [122, Theorem 3], h^* is within negligible statistical distance $q^{-\Omega(n)}$ to public keys generated by KeyGen. $\mathcal{B}$ will also generate an invalid, but correctly distributed, secret key polynomial f^* as described in the first step of the key generation process, so target secret key is $sk^* = f^*$.

Both honest and corrupted key generation are done as in the PS-NTRUReEncrypt scheme. Re-encryption key generation queries are done using the secret keys derived on the key generation queries. Once $\mathcal{A}$ queries the challenge oracle with inputs M_0 and M_1, $\mathcal{B}$ randomly picks $\delta \leftarrow \{0, 1\}$, and constructs the challenge ciphertext as $C^* = p \cdot C' + M_\delta$. Adversary $\mathcal{A}$ receives C^* and, eventually, outputs its guess δ'; if $\delta' = \delta$, then algorithm $\mathcal{B}$ outputs 1.

On the one hand, when the input to $\mathcal{B}$ is a sample from $A_{s,\psi}^\times$, then it will be of the form $(h', C' = h's + e)$, for some $s, e \leftarrow \psi$. In this case, the challenge ciphertext C^* is a correct encryption of M_δ under h^*, since $C^* = p \cdot C' + M_\delta = p(h's + e) + M_\delta = h^*s + pe + M_\delta$, and $\mathcal{A}$ will succeed in distinguishing M_δ with probability $\frac{1}{2} + \varepsilon - q^{-\Omega(n)}$. Hence, $\mathcal{B}$ will determine that the sample was from the $A_{s,\psi}^\times$ distribution with the same probability.

On the other hand, when the sample is from the uniform distribution over $\mathcal{R}_q^\times \times \mathcal{R}_q$ then C' is uniformly random in $\mathcal{R}_q$, and since $p \in \mathcal{R}_q^\times$, so is $p \cdot C'$. The challenge ciphertext $C^* = p \cdot C' + M_\delta$ will be then uniformly random in $\mathcal{R}_q$, as in the original distribution of ciphertexts, and independent of δ. In this case, algorithm $\mathcal{B}$ does not have any advantage in distinguishing the distribution and outputs 1 with probability $\frac{1}{2}$. Overall, algorithm $\mathcal{B}$ succeeds with probability $(\frac{1}{2})(\frac{1}{2} + \varepsilon - q^{-\Omega(n)}) + (\frac{1}{2})(\frac{1}{2}) = \frac{1}{2} + \frac{\varepsilon}{2} - q^{-\Omega(n)}$. This contradicts the Ring-LWE assumption when ε is non-negligible.

□

Note that in this simulation, each re-encryption key that involves the target public key is invalid, as generating a re-encryption key implies knowledge of the secret keys of the users involved. In this case, the target public key is constructed from $\mathcal{O}_{sample}$, so the simulator does not know the corresponding secret key. However, since the secret key of the target used generated at the beginning of the simulation is correctly distributed, any re-encryption key computed from this secret key is indistinguishable from a valid one. A similar argument is made in [13].

Correctness Following the description of the encryption, decryption and re-encryption algorithms, it is easy to informally check that the correctness conditions stablished in Definition 5.1 are fulfilled:

- For all (pk_A, sk_A) output by KeyGen and all messages $M \in \mathcal{M}$, it holds that $\mathsf{Dec}(sk_A, \mathsf{Enc}(pk_A, M)) = M$.

  The evaluation of the encryption of an arbitraty message $M \in \mathcal{M}$ gives $\mathsf{Dec}(psg_A f_A^{-1} + pe + M, sk_A)$ as a result. Next, as described in the decryption algorithm, we take $C' = f_A \cdot (psg_A f_A^{-1} + pe + M) = psg_A + pef_A + Mf_A$. Finally, since $f_A = 1 \mod p$ and $psg_A = pef_A = 0 \mod p$, we have that $C' \mod p = M$.

- For any sequence of pairs (pk_i, sk_i) output by KeyGen, with $0 \le i \le N$, all re-encryption keys $rk_{j \to j+1}$ output by $\mathsf{ReKeyGen}(sk_j, sk_{j+1})$, with $j < N$, all messages $M \in \mathcal{M}$, and all ciphertexts C_1 output by $\mathsf{Enc}(pk_1, M)$, it holds that:

  $$\mathsf{Dec}(sk_N, \mathsf{ReEnc}(rk_{N-1\to N}, ...\mathsf{ReEnc}(rk_{1\to 2}, C_1))) = M$$

  If for any $M \in \mathcal{M}$ correctness holds only with probability 1 minus a negligible quantity, we say that the scheme is correct with respect to $\mathcal{M}$.

  Let us assume the sequence of secret keys $f_0, f_1, ..., f_N$. Since our scheme is multi-hop, a ciphertext re-encrypted N times will be of the form:

  $$\begin{aligned} C_N &= pg_0 f_N^{-1} s + pe_0 f_0 f_N^{-1} + pe_1 f_1 f_N^{-1} + ... + pe_{N-1} f_{N-1} f_N^{-1} + pe_N + Mf_0 f_N^{-1} \\ &= pg_0 f_N^{-1} s + \left[\sum_{i=0}^{N-1} pe_i f_i f_N^{-1} \right] + pe_N + Mf_0 f_N^{-1} \qquad (5.1) \end{aligned}$$

  where e_i denotes the additional error terms added during the re-encryptions, and e_0 is the error term from the first (and unique) encryption. When decrypting C_N, and assuming there are no decryption failures, one obtains:

$$C'_N = C_N \cdot f_N = pg_0 s + \left[\sum_{i=0}^{N} pe_i f_i \right] + M f_0 \tag{5.2}$$

Since, $f_0 = 1 \mod p$ and $pg_0 s = pe_i f_i = 0 \mod p$, for $0 \geq i \geq N$, then we have that $C'_N \mod p = M$.

In the informal proof above, we have not considered the possibility of decryption failures. This is a recurrent issue from the first definition of the NTRU cryptosystem. It can be seen that the re-encryption process produces an increase in the error terms of the ciphertexts, which can potentially lead to decryption failures. We will now describe the correctness conditions of NTRUReEncrypt and prove that the decryption algorithm is capable of handling with this increase. This proof is basically a slight extension of the one given in [122, Lemma 3.7] and generalizes single encryption as well as multiple re-encryptions; in this case, single encryption is a handled as a particular case of multihop re-encryption (taking the number of re-encryptions $N = 0$).

Before continuing, we require two secondary lemmas, both presented in [122]. The first lemma provides bounds for the secret keys generated using the key generation algorithm, whilst the second provides bounds to the product of arbitrary polynomials of $\mathcal{R}$ with samples from a distribution from Ψ_α.

Lemma 5.5. *Let n be a power of 2, q a prime so that $q = 1 \mod 2n$, and $deg(p) \leq 1$. Let $\sigma \geq \sqrt{n \log n} \cdot q^{1/n}$. The secret key polynomials f and g returned by the KeyGen algorithm satisfy $\|f\| \leq 4\sqrt{n}\|p\|\sigma$ and $\|g\| \leq \sqrt{n}\sigma$, with probability $\geq 1 - 2^{-n+3}$.*

Lemma 5.6. *Let $y, r \in \mathcal{R}$, with r fixed and y sampled from a distribution from Ψ_α, with $\alpha q \geq n^{0.25}$. Then: $Pr\left[\|yr\|_\infty \geq \alpha q n^{-0.25} \omega(\log n) \cdot \|r\|\right] \leq n^{-\omega(1)}$*

Theorem 5.7. *If $deg(p) \leq 1$, $\omega(n^{0.25} \log n)\alpha N \|p\|^2 \sigma \leq 1$, and $n^{0.75} \leq \alpha q$, then the decryption algorithm of PS-NTRUReEncrypt succeeds in recovering M with probability $1 - n^{-\omega(1)}$ over the choice of s, e_i, f_i, g_i, for $0 \leq i \leq N$.*

Proof of Theorem 5.7. As stated, a ciphertext C_N re-encrypted N times has the form described in Equation 5.1. When decrypting C_N, the decryption algorithm computes C'_N, as described in Equation 5.2. This computation is implicitly performed modulo q, so $C'_N \in \mathcal{R}_q$. However, let us define $C''_N \in \mathcal{R}$, that is, not modulo q:

$$C''_N = pg_0 s + \left[\sum_{i=0}^{N} pe_i f_i\right] + Mf_0 \in \mathcal{R}$$

In order for the decryption algorithm to succeed $\|C''_N\|_\infty \leq q/2$, so $C'_N = C''_N$ in $\mathcal{R}$. Since $f_0 = 1 \mod p$ and $pg_0 s = pe_i f_i = 0 \mod p$, for $0 \leq i \leq N$, then we have that $C''_N \mod p = C'_N \mod p = M$ and the decryption algorithm succeeds. Then, it is sufficient to give an upper bound on the probability that $\|C''_N\|_\infty \geq q/2$.

From Lemma 5.5, polynomials g_0 and f_i, for $0 \geq i \geq N$, have Euclidean norms $\leq 4\sqrt{n}\|p\|\sigma$, with probability $\geq 1 - 2^{-n+3}$. From the properties of the ring $\mathcal{R}$ [5], we know that for all $u, v \in \mathcal{R}$, $\|u \cdot v\| \leq \sqrt{n} \cdot \|u\| \cdot \|v\|$. As a particular case, since $deg(p) \leq 1$, then $\|p \cdot u\| \leq 2\|p\| \cdot \|u\|$. Hence, it follows that $\|pf_i\|, \|pg_0\| \leq 8\sqrt{n}\|p\|^2\sigma$, with probability $\geq 1 - 2^{-n+3}$. Now, from Lemma 5.6, if $n^{0.25} \leq \alpha q$ we have that with probability $\geq 1 - n^{-\omega(1)}$:

$$\|pf_i e_i\|_\infty, \|pg_0 s\|_\infty \leq 8\alpha q n^{0.25}\omega(\log n)\|p\|^2\sigma$$

Apart from this, since $\|M\| \leq \|p\|$ [122], we know that:

$$\|f_0 M\|_\infty \leq \|f_0 M\| \leq 4n\|p\|^2\sigma$$

Hence $\|C''_N\|_\infty \leq [8(N+2)\alpha q n^{0.25}\omega(\log n) + 4n]\,\|p\|^2\sigma$, and taking $n^{0.75} \leq \alpha q$, then:

$$\|C''_N\|_\infty \leq (8N + 20)\alpha q n^{0.25}\omega(\log n)\|p\|^2\sigma$$

As we assumed initially that $\omega(n^{0.25}\log n)\alpha N\|p\|^2\sigma \leq 1$, then we have that $\|C''_N\|_\infty \leq q/2$, with probability $\geq 1 - n^{-\omega(1)}$. □

It can be seen that for the case of single encryption ($N = 0$), the determined bounds are the same than in [122, Lemma 3.7].

5.1.5 Some Experimental results

In order to validate the viability of the proposed proxy re-encryption schemes, we have developed and tested an implementation of our proposals. Since we propose two schemes, we have two different implementations. NTRUReEncrypt is implemented on top of an available open-source Java implementation of NTRU [78]. For PS-NTRUReEncrypt, due to its non-standard nature, we had to code it from scratch; however, we made use of the functionalities provided by the Java Lattice-Based Cryptography (jLBC) library [126]. Our execution environment consists on an Intel Core 2 Duo @ 2.66 GHz. Although this processor has two cores, our implementation only uses one of them, as it is not parallel.

Performance of NTRUReEncrypt

The first experiment consisted of measuring the performance of the first proposed scheme using different set of parameters. We have used the parameter sets proposed in [79], although other parameters could be studied. Besides the basic implementation, we have made also some optimizations, such as the possibility of using product-form polynomials [79], which has a great effect on performance.

In Table 5.4, we give the execution times for the encryption, decryption and re-encryption for each parameter set. Parameters are represented by a tuple (n, *prod*, k), where *prod* is a boolean parameter that indicates the use of product-form polynomials, and k is the claimed security level (in bits). Additional parameters are fixed, such as $q = 2048$ and $p = 3$.

For each set of parameters, the experiment consisted of a looped execution, where a random message from the plaintext space is sequentially encrypted, re-encrypted and decrypted, and new keys are used on each iteration. The time for each operation is computed as the average of 100 iterations. Since sampling random elements is an operation that can be performed off-line, we decided to exclude it from the time measurements; thus, times reflected in this experiment only reflect the operations that must be performed on-line. This approach was also followed by [124].

For example, the sixth set of parameters was already used for a previous NTRU benchmark [124], and we can see that our results are consistent with their study, where they obtained a measure of approximately 0.31 ms for encryption (3220 encryptions per second).

In addition, Table 5.4 also shows the average number of re-encryptions supported by each parameter set. As stated in Section 5.1.3, the inclusion of the error term pe during the re-encryption process produces an increasing error that grows on each hop, causing a decryption failure at some point. It can be seen that

Table 5.4: Computation time (in ms) and number of hops of NTRUReEncrypt for different parameters

Parameters	Enc.	Dec.	Re-Enc.	# Hops
(439, no, 128)	0.64	0.30	0.24	5
(439, yes, 128)	0.16	0.30	0.23	5
(1087, no, 256)	1.39	1.25	1.05	21
(1087, yes, 256)	0.48	1.26	1.07	15
(1171, no, 256)	0.80	1.12	1.14	21
(1171, yes, 256)	0.43	1.22	1.15	14
(1499, no, 256)	0.74	1.78	1.73	50
(1499, yes, 256)	0.32	1.67	1.66	42

this depends on the choice of parameters, so increasing the parameters actually decreases the probability of decryption failure.

Comparison of NTRUReEncrypt with other proxy re-encryption schemes

In order to benchmark our proposal with respect to other proxy re-encryption schemes, we have implemented three schemes. Two of them share similar properties than ours (i.e., bidirectional and multihop). One of them is the BBS scheme [43], which is CPA-secure as ours, while the other is the one proposed by Weng et al. [54], which is proven CCA-secure; note, however, that the latter scheme achieves a better notion of security than ours, and hence, the comparison is unbalanced in this case. Both of them are implemented in Java using elliptic curve cryptography over a prime field. Specifically, we used the NIST P-256 curve, which provides 128 bits of security [74]. In addition, and for the sake of comparison, we have also implemented a third scheme from Aono et al. [47]; note that in this case, this scheme is not bidirectional. For our scheme, we have used the *ees1171ep1* parameter set from [79], which achieves 256 bits of security. This parameter set corresponds to the sixth row of Table 5.4. We chose this parameter set because it was already used for a previous NTRU benchmark [124]. With regard to the experimental setting, we have used the same environment as before.

Figure 5.1 shows the results of our experiment. It can be seen that our scheme outperforms the others, as it is between 10 and 20 times faster. The scheme from Aono et al. shows a similar performance in encryption and decryption, but is 20 times slower for re-encryption. The results obtained for our scheme are consistent with previous benchmarks for NTRU [124], as we used the same parameters.

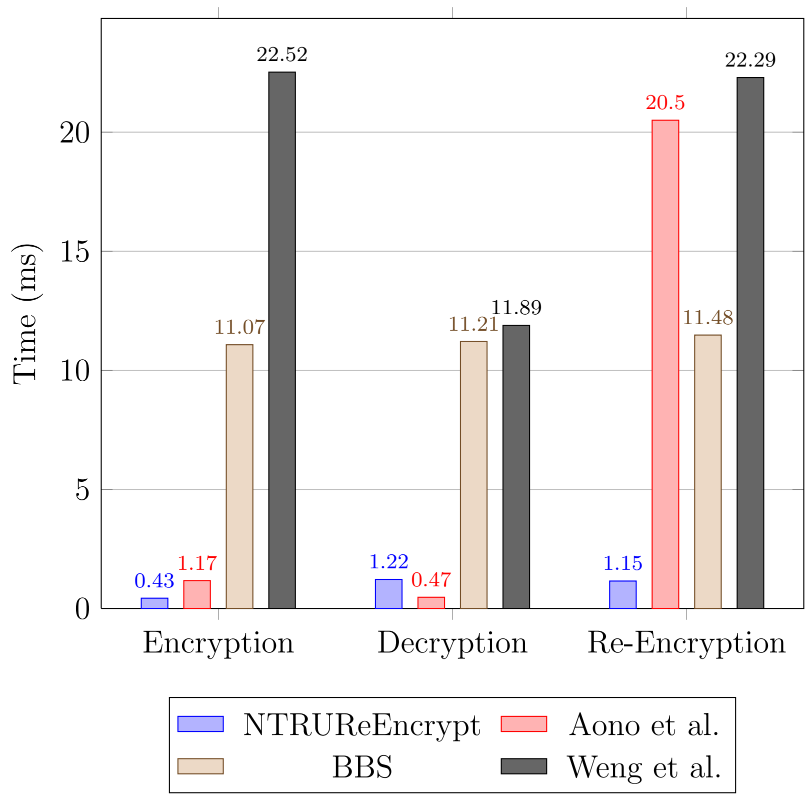


Figure 5.1: Comparison with other proxy re-encryption schemes

Table 5.5: Computation time (in ms) of PS-NTRUReEncrypt for different parameters

Parameters	Enc.	Dec.	Re-Enc.
$n = 32, \log_2 q = 23$	0.93	0.99	1.05
$n = 64, \log_2 q = 28$	4.53	4.23	4.32
$n = 128, \log_2 q = 32$	17.28	17.32	17.45
$n = 256, \log_2 q = 37$	80.64	81.045	86.56
$n = 512, \log_2 q = 41$	333.75	334.07	359.54
$n = 1024, \log_2 q = 46$	1333.03	1344.10	1461.46

For our experiment, we also implemented other unidirectional schemes, specifically, the ones from Ateniese et al. [11] and Libert and Vergnaud [12]. However, we exclude them from this comparison since they are much slower due to the use of bilinear pairings.

Performance of PS-NTRUReEncrypt

Table 5.5 shows the computation time of the main operations of PS-NTRUReEncrypt. It can be seen that the computation time for the main operations is approximately the same for each parameter set. This is also shown in Figure 5.2. Note that this figure is in logarithm scale, so the growth in computation time is exponential as n increases. This is a consequence of requiring that n is a power of 2, which restricts the choice of parameters.

Unlike the first scheme, we do not provide a comparison of PS-NTRUReEncrypt to other proxy re-encryption schemes. We cannot directly compare it to other schemes without properly analyzing the security level it achieves, which is done indirectly through evaluating the security against the best known lattice attacks [122].

With regard to the number of hops of this scheme, the results now are much better than in NTRUReEncrypt, where we had just up to 50 re-encryptions, with the parameter sets used. In this case, we were unable to find a maximum number of hops, having executed more than 10000 re-encryptions for each parameter set, without finding any decryption failure. This lead us to think that the parameters used in PS-NTRUReEncrypt are actually very conservative, which explains also that its performance is lower than NTRUReEncrypt. These parameters are, however, derived from the theoretic assumptions; it is an open problem to find ways to decrease them.

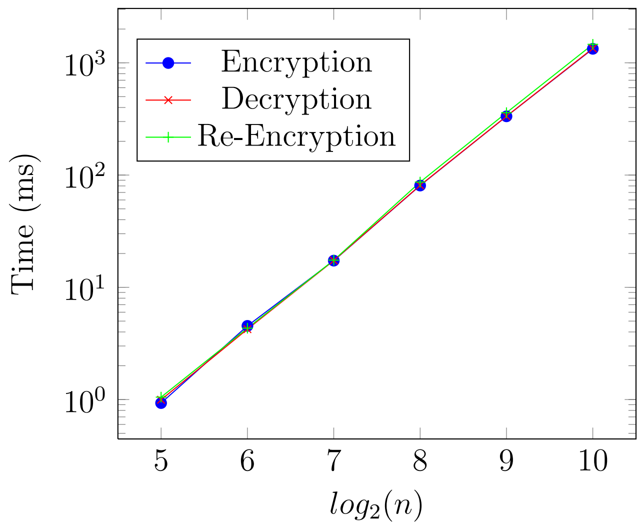


Figure 5.2: Performance of PS-NTRUReEncrypt

Table 5.6: Space costs (in KB) of PS-NTRUReEncrypt for different parameters

Parameters	Size of polynomials
$n = 32, \log_2 q = 23$	0.09
$n = 64, \log_2 q = 28$	0.22
$n = 128, \log_2 q = 32$	0.50
$n = 256, \log_2 q = 37$	1.16
$n = 512, \log_2 q = 41$	2.56
$n = 1024, \log_2 q = 46$	5.75

Other factor that could impact the results is that the implementation is not as optimized as NTRUReEncrypt, since we had to code it from scratch. In order to improve our implementation, we can make use of optimizations such as faster algorithms for polynomial multiplication, based on the Fast Fourier Transform (FFT).

As for the space costs, Table 5.6 shows the size in KB of the polynomials produced by different parameters. Recall that in our schemes, all the elements of the cryptosystem (i.e., public and secret keys, re-encryption keys, and ciphertexts) are represented by the same kind of polynomials.

5.1.6 Towards a Unidirectional Version of NTRUReEncrypt

There are several improvements that are desirable for our proposed schemes. Apart from enhancing the security notion, which is discussed in the next section, achieving a unidirectional (and non-interactive) version strikes as an obvious next step.

The first ideas towards a unidirectional scheme come from the study of a multikey fully homomorphic encryption scheme [127] that uses the NTRU version from Stehlé and Steinfeld as a building block. In this paper, the authors embed a secret key into a ciphertext, using the NTRU variant, for constructing the evaluation key of a fully homomorphic encryption scheme; this way, the secret key is stealthily used during the homomorphic evaluation procedure. It can be seen that this is somehow analogous to the re-encryption keys and re-encryption procedure, so a natural question is to wonder whether this very same approach can be reused for constructing unidirectional re-encryption keys.

Let us assume that the re-encryption key generation procedure of PS-NTRUReEncrypt is modified, so that instead of re-encryption keys of the form $rk_{A\to B} = f_A f_B^{-1}$, these are of the form $rk_{A\to B} = h_B z + pr + f_A$, where z, r are random vectors. A first implication of this change is that the re-encryption key generation procedure becomes non-interactive, since it does not require the delegator to know the secret key f_B of the delegatee, but the public one h_B. A second implication is that, considering that PS-NTRUReEncrypt is also one-way secure, then it should not be possible to extract the secret key from the re-encryption key. The unidirectional property is also obvious, as a particular re-encryption key cannot be used to derive the converse re-encryption key.

Although this approach looks promising, it is important to check if the correctness is preserved. In the case of PS-NTRUReEncrypt, this implies bounding the ciphertext noise in order to study the occurrence of decryption failures. Let $C_A = h_A s + pe + M$ be a regular ciphertext. Recall that when decrypting a re-encrypted ciphertext, the decryption algorithm first makes the following computation:

$$\begin{aligned} C'_B &= f_B \cdot (rk_{A\to B} \cdot C_A) \\ &= f_B \cdot (h_B z + pr + f_A)(h_A s + pe + M) \\ &= p \cdot [(g_B z + f_B r) \cdot C_A + g_A f_B s] + f_A f_B M \end{aligned} \tag{5.3}$$

Assuming no decryption failures, this expression is reduced modulo p, which should yield a correct decryption, since, being $f_A f_B \mod p = 1$, it holds that:

$$C'_B \mod p = f_A f_B M \mod p = M$$

However, these arguments can be false in the event of decryption failures. In fact, the noise represented by the term $p \cdot [(g_B z + f_B r) \cdot C_A + g_A f_B s]$ in Equation 5.3 can be quite large. Preliminary experimental results show that this noise is too big for allowing the decryption to be correct. This problem also seems to appear when we follow a similar approach for NTRUReEncrypt, which was more tolerable against noise. The fact that this approach does not yield immediate results for both NTRUReEncrypt and PS-NTRUReEncrypt hints that solving this problem is far from being obvious. An initial strategy towards solving this problem is to find a set of parameters that minimize the probability of decryption failures, while preserving the security of the schemes. This is still work in progress.

Apart from the decryption failures, the proposed approach, based on the encryption of secret keys acting as re-encryption keys, poses new challenges. The security concerns that arise when a secret key is encapsulated into a ciphertext are modeled by the notion of *circular security*. Informally, this notion studies the implications of providing the adversary with cycles of the following form:

$$\{\mathsf{Enc}(pk_1, sk_2), \mathsf{Enc}(pk_2, sk_3), ..., \mathsf{Enc}(pk_{n-1}, sk_n), \mathsf{Enc}(pk_n, sk_1)\}$$

Cash et al. show in [128] that if it is possible to obtain certain type of key cycles, then passive adversaries might be able to recover the secret keys involved, even when CCA-secure encryption schemes are used. Potentially, this has serious implications for this proposed approach, because re-encryption keys can be used to create key cycles.

5.1.7 Summary

In this section we described NTRUReEncrypt, a highly-efficient proxy re-encryption scheme based on the NTRU cryptosystem. This scheme is bidirectional and multihop, but not collusion-resistant. The key strength of this scheme is its performance. Experimental results show that this scheme outperforms previous proposals by an order of magnitude, and there is room for even more improvement, for instance using parallelization techniques, as shown in [124]. We believe that the level of efficiency shown by NTRUReEncrypt opens up new practical applications of proxy re-encryption in constrained environments. In addition to this scheme, we propose PS-NTRUReEncrypt, a provably-secure variant that is CPA-secure under the hardness of lattice problems, namely the Ring-LWE problem.

5.2 Application of Generic CCA-Secure Transformations to PRE

In the previous section, we described a proxy re-encryption scheme that is proven CPA-secure. Once one obtains schemes like this, an immediate objective would be to improve their security notion, hopefully achieving full CCA-security. To this end, there are two different strategies that can be applied: one is to redesign, from scratch, a new scheme based on the original; a second option is to try to bootstrap the achieved security notion into a stronger one by means of a generic method.

Several generic methods exist for achieving CCA-secure public-key encryption schemes from weakly secure cryptosystems, such as the Fujisaki-Okamoto and REACT transformations [17, 18]. To the best of our knowledge, this is not the case of proxy re-encryption. Therefore, it would be desirable to count on analogous constructions that allow PRE schemes to achieve better security notions. In this section, we study the adaptation of these transformations to proxy re-encryption and find both positive and negative results. On the one hand, we show that a direct and naive application of these transformations leads to flaws in the security proofs, which we spot in a dozen of schemes from the literature. On the other hand, we propose an extension of the Fujisaki-Okamoto transformation for PRE, which achieves a weak form of CCA-security in the random oracle model, and identify the sufficient conditions for applying it.

5.2.1 Introduction

A coveted goal for any cryptosystem is to satisfy a strong notion of security, relevant to its security objectives. In the case of Public-Key Encryption (PKE), indistinguishability against chosen-ciphertext attacks (IND-CCA) is widely regarded as the right notion of security [119]. Informally, the IND-CCA notion describes a security model where any adversary, even with access to a decryption oracle, is not able to distinguish messages for a given ciphertext.

The same desire for achieving right security notions drives the design of other kinds of cryptosystems, and proxy re-encryption is no different. Similarly to the PKE case, the IND-CCA notion is also considered as the target security notion for PRE schemes. In addition to the decryption capabilities, this notion naturally considers the ability of the adversary to re-encrypt chosen ciphertexts by means of a re-encryption oracle. However, this added capability poses an interesting challenge since the possibility of re-encrypting ciphertexts conflicts with the traditional view of CCA-security, which implies non-malleability of ciphertexts, as pointed out by Canetti and Hohenberger, the authors of one of the first CCA-secure PRE scheme [45]. Given the peculiarities of IND-CCA security in PRE, it is often not easy to devise schemes that achieve strong security notions. Most CCA-secure PRE schemes usually resort to additional constructions such as one-time signatures [45, 12], Schnorr signatures [54] and non-interactive zero-knowledge proofs [62], which have to be carefully integrated in an ad-hoc manner. Apart from this difficulty, there are also other factors, such as efficiency and design simplicity, which are usually negatively impacted in CCA-secure schemes.

In the context of PKE, several generic methods exist for achieving CCA-secure schemes from weakly secure cryptosystems. Fujisaki and Okamoto proposed in 1999 [71], and revisited recently in [17], a generic conversion for achieving CCA-security in the random oracle model from a weakly secure asymmetric cryptosystem and a symmetric encryption scheme. Similar transformations, such as REACT [18] and GEM [29], have been proposed since then.

A natural question is, then, to wonder whether an analogous transformation can be constructed for proxy re-encryption schemes. Direct application of the aforementioned transformations leads, in general, to flawed cryptosystems. In this chapter we show several examples from the literature where a naive application of this transformation leads to flaws in the security proofs.

On a positive note, we also describe sufficient conditions that allow to apply the Fujisaki-Okamoto directly. These conditions include achieving a weak notion of security, namely $\mathsf{IND\text{-}CCA}_{0,1}$ (defined in Chapter 4), where the adversary only has access to the re-encryption oracle before the challenge, and the satisfaction of a new property of PRE called "*perfect key-switching*", which characterizes schemes where the re-encryption process preserves the original randomness of ciphertexts. The schemes resulting from this transformation achieve $\mathsf{IND\text{-}CCA}_{2,1}$ security in the random oracle model, a notion slightly weaker than full CCA-security. As an illustration, we present an example of a scheme that satisfies the application conditions and we show the resulting scheme after the transformation. In addition to the Fujisaki-Okamoto proposal, we also outline how the REACT [18] and GEM [29] transformations could be extended for proxy re-encryption.

Finally, we discuss the applicability of the devised transformations to our PRE schemes based on NTRU.

5.2.2 Related work

The Fujisaki-Okamoto transformation, originally proposed in [71] and recently revisited in [17], was the first generic transformation from weakly secure encryption schemes to a CCA-secure public key cryptosystem. Later, Okamoto and Pointcheval described a more efficient construction, called REACT [18], which in turn inspired Coron et al. to propose GEM [29], a more complex construction, but that achieves shorter ciphertexts. Although both REACT and GEM are more efficient than the Fujisaki-Okamoto transformation, they require stronger assumptions on the underlying public-key scheme (see Section 5.2.6).

There are several examples of the application and modification of such transformations for other kinds of cryptosystems. Kitagawa et al. study in [129] the adaptation of both Fujisaki-Okamoto and REACT to Identity-Based Encryption, and estimate their reduction costs. Similar works exist in the context of certificateless public-key encryption [130], and certificated-based encryption [131]. Although these extensions are relevant to our work, the inherent difficulties that arise from the re-encryption capabilities of PRE pose a challenging problem.

To the best of our knowledge, there is no prior work on generic transformations for PRE, in the sense of constructions that increase the security of existing proxy re-encryption schemes. There are, however, a couple of works that propose generic methods for constructing CCA-secure PRE schemes out of other kinds of cryptosystems. Shao et al. propose in [132] a generic method based on a CCA-secure threshold encryption scheme; additionally they describe variants for achieving collusion resistance and Identity-based PRE. In a posterior work, Hanaoka et al. present in [133] another generic method for constructing CCA-secure PRE schemes, also based on the use of threshold encryption, together with a CCA-secure PKE scheme and a strongly unforgeable signature scheme. However, this latter method requires long keys and ciphertexts. For instance, a re-encryption key is made of two PKE public keys, one PKE ciphertext, a threshold encryption share and a signature.

5.2.3 Adapting the Fujisaki-Okamoto Transformation to PRE

In this section we describe the original Fujisaki-Okamoto transformation and describe the conditions that allow to apply it correctly to proxy re-encryption.

The Fujisaki-Okamoto Transformation

Fujisaki and Okamoto proposed in 1999 a generic transformation for achieving IND-CCA security in the random oracle model from a public key encryption scheme with one-way security under chosen plaintext attacks (OW-CPA) [71]. In order to do so, the generic transformation integrates the PKE scheme with an IND-CPA-secure symmetric scheme and a pair of hash functions. Recently, the authors presented a revised version [17], which will be the one we consider in this chapter.

The hybrid transformation, which we denote as Hyb, is as follows. Let PKE be a public-key encryption scheme, Sym a symmetric encryption scheme, and H and G hash functions. PKE is non-deterministic, so in addition to the public key and the message, its encryption function takes an additional parameter for introducing randomness in the ciphertext.

In order to encrypt a message m, the hybrid transformation first samples randomly a term σ from the message space of PKE. The message m is then encrypted with Sym using $G(\sigma)$ as key, which produces the term $c = \mathsf{Sym.Enc}(G(\sigma), m)$. Next, the σ term is encrypted with PKE, taking $H(\sigma, c)$ as the random coins. The hybrid transformation produces the following tuple as the encryption of message m:

$$\mathsf{Hyb.Enc}(pk, m) = (\mathsf{PKE.Enc}(pk, \sigma, H(\sigma, c)), c)$$

When decrypting a ciphertext, which we will denote by the tuple (e, c), the hybrid transformation performs the inverse procedures in reverse order: it decrypts σ from e, and computes $G(\sigma)$ in order to extract the decryption key for c, thus obtaining the original message m. However, an additional validation step is performed during the decryption. This step involves re-computing the term $e = \mathsf{PKE.Enc}(pk, \sigma, H(\sigma, c))$ of the ciphertext, ensuring this way that the ciphertext is valid. If this check does not succeed, the ciphertext is rejected.

This very last step is the reason why the Fujisaki-Okamoto transformation fails if applied as is in PRE: the re-encryption function change the ciphertexts in such a way that the validation checking that takes place during the decryption inevitably fails. In particular, the alteration produced by the re-encryption affects the original random coins introduced in the encryption; hence, the validation check will fail once a ciphertext is re-encrypted. Section 5.2.4 shows how the scheme from Aono et al. [47] is flawed precisely for omitting the validation step, so the problem went unnoticed.

As a solution to this problem, it is interesting to think of PRE schemes where the re-encryption does not alter the original randomness. In this section we characterize a property of PRE schemes that captures this notion, called *perfect key-switching*, and use it as one of the conditions to successfully apply the Fujisaki-Okamoto transformation to PRE.

A New Property of PRE: Perfect Key-Switching

In the previous section we identified that the alteration of the original randomness prevents Fujisaki-Okamoto's decryption procedure from validating a re-encrypted ciphertext. We are therefore interested on those schemes where the original randomness is preserved. This notion is captured by the following property.

Definition 5.8 (Perfect Key-Switching). *Let* $\Pi = (\mathsf{KeyGen}, \mathsf{ReKeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{ReEnc})$ *be a PRE scheme, with message space* $\mathcal{M}$ *and random coins space* $\mathcal{R}$*, and* λ *the security parameter. We say that* Π *satisfies the* perfect key-switching *property if for all keypairs* $(pk_i, sk_i), (pk_j, sk_j)$ *generated by* $\mathsf{KeyGen}(\lambda)$*, all* $m \in \mathcal{M}$*, all* $r \in \mathcal{R}$*, and all* $rk_{i \to j} = \mathsf{ReKeyGen}(pk_i, sk_i, pk_j, sk_j)$*, then*

$$\mathsf{ReEnc}(rk_{i\to j}, \mathsf{Enc}(pk_i, m, r)) = \mathsf{Enc}(pk_j, m, r)$$

It can be seen that the key-switching procedure that takes places during re-encryption is "perfect", in the sense that it does not affect the random coins used. Informally, the re-encryption simply "switches" one public key for another. Examples of this type of scheme are the BBS [43] and CH [45] schemes. It is easy to check that the former exhibits this property. The BBS scheme, based on the ElGamal cryptosystem, is constructed over a group $\mathbb{G}$ of prime order q, with generator g. Secret keys are of the form $sk = a \in \mathbb{Z}_q$, whereas public keys are $pk = g^a \in \mathbb{G}$. Ciphertexts are of the form $(pk^r, g^r \cdot m)$, for a random exponent r. Re-encryption keys are computed as $rk_{i\to j} = \frac{sk_j}{sk_i}$, so the re-encryption process simply consists on raising the pk^r component of the ciphertext to the re-encryption key. This scheme fulfills the perfect key-switching property since the re-encryption process gracefully removes the original public key and substitutes it with the new one, preserving the original randomness:

$$\begin{aligned}
\mathsf{ReEnc}(rk_{i\to j}, \mathsf{Enc}(pk_i, m, r)) &= \mathsf{ReEnc}(\frac{b}{a}, ((g^a)^r, g^r \cdot m)) \\
&= ((g^{ar})^{\frac{b}{a}}, g^r \cdot m) \\
&= (g^{br}, g^r \cdot m) = \mathsf{Enc}(pk_j, m, r)
\end{aligned}$$

Therefore, the original ciphertext $(g^{ar}, g^r \cdot m)$ is perfectly transformed after re-encryption into $(g^{br}, g^r \cdot m)$.

Note that Definition 5.8 can be generalized for PRE schemes with multiple encryption (and consequently, decryption) functions. For example, for the case of two encryption functions Enc_1 and Enc_2, as in the schemes from Ateniese et al. [11], the perfect key-switching condition can be restated as follows:

$$\mathsf{ReEnc}(rk_{i\to j}, \mathsf{Enc}_2(pk_i, m, r)) = \mathsf{Enc}_1(pk_j, m, r)$$

An immediate consequence of the perfect key-switching property is that it implies *proxy invisibility.* A PRE scheme is said to be proxy-invisible if a delegatee is unable to distinguish a ciphertext computed under her public key from a re-encrypted ciphertext, originally encrypted under another public key [11]. That is, the proxy is "invisible", in the sense that the delegatee cannot discern whether the proxy has transformed the ciphertexts. From this description, it is clear that perfect key-switching implies proxy invisibility: a re-encrypted ciphertext has exactly the same form as a ciphertext originally encrypted with the delegatee's public key. However, the converse implication (i.e., proxy invisibility implies perfect key-switching) is not necessarily true. For example, the third proposal from Ateniese et al. [11] is proxy invisible, but does not satisfy the perfect key-switching property, since the original randomness is altered after the re-encryption; in particular, the original random component r is polluted during the re-encryption with the original secret key a, so the new randomness becomes $r' = a \cdot r$. This does not affect security of the scheme in any way because the random elements are safely conveyed as exponents (i.e., exploiting this would imply solving the discrete logarithm problem), but it is sufficient for disallowing the reconstruction of a re-encrypted ciphertext from the original message and randomness, breaking the perfect key-switching property. Another interesting result is that the perfect key-switching property also implies that the PRE scheme cannot be *key-private*, a property of PRE schemes where the proxy cannot learn the identity of the involved users from the re-encryption key. This property was first defined in [64], where the authors formulated some necessary conditions for achieving key privacy. One of this conditions is that the re-encryption function must be probabilistic. However, it can be seen that our formulation of perfect key-switching requires deterministic re-encryption. Therefore, if a PRE scheme satisfies the perfect key-switching property, it cannot be key-private.

Extension of the Fujisaki-Okamoto transformation to PRE

The intuition behind how the Fujisaki-Okamoto transformation can be extended to PRE is simple: assuming the underlying PRE scheme satisfies the perfect key-switching property, then a re-encrypted ciphertext is completely indistinguishable from an original ciphertext created with the same randomness. Therefore, the re-encryption does not affect the transformation.

The extended transformation Let PRE be a proxy re-encryption scheme and Sym a symmetric encryption scheme. Let $\mathcal{M}^{pre}, \mathcal{C}^{pre}$ and $\mathcal{R}^{pre}$ denote the message, ciphertext and randomness spaces of PRE, respectively, while $\mathcal{M}^{sym}, \mathcal{C}^{sym}$ and $\mathcal{K}^{sym}$ denote the message, ciphertext and key spaces of Sym. The encryption function in PRE is non-deterministic, so besides the public key and the message, it takes a parameter from $\mathcal{R}^{pre}$ for introducing randomness in the ciphertext. Let H and G be hash functions, where $H : \mathcal{M}^{pre} \times \mathcal{C}^{sym} \to \mathcal{R}^{pre}$ and $G : \mathcal{M}^{pre} \to \mathcal{K}^{sym}$. We denote as Hyb to the hybrid scheme that results from applying the extended Fujisaki-Okamoto transformation, which is generically defined as follows:

- $\mathsf{Hyb.Setup}(\lambda) \to params'$. On input the security parameter λ, the setup algorithm first computes $params \leftarrow \mathsf{PRE.Setup}(\lambda)$ and outputs the set of global parameters $params' = params \cup \{H(\cdot), G(\cdot)\}$.

- $\mathsf{Hyb.KeyGen}(\lambda) \to (pk_i, sk_i)$. On input the security parameter λ, the key generation algorithm outputs $(pk_i, sk_i) \leftarrow \mathsf{PRE.KeyGen}(\lambda)$, which is the pair of public and secret keys for user i.

- $\mathsf{Hyb.ReKeyGen}(pk_i, sk_i, pk_j, sk_j) \to rk_{i\to j}$. On input the pair of public and secret keys (pk_i, sk_i) for user i and the pair of public and secret keys (pk_j, sk_j) for user j, the re-encryption key generation algorithm outputs the re-encryption key $rk_{i\to j} \leftarrow \mathsf{PRE.ReKeyGen}(pk_i, sk_i, pk_j, sk_j)$.

- $\mathsf{Hyb.Enc}(pk_i, m) \to (e_i, c_i)$. On input a public key pk_i and a message $m \in \mathcal{M}^{sym}$, the encryption algorithm first samples a random $\sigma \in \mathcal{M}^{pre}$, and computes $c_i \leftarrow \mathsf{Sym.Enc}(G(\sigma), m)$. Next, it computes $e_i \leftarrow \mathsf{PRE.Enc}(pk_i, \sigma, H(\sigma, c_i))$. Finally, it outputs ciphertext (e_i, c_i).

- $\mathsf{Hyb.ReEnc}(rk_{i\to j}, (e_i, c_i)) \to (e_j, c_j)$. On input a re-encryption key $rk_{i\to j}$ and a ciphertext (e_i, c_i), the re-encryption algorithm outputs the ciphertext:

$$(\mathsf{PRE.ReEnc}(rk_{i\to j}, e_i), c_i)$$

- $\mathsf{Hyb.Dec}(sk_i, (e_i, c_i)) \to m$. On input the secret key sk_i and a ciphertext (e_i, c_i), the decryption algorithm first computes $\sigma \leftarrow \mathsf{PRE.Dec}(sk_i, e_i)$, and verifies that $\sigma \in \mathcal{M}^{pre}$; otherwise, it outputs $\perp$. Next, it verifies that $e_i = \mathsf{PRE.Enc}(pk_i, \sigma, H(\sigma, c_i))$; otherwise, it outputs $\perp$. Finally, it outputs the result of the symmetric decryption algorithm $\mathsf{Sym.Dec}(G(\sigma), c_i)$.

Note that the resulting hybrid PRE scheme requires the public key during the decryption, since it needs to reconstruct the input ciphertext for validation purposes. Nevertheless, in order to preserve the generic syntax of PRE schemes, one could compute the public key from the secret key (if the underlying PRE scheme allows this possibility) or simply consider the public key as part of the secret key. This extension could also implement countermeasures against reject timing attacks, such as those proposed by Galindo et al. [134]; we will, however, consider them out of the scope of this thesis.

Correctness of the transformation It is necessary to prove that the extended hybrid transformation produces a correct PRE scheme. We assume that the underlying PRE and symmetric schemes are correct. Recall that we additionally require that the PRE scheme satisfies the perfect key-switching property. In PRE, besides the usual PKE condition for correctness, schemes must verify that re-encrypted ciphertexts are correctly decrypted, for any message m:

$$\mathsf{Dec}(sk_j, \mathsf{ReEnc}(rk_{i\to j}, \mathsf{Enc}(pk_i, m)) = m$$

Therefore, for the scheme that results from applying our hybrid transformation, we must prove that the following equation holds:

$$\mathsf{Hyb.Dec}(sk_j, \mathsf{Hyb.ReEnc}(rk_{i\to j}, \mathsf{Hyb.Enc}(pk_i, m)) = m \tag{5.4}$$

First, by definition of the encryption function of the hybrid transformation, $\mathsf{Hyb.Enc}(pk_i, m)) = (e_i, c_i)$, for $c_i = \mathsf{Sym.Enc}(G(\sigma), m)$ and $e_i = \mathsf{PRE.Enc}(pk_i, \sigma, H(\sigma, c_i)))$. Then, the left side of Equation 5.4 can be written as:

$$\mathsf{Hyb.Dec}(sk_j, \mathsf{Hyb.ReEnc}(rk_{i\to j}, (e_i, c_i))$$

Next, by definition of the re-encryption function of the hybrid transformation, we have that:

$$\mathsf{Hyb.ReEnc}(rk_{i\to j}, (e_i, c_i)) = (\mathsf{PRE.ReEnc}(rk_{i\to j}, e_i), c_i)$$

By assumption, the underlying PRE scheme satisfies the property of perfect key-switching. Then:

$$\begin{aligned}\mathsf{PRE.ReEnc}(rk_{i\to j}, e_i) &= \mathsf{PRE.ReEnc}(rk_{i\to j}, \mathsf{PRE.Enc}(pk_i, \sigma, H(\sigma, c_i))) \\ &= \mathsf{PRE.Enc}(pk_j, \sigma, H(\sigma, c_i)) = e_j \end{aligned} \tag{5.5}$$

Therefore, Equation 5.4 is equivalent to:

$$\mathsf{Hyb.Dec}(sk_j, (e_j, c_i)) = m$$

Finally, we must check that the decryption is correct. By definition of the decryption algorithm of the hybrid transformation, we first compute $\sigma' \leftarrow \mathsf{PRE.Dec}(sk_j, e_j)$, and verify that $\sigma' \in \mathcal{M}^{pre}$. By correctness of the PRE scheme, we have that $\sigma' = \sigma$ and $\sigma \in \mathcal{M}^{pre}$, so the verification succeeds. Next, we must verify that $e_j = \mathsf{PRE.Enc}(pk_j, \sigma', H(\sigma', c_i))$, which is true since $\sigma' = \sigma$ and the result of Equation 5.5. The last step outputs the result of the symmetric decryption algorithm $\mathsf{Sym.Dec}(G(\sigma), c_i)$, which is equal to m, since $\sigma' = \sigma$ and the symmetric encryption scheme is correct, by assumption. Therefore, Equation 5.4 holds, and the hybrid transformation is correct.

Security of the transformation Although the transformation seems to be rather straightforward, proving its security is a more elusive matter. The original Fujisaki-Okamoto transformation for PKE achieves IND-CCA2 security, requiring the underlying PKE scheme to be only one-way secure under chosen-plaintext attacks (OW-CPA). This is possible because, in the security proof, the decryption oracle can be constructed without knowledge of the secret key, using only the random oracle tables.

For PRE, it is also necessary to define a re-encryption oracle. A tempting possibility is to do it similarly to the decryption oracle: on input a re-encryption query $(pk_i, pk_j, (e_i, c_i))$, the simulator searches in the random oracle tables for a tuple (σ, c_i, h), such that $e_i = \mathsf{PRE.Enc}(pk_i, \sigma, h)$. If such tuple is found, then the re-encrypted ciphertext is $(\mathsf{PRE.Enc}(pk_j, \sigma, h), c_i)$; otherwise, the oracle cannot respond to the query and returns $\perp$.

This solution seems elegant and simple, but is flawed. If the adversary inputs an ill-formed ciphertext where the randomness does not come from the random oracle H, the oracle rejects the ciphertext, outputting $\perp$. However, in a real execution of the scheme, the re-encryption function is unable to verify whether the input ciphertext was created using H or not, since σ must be kept secret during re-encryption. Therefore, the security proof differs at this point from the real execution. Furthermore, the adversary can potentially make an arbitrary number of such queries, so it is not possible to "patch" the security proof by stilted techniques, such as the artificial aborts from Waters' IBE proof [61]. This very same problem seems to be common and appears in several PRE schemes that make use of constructions inspired in the Fujisaki-Okamoto transformation, rendering their security proofs invalid [55, 62, 54, 135, 136, 137, 138, 139, 140, 141, 142]. Section 5.2.4 analyzes this problem in detail.

Therefore, since the re-encryption function of our extended transformation cannot check whether the hash function H was used or not, it is necessary to construct the re-encryption oracle without the random oracle tables. Furthermore, since the simulator does not have access to the target secret key sk^*, it is not possible in general to compute re-encryption keys of the form $rk_{pk^* \to pk_j}$, for any user j.

In order to bypass these problems, we strengthen the requirements on the underlying PRE scheme: instead of OW-CPA security, we now require $\mathsf{IND\text{-}CCA}_{0,1}$ security. We then construct the security proof of the transformation as a reduction from the security of the underlying scheme, making use of its definition of the re-encryption oracle. This also implies that the transformation only achieves $\mathsf{IND\text{-}CCA}_{2,1}$ security.

The following theorem conveys our main security result. For simplicity in the proofs, we have opted for using the one-time pad rather than a generic symmetric encryption scheme, so the second component of ciphertexts is computed as $c = G(\sigma) \oplus m$, instead of $c = \mathsf{Sym.Enc}(G(\sigma), m)$.

Theorem 5.9 (Security of the transformation)**.** *Let Π be a PRE scheme that is γ-spread [17], fulfills the perfect key-switching property and is $\mathsf{IND\text{-}CCA}_{0,1}$-secure. Let Π' be the resulting scheme after applying the Fujisaki-Okamoto transformation for proxy re-encryption. Then, Π' is $\mathsf{IND\text{-}CCA}_{2,1}$-secure in the random oracle model.*

The proof for Theorem 5.9 is presented below. It basically consists on a reduction from the $\mathsf{IND\text{-}CCA}_{0,1}$ adversary to the $\mathsf{IND\text{-}CCA}_{2,1}$ adversary; since, by assumption, the underlying scheme is $\mathsf{IND\text{-}CCA}_{0,1}$-secure, then the transformed scheme must be $\mathsf{IND\text{-}CCA}_{2,1}$-secure.

Proof of Theorem 5.9. Here we prove that if proxy re-encryption scheme Π is secure in the $\mathsf{IND\text{-}CCA}_{0,1}$ sense, then the scheme Π^{hyb}, which results from applying the Fujisaki-Okamoto transformation to Π as described in Section 5.2.3, is $\mathsf{IND\text{-}CCA}_{2,1}$-secure in the random oracle model. We show how to use an $\mathsf{IND\text{-}CCA}_{2,1}$ adversary that breaks Π^{hyb} to construct an $\mathsf{IND\text{-}CCA}_{0,1}$ adversary that breaks Π.

Let $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$ be a $\mathsf{IND\text{-}CCA}_{2,1}$ adversary attacking Π^{hyb}; that is, it wins the $\mathsf{IND\text{-}CCA}_{2,1}$ game with non-negligible advantage ε^{hyb}. We want to show that it is possible to construct an adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ attacking Π that wins the $\mathsf{IND\text{-}CCA}_{0,1}$ security game with also non-negligible advantage. To this end, we follow a strategy similar to Shoup and Gennaro's proof for the TDH1 cryptosystem [143]: if the $\mathsf{IND\text{-}CCA}_{2,1}$ adversary wins, then he must have queried any of the random oracles with the same input used for creating the challenge; the simulation is maintained up to this point, but after that, it is no longer necessary since we already are able of producing the response for the $\mathsf{IND\text{-}CCA}_{0,1}$ adversary.

```
Algorithm $A_1(pk^*)$
    $(x_0, x_1, s_B) = \mathcal{B}_1(pk^*)$
    Sample random $m_0, m_1 \in \mathcal{M}^{pre}$
    $s_A = (x_0, x_1, s_B)$
    Return $(m_0, m_1, s_A)$
```

```
Algorithm $A_2(m_0, m_1, s_A, e^*)$
    Parse $s_A$ as $(x_0, x_1, s_B)$
    Sample random $\beta, \mu \in \{0, 1\}$
    $c^* = x_\beta \oplus G(m_\beta)$
    Execute $\mathcal{B}_2(x_0, x_1, s_B, (e^*, c^*))$
    If $\mathcal{B}_2$ aborted, then return $\beta$
    Else, return $\mu$
```

Figure 5.3: Adversary $\mathcal{A}$

Adversaries $\mathcal{A}$ and $\mathcal{B}$ have access to the oracles that correspond to their respective attack models. In short, $\mathcal{B}$ has access to all oracles, except to the re-encryption oracle in phase 2 (which corresponds to the $\mathsf{CCA}_{2,1}$ attack model), whereas $\mathcal{A}$ only has access to the key generation oracles and the re-encryption oracle in phase 1 (which corresponds to the $\mathsf{CCA}_{0,1}$ attack model). Note also that $\mathcal{B}$ has access to random oracles H and G. We assume that $\mathcal{B}$ makes at most q_{dec} to the decryption oracle.

Figure 5.3 shows how adversary $\mathcal{A}$ can be constructed using $\mathcal{B}$. Basically, $\mathcal{A}$ simulates the view of the $\mathsf{IND\text{-}CCA}_{2,1}$ game to $\mathcal{B}$, using his own challenge to construct $\mathcal{B}$'s challenge. Since he does not know m_δ, he randomly chooses m_β for the rest of the challenge ciphertext. Therefore, half of the times, $\mathcal{B}$'s view is correct and $\mathcal{A}$ uses it for deciding δ.

Consequently, $\mathcal{A}$ should simulate the view of $\mathcal{B}$, that is, it should answer $\mathcal{B}$'s oracle queries, including the random oracles. The latter are simulated by $\mathcal{A}$ so that their output is generated on-demand. This corresponds to the typical intuition of the random oracle: a function that returns a randomly assigned output for each possible input, sampled uniformly from its output domain. $\mathcal{A}$ maintains a list of tuples for each random oracle, where each tuple contains the input and output of a query: $\mathcal{L}_H$ is the list for oracle H and $\mathcal{L}_G$ the list for oracle G. Key generation oracle queries (i.e., $\mathcal{O}_{corrupt}, \mathcal{O}_{honest}$ and $\mathcal{O}_{rkgen}$) are trivially answered by relaying them to $\mathcal{A}$'s oracles.

All that remains is to tackle with decryption and re-encryption queries. Figure 5.4 shows the algorithms that simulate $\mathcal{B}$'s view of the decryption and re-encryption oracles. Recall that, according to the $\mathsf{CCA}_{2,1}$ attack model, the decryption oracle is available in both phases, whilst the re-encryption oracle is only available in phase 1.

This definition of $\mathcal{B}$' decryption oracle is essentially the same than the one found in Fujisaki and Okamoto's proof [17] and does not need any secret keys. The

```
Algorithm $\mathcal{O}^B_{dec}(pk_i, (e, c))$
    Search $(\sigma, c, h) \in \mathcal{L}_H$,
            such that $e = \mathsf{PRE.Enc}(pk_i, \sigma, h)$
    If such tuple does not exist, then return $\perp$
    Return $c \oplus G(\sigma)$
```

```
Algorithm $\mathcal{O}^B_{reenc}(pk_i, pk_j, (e, c))$
    $e' = \mathcal{O}^A_{reenc}(pk_i, pk_j, e)$
    If $e' = \perp$, then return $\perp$
    Else, return $(e', c)$
```

Figure 5.4: Decryption and re-encryption oracles for $\mathcal{B}$

perfect key-switching property of the underlying PRE scheme guarantees that the simulation of this oracle is correct, even when the input ciphertext is a re-encryption. Note also that it is possible that $\mathcal{B}$ submits a valid ciphertext without having used the random oracles, so it gets wrongfully rejected. This event is called Bad in Fujisaki and Okamoto's proof; assuming that the underlying PRE scheme is γ-spread, the probability of this event is bound by $q_{dec} \cdot 2^{-\gamma}$.

Similarly, $\mathcal{B}$' re-encryption oracle does not require re-encryption keys, but relies on $\mathcal{A}$'s re-encryption oracle, which is provided in the $\mathsf{IND\text{-}CCA}_{0,1}$ security game. The simulation of this oracle is perfect. Key generation oracles are also perfectly simulated, since they depend exclusively on $\mathcal{A}$'s oracles.

The simulation of the random oracles is perfect, until $\mathcal{B}$ queries $G(m_\beta)$ or $H(m_\beta, \cdot)$. If this event occurs, $\mathcal{A}$ aborts the simulation and outputs β, following the strategy mentioned at the beginning. Otherwise, the simulation continues until $\mathcal{B}$ halts, and $\mathcal{A}$ outputs a random bit μ.

Assuming that Bad does not occur, it can be seen that when $\beta = \delta$ in $\mathcal{A}_2$, which happens with probability $\frac{1}{2}$, the challenge ciphertext is identically distributed as the one defined by the security game; in this case, $\mathcal{B}$ should win the game with probability $\frac{1}{2} + \varepsilon^{hyb}$, and therefore, should query $G(m_\beta)$ or $H(m_\beta, \cdot)$ with the same probability. Therefore, $\mathcal{A}$ wins the game with probability $\frac{1}{2} + \varepsilon^{hyb}$ too. On the contrary, when $\beta \neq \delta$, then $\mathcal{B}$ does not have any advantage for distinguishing the challenge, since x_β is information-theoretically hidden, so $\mathcal{A}$ wins the game with probability $\frac{1}{2}$. Therefore, the success probability of $\mathcal{A}$ under this assumption is:

$$Pr[\delta = \delta' | \neg \mathsf{Bad}] = \frac{1}{2} \cdot (\frac{1}{2} + \varepsilon^{hyb}) + \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{2} + \frac{\varepsilon^{hyb}}{2}$$

Finally, taking into consideration the probability that Bad occurs, the overall success probability and advantage of adversary $\mathcal{A}$ are, respectively:

$$\begin{aligned} Pr[\delta = \delta'] &\geq Pr[\delta = \delta' | \neg \mathsf{Bad}] Pr[\neg \mathsf{Bad}] \\ &\geq (\frac{1}{2} + \frac{\varepsilon^{hyb}}{2}) \cdot (1 - q_{dec} \cdot 2^{-\gamma}) \end{aligned}$$

$$\varepsilon^{pre} \geq \frac{\varepsilon^{hyb}}{2} - \frac{(1 + \varepsilon^{hyb}) \cdot q_{dec}}{2^{\gamma+1}}$$

□

An interesting question is why not aim for $\mathsf{IND\text{-}CCA}_{2,2}$ security. The answer is that the extended transformation, as is, is vulnerable to certain type of attacks that use the re-encryption oracle after the challenge. Since the re-encryption function simply re-encrypts the asymmetric part, it cannot verify whether the symmetric part is valid or not. For example, suppose that the adversary takes the challenge ciphertext (e^*, c^*), produces $(e^*, c^* \oplus K)$ for some value $K \in \mathcal{C}^{sym}$, and asks for the re-encryption of the resulting ciphertext from the target user to a corrupt one. This query is legal because $(e^*, c^* \oplus K)$ is, technically, not a derivative of the challenge ciphertext. Now the adversary only has to decrypt the response, once he removes the mask K.

As we have discussed before, the difficulty of constructing the re-encryption oracle in the security proof constrains both the security requirements and expectations of this transformation. An interesting possibility is to modify the transformation so as to be able to construct the security proof without these constraints. To this end, an option is to use some kind of non-interactive zero-knowledge (NIZK) proof in order to extract the randomness used during encryption, as well as to sign the rest of the ciphertext. However, because of the rewinding problems that appear in the security proof, which make it run in exponential time, this is not possible in general. To solve this, Canard et al. propose in [144] the use of a NIZK proof with online extractor (NIZKOE) as a way to patch the flawed scheme from Chow et al. [135]. In this chapter, however, we focus only on what it can be achieved without extending the transformation with new primitives.

5.2.4 An Account of Flawed Schemes

A Frequent Error in CCA-security Proofs in the Random Oracle model

In Section 5.2.3, we discussed the problems that arise in the security proof when generic transformations are directly applied in PRE, in particular when the re-encryption oracle is based on the random oracle tables. This solution is incorrect, as the security proof differs from the real execution for certain re-encryption queries.

We surveyed the literature of CCA-secure PRE schemes in the Random Oracle model, looking for instances of this flaw. Our study yielded eleven vulnerable schemes [55, 62, 54, 135, 136, 137, 138, 139, 140, 141, 142], althought this list might not be exhaustive.

Although these PRE schemes have different characteristics, such as being bidirectional [54], conditional [55] or identity-based [141], all of them share the same intrinsic problem that arises from a wrong integration of the Fujisaki-Okamoto transformation or other related techniques. It is important to note that this problem was first identified by Canard et al. in [144], but restricted to the analysis of scheme from Chow et al. [135].

The problem is based on two different issues. First, in the encryption function, secret values are used as input to a hash function to produce the randomness needed for encryption. For instance, in the original Fujisaki-Okamoto transformation [71], the session key σ and the original message m are used as this input: it is clear that m should not be disclosed during re-encryption; in the same manner, neither σ should be revealed, since this could be used to decrypt the message. However, this also means that in a real execution of the re-encryption function, it is not possible to verify that the hash function has been used correctly (i.e., (σ, m) was used as input to H).

Second, the CCA attack model in PRE implies that queries of the form $(pk^*, pk_j, \hat{c})$, where user j is corrupt and $\hat{c}$ is not a derivative of the challenge, are legitimate and must be answered correctly. This requirement can be inferred from usual security models for CCA in proxy re-encryption, such as [45, 120], and in fact, represents a critical, yet often overlooked, point in most CCA security proofs. In particular, the simulation of these queries in the security proofs of the identified schemes relies on examining the random oracle tables, which contain the input and output of previous random oracle queries, as generally the challenger does not have access to the target secret key sk^*, which is necessary for generating the corresponding re-encryption key.

Taking both issues into consideration, let us suppose that the adversary creates an ill-formed ciphertext $\hat{c}$ under the target public key pk^*, where the randomness r is not derived from $H(\sigma, m)$, but chosen randomly. Next, the adversary calls the re-encryption oracle with the input $(pk^*, pk_j, \hat{c})$, for some corrupt user j. Note that this query is legal, since it is not related to the challenge ciphertext (in fact, it can occur in phase 1), and the simulator must answer it correctly since the adversary can check whether its decryption results in the same original message. The re-encryption strategy based on the random oracle tables does not work in this case, since the adversary did not use the random oracle, so the challenger cannot respond to this type of queries. The security proofs in the identified schemes opt for returning $\perp$, indicating that the ciphertext is invalid. However, this approach differs from the real execution, where the re-encryption function is unable to check whether the random oracle was used or not (i.e., that the ciphertext is valid or not), since the input to the random oracle should be hidden from the proxy; the re-encryption will work normally. Consequently, these security proofs are flawed, as the security proofs differ from the real execution.

Let us illustrate this flaw by analyzing one of the identified schemes. The PRE scheme from Weng et al. [54, 50] is bidirectional, single-hop, interactive and is allegedly CCA-secure in the random oracle model. Although this scheme does not use the Fujisaki-Okamoto transformation, it integrates a variation of the Hashed ElGamal scheme. Still, the same error in the security proof applies, since the randomness is generated from a value that must be kept hidden from the proxy (i.e., the original message m), which implies it can only be correctly verified during decryption, and not during re-encryption. The following is a slightly simplified version of the scheme:

- $\mathsf{Setup}(\lambda)$: The setup algorithm first determines the cyclic group $\mathbb{G}$ of order q, with q a prime of λ bits. A generator $g \in \mathbb{G}$ is chosen randomly. The message space is $\mathcal{M} = \{0,1\}^n$, where n is polynomial in the security parameter λ. It is also required a set of hash functions H_1, H_2 and H_3, where $H_1 : \{0,1\}^n \times \mathbb{G} \to \mathbb{Z}_q, H_2 : \mathbb{G} \to \{0,1\}^n$ and $H_3 : \mathbb{G}^2 \times \{0,1\}^n \to \mathbb{Z}_q$. The global parameters are represented by the tuple:

$$params = (q, \mathbb{G}, g, H_1, H_2, H_3)$$

- KeyGen(params): Sample a random $x_i \in \mathbb{Z}_q$, and compute the public and private key of user i as $pk_i = g^{x_i}$ and $sk_i = x_i$.
- $\mathsf{ReKeyGen}(sk_i, sk_j)$: The re-encryption key from user i to user j is computed as $rk_{i\to j} = sk_j/sk_i$.
- $\mathsf{Enc}(pk_i, m)$: First, compute $r = H_1(m, pk_i)$. Next, sample random $u \in \mathbb{Z}_q$, and compute $D = (pk_i)^u$, $E = (pk_i)^r$, $F = H_2(g^r) \oplus m$, and $s = u + r \cdot H_3(D, E, F) \mod q$. The ciphertext is the tuple $CT_i = (D, E, F, s)$.
- $\mathsf{ReEnc}(rk_{i\to j}, CT_i = (D, E, F, s))$: First, check that the condition $(pk_i)^s = D \cdot E^{H_3(D,E,F)}$ holds; otherwise, output $\perp$. The re-encryption of ciphertext CT_i is the tuple $CT_j = (pk_i, E', F)$, where $E' = E^{rk_{i\to j}} = (pk_j)^r$.

- $\mathsf{Dec}_1(sk, CT = (pk_i, E', F))$: Given a first-level ciphertext CT, the original message is computed as $m = F \oplus H_2((E')^{1/sk})$. Finally, check that $E' = pk^{H_1(m,pk_i)}$ holds and return m; otherwise, return $\perp$.

- $\mathsf{Dec}_2(sk, CT = (D, E, F, s))$: Given a second-level ciphertext CT, first check that the condition $pk^s = D \cdot E^{H_3(D,E,F)}$ holds; otherwise, output $\perp$. The original message is computed as $m = F \oplus H_2(E^{1/sk})$. Finally, check that $E = pk^{H_1(m,pk)}$ holds and return m; otherwise, return $\perp$.

Note that the check that is made during re-encryption is unable to verify whether $r = H_1(m, pk_i)$. For instance, if the value r is chosen randomly, the re-encryption still works normally. Therefore, the security proof should also behave in the same way, but we will see below that this is not the case. Figure 5.5 shows the definition of the re-encryption oracle given in the security proof of this scheme.

```
Algorithm $\mathcal{O}_{reenc}(pk_i, pk_j, CT_i = (D, E, F, s))$
  If $(pk_i)^s \neq D \cdot E^{H_3(D,E,F)}$ return $\perp$
  If $i$ and $j$ are both honest or corrupt
    Compute $E' = E^{x_j/x_i}$
  Else
    Search tuple $(m, pk_i, r) \in \mathcal{L}_{H_1}$, such that $(pk_i)^r = E$
    If such tuple exists, then compute $E' = (pk_j)^r$
    Else return $\perp$
  Return $CT_j = (pk_i, E', F)$
```

Figure 5.5: Re-encryption oracle from Weng et al.'s security proof [54]

It can be seen that when users i and j are both honest or corrupt, the re-encryption oracle simply computes the re-encryption keys; otherwise, it resorts to examining $\mathcal{L}_{H_1}$, the random oracle table for H_1.

However, as pointed out previously, this oracle is unable to correctly answer queries of the form $(pk^*, pk_j, \hat{c})$, where $\hat{c}$ is an ill-formed ciphertext under the target public key pk^* whose randomness r is not derived from $H_1(m, pk^*)$. In this case, the target user is honest, by definition, whereas user j is corrupt, so the re-encryption oracle resorts to the random oracle table $\mathcal{L}_{H_1}$, which does not contain any tuple of the form (m, pk^*, r), and returns $\perp$. This behavior deviates from the real-world execution of the re-encryption function, where no ciphertext is rejected when $r \neq H_1(m, pk^*)$.

Another example of wrong usage of the transformation

In addition to the flawed schemes presented in the previous section, we describe here a scheme where the Fujisaki-Okamoto transformation is incorrectly applied. In [47], Aono et al. proposed a lattice-based encryption scheme which is proven CPA-secure in the standard model. Then, on top of this scheme, they construct a "CCA-secure" version in the random oracle model, using the generic conversion from Fujisaki and Okamoto [71]. However, this version is flawed because it does not perform the validation step during decryption. If one forces this check, decryption will always fail for re-encrypted ciphertexts, breaking the correctness of the scheme.

The PRE scheme from Aono et al. In this subsection we describe the scheme from Aono et al. This scheme is based upon a lattice cryptosystem from Lindner and Peikert [33], whose hardness relies on the Learning With Errors (LWE) problem [32]. Some details are omitted for clarity; we refer the interested reader to [47] and [33] for a complete description of the scheme.

Let n be the security parameter, q a prime number, and l a fixed message length. Set $k = \log q$. Let the randomly generated matrix $A \in \mathbb{Z}_q^{n \times l}$ be publicly known. Let $D^{n_1 \times n_2}$ be a gaussian noise distribution over $\mathbb{Z}^{n_1 \times n_2}$. Auxiliary function $Power2 : \mathbb{Z}_q^{n \times l} \to \mathbb{Z}_q^{nk \times l}$ is defined as

$$Power2(X) = \begin{bmatrix} X \\ 2X \\ \vdots \\ 2^{k-1}X \end{bmatrix}$$

while function $Bits : \mathbb{Z}_q^{1 \times n} \to \{0,1\}^{1 \times nk}$ produces a bit representation of the elements of input vector v, such that $Bits(v) \cdot Power2(X) = v \cdot X \in \mathbb{Z}_q^{1 \times l}$. Details about noise distribution D and functions $Power2(\cdot)$ and $Bits(\cdot)$ are omitted here, but can be found on [47, 33].

The scheme is as follows:

- $\mathsf{KeyGen}(n)$: Sample $R, S \leftarrow D^{n \times l}$. The public key is $pk = P = R - AS \in \mathbb{Z}_q^{n \times l}$, while the secret key is $sk = S \in \mathbb{Z}^{n \times l}$.
- $\mathsf{Enc}(pk = P, m \in \{0,1\}^l)$: Sample gaussian noise vectors f_1, f_2 from $D^{1 \times n}$, and f_3 from $D^{1 \times l}$. Encode the message $m \in \{0,1\}^l$ to $m \cdot \lfloor \frac{q}{2} \rfloor \in \mathbb{Z}_q^{1 \times l}$.

Output the ciphertext $e = (e_1, e_2) \in \mathbb{Z}_q^{1\times(n+l)}$, where

$$\begin{aligned} e_1 &= f_1 A + f_2 \\ e_2 &= f_1 P + f_3 + m \cdot \lfloor \frac{q}{2} \rfloor \end{aligned}$$

- $\mathsf{Dec}(sk = S, e = (e_1, e_2))$: Compute $\bar{m} = e_1 S + e_2 \in \mathbb{Z}_q^{1\times l}$. Let $(\bar{m}_1, ..., \bar{m}_l)$ be the individual elements of $\bar{m}$. Output the decrypted message $m = (m_1, ..., m_l)$, where $m_i = 0$ if $\bar{m}_i \in [-\lfloor \frac{q}{4} \rfloor, \lfloor \frac{q}{4} \rfloor) \subset \mathbb{Z}_q$, and $m_i = 1$ otherwise.

- $\mathsf{ReKeyGen}(sk_i = S_i, pk_j = P_j, sk_j = S_j)$: Sample $X \in \mathbb{Z}_q^{nk\times n}$ from the uniform distribution and E from $D^{nk\times l}$. Output the re-encryption key $rk_{i\to j} = (P_j, Q)$, where

$$Q = \begin{bmatrix} X & -XS_j + E + Power2(S_i) \\ 0 & I \end{bmatrix}$$

- $\mathsf{ReEnc}(rk_{i\to j} = (P_j, Q), e = (e_1, e_2))$: Sample gaussian noise vectors g_1, g_2 from $D^{1\times n}$, and g_3 from $D^{1\times l}$. Compute

$$g_1[A|P_j] + [g_2|g_3] + [Bits(e_1)|e_2] \cdot Q \in \mathbb{Z}_q^{1\times(n+l)}$$

and parse the result as $[e'_1|e'_2]$, where $e'_1 \in \mathbb{Z}_q^{1\times n}$ and $e'_2 \in \mathbb{Z}_q^{1\times l}$. Output the re-encrypted ciphertext $e' = (e'_1, e'_2)$. It can be seen that

$$\begin{aligned} e'_1 &= g_1 A + g_2 + Bits(e_1)X \\ e'_2 &= g_1 P_j + g_3 + Bits(e_1)(E - XS_j) + e_1 S_i + e_2 \end{aligned}$$

Description of the flaw The flaw is based on the fact that a direct application of the Fujisaki-Okamoto conversion does not work in general for proxy re-encryption, since the validity check during the decryption fails for re-encrypted ciphertexts. In the definition of their CCA scheme, they only consider the validity of decryption in the case of original ciphertexts. However, when one considers re-encrypted ciphertexts, it can be seen that the validity check during the decryption *always* fails.

Let us consider the CCA-secure version of the scheme, after applying directly the Fujisaki-Okamoto transformation, and let us define a scenario with two users i and j, with public keys P_i and P_j, respectively. A ciphertext originally encrypted for user i is a tuple (e_1, e_2, c) where:

$$\begin{aligned} e_1 &= f_1 A + f_2 \\ e_2 &= f_1 P_i + f_3 + \sigma \cdot \lfloor \frac{q}{2} \rfloor \\ c &= \mathsf{Sym.Enc}(G(\sigma), m) \end{aligned}$$

The terms (e_1, e_2) are the encryption of σ with the PRE scheme, where the term c is the symmetric encryption of the original message, with key $G(\sigma)$. Thus, the original random coins in this case are the noise vectors (f_1, f_2, f_3), which by the definition of the transformation, are computed from $H(\sigma, c)$. In the random oracle model, if this query to H has not been done previously, H produces a random output and records the tuple $(\sigma, c, f_1, f_2, f_3)$ for future queries.

Now lets consider that this ciphertext, originally intended for user i, is re-encrypted for some user j. Let g_1, g_2, g_3 be the random vectors introduced by the re-encryption function, X, E random matrices defined by user j during the interactive process for generating re-encryption keys and S_i, S_j the secret keys of users i and j. The transformation only re-encrypts the terms (e_1, e_2), so the result is a tuple (e'_1, e'_2, c) of the form:

$$\begin{aligned} e'_1 &= g_1 A + g_2 + Bits(e_1)X \\ e'_2 &= g_1 P_j + g_3 + Bits(e_1)(E - XS_j) + e_1 S_i + e_2 \\ c &= \mathsf{Sym.Enc}(G(\sigma), m) \end{aligned}$$

Now user j decrypts this ciphertext. Recall that the Fujisaki-Okamoto conversion dictates that the output of the asymmetric encryption must be reconstructed, using the same inputs. By the correctness of the original PRE scheme, she correctly receives σ. However, the output of $H(\sigma, c)$ must be again (f_1, f_2, f_3), by the definition of the random oracle. When trying to reconstruct (e'_1, e'_2) she would obtain $\hat{e}'_1 = f_1 A + f_2$, $\hat{e}'_2 = f_1 P_j + f_3 + \sigma \cdot \lfloor \frac{q}{2} \rfloor$, and the validation would fail since $(\hat{e}'_1, \hat{e}'_2) \neq (e'_1, e'_2)$. Therefore, the decryption of re-encrypted ciphertexts will always fail.

5.2.5 Applying the proposed transformation

For illustration purposes, in this section we present a PRE scheme that is secure in the sense of $\mathsf{IND\text{-}CCA}_{0,1}$, assuming the 3-wDBDHI problem [12] is hard. We later apply our extended Fujisaki-Okamoto transformation for obtaining a $\mathsf{IND\text{-}CCA}_{2,1}$-secure scheme.

The original scheme

This scheme is based on a scheme from Ateniese et al. [11] that lacked a security proof. In order to prove that this scheme is $\mathsf{IND\text{-}CCA}_{0,1}$ secure, we borrowed some ideas from the RCCA secure scheme from Libert and Vergnaud [12]. The scheme is as follows:

- $\mathsf{Setup}(1^\lambda)$: The setup algorithm first determines the cyclic groups $\mathbb{G}$ and $\mathbb{G}_T$ of order q, with q a prime of λ bits, and a bilinear pairing e, so that $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. A set of generators $g, u, v \in \mathbb{G}$ is chosen randomly, and $Z = e(g, g)$. Function $F : \mathbb{Z}_q \to \mathbb{G}$ is defined as $F(t) = u^t \cdot v$. The global parameters are represented by the tuple:

$$params = (g, u, v, F(\cdot), Z)$$

- $\mathsf{KeyGen}(params)$: Sample a random $x_i \in \mathbb{Z}_q$, and compute the public and private key of user i as $pk_i = g^{x_i}$ and $sk_i = x_i$.
- $\mathsf{ReKeyGen}(sk_i, pk_j)$: The re-encryption key from user i to user j is computed as
$$rk_{i\to j} = (pk_j)^{1/sk_i} = g^{sk_j/sk_i}$$
- $\mathsf{Enc}_2(pk_i, m)$: Sample random $r, t \in \mathbb{Z}_q$. The second-level encryption of m under pk_i is the tuple $CT_i = (t, C_0, C_1, C_2)$, where
$$C_0 = F(t)^r \qquad C_1 = Z^r \cdot m$$
$$C_2 = (pk_i)^r = g^{x_i r}$$
- $\mathsf{Enc}_1(pk_i, m)$: The first-level encryption of m under pk_i is exactly as the second-level, except that $C_2 = e(g, pk_i)^r = Z^{x_i r}$.
- $\mathsf{ReEnc}(rk_{i\to j}, CT_i = (t, C_0, C_1, C_2))$: Check that the condition $e(C_0, pk_i) = e(C_2, F(t))$ holds; otherwise, output $\perp$. The re-encryption of the second-level ciphertext CT_i is the tuple $CT_j = (t, C_0, C_1, C_2')$, where $C_2' = e(C_2, rk_{i\to j}) = Z^{x_j r}$.
- $\mathsf{Dec}_1(sk_i, CT_i = (t, C_0, C_1, C_2))$: Given a first-level ciphertext CT_i, the original message is computed as
$$m = \frac{C_1}{C_2^{1/sk_i}}$$

- $\mathsf{Dec}_2(sk_i, CT_i = (t, C_0, C_1, C_2))$: Given a second-level ciphertext CT_i, the original message is computed as

$$m = \frac{C_1}{e(g, C_2)^{1/sk_i}}$$

The first important characteristic of this scheme is that it fulfills the perfect key-switching property, since the original randomness used in second-level ciphertext is preserved after re-encryption, so a re-encrypted ciphertext is equal to first-level ciphertext encrypted with the same randomness. More formally, for all messages m, randomness t, r and public keys pk_i, pk_j, it holds that:

$$\mathsf{ReEnc}(rk_{i\to j}, \mathsf{Enc}_2(pk_i, m; t||r)) = \mathsf{Enc}_1(pk_j, m; t||r)$$

Assuming we use the same randomness $t||r$ for both encryptions, then it can be seen that a re-encrypted ciphertext is computed with the same operations than a first-level encryption, except for element C_2. The C_2 component of a first-level encryption under pk_j is computed as $C_2 = e(g, pk_j)^r = Z^{x_j r}$, while the same component in a re-encrypted ciphertext is $C_2' = e((pk_i)^r, rk_{i\to j}) = e(g^{x_i r}, g^{x_j/x_i}) = Z^{x_j r}$, yielding the same result. Therefore, this scheme satisfies the perfect key-switching property.

In addition, it can be seen that it is also well-spread, since for any message m there can be $q^2 \approx 2^{2\lambda}$ different ciphertexts: component t is sampled randomly from $\mathbb{Z}_q$, and the rest of the components resemble an ElGamal ciphertext, for random $r \in \mathbb{Z}_q$.

Proving $\mathsf{IND\text{-}CCA}_{0,1}$ security

In this section we prove that the scheme is secure under the $\mathsf{IND\text{-}CCA}_{0,1}$ notion, assuming the 3-wDBDHI problem is hard. We use an alternative definition of this hard problem, due to Libert and Vergnaud [12].

Definition 5.10 (3-wDBDHI problem). *Given a tuple $(g, g^a, g^{a^2}, g^{1/a}, g^b, e(g,g)^d)$, the 3-weak Decisional Bilinear DH Inversion problem (3-wDBDHI) in $(\mathbb{G}, \mathbb{G}_T)$ is to decide whether $d = b/a^2$.*

The proof consists on a reduction from the 3-wDBDHI problem to the $\mathsf{IND\text{-}CCA}_{0,1}$ security of the scheme. Suppose that the scheme is not $\mathsf{IND\text{-}CCA}_{0,1}$ secure, then there is an adversary $\mathcal{B}$ that wins the $\mathsf{IND\text{-}CCA}_{0,1}$ game with non-negligible advantage ε, so its success probability is $\frac{1}{2} + \varepsilon$. From this adversary, we can construct an algorithm $\mathcal{A}$ that solves the 3-wDBDHI problem with probability

$\frac{1}{2} + \frac{\varepsilon}{2} - q_{reenc} \cdot 2^{-\lambda-1}$, where q_{reenc} is the number of queries to the re-encryption oracle made by $\mathcal{B}$.

$\mathcal{A}$ receives as input a tuple $(g, g^a, g^{a^2}, g^{1/a}, g^b, Z^d)$, and uses it to simulate the environment for the adversary $\mathcal{B}$. First, $\mathcal{A}$ samples random $t^*, \alpha_1, \alpha_2 \in \mathbb{Z}_q$, and sets $u = (g^a)^{\alpha_1}$ and $v = (g^a)^{-t^*\alpha_1}(g^{a^2})^{\alpha_2}$, so $F(t) = g^{a\alpha_1(t-t^*)}g^{a^2\alpha_2}$. Note that $F(t^*) = g^{a^2\alpha_2}$; we will use this later in the proof.

The public key of the target user is set as $pk^* = g^{a^2}$. For honest users, $\mathcal{A}$ samples random $w_i \in \mathbb{Z}_q$ and sets $pk_i = g^{aw_i}$. For corrupt users, $\mathcal{A}$ simply runs the key generation algorithm and return the resulting public and private key pair $(pk_i = g^{x_i}, sk_i = x_i)$.

Re-encryption keys $rk_{i\to j}$ are generated as follows:

- Honest user to target user: $rk_{i\to *} = (g^a)^{1/w_i}$
- Target user to honest user: $rk_{*\to j} = (g^a)^{w_j}$
- Honest user to honest user: $rk_{i\to j} = g^{w_j/w_i}$
- Honest user to corrupt user: $rk_{i\to j} = (g^{1/a})^{x_j/w_i}$
- Corrupt user to another user: $rk_{i\to j} = (pk_j)^{1/x_i}$

Since the attack model is $\mathsf{CCA}_{0,1}$, it is only necessary to simulate the re-encryption oracle in Phase 1 (i.e., before the challenge). It is possible to simulate correctly all possible re-encryption queries $\mathcal{O}_{reenc}(pk_i, pk_j, CT_i)$ using the re-encryption keys described, except for the case when $pk_i = pk^*$ and user j is corrupt. These queries are solved as below, without requiring the corresponding re-encryption key. Since $pk_i = pk^*$, the ciphertext CT_i is a tuple $(t, F(t)^r, m \cdot Z^r, (pk^*)r) = (t, (g^{a\alpha_1(t-t^*)}g^{a^2\alpha_2})^r, m \cdot Z^r, g^{a^2 r})$. In order to compute the re-encryption, the challenger produces the auxiliary value g^{ar} in the following way:

$$g^{ar} = \frac{C_0}{C_2^{\alpha_2}}^{\frac{1}{(t-t^*)\alpha_1}} = \left(\frac{F(t)}{g^{a^2\alpha_2}}\right)^{\frac{r}{(t-t^*)\alpha_1}}$$

Once $\mathcal{A}$ computes g^{ar}, the re-encryption of C_2 (which, as mentioned before, is the only component of the ciphertext that changes after re-encryption) is as follows:

$$C_2' = e((g^{1/a})^{x_j}, g^{ar}) = Z^{x_j r}$$

Note that this procedure is correct except for the case then $t = t^*$. Since t^* is not disclosed by $\mathcal{A}$ in Phase 1, and t is sampled randomly, the case when $t = t^*$ can happen only with negligible probability $1/q \approx 2^{-\lambda}$, for each re-encryption query.

We assume that the adversary $\mathcal{B}$ can make up to q_{reenc} queries, so the overall probability of this type of events is $q_{reenc} \cdot 2^{-\lambda}$.

The challenge ciphertext is constructed as:

$$CT^* = (t^*, (g^b)^{\alpha_2}, m_\delta \cdot Z^d, g^b)$$

$\mathcal{A}$ runs adversary $\mathcal{B}$ to obtain the guess δ' and decides that $d = b/a^2$ when $\delta = \delta'$. Note that when $d = b/a^2$, the challenge ciphertext is a valid encryption of m_δ under pk^*, where the random exponent r is defined implicitly as $r = b/a^2$.

$$CT^* = (t^*, F(t^*)^{b/a^2}, m_\delta \cdot Z^{b/a^2}, (g^{a^2})^{b/a^2})$$

Therefore, in this case $\mathcal{B}$ guesses δ correctly with probability $\frac{1}{2} + \varepsilon - q_{reenc} \cdot 2^{-\lambda}$ and $\mathcal{A}$ solves the 3-wDBDHI problem with the same probability. On the contrary, when d is random, m_δ is information-theoretically hidden, so the probability of $\mathcal{B}$ guessing δ correctly is $\frac{1}{2}$. The overall success probability of $\mathcal{A}$ is then $\frac{1}{2} + \frac{\varepsilon}{2} - q_{reenc} \cdot 2^{-\lambda-1}$.

A $\mathsf{IND\text{-}CCA}_{2,1}$-secure PRE scheme

Since the previous scheme is $\mathsf{IND\text{-}CCA}_{0,1}$-secure, satisfies the perfect key-switching property and is well-spread, then it can be extended for $\mathsf{IND\text{-}CCA}_{2,1}$ security using our transformation. The resulting scheme is very similar, except that now r and t are not chosen randomly, but using a hash function H. The scheme is as follows:

- $\mathsf{Enc}_2(pk_i, m)$: Sample random $\sigma \in \mathbb{G}_T$ and compute $C_3 = G(\sigma) \oplus m$. Next, the randomness used for encryption is produced as $t||r = H(\sigma, C_3)$. The second-level encryption of m under pk_i is the tuple $CT_i = (t, C_0, C_1, C_2, C_3)$, where

$$C_0 = F(t)^r \qquad C_1 = Z^r \cdot m \qquad C_2 = (pk_i)^r = g^{x_i r}$$

- $\mathsf{Enc}_1(pk_i, m)$: The first-level encryption of m under pk_i is exactly as the second-level, except that $C_2 = e(g, pk_i)^r = Z^{x_i r}$.

- $\mathsf{ReEnc}(rk_{i\to j}, CT_i = (t, C_0, C_1, C_2, C_3))$: Check that the condition $e(C_0, pk_i) = e(C_2, F(t))$ holds; otherwise, output $\perp$. The re-encryption of the second-level ciphertext CT_i is the tuple $CT_j = (t, C_0, C_1, C'_2, C_3)$, where $C'_2 = e(C_2, rk_{i\to j}) = Z^{x_j r}$.

- $\mathsf{Dec}_1(sk_i, CT_i = (t, C_0, C_1, C_2))$: Given a first-level ciphertext CT_i, first decrypt σ as $\sigma = \frac{C_1}{C_2^{1/sk_i}}$, and use it to extract the original randomness $t||r = H(\sigma, C_3)$. Next, check that $F(t)^r = C_0$, $Z^r\sigma = C_1$, and $e(g, pk_i)^r = C_2$ hold and return $m = G(\sigma) \oplus C_3$; otherwise, return $\perp$.
- $\mathsf{Dec}_2(sk_i, CT_i = (t, C_0, C_1, C_2))$: Given a second-level ciphertext CT_i, first decrypt σ as $\sigma = \frac{C_1}{e(g,C_2)^{1/sk_i}}$, and use it to extract the original randomness $t||r = H(\sigma, C_3)$. Next, check that $F(t)^r = C_0$, $Z^r\sigma = C_1$, and $(pk_i)^r = C_2$ hold and return $m = G(\sigma) \oplus C_3$; otherwise, return $\perp$.

According to Theorem 5.9, this scheme is $\mathsf{IND\text{-}CCA}_{2,1}$-secure in the random oracle model.

5.2.6 Towards Other Generic Transformations for Proxy Re-Encryption

It is worthwhile thinking about other possible generic transformations for achieving strong security notions in the context of proxy re-encryption. In Section 5.2.3, we have shown how to directly apply the Fujisaki-Okamoto generic transformation. However, this transformation has two main drawbacks:

- During the decryption procedure it is necessary to encrypt again the ciphertext with the PRE scheme to check if it is the same as the one received. This, of course, produces a computational overhead in the decryption.
- The underlying PRE scheme has to satisfy the perfect key-switching property. This is not always the case, since often the re-encryption function leaves "remnants" of the previous public key, which would make the decryption check to fail inevitably.

It is desirable to come up with other transformations that achieve better security without requiring the perfect key-switching property and the overhead produced by re-computing the ciphertext during decryption. The latter problem is solved in the PKE context by some transformations such as REACT [18] and GEM [29]. These transformations achieve CCA security, also in the random oracle model, by including a hash of the ciphertext that acts as a checksum. For example, the encryption output $\mathsf{Hyb.Enc}(pk, m)$ in REACT is a ciphertext of the form:

$$\mathsf{Hyb.Enc}(pk, m) = (\underbrace{\mathsf{PKE.Enc}(pk, \sigma)}_{e}, \underbrace{\mathsf{Sym.Enc}(G(\sigma), m)}_{c}, \underbrace{H(\sigma, m, e, c)}_{h})$$

The decryption process is simple. It first deciphers σ from e; next, it extracts m from c using the key $G(\sigma)$; and, finally, it verifies that $H(\sigma, m, e, c) = h$.

As in the case of the Fujisaki-Okamoto transformation, directly applying REACT to a PRE scheme would not work in general, since the re-encryption process invalidates the validity check. This check is based on a hash of rest of ciphertext, so a re-encrypted ciphertext will not produce the same hash value h. However, we could modify the transformation so the hash does not include the component e, which is altered during the re-encryption. Then, ciphertexts would be of the form:

$$\mathsf{Hyb.Enc}(pk, m) = (\underbrace{\mathsf{PRE.Enc}(pk, \sigma)}_{e}, \underbrace{\mathsf{Sym.Enc}(G(\sigma), m)}_{c}, \underbrace{H(\sigma, m, c)}_{h})$$

It can be seen that the only difference with respect the original transformation is in the component h. The re-encryption process would be similar to our Fujisaki-Okamoto extension, re-encrypting the component e with the underlying PRE scheme:

$$\mathsf{Hyb.ReEnc}(rk, (e, c, h)) = (\underbrace{\mathsf{PRE.ReEnc}(rk, e)}_{e'}, c, h)$$

Assuming the PRE scheme is correct, the decryption process will work too. Decrypting e' will output σ, so the process remains the same, except for the checksum h, which now does not consider the e' term.

The modification we introduced in this transformation can be seen as a relaxation of the validity check that takes place during the decryption, since we change it from $H(\sigma, m, e, c)$ to $H(\sigma, m, c)$. Dropping the term e from the checksum implies that we are not concerned anymore with possible alterations on this part; however, the checksum still contains the original message m, its encryption c, and the term σ. This prompt us to analyze what alterations are possible in the term e so that the decryption is still correct.

It is clear that σ cannot be altered, since this would imply not being able to recover the key for extracting m from c, so the only options are changing the public key (which is precisely the goal of PRE) or the original random coins used (which is what happens in PRE schemes that do not satisfy the perfect key-switching property). Therefore, any ciphertext that decrypts to the original message would be deemed valid, regardless of being altered (e.g., by re-encryption).

This idea corresponds precisely to the notion of *Replayable CCA* (RCCA) from [145], which can be considered sufficiently secure for many existing applications of CCA security, and indeed, is the target security notion for some PRE schemes, such as the one from Libert and Vergnaud [12]. Note that RCCA also allows any possible kind of alterations, even those not related to the re-encryption of ciphertexts, as long as it decrypts to the original message.

It would appear, then, that a modified REACT transformation could be defined for PRE, achieving a security notion similar to RCCA. However, some of the problems mentioned in Section 5.2.3 also arise here: it is still not possible to perform the validation step during re-encryption, since the hash input contains information that is hidden during this process, and the construction of the re-encryption oracle in the security proof cannot be based on the random oracle tables either. This poses similar challenges to the Fujisaki-Okamoto case when it comes to proving the security of the transformation.

Moreover, REACT puts additional restrictions on the underlying asymmetric scheme. In particular, it requires the scheme to be *one-way secure under plaintext-checking attacks* ($\mathsf{OW\text{-}PCA}$), in order to enable the construction of the decryption oracle in the security proof. Informally, this notion means that the scheme must not allow the adversary to recover the plaintext from a given ciphertext, even if she has access to a plaintext-checking oracle. This oracle is able to tell whether, for input (m, c), the ciphertext c is an encryption of the plaintext m. An implication of this requirement is that, when considering a PCA adversary, the hardness assumption may vary with respect to traditional adversary models, such as CPA. For instance, the ElGamal encryption scheme, which is $\mathsf{OW\text{-}CPA}$ secure under the Computational Diffie-Hellman (CDH) assumption, is $\mathsf{OW\text{-}PCA}$ secure under the Gap Diffie-Hellman (gap-DH) assumption [18]. The explanation of this change is that the PCA oracle usually is equivalent to a solver of a hard decisional problem (e.g., the PCA oracle for ElGamal is equivalent to a DDH oracle). Therefore, the security of the scheme under $\mathsf{OW\text{-}PCA}$ must be based on a hardness assumption that still holds when the adversary has access to an oracle of some hard decisional problem (i.e., the PCA oracle). Some computational problems are still hard when one has a solver for the decisional version, and this is precisely the essence behind the concept of "gap problems", defined by Okamoto and Pointcheval in [28].

In the same way, the extension of REACT to PRE would require the underlying PRE schemes to be $\mathsf{OW\text{-}PCA}$. This is the case of several PRE schemes. For instance, it can be shown that the BBS scheme [43] is $\mathsf{OW\text{-}PCA}$-secure under the gap-DH assumption, as ElGamal. However, at the same time, in order to be able to define the re-encryption oracle, the security proof for this modification of REACT seems to require to take the form of a reduction to the security of the underlying scheme under a notion that provides a re-encryption oracle. For the Fujisaki-Okamoto case, we opted for requiring $\mathsf{IND\text{-}CCA}_{0,1}$ so as to simplify the proof, but in this case $\mathsf{OW\text{-}CCA}_{0,1}$ is a more proper choice. Therefore, the underlying scheme should satisfy the notion of one-wayness under an attack model that combines PCA and $\mathsf{CCA}_{0,1}$.

The resulting transformation would achieve an intermediate notion between RCCA and $\mathsf{IND\text{-}CCA}_{2,1}$, and may be applicable to a wider class of PRE schemes than the Fujisaki-Okamoto extension, since it does not require the schemes to satisfy the perfect key-switching property. However, it is an open issue to analyze these security definitions in detail, in order to provide a complete proof of the security of this potential generic transformation for PRE.

Finally, it is worth mentioning that there is at least another generic transformation, to which the same argumentation seems to apply. The GEM transformation [29] is very similar to REACT, but with a more involved construction that achieves shorter ciphertexts. In GEM, ciphertexts are of the form:

$$\mathsf{Hyb.Enc}(pk, m) = (\underbrace{\mathsf{PKE.Enc}(pk, \sigma)}_{e}, \underbrace{\mathsf{Sym.Enc}(G(\sigma, e), m)}_{c})$$

where r is a random term, $s = F(m, r)$, and $\sigma = s||(r \oplus H(s))$. In order to decrypt a ciphertext (e, c), it first deciphers σ from e, and extracts m from c using the key $G(\sigma, e)$; next, it parses $s||t$ from σ and computes $r = t \oplus H(s)$; and, finally, it verifies whether $F(m, r) = s$.

Being similar to REACT, the strategy for modifying GEM for proxy re-encryption would be similar too. The part of the ciphertext produced by the underlying PRE scheme would not be used as input to the hash function G, so the re-encryption procedure does not break the validation. Thus, the encryption is modified for producing ciphertexts of the form:

$$\mathsf{Hyb.Enc}(pk, m) = (\underbrace{\mathsf{PRE.Enc}(pk, \sigma)}_{e}, \underbrace{\mathsf{Sym.Enc}(G(\sigma), m)}_{c})$$

Therefore, re-encryption is identical to that of the Fujisaki-Okamoto extension:

$$\mathsf{Hyb.ReEnc}(rk, (e, c)) = (\underbrace{\mathsf{PRE.ReEnc}(rk, e)}_{e'}, c)$$

The arguments regarding the security of this extension of GEM are very similar to those for the extension of REACT.

5.2.7 Applicability to NTRUReEncrypt

At the beginning of this section, we mentioned that our original motivation for this contribution was to improve the security notion of the schemes we proposed in the first half of this chapter. Our results are mixed: although our proposed transformations can be applied to a wide range of PRE schemes, unfortunately, they cannot be applied to our NTRU-based schemes.

The application to $\mathsf{NTRUReEncrypt}$ is out of scope since this scheme is not provably secure, while all our proposals require the original scheme to fulfill some predetermined security notion. In contrast, the second variant $\mathsf{PS\text{-}NTRUReEncrypt}$ was, indeed, proven CPA-secure under the LWE assumption.

However, it is easy to see that $\mathsf{PS\text{-}NTRUReEncrypt}$ does not fulfill the perfect key switching property, so it is not possible to directly apply our variant of Fujisaki-Okamoto. With respect to our REACT and GEM proposals, recall that they require the scheme to be $\mathsf{OW\text{-}PCA}$, which means that the adversary should not be able to break the one-wayness even if he has access to a plaintext checking oracle. In practice, this oracle is usually implemented as a decisional oracle of the corresponding hard problem, thus forcing the scheme to rely on a gap variant of the original problem. This represents a unavoidable obstacle, since the decisional and computational version of LWE are equivalent, as pointed out by Peikert in [146], which negates the possibility of defining a gap problem for LWE. This problem affects several schemes, such as [47, 46, 48] and our proposal $\mathsf{PS\text{-}NTRUReEncrypt}$, so they cannot achieve $\mathsf{OW\text{-}PCA}$ security.

5.2.8 Summary

In this section we analyze the integration of generic transformations to proxy re-encryption and find both positive and negative results. On the one hand, we first describe why it is not possible to directly integrate known transformations, such as Fujisaki-Okamoto and REACT, with weakly-secure PRE schemes due to general obstacles coming from the constructions and the security models, and we show twelve PRE schemes that are flawed as a consequence of these problems. These transformations are artifacts conceived for securing public-key encryption schemes, and cannot be used as is for proxy re-encryption due to the special nature of the re-encryption capability. On the other hand, we also show that, under some assumptions that include the satisfaction of a new property of PRE called "*perfect key-switching*", the Fujisaki-Okamoto transformation can be used to generically bootstrap a weak notion of security ($\mathsf{IND\text{-}CCA}_{0,1}$) into a much stronger notion ($\mathsf{IND\text{-}CCA}_{2,1}$), in the random oracle model. However, to achieve full CCA-security (i.e., $\mathsf{IND\text{-}CCA}_{2,2}$), it appears to be necessary to apply ad-hoc modifications. For illustrating our proposal we present a PRE scheme that satisfies the conditions for applying the Fujisaki-Okamoto extension and show the resulting scheme after the transformation.

Other generic transformations for public-key encryption are also discussed for its application in proxy re-encryption. We show how the REACT and GEM transformations [18, 29] can be modified to support re-encryptions. Since the perfect key-switching property is no longer required, these proposals are potentially applicable to a wider class of schemes, which makes them very attractive. In addition, they are more efficient than the Fujisaki-Okamoto transformation, since they do not require to reconstruct the ciphertext during decryption. The resulting transformations seem to achieve an intermediate notion between Replayable CCA (RCCA) and $\mathsf{IND\text{-}CCA}_{2,1}$, although it is an open issue to analyze these security definitions in detail, in order to provide a complete proof of the security of these constructions. This is left as future work.

Other future lines of research include working towards concrete estimations of the obtained security level of the extended Fujisaki-Okamoto transformation. Finally, the transformations discussed here are all defined for the random oracle model. It is an open problem to devise generic transformations that are valid in the standard model.

Chapter 6

Applications of Proxy Re-Encryption

The final part of this thesis is devoted to concrete applications of proxy re-encryption. In the introduction of this thesis we presented the secure data sharing scenario as our motivating starting point. Section 3 further discusses this scenario, as it is often addressed in the applications of PRE in the current scientific literature. This fact is in line with one of the research postulates of this thesis, namely, that PRE is a suitable tool within solutions to the secure data sharing scenario. The applications presented in this chapter can be seen as particular instantiations of this generic setting, all of them solved using proxy re-encryption.

The first application is a privacy-preserving model for Identity Management as a Service, called BlindIdM. In this model, identity information is stored encrypted at cloud identity providers and processed in a *blind* manner, which removes the necessity of trusting that the cloud identity provider will not read the data. One of the prime aspects of this model is that it is integrated to standard identity management protocols, in particular with the SAML 2.0 framework.

The second application is inspired by one of the application use cases discussed in the introduction to this thesis: the case of Big Data Analytics in the cloud. We show here a cryptographically-enforced access control system for Hadoop, which is one of the most prominent Big Data Analytics frameworks in use nowadays. In this system, the data is in encrypted form and the owner can delegate access rights to computing clusters in the cloud for processing.

These two applications are obvious instances of the generic secure data sharing scenario: encrypted information is stored and managed by a semitrusted cloud provider. In contrast, the third application is a less evident instantiation, since it describes the construction of an escrowed decryption system using proxy re-encryption. The basic idea is to use the conventional PKE-based functions of the PRE scheme as a regular PKE scheme, and to use the re-encryption function to define an escrowed decryption capability. A differential aspect of this proposal is that this capability is distributed among a set of trusted parties called *escrow custodians*, which can respond to petitions from *escrow authorities* (e.g., the government) by re-encrypting ciphertexts; only if all of the custodians participate in this process, the escrow authority can achieve the escrow decryption.

6.1 BlindIdM: Privacy Preserving Identity Management as a Service

6.1.1 Introduction

Cloud computing has recently burst onto the technology and business scenes, promising great technical and economic advantages. One of the principal benefits of cloud computing is that it represents a model of utility computing, capable of offering on-demand provisioning of computing resources, such as storage, processing and networking. This provision of resources is metered for billing purposes, making a "pay-as-you-go" model possible that permits companies and organizations to transform capital expenditures, such as acquisition of specific hardware, into operational expenditures; this paradigm can be contrasted with previous models, based on the acquisition of equipment and software licences. The main benefits that organisations expect from adopting the cloud computing paradigm are an improved flexibility and scalability of their IT services, as well as the resulting cost savings from the outsourcing of such services [1].

Within the internal processes of most organizations, identity management stands out for its ubiquitous nature, as it plays a key role in authentication and access control. However, it also introduces an overhead in cost and time, and in most cases, specialized applications and personnel are required for setting up and integrating identity management systems, as well as for managing identity information. As has already happened for other kinds of services, the cloud paradigm represents an innovative opportunity to externalize the identity management processes, offering what has been called *Identity Management as a Service* (*IDaaS*) [147]. Identity Management as a Service is the cloud industry's response to the problem of identity management within companies and organizations, allowing them to outsource the identity management service from their internal infrastructures and deploy it in the cloud provider. In other words, it permits moving identity management from an *on-premise* delivery model to an *on-demand* model. Additionally, IDaaS opens up a new business opportunity for cloud providers and vendors, broadening their service offering.

As described in the introduction of this thesis, the advent of cloud computing has raised great expectations regarding efficiency, cost reduction and simplification of business processes, but at the same time has also increased security and privacy risks. This very same conflict also applies to the IDaaS case: although it offers organizations a great opportunity to cut capital costs (as well as some operational ones, such as specialized personnel), it also introduces a variant of one of the classic problems of cloud computing: the loss of control over outsourced data, which in this case is information about users' identity. For instance, according to a recent survey from Cisco to IT specialists and decision makers [148], data protection is regarded as the top barrier that impedes the migration to the cloud.

The principal motivation behind this contribution is putting the identity provider into the cloud landscape, where data storage and processing could be offered by possibly untrusted cloud providers, but still offer an identity management service that guarantees user's privacy and control. To this end, we define BlindIdM, a privacy-preserving IDaaS model where identity information is stored and processed in a blind manner, removing the necessity of trusting that the cloud identity provider will not read the data. Such a concept is a novel contribution to both the field of identity management and privacy-enhanced technologies. Our model, which uses the standard SAML 2.0 as the underlying identity management protocol, applies proxy re-encryption techniques to achieve end-to-end confidentiality of the identity information, while allowing the cloud to provide an identity service. This can be seen as an instantiation of the secure data sharing scenario, applied to the problem of outsourcing the identity management service.

6.1.2 The Path to Identity Management as a Service

Identity information is steadily becoming an essential enabler of today's digital society, as it is considered a key component in the interactions between end-users, service providers, and intermediaries. At the same time, it is also becoming more and more valuable for the organizations that manage this kind of information because of its usefulness for marketing and strategic development purposes or, simply, to be sold to interested third parties [149]. Thus, identity management remains an important challenge in the field of information security and privacy, and spans several subareas, such as usability and user experience, authentication methods, or trust and reputation management [150].

Within the organizations' environment, identity management is one of the most commonly deployed services because of its importance for authentication and access control. However, it is regarded by enterprises as one of the most time-consuming and complex tasks within their internal business processes. It introduces an overhead in cost and time, and in most cases, specific applications and personnel are required for managing, integrating and maintaining this service. This is even more troublesome when some kind of identity service is offered to external users, such as clients, contractors, or providers.

An identity management system (IMS) facilitates the creation, storage, and usage of the identity information of the individuals from a organization [151]. Traditionally, identity management systems were designed to be used internally in the organizations and companies, in a centralized and local manner, which has been called the *silo model*. However, as the Internet has gained in popularity, the number of possible interactions that a user can have with service and resource providers has increased dramatically. This fact leads to an unwanted effect called *identity fragmentation*, since users are then obliged to register several accounts, one for each service provider; that is, their identity information is partially replicated and fragmented throughout a group of service providers. Furthermore, each of these fragments of identity is normally associated with passwords that must be memorized by the users, which is prone to usability and security problems, such as password reuse. The problem of identity fragmentation evidences the drawbacks of the traditional isolated model of identity management, and has motivated the development of more flexible schemes that are centered on enhancing the interoperability.

Federated Identity Management

Federated Identity Management (*FIM*) is a solution to overcome these difficulties. FIM is a set of distributed technologies and processes that enable information portability between different domains, which permits both a dynamic distribution of identity information and delegation of associated tasks, such as authentication or user provisioning. Thus, organizations coordinate with each other to form federations for exchanging identity information. One of the key aspects of this model is the establishment of trust relationships between the members of the federation, which enables them to believe the statements made within the federation. This way, although users are authenticated by their local organization, they are able to access services and resources from other organizations of the federation. SAML [19], Shibboleth [152] and WS-Federation [153] are examples of systems and standards for federated identity management.

The parties involved in a federated identity interaction are required to mutually exchange identity information for identification and authentication purposes regardless of whether they have previous knowledge of each others' identity information or not. The main actors that participate in these interactions are [154],[155]:

- *Users*, the subjects of the identity information; most of the times they are also the principal source of this information. Users are generally the actors that request resources and services through their interaction with applications and online services. Users perform this interaction through a *user agent*, which is usually a browser, but it could also be a specific application.

- *Service Providers* (*SP*), the entities that provide services and resources to users or other entities. In a federated identity management context, service providers outsource the processes of authentication and management of users to identity providers. Because of this, service providers act as consumers of user's identity information, following a determined identity management protocol.

- *Identity Providers* (*IdP*), which are specialized entities that are able to authenticate users and to provide the result of this authentication to service providers, without revealing additional information about the user. The information that they exchange with service providers may even be just a statement about the success of the authentication of the user, enabling the user to access the service anonymously. Identity providers are also responsible for managing the identity information of their associated users, and in some cases, they may certify it.

Figure 6.1 shows a high-level view of a federated identity setting, where a host organization acts as a federated identity provider. In this setting, an employee from the host organization requests a service from the service provider, who in turn asks the organization for identity information about its employee.

Before accepting the supplied identity information, the service provider must trust the host organization, acknowledging it as a reliable identity provider. Trust in this case is normally achieved out of band through some physical transaction such as a legal agreement, and later reflected in the identity federation system through some technical mechanisms such as WS-Trust or SAML Metadata; in practice, each consumer entity has a list of trusted issuers of identity information.

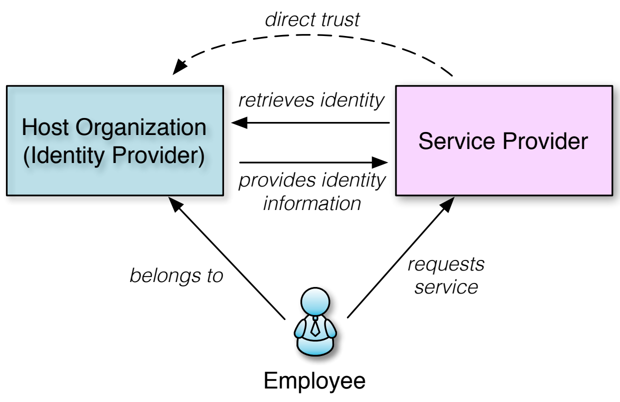


Figure 6.1: Federated Identity Management System

Identity Management as a Service

The federated identity model is widely used in organizations, deployed as an on-premise service. Although it has led to great advantages with respect to interoperability of identities, it has also introduced cost and time overheads, since it usually requires specialized applications and personnel for setting up, integrating and managing this process.

However, the emergence of the cloud as a ubiquitous technology within today's organizations, has led to Identity Management as a Service, a natural answer from the cloud industry to the enterprise identity management problem. Examples of such cloud-based identity services are Windows Azure Active Directory [156] and CA CloudMinder Identity Management [157]. IDaaS can be seen as a refinement of the federated model, which takes the efficiency of the cloud in its favor for offering specialized outsourcing of identity management. Among the benefits of Identity Management as a Service we find:

- More flexibility, scalability and stability for high demand environments, with a growing number of users and thousands of identities.
- Reduction of costs, since IDaaS providers can focus on providing more efficient and specialized identity services to organizations.
- Better security measures and mechanisms, implemented in dedicated systems and facilities.
- Improved compliance and business processes audits due to the high specialization that an IDaaS provider can achieve. These providers can also implement common policies for all their customers at a lower cost.

There are, however, multiple risks associated with Identity Management as a Service; most of them are a consequence of the identity providers managing and storing a large amount of identity information [158], while some of them are inherent to the cloud computing paradigm [159, 160]. We identify the following principal risks:

- Identity providers are appealing targets to attackers as they represent a single point of failure because they centralize users' personal information; security breaches and insider attacks are potentially dangerous as they may disclose the personal information of a large number of users. The fact that this kind of information is protected by specific regulations, such as the European Data Protection Directive, in the case of the EU [161], demands a strong protection of its storage, processing and communication.

- Cloud providers are susceptible to being subpoenaed for users' data, in the case there is some legal, administrative or criminal investigation running [162]. What is worse, it is possible that providers respond positively to these requests for information, even if they are not made with the proper judicial guarantees, due to a lack of legal understanding.

- Identity providers are in a privileged position to collect additional information about users without their consent, such as the sites the user visits, for profiling purposes.

- In the absence of cryptographic means, it is not possible to actually limit the access of cloud providers to the data they must steward. That is, there is almost no risk of being discovered accessing users' information without their consent.

- Another major risk is the existence of cloud providers in foreign countries (i.e., located in a different country to the owner of the data) with different, and possibly conflicting, laws and regulations regarding privacy and data protection. For example, in the case of the US, the USA PATRIOT Act [163] allows the government to check the data that is processed or stored within its jurisdiction, even without the knowledge of the owner of the data.

- The security guarantees and requirements of the cloud providers are disparate. These requirements not only include technological aspects, but also policies regarding the hiring of staff, access to premises and equipment, physical security measures, etc.

Hence, it is obvious that externalizing the management of identity information to the cloud implies a loss of control for users and organizations. This in turn signifies an empowerment of cloud identity providers; that is, there is an inversion of the control over the identity information. This leads to the identity provider accumulating enough power for the users to incur damages, losses or risks in the case of a disclosure of private data.

6.1.3 BlindIdM: Privacy-Preserving IDaaS

We have seen before that in an IDaaS scenario, organizations entrust their corporate identity information to cloud identity providers, which are then responsible for storing and managing this information. These systems rely on the existence of a strong relationship of trust between the organizations and the cloud identity providers, since they trust that their identity information will be managed properly and that the provider will respect the confidentiality; however, current cloud providers do not implement real mechanisms for preventing themselves from betraying this trust. This concern led us to conceive of the concept of *Blind Identity Management* (BlindIdM), a system whereby the cloud identity provider is able to offer an identity information service, without knowing the actual information of the users; that is, it provides this service in a *blind*[1] manner.

This is a great innovation with respect to current identity management systems, where users' identity information is managed by the identity provider and the user is obliged to trust that the provider will make proper use of his data and will guarantee its protection. Our intention is that this model will enable organizations to choose a cloud identity provider without necessarily establishing a strong bond of trust with it; i.e., they do not have to trust that the cloud identity provider will respect data privacy. Instead, the sturdiness of the underlying cryptographic schemes should be sufficient to guarantee such protection.

In contrast to a full outsourcing of the identity management system, we have opted for a hybrid approach, where the authentication remains on-premises at the host organization. The novel aspect of our proposal lies in the protection of data: the host organization encrypts users' identity information prior to outsourcing it to the cloud, in such a way that it is still usable by the cloud identity provider without being able to be read.

Before continuing, we will firstly describe the general setting and trust model that we will consider; secondly, we will briefly describe the underlying identity management framework, SAML 2.0; thirdly, we will explain in detail our proposal

[1]The term *blind* is used here in an analogous way as in *blind* signature, which is a signature scheme that enables the signer to perform a signature without knowing the content of the underlying message.

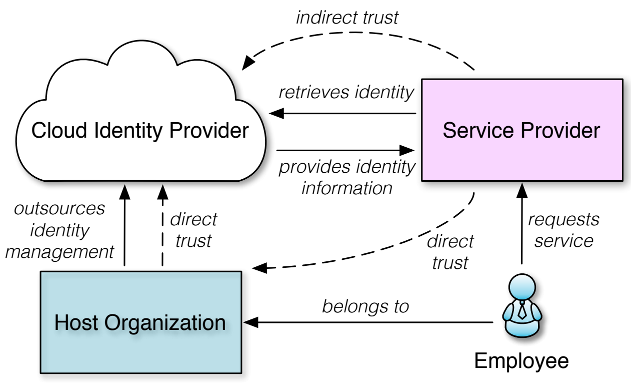


Figure 6.2: Relationships between entities

for privacy-preserving Identity Management as a Service; and finally, we will provide an analysis of our proposal.

Trust Model and Assumptions

In our model, we will assume a federated identity setting, similar to that shown in Figure 6.1, but where the host organization partially outsources the identity management processes to a cloud identity provider, while retaining the authentication service on-premises. The cloud identity provider now acts as an intermediary in the identity interactions, and is also in charge for storing and supplying identity information; Figure 6.2 shows this setting. Optionally, other kinds of actors may come into play such as attribute issuing authorities, certification providers, identity brokers, etc; however, we will restrict the scope of this work to the basic case.

The goal of our approach is to provide a means for constructing *blind identity providers*, which could be capable of operating without having access to users' information. In other words, we consider the cloud identity provider as an adversary, as described in Section 1.1.1. In that section, we identified three types of cloud providers depending on their capabilities and their level of trust: fully trusted, honest-but-curious, and malicious. In turn, the honest-but-curious category can be subdivided in data-curious and access-curious, which denote if the main interest of the cloud provider is the data itself or the access patterns, respectively. In the BlindIdM model, as in the rest of this thesis, we restrict ourselves to data-curious providers; we will assume then that the identity provider may have some incentive to read users' data without their consent, but will not try to track users' behavior and access patterns.

The problem that arises from considering access-curious providers has been widely studied, as it is what one normally encounters when privacy is addressed in the context of identity management; anonymization techniques, such as pseudonyms [164], are among the solutions that are usually proposed in this respect. As aforementioned, in this thesis we will not tackle this problem and we will focus instead on protecting data privacy; however, a more complete solution that takes this issue into account is left open as future work.

As stated before, Figure 6.2 depicts the main interactions in our proposed model. With regard to trust relationships, the introduction of the cloud identity provider makes them more complex than in the federated identity setting. On the one hand, we still assume that the service provider fully trusts the host organization as a reliable and valid source of identity information; this trust is achieved as in the federated case, through out-of-band agreements and metadata. On the other hand, since the host organization outsources part or all of its identity management system, it is clear that the organization must have some level of trust in the cloud identity provider. In this case, trust is reflected in SLAs and in metadata as well. As a consequence of these direct trust relationships, we assume that in this setting the service provider indirectly trusts the cloud identity provider, as there is an implicit chain of trust between the two entities. That is, there is no explicit agreement or metadata that expresses this trust relationship, but the service provider can be confident of the trustworthiness of the cloud provider.

Underlying Identity Management Framework

Apart from the use of proxy re-encryption, our system is based in SAML 2.0 as the underlying identity management protocol. We have chosen SAML because of its wide adoption, its extensibility and its ingrained mechanisms for establishing trust between the entities.

SAML 2.0 (Security Assertion Markup Language) [19] is a standard XML-based framework that enables the description and exchange of identity information between different security domains. With SAML, identity information is expressed in the form of assertions, which are a set of statements about a subject; these statements cover different aspects, such as authentication, authorization and identity attributes.

The SAML framework also specifies the protocols for issuing and exchanging assertions, such as the Authentication Request protocol, a request/response protocol that permits entities to ask for an authentication statement, and optionally identity attributes. In this protocol, the requester sends a SAML `AuthnRequest`

```
<saml:Attribute Name="givenName"
   ext:OriginalIssuer="http://idp.host.org"
   xmlns:ext="urn:oasis:names:tc:SAML:attribute:ext">
   <saml:AttributeValue>John</saml:AttributeValue>
</saml:Attribute>
```

Figure 6.3: SAML Attribute

message to an identity provider, which in turn replies with a SAML `Response` containing assertions about the request. The technical details about how to achieve this message exchange depend on the specific SAML bindings and profiles in use; here, we will use the Web Browser SSO Profile and HTTP POST Binding as a basis for the identity interactions. The Authentication Request protocol also permits the hybrid approach for authentication we have chosen, as this possibility is considered in the SAML specification. In this case, the cloud identity provider acts as a *proxying identity provider*, and the host organization is the *authentication provider*.

SAML attributes are used to express identity information about the subject of the assertion; Figure 6.3 shows an example of such element. In our proposal, we make extensive use of this construction, as we take it as the basic medium for conveying encrypted identity information.

SAML also allows the expression of metadata for both service and identity providers using the SAML Metadata specification [165]. Metadata is what enables the expression of prior trust relationships and makes secure transactions possible.

Description of BlindIdM

We now describe BlindIdM, a privacy-preserving model for blind Identity Management as a Service. In this model, as in the usual identity management systems, there are three main types of actors, namely, users, service providers and identity providers. In our scenario, the host organization (including all the employees) acts as the user, and the identity management of the organization is outsourced to a cloud identity provider. These entities are capable of interacting following a pre-defined identity management protocol. From a high-level viewpoint, the goal of these interactions is the exchange of identity information, that generally flows from the user (in our case, from the host organization), acting as a source of information, to the service provider, acting as a consumer of information. BlindIdM permits this information to leave the source and arrive at its destination in an encrypted form, achieving end-to-end confidentiality. Our goal now is to describe how encrypted information can flow from the host organization to service

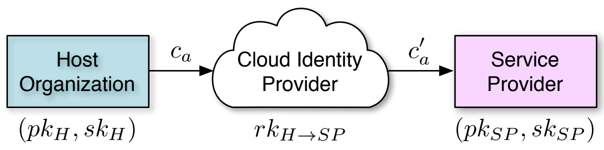


Figure 6.4: Data flow of BlindIdM

providers, without the identity provider being able to read it.

A high-level diagram of our proposal is shown in Figure 6.4; this diagram depicts the main flow of information in our system, where the host organization encrypts the identity information under its public key pk_H and sends it to the cloud identity provider. The use of proxy re-encryption enables the identity provider to transform these ciphertexts into encrypted attributes under the public key of the service provider, pk_{SP}; in order to do so, the identity provider needs a re-encryption key $rk_{H \to SP}$ generated by the host organization and provided beforehand.

We will now proceed to detail the steps of the operation of the BlindIdM system. Note that we are using SAML as the underlying protocol; here, we will describe an identity interaction using the SAML Authentication Request protocol. As for proxy re-encryption, our model does not require a specific scheme.

Phase 1. Generation of public and private keys. Both the host organization and the service provider create their pairs of public and private keys, respectively (pk_H, sk_H) and (pk_{SP}, sk_{SP}). For illustration purposes we will assume that there is only one service provider, but there could be any number of service providers. Furthermore, the service provider can create its pair of keys at any moment, as long as it is done before phase 3.

Phase 2. Encryption of identity information and outsourcing. The host organization must encrypt the identity information of its employees prior to externalizing it to the cloud. To do so, they use their public key pk_H; for simplicity we will assume that identity information is in the form of attributes, where each attribute a is a tuple $(a.metadata, a.value)$, where $metadata$ describes any metadata about the attribute, including its name and format. Therefore, the identity information of each employee U is a pair $(ID_U, \{a : a \text{ is an attribute of the employee } U\})$, where ID_U is the identifier of such employee.

In our approach we encrypt just the attribute value, leaving the attribute metadata in clear, which eases the integration of our solution with existing directory

```
<saml:Attribute Name="givenName"
   ext:OriginalIssuer="http://idp.host.org"
   xmlns:ext="urn:oasis:names:tc:SAML:attribute:ext">
   <saml:AttributeValue>
      <!-- Encrypted AttributeValue content -->
      <xenc:EncryptedData
         Type="http://www.w3.org/2001/04/xmlenc#Content">
         <!-- Symmetric encryption algorithm (AES-128) used -->
         <xenc:EncryptionMethod
            Algorithm=".../xmlenc#aes128-cbc"/>
         <!-- Encapsulated symmetric key using AFGH scheme -->
         <ds:KeyInfo>
            <xenc:EncryptedKey
               Recipient="http://serviceprovider.com">
               <!-- Proxy re-encryption algorithm (AFGH05)
                  used for key encapsulation -->
               <xenc:EncryptionMethod
                  Algorithm="urn:proxyreencryption:afgh05"/>
               <!-- Encapsulated symmetric key -->
               <xenc:CipherData>
                  <xenc:CipherValue>
                     Ke41X+w...
                  </xenc:CipherValue>
               </xenc:CipherData>
            </xenc:EncryptedKey>
         </ds:KeyInfo>
         <!-- Encrypted SAML Attribute Value content -->
         <xenc:CipherData>
            <xenc:CipherValue>39UCe3/sA...</xenc:CipherValue>
         </xenc:CipherData>
      </xenc:EncryptedData>
   </saml:AttributeValue>
</saml:Attribute>
```

Figure 6.5: SAML Attribute with encrypted AttributeValue content

services. The cloud provider is aware of the name of the attributes, but not of their content. It is also worth mentioning that we do not directly encrypt attribute values with the PRE encryption function $\mathsf{PRE.Enc}$; instead, we use a hybrid approach, encrypting a fresh key K_a for each attribute a, which is then passed to a symmetric encryption algorithm $\mathsf{Sym.Enc}$ (such as AES) for encrypting the attribute value. This way, the PRE encryption function is only used to cipher a fixed-length input (the key K_a), whilst the symmetric algorithm performs the bulk of the work. This approach is used not only for efficiency, but for input length reasons, since attribute values have a wide range of possible lengths. Thus, an outsourced attribute c_a is generated in the following way:

$$c_a = (a.metadata, \mathsf{PRE.Enc}(pk_H, K_a), \mathsf{Sym.Enc}(K_a, a.value))$$

An example of the SAML representation of an outsourced attribute c_a with an encrypted value is shown in Figure 6.5; this is the encrypted version of the same attribute shown in Figure 6.3. SAML permits putting any arbitrary content within the `AttributeValue` element, so we have used the XML Encryption specification [166] (used also in the SAML core specification) to express the hybrid encryption mechanisms and convey the cipher data and the key material.

Once the encryption process is complete, the host organization outsources the identity information to the cloud identity provider. The identity information in the cloud for each employee is in the form $(ID_U, \{c_a : c_a$ is an outsourced attribute of the employee $U\})$. It is important to note that this same approach can be used to update attributes, so it does not represent any difficulty for our system; the new encrypted attribute simply substitutes the previous one.

Phase 3. Trust establishment and generation of re-encryption keys. During this phase, service providers establish a trust relationship with the host organization, which is needed for deeming as valid the claims it makes. This relationship is bidirectional, since the host organization must also trust the service provider in order to release the identity information. As in the case of federated identity management, this trust relationship is usually a consequence of a prior out-of-band agreement.

SAML permits the expression of metadata for both service providers and identity providers using the SAML Metadata specification [165]; indeed, metadata is essential to the proper operation of some of the SAML protocols. The publication of keys and certificates, such as X.509 certificates, through the `KeyDescriptor` element, is among the crucial aspects that are covered in the metadata; we can make use of this method to publish the service provider's public key pk_{SP}. SAML also permits expressing which attributes are required during authentication requests, using a specific element called `AttributeConsumingService`. Figure 6.6 shows an extract of the metadata file of the service provider, where these elements appear.

The cloud identity provider also needs a metadata file, but in this case, it does not require special attention, as the cloud provider does not have any key material prone to be distributed, other than its X.509 certificates. This metadata also contains the information about service endpoints for SAML protocols.

Once the host organization trust a certain SP, they use its public key pk_{SP} together with their private key sk_H to create the re-encryption key $rk_{H \to SP}$, which is then sent to the cloud identity provider. The host organization obtains the public key from the service provider's metadata. This key allows the identity provider to re-encrypt the ciphertexts in order to be decryptable by the service provider using its private key sk_{SP}. The re-encryption key can be seen also as

```
<md:EntityDescriptor
   entityID="http://www.serviceprovider.com" ...>
   <md:SPSSODescriptor ...>
      <!-- Proxy re-encryption public key of SP -->
      <md:KeyDescriptor use="encryption">
         <ds:KeyInfo>
            <ds:KeyName>
               Proxy Re-Encryption Public Key
            </ds:KeyName>
            <ds:KeyValue>
               <pre:PublicKey
                  Type="urn:proxyreencryption:afgh05">
                  YTdUs3...
               </pre:PublicKey>
            </ds:KeyValue>
         </ds:KeyInfo>
      </md:KeyDescriptor>
      ...
      <!-- AttributeConsumingService elements -->
      <md:AttributeConsumingService index="1">
         <md:ServiceName>Service Provider</md:ServiceName>
         <md:RequestedAttribute
            Name="givenName" isRequired="false"/>
         <md:RequestedAttribute
            Name="cn" isRequired="true"/>
      </md:AttributeConsumingService>
      ...
   </md:SPSSODescriptor>
</md:EntityDescriptor>
```

Figure 6.6: SAML Metadata of the service provider

an authorization token, and can be revoked by the host organization by asking the cloud identity provider to remove it. Bear in mind that we are assuming an honest-but-curious cloud provider, which will follow the instructions given by the host organization. Another possibility could be to encrypt again the attributes with a new public key pk'_H, and upload them to the cloud provider; this option is highly inefficient but does not require any other changes in our model.

Phase 4. Identity information interaction. Once our system is properly deployed, an employee may want to retrieve a resource from the service provider, which requires authentication and additional identity information from the employee. The goal in this phase is the dispatch of identity information of an employee, which is stored in the cloud identity provider, to the service provider. Moreover, the authentication takes place within the hosted organization, so its result must also be communicated to the service provider. As stated before, we are using SAML as the underlying identity protocol, and in particular, we will describe how our model fits within the Authentication Request protocol. In our case, this protocol permits the service provider to request an assertion about the

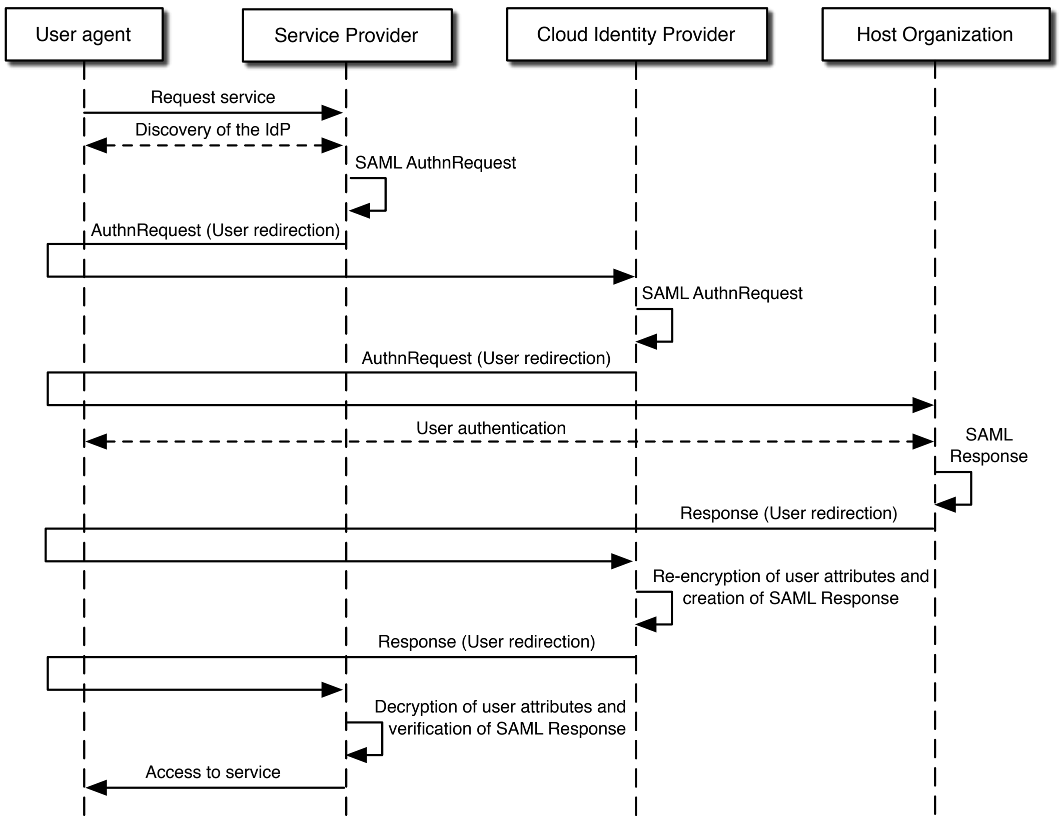

User agent
Service Provider
Cloud Identity Provider
Host Organization
Request service
Discovery of the IdP
SAML AuthnRequest
AuthnRequest (User redirection)
SAML AuthnRequest
AuthnRequest (User redirection)
User authentication
SAML Response
Response (User redirection)
Re-encryption of user attributes and creation of SAML Response
Response (User redirection)
Decryption of user attributes and verification of SAML Response
Access to service

Figure 6.7: Sequence diagram of the authentication request

identity of the employee, including encrypted attributes.

Figure 6.7 shows the protocol interaction that takes place between the four entities involved: the employee as a user, the service provider as requester, the cloud provider as a proxying identity provider, and the host organization as authentication provider. The full sequence, assuming there is no security context for the employee at the service provider, works as follows:

1. The employee tries to access a protected resource offered by the service provider.

2. A discovery process occurs between the service provider and the user to find out the location of the identity provider; this process is out of the scope of both SAML and our system. The simplest option is that the user simply provides the location of the identity provider; a more complex option could be to integrate SAML with other discovery mechanisms, such as Yadis or XRI.

3. The service provider creates a SAML `AuthnRequest`.
4. The user agent gets redirected to the cloud identity provider through an HTML form, as explained in the SAML HTTP POST binding [167].
5. The cloud identity provider receives the authentication request. As we have mentioned before, BlindIdM uses a hybrid IDaaS approach, where the authentication remains at the premises of the host organization; as a consequence, the cloud provider must devolve the authentication to the host organization using the proxying mechanisms defined in the SAML Authentication Request protocol. In this case, the cloud identity provider issues a second authentication request, addressed to the host organization.
6. The user agent gets redirected to the host organization.
7. The employee is authenticated to the host organization. The authentication method is beyond of the scope of this work; for simplicity, we will assume that a password-based method is used.
8. The cloud provider constructs a SAML `Response` that responds to the second authentication request, and that conveys the authentication result and the identifier of the employee.
9. Once again, the user agent gets redirected to the cloud identity provider, delivering the authentication response from the host organization.
10. The cloud identity provider gets the encrypted attributes associated to the provided employee's identifier and, using the proper re-encryption key $rk_{H \to SP}$ (obtained during the previous phase), proceeds to re-encrypt the ciphered attributes; actually, for each attribute a, it only has to re-encrypt the ciphered symmetric key K_a. Let $c_a = (c_{a,1}, c_{a,2}, c_{a,3})$ be one of the outsourced attributes, and PRE.ReEnc be the re-encryption function; then, the re-encrypted attribute c'_a is:

$$c'_a = (c_{a,1}, \mathsf{PRE.ReEnc}(rk_{H \to SP}, c_{a,2}), c_{a,3}))$$

    Once the attributes are re-encrypted, the cloud provider issues a SAML `Assertion` that includes the encrypted attributes within an attribute statement, as well as the authentication statement from the host organization (obtained in the previous step); an `AuthenticatingAuthority` element is also included in the authentication statement, which references the authentication provider (in this case, the host organization). The cloud provider then encloses the assertion in the authentication response. Figure 6.8 shows the SAML `Assertion`.

```
<saml:Assertion ID="#ASSERTION_ID" IssueInstant="2012-05-06T11:39:08Z" Version="2.0">
   <saml:Issuer>https://cloudidp.com</saml:Issuer>
   <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">...</ds:Signature>
   <saml:Subject>
      <saml:NameID Format="urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress">
         jdoe@host.org
      </saml:NameID>
      <saml:SubjectConfirmation Method="urn:oasis:names:tc:SAML:2.0:cm:bearer">
         <saml:SubjectConfirmationData Recipient="http://serviceprovider.com"
            InResponseTo="#REQUEST_ID" NotOnOrAfter="2012-05-06T11:43:36Z"/>
      </saml:SubjectConfirmation>
   </saml:Subject>
   <saml:Conditions NotBefore="2012-05-06T11:38:36Z" NotOnOrAfter="2012-05-06T11:43:36Z">
      <saml:AudienceRestriction>
         <saml:Audience>http://serviceprovider.com</saml:Audience>
      </saml:AudienceRestriction>
   </saml:Conditions>
   <saml:AuthnStatement AuthnInstant="2012-05-06T11:38:36Z" SessionIndex="#SESSION_ID">
      <saml:AuthnContext>
         <saml:AuthnContextClassRef>
            urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport
         </saml:AuthnContextClassRef>
         <saml:AuthenticatingAuthority>http://idp.host.org</saml:AuthenticatingAuthority>
      </saml:AuthnContext>
   </saml:AuthnStatement>
   <saml:AttributeStatement>
      <!-- SAML Attribute with encrypted AttributeValue content -->
      <saml:Attribute Name="givenName" xmlns:ext="urn:oasis:names:tc:SAML:attribute:ext"
                ext:OriginalIssuer="http://idp.host.org">
         ...
      </saml:Attribute>
   </saml:AttributeStatement>
</saml:Assertion>
```

Figure 6.8: SAML Assertion including an encrypted AttributeValue

11. Once again, the user agent gets redirected to the service provider, delivering the authentication response from the cloud identity provider, which includes the re-encrypted attributes.

12. The service provider verifies the authentication response and extracts the encrypted attributes from the assertion. Now, it simply has to decrypt the ciphertexts using its private key sk_{SP}. Lets assume that $c'_a = (c'_{a,1}, c'_{a,2}, c'_{a,3})$ is one of the received attributes, $\mathsf{Sym.Dec}$ is the symmetric decryption algorithm, and $\mathsf{PRE.Dec}$ is the PRE decryption function; then, the decrypted attribute a' is:

$$\begin{aligned} a' &= (c'_{a,1}, \mathsf{Sym.Dec}(\mathsf{PRE.Dec}(sk_{SP}, c'_{a,2}), c'_{a,3})) \\ &= (a.metadata, \mathsf{Sym.Dec}(K_a, c'_{a,3})) \\ &= (a.metadata, a.value) \end{aligned}$$

Analysis

The main requirement of our model is to achieve end-to-end confidentiality for the identity information, enabling it to be stored in the cloud and managed blindly. Taking into account the trust model that we are using, that is, honest-but-curious providers, we argue that this requirement is fulfilled since the identity provider does not have access at any moment to the decryption keys.

The cloud provider has control only over the re-encryption process, but requires re-encryption keys that are generated by the host organization using its private key. As we have stated before, the re-encryption key, apart from making the re-encryption of ciphertexts possible, also acts as an authorization token, since it is generated by the host organization to give access to service providers to the identity information of its employees. Since we are assuming a honest-but-curious cloud provider, we can assume that it will remove the re-encryption key when asked. Ideally, temporary re-encryption keys that are valid only for a specific period of time would be used, so keys should not be valid if used at any other moment; this way, the re-encryption process could be cryptographically controlled by the host organization. To date, we have not seen any proxy re-encryption scheme that deals with re-encryption keys that are valid for a particular period of time only.

It is important to note that our proposal does not require any change in the SAML framework, as we are respecting its protocols and constructions. Our model requires just a few extension points in the cloud identity provider and service provider, in order to re-encrypt and decrypt the attributes, respectively. We provide explicit SAML constructions that reflect how to realize our system using this framework.

Discussion: Privacy and Confidentiality

Privacy is a vague concept that on many occasions is used as an umbrella term, including other related concepts, such as confidentiality, unlinkability, anonymity, etc. For example, the term *privacy* is often used in the context of the unlinkability property; however, as we have already mentioned, unlinkability is not what we are addressing in this thesis, but rather data confidentiality, as one of our goals is that identity information remains inaccessible to attackers, unauthorized entities, and even the cloud provider itself.

According to [168], *privacy* is defined as *"the right of an entity (normally a person), acting in its own behalf, to determine the degree to which it will interact* *with its environment, including the degree to which the entity is willing to share information about itself with others"*, while *data confidentiality* is defined as *"the property that data is not disclosed to system entities unless they have been authorized to know the data"*.

Taking these definitions into consideration, we argue that our system is *privacy-preserving* because it provides a data confidentiality service, in our case through the use of proxy re-encryption. This service cryptographically protects users' identity information and controls and limits the disclosure of private information with regard to the cloud provider, who acts as an intermediary in the identity management interactions.

6.1.4 Related Work

The problem of privacy in identity management is a widely studied subject. However, the data confidentiality aspects of privacy are seldom tackled. In [169], we proposed an early version of our model, a user-centric IDaaS system based in OpenID and proxy re-encryption. An overview of this system is shown in Figure 6.9. It can be seen how the basic information flow of BlindIdM is already present here. Although conceived as a proof of concept, this is, to the best of our knowledge, the first work that achieves blind processing of identity information; however, trust issues arise as OpenID does not provide proper mechanisms for establishing trust. This proposal is useful for user-centric scenarios where service providers can fully trust end-users without the identity provider being able to assert any claim. One interesting aspect of this work is an economic assessment of the viability of the proposal; in rough numbers, they estimate that the cost for 2000 operations (i.e., encryptions, re-encryptions or decryptions) is 1 USD cent. This assessment is very relevant to our proposal, since the cryptographic procedures are very similar, and therefore, the economic assessment is relevant for our case.

In [170], the authors propose a solution based on deploying active bundles in the cloud provider. An active bundle is a mobile agent, in this case a virtual machine, which contains the identity information of the user and that is protected by cryptographic means. Every time an operation involves the use of identity information, the cloud provider interacts with an active bundle to retrieve this information. However, this approach seems to be impractical because of the large overhead that the use of a large container for data (a VM) introduces. Moreover, the proposal does not detail any procedure to transport these active bundles to the cloud in an efficient manner.

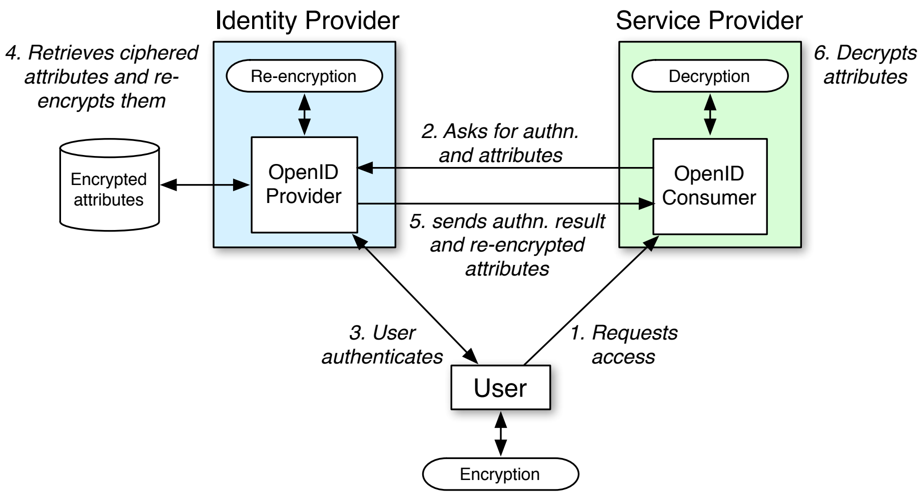


Figure 6.9: IDaaS system based in OpenID and proxy re-encryption

Another proposal, based on the use of sticky policies and trusted computing, is presented in [149]. This paper presents an interesting approach where information, together with a specific policy that should be enforced in order to disclose the data, is obfuscated before leaving the users' domain. In this approach, a trusted authority is in charge of giving the receiver the means to de-obfuscate the information, after verifying that the receiver complies with its associated policy; trusted computing is used to ensure the integrity of both software and hardware environments of the receiver. However, this work focuses on the direct sharing of information, which makes it unusable in an identity management setting, where an identity provider is used as an intermediary and must somehow manage this information.

Much work has been carried out regarding unlinkability of users with respect to the other entities involved in the identity management processes. For example, in [171] the authors present PseudoID, a model for private federated login that achieves unlinkability of users to visited sites. To this end, a blind signature service participates during the generation of an access token that is handed to the identity provider; this access token consists of a pseudonym and a secret value, that are both used to anonymously authenticate the user. Although this work presents an interesting contribution to privacy-enhanced identity providers, it is centered on the unlinkability aspects of the authentication of users. Moreover, this model is not suitable for maintaining users' information in the identity providers, since the providers are unable to correlate users to their pseudonyms.

With regard to the intersection of identity management, privacy and cloud computing, there has also been some research done. In [172], the authors propose

SPICE, an identity management system for cloud environments whose main goal is to preserve users' privacy. SPICE satisfies a set of properties that the authors claim an identity management system in the cloud should fulfill, such as unlinkability and delegatable authentication. In order to accomplish this, SPICE uses a re-randomizable group signature scheme. However, the goal of SPICE is not the same as ours, since we are not tackling unlinkability, but data confidentiality. In [173], a privacy-preserving identity management system for cloud environments is presented; this system is based on zero-knowledge proofs that allow the user to prove the knowledge of a set of attributes without revealing their value. The problem of heterogeneity of attributes representation is also addressed in this work by using ontology mapping techniques. However, the authors do not tackle the privacy issues that are the main concern of our work, since in their setting, identity providers store in clear the values of the attributes of the users.

6.1.5 Summary

We proposed a solution to the problem of privacy, in the sense of data confidentiality, for Identity Management as a Service. BlindIdM is a model for Identity Management as a Service that guarantees user's privacy and control even when data storage and processing is performed by untrusted clouds. Our main contribution is the construction of a privacy-preserving IDaaS system, where the cloud identity provider is able to offer an identity information service without knowing the actual personal information of the users. Our system uses SAML 2.0 as the underlying identity management protocol and proxy re-encryption as a means for achieving blind handling of identity information; this way, the cloud provider transforms encrypted attributes by the host organization into ciphertexts for the service provider, without being able to read their content during this process. In addition, we use standard SAML constructions for conveying this information. We believe that this approach opens up new possibilities regarding privacy in the field of identity management.

6.2 Data Confidentiality in Cloud-Based Hadoop Clusters

In the introduction of this thesis, we described several application use cases that can be seen as instances of the secure data sharing problem that serves as motivation. In particular, the Big Data Analytics in the Cloud was one of these use cases. In this section, we describe a cryptographically-enforced access control system for Hadoop, a widely-used Big Data framework. By using it, the data is in encrypted form and the owner can delegate access rights to the computing cluster for processing. Our proposed solution fits in well with the outsourcing of Big Data processing to the cloud, since information can be stored in encrypted form in external servers in the cloud and processed only if access has been delegated. Experimental results show that the overhead produced by our solution is manageable, which makes our proposal suitable for some applications

6.2.1 Introduction

Data exploitation is quickly becoming one of the main drivers of the economy in this century. Companies, organizations and governments are steadily adopting technologies to leverage the value of large and heterogenous repositories of data, in order to improve their operations and increase their profit. The exploitation of these large quantities of data, along with the technological solutions created to enable it, is what is being called "*Big Data*".

In simple words, Big Data implies the use of vast amounts of data, usually in the form of a collection of datasets, which makes processing and maintenance virtually impossible from the traditional perspective of information management. According to NIST [174], "*Big Data refers to digital data volume, velocity, variety and/or veracity that: (1) enable novel approaches to frontier questions previously inaccessible or impractical using current or conventional methods; (2) and/or exceed the capacity or capability of current or conventional methods and systems*". Thus, the paradigm of Big Data implies the exploitation of large datasets, the volume of which tips over the edge of being tractable by traditional (current) approaches. It is important to understand that as our technology evolves and our capacity to process and store data increases, the concept of Big Data evolves in parallel; thus, basically, what is Big Data today, will be "*small data*" tomorrow.

Apart from the inherent difficulties that the management of massive amounts of data entails, the use of Big Data faces the problems posed by the protection of the information itself, since in a lot of cases the information stored is sensitive or personal data. The presence of high quantities of unprotected and sensitive information is a magnet for malicious agents (insiders and outsiders), which can make a profit by selling or exploiting these large repositories. Companies and organizations that manage these vast amounts of data face the problem of handling extremely diverse kinds of delicate information, which in most cases is stored in clear, since security is delegated to access control enforcement layers, which are implemented on top of the actual data stores. Data disclosures in such contexts could cause great harm to individuals, businesses and governments. Nevertheless, the truth is that although enforcement mechanisms exist that control access to data, some technical staff, such as system administrators, are often able to bypass these traditional access control systems and read data at will, e.g., directly in clear on the file system. Therefore, it is of paramount importance to rely on stronger safeguards such as the use of cryptography. As recently noted by the Cloud Security Alliance in [175]: "*[...] sensitive data must be protected through the use of cryptography and granular access control*".

Within the Big Data community, Apache Hadoop [20] stands out as the most prominent framework for processing big datasets. Apache Hadoop is a framework that enables the storing and processing of large-scale datasets by clusters of machines. The strategy of Hadoop is to divide the workload into parts and spreading them throughout the cluster. However, even though Hadoop was not designed with security in mind, it is widely used by organizations that have strong security requirements regarding data protection.

In this section we propose a delegated access solution for Hadoop, which uses proxy re-encryption to construct a cryptographically-enforced access control system. The goal of our proposal is to enable Hadoop to achieve data protection while preserving its capacity to process massive amounts of information. This way organizations can securely leverage the value of Big Data for their business, in compliance with security and privacy regulations, such as HIPAA and PCI.

In the introduction to this thesis, we described an example of application use case based on the concept of Big Data Analytics in the cloud. This is a very appealing solution for small organizations, which are not in position of acquiring and maintaining the necessary infrastructure to run Big Data frameworks on premise; instead, they can use on-demand high-end clusters in the cloud for analyzing massive amounts of data. Nevertheless, the adoption of the cloud paradigm does not come at no price. There are several risks, such as the ones that stem for a multi-tenant environment. Jobs and data from different tenants are then kept together under the same cluster in the cloud, which could be unsafe when one considers the weak security measures provided by Hadoop. As already noted by several works [176, 175], the use of encryption for protecting data at rest can decrease the risks associated to data disclosures in such scenario. Our proposed solution fits in well with the outsourcing of Big Data processing to the cloud, since information can be stored in encrypted form in external servers in the cloud and processed only if access has been delegated.

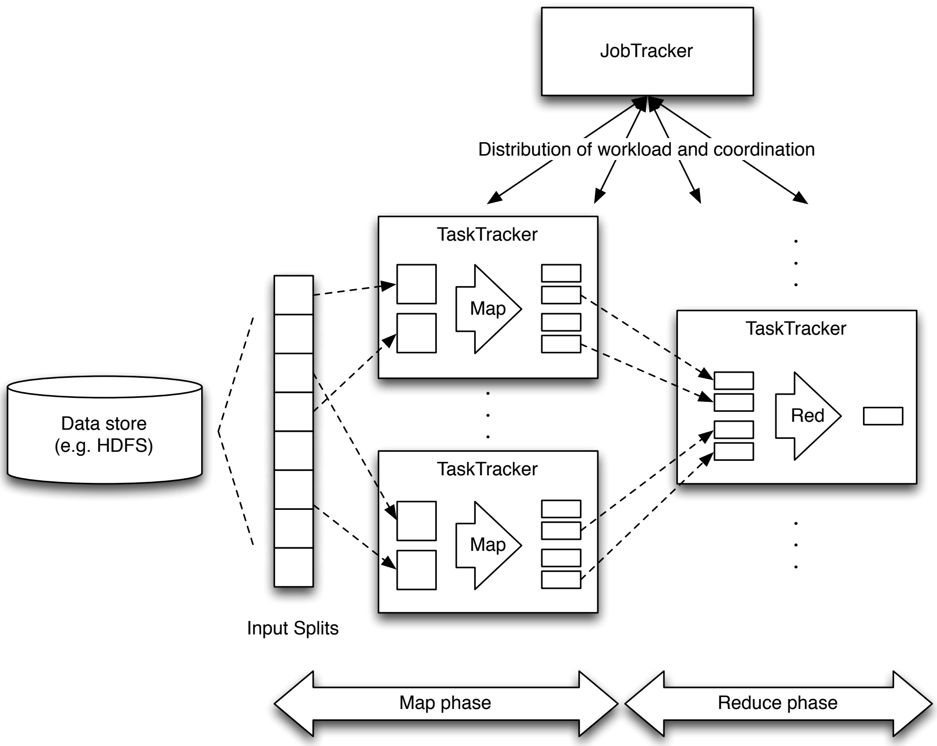


Figure 6.10: Map and Reduce phases in a Hadoop job

6.2.2 The Hadoop Framework

Before introducing our proposal, we will give a brief overview of what is the Hadoop framework and how it works. Hadoop is a framework for processing massive amounts of data in a large-scale, distributed way. In order to do so, Hadoop adopts the MapReduce programming paradigm, which permits to spread the workload across a cluster of machines, usually hundreds or thousands. In Hadoop, all the operations or tasks are executed by nodes in the cluster. There are two kinds of active elements in Hadoop: (i) the JobTracker, which distributes the workload across the cluster by assigning individual tasks to worker nodes and is in charge of its coordination, and (ii) the TaskTrackers, which simply execute the tasks they are assigned.

In the MapReduce paradigm, each portion of the workload must be independent from the others, in order to leverage the potential of massive parallelization. For this reason, a MapReduce job is designed to be executed in two phases, Map and Reduce, as shown in Figure 6.10. In the Map phase, the input data is split and processed in parallel by the cluster. For each data split, the JobTracker assigns a Map task to a TaskTracker that has an available slot (a single TaskTracker can handle several tasks). The output of each Map task is a list of key-value pairs. Roughly speaking, each individual data record in a split is used for producing a key-value pair during this phase. Next, this intermediate data is partitioned and sorted with respect to key, and stored locally. For each partition, the JobTracker assigns a Reduce task to an available TaskTracker. Now the Reduce tasks have to fetch the intermediate data generated in the previous phase; for this reason, this part represents the main communication bottleneck in a MapReduce job. Once intermediate data is retrieved, each Reduce task sorts and merges all his partitions, and the Reduce operation is executed. Finally, the data is combined into one or a few outputs.

The Hadoop framework also defines a special filesystem designed for achieving fault-tolerance and high throughput for large-scale processing, called Hadoop Distributed File System (HDFS); however, Hadoop can be used with other data sources, such as databases or FTP. In HDFS, files are split in large blocks of a determined size (default size is 64MB), which are randomly distributed across the cluster of machines. HDFS also replicates each block into different machines in order to achieve data redundancy. The number of replicas for each block is dictated by the redundancy factor, which is 3 by default. Thus, blocks corresponding to a single file will be distributed into several machines, and each of them will be replicated several times. In case that a node of the cluster is not accessible due to a failure, there are more available copies of the same block.

One of the most prominent characteristics of Hadoop is that it leverages data locality for reducing communication overhead. In order to do so, Hadoop exploits the topology of the cluster by assigning tasks to nodes that are close to the input data, preferably local to it. Another important aspect is the way it provides fault-tolerance. In the event of task failure, Hadoop handles the failure automatically by re-assigning the task to a different node, taking advantage of the multiple copies of each block.

Hadoop clusters usually store huge amounts of data from different sources, owners and degrees of sensitivity. However, because of its nature, Hadoop can be considered as a multi-tenant service and several jobs from different users can be executed at the same time on the cluster. Also, in the case of HDFS, data is distributed evenly through the cluster, so it is possible that one node stores and process data from different tenants at the same time, which can also introduce security threats, such as accessing to intermediate output of other tenants, to concurrent tasks of other jobs or to HDFS blocks on a node through the local filesystem [177]. Some of these problems could be mitigated using a cryptographically-enforced access control approach, such as our proposal.

6.2.3 PRE-based Delegated Access System for Hadoop

In this section, we describe a cryptographically-enforced access control system for Hadoop, based on proxy re-encryption. By using it, the data is stored in encrypted form and the owner can delegate access rights to the computing cluster for processing.

Before proceeding with the description of the system, it is necessary to specify the requirements of the proxy re-encryption scheme. An obvious one is to be very efficient, since Big Data applications are extremely intensive from a computational point of view. A second requirement is that the scheme has to be transitive. This property will allow us to derive re-encryption keys from a single "*master*" re-encryption key generated by the data owner; this is explained in more detail in Section 6.2.3. Examples of schemes that fulfill these characteristics is the one from Weng et al. [54], which unlike most of others proxy re-encryption schemes it is not based in costly bilinear pairing operations, and our scheme NTRUReEncrypt, described in Section 5.1.

The data lifecycle of our proposal is composed of three phases:

1. Production phase: during this phase, data is generated by different data sources, and stored encrypted under the owner's public key for later processing.
2. Delegation phase: in this phase, the data owner produces the necessary master re-encryption key for initiating the delegation process; once this phase concludes, data owner does not need to participate again.
3. Consumption phase: This phase occurs each time a user of the Hadoop cluster submits a job; is in this phase where encrypted data is read by the worker nodes of the cluster. At the beginning of this phase, re-encryption keys for each job are generated.

Production phase

This phase comprises the generation of the data by different sources, and its storage in encrypted form. We assume a scenario where for each dataset, the are multiple data sources and only one dataset owner. In our proposal, we establish that data of each owner is stored encrypted using his public key pk_{DO}. One advantage of using here a public key cryptosystem is that input data can be generated from disparate sources and still be protected from its origin, without requiring to agree on a common secret key for all the sources. This aspect is in accordance to the domains associated to the secure data sharing scenario (see Section 1.1.1).

Let us assume that a data producer (which can be either the dataset owner himself or an external source) stores a file into a cluster with HDFS, and this file is split in N blocks $(b_1, ..., b_N)$. Recall that, when a job is submitted to the cluster, Hadoop first splits input data and then assigns a Map task for each split. In the most common case, Hadoop is implemented using HDFS as the underlying filesystem, so each split will usually match a HDFS block. From now on, we will assume then that data is encrypted on a block-by-block basis since this is the more efficient approach, although our solution could be adapted to other levels of granularity and other filesystems.

For each data block b_i, the data producer will generate a fresh symmetric key r_i that is used for encapsulating the data through a symmetric encryption scheme $\mathsf{Sym.Enc}$, such as AES. Encrypted data is then of the form $\mathsf{Sym.Enc}(r_i, b_i)$. The data key r_i is in turn encapsulated using the encryption function $\mathsf{PRE.Enc}$ of the proxy re-encryption scheme with the public key of the dataset owner, pk_{DO}, obtaining an *encrypted lockbox* $\mathsf{PRE.Enc}(pk_{DO}, r_i)$. Thus, for each block b_i, we obtain an encrypted pair of the form $(\mathsf{PRE.Enc}(pk_{DO}, r_i); \mathsf{Sym.Enc}(r_i, b_i))$, which is the data that is finally stored.

Delegation phase

The goal of this phase is that the dataset owner produces a master re-encryption key mrk_{DO} to allow the delegation of access to the encrypted data. This master re-encryption key is used to derive re-encryption keys in the next phase. The delegation phase is done only once for each computing cluster and involves the interaction of three entities:

- Dataset Owner (DO), whose public key pk_{DO} is used to encrypted generated data for consumption. The data owner also has a secret key sk_{DO}.
- Delegation Manager (DM), which belongs to the security domain of the data owner, and it is trusted by him. This entity can either be local or external to the computing cluster. One of the benefits of being external to the cluster is that the data owner can then control the issuing of re-encryption keys during the consumption phase, since this entity is involved in all the subsequent access delegations. The delegation manager has a pair of public and secret keys, pk_{DM} and sk_{DM}.
- Re-Encryption Key Generation Center (RKGC): This entity is local to the

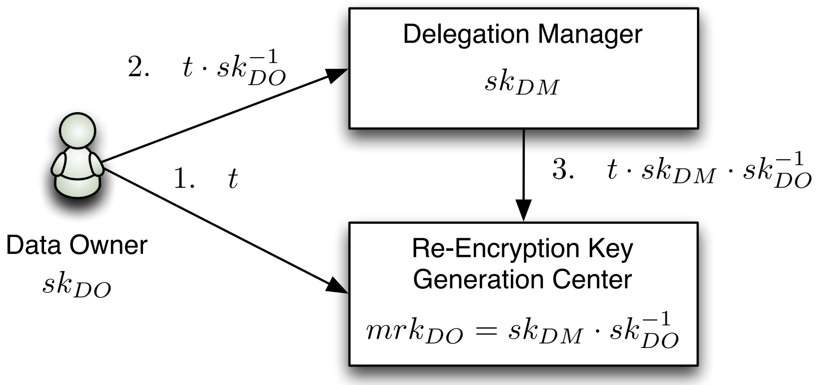

Delegation Manager
sk_{DM}
2. $t \cdot sk_{DO}^{-1}$
1. t
3. $t \cdot sk_{DM} \cdot sk_{DO}^{-1}$
Data Owner
sk_{DO}
Re-Encryption Key Generation Center
$mrk_{DO} = sk_{DM} \cdot sk_{DO}^{-1}$

Figure 6.11: Delegation protocol

cluster and will be responsible for generating all the re-encryption keys needed for access delegation during the consumption phase.

In order to create the master re-encryption key mrk_{DO}, which is actually $mrk_{DO} = rk_{DO \to DM} = sk_{DM} \cdot sk_{DO}^{-1}$, these three entities follow a simple three-party protocol, so no secret keys are shared, as depicted in Figure 6.11. The value t used during this protocol is simply a random value that is used to blind the secret key. At the end of this protocol, the RKGC possesses the master re-encryption key mrk_{DO} that later will be used for generating the rest of re-encryption keys in the consumption phase, making use of the transitive property of the proxy re-encryption scheme.

Consumption phase

This phase is performed each time a user submits a job to the Hadoop cluster. First, the client application submits a job configuration to the JobTracker. This configuration includes the specification of the Map and Reduce functions, the path to the input files and the path of the desired output. Additionally, a pair of public and private keys for the TaskTrackers is initialized in this step; these keys will be used later during the encryption and decryption process. For simplicity, we assume that a common pair of public and private keys, pk_{TT} and sk_{TT}, is shared by all the TaskTrackers; however, each TaskTracker could have a different pair if necessary and the process would be the same.

Next, we need to generate re-encryption keys for each TaskTracker, in our case only one, as we assumed only one pair of public and secret keys. In this step, the Delegation Manager, the Re-Encryption Key Generation Center, the Job-

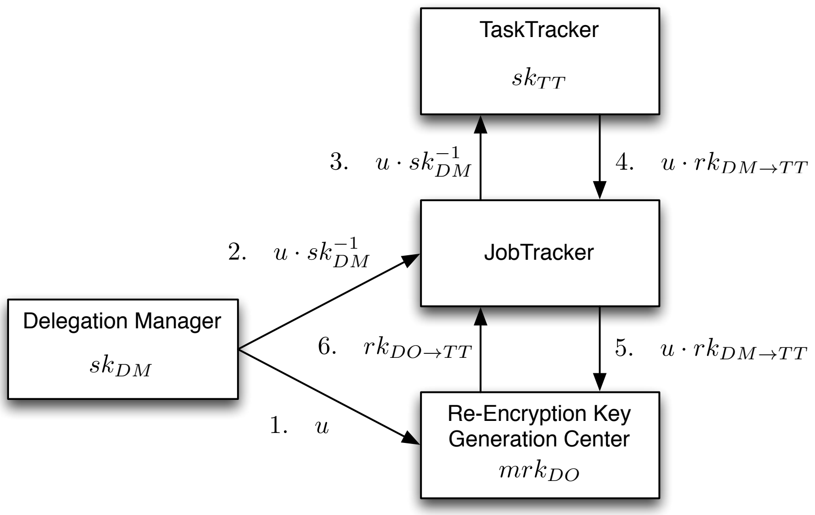

TaskTracker
sk_{TT}
3. $u \cdot sk_{DM}^{-1}$
4. $u \cdot rk_{DM\to TT}$
2. $u \cdot sk_{DM}^{-1}$
JobTracker
Delegation Manager
sk_{DM}
6. $rk_{DO\to TT}$
5. $u \cdot rk_{DM\to TT}$
1. u
Re-Encryption Key Generation Center
mrk_{DO}

Figure 6.12: Re-Encryption Key Generation protocol

Tracker and one of the TaskTrackers with public key pk_{TT} interact in order to generate the re-encryption key $rk_{DO\to TT}$, as depicted in Figure 6.12; in this case, u is the random blinding value. The final output of this interaction is a re-encryption key $rk_{DO\to TT}$ held by the JobTracker, who will be the one performing re-encryptions. This process could be repeated in case that more TaskTrackers' keys are in place.

Now that re-encryption keys have been generated, the JobTracker determines the input set, which is specified by the job configuration, in order to find the number of input splits. Recall that the number of map tasks depends on the number of input splits. Following Hadoop's data locality principle in order to save network bandwidth, the JobTracker will select a set of TaskTrackers that are close to the input data in terms of network proximity, and will send the task requests to this set of TaskTrackers. Before each TaskTracker being able to do any processing, encrypted blocks must be deciphered. In order to do so, each TaskTracker needs to request the re-encryption of the encrypted lockbox for each block to the JobTracker. When the re-encryption is done, the JobTracker sends back the re-encrypted lockbox, which is next deciphered by the TaskTracker for extracting the symmetric key of the content. Once the block is decrypted, data is ready for being extracted by the TaskTracker. The map process now continues in the same way than in regular Hadoop: each TaskTracker invokes the map function for each record in the input split, producing a set of key-value pairs. This intermediate data is sorted and partitioned with respect to the key and stored in local files, one for each reducer. These intermediate files are also encrypted, but

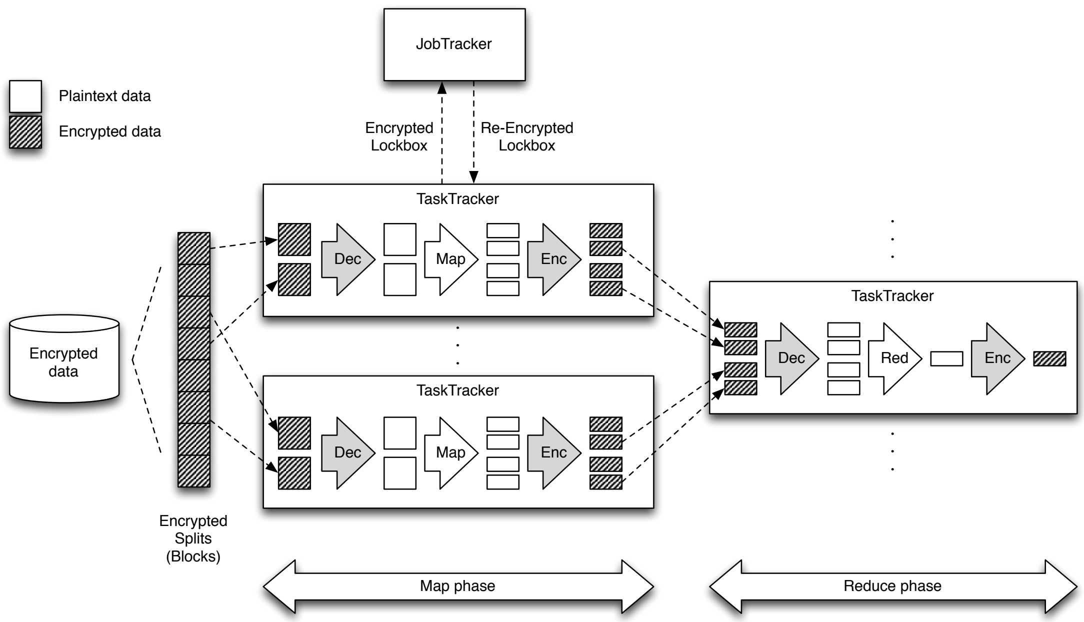


Figure 6.13: Main diagram of the proposed solution

this time with the public key of the Reducer TaskTrackers. Since we assume that all TaskTrackers share the same pair, this key will be pk_{TT}; however, a different set of keys could be used.

When the map task finishes, its TaskTracker notifies the JobTracker about the completion, and once all the TaskTracker complete their map tasks, the JobTracker will select a set of TaskTrackers for performing the Reduce tasks. Each reduce task will first read the intermediate output files remotely and decrypt them using their secret key sk_{TT}. Now that the intermediate files are in clear, they sort and merge the output files and execute the reduce function, which produces an aggregated value for each key; the results are written in one output file per reduce task. The final output can be encrypted using the public key of the client; for simplicity, we can assume that the client in this case is the data owner, so the public key is pk_{DO}. Finally, output files are stored in HDFS.

The full procedure is depicted in Figure 6.13. It can be seen that our solution basically extends the regular Hadoop flow to support the encryption and decryption of input splits; see Figure 6.10 for comparison.

6.2.4 Experimental results

For our experiments, we have executed the main part of the consumption phase of a job, where the processing of the data occurs. The other phases are, from the Hadoop perspective, offline processes, since are not related with Hadoop's flow. Our experiments are executed in a virtualized environment on a rack of IBM BladeCenter HS23 servers connected through 10 gigabit Ethernet, running VMware ESXi 5.1.0. Each of the blade servers is equipped with two quad-core Intel(R) Xeon(R) CPU E5-2680 @ 2.70GHz. We set up a cluster of 17 VMs (the master node, which contains the JobTracker and the NameNode, and 16 slave nodes, each of them holding a TaskTracker and a DataNode). Each of the VMs in this environment is provided with two logical cores and 4 GB of RAM, running a modified version of Hadoop 1.2.1 that implements a prototype of our proposal.

As for the cryptographic details, the proxy re-encryption scheme is implemented using elliptic curve cryptography over a prime field. In particular, we implemented the proxy re-encryption scheme from Weng et al. using the NIST P-256 curve, which provides 128 bits of security and is therefore appropriate for encapsulating 128 bits symmetric keys [74]. With respect to the symmetric encryption algorithm we chose AES-128. We will also make use of the built-in support for AES included in some Intel processors through the AES-NI instruction set.

The experiment consisted on the execution of one of the sample programs included in Hadoop, the WordCount benchmark, a simple application that counts the occurrence of words over a set of files. In the case of our experiment, the job input was a set of 1800 encrypted files of 64 MB each; in total, the input contains 28.8 billions of words and occupies approximately 112.5 GB. The size of each input file is slightly below 64 MB, in order to fit HDFS blocks.

We executed two runs over the same input: the first one using a clean version of Hadoop and the second one using a modified version with a prototype of our proposal. The total running time of the experiment was 1932.09 and 1960.74 seconds, respectively. That is, a difference of 28.74 seconds, which represents a relative overhead of 1.49% for this experiment. We believe that this overhead is acceptable for most applications.

The most critical part of the execution is at the beginning of each Map task, when for each encrypted split the TaskTracker has to ask for the corresponding re-encrypted lockbox to the JobTracker, decrypt it and perform a symmetric decryption of the data block. The duration of this process is more or less constant, as it mostly depends on the size of the encrypted data block. The implication of

Table 6.1: Time cost for the main cryptographic operations

Operation	Time (ms)
Block Encryption (AES-128, 64 MB)	214.62
Block Decryption (AES-128, 64 MB)	116.81
Lockbox Encryption (PRE scheme)	17.84
Lockbox Re-Encryption (PRE scheme)	17.59
Lockbox Decryption (PRE scheme)	11.66

this is that relative overhead will depend drastically on the duration of the map phase. On the one hand, if the map phase is very intensive, then the overhead introduced by our solution will be relative small, in comparison with the processing of the input splits. On the other hand, if the map phase is light, then the overhead will be very significant. In the case of our experiment, where the processing of input splits in each map task takes approximately 34 seconds, the overhead introduced by our solution is very small, as the duration of the cryptographic operations is within the order of milliseconds. Table 6.1 shows the measured time cost associated to the main cryptographic operations of our solution.

6.2.5 Related work

The integration of encryption technologies in Hadoop is a topic that is being explored recently. Park and Lee present in [178] a modification of the Hadoop architecture in order to integrate symmetric encryption in HDFS. They also a perform an experimental evaluation of their solution and claim that the overhead is less than 7%. However, as their solution only considers the use of symmetric encryption, the secret keys used for encrypting have to be shared with the computing nodes. In a similar work, Lin et al [179] show the impact of integrating RSA and pairing-based encryption on HDFS. In their experiments, performance is affected by between 20% and 180%.

On the industry side there has also been some efforts for improving access control in Big Data environments. For instance, Apache Accumulo [180] is a distributed key-value store which provides cell-level access control. This feature is among the key differentiators of Accumulo with respect to other non-relational data stores. The data model of Accumulo includes a *Visibility* field for each cell, which describes the access control policy for said cell. However, data confidentiality depends on the enforcement layer of Accumulo, so the data is still stored in clear.

6.2.6 Summary

In this section we have addressed the problem of how to integrate data confidentiality and massive processing in the Big Data scenario; widely accepted solutions, such as Hadoop, do not offer the proper means to protect data at rest. We propose a cryptographically-enforced access control system for Hadoop, based on proxy re-encryption. In our solution, stored data is always encrypted and encryption keys do not need to be shared between different data sources. Nevertheless, the use of proxy re-encryption allows stored data to be re-encrypted into ciphered data that the cluster nodes can decrypt with their own keys when a job is submitted. In order to permit this, the data owner has to first grant access rights (in the form of decryption capabilities) to the Hadoop cluster. Experimental results show that the overhead produced by the encryption and decryption operations is manageable, so our proposal is suitable for some applications. In particular, the main insight we extract from this experiment is that our proposal will fit well in applications with an intensive map phase, since then the overhead introduced by the use of cryptography will be reduced.

6.3 Escrowed Decryption System

The applications we discussed before were immediate instances of the secure data sharing problem: data is outsourced to cloud providers, which are in charge of storing and sharing it in a secure way, without being able to learn anything about this information. In this section, however, we propose a different, yet related, application: the construction of a public-key encryption with escrowed decryption. The innovative aspect of this proposal is that proxy re-encryption is used to build the escrowed decryption capability, which allows to not resort to escrowing private keys, which is the usual approach; instead, the escrow authority only gets re-encryptions of the requested ciphertexts, which can be decrypted with the authority's private key.

6.3.1 Introduction

Let us consider the scenario posed by the dichotomy between data confidentiality and law enforcement investigations in digital communication networks, such as the Internet. There is a growing concern coming from governments and law enforcement agencies (LEAs) with respect to the alleged "impunity" that is derived from the use of private and confidential communications. This has motivated the proposal of surveillance mechanisms as a means to detect and prevent illegal activities (e.g., child pornography) and national threats (e.g., terrorism). There is no doubt that the goal of protecting citizens from these dangers is noble; the same cannot be said of all the methods to achieve this.

There is an on-going debate nowadays regarding whether mechanisms for breaking confidentiality of communications are a legitimate method to fight the mentioned threats. One of the most immediate concern is the perceived lack of accountability from the government and LEAs; in other words, if such mechanisms were available, there is no hindrance for the government and LEAs to use it as an effective mass surveillance system.

Concerning this problem, Liu, Ryan and Chen proposed in [181] a protocol for *Accountable Escrowed Encryption*, which, on the one hand, allows escrow authorities (a term that subsumes government and LEAs) to decrypt suspicious ciphertexts, but on the other hand, enables to hold them accountable. Their proposal consists on a special encryption scheme, similar to ElGamal, which has a built-in escrowed decryption capability. In order to use this capability to decrypt a ciphertext, escrow authorities have to follow a protocol involving a set of trusted entities called *custodians*; only if every custodian collaborates, the requesting escrow authority is capable of decrypting the ciphertext. Note that this solution does not require *key escrow* (i.e., a “backdoor” in the encryption scheme for extracting users’ private key), since only the decryption capability is escrowed. Another core aspect of this proposal is that the custodians are able to inform citizens of the number of escrow requests they receive by recording such information on a public log, facilitating the society to react through democratic procedures (i.e., demanding “less/more decryption” from escrow authorities); this way, the authors argue, a balance between societal security and individual privacy can be achieved.

In this paper we propose an alternative construction for the Accountable Escrowed Encryption scheme from [181]. Our proposal solves several problems from the original scheme, related to security and efficiency. The proposed construction is similar in essence but makes use of proxy re-encryption as a means for building the escrowed decryption capability; that is, the trusted custodians re-encrypt ciphertexts, in a distributed way, upon request from the escrow authority; the re-encrypted ciphertext can be opened them by the escrow authority.

It can be seen that this setting is reminiscent of the secure data sharing scenario, although with some differences. In this case, there is no encrypted data outsourced to cloud storage providers, but encrypted data circulating on communication networks, that is susceptible of being intercepted by the government and LEAs. Consequently, there are no cloud storage providers, but trusted custodians, which are requested by government and LEAs to perform escrow decryption of ciphertexts. In other words, government and LEAs are data consumers that request delegated access to encrypted information, as in the secure data sharing scenario.

6.3.2 PKE with PRE-based Escrowed Decryption

In this section we describe our proposal. Formally, it is a new type of cryptosystem that consists of a public-key encryption scheme with an added escrowed decryption capability, which we construct through proxy re-encryption.

Recall that our proposal is intended to solve some problems of the original scheme from Liu, Ryan and Chen [181]. One of these problems is that a collusion of custodians can potentially decrypt any message if they have access to a full ciphertext. That is, one have to make the assumption that the custodians are trusted for not doing this. In our proposal, the custodians are never able to see the underlying message, even if they have access to the whole ciphertext.

Another problem of the original scheme is the efficiency of the solution. The protocol for escrow decryption in the original proposal is composed of 2 synchronous rounds, and each round involves all custodians; that is, the escrow authority has to first engage all custodians in a first round of interactions (ideally in parallel), and only when all of them have responded, it can continue with a second round of interactions. The reason behind this characteristic is that the escrow authority has to make some intermediate computations that are required for the second round, and it needs all the responses from the first round to do this. In our proposal, we reduce the escrow decryption protocol to a single round.

Our idea is to base the escrowed decryption protocol on a "shared" re-encryption by the custodians, in a way similar to the decryption procedure of a (n, n)-threshold encryption scheme [143]; that is, our solution ensures that the escrow authority is able to decrypt ciphertexts intended for suspicious users as long as he engage the collaboration of all the escrow custodians. Inspired by this idea, the following is the generic syntax of this new cryptosystem:

- $\mathsf{Setup}(\lambda) \to params$. On input the security parameter λ, the setup algorithm outputs the set of global parameters $params$, which includes the public and private key of the escrow authority (pk_{EA}, sk_{EA}).
- $\mathsf{KeyGen}(pk_{EA}) \to (pk, sk, \{\kappa_i\}_{i=1}^{n})$. On input the escrow authority's public key pk_{EA}, the key generation algorithm outputs the public and private key of user U, (pk, sk), and a set of *escrow shares*), $\{\kappa_i\}_{i=1}^{n}$.

- $\mathsf{Enc}(pk, m) \rightarrow CT$. On input a public key pk and a message m, the encryption algorithm outputs ciphertext CT.
- $\mathsf{Dec}(sk, CT) \rightarrow m$. On input the secret key sk and a ciphertext CT, the decryption algorithm outputs the original message m.
- $\mathsf{ShareReEnc}(\kappa_i, CT) \rightarrow \rho_i$. On input a re-encryption share κ_i and a ciphertext CT, this algorithm outputs the *re-encryption share* (or *escrow share*) ρ_i.
- $\mathsf{Comb}(sk_{EA}, CT, \{\rho_i\}_{i=1}^{n}) \rightarrow m$. On input the escrow authority's secret key sk_{EA}, a ciphertext CT, and a complete set of re-encryption shares $\{\rho_i\}_{i=1}^{n}$, the combination algorithm outputs the original message m.

Our proposal

Since our proposal in actually an enhancement of the one from Liu, Ryan and Chen [181], we maintain the same setting, with the same actors. Apart from the actors mentioned before, namely users, escrow authority and custodians, their proposal also contained a Certification Authority (CA) that participates in the key generation process, ensuring that the escrow capabilities are properly preserved.

We base our solution on the PRE scheme from Ateniese et al. [11], with two main modifications:

- Re-encryption is split among several custodians. The re-encryption key is split into n shares, so $rk = \prod_{i=1}^{n} rk_i$. Then, for the decryption of re-encrypted ciphertexts, we make use of the multiplicative homomorphic properties of the scheme, as $\mathsf{ReEnc}(rk, CT) = \prod_{i=1}^{n} \mathsf{ReEnc}(rk_i, CT)$.
- The original scheme from Ateniese et al. allow to encrypt ciphertexts that are not re-encryptable (called "first-level ciphertexts"), which can be used to bypass our re-encryption-based escrow. We get rid of this functionality by modifying the key generation in such a way that users' public key cannot be used to create first-level ciphertexts.

Our scheme is defined as follows. Note that although in the generic syntax we described functions, our solution requires some of them to be implemented as two-party protocols.

Setup. Let $\mathbb{G}$ and $\mathbb{G}_T$ be cyclic groups of order q, and $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be a bilinear pairing. Let g be a generator of $\mathbb{G}$, and $Z = e(g, g)$. The public key of the escrow authority is $pk_{EA} = g^a$, and the corresponding secret key $sk_{EA} = a$. This public key is certified by the CA in the usual way.

Key Generation. This subprotocol can be seen as a secure two-party protocol between the user and the CA that realizes the functionality of the KeyGen algorithm. The result of this subprotocol is that the user gets his public and private keys certified, without the CA learning the private key, but with the assurance that valid escrow shares are generated.

The user U first selects random secrets $u, \beta \in \mathbb{Z}_q$. He chooses a set of custodians who he trust. We denote as $\mathfrak{C}$ to the set of indices of such custodians; for simplicity, we will assume that $\mathfrak{C} = \{1, ..., n\}$.

For each custodian C_i, where $2 \leq i \leq n$, user U chooses a random $\widetilde{\kappa}_i \in \mathbb{G}$. For custodian C_1, he computes $\widetilde{\kappa}_1 = (pk_{EA})^{\beta/u} \cdot \left(\prod_{i=2}^{n} \widetilde{\kappa}_i\right)^{-1}$. Consequently, $\prod_{i=1}^{n} \widetilde{\kappa}_i = g^{a\beta/u}$. He sends $(g^u, \mathfrak{C}, \{\widetilde{\kappa}_i\}_{i=1}^n, g^\beta)$ to CA for certification.

CA verifies that the set of partial escrow shares is valid by checking that the following equation holds:

$$e(\prod_{i=1}^{n} \widetilde{\kappa}_i, g^u) = e(pk_{EA}, g^\beta) \tag{6.1}$$

CA chooses $s, \gamma \in \mathbb{Z}_q$ and compute user U's public key $pk = ((g^u)^s, e(g^\beta, g)^{s\gamma}) = (g^{su}, Z^{sv})$, which implicitly defines $v = \beta \cdot \gamma$. This public key is then signed and certified by the CA. Now, the CA concludes the generation of the set of escrow shares, by computing $\kappa_i = (\widetilde{\kappa}_i)^\gamma$ for each custodian. Note that $\prod_{i=1}^{n} \kappa_i = (\prod_{i=1}^{n} \widetilde{\kappa}_i)^\gamma = (g^{a\beta/u})^\gamma = g^{av/u}$. The CA sends each escrow share to a different custodian in $\mathfrak{C}$ through a secure channel. Finally, CA returns (pk, g^γ) to user U, who sets $sk = (g^\gamma)^{\beta/u} = g^{v/u}$. Figure 6.14 shows the interactions of this subprotocol.

Encryption. Anyone can produce a ciphertext under user U's public key $pk = (g^{su}, Z^{sv})$ as follows: choose a random $r \in \mathbb{Z}_q$ and output the ciphertext

$$CT = ((g^{su})^r, (Z^{sv})^r \cdot m) = (g^{sur}, Z^{svr} \cdot m)$$

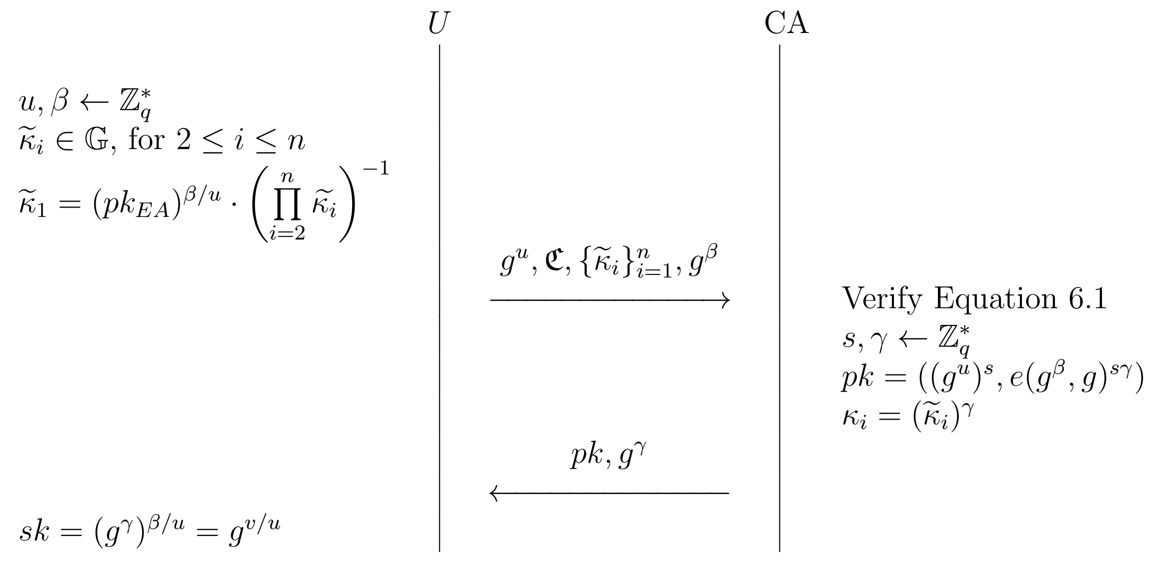

U
CA
$u, \beta \leftarrow \mathbb{Z}_q^*$
$\widetilde{\kappa}_i \in \mathbb{G}$, for $2 \leq i \leq n$
$\widetilde{\kappa}_1 = (pk_{EA})^{\beta/u} \cdot \left(\prod_{i=2}^{n} \widetilde{\kappa}_i\right)^{-1}$
$g^u, \mathfrak{C}, \{\widetilde{\kappa}_i\}_{i=1}^n, g^\beta$
Verify Equation 6.1
$s, \gamma \leftarrow \mathbb{Z}_q^*$
$pk = ((g^u)^s, e(g^\beta, g)^{s\gamma})$
$\kappa_i = (\widetilde{\kappa}_i)^\gamma$
pk, g^γ
$sk = (g^\gamma)^{\beta/u} = g^{v/u}$

Figure 6.14: Key Generation subprotocol

Decryption. User U can decrypt a ciphertext $CT = (CT_1, CT_2) = (g^{sur}, Z^{svr} \cdot m)$ as follows:

$$m = \frac{CT_2}{e(CT_1, sk)} = \frac{Z^{svr} \cdot m}{e(g^{sur}, g^{v/u})}$$

Escrow decryption. This subprotocol is basically the combination of the $\mathsf{ShareReEnc}$ and Comb algorithms, as seen in Figure 6.15. When the escrow authority wants to decrypt a ciphertext $CT = (CT_1, CT_2) = (g^{sur}, Z^{svr} \cdot m)$ that is encrypted under user U's public key, they give $CT_1 = g^{sur}$ to each custodian in $\mathfrak{C}$, and obtain the re-encryption share $\rho_i = \mathsf{ShareReEnc}(\kappa_i, CT_1) = e(\kappa_i, g^{sur})$ in response. Note that although we defined the syntax of $\mathsf{ShareReEnc}$ such that it takes the whole ciphertext as input, in our proposal only the first component is necessary.

After getting the re-encryption shares $\{\rho_i\}_{i=1}^n$ from all custodians, the escrow authority executes the combination algorithm Comb to decrypt the ciphertext. In our proposal, this algorithm is defined as follows:

$$\mathsf{Comb}(sk_{EA}, CT, \{\rho_i\}_{i=1}^n) = \frac{CT_2}{(\prod_{i=1}^{n} \rho_i)^{1/sk_{EA}}}$$

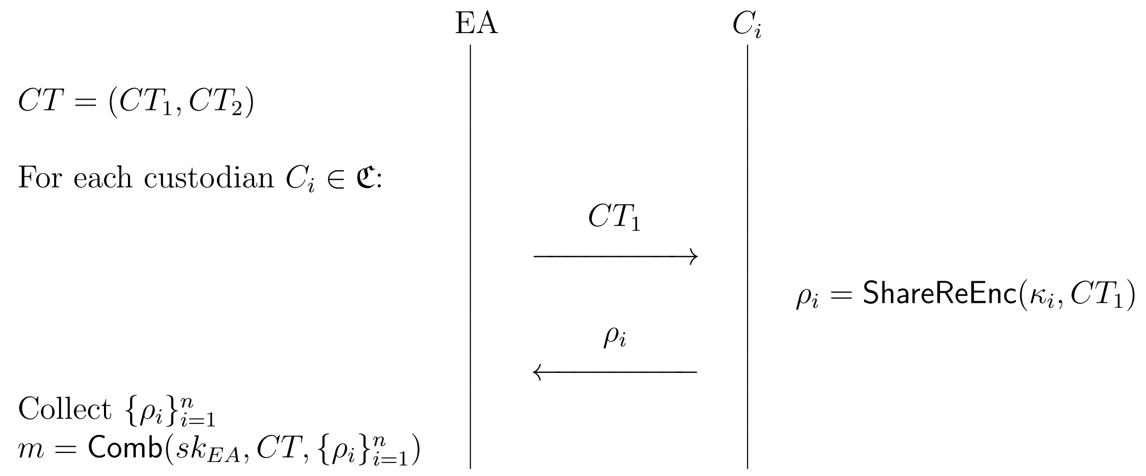

EA
C_i
$CT = (CT_1, CT_2)$
For each custodian $C_i \in \mathfrak{C}$:
CT_1
$\rho_i = \mathsf{ShareReEnc}(\kappa_i, CT_1)$
ρ_i
Collect $\{\rho_i\}_{i=1}^n$
$m = \mathsf{Comb}(sk_{EA}, CT, \{\rho_i\}_{i=1}^n)$

Figure 6.15: Escrow Decryption subprotocol

6.3.3 Analysis of our solution

In this section we evaluate the correctness and security of our solution. Additionally, we analyze its performance, by comparing it with the scheme from Liu, Ryan and Chen [181].

Correctness

Let $CT = (CT_1, CT_2) = (g^{sur}, Z^{svr} \cdot m)$ be a ciphertext encrypted under user U's public key. The regular decryption procedure works since:

$$\mathsf{Dec}(sk, CT) = \frac{CT_2}{e(CT_1, sk)} = \frac{Z^{svr} \cdot m}{e(g^{sur}, g^{v/u})} = \frac{Z^{svr} \cdot m}{Z^{svr}} = m$$

Assuming the Key Generation subprotocol is correct (which is ensured by the CA), then the escrow decryption procedure is also correct. Recall that when the escrow authority wants to decrypt a ciphertext CT, he gives CT_1 to each custodian in $\mathfrak{C}$, and obtain the re-encryption share $\rho_i = \mathsf{ShareReEnc}(\kappa_i, CT_1) = e(\kappa_i, g^{sur})$ in response. In the next step, the escrow authority combines all the escrow shares:

$$\prod_{i=1}^{n} \rho_i = \prod_{i=1}^{n} e(\kappa_i, g^{sur})$$

Thus, by the bilinear property of the pairing:

$$\prod_{i=1}^{n} e(\kappa_i, g^{sur}) = e(\prod_{i=1}^{n} \kappa_i, g^{sur})$$

Next, since the Key Generation subprotocol ensures that $\prod_{i=1}^{n} \kappa_i = g^{av/u}$, then:

$$\prod_{i=1}^{n} \rho_i = e(g^{av/u}, g^{sur}) = Z^{savr}$$

Finally, in the combination algorithm, the escrow authority uses this value to decrypt the ciphertext:

$$\mathsf{Comb}(sk_{EA}, CT, \{\rho_i\}_{i=1}^{n}) = \frac{CT_2}{(\prod_{i=1}^{n} \rho_i)^{1/sk_{EA}}} = \frac{Z^{svr} \cdot m}{(Z^{savr})^{1/a}} = m$$

Security

In order to evaluate to security of our proposal, we prove that it complies with the IND-CPA notion, as is usual in PKE schemes. We are not targeting here the corresponding PRE notion, since this is not a PRE scheme: it is only possible to re-encrypt to the escrow authority, not to any other regular user. Note, however, that in our case, the key generation protocol provides additional information that should be known by the adversary. In addition, since we are targeting a CPA attack model, there is no need to describe any oracle.

Let us assume that there is an adversary $\mathcal{B}$ that wins the $\mathsf{IND\text{-}CPA}$ game with non-negligible advantage ε. Then, we can use $\mathcal{B}$ to construct an algorithm $\mathcal{A}$ that solves the 1-wDBDHI problem (see Definition 2.12) with the same advantage.

$\mathcal{A}$ receives as input a tuple (g, g^u, g^w, Z^d), and his goal is to decide whether $d = w/u$; he uses this tuple to simulate the environment for the adversary $\mathcal{B}$.

First, $\mathcal{A}$ publishes the global parameters $(q, e, \mathbb{G}, \mathbb{G}_T, g)$, as usual; next, samples random $a \in \mathbb{Z}_q$ and sets the escrow authority's public and private keys $(pk_{EA}, sk_{EA}) = (g^a, a)$.

The next step is simulating the key generation subprotocol; in this case, $\mathcal{A}$ releases to $\mathcal{B}$ all the information that is produced in the subprotocol, except for the user's secret values and the escrow shares, since this would allow to corrupt the target user and would make $\mathcal{B}$ win the game trivially. $\mathcal{A}$ samples random $s, \beta, \gamma \in \mathbb{Z}_q$, and sets the public key of the target as $pk^* = ((g^u)^s, Z^{s\beta\gamma}) = (g^{us}, Z^{sv})$. Now $\mathcal{A}$ returns $\mathcal{B}$ the tuple $(g^u, g^\beta, s, \gamma)$; the first two elements are the user's input to the subprotocol, while the last two are the CA's input.

Table 6.2: Computational costs of selected PRE schemes

Algorithm	Liu, Ryan and Chen [181]	Our proposal
Key Generation (User)	$3t_e$	$4t_e$
Key Generation (CA)	$3t_e$	$2t_p + 2t_e + t_{e_T}$
Encryption	$3t_e$	$t_e + t_{e_T}$
Decryption	$2t_e$	t_p
Escrow Decryption (Custodian)	Round 1: t_e Round 2: t_e	t_p
Escrow Decryption (Escrow Agent)	–	t_{e_T}

The challenge ciphertext is constructed as:

$$CT^* = ((g^w)^s, m_\delta \cdot (Z^d)^{s\beta\gamma}) = (g^{ws}, m_\delta \cdot (Z^d)^{sv})$$

$\mathcal{A}$ runs adversary $\mathcal{B}$ to obtain the guess δ' and decides that $d = w/u$ when $\delta = \delta'$. Note that when $d = w/u$, the challenge ciphertext is a valid encryption of m_δ under pk^*, where the random exponent r is defined implicitly as $r = w/u$.

$$CT^* = ((g^{us})^{w/u}, m_\delta \cdot (Z^{sv})^{w/u})$$

Therefore, in this case $\mathcal{B}$ guesses δ correctly with probability $\frac{1}{2} + \varepsilon$ and $\mathcal{A}$ solves the 1-wDBDHI problem with the same probability. On the contrary, when d is random, m_δ is information-theoretically hidden, so the probability of $\mathcal{B}$ guessing δ correctly is $\frac{1}{2}$. Thus, the overall success probability of $\mathcal{A}$ is $\frac{1}{2} + \frac{\varepsilon}{2}$.

Performance

Similarly to the study done in Section 3.3.3, we analyze computational costs in terms of the main operations performed, which in this case are the exponentiation and the pairing (denoted by t_e and t_p, respectively). In the case of pairing groups, the computational costs of exponentiations of group elements are different. For this reason, when necessary we will make the distinction between exponentiations in $\mathbb{G}$ (denoted by t_e) and in $\mathbb{G}_T$ (denoted by t_{e_T}). We ignore other minor costs, such as multiplications and inversions. Note, however, that both in our solution and in [181], the number of multiplications depends mainly on the number of custodians; we can safely assume that a realistic implementation will involve only a reduced number of custodians.

The results of our performance comparison are shown in Table 6.2. The price of solving the problems in [181] implies increased costs in all the operations, although there are some remarks on this matter. Firstly, communication costs are not addressed here, since they depend on the specific communication network in use, but our costs in this dimension are half than in [181] because we get rid of the second round. Secondly, the Key Generation in [181] do not consider any specific mechanism for ensuring that the users' input is valid; since this is not defined in their solution, we do not count anything for them on this respect.

6.3.4 Summary

In this section we have described a public-key encryption scheme that allows an escrow authority to decrypt messages, without escrowing users' secret keys, but only the decryption capability. We construct the escrow capability by means of proxy re-encryption. Our solution, inspired by the Accountable Escrow Encryption scheme from Liu, Ryan and Chen [181], solves some of the problems that arise in this proposal, such as removing the possibility of a collusion of custodians and halving communication costs.

As future work, it is necessary to further formalize the security notions of this new cryptosystem. In particular, it is interesting to define an equivalent of CCA-security on this context, and to propose a solution that satisfies this notion. Another line of research is to solve the trust problem. The proposed solution involves a high level of trust in the CA, since it must be trusted to not combine all the escrow shares; otherwise, the CA could render useless the custodians, and the government could collude together with the CA to decrypt messages bypassing the accountability provided by the custodians. In order to solve this, a possible extension is that the custodians have also a public-private key pair, which is necessary during re-encryption.

Chapter 7

Conclusions

This final chapter constitutes a brief recapitulation of the work done on this thesis. We first summarize the secure data sharing problem as the motivational scope and discuss the readiness of proxy re-encryption as part of the solution. We then present a description of each of our contributions, which we classify from the viewpoint of the goals we established in the introduction of this thesis. To conclude, we discuss future lines of work, motivated by open problems that require further research.

7.1 Proxy Re-Encryption and the Secure Data Sharing Scenario

As described in the introduction, one of the postulates of this thesis is that proxy re-encryption is a prime candidate for the construction of viable solutions to the secure data sharing problem. In this scenario, data owners entrust their data to cloud providers, who are then responsible for storing and managing it. This implies the existence of a strong trust relationship between the data owners and the cloud providers, since the former trust that their information will be managed properly and that the provider will respect the confidentiality; however, current cloud providers do not implement real mechanisms to prevent themselves from betraying this trust. This concern led us to advocate for the use of proxy re-encryption as a viable solution to this problem, so that cloud providers are still able to offer a storage and sharing service, without having access to the actual data.

7.1.1 Incentives

At this point is useful to think about what kinds of incentives may motivate a cloud provider to offer a storage service without being able to read the outsourced information. From a strictly economic point of view, it may not make sense to provide these services for free, since they will probably incur more expenses as a result of implementing additional security mechanisms. Furthermore, they will lose control over the user's data, which is currently a valuable asset. Still, there are some incentives that could encourage cloud providers to offer such a *blind* service.

Compliance with Data Privacy Laws and Regulations Cloud providers may be seen in the eyes of the law as stores and processors of sensitive information. In consequence, they are obliged to comply with specific laws and regulations regarding data protection, such as the EU Data Protection Directive, in case of *Personal Identifiable Information*(*PII*), or the Health Insurance Portability and Accountability Act (HIPPA) [8], in the case of healthcare information. Some of these regulations demand that sensitive information must be protected using appropriate encryption techniques.

Therefore, a solution to secure data sharing based on proxy re-encryption, which achieves data confidentiality through encryption mechanisms and permits cloud providers to still provide storage and sharing services without having the chance to access the data, could be very useful to help them comply with these kinds of regulations. We argue that, given the proper cryptographic safeguards, encrypted data is no longer private, and could even be freely distributed without compromising users' privacy.

Minimization of Liability Currently there is a lot of discussion, especially from the cloud industry and lawmakers, with regard to liability in cloud computing due to its nature of outsourced service provision. Although cloud providers currently try to reduce their liability through specific clauses in SLAs, legal responsibility for the data in the cloud also lies on the side of the cloud provider. There are a lot of examples from blog sites, Internet forums, or file hosting services (such as the Megaupload case in 2012 [182]), where the owners of these services are indicted for hosting illegal or defamatory material, even though they have not generated said content.

In contrast, given that in our proposed approach to the problem, outsourced data is encrypted prior to arriving the cloud and the cloud provider does not hold the decryption keys, liability is drastically reduced, as they are unable to read user's data. Take a shipping service as an analogy: they will not be liable for any illegal or dangerous item delivered through their service, since they cannot open the packages and inspect their content (or at least, every delivered package). As a consequence of this, users should be the ones designated as liable and subject to the enforcement of key disclosure laws.

Data Confidentiality as an Added Value An interesting incentive for cloud providers could be the possibility of offering secure data processing and confidentiality as an added value. Setting aside legal and regulatory aspects, this model could help a cloud provider to offer a competitive advantage over the rest. We expect the topics addressed in this thesis to contribute to the blossoming of a business model based on respect for users' privacy and data confidentiality. Currently, there are some cloud services, such as PrivateSky [183] or CipherCloud [184] that have built their business model on data confidentiality as an added value.

7.1.2 Towards application to the Real World

It is also worth finding out whether there are industrial initiatives towards the application of PRE in real commercial solutions. Although we are not presently aware of widely-used real-world applications of proxy re-encryption, there are some indications that show that the IT industry is becoming increasingly interested in it, particularly with regard to its application to the secure data sharing scenario. Nishimaki and Xagawa recently noted in [70] that Toshiba has released, on the Japanese market, a cloud storage service that use proxy re-encryption to ensure data confidentiality, called "Digital Kashikinko" [185] (which means "digital safety box"). To the best of our knowledge, this is the first commercial application of proxy re-encryption. Although the specific details are not known (as for most commercial applications), the associated publication describes a typical PRE-based architecture.

An interesting way of probing the industry's interest in the topic is to look for patents that use proxy re-encryption. As conjectured in this thesis, the application of PRE to the cloud seems to be recurring, given its natural potential in this scenario. We have found patents from Toshiba [186], Huawei [187, 188], Nokia [189], and Gemalto [190], among others, which present approaches that are, essentially, instances of the secure data sharing scenario. With regard to other use cases, Apple has patented a solution for DRM protection based in PRE [191].

The authors of one of the first PRE schemes [11] also presented a patent [192] that essentially covers their original proposal. We foresee a growing number of patents in these areas, which would ultimately lead to new products and services based on this cryptographic primitive.

7.2 Contributions

Throughout this thesis we have worked on several topics, always with proxy re-encryption as the backbone. In this section we provide a summary of our contributions, classified from the point of view of the goals presented in the introduction.

7.2.1 Better understanding

As proxy re-encryption is the central topic of this thesis, it became necessary to first perform an extensive review of the scientific literature around this cryptosystem. One of the contributions of this thesis is that we survey and analyze the current state of research on proxy re-encryption, for both constructions and applications. The most prominent proxy re-encryption schemes are studied in light of relevant properties and security models. We have additionally provided a comparative analysis of the performance of selected schemes, both from the theoretical and experimental points of view. With regards to applications of PRE, we have performed an extensive review of the available literature following a bibliometric approach.

A core contribution of this thesis that is related to this goal is the identification of a parametric family of attack models for PRE, which not only considers the availability of the decryption oracle (as in PKE), but also the re-encryption oracle. These attack models for PRE allow us to define a collection of security notions, whose relations we analyze, and that is useful for achieving a better understanding of the definitions of security upon which proxy re-encryption schemes are designed. Of particular interest is the study of the separations between the obtained notions of security. We have focused on those that arise by exploiting the private re-encryption keys property – or more accurately – the failure to fulfill this property. The consequence of these separations is that schemes that leak re-encryption keys through queries to the re-encryption oracle cannot achieve strong security notions. These results strengthen the idea that meaningful chosen-ciphertext attacks for PRE should consider both oracles (i.e., decryption and re-encryption). We illustrated these findings by showing an attack on a recent PRE scheme from PKC 2014 [48] that is said to be "CCA1-secure".

7.2.2 More security

A substantial part of this thesis has been devoted to the security of proxy re-encryption. A first contribution on this matter is the definition of a parametric family of attack models, mentioned before. This contribution can be seen as a design aid, which can be used by cryptographers to devise PRE schemes that achieve intermediate security notions or to reason about the actual security of schemes. The second contribution in this area is the definition of a generic transformation that bootstraps weakly-secure PRE schemes into stronger ones. This construction extends the Fujisaki-Okamoto transformation to the PRE context, which is far from being obvious. The most immediate challenge we had to tackle here is that the possibility of re-encrypting conflicts with the validity checks that are necessary during decryption; we identified twelve PRE schemes that are flawed as a consequence of this problem. We proved that, under some assumptions, which include the satisfaction of a new property of PRE called "*perfect key-switching*", the Fujisaki-Okamoto transformation can be used to generically bootstrap a weak notion of security ($\mathsf{IND\text{-}CCA}_{0,1}$) into a much stronger notion ($\mathsf{IND\text{-}CCA}_{2,1}$), in the random oracle model.

Other generic transformations for public-key encryption are also studied for its application in proxy re-encryption. We discussed how the REACT and GEM transformations [18, 29] can be modified to support re-encryptions. Since the perfect key-switching property is no longer required, these proposals are potentially applicable to a wider class of schemes, which makes them very attractive. In addition, they are more efficient than the Fujisaki-Okamoto transformation, since they do not require to reconstruct the ciphertext during decryption.

7.2.3 Better performance

Another contribution of this thesis is NTRUReEncrypt, a highly-efficient proxy re-encryption scheme based on the NTRU cryptosystem. This scheme is bidirectional and multihop, but not collusion-resistant. The key strength of this scheme is its performance: experimental results show that this scheme outperforms previous proposals by an order of magnitude, and there is room for even more improvement, for instance using parallelization techniques, as shown in [124]. We believe that the level of efficiency shown by NTRUReEncrypt opens up new practical applications of proxy re-encryption in constrained environments.

7.2.4 Bringing theory and practice together

From the beginning of this research effort, we wanted to make contributions to both theory and practice. This conviction is indeed reflected in the title and organization of this dissertation.

An example of this research philosophy is found in our study of the efficiency of selected PRE schemes. This analysis is performed, not only from a theoretical perspective (which is the usual practice in current literature), but also empirically, using our own implementation. We believe this aspect is especially important, since it has a direct impact on the incurred costs of any solution based on proxy re-encryption.

Another example is found in the proposed applications, where we have strived to not only define generic architectures, but also integrate our solutions with real-world technologies. For instance, in the case of our model for privacy-preserving identity management in the cloud, we tightly integrated it with SAML 2.0, a standard protocol for identity management, while in the the case of the Big Data application, we instantiated it with the Apache Hadoop system, a widely used framework for Big Data Analytics.

7.3 Future work

The last part of this chapter is devoted to several research questions that we have identified in the course of this thesis and which remain open for further study. Although some of them have been discussed in their corresponding chapters, we gather them together here for the reader's convenience.

From the systematic analysis of the literature we performed in Chapter 3, we detected a very interesting trend with regard to PRE schemes. The experimental results presented in Section 3.3.3 show that lattice-based PRE schemes seem to be more efficient in comparison to previous constructions (our NTRU-based proposal is an example). However, at the same time, the highest notion achieved so far by lattice-based PRE is $\mathsf{IND\text{-}CCA}_{1,0}$ through the scheme from Kirshanova [48]. This clearly represents a weakness of the current state of research regarding these kinds of schemes. Therefore, further work is required in order to devise more secure lattice-based schemes.

In Chapter 4 we defined a family of attack models and security notions for proxy re-encryption, and studied some of the relations that arise between these notions. However, we do not rule out the possibility of other possible separations and implications; it is therefore an open question to further study additional relations among security notions.

With regard to the PRE schemes based on NTRU proposed in Section 5.1, there are several areas with potential for improvement. The most pressing are achieving CCA-security and the definition of a unidirectional and collision-resistant scheme. The former issue was the initial motivation for the generic transformations that we proposed in Section 5.2, although, in the end, it could not benefit from them; the latter issue is sketched in Section 5.1.6 and looks promising, although it requires further work. Another subject is improving the parameters of NTRUReEncrypt, since this could decrease the probability of decryption failures after multiple re-encryptions. Additionally, it would be interesting to come up with better bounds for the provably-secure version and an analysis of the selection of parameters based on the best known lattice attacks.

As for the generic transformations we proposed in Section 5.2, future lines of research include working towards concrete estimations of the obtained security level of the extended Fujisaki-Okamoto transformation; this could allow sets of parameters to be fixed and to perform meaningful comparisons with other schemes to be made. A more ambitious problem is to propose transformations that are not defined in the random oracle model.

Finally, there are also potential lines of improvement for the applications proposed in Chapter 6. Most of them are of a technical nature: for instance, the BlindIdM model can be refined in order to adapt it to the System for Cross-domain Identity Management (SCIM) specification [193], an open standard from the IETF that is designed to simplify user management in cloud-based services and applications by facilitating the exchange of identity information; in the case of the Big Data application, it would be interesting to improve the integration of our solution with the Hadoop architecture, in particular, with the HDFS layer, which has a lot of potential. With respect to the escrowed decryption system, an immediate improvement would be to devise a scheme that achieves CCA-security and that reduces the trust assumption on the Certification Authority.

Apéndice A

Resumen

La externalización de la gestión de la información es una práctica cada vez más común, siendo la computación en la nube (en inglés, *cloud computing*) el paradigma más representativo. Sin embargo, este enfoque genera también preocupación con respecto a la seguridad y privacidad debido a la inherente pérdida del control sobre los datos. Las soluciones tradiciones, principalmente basadas en la aplicación de políticas y estrategias de control de acceso, solo reducen el problema a una cuestión de confianza, que puede romperse fácilmente por los proveedores de servicio, tanto de forma accidental como intencionada. Por lo tanto, proteger la información externalizada, y al mismo tiempo, reducir la confianza que es necesario establecer con los proveedores de servicio, se convierte en un objetivo inmediato. Las soluciones basadas en criptografía son un mecanismo crucial de cara a este fin.

Esta tesis está dedicada al estudio de un criptosistema llamado *recifrado delegado* (en inglés, *proxy re-encryption*), que constituye una solución práctica a este problema, tanto desde el punto de vista funcional como de eficiencia. El recifrado delegado es un tipo de cifrado de clave pública que permite delegar en una entidad la capacidad de transformar textos cifrados de una clave pública a otra, sin que pueda obtener ninguna información sobre el mensaje subyacente. Desde un punto de vista funcional, el recifrado delegado puede verse como un medio de delegación segura de acceso a información cifrada, por lo que representa un candidato natural para construir mecanismos de control de acceso criptográficos. Aparte de esto, este tipo de cifrado es, en sí mismo, de gran interés teórico, ya que sus definiciones de seguridad deben balancear al mismo tiempo la seguridad de los textos cifrados con la posibilidad de transformarlos mediante el recifrado, lo que supone una estimulante dicotomía.

Las contribuciones de esta tesis siguen un enfoque transversal, ya que van desde las propias definiciones de seguridad del recifrado delegado, hasta los detalles específicos de potenciales aplicaciones.

A.1 Motivación, retos y contribuciones

Existe un creciente interés por el modelo de computación en la nube, debido en parte al abaratamiento de costes que supone, sobre todo para las pequeñas empresas, a la hora de escalar y redimensionar su negocio. La idea básica tras el concepto de *cloud* es la de proveer al usuario de una abstracción de un conjunto disponible de recursos de computación, almacenamiento y comunicaciones; en este caso, el modelo de negocio se basa en que los usuarios pagan en función del uso de los servicios ofrecidos por el cloud. Sin embargo, la externalización inherente a este paradigma implica también un riesgo para los usuarios, que ahora están obligadas a confiar en la honestidad y fiabilidad de un proveedor de servicios de cloud. En principio, nada impide a cualquier proveedor acceder a la información de los usuarios, por lo que estos últimos solo pueden confiar en que esto no suceda.

La mayor parte de las medidas de seguridad en entornos cloud (como sistemas de control de acceso, cortafuegos de red, etc.) están destinadas a la protección frente a atacantes externos; no obstante, las amenazas internas son potencialmente más peligrosas, ya que a menudo pueden sortear las medidas de seguridad habituales. Por tanto, una premisa más realista es que los atacantes (ya sea externos o internos), pueden acceder a los datos almacenados. La consecuencia lógica, por tanto, es que es necesario introducir mecanismos de cifrado para mantener la confidencialidad y privacidad. Aunque el uso de mecanismos de cifrado ofrece protección ante atacantes externos e internos, dificulta (y en algunos casos, imposibilita) la gestión y procesamiento de los datos. Es por ello necesario recurrir a mecanismos de cifrado que trasciendan los tradicionales, de forma que se pueda contar con ciertos niveles de funcionalidad sin perjudicar la seguridad. Una de las funcionalidades más básicas que podemos considerar es la compartición de los datos, que es la que vamos a considerar como motivación de esta tesis.

En un escenario de *compartición segura de datos*, identificamos los siguientes actores (y dominios asociados), como se ilustra en la Figura A.1:

- Productores de datos: Son las entidades que generan datos. En nuestro escenario diferenciamos quién crea los datos de quién es el dueño, lo que nos permite incluir en este escenario casos de uso más diversos y complejos.

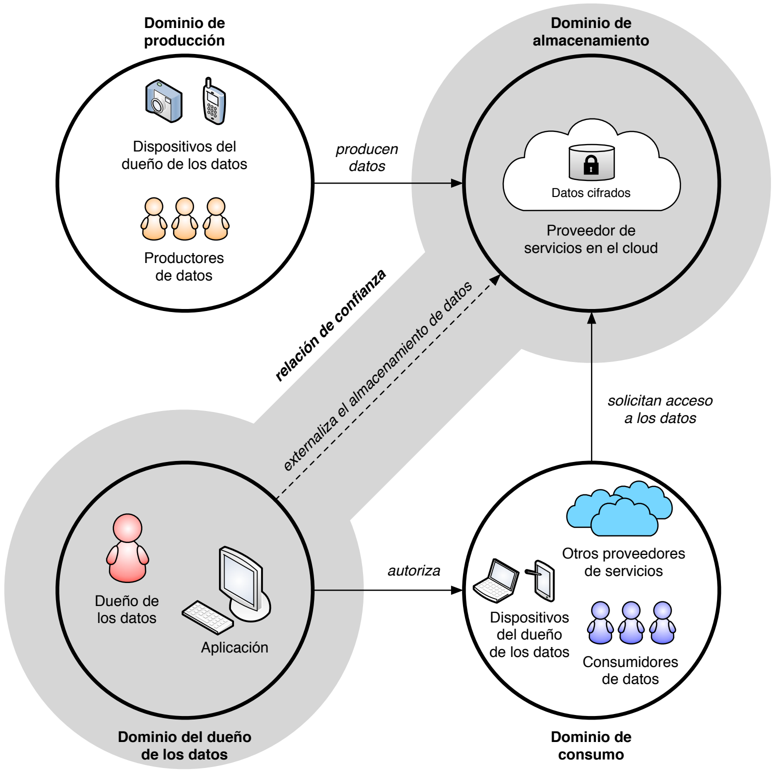

Dominio de producción
Dispositivos del dueño de los datos
Productores de datos
producen datos
Dominio de almacenamiento
Datos cifrados
Proveedor de servicios en el cloud
relación de confianza
externaliza el almacenamiento de datos
solicitan acceso a los datos
Dueño de los datos
Aplicación
autoriza
Otros proveedores de servicios
Dispositivos del dueño de los datos
Consumidores de datos
Dominio del dueño de los datos
Dominio de consumo

Figura A.1: Actores y dominios en el escenario de compartición segura de datos

Los productores pueden cifrar la información desde su dominio, e incluso, enviarla directamente al proveedor de almacenamiento, de forma que la interacción con el dueño de los datos no es un requisito en este escenario.

- Dueño de los datos: La entidad que posee los datos y que quiere compartirlos. Su principal función es autorizar consumidores a acceder a sus datos.

- Proveedor de almacenamiento: Es una entidad especializada en gestionar datos de usuarios y de proporcionar un servicio de compartición. Dado que un proveedor de cloud es un gran ejemplo de este tipo de actor, nos referiremos a esta entidad también como "proveedor de cloud". Este actor es semiconfiable, ya que asumiremos que proporciona el servicio de manera correcta y honesta, pero al mismo tiempo tiene incentivos para tratar de leer los datos que tenga en su poder; en este caso, se dice que es de tipo "honesto pero curioso".

- Consumidores de datos: Estas entidades son los receptores legítimos de la información compartida, que obtienen mediante interacción con el proveedor de almacenamiento.

- Actores externos: Un quinto tipo de actor, omitido en la Figura A.1, es toda aquella entidad externa que tenga un interés en acceder a la información protegida (como por ejemplo, hackers). Este tipo de actores deben enfrentarse a las medidas de seguridad tradicionales (p. ej. cortafuegos), y en el peor de los casos, si logran acceder verán información cifrada.

Si usáramos exclusivamente técnicas convencionales de cifrado (p. ej., de tipo simétrico como AES, o asimétrico como RSA) no podríamos resolver de forma adecuada la problemática de la compartición segura de datos, ya que nos encontraríamos con grandes problemas de gestión y distribución de claves debido a la configuración de actores existente. Uno de los problemas más acuciantes es que los productores no conocen de antemano quiénes van a ser los consumidores de la información cifrada, por lo que se ven obligados a cifrarla con una clave controlada por el dueño de los datos, de forma que pueda posteriormente autorizar a los consumidores correspondientes. Con las técnicas convencionales, esto implica que el dueño de los datos debe descifrar los datos y cifrarlos de nuevo con una clave conocida por los consumidores, lo que es una solución muy ineficiente. El problema se vuelve inmensamente complejo cuando hay numerosos conjuntos de datos, productores y consumidores.

Una manera elegante de resolver este problema es mediante el uso de esquemas de *recifrado delegado* (en inglés, *proxy re-encryption*), un tipo de cifrado de clave pública que permite delegar en una entidad de recifrado (referida en inglés como *proxy*) la capacidad de transformar textos cifrados de una clave pública a otra, sin que pueda obtener ninguna información sobre el mensaje subyacente, como puede verse en la Figura A.2. Para ello, la entidad de recifrado necesita que el destinatario original le envíe previamente una clave de recifrado, que le permite realizar la transformación. Desde un punto de vista funcional, el recifrado delegado puede verse como un medio de delegación segura de acceso a información cifrada.

En esta tesis, postulamos que el recifrado delegado es un candidato natural para el escenario de compartición segura de datos, ya que permite construir sistemas de control de acceso basados en la criptografía. En una solución basada en el recifrado delegado, la información a proteger se almacena en el cloud de forma cifrada, aunque puede compartirse con consumidores autorizados por medio de la operación de recifrado, todo ello manteniendo la confidencialidad de los datos respecto a entidades no autorizadas y el propio proveedor de cloud.

Sin embargo, existen varios retos de investigación que deben tratarse antes de avanzar en el uso del recifrado delegado en el mundo real, que clasificamos de acuerdo a su nivel de abstracción:

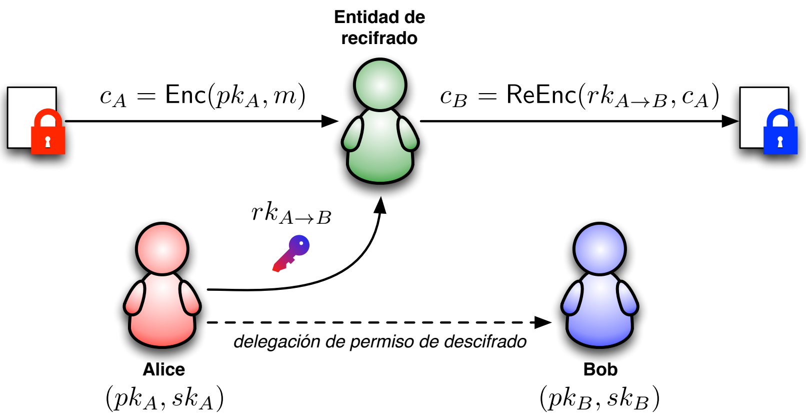

Entidad de recifrado
$c_A = \mathsf{Enc}(pk_A, m)$
$c_B = \mathsf{ReEnc}(rk_{A\to B}, c_A)$
$rk_{A\to B}$
delegación de permiso de descifrado
Alice
(pk_A, sk_A)
Bob
(pk_B, sk_B)

Figura A.2: Recifrado Delegado

- Definiciones de seguridad: Los retos en este nivel están asociados al propio concepto de seguridad de los esquemas de recifrado delegado, desde un punto de vista criptográfico. Esto involucra diversas necesidades, como el análisis de las nociones de seguridad, el modelado de adversarios y la investigación de propiedades relevantes, siendo la primera la más acuciante. En los esquemas de recifrado delegado es habitual reutilizar nociones de seguridad heredadas de la criptografía de clave pública; aunque esto tiene sentido inicialmente, no se ha estudiado con detenimiento el impacto en las nociones de seguridad que surge de dotar a un adversario de la capacidad de recifrar.

- Construcciones: A este nivel, otras aspectos se tornan más relevantes, como las técnicas concretas para dotar de seguridad a los esquemas y el desempeño logrado. Respecto a lo primero, existen varias técnicas genéricas para mejorar la seguridad de los esquemas de clave pública; sin embargo, no contamos con técnicas similares para el recifrado delegado. Respecto a lo segundo, es importante que un buen desempeño sea un objetivo prioritario, ya que estamos defendiendo el uso del recifrado delegado como mecanismo de control de acceso, lo que implica un uso potencial de este tipo de esquemas en entornos de alta demanda; no podemos olvidar tampoco, que cualquier sobrecarga computacional o de comunicaciones tiene, a la larga, un impacto económico, que puede afectar a la viabilidad de las soluciones.

- Aplicaciones: El escenario de compartición segura de datos puede instanciarse con diversos casos de uso, lo que requiere integrar los mecanismos de recifrado delegado de manera correcta, analizando protocolos, arquitecturas e implementaciones. Estos aspectos son muy importantes, ya que a menudo se obvia el enlace entre las contribuciones teóricas y los sistemas reales.

Estos retos constituyen las cuestiones dominantes en esta tesis, y por lo tanto, el recifrado delegado es el tema central. Teniendo estos retos en mente, en esta tesis hemos realizado las siguientes contribuciones:

- Estudiamos los conceptos básicos del recifrado delegado, incluyendo definiciones, modelos de seguridad y propiedades.
- Analizamos el estado del arte sobre esquemas de recifrado delegado, incluyendo un estudio comparativo del desempeño, tanto desde un punto de vista teórico como experimental. Además, estudiamos la literatura en cuanto aplicaciones del recifrado delegado, y en particular, las asociadas al problema de la compartición segura de información.
- Examinamos las nociones de seguridad del recifrado delegado e identificamos una familia paramétrica de modelos de ataque, que considera la disponibilidad tanto del oráculo de descifrado como de recifrado. Esta familia de modelos de ataque nos permite definir un conjunto de nociones de seguridad, cuyas relaciones también analizamos.
- Estudiamos la aplicabilidad de transformaciones genéricas para mejorar la seguridad del recifrado delegado. En particular nos centramos en la transformación Fujisaki-Okamoto, y formulamos las condiciones que nos permiten aplicarla, incluyendo una nueva propiedad que llamamos "sustitución perfecta de claves".
- Proponemos un nuevo esquema de recifrado delegado altamente eficiente, llamado NTRUReEncrypt, y basado en el criptosistema NTRU [16]. Además, describimos una versión para la que demostramos su seguridad.
- Proponemos varias aplicaciones del recifrado delegado: un modelo de gestión de identidad respetuoso con la privacidad, un sistema de delegación de acceso a información cifrada en clusters Big Data, y un criptosistema con descifrado en depósito.

A.2 Análisis sistemático del estado del arte sobre el recifrado delegado

En este capítulo, hacemos un estudio detallado del recifrado delegado, y analizamos el estado del arte, tanto de esquemas como de aplicaciones, siguiente un método bibliométrico. Mediante este método filtramos una lista inicial de 152 publicaciones, donde 83 son sobre esquemas y 69 sobre aplicaciones. Para las publicaciones menos recientes, usamos el número de citas como heurístico de calidad, aunque en todo momento es necesario una comprobación manual para evitar descartar publicaciones relevantes. El resultado de esta fase es una colección de 58 publicaciones, donde 13 son de esquemas y 45 de aplicaciones.

A.2.1 Esquemas

La primera parte de este trabajo se centra en estudiar los conceptos básicos y propiedades principales del recifrado delegado, así como de un análisis exhaustivo de los principales esquemas. Seleccionamos un total de 13 publicaciones [43, 11, 45, 13, 14, 63, 64, 54, 12, 46, 47, 48, 49], de las cuales surgen 19 esquemas. El objetivo es estudiar las características de estos esquemas, en función de las propiedades más importantes; ver la Sección 3.3.1 para más detalles. Esto nos permite realizar también un estudio comparativo, recogido en la Sección 3.3.2. Finalmente, en la Sección 3.3.3, hacemos un análisis del desempeño de los esquemas, tanto desde un punto de vista teórico, como práctico.

A.2.2 Aplicaciones

La segunda parte de este estudio, contenida en la Sección 3.4, está dedicada al análisis de 45 aplicaciones del recifrado delegado. En un análisis preliminar, clasificamos cada aplicación respecto al objetivo que persigue, el escenario y la funcionalidad implementada. El resultado no es sorprendente: la mayoría de las aplicaciones están centradas en proteger la confidencialidad de datos almacenados en escenarios tipo cloud, de forma similar al escenario de la compartición segura de datos. Las soluciones propuestas son variaciones de la plantilla usual de aplicación del recifrado delegado a este problema.

A.3 Familia paramétrica de modelos de ataque

El recifrado delegado se puede considerar como un tipo de criptosistema de clave pública, por lo que es natural reutilizar sus nociones de seguridad (ver Sección 2.3.1). Por tanto, los esquemas de recifrado intentan lograr nociones de seguridad del tipo IND-CPA [11], IND-CCA1 [48] y IND-CCA2 [45]. Sin embargo, a menudo estas nociones se definen de manera ad-hoc para cada esquema, con sutiles diferencias y restricciones, lo que dificulta la comparación y análisis. Esto está causado por una falta de definiciones comunes de las nociones de seguridad para el recifrado delegado, y en particular, de los modelos de ataque. En comparación con el caso de la criptografía de clave pública, los modelos de ataque para el recifrado delegado son potencialmente más complejos, ya que deben considerar no solo la disponibilidad del oráculo de descifrado, si no también el de recifrado.

En este trabajo exploramos estas sutilezas mediante una familia de modelos de ataque parametrizada por la disponibilidad de ambos oráculos. Esto permite, a su vez, definir una familia de nociones de seguridad, que analizamos en función de las relaciones de implicación y separación que surgen entre dichas nociones. Las separaciones que identificamos son consecuencia del estudio de una nueva propiedad del recifrado delegado que llamamos "privacidad de las claves de recifrado" y que formaliza el concepto de que las claves de recifrado no deben filtrarse mediante consultas al oráculo de recifrado. Finalmente, mostramos cómo un esquema de recifrado presentado en PKC 2014 [48] es vulnerable a raíz de no cumplir dicha propiedad.

A.3.1 Modelos de ataque y nociones de seguridad

Un modelo de ataque define las capacidades de un adversario hipotético, normalmente a través de la disponibilidad de oráculos con una determinada funcionalidad. Bellare et al. definen en [24] una regla mnemotécnica para identificar los modelos de ataque CCA2, CCA1 y CCA0 (es decir, CPA) en la criptografía de clave pública, según la cual el índice en CCAi especifica la última fase en la que el adversario tiene acceso al oráculo de descifrado. De forma análoga, en el contexto del recifrado delegado definimos una familia de modelos de ataque de la forma $\mathsf{CCA}_{i,j}$, con $i, j \in \{0, 1, 2\}$, en donde los índices i y j especifican la última fase en la que el adversario tiene acceso a los oráculos de descifrado y recifrado, respectivamente. Esto da lugar a un conjunto de nueve modelos. Como casos extremos tenemos $\mathsf{CCA}_{0,0}$, equivalente a CPA, y $\mathsf{CCA}_{2,2}$, que representa el modelo de ataque más completo. La única diferencia entre los nueve modelos de ataque es la disponibilidad de los oráculos de descifrado y recifrado; el resto de oráculos habituales en el recifrado delegado se mantienen.

Al igual que en la criptografía de clave pública, las nociones de seguridad se construyen como una combinación entre un objetivo de seguridad (como p. ej, indistinguibilidad de los cifrados (IND), que formaliza la incapacidad del adversario de distinguir entre qué mensaje está cifrado en un determinado criptograma) y un modelo de ataque, que en este caso pertenece a la familia paramétrica. Si fijamos el objetivo de seguridad IND obtenemos nueve nociones de seguridad, de la forma $\mathsf{IND\text{-}CCA}_{i,j}$, con $i, j \in \{0, 1, 2\}$.

A.3.2 Relaciones entre nociones

Implicaciones

Las nociones se seguridad obtenidas pueden organizarse jerárquicamente, como muestra la Figura 4.1, en donde pueden verse las implicaciones triviales entre nociones. Los Teoremas 4.4 y 4.5 formalizan la idea de que un esquema que es seguro en una determinada noción, lo sigue siendo en nociones inferiores en la jerarquía.

Separaciones

Aparte de las implicaciones triviales que acabamos de comentar, en este trabajo estudiamos algunas separaciones – lo que es, si cabe, más interesante. La Figura 4.2 muestra las relaciones encontradas en la jerarquía de nociones, enfatizando aquellas que representan una separación.

Las separaciones analizadas son consecuencia de las vulnerabilidades de los esquemas de recifrado delegado que no satisfacen la propiedad de *privacidad de las claves de recifrado.* Esta propiedad, descrita primero en [45] de manera informal, captura la noción de que un adversario no debe de ser capaz de extraer una clave de recifrado a partir de un texto cifrado y su correspondiente recifrado. En este trabajo, hacemos una definición más formal de dicha propiedad (Definición 4.6), según la cual un esquema de recifrado delegado la cumple si la probabilidad de que un adversario que tiene acceso al oráculo de recifrado $\mathcal{O}_{reenc}$ tenga una probabilidad insignificante de computar una clave de recifrado válida.

En función del tipo de clave de recifrado obtenida, las consecuencias para el esquema pueden ser más o menos graves. Si la clave obtenida permite recifrar desde un usuario honesto hacia otro usuario honesto, entonces podemos demostrar que el esquema no puede lograr seguridad $\mathsf{IND\text{-}CCA}_{2,1}$ o $\mathsf{IND\text{-}CCA}_{2,2}$ (Teorema 4.7). Sin embargo, si la clave obtenida permite recifrar desde un usuario honesto hacia uno corrupto, entonces el esquema no puede ser ni siquiera de tipo $\mathsf{IND\text{-}CCA}_{0,1}$, lo que produce una separación mucho más grande entre nociones (Teorema 4.8).

Para demostrar ambos teoremas se siguen estrategias similares. La primera es que, asumiendo que el adversario puede extraer una clave de recifrado desde el usuario objetivo hacia uno honesto mediante llamadas al oráculo de recifrado, entonces este puede ganar el juego si usa esta clave para recifrar localmente el reto c^* en c_h, para posteriormente utilizar el oráculo de descifrado sobre c_h; esto funciona porque se siguen cumpliendo las restricciones relativas a los derivados del reto, ya que no se ha utilizado el oráculo de recifrado para recifrarlo. La separación asociada a este teorema, $\textsf{IND-CCA}_{2,0} \not\Rightarrow \textsf{IND-CCA}_{2,1}$, surge al observar que no es posible seguir el ataque descrito cuando no se tiene disponibilidad del oráculo de recifrado para obtener la clave de recifrado.

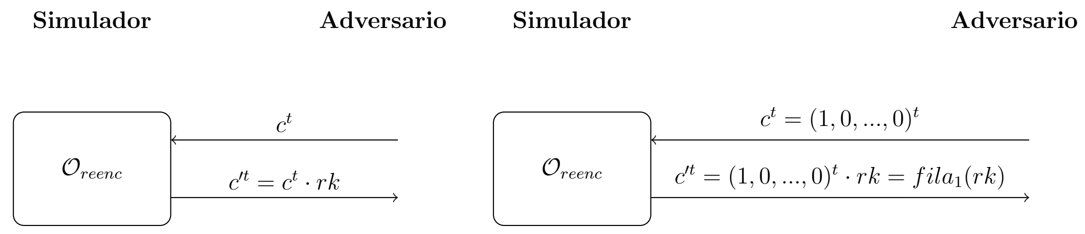
La estrategia utilizada para la segunda demostración es similar, excepto que ahora la clave de recifrado obtenida permite recifrar hacia un usuario corrupto; consecuentemente, no es necesario utilizar el oráculo de descifrado al final. Por lo tanto, la separación asociada es $\textsf{IND-CCA}_{2,0} \not\Rightarrow \textsf{IND-CCA}_{0,1}$, ya que se sigue necesitando el oráculo de recifrado en algún momento para obtener la clave de recifrado.

A.3.3 Atacando un esquema que no garantiza la privacidad de las claves de recifrado

La importancia de la propiedad de privacidad de las claves de recifrado y de estudiar correctamente los modelos de ataque queda de manifiesto con una vulnerabilidad que detectamos en el esquema de Kirshanova [48], presentado en PKC 2014, y que logra supuestamente "seguridad CCA1". Al analizar su prueba de seguridad, observamos que no confiere una capacidad adecuada de recifrado al adversario, correspondiente a un modelo de ataque CCA1. Nuestro estudio revela que en realidad este esquema es de tipo $\textsf{IND-CCA}_{1,0}$, ya que aunque permite un oráculo de descifrado en la primera fase, el esquema es vulnerable si se introduce un oráculo de recifrado. El motivo es que no cumple la propiedad de privacidad de las claves de recifrado, por lo que se puede usar un ataque similar a los descritos en las separaciones anteriores.

El esquema de Kirshanova es uno de los primeros esquemas de recifrado basados en retículos. Es una extensión del cifrado de clave pública de Micciancio y Peikert [40]. En la Sección 4.4 puede verse una descripción detallada del esquema.

La idea básica del ataque es la siguiente: la clave de recifrado es una matriz rk, y la operación de recifrado consiste únicamente en la multiplicación de un vector fila, que contiene el texto cifrado, con dicha matriz, tal como muestra la Figura A.3a. Por lo tanto, si usamos como vector de entrada un vector unitario (o más específicamente, los vectores de la base canónica de dimensión correspondiente), obtendremos una de las filas de dicha matriz, como se ve en la Figura A.3b. Repitiendo este proceso por cada fila de la matriz rk, podemos recuperarla por completo. En la Sección 4.4, describimos además un refinamiento de este ataque, en el cual en lugar de usar vectores unitarios como entrada, usamos vectores aleatorios, de tal forma que la agrupación de estos vectores dé lugar a una matriz



(a) El oráculo de recifrado (b) Vector unitario como entrada

Figura A.3: Filtrando la clave de recifrado en el esquema de Kirshanova

invertible P. Si agrupamos las respuestas en forma de matriz obtendremos, por tanto, $P \cdot rk$; el siguiente paso es multiplicar por la izquierda esta matriz con P^{-1} para recuperar la clave de recifrado rk. Una vez se ha filtrado la clave de recifrado, podemos continuar con la estrategia descrita en la demostración del Teorema 4.8, de forma que el adversario gane el juego.

A.4 Esquemas de recifrado delegado basados en NTRU

Otro de los temas que tratamos en esta tesis es el uso de fundamentos alternativos para la construcción de esquemas de recifrado delegado, que permitan lograr mayor seguridad y/o eficiencia. La mayoría de los esquemas de recifrado delegado están basados en problemas relativos al Logaritmo Discreto. En este capítulo estudiamos el criptosistema NTRU, un destacado ejemplo de la criptografía basada en retículos[1]. A partir de este esquema, proponemos una variante de recifrado delegado llamada NTRUReEncrypt. Nuestra propuesta es un esquema bidireccional y multiuso, altamente eficiente. Además de este esquema, también describimos otra variante cuya seguridad es demostrable; en concreto, probamos que logra seguridad IND-CPA bajo la asunción Ring-LWE. Nuestras propuestas son los primeros esquemas de recifrado delegado basados en NTRU. Los resultados experimentales muestran que la eficiencia de nuestro primer esquema es ampliamente superior a esquemas anteriores, por un orden de magnitud.

[1]También llamados *redes*, término que evitaremos por su posible confusión con las redes de comunicaciones.

A.4.1 La criptografía basada en retículos y NTRU

La inmensa mayoría de los esquemas de recifrado delegado están basados en grupos cíclicos o en emparejamientos criptográficos (en inglés, *pairings*); otros fundamentos, sin embargo, han recibido menos atención. Recientemente, se han propuesto algunos esquemas de recifrado basados en retículos [46, 47]. La criptografía basada en retículos es un campo muy prometedor, principalmente por su papel en la criptografía post-cuántica (es decir, asumiendo la existencia de ordenadores cuánticos eficientes), aunque en algunos casos, también por su desempeño. El mejor ejemplo de criptosistema basado en retículos es NTRU, originalmente propuesto por Hoffstein, Pipher y Silverman en [16]. El principal interés en NTRU radica en su desempeño, mucho mayor que otros esquemas de clave pública e incluso comparable a cifrados simétricos. Por ejemplo, versiones optimizadas de NTRU ejecutadas en GPUs son hasta 1000 veces más rápidas que RSA y 100 veces que curvas elípticas [124]. Esto se debe a que la operación principal en NTRU es la multiplicación de polinomios con coeficientes relativamente pequeños; esta operación se puede implementar de forma muy eficiente e incluso en paralelo. Además, también implica menores tamaños de clave. Gracias a su eficiencia, es considerado como el esquema basado en retículos más práctico y maduro actualmente, hasta el punto de ser estandarizado (IEEE Std 1363.1-2008 [79], ANSI X9.98-2010 [123]).

A.4.2 El esquema NTRUReEncrypt

Antes de explicar el esquema, es necesario establecer las bases sobre las que lo definimos. El criptosistema NTRU original está definido sobre el anillo cociente de polinomios $\mathcal{R}_{NTRU} = \mathbb{Z}[x]/(x^n - 1)$, con n primo. Por lo tanto, los elementos de $\mathcal{R}_{NTRU}$ son polinomios enteros de grado menor a n. Otros parámetros de NTRU son el entero q, que es una potencia de 2 del mismo orden de magnitud que n, y el polinomio $p \in \mathcal{R}_{NTRU}$, que suele tomar valores $p = 3$ ó $p = x + 2$. En general, las operaciones sobre polinomios se desarrollarán en $\mathcal{R}_{NTRU}$ módulo q y módulo p, a los que nos referiremos respectivamente como $\mathcal{R}_{NTRU}/q$ y $\mathcal{R}_{NTRU}/p$.

Nuestro esquema es realmente una extensión del criptosistema NTRU original, que consta de los siguientes algoritmos:

- KeyGen(): El resultado del algoritmo de generación de claves para un usuario A es un par de claves (pk_A, sk_A). Primero se eligen un par de polinomios $f_A, g_A \in \mathcal{R}_{NTRU}$ aleatoriamente, aunque con un determinado número de coeficientes iguales a 0, -1, y 1. Además, f_A debe ser congruente con 1 módulo p y tener un inverso f_A^{-1} en $\mathcal{R}_{NTRU}/q$. La clave privada sk_A es el polinomio f_A, mientras que la clave pública pk_A es el polinomio $h_A = p \cdot g_A \cdot f_A^{-1} \mod q$.

- $\mathsf{Enc}(pk_A, M)$: Tomando como entrada una clave pública $pk_A = h_A$ y un mensaje $M \in \mathcal{R}_{NTRU}/p$, el algoritmo de cifrado genera un polinomio aleatorio pequeño $s \in \mathcal{R}_{NTRU}$, y produce el texto cifrado $C_A = h_A s + M$.

- $\mathsf{Dec}(sk_A, C_A)$: Tomando como entrada la clave secreta $sk_A = f_A$ y un texto cifrado C_A, el algoritmo de descifrado computa primero $C'_A = (C_A \cdot f_A) \mod q$ y devuelve el mensaje original $M = (C'_A \mod p)$.

A este conjunto de algoritmos le añadimos las funciones necesarias para obtener un esquema de recifrado delegado:

- $\mathsf{ReKeyGen}(sk_A, sk_B)$: Tomando como parámetros de entrada las claves secretas $sk_A = f_A$ y $sk_B = f_B$, el algoritmo de generación de claves de recifrado computa la clave de recifrado entre los usuarios A y B como $rk_{A\to B} = sk_A \cdot sk_B^{-1} = f_A \cdot f_B^{-1}$.

- $\mathsf{ReEnc}(rk_{A\to B}, C_A)$: Tomando como entrada una clave de recifrado $rk_{A\to B}$ y un texto cifrado C_A, el algoritmo de recifrado toma un polinomio aleatorio $e \in \mathcal{R}_{NTRU}$ y produce el texto cifrado $C_B = C_A \cdot rk_{A\to B} + pe$.

En la Sección 5.1.3 se demuestra la corrección de este esquema. En cuanto a sus propiedades, es fácil comprobar que el esquema es bidireccional, ya que dada una clave $rk_{A\to B} = f_A f_B^{-1}$, cualquiera puede computar la clave opuesta $rk_{B\to A} = (rk_{A\to B})^{-1} = f_B f_A^{-1}$. Este esquema es también multiuso, ya que se permiten múltiples recifrados, aunque de forma limitada, ya que el ruido añadido durante el recifrado se va acumulando hasta que hace imposible el descifrado.

Puede verse que los costes computacionales asociados a NTRUReEncrypt en los principales algoritmos (es decir, cifrado, descifrado y recifrado) solo implican una multiplicación. Esta es la operación más importante en NTRU, que puede hacerse en tiempo cuasi-lineal usando la Transformada Rápida de Fourier [124]. Nuestros resultados experimentales muestran que nuestro esquema de recifrado delegado preserva el desempeño original de NTRU. Además, con el fin de comparar nuestra propuesta con respecto a otros esquemas, hemos implementado tres de ellos con características similares (BBS [43], Weng et al. [54] y Aono et al. [47]); este último es también basado en retículos. La Figura A.4 muestra los resultados de nuestros experimentos. Puede verse que nuestro esquema supera a los demás por un orden de magnitud (entre 10 y 20 veces más rápido). El esquema de Aono et al. muestra un desempeño similar en cuanto a cifrado y descifrado, pero es 20 veces más lento en el recifrado.

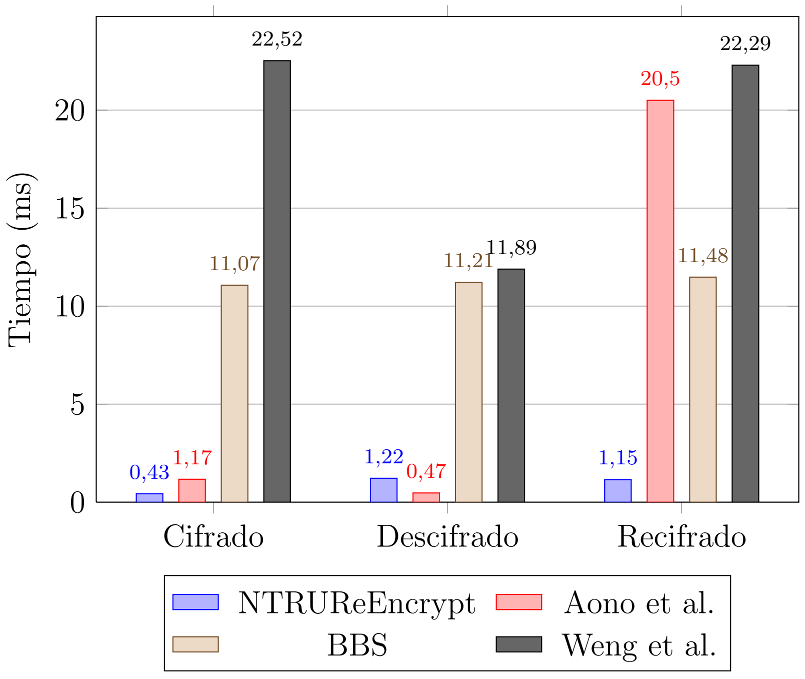


Figura A.4: Comparación con otros esquemas de recifrado

Tabla A.1: Comparación de costes espaciales (en KB)

Elemento	Aono et al. [47]	NTRUReEncrypt
Claves públicas	60.00	1.57
Claves privadas	60.00	1.57
Claves de recifrado	2520.00	1.57
Textos cifrados	0.66	1.57

En cuanto a los costes espaciales, estos son también de orden cuasi-lineal; por contra, otros esquemas basados en retículos, como el de Aono et al. [47], tienen tamaños de clave y de mensajes que crecen de forma cuadrática. La Tabla A.1 ilustra este hecho al mostrar una comparación entre el esquema de Aono et al. y el nuestro, para unos determinados parámetros. Como usamos el mismo tipo de polinomios para representar todas las claves y los textos cifrados, todos ellos tienen el mismo tamaño (1.57 KB). El caso de las claves de recifrado es especialmente representativo, ya que la diferencia entre ambos esquemas es de tres órdenes de magnitud.

A.4.3 El esquema PS-NTRUReEncrypt

Uno de los problemas del esquema NTRUReEncrypt es que, al igual que NTRU, no tiene una demostración de seguridad. Stehlé y Steinfeld propusieron en [73, 122] una variante de NTRU cuya seguridad CPA se demuestra ante la asunción Ring-LWE, relativa a un problema intratable de la criptografía sobre retículos. Su variante es muy similar al esquema NTRU, con ligeras variaciones menores en la generación de claves y en el algoritmo de cifrado. La diferencia fundamental, no obstante, está en la elección de los anillos de polinomios utilizados, ya que permiten reducir la seguridad del esquema a la dificultad en resolver el problema Ring-LWE.

Basándonos en esta variante, definimos también un nuevo esquema de recifrado delegado llamado PS-NTRUReEncrypt. Por motivos de espacio, no entraremos en explicar los detalles de este esquema, aunque destacamos que realizamos también una demostración de seguridad y de corrección. El esquema resultante es también bidireccional y multiuso.

A.5 Aplicabilidad de transformaciones genéricas para seguridad CCA

El segundo esquema que propusimos en el capítulo anterior logra seguridad CPA, aunque lo deseable sería conseguir seguridad CCA. Hay dos estrategias posibles que pueden seguirse: una es rediseñar completamente el esquema, y otra es tratar de mejorar la noción de seguridad original usando algún método genérico.

Para el caso de la criptografía de clave pública, existen transformaciones genéricas, como la Fujisaki-Okamoto y REACT [17, 18]. Sin embargo, este no es el caso del recifrado delegado. En este trabajo, estudiamos la adaptación de estas transformaciones al contexto del recifrado delegado y encontramos resultados tanto positivos como negativos.

A.5.1 Transformaciones para el recifrado delegado

La transformación Fujisaki-Okamoto original

Fujisaki y Okamoto proponen en [71, 17] una transformación genérica para lograr seguridad IND-CCA en el modelo del oráculo aleatorio a partir de un esquema de clave pública que logre seguridad OW-CPA, una débil noción de seguridad. Para ello, la transformación integra también un esquema de cifrado simétrico y funciones hash. La transformación Fujisaki-Okamoto, que denotaremos como Hyb, asume un cifrado de clave pública PKE, un cifrado simétrico Sym, y funciones hash H y G. Asumiremos también que la función de cifrado en PKE, a parte de tomar una clave pública y un mensaje como entrada, también toma un parámetro adicional para añadir aleatoriedad.

Para cifrar un mensaje m, la transformación toma primero un elemento aleatorio σ del espacio de mensajes de PKE. El mensaje m es cifrado con Sym usando $G(\sigma)$ como clave, lo que produce $c = \mathsf{Sym.Enc}(G(\sigma), m)$. A continuación, el término σ se cifra con PKE, tomando $H(\sigma, c)$ como la aleatoriedad de entrada. Por tanto, la transformación produce la siguiente tupla como el texto cifrado de m:

$$\mathsf{Hyb.Enc}(pk, m) = (\mathsf{PKE.Enc}(pk, \sigma, H(\sigma, c)), c)$$

Para descifrar una de estas tuplas (e, c), la transformación sigue el procedimiento inverso: descifra σ a partir de e, y computa $G(\sigma)$ para obtener la clave de descifrado de c, lo que le permite calcular el mensaje original m. Una sutileza de esta transformación es que después de descifrar el mensaje m, es necesario recalcular $e = \mathsf{PKE.Enc}(pk, \sigma, H(\sigma, c))$ para verificar que es igual que el original. Si no lo es, el texto cifrado se considera como no válido.

Es precisamente este último paso el motivo por el cual la transformación falla si se aplica directamente con un esquema de recifrado delegado: la función de recifrado cambia los textos cifrados de forma que esta validación falla inevitablemente, concretamente al alterar la aleatoriedad original. En la Sección 5.2.4 describimos cómo el segundo esquema de Aono et al. [47] falla precisamente por no considerar esta sutileza.

Extendiendo la transformación Fujisaki-Okamoto

El problema anterior nos lleva a preguntarnos si existen esquemas de recifrado en donde la función de recifrado no altere la aleatoriedad original. En este trabajo caracterizamos una propiedad que captura esta noción, llamada *sustitución perfecta de claves*, y la utilizamos como una de las condiciones para aplicar exitosamente la transformación Fujisaki-Okamoto al recifrado delegado.

Informalmente, un esquema de recifrado con sustitución perfecta de claves se caracteriza porque el resultado de la función de recifrado desde una clave a otra es idéntico al que ocurriría si se cifrara directamente con la segunda clave y usando la misma aleatoriedad. En otras palabras, el recifrado "sustituye" limpiamente una clave por otra, sin que se vea afectada la aleatoriedad original. La Definición 5.8 captura esta noción de manera formal:

$$\mathsf{ReEnc}(rk_{i\to j}, \mathsf{Enc}(pk_i, m, r)) = \mathsf{Enc}(pk_j, m, r)$$

Los esquemas de recifrado BBS [43] y Canetti-Hohenberger [45] son ejemplos que cumplen esta propiedad. Por ejemplo, en el esquema BBS, los textos cifrados son de la forma $(pk^r, g^r \cdot m)$, para un exponente aleatorio r y claves publicas de la forma $pk = g^a$. Las claves de recifrado son cocientes entre exponentes $rk = y/x$. Se puede comprobar que:

$$\begin{aligned}\mathsf{ReEnc}(rk_{A\to B}, \mathsf{Enc}(pk_A, m, r)) &= \mathsf{ReEnc}(\frac{b}{a}, ((g^a)^r, g^r \cdot m)) \\ &= ((g^{ar})^{\frac{b}{a}}, g^r \cdot m) \\ &= (g^{br}, g^r \cdot m) = \mathsf{Enc}(pk_B, m, r)\end{aligned}$$

Puede verse que el texto cifrado original $(g^{ar}, g^r \cdot m)$ se transforma limpiamente en $(g^{br}, g^r \cdot m)$.

Una vez asumimos que el esquema de recifrado delegado cumple esta propiedad, el problema que aparecía en el paso de validación al intentar aplicar la transformación Fujisaki-Okamoto desaparece. La extensión de esta transformación al recifrado delegado es idéntica en cuanto a los algoritmos de generación de claves, cifrado y descifrado; queda, por tanto, definir las funciones de recifrado y generación de claves de recifrado.

- $\mathsf{Hyb.ReKeyGen}(pk_i, sk_i, pk_j, sk_j) \to rk_{i\to j}$. Tomando como entrada dos pares de claves públicas y privadas, correspondientes a los usuarios i y j, este algoritmo devuelve $rk_{i\to j} \leftarrow \mathsf{PRE.ReKeyGen}(pk_i, sk_i, pk_j, sk_j)$.
- $\mathsf{Hyb.ReEnc}(rk_{i\to j}, (e_i, c_i)) \to (e_j, c_j)$. Tomando como entrada una clave de recifrado $rk_{i\to j}$ y un texto cifrado (e_i, c_i), el algoritmo de recifrado devuelve el texto cifrado $(\mathsf{PRE.ReEnc}(rk_{i\to j}, e_i), c_i)$.

En la Sección 5.2.3 mostramos que esta extensión es correcta y proporcionamos una demostración de seguridad. Los problemas mencionados en la sección anterior con la definición del oráculo de recifrado en el modelo del oráculo aleatorio implican que es necesario asumir que el esquema original tiene seguridad $\mathsf{IND\text{-}CCA}_{0,1}$ (noción inmediatamente superior a IND-CPA) y que el esquema que se obtiene tras la transformación es de tipo $\mathsf{IND\text{-}CCA}_{2,1}$ (noción inmediatamente inferior a IND-CCA2). El lector puede encontrar más información sobre estas nociones en la Sección 4.3, así como una descripción gráfica de sus relaciones en la Figura 4.2.

Para ilustrar la transformación propuesta, en la Sección 5.2.5 la aplicamos a un esquema que cumple las condiciones (es decir, seguridad $\mathsf{IND\text{-}CCA}_{0,1}$ y sustitución perfecta de claves), obteniéndose un esquema con seguridad $\mathsf{IND\text{-}CCA}_{2,1}$ en el modelo del oráculo aleatorio. Finalmente, en la Sección 5.2.6 describimos cómo podrían extenderse otras transformaciones genéricas similares, como REACT [18] y GEM [29].

A.5.2 Imposibilidad de aplicación directa

La otra mitad de este trabajo consiste en mostrar que la aplicación directa de estas transformaciones implica fallos sutiles en las demostraciones de seguridad. En este trabajo detectamos fallos en una docena de esquemas. Uno de ellos es el esquema de Aono et al. [47], cuyo fallo describimos en la sección anterior. Los once esquemas restantes [55, 62, 54, 135, 136, 137, 138, 139, 140, 141, 142] tienen otro tipo de fallo, mucho más sutil y difícil de detectar; esta lista no es exhaustiva, por lo que no descartamos otros esquemas vulnerables.

Para comprender el fallo es necesario conocer cómo se hace la prueba de seguridad en la transformación Fujisaki-Okamoto original. Como el objetivo de esta transformación es lograr seguridad CCA, es necesario definir un oráculo de descifrado. En la prueba, este oráculo hace uso de las tablas asociadas a los oráculos aleatorios, que contienen un registro con las entradas y salidas de estas funciones. Asumiendo acceso a dichas tablas, es posible construir un oráculo de descifrado que no requiera la correspondiente clave de descifrado.

Esta estrategia, que funciona muy bien para el oráculo de descifrado, no puede usarse para el oráculo de recifrado. El problema consiste en que la demostración difiere de la ejecución real del esquema para cierto tipo de consultas de recifrado, lo que invalida las demostraciones. Este es precisamente el fallo común en los once esquemas anteriores.

El problema se basa en que en la función de cifrado los valores aleatorios se obtienen al llamar a una función hash con un valor que se mantiene secreto (por ejemplo, el propio mensaje original). Durante el descifrado, estos valores se pueden obtener de nuevo, por lo que es posible recuperar los valores aleatorios para realizar el paso de validación. Sin embargo, esto no es así en el recifrado, ya que los valores secretos no se desvelan en este proceso, por lo que no es posible verificar aquí si el texto cifrado se generó correctamente (es decir, usando las funciones hash de manera adecuada) o no. En las demostraciones asociadas a los esquemas fallidos, si un adversario hace una consulta a un oráculo de recifrado usando un texto cifrado no válido (por ejemplo, sin usar las funciones hash), entonces su oráculo de recifrado rechaza el texto cifrado al considerarlo inválido, ya que no hay un registro correspondiente en las tablas de los oráculos aleatorios. No obstante, en una ejecución real, no es posible comprobar esto, por lo que el recifrado va a funcionar normalmente. Consecuentemente, estas demostraciones de seguridad no son correctas al no corresponderse con el comportamiento real de los esquemas. En la Sección 5.2.4 damos más detalles sobre este fallo y lo ilustramos con el análisis de uno de los esquemas vulnerables.

A.6 Aplicaciones del recifrado delegado

La parte final de esta tesis está dedicada a aplicaciones concretas del recifrado delegado. La primera de ellas es un modelo de gestión de identidad respetuoso con la privacidad, llamado BlindIdM, en el que la información de identidad se mantiene cifrada en todo momento, pero manteniendo la funcionalidad. Además, describimos su integración con el estándar de gestión de identidad SAML 2.0. La segunda aplicación integra el recifrado delegado con Apache Hadoop, un sistema de computación para Big Data, de forma que los nodos de computación solo pueden descifrar la información en el momento de procesarla. La tercera aplicación es un criptosistema con descifrado en depósito, que funciona como un esquema de clave pública normal, pero permite a las autoridades descifrar criptogramas sospechosos (es decir, la capacidad de descifrar está en depósito), usando técnicas de recifrado delegado.

A.6.1 BlindIdM: Gestión de Identidad a Ciegas

La gestión de la identidad es uno de los procesos internos más importantes en las empresas y organizaciones, ya que tiene un papel ubicuo en el resto de procesos al formar parte de la autenticación, autorización, control de acceso, etc. Sin embargo, al mismo tiempo introduce una sobrecarga tanto en costes económicos como en tiempo, y en muchas ocasiones requiere el uso de aplicaciones y personal específico para la gestión y mantenimiento. Al igual que ya ha sucedido para muchos otros servicios, el paradigma del cloud representa una gran oportunidad para externalizar estos procesos.

Sin embargo, esto también conlleva riesgos ya que la información pasa a estar en un dominio externo; esta información, además, es sensible al tratarse de información de identidad. Lo habitual es simplemente confiar en el que proveedor de cloud no va a hacer un uso inadecuado de estos datos, pero hay múltiples casos en donde esto no ha sido así, ya sea de forma accidental o intencionada.

En esta aplicación proponemos un modelo de gestión de identidad en donde no sea necesaria esta relación de confianza. En su lugar, usaremos mecanismos criptográficos (concretamente, recifrado delegado) para garantizar la privacidad y confidencialidad de los usuarios, de forma que el proveedor de identidad en el cloud no sea capaz de leer los datos en su poder al estar cifrados, y al mismo tiempo, pueda seguir proporcionando un servicio de gestión de identidad.

Hacia la Gestión de Identidad como Servicio

El modelo de Federación de Identidad, ampliamente utilizado hoy en día, permite la portabilidad entre diferentes dominios, lo que fomenta una distribución dinámica de información de identidad y la delegación de subtareas, como la autenticación o el aprovisionamiento de usuarios. Uno de los aspectos clave de este modelo es el establecimiento de relaciones de confianza entre los miembros de una federación, lo que les permite dar validez a la información proporcionada por otros miembros. Los principales actores que participan en las interacciones de identidad son [155]:

- Usuarios, que son los sujetos en la información de identidad, y generalmente los actores que solicitan el acceso a recursos y servicios.
- Proveedores de Servicio (*SP*), que son entidades que proporcionan servicios y recursos a otras entidades.
- Proveedores de Identidad (*IdP*), entidades especializadas en gestionar información de identidad y proporcionarla a los proveedores de servicio.

En la Figura A.5a se aprecia una vista a alto nivel de un escenario de federación de identidad, donde una organización actúa como proveedor de identidad. En este escenario, uno de sus empleados solicita acceso a un servicio de un proveedor de servicios, que a su vez se dirige al proveedor de identidad para obtener información sobre este empleado.

Aunque la federación de identidad representa grandes ventajas en cuanto a interoperabilidad, también implica unos costes y sobrecargas, ya que requiere el uso de aplicaciones especiales y personal cualificado para su instalación, integración y gestión. El modelo de "Gestión de Identidad como Servicio" (en inglés, Identity Management as a Service (IDaaS)) puede verse como un refinamiento del modelo de identidad federada, en el que se usan las ventajas del cloud para ofrecer la externalización de la gestión de identidad. Entre los beneficios del modelo IDaaS destacamos: (i) mayor flexibilidad, escalabilidad y estabilidad en entornos de alta demanda; (ii) reducción de costes, ya que los proveedores de IDaaS pueden especializarse en proporcionar servicios más eficientes; y (iii), mejores medidas de seguridad, al utilizar estos proveedores sistemas e instalaciones dedicadas.

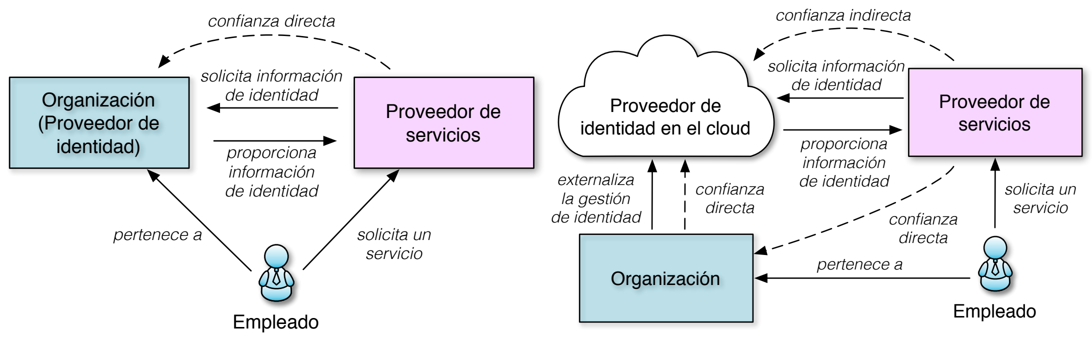
Sin embargo, existen también riesgos, que coinciden con los riesgos habituales del modelo de computación en la nube. Es obvio que la externalización de la gestión de identidad implica para las organizaciones una pérdida del control sobre esta información, que ahora recae sobre el proveedor de identidad.

El modelo BlindIdM

Para solucionar los problemas que surgen al gestionar la información de identidad en el cloud hemos propuesto el modelo BlindIdM (o Gestión de Identidad a Ciegas), en el que el proveedor de identidad en el cloud puede ofrecer un servicio de gestión de identidad sin necesidad de conocer la información de los usuarios (en otras a palabras, a ciegas). Esto es una gran innovación con respecto a los actuales sistemas de gestión de identidad, en los que los usuarios deben confiar en que los proveedores van a proteger la información y no harán un mal uso de ella. La solución en nuestro modelo pasa por mantener la información cifrada en todo momento, desde antes de llegar al proveedor de cloud, pero de forma que aún pueda ser usada por este; para ello, proponemos usar esquemas de recifrado delegado.

En nuestro modelo asumiremos un escenario de tipo federación de identidad, similar al mostrado en la Figura A.5a, pero en el que la organización externaliza los procesos de gestión de identidad al cloud, aunque manteniendo los servicios de autenticación. El proveedor de identidad en el cloud actúa de intermediario en las interacciones de identidad, y está a cargo de almacenar y proporcionar dicha información. La Figura A.5b muestra el escenario propuesto. En cuanto al modelo de confianza, asumimos entonces que el proveedor de identidad es un adversario de tipo "honesto pero curioso", lo que significa que seguirá los protocolos establecidos honestamente pero que intentará leer la información en su poder.

Nuestra propuesta concreta para el modelo BlindIdM usa SAML 2.0 como el protocolo subyacente de gestión de identidad y el recifrado delegado para lograr la confidencialidad extremo a extremo de la información de identidad. En el escenario planteado por este modelo, la organización (a la que pertenecen todos los empleados) actúa como dueño de los datos. Esta externaliza la gestión de identidad al proveedor de identidad en el cloud. Por lo tanto, el flujo de información de identidad se dirige desde el dueño de los datos (es decir, la organización) hacia los consumidores (es decir, los proveedores de servicio), pasando a través del proveedor de cloud. Suponemos que la información de identidad está contenida


confianza directa
solicita información de identidad
Organización (Proveedor de identidad)
Proveedor de servicios
proporciona información de identidad
pertenece a
solicita un servicio
Empleado
confianza indirecta
solicita información de identidad
Proveedor de identidad en el cloud
Proveedor de servicios
proporciona información de identidad
externaliza la gestión de identidad
confianza directa
confianza directa
solicita un servicio
Organización
pertenece a
Empleado

(a) Federación de Identidad (b) Modelo BlindIdM

Figura A.5: Relaciones entre entidades en diferentes modelos

en atributos, consistentes en una tupla con metadatos y el valor asociado, del tipo $a = (a.metadatos, a.valor)$; por ejemplo, un atributo que representa el correo electrónico de un usuario es $a_{\mathsf{email}} = (\text{“email”}, \text{“dnunez@lcc.uma.es”})$. Esta estructura basada en atributos es muy habitual en sistemas de gestión de identidad, y en particular, en SAML 2.0, que usa el lenguaje XML para expresarlo.

En nuestro modelo, la información está cifrada desde un principio por la organización, usando su clave pública pk_H, antes de enviarla al cloud. Por tanto, por cada atributo de usuario se genera una versión cifrada; para ello, y por motivos de eficiencia, se hace uso de un método híbrido de cifrado incluyendo también un cifrado simétrico. El resultado es un atributo cifrado de la forma:

$$c_a = (a.metadatos, \mathsf{PRE.Enc}(pk_H, K_a), \mathsf{Sym.Enc}(K_a, a.valor)),$$

donde K_a es una clave simétrica aleatoria; puede verse que los metadatos se mantienen en claro para facilitar su gestión. SAML 2.0 tiene también mecanismos para representar atributos cifrados.

Una vez la información está almacenada en el cloud, en forma de atributos cifrados, el uso de un esquema de recifrado delegado permite al proveedor de identidad transformarlos en atributos cifrados que un proveedor de servicio puede descifrar con su clave privada sk_{SP}. Para ello, el proveedor de identidad necesita una clave de recifrado $rk_{H \to SP}$ generada por la organización, y que es única para cada proveedor de servicio SP. Cuando un proveedor de servicio pide un atributo de identidad, el proveedor de identidad recifra el correspondiente atributo cifrado usando la clave de recifrado asociada a dicho proveedor de servicio, y le devuelve otro atributo cifrado, todo ello integrado con los protocolos habituales de SAML 2.0.

Puede verse que el principal requisito de nuestro modelo, que es que el servicio de gestión de identidad pueda seguir proporcionándose sin necesidad de que el proveedor acceda a la información, se cumple gracias a la integración del recifrado delegado en el protocolo de gestión de identidad. Finalmente, es importante recalcar que nuestra propuesta no requiere ningún cambio en el protocolo SAML, por lo que podría integrarse fácilmente con sistemas reales.

A.6.2 Acceso Delegado a Datos Cifrados en Hadoop

Otra aplicación interesante del recifrado delegado puede encontrarse en el área del *Big Data*. Este término engloba a todas aquellas tecnologías que se utilizan para almacenar y procesar cantidades ingentes de información. El Big Data hace posible extraer información útil para empresas y gobiernos a partir de grandes repositorios de información. Sin embargo, aunque en algunos casos esta información puede ser sensible (por ejemplo, información médica o financiera), los sistemas de Big Data actuales siguen almacenándola en claro.

El más claro ejemplo es Apache Hadoop [20], uno de los sistemas de procesamiento Big Data más conocidos. Apache Hadoop permite el almacenamiento y procesamiento de grandes conjuntos de datos mediante *clusters* de computación. La estrategia de Hadoop consiste en dividir la carga de trabajo en partes y en distribuirla a través del cluster. Sin embargo, su arquitectura no se diseñó pensando en la seguridad. En este trabajo, describimos un sistema de control de acceso para clusters Hadoop usando recifrado delegado.

El sistema Hadoop

El sistema Hadoop implementa el paradigma de computación MapReduce, que permite distribuir la carga de trabajo en un cluster. Todas las operaciones en Hadoop se ejecutan por nodos del cluster. Estos nodos pueden tomar dos papeles principales: (i) el JobTracker, que distribuye la carga de trabajo en el cluster, asignando tareas individuales a los demás nodos, y (ii) los TaskTrackers, que simplemente ejecutan las tareas asignadas. En el paradigma MapReduce, cada porción de trabajo debe de ser independiente de las demás, de forma que se pueda aprovechar el potencial de la paralelización intensiva. Por este motivo, un trabajo en MapReduce está separado en dos fases, Mapeo (*Map*) y Reducción (*Reduce*), como puede verse en la Figura A.6a. En la fase de mapeo, los datos de entrada son divididos y procesados en paralelo por los nodos. Cada porción de trabajo es asignada a un TaskTracker libre. La salida de cada operación de mapeo es una lista de pares clave-valor, que es ordenada, particionada y almacenada localmente en los nodos que la generaron. Para cada partición, el JobTracker asigna una tarea de reducción a un TaskTracker libre, que debe leer remotamente los datos generados anteriormente, por lo que esta fase representa el mayor cuello de botella. Una vez estos datos intermedios son recuperados, cada tarea de reducción los combina. Finalmente, se recogen los resultados de todas las tareas de reducción. Una de las principales características de Hadoop es que aprovecha la localización de los datos para reducir las comunicaciones. Para ello, Hadoop asigna las tareas de mapeo a los nodos que estén más cercanos a los datos que deben procesar, idealmente, en los propios discos duros de las máquinas, lo que consigue que la fase de mapeo sea extraordinariamente eficiente.

Integrando el Recifrado Delegado con Hadoop

Nuestra solución consiste en integrar un control de acceso criptográfico basado en recifrado delegado con el sistema Hadoop. La idea principal es cifrar la información desde el origen, de forma que se almacene cifrada en todo momento. Los únicos puntos en los que se requiere la información sin cifrar son en las operaciones de mapeo y reducción. Por lo tanto, en nuestra solución, los nodos que deban procesar un bloque de información cifrada pueden pedir un acceso delegado a dicha información, actuando como consumidores. Para ello, se siguen una serie de protocolos que permiten la generación de claves de recifrado y el propio recifrado de los datos. La Figura A.6b muestra el funcionamiento simplificado de nuestra solución, que como puede apreciarse, es una extensión del flujo de datos normal en Hadoop.

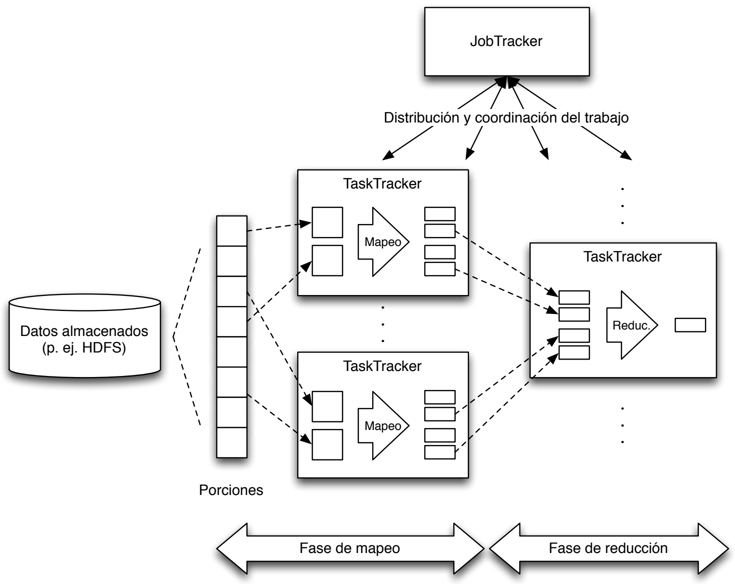
El ciclo de vida de los datos en nuestra solución se compone de las siguientes fases:

1. Fase de producción: durante esta fase, los datos son generados por diversas fuentes y almacenados cifrados con la clave pública del dueño para su procesamiento posterior.

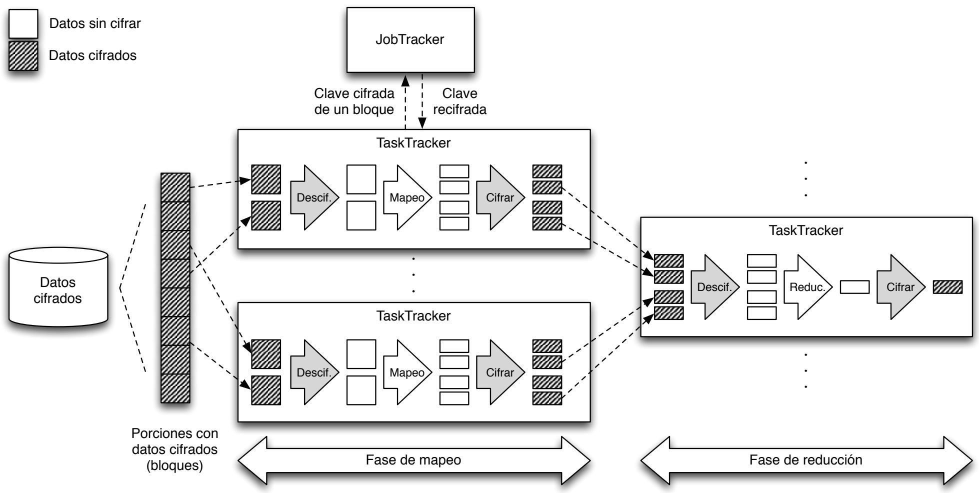
2. Fase de delegación: durante esta fase, el dueño de los datos genera las claves necesarias para permitir el acceso delegado. Después de esto, no necesita participar de nuevo.

3. Fase de consumo: esta fase ocurre cada vez que el cluster Hadoop ejecuta una tarea de procesamiento. Es en esta fase en donde los nodos del cluster leen la información cifrada.

Hemos omitido los detalles de cada fase en este resumen; la Sección 6.2.3 los explica en detalle.



(a) Funcionamiento de Hadoop



(b) Funcionamiento del sistema propuesto

Figura A.6: Integración de Hadoop con Recifrado Delegado

En cuanto a resultados experimentales, hemos replicado el funcionamiento principal, que ocurre en la fase de consumo. Este experimento se ejecutó en un cluster virtual de 17 máquinas virtuales, cada una con doble procesador y 4 GB de RAM. El experimento consistió en la ejecución del programa de pruebas WordCount, uno de los que se incluyen por defecto en Hadoop, y que simplemente cuenta la ocurrencia de palabras en un conjunto de ficheros. Para nuestro experimento, creamos un conjunto de datos con 1800 ficheros cifrados de 64 MB cada uno; en total, los datos de entrada contenían casi 29 mil millones de palabras y ocupaban 112 GB.

El tiempo de ejecución para las dos versiones de Hadoop comparadas (la original y nuestra solución) fueron de 1932.09 and 1960.74 segundos, respectivamente. Esto es, una diferencia de 28.74 segundos, lo que representa una sobrecarga relativa del 1.49 %. Creemos que una sobrecarga de esta magnitud es aceptable para muchas aplicaciones.

A.6.3 Criptosistema con descifrado en depósito

Las aplicaciones anteriores era instancias claras del escenario de compartición segura de datos. La tercera aplicación que proponemos, sin embargo, es algo diferente: un criptosistema de clave pública con descifrado en depósito (en inglés, *escrowed decryption*). La innovación de esta propuesta es el uso del recifrado delegado para construir la funcionalidad del descifrado en depósito, que permite a las autoridades descifrar mensajes sin recurrir a la obtención de claves privadas, que es el método habitual de los criptosistemas con claves en depósito (en inglés, *key escrow*), si no a través del recifrado.

La motivación de esta aplicación es un tema controvertido en la actualidad. Hay una gran preocupación por parte de gobiernos y autoridades respecto a la impunidad que se origina de la privacidad y confidencialidad en las comunicaciones digitales, que en ocasiones sirve para encubrir actividades ilegales y amenazas para la seguridad. La controversia surge del planteamiento de si romper está confidencialidad es un método legítimo o no. Uno de las principales argumentos en contra es la falta de mecanismos de rendición de cuentas, que permitan fiscalizar el uso (o mejor dicho, el abuso) por parte de las autoridades.

A este respecto, Liu, Ryan y Chen proponen en [181] un criptosistema fiscalizable con descifrado en depósito (en inglés, *Accountable Escrowed Encryption*), que permite a las autoridades descifrar textos cifrados sospechosos, pero al mismo tiempo, les obliga a rendir cuentas. Su propuesta consiste en un esquema de cifrado similar a ElGamal, pero que incluye una funcionalidad de descifrado en depósito. Para usarla, las autoridades deber seguir un protocolo que involucra a un conjunto de entidades confiables llamadas custodios; solo si todos los custodios colaboran, las autoridades son capaces de descifrar el texto cifrado. De esta forma no es necesario poner en depósito las claves privadas de los usuarios, si no la capacidad de descifrar. Para conseguir la rendición de cuentas de las autoridades, los custodios registran públicamente todas las operaciones de descifrado en depósito que realicen, de forma que la sociedad pueda actuar en consecuencia a través de cauces democráticos, demandando, por ejemplo, "menos descifrados" por parte de las autoridades.

Aquí proponemos una construcción alternativa, que resuelve algunos problemas en cuanto a seguridad y eficiencia. Uno de los problemas es que una colusión de custodios puede descifrar cualquier mensaje si tienen acceso a un texto cifrado completo; por tanto, es necesario asumir que al menos uno de los custodios no va a participar en una colusión. En nuestra propuesta, por contra, los custodios no pueden descifrar el mensaje, ya que su única capacidad es recifrar textos cifrados.

Otro problema es la eficiencia. El protocolo propuesto en [181] para el descifrado en depósito está compuesto de dos rondas síncronas, y en cada una participan todos los custodios. Es decir, por cada petición, las autoridades deben comunicarse en una primera ronda con todos los custodios, y, solo cuando todos han respondido, pueden continuar con una segunda ronda de interacciones. En nuestra propuesta, solo hay una ronda.

El sistema propuesto

Nuestra propuesta es similar en esencia, pero hace uso del recifrado delegado para construir la funcionalidad de descifrado en depósito, de forma que, ante una petición de las autoridades, los custodios recifran distribuidamente un texto cifrado sospechoso, que finalmente puede ser descifrado por las autoridades. Para conseguir el recifrado distribuido, es necesario "dividir" la clave de recifrado entre los custodios, de forma similar a un criptosistema con umbral [143].

Nuestra solución está formada por los siguientes algoritmos, que describimos brevemente:

- $\mathsf{Setup}(\lambda) \rightarrow params$. Los parámetros globales incluyen el emparejamiento criptográfico e y los grupos cíclicos asociados $\mathbb{G}$ y $\mathbb{G}_T$. Además se genera un par de claves pública y privada para las autoridades, $pk_{EA} = g^a$ y $sk_{EA} = a$.
- $\mathsf{KeyGen}(pk_{EA}) \rightarrow (pk, sk, \{\kappa_i\}_{i=1}^n)$. La generación de claves de usuario consiste en un protocolo entre el usuario y una entidad de certificación. Obviaremos en este resumen los detalles del protocolo. El resultado incluye las clave públicas y privadas del usuario, $pk = (g^{su}, Z^{sv})$ y $sk = g^{v/u}$, así como una división en n partes de la clave de recifrado hacia pk_{EA}, que denomina-

remos κ_i. El protocolo garantiza que solo el usuario conoce la clave privada y que las partes κ_i son correctas.

- $\mathsf{Enc}(pk, m) \rightarrow CT$. Para cifrar un mensaje m con la clave pública $pk = (g^{su}, Z^{sv})$, se elige un $r \in \mathbb{Z}_q$ aleatorio y se devuelve el siguiente texto cifrado:
$$CT = ((g^{su})^r, (Z^{sv})^r \cdot m) = (g^{sur}, Z^{svr} \cdot m)$$

- $\mathsf{Dec}(sk, CT) \rightarrow m$. Para descifrar un texto cifrado $CT = (CT_1, CT_2) = (g^{sur}, Z^{svr} \cdot m)$ se hace el siguiente cómputo:
$$m = \frac{CT_2}{e(CT_1, sk)} = \frac{Z^{svr} \cdot m}{e(g^{sur}, g^{v/u})}$$

- $\mathsf{ShareReEnc}(\kappa_i, CT) \rightarrow \rho_i$. Un custodio recifra parcialmente texto cifrado $CT = (CT_1, CT_2) = (g^{sur}, Z^{svr} \cdot m)$ al hacer el cómputo $\rho_i = e(\kappa_i, g^{sur})$.

- $\mathsf{Comb}(sk_{EA}, CT, \{\rho_i\}_{i=1}^n) \rightarrow m$. Las autoridades pueden descifrar un texto cifrado $CT = (CT_1, CT_2)$ si obtienen los recifrados parciales $\{\rho_i\}_{i=1}^n$ de todos los custodios y los combinan de forma que:
$$m = \frac{CT_2}{(\prod_{i=1}^{n} \rho_i)^{1/sk_{EA}}}$$

En la Sección 6.3.3 mostramos que la funcionalidad es correcta y que el esquema cumple con la noción de seguridad IND-CPA para cifrados de clave pública. Además, hacemos una comparación con el sistema de Liu, Ryan y Chen [181], en el que mostramos que las mejoras en cuanto a funcionalidad que proporciona nuestro sistema tienen un coste computacional algo mayor, aunque es importante considerar la ganancia en costes de comunicaciones (una ronda frente a dos rondas síncronas).

A.7 Conclusiones y trabajo futuro

Esta tesis se ha centrado, principalmente, en el estudio del recifrado delegado. La motivación inicial surge de la problemática de la compartición segura de datos, una funcionalidad especialmente relevante hoy en día, ya que es muy habitual tener información almacenada en la nube. El recifrado delegado constituye una elegante forma de mantener los datos cifrados en todo momento, permitiendo al proveedor de almacenamiento compartir la información sin necesidad de acceder a los datos.

En esta tesis describimos diversos retos de investigación asociados a los esquemas de recifrado delegado, que surgen a distintos niveles de abstracción, y contribuimos a la resolución de algunos de ellos.

A nivel más fundamental, era necesario contribuir a un mejor entendimiento de este tipo de criptosistemas. Para ello, realizamos un análisis exhaustivo de la literatura científica, tanto de los esquemas de recifrado delegado, como de sus aplicaciones. Una contribución clave de esta tesis es la identificación de una familia paramétrica de modelos de ataque para el recifrado delegado, que considera separadamente la capacidad de descifrar de la de recifrar. Esto nos permite definir, a su vez, una familia de nociones de seguridad, cuyas relaciones analizamos. De especial interés es el estudio de las separaciones entre nociones, que explotamos mediante el análisis de una propiedad deseable en el recifrado delegado: la privacidad de las claves de recifrado. Descubrimos que los esquemas que no cumplen esta propiedad no pueden lograr una noción de seguridad significativa, y lo ilustramos al mostrar un ataque a un esquema reciente.

Una parte sustancial de esta tesis está dedicada a la seguridad de los esquemas de recifrado delegado. Aparte de la investigación sobre modelos de ataque, que se sitúa en el plano de los fundamentos del concepto de seguridad, destacamos el estudio sobre transformaciones genéricas para incrementar la seguridad, más cercano a las construcciones concretas. A este respecto definimos una extensión de la transformación Fujisaki-Okamoto e identificamos las condiciones que permiten aplicarla. La aplicación de esta y otras transformaciones genéricas no es obvia, ya que la funcionalidad de recifrado interfiere con los mecanismos de comprobación de validez que son inherentes a estas transformaciones. Como prueba de ello, mostramos hasta 12 esquemas de recifrado delegado que son incorrectos a causa de esta problemática.

Otra contribución de esta tesis es NTRUReEncrypt, un esquema de recifrado delegado altamente eficiente, basado en el criptosistema NTRU. El desempeño de este esquema es altamente superior a otros esquemas propuestos, hasta por un orden de magnitud. Asimismo, describimos una variante cuya seguridad puede reducirse a problemas intratables sobre retículos.

Finalmente, un principio que ha guiado esta tesis desde el principio es el enlace de la teoría con el mundo real. A este respecto, hemos procurado que las aplicaciones no quedaran definidas a un nivel abstracto, si no que que se integraran con sistemas y tecnologías del mundo real, en la medida de lo posible. Este es el caso tanto de la aplicación de gestión de identidad a ciegas, que integramos con el estándar de gestión de identidad SAML 2.0, como de la aplicación de Big Data, que implementamos en el sistema Apache Hadoop. Proponemos también un criptosistema con descifrado en depósito, una solución técnica que se enmarca en el debate actual acerca de la dicotomía entre la privacidad individual y la seguridad de la sociedad.

Como trabajo futuro, hay diversas líneas de actuación con problemáticas interesantes. Por ejemplo, no descartamos que el estudio de relaciones entre los modelos de ataque y nociones de seguridad puede ampliarse con nuevas implicaciones y separaciones. En relación con los esquemas basados en NTRU, hay diversas áreas de trabajo futuro. Las más importantes son conseguir seguridad CCA y definir un esquema unidireccional. La primera cuestión fue lo que motivó inicialmente el trabajo en transformaciones genéricas, aunque finalmente no pudo beneficiarse de ellas. La segunda cuestión se trata de forma preliminar en la Sección 5.1.6 y requiere trabajo adicional.

Respecto a la contribución sobre transformaciones genéricas, una línea futura de trabajo consiste en estimar cuantitativamente el nivel de seguridad obtenido, de forma que se puedan inferir conjuntos de parámetros concretos. Aparte de esto, un problema ambicioso es desarrollar transformaciones genéricas en el modelo estándar.

Finalmente, también hay posibles mejoras en el ámbito de las aplicaciones propuestas. Algunas de estas mejoras son puramente técnicas: por ejemplo, el modelo BlindIdM podría refinarse mediante la integración de especificaciones como SCIM (System for Cross-domain Identity Management) [193], un estándar abierto diseñado para simplificar la gestión de identidad en servicios y aplicaciones del cloud y facilitar el intercambio de información de identidad. En el caso de la aplicación de Big Data, hay mucho potencial si la solución se integra con HDFS, el sistema de ficheros nativo de Hadoop. Respecto al criptosistema con descifrado delegado, sería conveniente lograr que el esquema consiguiera seguridad CCA, así como reducir la confianza necesaria en la entidad de certificación.

Bibliography

[1] John Hermans and Mike Chung. KPMG's 2010 Cloud Computing Survey. Technical report, KPMG, 2010.

[2] Jay Heiser and Mark Nicolett. Assessing the security risks of cloud computing. Technical report, Gartner Inc., 2008.

[3] Raluca Ada Popa. *Building practical systems that compute on encrypted data*. PhD thesis, Massachusetts Institute of Technology, 2014.

[4] Christoph Bösch, Pieter Hartel, Willem Jonker, and Andreas Peter. A survey of provably secure searchable encryption. *ACM Computing Surveys (CSUR)*, 47(2):18, 2014.

[5] Craig Gentry. Fully homomorphic encryption using ideal lattices. In *Proceedings of the Forty-first Annual ACM Symposium on Theory of Computing*, STOC '09, pages 169–178, New York, NY, USA, 2009. ACM.

[6] Brian Hayes. Alice and Bob in cipherspace. *American Scientist*, 100(5):1, 2012.

[7] Y. Chen and R. Sion. On securing untrusted clouds with cryptography. In *Proc. 9th annual ACM workshop on Privacy in the electronic society*, pages 109–114. ACM, 2010.

[8] U.S. Congress. Health insurance portability and accountability act, 1996.

[9] Hadoop on Google Cloud Platform. https://cloud.google.com/solutions/hadoop/.

[10] MapR in the Cloud. http://www.mapr.com/products/mapr-cloud.

[11] G. Ateniese, K. Fu, M. Green, and S. Hohenberger. Improved proxy re-encryption schemes with applications to secure distributed storage. *ACM Transactions on Information and System Security*, 9(1):1–30, 2006.

[12] B. Libert and D. Vergnaud. Unidirectional chosen-ciphertext secure proxy re-encryption. *Information Theory, IEEE Transactions on*, 57(3):1786–1802, 2011.

[13] M. Green and G. Ateniese. Identity-based proxy re-encryption. In *Applied Cryptography and Network Security*, pages 288–306. Springer, 2007.

[14] C.K. Chu and W.G. Tzeng. Identity-based proxy re-encryption without random oracles. *Information Security*, pages 189–202, 2007.

[15] Mihir Bellare, Dennis Hofheinz, and Eike Kiltz. Subtleties in the definition of IND-CCA: When and how should challenge decryption be disallowed? *Journal of Cryptology*, 28(1):29–48, 2015.

[16] Jeffrey Hoffstein, Jill Pipher, and Joseph H Silverman. NTRU: A ring-based public key cryptosystem. In *Algorithmic number theory*, pages 267–288. Springer, 1998.

[17] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. *Journal of cryptology*, 26(1):80–101, 2013.

[18] Tatsuaki Okamoto and David Pointcheval. REACT: Rapid enhanced-security asymmetric cryptosystem transform. In *Topics in Cryptology—CT-RSA 2001*, pages 159–174. Springer, 2001.

[19] OASIS Security Services TC. Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0, 2005.

[20] Apache Hadoop. http://hadoop.apache.org/.

[21] Jonathan Katz and Yehuda Lindell. *Introduction to modern cryptography.* CRC Press, 2014.

[22] Joan Daemen and Vincent Rijmen. *The design of Rijndael: AES-the advanced encryption standard.* Springer Science & Business Media, 2013.

[23] Neal Koblitz and Alfred J Menezes. Another look at "provable security". *Journal of Cryptology*, 20(1):3–37, 2007.

[24] Mihir Bellare, Anand Desai, David Pointcheval, and Phillip Rogaway. Relations among notions of security for public-key encryption schemes. In *Advances in Cryptology—CRYPTO'98*, pages 26–45. Springer, 1998.

[25] Henk C.A. Van Tilborg and Sushil Jajodia. *Encyclopedia of cryptography and security.* Springer Science & Business Media, 2011.

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

[27] Ueli Maurer and Stefan Wolf. Lower bounds on generic algorithms in groups. In *Advances in Cryptology—EUROCRYPT'98*, pages 72–84. Springer, 1998.

[28] Tatsuaki Okamoto and David Pointcheval. The gap-problems: A new class of problems for the security of cryptographic schemes. In *Public Key Cryptography*, pages 104–118. Springer, 2001.

[29] Jean-Sébastien Coron, Helena Handschuh, Marc Joye, Pascal Paillier, David Pointcheval, and Christophe Tymen. GEM: A generic chosen-ciphertext secure encryption method. In *Topics in Cryptology—CT-RSA 2002*, pages 263–276. Springer, 2002.

[30] Yevgeniy Dodis and Aleksandr Yampolskiy. A verifiable random function with short proofs and keys. In *Public Key Cryptography-PKC 2005*, pages 416–431. Springer, 2005.

[31] Dan Boneh, Xavier Boyen, and Eu-Jin Goh. Hierarchical identity based encryption with constant size ciphertext. In *Advances in Cryptology–EUROCRYPT 2005*, pages 440–456. Springer, 2005.

[32] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM (JACM)*, 56(6):34, 2009.

[33] Richard Lindner and Chris Peikert. Better key sizes (and attacks) for LWE-based encryption. In *Topics in Cryptology–CT-RSA 2011*, pages 319–339. Springer, 2011.

[34] Chris Peikert. Public-key cryptosystems from the worst-case shortest vector problem. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 333–342. ACM, 2009.

[35] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 197–206. ACM, 2008.

[36] Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. *SIAM Journal on Computing*, 43(2):831–871, 2014.

[37] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. *Journal of the ACM (JACM)*, 60(6):43, 2013.

[38] Miklós Ajtai. Generating hard instances of lattice problems. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 99–108. ACM, 1996.

[39] Oded Goldreich, Shafi Goldwasser, and Shai Halevi. Collision-free hashing from lattice problems. In *Electronic Colloquium on Computational Complexity (ECCC)*, volume 3, pages 236–241, 1996.

[40] Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In *Advances in Cryptology–EUROCRYPT 2012*, pages 700–718. Springer, 2012.

[41] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proceedings of the 1st ACM conference on Computer and communications security*, pages 62–73. ACM, 1993.

[42] Masahiro Mambo and Eiji Okamoto. Proxy cryptosystems: Delegation of the power to decrypt ciphertexts. *IEICE transactions on fundamentals of electronics, Communications and computer sciences*, 80(1):54–63, 1997.

[43] M. Blaze, G. Bleumer, and M. Strauss. Divertible protocols and atomic proxy cryptography. *Advances in Cryptology—EUROCRYPT'98*, pages 127–144, 1998.

[44] Jun Shao. Bibliography on proxy re-cryptography. `http://ndc.zjgsu.edu.cn/~jshao/prcbib.htm`, 2015.

[45] R. Canetti and S. Hohenberger. Chosen-ciphertext secure proxy re-encryption. In *Proceedings of the 14th ACM conference on Computer and communications security*, pages 185–194. ACM, 2007.

[46] Keita Xagawa and Keisuke Tanaka. Proxy re-encryption based on learning with errors. In *Proceedings of the 2010 Symposium on Cryptography and Information Security (SCIS 2010)*, 2010.

[47] Yoshinori Aono, Xavier Boyen, Le Trieu Phong, and Lihua Wang. Key-private proxy re-encryption under LWE. In *Progress in Cryptology–INDOCRYPT 2013*, pages 1–18. Springer, 2013.

[48] Elena Kirshanova. Proxy re-encryption from lattices. In *Public-Key Cryptography–PKC 2014*, pages 77–94. Springer, 2014.

[49] David Nuñez, Isaac Agudo, and Javier Lopez. NTRUReEncrypt: An efficient proxy re-encryption scheme based on NTRU. In *Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security*, ASIA CCS '15, pages 179–189, New York, NY, USA, 2015. ACM.

[50] R.H. Deng, J. Weng, S. Liu, and K. Chen. Chosen-ciphertext secure proxy re-encryption without pairings. In *Proceedings of the 7th International Conference on Cryptology and Network Security*, pages 1–17. Springer-Verlag, 2008.

[51] Ran Canetti, Shai Halevi, and Jonathan Katz. Chosen-ciphertext security from identity-based encryption. In *Advances in Cryptology-Eurocrypt 2004*, pages 207–222. Springer, 2004.

[52] Jiang Zhang, Zhenfeng Zhang, and Yu Chen. PRE: Stronger security notions and efficient construction with non-interactive opening. *Theoretical Computer Science*, 542:1–16, 2014.

[53] Lihua Wang, Licheng Wang, Masahiro Mambo, and Eiji Okamoto. New identity-based proxy re-encryption schemes to prevent collusion attacks. In *Pairing-Based Cryptography-Pairing 2010*, pages 327–346. Springer, 2010.

[54] Jian Weng, Robert H Deng, Shengli Liu, and Kefei Chen. Chosen-ciphertext secure bidirectional proxy re-encryption schemes without pairings. *Information Sciences*, 180(24):5077–5089, 2010.

[55] Jian Weng, Robert H Deng, Xuhua Ding, Cheng-Kang Chu, and Junzuo Lai. Conditional proxy re-encryption secure against chosen-ciphertext attack. In *Proceedings of the 4th International Symposium on Information, Computer, and Communications Security*, pages 322–332. ACM, 2009.

[56] G. Ateniese, K. Fu, M. Green, and S. Hohenberger. Improved proxy re-encryption schemes with applications to secure distributed storage. In *Proceedings of the 12th Annual Network and Distributed System Security Symposium*, pages 29–44, 2005.

[57] The JHU-MIT Proxy Re-cryptography Library. `http://spar.isi.jhu.edu/~mgreen/prl/`.

[58] Dan Boneh and Matt Franklin. Identity-based encryption from the weil pairing. In *Advances in Cryptology—CRYPTO 2001*, pages 213–229. Springer, 2001.

[59] Woo Kwon Koo, Jung Yeon Hwang, and Dong Hoon Lee. Security vulnerability in a non-interactive ID-based proxy re-encryption scheme. *Information Processing Letters*, 109(23):1260–1262, 2009.

[60] Craig Gentry and Alice Silverberg. Hierarchical id-based cryptography. In *Advances in cryptology—ASIACRYPT 2002*, pages 548–566. Springer, 2002.

[61] Brent Waters. Efficient identity-based encryption without random oracles. In *Advances in Cryptology–EUROCRYPT 2005*, pages 114–127. Springer, 2005.

[62] J. Shao and Z. Cao. CCA-secure proxy re-encryption without pairings. In *Proceedings of the 12th International Conference on Practice and Theory in Public Key Cryptography: PKC'09*, pages 357–376. Springer-Verlag, 2009.

[63] Toshihiko Matsuo. Proxy re-encryption systems for identity-based encryption. In *Pairing-Based Cryptography–Pairing 2007*, pages 247–267. Springer, 2007.

[64] G. Ateniese, K. Benson, and S. Hohenberger. Key-private proxy re-encryption. *Topics in Cryptology–CT-RSA 2009*, pages 279–294, 2009.

[65] Mihir Bellare, Alexandra Boldyreva, Anand Desai, and David Pointcheval. Key-privacy in public-key encryption. In *Advances in Cryptology—ASIACRYPT 2001*, pages 566–582. Springer, 2001.

[66] Claus-Peter Schnorr. Efficient signature generation by smart cards. *Journal of cryptology*, 4(3):161–174, 1991.

[67] B. Libert and D. Vergnaud. Unidirectional chosen-ciphertext secure proxy re-encryption. *Public Key Cryptography–PKC 2008*, pages 360–379, 2008.

[68] J. W. Seo, D. H. Yum, and P. J. Lee. Comments on "unidirectional chosen-ciphertext secure proxy re-encryption". *Information Theory, IEEE Transactions on*, PP(99):1, 2012.

[69] Kunwar Singh, C Pandu Rangan, and AK Banerjee. Cryptanalysis of unidirectional proxy re-encryption scheme. In *Information and Communication Technology*, pages 564–575. Springer, 2014.

[70] Ryo Nishimaki and Keita Xagawa. Key-private proxy re-encryption from lattices, revisited. *IEICE TRANSACTIONS on Fundamentals of Electronics, Communications and Computer Sciences*, 98(1):100–116, 2015.

[71] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In *Advances in Cryptology—CRYPTO'99*, pages 537–554. Springer, 1999.

[72] Phong Nguyen and David Pointcheval. Analysis and improvements of ntru encryption paddings. *Advances in Cryptology—CRYPTO 2002*, pages 149–170, 2002.

[73] Damien Stehlé and Ron Steinfeld. Making NTRU as secure as worst-case problems over ideal lattices. In *Advances in Cryptology–EUROCRYPT 2011*, pages 27–47. Springer, 2011.

[74] E. Barker, L. Chen, A. Roginsky, and M. Smid. Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography. NIST special publication 800-56A (Revision 2), NIST, May 2013.

[75] Angelo De Caro and Vincenzo Iovino. jPBC: Java pairing based cryptography. In *Proceedings of the 16th IEEE Symposium on Computers and Communications, ISCC 2011*, pages 850–855. IEEE, 2011.

[76] Steven D Galbraith, Kenneth G Paterson, and Nigel P Smart. Pairings for cryptographers. *Discrete Applied Mathematics*, 156(16):3113–3121, 2008.

[77] Elaine Barker, William Barker, William Burr, William Polk, and Miles Smid. Recommendation for key management — part 1: General. Technical report, 2005.

[78] Java implementation of NTRUEncrypt and NTRUSign. `http://tbuktu.github.io/ntru/`.

[79] W Whyte, N Howgrave-Graham, J Hoffstein, J Pipher, JH Silverman, and P Hirschhorn. IEEE P1363.1: Draft standard for public-key cryptographic techniques based on hard problems over lattices. Technical report, IEEE, 2008.

[80] Shucheng Yu, Cong Wang, Kui Ren, and Wenjing Lou. Achieving secure, scalable, and fine-grained data access control in cloud computing. In *INFOCOM, 2010 Proceedings IEEE*, pages 1–9. Ieee, 2010.

[81] Namje Park. Secure data access control scheme using type-based re-encryption in cloud environment. In *Semantic Methods for Knowledge Management and Communication*, pages 319–327. Springer, 2011.

[82] Guojun Wang, Qin Liu, Jie Wu, and Minyi Guo. Hierarchical attribute-based encryption and scalable user revocation for sharing data in cloud servers. *computers & security*, 30(5):320–331, 2011.

[83] Junbeom Hur. Improving security and efficiency in attribute-based data sharing. *Knowledge and Data Engineering, IEEE Transactions on*, 25(10):2271–2282, 2013.

[84] Piotr K Tysowski and M Anwarul Hasan. Hybrid attribute-and re-encryption-based key management for secure and scalable mobile applications in clouds. *Cloud Computing, IEEE Transactions on*, 1(2):172–186, 2013.

[85] Hsiao-Ying Lin and W-G Tzeng. A secure erasure code-based cloud storage system with secure data forwarding. *Parallel and Distributed Systems, IEEE Transactions on*, 23(6):995–1003, 2012.

[86] Weiwei Jia, Haojin Zhu, Zhenfu Cao, Lifei Wei, and Xiaodong Lin. Sdsm: a secure data service mechanism in mobile cloud computing. In *Computer Communications Workshops (INFOCOM WKSHPS), 2011 IEEE Conference on*, pages 1060–1065. IEEE, 2011.

[87] Qin Liu, Guojun Wang, and Jie Wu. Time-based proxy re-encryption scheme for secure data sharing in a cloud environment. *Information Sciences*, 258:355–370, 2014.

[88] Yanjiang Yang and Youcheng Zhang. A generic scheme for secure data sharing in cloud. In *Parallel Processing Workshops (ICPPW), 2011 40th International Conference on*, pages 145–153. IEEE, 2011.

[89] Huijun Xiong, Xinwen Zhang, Danfeng Yao, Xiaoxin Wu, and Yonggang Wen. Towards end-to-end secure content storage and delivery with public cloud. In *Proceedings of the second ACM conference on Data and Application Security and Privacy*, pages 257–266. ACM, 2012.

[90] Jinguang Han, Willy Susilo, and Yi Mu. Identity-based data storage in cloud computing. *Future Generation Computer Systems*, 29(3):673–681, 2013.

[91] Hsiao-Ying Lin, John Kubiatowicz, and Wen-Guey Tzeng. A secure fine-grained access control mechanism for networked storage systems. In *Software Security and Reliability (SERE), 2012 IEEE Sixth International Conference on*, pages 225–234. IEEE, 2012.

[92] David Nuñez and Isaac Agudo. BlindIdM: A privacy-preserving approach for identity management as a service. *International Journal of Information Security*, 13(2):199–215, 2014.

[93] Bernd Zwattendorfer, Daniel Slamanig, Klaus Stranacher, and Felix Hörandner. A federated cloud identity broker-model for enhanced privacy via proxy re-encryption. In *Communications and Multimedia Security*, pages 92–103. Springer, 2014.

[94] Zeeshan Pervez, Ammar Ahmad Awan, Asad Masood Khattak, Sungyoung Lee, and Eui-Nam Huh. Privacy-aware searching with oblivious term matching for cloud storage. *The Journal of Supercomputing*, 63(2):538–560, 2013.

[95] Lei Xu, Weidong Shi, and Taeweon Suh. Pfc: Privacy preserving fpga cloud-a case study of mapreduce. In *Cloud Computing (CLOUD), 2014 IEEE 7th International Conference on*, pages 280–287. IEEE, 2014.

[96] Peishun Wang, Yi Mu, Willy Susilo, and Jun Yan. Privacy preserving protocol for service aggregation in cloud computing. *Software: Practice and Experience*, 42(4):467–483, 2012.

[97] Chuanyi Liu, Xiaojian Liu, and Lei Wan. Policy-based de-duplication in secure cloud storage. In *Trustworthy Computing and Services*, pages 250–262. Springer, 2013.

[98] Jun Shao, Rongxing Lu, and Xiaodong Lin. Fine: A fine-grained privacy-preserving location-based service framework for mobile devices. In *INFOCOM, 2014 Proceedings IEEE*, pages 244–252. IEEE, 2014.

[99] Rolf H Weber and Dominic Nicolaj Staiger. Cloud computing: a cluster of complex liability issues. *Web Journal of Current Legal Issues*, 20(1), 2014.

[100] Andreas Gauger. Amazon's "mountain of margin" in aws. `https://blog.profitbricks.com/amazons-mountain-of-margin-in-aws/`.

[101] Gelareh Taban, Alvaro A Cárdenas, and Virgil D Gligor. Towards a secure and interoperable DRM architecture. In *Proceedings of the ACM workshop on Digital rights management*, pages 69–78. ACM, 2006.

[102] Sangho Lee, Heejin Park, and Jong Kim. A secure and mutual-profitable drm interoperability scheme. In *Computers and Communications (ISCC), 2010 IEEE Symposium on*, pages 75–80. IEEE, 2010.

[103] Qin Qiu, Zhi Tang, and Yinyan Yu. A decentralized authorization scheme for drm in p2p file-sharing systems. In *Consumer Communications and Networking Conference (CCNC), 2011 IEEE*, pages 136–140. IEEE, 2011.

[104] Nakul Joshi and Ronald Petrlic. Towards practical privacy-preserving digital rights management for cloud computing. In *Consumer Communications*

and Networking Conference (CCNC), 2013 IEEE, pages 265–270. IEEE, 2013.

[105] Yun-Peng Chiu, Chin-Laung Lei, and Chun-Ying Huang. Secure multicast using proxy encryption. In *Information and Communications Security*, pages 280–290. Springer, 2005.

[106] Ritesh Mukherjee and J William Atwood. Scalable solutions for secure group communications. *Computer Networks*, 51(12):3525–3548, 2007.

[107] Chun-Ying Huang, Yun-Peng Chiu, Kuan-Ta Chen, and Chin-Laung Lei. Secure multicast in dynamic environments. *Computer Networks*, 51(10):2805–2817, 2007.

[108] Yiliang Han, Xiaolin Gui, Xuguang Wu, and Xiaoyuan Yang. Proxy encryption based secure multicast in wireless mesh networks. *Journal of network and computer applications*, 34(2):469–477, 2011.

[109] Thomas S Heydt-Benjamin, Hee-Jin Chae, Benessa Defend, and Kevin Fu. Privacy for public transportation. In *Privacy Enhancing Technologies*, pages 1–19. Springer, 2006.

[110] Florian Kerschbaum and Alessandro Sorniotti. Rfid-based supply chain partner authentication and key agreement. In *Proceedings of the second ACM conference on Wireless network security*, pages 41–50. ACM, 2009.

[111] Qiang Yan, Yingjiu Li, Robert H Deng, et al. Anti-tracking in rfid discovery service for dynamic supply chain systems. *International Journal of RFID Security and Cryptography (IJRFIDSC)*, 1(1/2):25–35, 2012.

[112] Jun Liu, Xiaoyan Hong, Qunwei Zheng, and Lei Tang. Privacy-preserving quick authentication in fast roaming networks. In *Local Computer Networks, Proceedings 2006 31st IEEE Conference on*, pages 975–982. IEEE, 2006.

[113] Tat Wing Chim, Siu-Ming Yiu, Lucas CK Hui, and Victor OK Li. Mlas: Multiple level authentication scheme for vanets. *Ad Hoc Networks*, 10(7):1445–1456, 2012.

[114] Changyu Dong and Naranker Dulay. Longitude: a privacy-preserving location sharing protocol for mobile applications. In *Trust Management V*, pages 133–148. Springer, 2011.

[115] Matthew M Lucas and Nikita Borisov. Flybynight: mitigating the privacy risks of social networking. In *Proceedings of the 7th ACM workshop on Privacy in the electronic society*, pages 1–8. ACM, 2008.

[116] Junzhou Luo, Xiaogang Wang, and Ming Yang. A resilient p2p anonymous routing approach employing collaboration scheme. *J. UCS*, 15(9):1797–1811, 2009.

[117] Mihaela Ion, Giovanni Russello, and Bruno Crispo. Design and implementation of a confidentiality and access control solution for publish/subscribe systems. *Computer networks*, 56(7):2014–2037, 2012.

[118] Hwajeong Seo and Howon Kim. Zigbee security for visitors in home automation using attribute based proxy re-encryption. In *Consumer Electronics (ISCE), 2011 IEEE 15th International Symposium on*, pages 304–307. IEEE, 2011.

[119] Victor Shoup. *Why chosen ciphertext security matters.* IBM TJ Watson Research Center, 1998.

[120] David Nuñez, Isaac Agudo, and Javier Lopez. A parametric family of attack models for proxy re-encryption. In *Proceedings of the 28th IEEE Computer Security Foundations Symposium*, CSF'15, pages 290–301. IEEE Computer Society, 2015.

[121] Jun Shao, Peng Liu, and Jian Weng. CCA-Secure PRE scheme without public verifiability. *IACR Cryptology ePrint Archive*, 2010:357, 2010.

[122] Damien Stehlé and Ron Steinfeld. Making NTRUEncrypt and NTRUSign as secure as standard worst-case problems over ideal lattices. *IACR Cryptology ePrint Archive*, 2013:4, 2013.

[123] ANSI X9.98: Lattice-based polynomial public key establishment algorithm for the financial services industry. Technical report, ANSI, 2010.

[124] Jens Hermans, Frederik Vercauteren, and Bart Preneel. Speed records for ntru. In *Topics in Cryptology-CT-RSA 2010*, pages 73–88. Springer, 2010.

[125] Daniel V Bailey, Daniel Coffin, Adam Elbirt, Joseph H Silverman, and Adam D Woodbury. NTRU in constrained devices. In *Cryptographic Hardware and Embedded Systems—CHES 2001*, pages 262–272. Springer, 2001.

[126] Angelo De Caro. Java Lattice Based Cryptography Library (jLBC). `http://gas.dia.unisa.it/projects/jlbc/`.

[127] Adriana López-Alt, Eran Tromer, and Vinod Vaikuntanathan. On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption. In *Proceedings of the 44th symposium on Theory of Computing*, pages 1219–1234. ACM, 2012.

[128] David Cash, Matthew Green, and Susan Hohenberger. New definitions and separations for circular security. In *Public Key Cryptography–PKC 2012*, pages 540–557. Springer, 2012.

[129] Takashi Kitagawa, Peng Yang, Goichiro Hanaoka, Rui Zhang, Hajime Watanabe, Kanta Matsuura, and Hideki Imai. Generic transforms to acquire CCA-security for identity based encryption: The cases of fopkc and react. In *Information Security and Privacy*, pages 348–359. Springer, 2006.

[130] Zhaohui Cheng, Liqun Chen, Li Ling, and Richard Comley. General and efficient certificateless public key encryption constructions. In *Pairing-Based Cryptography–Pairing 2007*, pages 83–107. Springer, 2007.

[131] Yang Lu, Jiguo Li, and Junmo Xiao. Applying the Fujisaki-Okamoto conversion to certificate-based encryption. In *Electronic Commerce and Security, 2008 International Symposium on*, pages 296–300. IEEE, 2008.

[132] Jun Shao, Zhenfu Cao, and Peng Liu. SCCR: a generic approach to simultaneously achieve cca security and collusion-resistance in proxy re-encryption. *Security and Communication Networks*, 4(2):122–135, 2011.

[133] Goichiro Hanaoka, Yutaka Kawai, Noboru Kunihiro, Takahiro Matsuda, Jian Weng, Rui Zhang, and Yunlei Zhao. Generic construction of chosen ciphertext secure proxy re-encryption. In *Topics in Cryptology–CT-RSA 2012*, pages 349–364. Springer, 2012.

[134] David Galindo, Sebastià Martín, Paz Morillo, and Jorge L Villar. Fujisaki-Okamoto hybrid encryption revisited. *International Journal of Information Security*, 4(4):228–241, 2005.

[135] Sherman SM Chow, Jian Weng, Yanjiang Yang, and Robert H Deng. Efficient unidirectional proxy re-encryption. In *Progress in Cryptology–AFRICACRYPT 2010*, pages 316–332. Springer, 2010.

[136] Jian Weng, Yanjiang Yang, Qiang Tang, Robert H Deng, and Feng Bao. Efficient conditional proxy re-encryption with chosen-ciphertext security. In *Proceedings of the 12th International Conference on Information Security*, pages 151–166. Springer-Verlag, 2009.

[137] Chul Sur, Chae Duk Jung, Youngho Park, and Kyung Hyune Rhee. Chosen-ciphertext secure certificateless proxy re-encryption. In *Communications and Multimedia Security*, pages 214–232. Springer, 2010.

[138] Xiaoqi Jia, Jun Shao, Jiwu Jing, and Peng Liu. CCA-secure type-based proxy re-encryption with invisible proxy. In *Computer and Information*

Technology (CIT), 2010 IEEE 10th International Conference on, pages 1299–1305. IEEE, 2010.

[139] Liming Fang, Willy Susilo, Chunpeng Ge, and Jiandong Wang. Interactive conditional proxy re-encryption with fine grain policy. *Journal of Systems and Software*, 84(12):2293–2302, 2011.

[140] Jun Shao, Guiyi Wei, Yun Ling, and Mande Xie. Identity-based conditional proxy re-encryption. In *Communications (ICC), 2011 IEEE International Conference on*, pages 1–5. IEEE, 2011.

[141] Liming Fang, Willy Susilo, Chunpeng Ge, and Jiandong Wang. Hierarchical conditional proxy re-encryption. *Computer Standards & Interfaces*, 34(4):380–389, 2012.

[142] Jun Shao. Anonymous ID-based proxy re-encryption. In *Information Security and Privacy*, pages 364–375. Springer, 2012.

[143] Victor Shoup and Rosario Gennaro. Securing threshold cryptosystems against chosen ciphertext attack. *Journal of Cryptology*, 15(2):75–96, 2002.

[144] Sébastien Canard, Julien Devigne, and Fabien Laguillaumie. Improving the security of an efficient unidirectional proxy re-encryption scheme. *Jounal of Internet Services and Information Security*, pages pp140–160, 2011.

[145] Ran Canetti, Hugo Krawczyk, and Jesper B Nielsen. Relaxing chosen-ciphertext security. In *Advances in Cryptology-CRYPTO 2003*, pages 565–582. Springer, 2003.

[146] Chris Peikert. Lattice cryptography for the internet. In *Post-Quantum Cryptography*, pages 197–219. Springer, 2014.

[147] Security guidance for critical areas of focus in cloud computing, version 3.0. Technical report, Cloud Security Alliance, 2011.

[148] Cisco global cloud networking survey. Technical report, Cisco, 2012.

[149] M. Casassa Mont, S. Pearson, and P. Bramhall. Towards accountable management of identity and privacy: Sticky policies and enforceable tracing services. In *Proc. 14th International Workshop on Database and Expert Systems Applications*, pages 377–382. IEEE, 2003.

[150] R. Dhamija and L. Dusseault. The seven flaws of identity management: Usability and security challenges. *Security & Privacy, IEEE*, 6(2):24–29, 2008.

[151] M. Hussain. *The Design and Applications of a Privacy-Preserving Identity and Trust-Management System*. PhD thesis, School of Computing, Queen's University, 2010.

[152] Shibboleth Consortium. Shibboleth. `http://shibboleth.net/`.

[153] OASIS Web Services Federation TC. Web Services Federation Language (WS-Federation) Version 1.2, 2009.

[154] OASIS Security Services TC. Glossary for the OASIS Security Assertion Markup Language (SAML) V2.0, 2005.

[155] E. Maler and D. Reed. The venn of identity: Options and issues in federated identity management. *Security & Privacy, IEEE*, 6(2):16–23, 2008.

[156] Microsoft. Windows Azure Active Directory. `http://www.windowsazure.com/en-us/home/features/identity/`.

[157] CA Technologies. CA CloudMinder Identity Management. `http://www.ca.com/us/cloudminder-identity-management`.

[158] S. Clauß and M. Köhntopp. Identity management and its support of multilateral security. *Computer Networks*, 37(2):205–219, 2001.

[159] S. Pearson and A. Benameur. Privacy, security and trust issues arising from cloud computing. In *2nd IEEE International Conference on Cloud Computing Technology and Science (CloudCom)*, pages 693–702. IEEE, 2010.

[160] Top threats to cloud computing, version 1.0. Technical report, Cloud Security Alliance, 2010.

[161] E.U. Comission. Council Directive 95/46/EC: On the protection of individuals with regard to the processing of personal data and on the free movement of such data, 1995.

[162] Scott Shane and John F. Burns. U.S. Subpoenas Twitter Over WikiLeaks Supporters. The New York Times, January 8, 2011.

[163] U.S. Congress. Uniting and strengthening america by providing appropriate tools required to intercept and obstruct terrorism act, 2001.

[164] J. Camenisch and E. Van Herreweghen. Design and implementation of the idemix anonymous credential system. In *Proceedings of the 9th ACM conference on Computer and communications security*, pages 21–30. ACM, 2002.

[165] OASIS Security Services TC. Metadata for the OASIS Security Assertion Markup Language (SAML) V2.0, 2005.

[166] W3C. XML Encryption Syntax and Processing Version 1.0. W3C Recommendation, W3C, 2002. http://www.w3.org/TR/xmlenc-core/.

[167] OASIS Security Services TC. Bindings for the OASIS Security Assertion Markup Language (SAML) V2.0, 2005.

[168] R. Shirey. Internet Security Glossary, Version 2. RFC 4949 (Informational), August 2007.

[169] David Nuñez, Isaac Agudo, and Javier Lopez. Integrating openid with proxy re-encryption to enhance privacy in cloud-based identity services. In *Proceedings of the 2012 IEEE 4th International Conference on Cloud Computing Technology and Science (CloudCom)*, pages 241 – 248, Taipei, Taiwan, Dec 2012 2012. IEEE Computer Society, IEEE Computer Society.

[170] P. Angin, B. Bhargava, R. Ranchal, N. Singh, L.B. Othmane, L. Lilien, and M. Linderman. An entity-centric approach for privacy and identity management in cloud computing. In *29th IEEE Symposium on Reliable Distributed Systems*, pages 177–183, 2010.

[171] Arkajit Dey and Stephen Weis. PseudoID: Enhancing privacy in federated login. In *Hot Topics in Privacy Enhancing Technologies*, pages 95–107, 2010.

[172] S. Chow, Y.J. He, L. Hui, and S. Yiu. SPICE–simple privacy-preserving identity-management for cloud environment. In *Applied Cryptography and Network Security*, pages 526–543. Springer, 2012.

[173] E. Bertino, F. Paci, R. Ferrini, and N. Shang. Privacy-preserving digital identity management for cloud computing. *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, 32(1):21–27, 2009.

[174] NIST. Big Data Working Group. http://bigdatawg.nist.gov/.

[175] Expanded Top Ten Big Data Security and Privacy Challenges. Technical report, Cloud Security Alliance, 2013.

[176] Sabrina De Capitani di Vimercati, Sara Foresti, and Pierangela Samarati. Managing and accessing data in the cloud: Privacy risks and approaches. In *Risk and Security of Internet and Systems (CRiSIS), 2012 7th International Conference on*, pages 1–9. IEEE, 2012.

[177] Devaraj Das, Owen O'Malley, Sanjay Radia, and Kan Zhang. Adding Security to Apache Hadoop. Technical Report 1, Hortonworks, 2011.

[178] Seonyoung Park and Youngseok Lee. Secure Hadoop with Encrypted HDFS. In *Grid and Pervasive Computing*, pages 134–141. Springer, 2013.

[179] Hsiao-Ying Lin, Shiuan-Tzuo Shen, Wen-Guey Tzeng, and B.-S.P. Lin. Toward data confidentiality via integrating hybrid encryption schemes and Hadoop distributed file system. In *Advanced Information Networking and Applications (AINA), IEEE 26th International Conference on*, pages 740–747, 2012.

[180] Apache Accumulo. http://accumulo.apache.org/.

[181] Jia Liu, Mark D Ryan, and Liqun Chen. Balancing societal security and individual privacy: Accountable escrow system. In *Computer Security Foundations Symposium (CSF), 2014 IEEE 27th*, pages 427–440. IEEE, 2014.

[182] Geoffrey A. Fowler, Devlin Barrett, and Sam Schechner. U.S. shuts offshore file-share 'locker'. The Wall Street Journal, January 20, 2012.

[183] Certivox. PrivateSky. `http://privatesky.me/`.

[184] CipherCloud. CipherCloud Gateway. `http://www.ciphercloud.com/`.

[185] Masaaki Miki, Eiji Hayashi, and Hideki Shingai. Highly reliable and highly secure online storage platform supporting "timeon" regza cloud service. *Toshiba Review*, 68(5):25–27, 2013. In Japanese.

[186] M. Shimano, G. Fujino, M. Miki, Y. Tsuzuki, I. TAKEYASU, and E. TOKITA. Key change management apparatus and key change management method, April 3 2014. US Patent App. 13/928,112.

[187] L. Xu and X. Wu. Proxy-based encryption method, proxy-based decryption method, network equipment, network device and system, October 28 2014. US Patent 8,873,754.

[188] G. Zhang. Data protection method and data protection system, September 17 2013. US Patent 8,539,606.

[189] D. BISWAS. Methods and apparatus for sharing real-time user context information, February 4 2014. US Patent 8,645,682.

[190] P. Paillier and A. Gouget. Data providing process based on an ibpe scheme, February 4 2014. US Patent 8,644,509.

[191] A.J. Farrugia, N. Sullivan, G. Fasoli, and M. Ciet. Encryption method and apparatus using composition of ciphers, March 25 2014. US Patent 8,681,975.

[192] S.R. Hohenberger, K. Fu, G. Ateniese, and M. Green. Unidirectional proxy re-encryption, January 10 2012. US Patent 8,094,810.

[193] System for cross-domain identity management. `http://www.simplecloud.info/`.